Nonstationary Color Tracking for Vision-Based Human–Computer Interaction

Ying Wu, *Member, IEEE,* and Thomas S. Huang, *Life Fellow, IEEE*

***Abstract*—Skin color offers a strong cue for efficient localization and tracking of human body parts in video sequences for vision-based human–computer interaction. Color-based target localization could be achieved by analyzing segmented skin color regions. However, one of the challenges of color-based target tracking is that color distributions would change in different lighting conditions such that fixed color models would be inadequate to capture nonstationary color distributions over time. Meanwhile, using a fixed skin color model trained by the data of a specific person would probably not work well for other people. Although some work has been done on adaptive color models, this problem still needs further studies. This paper presents our investigation of color-based image segmentation and nonstationary color-based target tracking, by studying two different representations for color distributions. In this paper, we propose the structure adaptive self-organizing map (SASOM) neural network that serves as a new color model. Our experiments show that such a representation is powerful for efficient image segmentation. Then, we formulate the nonstationary color tracking problem as a *model transduction* problem, the solution of which offers a way to adapt and transduce color classifiers in nonstationary color distributions. To fulfill model transduction, this paper proposes two algorithms, the *SASOM transduction* and the *discriminant expectation–maximazation (EM)*, based on the SASOM color model and the Gaussian mixture color model, respectively. Our extensive experiments on the task of real-time face/hand localization show that these two algorithms can successfully handle some difficulties in nonstationary color tracking. We also implemented a real-time face/hand localization system based on such algorithms for vision-based human–computer interaction.**

***Index Terms*—Color-based image segmentation, color model, discriminant analysis, expectation–maximization (EM), nonstationary color tracking, structure adaptive self-organizing map (SASOM), vision-based human–computer interaction.**

I. Introduction

In current virtual environment (VE) applications, keyboards, mice, wands, and joysticks are the most popular input devices. However, those devices are either inconvenient or unnatural when providing three-dimensional (3-D) or high degrees of freedom (DOF) inputs. To achieve immersive human–computer interaction, human body parts, e.g., the hand, could be considered as a natural input "device," which motivates the research of tracking, analyzing, and recognizing human body movements [25], [31], [32]. An application example is the gesture interfaces for virtual environments, in which a set of hand gestures could be use to represent some commanding inputs such as pointing, rotating, starting, stopping, etc. Although the goal of such immersive interfaces is to recognize and understand the human body movements, the first step to achieve this goal is to reliably localize and track such human body parts as the face and the hand. Magnetic sensors have been used to fulfill these tasks. However, many magnetic sensors are plagued by magnetic interferences [32]. Alternatively, we could consider other techniques that are based on noninvasive visual sensors, by which the motion of the target could be inferred by analyzing video inputs. We usually call the interaction based on visual sensory inputs vision-based interaction (VBI).

In most VBI, localizing and tracking targets in video sequences provide inputs to the steps of target recognition and action recognition. Visual localization and tracking are confronted by the difficulties of complex backgrounds, unknown lighting conditions and complex target movements. When we need to analyze multiple targets simultaneously, the problem becomes even more challenging since different targets would induce occlusion. The robustness, accuracy, and speed are important to evaluate tracking algorithms.

Different image features provide different cues for tracking algorithms. Edge-based approaches match image edges in images, and region-based approaches match image templates and image regions. Under the small motion assumption that assumes there is little motion difference between two consecutive image frames, these approaches could achieve reasonable results. However, when this assumption does not hold, which could be very likely to happen in practice, these tracking algorithms would probably lose track, and the recovery of tracking would depend on other remedies. In addition, these methods usually involve manual initialization.

An alternative is the blob-based approach, which does not use local image information such as edges and regions. Instead, it represents the target by its color and motion such that the localization and tracking can be fulfilled by segmenting the target out from the images. For example, when we need to localize the hand in video sequences, it would be very difficult to represent the hand based only on edges or image appearances due to the highly articulated finger movements. In addition, there are very large variations in hand appearances from different view directions. On the other hand, when we notice the uniqueness of the flesh-tone, color-based segmentation approaches could

Manuscript received April 15, 2001; revised October 30, 2001. This work was supported by Northwestern Faculty Startup Funds, the National Science Foundation under Grants CDA-96-24396 and EIA-99-75019, and the NSF Alliance Program.

Y. Wu is with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: yingwu@ece.northwestern.edu).

T. S. Huang is with the Beckman Institute, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (e-mail: huang@ifp.uiuc.edu).

Publisher Item Identifier S 1045-9227(02)04425-9.

afford efficient and robust visual localization. Certainly, combining the above two approaches by integrating multiple visual cues would result in more robust tracking systems [15], [33].

Meanwhile, efficient segmentation is also desirable for tracking bootstrapping and reinitialization. Recently, some successful tracking systems have been built based on skin color [7], [14], [18], [29], [34]. A simple approach is to collect skin color pixel samples from the target and to train a color classifier, such that skin color regions could be segmented by classifying and grouping input color pixels. To alleviate the difficulty of the large variation in flesh-tone among different people, one of the solutions is to tune the color classifier through a huge training data set collected from many people [17]. Unfortunately, in practice, there are still some complications. One of them is that color distributions may change with lighting conditions. As a result, a fixed skin color model may not work well all the time. Another difficulty is that collecting such a large labeled training data set is not trivial at all. An effective approach to this problem is to adapt color models to different lighting conditions and different people. Color representations and color model adaptation schemes are the two important issues to study. We will discuss these issues in more details in Sections II–VII.

In this paper, we will study two different representations for color distributions, the structural adaptive self-organizing map (SASOM) model and the Gaussian mixture model, for the tasks of adaptive color-based segmentation and nonstationary color tracking. In Section II, we will describe different color spaces and models. In Section III, we will present a novel SASOM neural network, which will be employed as a new color model. An interesting aspect of such SASOM neural network is that its structure could be learned through training. An analysis of the stationary status of SOM neural network will be also given in that section. Section IV will give a formulation of the color tracking problem. Based on the SASOM color model and the Gaussian mixture color model, Section V will present two algorithms, the SASOM transduction algorithm and the discriminant expectation–maximazation (EM) or (D-EM) algorithm. Different from the methods of constructing a specific skin color model, our proposed approach tries to adapt the models to nonstationary color distributions by transducing a learned color model through image sequences. Section VI will report some of our experiments on the proposed SASOM color model, and the two different color tracking algorithms. We will conclude the paper in Section VII by summarizing the paper and giving some thoughts about future studies.

II. Representations of Color Distributions

A. Color Spaces

Digital color images consist of color pixels, each of which is associated with a color feature vector. Different color spaces, such as the hue, saturation, and value (HSV) space, the cyan, magenta, and yellow (CMY) space, the red, green, and blue (RGB) space, and the normalized-RGB space, have been used in current research. Many color histogram-based techniques use two-dimensional (2-D) subspaces of these 3-D color spaces, partly because of the demanding requirements of computational resources for 3-D histograms. For example, the HSV space could be reduced to its HS subspace by ignoring the V components. However, hue and saturation components become unstable when the intensity of a pixel becomes too large or too small, which means that the H and S components would be meaningless for dark or bright pixels. As a result, a simple intensity thresholding method could segment bright objects from a dark background very well, while a color-based segmentation method using the HS subspace would probably fail. Therefore, simply reducing the dimensionality of a 3-D color space to a 2-D subspace might lose valuable color information.

Although these 3-D color spaces have substantial physical meanings, it seems that none of them is able to give satisfactory color invariants through different lighting conditions. The issue of selecting good color features for the target in color-based segmentation should be addressed. Considering that the HSV color space is not a linear transformation of the RGB space, we may want to use a higher dimensional color space such as six–dimensional (6-D) by compounding the HSV with the RGB components. Since this higher dimensional color space is redundant, a linear subspace could be found by performing some dimension reduction techniques such as the principal component analysis (PCA) technique and the multiple discriminant analysis (MDA) technique, which will be described in detail in Section V-B. By this means, good color features for color pixel classification could be selected automatically.

B. Representations of Color Distributions

Skin color offers an effective and efficient way to localize and track hands and faces in vision-based human-computer interaction. The core of color-based tracking is color-based segmentation. According to the representations of color distribution, current color-based tracking approaches can be classified into two general categories: nonparametric [17], [18], [28], [34] and parametric [26], [30], [35].

One of the nonparametric approaches is based on color histograms [17], [18], [28]. Since a color space is quantized by the structure of a histogram, this technique is confronted by the same difficulty as the nonparametric density estimation task, in which the level of quantization will affect the estimation. Generally, nonparametric approaches work effectively when the quantization level could be properly set and there are sufficient data available. However, how to select a good quantization level for color histograms is not trivial. Although nonuniform quantization schemes would perform better than uniform quantization, they are much more complicated. An alternative nonparametric approach proposed in this paper is based on the SOM, an unsupervised clustering algorithm to approximate color distributions. The details will be presented in later sections. SOM can be viewed as a neural-network-based vector quantization (VQ) algorithm. Instead of specifying the structure of SOM, the proposed SASOM algorithm has the ability to find an appropriate structure by embedded schemes of growing, pruning, and merging.

Parametric approaches model color densities in parametric forms such as the Gaussian model or the Gaussian mixture model [26], [30], [35]. Let $\mathbf{x}$ be the color feature vector for each pixel. The color distribution of an image can be represented by a mixture density

$$p(\mathbf{x}|\mathbf{\Theta}) = \sum_{j=1}^{C} p\left(\mathbf{x}|O_j;\theta_j\right) p(O_j) \tag{1}$$

where $\sum_{j=1}^{C} p(O_j) = 1$ and where $p\left(\mathbf{x}|O_j;\theta_j\right)$ is the conditional density for a pixel belonging to an object O_j in the image, and it has been parameterized by θ_j, and $\mathbf{\Theta} = \{\theta_j, j = 1, \ldots, C\}$. This conditional density can also be modeled by Gaussian mixtures

$$p\left(\mathbf{x}|O_j;\theta_j\right) = \sum_{k=1}^{T} p(\mathbf{x}|c_k;\theta_{jk}) p(c_k) \tag{2}$$

where $\sum_{k=1}^{T} p(c_k) = 1$ and where $p(\mathbf{x}|c_k;\theta_{jk})$ is the conditional density for a pixel belonging to a color component c_k of the object O_j in the image. Each mixture component can be modeled by a Gaussian model with mean μ_k and covariance matrix $\mathbf{\Sigma}_k$. EM offers a way to fit probabilistic models to the observation data. The difficulty of *model order selection* could be handled by heuristics [26], cross-validation, or model selection.

C. Color-Based Segmentation

Color is a strong cue for image segmentation [5]. Both parametric and nonparametric approaches have been studied for segmentation. Histogram-based segmentation approaches such as color predicate (CP) [18] work well when appropriately thresholding the histogram. However, there are no obvious ways to find correct thresholds. Parametric approaches make use of parametric color models based on the Gaussian model or the Gaussian mixture model [17], [26]. A difficulty is that there would not be enough prior knowledge to determine the number of components of the mixture density in advance.

Since the computational resources needed in color histogramming techniques generally grows with respect to the dimensionality of the color space, it seems that a less computationally expensive scheme should be found to handle the quantization of the color space. In this paper, we propose an SASOM neural network to approximate color distributions, and segmentation is achieved by the competition among the neurons in the SASOM. The details will be presented in Section III.

D. Color Distributions Under Nonstationary Illumination

It is straightforward that similar color pixels in an image could be grouped together to facilitate the separation of the foreground target from the background environment. However, when we look into the flesh-tone distributions in video inputs in some VE applications, the segmentation problem is confronted by such challenges as large variation in skin tone, unknown lighting conditions, and dynamic scenes. In order to achieve user-independence, segmentation-based tracking algorithms should be able to deal with the large variation in skin color for different people. One possible solution is to make a generic statistical model of skin color by collecting a huge training data set [17] so that the generic color model could work for every user.

Even though such a good generic color model can be obtained, we have to face another difficulty in color tracking: generic color models would be incapable to handle changing lighting conditions unless some invariants could be found. Many color tracking techniques assume controlled lighting conditions. However, in many cases, the target may be shadowed by other objects or by the target itself so that the color looks very different. What is more, we cannot assume constant lighting sources, since the lighting directions, intensities, and tones might also change. In some VE applications, since the graphics rendered on the display keeps changing, the reflected lights would change the apparent color of the target. This color constancy problem is not trivial in color tracking for vision-based interaction.

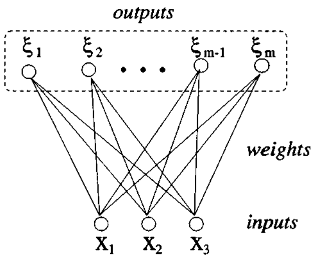


Fig. 1. 1-D SOM structure.

Because of dynamic scenes and changing lighting conditions, color distributions over time are generally nonstationary, since the statistics of color distributions might change with time. If a color classifier is trained under a specific condition, it may not work well in other scenarios.

Some researchers have looked into the nonstationary color distribution problem in color tracking [26], [29], [33]. A scheme of color model adaptation was addressed in [26], in which a Gaussian mixture model was used to represent color distribution, and a linear extrapolation scheme was employed to adjust the parameters of the model by a set of labeled training data drawn from a new image frame. However, since the new image is not segmented, this labeled data set is not reliable. Other color adaptive methods take advantage of other visual cues as external tracking priors [29], [33].

In this paper, to approach this nonstationary color tracking problem, two schemes will be presented. One is the *SASOM transduction*, which updates the weights and structure of an SASOM to capture the new color distribution based on both labeled and unlabeled color pixel samples. Another scheme is called the D-EM algorithm, which approaches such nonstationary adaptation problem in an EM framework. The advantage of these two schemes is that they do not require a large number of labeled training color pixels.

III. Segmentation Based on SASOM

SOM [6], [19], [20] could be used to visualize and interpret large high-dimensional data set by mapping them to a low-dimensional space based on a competitive learning scheme. SOM consists of an input layer and an output layer. Fig. 1 shows the structure of one-dimensional (1-D) SOM.

The number of nodes in the input layer is the same as the dimensionality of input vectors. The structure of the output layer can be 1-D or 2-D connected neurons that are linked to every input node. A weight vector is associated with each link. Through competition among the neurons[1] in the output layer, the index of the winner neuron is taken as the output of SOM to the input vector. The Hebbian learning rule adjusts the weights of the winner neuron and its neighborhood neurons in training. SOM is highly related to VQ [1], [13] and the k-mean clustering technique. An interesting characteristics of SOM is its property of partial data density preservation.

A. An Analysis of the Stationary Status of SOM

One of the problems of many clustering algorithms is that the number of clusters should be specified in advance. The performance of clustering algorithms depends on the number of clusters. It is the same case in the standard SOM algorithm, which is also confronted by the structure learning difficulty. In SOM, when the way of linking of neurons is fixed, the variable left for representing structure of SOM will be the number of neurons. Different SOM structures, e.g., different number of neurons, will lead to different tessellations of the data space. If fewer neurons are used, inputs from lower density regions will be dominated by those from higher density regions in data space. On the other hand, if more neurons are used, SOM training is unlikely to form an ordered mapping, since the training will probably get trapped in one of the local minima.

Many researchers have investigated the structural level learning of neural networks [2], [4], [11], [12], [16], [21]. A straightforward approach is to validate a set of neural networks with different structures. Since the structural level adaptation implies an optimization in a continuous function space, such validating scheme can only test a very small set of hypotheses. Alternatively, people also looked into evolutionary schemes to find optimal structures for neural networks [2], [16]. Although such evolutionary optimization has nice global properties, they are generally slow and computationally intensive. A different methodology to approach this problem is to parameterize the structure such that the structural level adaptation could be fulfilled through optimizing the structure parameters [21]. On the other hand, many researchers have been looking into the approaches of dynamically adjust the structure in training [4], [11], [12]. For example, the approach of growing cell structure was proposed in [11], in which the numbers and the linking of neurons could be adjusted during training.

Meanwhile, there have been many studies on the property of density preservation of SOM [9], [10], [22]–[24], [27]. An interesting result regarding to such property is that the neuron density is proportional to $p(\mathbf{x})^{2/3}$, where $p(\mathbf{x})$ is the probability distribution of the inputs. Such conclusion was reached by assuming a fixed structure of SOM. However, when directly applying SOM to some vector quantization tasks, e.g., the color segmentation task that needs a quantization of color spaces, better results could be achieved if the neuron density is proportional to $p(\mathbf{x})$ instead of $p(\mathbf{x})^{2/3}$.

Some studies observed that when every neuron has equal wining probability for the entire data set in the stationary status of SOM, the neuron density will be proportional to $p(\mathbf{x})$[11], [36]. A structural adaptation was employed in [11], while a different weight adjusting scheme was used in [36]. It seems that there exists two extremes: 1) when every neurons has equal wining probability, the neuron density will be proportional to $p(\mathbf{x})$, i.e., more neurons will be allocated to represent higher density regions in the data space and 2) when the neuron density is uniform, the wining probabilities of neurons will be proportional to $p(\mathbf{x})$, i.e., neurons that represent higher density regions of the data space will have higher wining probabilities. It seems that the standard SOM with fixed structure and with the Hebbian learning rule falls in between of these two extremes, i.e., the neuron density is proportional to $p(\mathbf{x})^{2/3}$.

Let $\mathbf{x}$ denote a data point in the data space, and a set of samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ are drawn from the probability distribution function $p(\mathbf{x})$. And let $\mathbf{w}$ denote the weight vector for each neuron of SOM, which represents a point in the weight space. Also we use ξ to represent the position of each neuron in the weight space. Given a data point $\mathbf{x}$, there is a winner neuron, whose position will be denoted by ξ^*, which is a function of $\mathbf{x}$, i.e., $\xi^*(\mathbf{x})$. Meanwhile, the neighborhood function in SOM could be denoted by $\Lambda(\xi, \xi^*)$. Generally, $\Lambda(\xi, \xi^*) = \Lambda(\xi - \xi^*(\mathbf{x}))$ holds, which means that $\Lambda(\xi, \xi^*)$ is related only to the distance $\xi - \xi^*(\mathbf{x})$. Also, we let $0 < \Lambda(\xi, \xi^*) < 1$ and $\Lambda(\xi^*) = 1$.

When we consider $\Lambda(\xi, \xi^*(\mathbf{x}))$ as the conditional probability of selecting a winner neuron of weight $\mathbf{w}$ and position ξ, given an input data sample $\mathbf{x}$, we have

$$P_{\text{win}}(\xi|\mathbf{x}) \propto \Lambda(\xi, \xi^*(\mathbf{x})) \tag{3}$$

which means that the wining probability of a certain neuron given a specific input is determined by the weight of such neuron and the weight of the actual winner neuron. Consequently, we could write the wining probability of a certain neuron under the entire training set

$$P_{\text{win}}(\xi) \propto \sum_{k=1}^{N} \Lambda(\xi - \xi^*(\mathbf{x}))p(\mathbf{x}). \tag{4}$$

We could write it in a continuous form as

$$P_{\text{win}}(\xi) = \frac{1}{Z}\int_{\Omega} \Lambda(\xi - \xi^*(\mathbf{x}))p(\mathbf{x})d\mathbf{x} \tag{5}$$

where Z is a normalization factor. If every neuron has the same wining probability, i.e.,

$$P_{\text{win}}(\xi) = \text{Const.} \tag{6}$$

We could write

$$\frac{\partial P_{\text{win}}(\xi)}{\partial \xi} = \frac{1}{Z}\int_{\Omega} \Lambda'(\xi - \xi^*(\mathbf{x}))p(\mathbf{x})d\mathbf{x} = 0. \tag{7}$$

Let $\epsilon = \xi^*(\mathbf{x}) - \xi$, which means the distance between a certain neuron and the winner neuron in terms of position. Since ξ^* reflects all such input data points that have ξ^* as their winner neuron, so we could write $\mathbf{x} = \mathbf{w}(\xi + \epsilon)$. Furthermore, we could write the expansion of $p(\mathbf{x})$ in terms of $\xi^*(\mathbf{x}) - \xi$

$$p(\mathbf{x}) = p(\mathbf{w}(\xi + \epsilon)) \approx p(\mathbf{w}) + \epsilon p'(\mathbf{w})\mathbf{w}'(\xi). \tag{8}$$

Meanwhile, we have

$$\begin{aligned} d\mathbf{x} =& d\mathbf{w}(\xi + \epsilon) = \mathbf{w}'(\xi + \epsilon)d\epsilon \\ \approx& (\mathbf{w}' + \epsilon\mathbf{w}'')d\epsilon \end{aligned} \tag{9}$$

$$\Lambda'(\xi - \xi^*(\mathbf{x})) = \Lambda'(-\epsilon) = -\Lambda'(\epsilon). \tag{10}$$

[1] We use “neurons” for short of “output neurons.”

Plugging (8)–(10) into (7), we have

$$\int_{\Omega} \left[p(\mathbf{w}) + \epsilon p'(\mathbf{w})\mathbf{w}'\right]\left[\mathbf{w}' + \epsilon \mathbf{w}''\right] \Lambda'(\epsilon) d\epsilon = 0. \tag{11}$$

Since $\Lambda'(\epsilon)$ is an odd function, i.e., $\Lambda'(-\epsilon) = -\Lambda'(\epsilon)$, we will have

$$\int_{\Omega} \left[p(\mathbf{w})\mathbf{w}'' + p'(\mathbf{w})(\mathbf{w}')^2\right] \epsilon \Lambda'(\epsilon) d\epsilon = 0. \tag{12}$$

Furthermore, we obtain

$$p(\mathbf{w})\mathbf{w}'' + p'(\mathbf{w})(\mathbf{w}')^2 = 0 \tag{13}$$

i.e.,

$$\frac{\mathbf{w}''}{\mathbf{w}'} = -\frac{p'(\mathbf{w})\mathbf{w}'}{p(\mathbf{w})}.$$

Obviously, we can write

$$\frac{d}{d\xi} \log \mathbf{w}' = -\frac{d}{d\xi} \log p(\mathbf{w}).$$

So, we obtain

$$\frac{d\mathbf{w}}{d\xi} = \mathbf{w}' = \frac{1}{p(\mathbf{w})}. \tag{14}$$

When the training of SOM reaches its stationary status, the weight space will approximate the data space. We could write

$$\frac{d\xi}{d\mathbf{w}} \propto p(\mathbf{x}) \tag{15}$$

if $P_{\text{win}}(\xi) = \text{Const}$ holds in the stationary status.

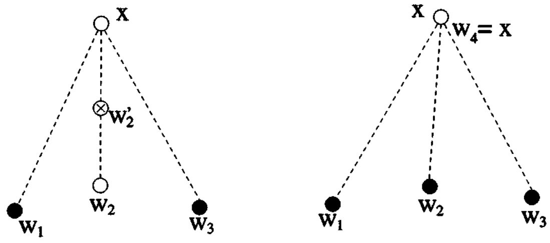
B. An SASOM

There would be two problems if we apply the standard SOM directly for color segmentation: 1) how can we determine the structure of the SOM? 2) and does the standard SOM really capture the color distributions? Unfortunately, there are no general ways of determining the structure of SOM. In addition, the standard SOM will not accurately capture the data distribution as described in Section III-A.

One possible approach to the structure determination could be cross-validation. Although the structure of the SOM, such as the number of neurons, is fixed each time, a good structure could be determined after validating several different structures. However, this approach does not offer flexibility in training, and it is not efficient. An alternative approach embeds some heuristics of changing the structure dynamically in training. Our algorithm, the SASOM, can automatically find an appropriate number of neurons based on a set of heuristics such as growing, pruning and merging. With such set of heuristics, when the SASOM reach its stationary status, $P_{\text{win}}(\xi) = \text{Const}$ will hold, such that the neuron density will be proportional to the data distribution as the analysis in Section III-A. We present the growing, pruning and merging schemes below.

Growing Scheme: In the standard SOM training algorithm, the response of a neuron is the distance between the input vector and the weight vector of the neuron. The distance measurement can be defined as

$$\mathcal{D}(\mathbf{x}, \mathbf{w}_i) = ||\mathbf{x} - \mathbf{w}_i|| \tag{16}$$



(a) (b)

Fig. 2. Growing scheme of SASOM. $\mathbf{w}_i$ is the weight vector, and $\mathbf{x}$ is an input vector. (a) When the input vector is too far from all weight vectors so that the responses of all neurons are nearly the same, if current input $\mathbf{x}$ is in the data cluster represented by one of these neurons, say $\mathbf{w}_2$, the weight vector of that neuron will be misplaced unnecessarily to $\mathbf{w}'_2$ in SOM training. (b) In this situation, a new neuron is created and its weight will be set by $\mathbf{w}_4 = x$.

where $\mathcal{D}$ is a distance measurement between the input vector $\mathbf{x}$ and the weight vector $\mathbf{w}_i$ of the ith neuron of SOM. We call it the response value of the neuron. The measurement here is the Euclidean distance, however, other distance measurements could also be employed.

In standard SOM, the neuron with the smallest response is taken as the winner c

$$c = \arg\min_i \mathcal{D}(\mathbf{x}, \mathbf{w}_i). \tag{17}$$

In some cases, however, when the responses of all neurons are nearly the same, determining the winner by finding the one with the smallest response is not suitable. In this situation, the input vector may be too far from all weight vectors or may be around the center of the convex hull of the weight vectors. In this case, the input data point $\mathbf{x}$ may be drawn from any or none of the data clusters represented by these neurons $\mathbf{w}_i$. As a result in training, the weight vector of the selected neuron could be misplaced unnecessarily by adjusting its weight. So, it is not a robust way to take the neuron of the smallest response as the winner. In this situation, a new neuron could be generated, and be inserted it to the current structure by taking the input vector as its initial weight, which is illustrated in Fig. 2.

By comparing the mean value and the median value of the response values of all neurons, we make a rule to detect this situation, in which if or not a new neuron should be created. The competition can be described as

$$v_i = \mathcal{D}(\mathbf{x}, \mathbf{w}_i) \quad \forall\, i \in \{1, \ldots, M\} \tag{18}$$

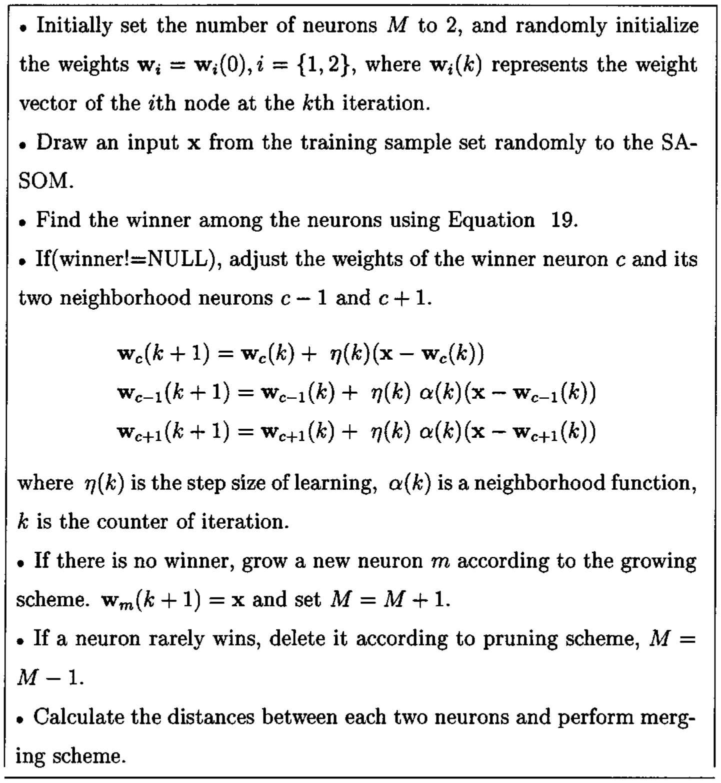
where v_i is the response of the ith neuron with weight vector $\mathbf{w}_i$, and M is the number of neurons. The winner can be selected by

$$c = \begin{cases} \text{NULL}, & \text{if } \texttt{mean}(\mathbf{v}) \approx \texttt{median}(\mathbf{v}) \\ \arg\min_i v_i, & \text{otherwise} \end{cases} \tag{19}$$

where $\mathbf{v} = \{v_1, v_2, \ldots, v_M\}$ and M is the number of neurons.

Pruning Scheme: In the training process, when a neuron is rarely to be a winner, it means that the data cluster represented by this neuron is of very low density or might be taken as noise. As a result, such a neuron can be pruned. In practice, a threshold is set to determine these neurons.

Merging Scheme: In the training process, the distance between two weight vectors of each two neurons are calculated. If two weight vectors are close enough, we can merge these two neurons by assigning the average of the two weights to a new neuron and deleting these two neurons.



- Initially set the number of neurons M to 2, and randomly initialize the weights $\mathbf{w}_i = \mathbf{w}_i(0), i = \{1, 2\}$, where $\mathbf{w}_i(k)$ represents the weight vector of the ith node at the kth iteration.
- Draw an input $\mathbf{x}$ from the training sample set randomly to the SA-SOM.
- Find the winner among the neurons using Equation 19.
- If(winner!=NULL), adjust the weights of the winner neuron c and its two neighborhood neurons $c - 1$ and $c + 1$.

$$\begin{aligned}
\mathbf{w}_c(k+1) &= \mathbf{w}_c(k) + \eta(k)(\mathbf{x} - \mathbf{w}_c(k)) \\
\mathbf{w}_{c-1}(k+1) &= \mathbf{w}_{c-1}(k) + \eta(k)\, \alpha(k)(\mathbf{x} - \mathbf{w}_{c-1}(k)) \\
\mathbf{w}_{c+1}(k+1) &= \mathbf{w}_{c+1}(k) + \eta(k)\, \alpha(k)(\mathbf{x} - \mathbf{w}_{c+1}(k))
\end{aligned}$$

where $\eta(k)$ is the step size of learning, $\alpha(k)$ is a neighborhood function, k is the counter of iteration.

- If there is no winner, grow a new neuron m according to the growing scheme. $\mathbf{w}_m(k+1) = \mathbf{x}$ and set $M = M + 1$.
- If a neuron rarely wins, delete it according to pruning scheme, $M = M - 1$.
- Calculate the distances between each two neurons and perform merging scheme.

Fig. 3. The training algorithm of the SASOM.

The algorithm of SASOM is summarized in Fig. 3.

We performed color-based image segmentation based on such an SASOM model. In our segmentation algorithm, training data set is collected from one color image, and each data vector is weighted HSI vector, i.e., $\mathbf{x} = \{\alpha H, \beta S, \gamma I\}$, where we set $\alpha = \beta = 1$ and $\gamma = 0.1$. Pixels with large and small intensities are not included in the training data set, because hue and saturation become unstable in this range. Once trained, the SASOM is used to label each pixel by its HSI value. The neuron indexes are used as labels for color image pixels. Some experiment results will be presented in Section VI-A.

IV. The Nonstationary Color Tracking Problem

It is a good practice to learn a generic color classifier for color-based segmentation by collecting a large labeled data set [17]. If some color invariants could be found, learning such a color classifier would suggest a direct and robust way to color tracking. However, when we consider the nonstationary color distribution over time, we do not generally expect to find such invariants.

The approach taken in [17] is an *inductive learning* approach, by which the learned color classifier should be able to classify any pixel in any image. Generally, this color classifier would be highly nonlinear, and a huge labeled training data set is required to achieve good generalization. In fact, learning such a highly nonlinear color classifier for all lighting conditions and all images may not be necessary, because the requirement of the generalization could be relaxed to a subset of the data space. This is the exact case in color tracking. As interesting thing is that a color classifier M_t at time frame t will be only used to classify pixel $\mathbf{x}_j$ in the current specific image I_t. We may not care how M_t works for other images. So we may expect that M_t could be simpler than a general purpose classifier. When a new image I_{t+1} at time $t+1$ comes in, this specific classifier M_t should be *transduced* to a new classifier M_{t+1} which works just for the new unsegmented image I_{t+1} instead of I_t. The classification can be described as

$$y_i = \arg \max_{j=1,\ldots,C} p\left(y_j | \mathbf{x}_i, M_t, I_{t+1} : \forall\, \mathbf{x}_i \in I_{t+1}\right) \tag{20}$$

where y_i is the label of $\mathbf{x}_i$, and C is the number of classes. In this sense, we do not care the performance of the classifier M_{t+1} for the data outside I_{t+1}. We call the problem of transducing the classifier M_t to M_{t+1} given unlabeled I_{t+1} *model transduction*. Fig. 4 shows the transduction of color classifiers.

This model transduction may not always be feasible unless we know the joint distribution of I_t and I_{t+1}. Unfortunately, such joint probability is generally unknown since we may not have enough *a priori* knowledge about the transition in a color space over time. One approach is to assume a transition model, like the case in motion tracking by Kalman filter or condensation [3], so that we can explicitly model $p(I_{t+1}|I_t)$. One of the difficulties of this approach is that a fixed transition model is unable to capture complex dynamics. Although the issue of motion model switching by learning transition models has been addressed in [3], their scheme is not general. Another difficulty is that it may not be easy to identify parameters of the transition models due to the insufficient labeled training data. The approach used in [26] assumes a linear transition model. However, the transition (updating) of color models is plagued since the current image has not been segmented yet.

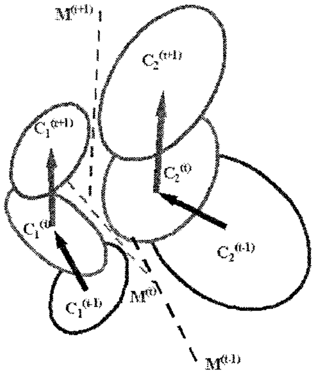


Fig. 4. An illustration of transduction of classifiers.

However, our assumption is different from the transition model assumption. We assume that the classifier M_t at time t can give "confident" labels to several samples in I_{t+1}, so that the data in I_{t+1} can be divided into two parts: labeled data set $\mathcal{L} = \{(\mathbf{x}_j, y_j), j = 1, \ldots, N\}$, and unlabeled set $\mathcal{U} = \{\mathbf{x}_j, j = 1, \ldots, M\}$, where N and M are the size of the labeled set and unlabeled set, respectively, $\mathbf{x}_j$ is the color feature vector, and y_j is its label (such as skin or nonskin). Here, $\mathcal{L}$ and $\mathcal{U}$ are from the same distribution. Consequently, the transductive classification can be written as

$$y_i = \arg \max_{j=1,\ldots,C} p\left(y_j | \mathbf{x}_i, \mathcal{L}, \mathcal{U} : \forall\, \mathbf{x}_i \in \mathcal{U}\right). \qquad (21)$$

In this formulation, the specific classifier M_t is transduced to another classifier M_{t+1} by combining a large unlabeled data set from I_{t+1}.

V. Transductive Color Tracking

One of the problems of tracking by color-based segmentation is that the unknown lighting conditions may change the color of the target. Even in the case of fixed lighting sources, the target color may be still different over a video sequence, since the target might be shadowed by other objects. These situations confront the approaches that make use of a fixed skin color model, since the distribution of skin color is nonstationary through image sequences. This section presents two different model transduction approaches to the nonstationary color tacking problem. One of them is based on the SASOM model, and the other is based on the Gaussian mixture model.

A. Model Transduction Based on SASOM

The color distribution of each image frame could be modeled by an SASOM, in which each neuron represents a color cluster for the image at current time frame. Such SASOM also offers a simple color classifier by neuron competition, through which the image can be segmented. However, this classifier may not be good for the next image frame because of the nonstationary nature of color distributions. A new SASOM is needed for the new image frame.

Our solution to this problem is called *SASOM transduction*, which is to update the weights and structure of the trained SASOM according to a set of new training data so that the transduced SASOM captures the color distribution of the new image. The new training data set for transduction consists of both labeled and unlabeled samples. The algorithm is described below.

- $\mathcal{W}^{(n-1)} = \left\{\mathbf{w}_i^{(n-1)}, i = 1, \ldots, C^{(n-1)}\right\}$ are the weights of SASOM at time frame $n-1$. The training data set $\mathcal{X}^{(n)} = \left\{\mathbf{x}_k^{(n)}, k = 1, \ldots, N\right\}$ is drawn randomly from the image at time frame n. We use $\mathcal{W}^{(n)}$ to represent SASOM at time frame n.
- The training data set $\mathcal{X}^{(n)}$ is classified by the SASOM $\mathcal{W}^{(n-1)}$, and is partitioned into two parts: a labeled data set $\mathcal{X}_l^{(n)}$ and an unlabeled data set $\mathcal{X}_u^{(n)}$. If a sample $\mathbf{x}_k^{(n)}$ is confidently classified by $\mathcal{W}^{(n-1)}$, then put this sample to the set $\mathcal{X}_l^{(n)}$ and label it with the index of the winner neuron of $\mathcal{W}^{(n-1)}$; otherwise, put it to $\mathcal{X}_u^{(n)}$ and let it unlabeled.
- Unsupervised updating: The algorithm described in Section III is employed to update $\mathcal{W}^{(n-1)}$ by the unlabeled data set $\mathcal{X}_u^{(n)}$.
- Supervised updating: The labeled data set $\mathcal{X}_l^{(n)}$ is used in this step. $(\mathbf{x}_k, l_k)$ is drawn from $\mathcal{X}_l^{(n)}$, where l_k is the label for $\mathbf{x}_k$. The winner neuron for the input $\mathbf{x}_k$ is c.

$$\mathbf{w}_c^{(n)} = \begin{cases} \mathbf{w}_c^{(n-1)} + \alpha\left(\mathbf{x}_k - \mathbf{w}_c^{(n-1)}\right), & \text{if } c = l_k; \\ \mathbf{w}_c^{(n-1)} - \alpha\left(\mathbf{x}_k - \mathbf{w}_c^{(n-1)}\right), & \text{if } c \neq l_k;. \end{cases}$$

After several iterations, the SASOM at time frame $n-1$ is transduced to n.

B. Model Transduction Based on Gaussian Mixtures

In Sections I–IV, we have presented a nonparametric approach based on SASOM for color segmentation and nonstationary color tracking. We also investigate a parametric approach. Our basic idea is to using unlabeled data to help supervised learning based on the EM framework.

1) The EM Framework: When we treat the pixels in the new images as unlabeled data, the EM approach can be applied to this transductive learning problem, since the labels of unlabeled pixels can be treated as missing values.

The training data set $\mathcal{D}$ is a union of a set of labeled data set $\mathcal{L}$ and a set of unlabeled set $\mathcal{U}$. When we assume sample independency, the model parameters $\mathbf{\Theta}$ can be estimated by maximizing *a posteriori* probability $p\left(\mathbf{\Theta}|\mathcal{D}\right)$. Equivalently, this can be done by maximizing $\lg\left(p(\mathbf{\Theta}|\mathcal{D})\right)$. Let $l(\mathbf{\Theta}|\mathcal{D}) = \lg\left(p(\mathbf{\Theta})p(\mathcal{D}|\mathbf{\Theta})\right)$. When introducing a binary indicator $\mathbf{z}_i = (z_{i1}, \ldots, z_{iC})$, where $z_{ij} = 1$ iff $y_i = O_j$, and $z_{ij} = 0$ otherwise, we have

$$l(\Theta|\mathcal{D}, \mathcal{Z}) = \lg\left(p(\Theta)\right) + \sum_{\mathbf{x}_i \in \mathcal{D}} \sum_{j=1}^{C} z_{ij} \lg\left(p(O_j|\Theta) p(\mathbf{x}_i|O_j; \Theta)\right). \qquad (22)$$

The EM algorithm estimates the parameters $\mathbf{\Theta}$ by an iterative hill climbing procedure, which alternatively calculates $E[\mathcal{Z}]$, the expected values for all missing data, and estimates the parameters $\mathbf{\Theta}$ given $E[\mathcal{Z}]$. The EM algorithm generally reaches a local maximum of $l(\mathbf{\Theta}|\mathcal{D})$. It consists of two iterative steps.

- E-step: set $\hat{\mathcal{Z}}^{(k+1)} = E\left[\mathcal{Z}|\mathcal{D};\hat{\Theta}^{(k)}\right]$
- M-step: set $\hat{\Theta}^{(k+1)} = \arg\max_\theta p\left(\Theta|\mathcal{D};\hat{\mathcal{Z}}^{(k+1)}\right)$

where $\hat{\mathcal{Z}}^{(k)}$ and $\hat{\Theta}^{(k)}$ denote the estimation for $\mathcal{Z}$ and Θ at the kth iteration, respectively.

When the size of the labeled set is small, EM basically performs an unsupervised learning, except that labeled data are used to identify the components. If the probabilistic structure, such as the number of components in mixture models, is known in advance, EM could estimate true probabilistic model parameters. Otherwise, the performance could be very bad. Such a structure assumption for the probabilistic structure of the data space is important for the success of EM. Generally, when we do not have such prior knowledge about the data distribution, a Gaussian distribution could be assumed to represent a class. However, this assumption is often invalid in practice, which is partly the reason that unlabeled data could hurt the classifier.

When such a structure assumption does not hold, EM could probably fail. One approach to this problem is to try every possible structure and select the best one. However, it needs more computational resources. An alternative is to find a mapping such that the data points are clustered in the mapped data space, in which the probabilistic structure could be simplified and captured by simpler Gaussian mixtures.

2) Multiple Discriminant Analysis (MDA): MDA [8] offers a possible way to relax the assumption of probabilistic structure. MDA is a natural generalization of Fisher's linear discrimination (LDA) in the case of multiple classes. MDA offers many advantages and has been successfully applied to many tasks. The basic idea behind MDA is to find a linear transformation $\mathbf{W}$ to map the original d_1 dimensional data space to a new d_2 space such that the ratio between the between-class scatter and within-class scatter is maximized in the new space.

MDA offers a means to catch major differences between classes and discount factors that are not related to classification. Some features most relevant to classification are automatically selected or combined by the linear mapping $\mathbf{W}$ in MDA, although these features may not have substantial physical meanings any more. Another advantage of MDA is that the data are clustered to some extent in the projected space, which makes it easier to select the structure of Gaussian mixture models. Details can be found in [8].

3) The D-EM Algorithm: It is apparent that MDA is a supervised statistical method, which requires a large set of labeled samples to estimate some statistics such as mean and covariance in training. When we do not have a large training data set at hand, we may want to think of combining EM with MDA to make up the number of labeled data. By combining MDA with the EM framework, our proposed method, the D-EM, is such a way to make use of both labeled and unlabeled training data by combining supervised and unsupervised paradigms. The basic idea of D-EM is to identify some "similar" samples in the unlabeled data set to enlarge the labeled data set so that supervised techniques are made possible in such an enlarged labeled set.

D-EM begins with a weak classifier learned from the initial labeled set. Certainly, we do not expect much from this weak classifier. However, for each unlabeled sample $\mathbf{x}_j \in \mathcal{U}$, the classification confidence $\mathbf{w}_j = \{w_{jk}, k = 1,\ldots,C\}$ can be given based on the probabilistic label $\mathbf{l}_j = \{l_{jk}, k = 1,\ldots,C\}$ assigned by this weak classifier

$$l_{jk} = \frac{p\left(\mathbf{W}^T\mathbf{x}_j|O_k\right)p(O_k)}{\sum_{k=1}^{C} p\left(\mathbf{W}^T\mathbf{x}_j|O_k\right)p(O_k)} \tag{23}$$

$$w_{jk} = \lg\left(p(\mathbf{W}^T\mathbf{x}_j|O_k)\right) \quad k = 1,\ldots,C. \tag{24}$$

Equation (24) is just a heuristic to weight unlabeled data $\mathbf{x}_j \in \mathcal{U}$, although there may be many other choices.

After that, MDA is performed on the new weighted data set

$$\mathcal{D}' = \mathcal{L}\bigcup\{\mathbf{x}_j, \mathbf{l}_j, \mathbf{w}_j : \forall\, \mathbf{x}_j \in \mathcal{U}\}$$

which is linearly projected to a new space of dimension $C-1$ but unchanging the labels and weights

$$\hat{\mathcal{D}} = \left\{\mathbf{W}^T\mathbf{x}_j, y_j : \forall\, \mathbf{x}_j \in \mathcal{L}\right\}\bigcup\left\{\mathbf{W}^T\mathbf{x}_j, \mathbf{l}_j, \mathbf{w}_j : \forall\, \mathbf{x}_j \in \mathcal{U}\right\}.$$

Then parameters $\mathbf{\Theta}$ of the probabilistic models are estimated on $\hat{\mathcal{D}}$, so that the probabilistic labels are given by the Bayesian classifier according to (23). The algorithm iterates over these three steps, expectation-discrimination-maximization. The algorithm can be terminated by several methods such as presetting the iteration times, comparing a threshold and the difference of the parameters between consecutive two iterations, and using cross-validation.

It should be noted that the simplification of probabilistic structures is not guaranteed in MDA. If the components of data distribution are mixed up, it is very unlikely to find such a linear mapping. Our experiments show that D-EM works better than pure EM.

4) Model Transduction by D-EM: The application of D-EM to color tracking is straightforward. In our current implementation, in the transformed space, both classes (foreground and background) are represented by a Gaussian distribution with three parameters, the mean μ_i, the covariance $\mathbf{\Sigma}_i$ and *a priori* probability P_i.

We use three schemes to bootstrap the tracking. The first method is to manually collect and label some pixels (100 samples) from both the interested object and background. An alternative is to put the interested object in the middle of the image so that some data can be automatically collected. The third method is to detect the moving region by image differences in the first several frames. We assume that we are interested in the object with the largest motion.

For each new image I_t, by setting a confidence level, the color classifier M_{t-1} at time $t-1$ divides I_t into two parts: labeled set $\mathcal{L}_t$ and unlabeled set $\mathcal{U}_t$. $\mathcal{L}_t$ is confidently labeled by M_{t-1}. The D-EM algorithm identifies some "similar" samples in $\mathcal{U}_t$ to the labeled samples in an unsupervised sense. Therefore, good discriminating color features can be automatically selected through the enlarged labeled data set. After a Bayesian classifier is designed in the new feature space, it is used to probabilistically label I_t. Through several iterations, the classifier M_{t-1} has been transduced to M_t by D-EM.

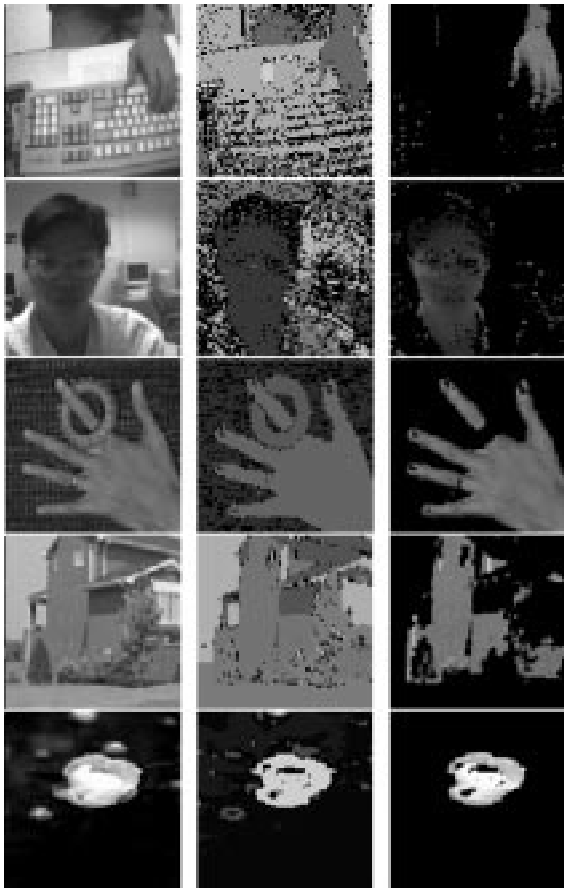


Fig. 5. Some results of image segmentation based on SASOM. Left column: source color images. Middle column: segmented images. Right column: interested color regions.

VI. Experiments

This section reports our experiments of SASOM for image segmentation, SASOM transduction, and D-EM for nonstationary color tracking of faces and hands in video sequences.

A. Experiments Based on SASOM

Our color segmentation algorithm based on SASOM has been tested on a large variety of pictures. And our localization system that integrates this color segmentation algorithm has run under a wide range of operating conditions. Such real-time system has been employed in vision-based gesture analysis [31]. Extensive experiments show that our color segmentation algorithm is fast, automatic, and accurate, and the proposed localization system is robust, real-time, and reliable. This color segmentation algorithm can also be applied to other segmentation tasks.

1) Performance of Segmentation Using SASOM: One parameter we should specify in SASOM is the maximum number of neurons. If the scene is simple, we set the maximum number to two or three. If the scene is complex, we set it to ten or more. In between, we use six.

Fig. 5 show some segmentation results. The left column shows source color images, the middle column shows segmented images, and the right column shows separated color regions. The color of each segmented color region is the average color of this region. Each pixel in the source images is assigned a label by the SASOM algorithm, and this label is used as a mask to separate the corresponding color region. Our segmentation algorithm works well through these experiments. When the background has less color distracters, this algorithm finds exact color regions. Since texture is not used in the segmentation, segmentation results will be noisy when there is color distracter texture in the background. Hand and face images are taken from a cheap camera in the indoor environment in our labs. Our algorithm can also successfully segment hand regions and face regions.

2) Performance of Hand Tracking Using SASOM: A typical hand-tracking scenario is controlling the display or simulating a 3-D mouse in desktop environments. A camera mounted at the top of a desktop computer looks below at the keyboard area, and gives an image sequence of a moving hand. Another typical application is to track human face. Our localization system is able to simultaneously localize multiple objects, which is useful in tracking of moving human.

Since our localization system is essentially based on a global segmentation algorithm, it does not largely rely on the tracking results of previous frames. Even if the tracker may get lost in some frames for some reasons, it can recover by itself without interfering the targets. In this sense, the tracking system is very robust.

Our proposed system can handle changing lighting conditions to some extend because of the transduction of the SASOM color classifier. At the same time, since the hue and saturation are given more weight than intensity, our system is insensitive to the change of lighting intensity such as in the situations that the objects are shadowed or the intensity of the light source changes. However, there are still some problems. Insufficient lighting, too strong lighting, very dark, or bright backgrounds may bring some troubles to the color segmentation algorithm, since hue and saturation become unstable and the system does not give more weights to intensity. If the lighting condition changes dramatically, the color segmentation algorithm may fail since the color model transduction cannot be guaranteed.

Some hand tracking results in our experiments are given in Fig. 6. In this experiment, a hand is moving around with the interference of a moving book. The book is also casting shadows so that the color of skin is changing. The blue boxes are the bounding boxes of the interested color region (Demo video sequence can be obtained at http://www.ece.northwestern.edu/~yingwu).

Our tracking system is very robust and efficient from this experiment with cluttered backgrounds. Since a book is interfering the hand by shading the light, such system can still find a correct bounding box. Sometimes, due to the sudden change of lighting conditions, the tracker may be lost. However, it can quickly recover. Different skin tones do not affect our system. The color region of interest in the first image is used to initialize the SASOM so that it can work with nearly any users. Our system has been tested in extensive experiments.

B. Experiments Based on Gaussian Mixture Model

1) A Simulation of D-EM: To validate the effectiveness of D-EM, we performed a simulation experiment. At current time t in tracking, since the color model M_{t-1} may not be

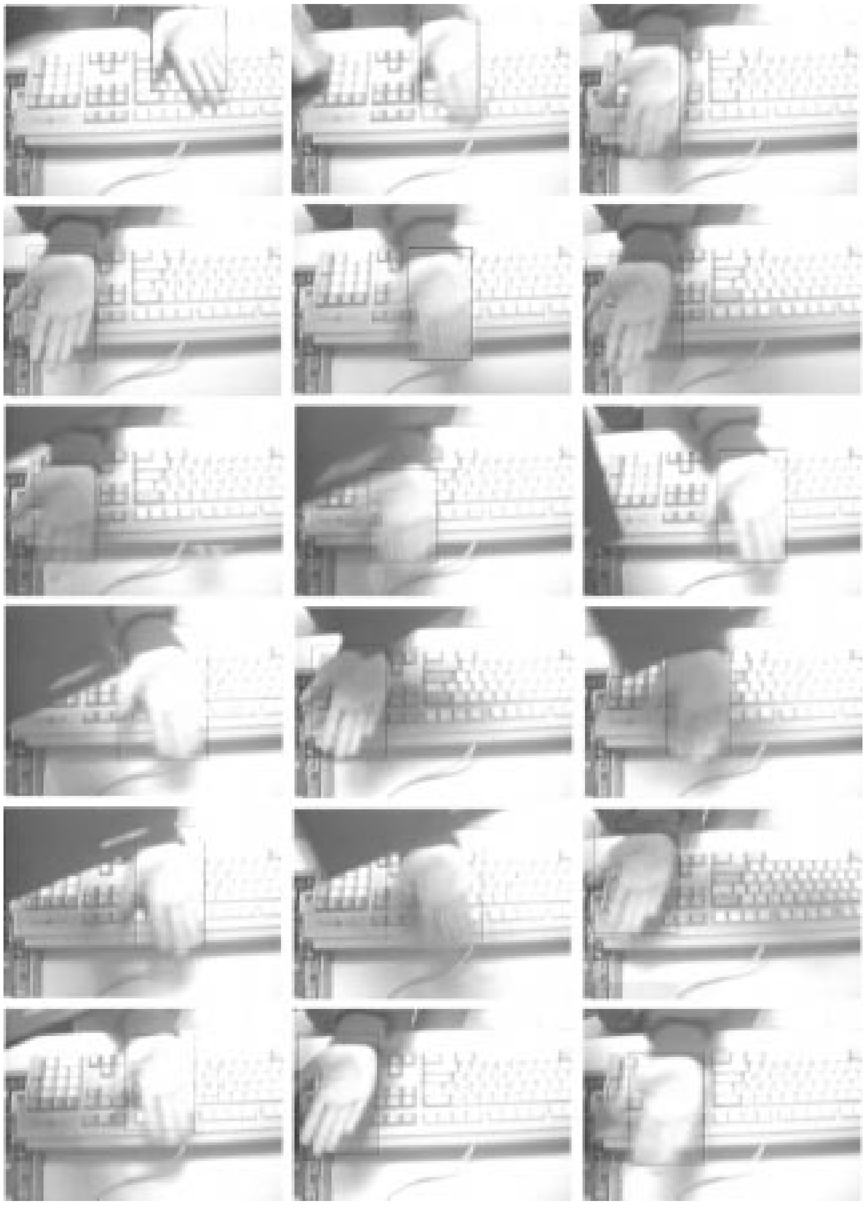


Fig. 6. Results of hand tracking with 18 frames taken from image sequences. A moving hand with interfering of a book is localized. The blue boxes are the bounding box of the interested color region.

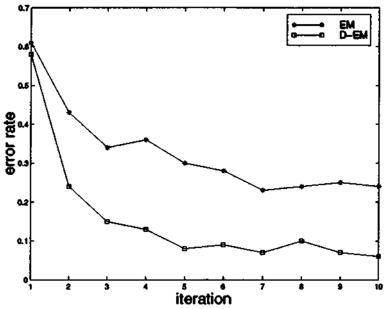


Fig. 7. A comparison between EM and D-EM. Both EM and D-EM converge after several iterations, but D-EM gives a lower classification error rate.

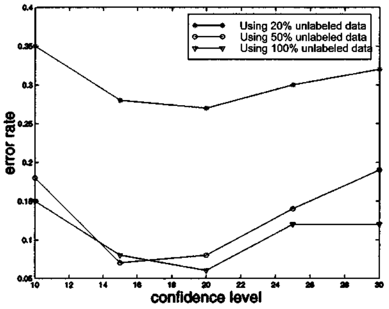


Fig. 8. The effect of number of labeled and unlabeled data in D-EM. Different numbers of labeled and unlabeled data are feeded to D-EM. When using 3750 unlabeled data, the lowest error rate drops to 6.9%.

able to give a good segmentation on the image I_t, the image at time t is not labeled (segmented) so that the ground truth for the new data set is not available. However, to evaluate our algorithm, we assume a known ground truth in order to calculate classification errors, although such a ground truth is not available in real applications.

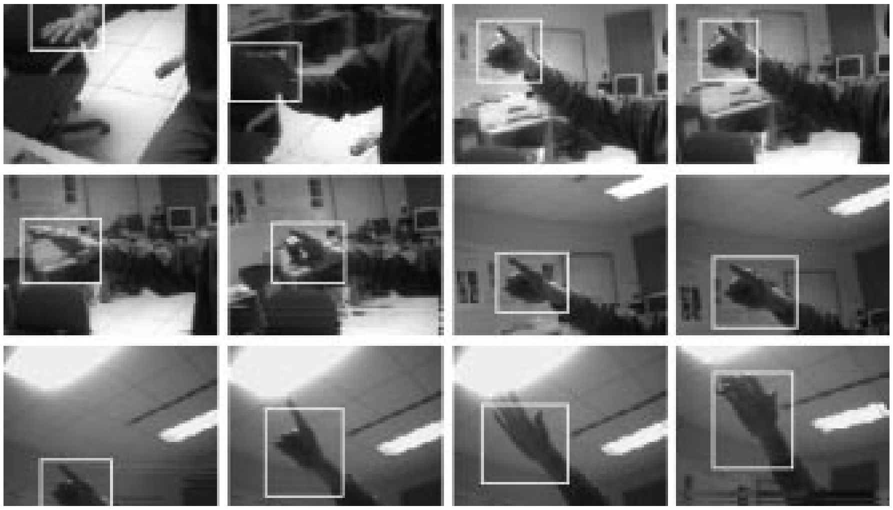


Fig. 9. Hand localization by D-EM.

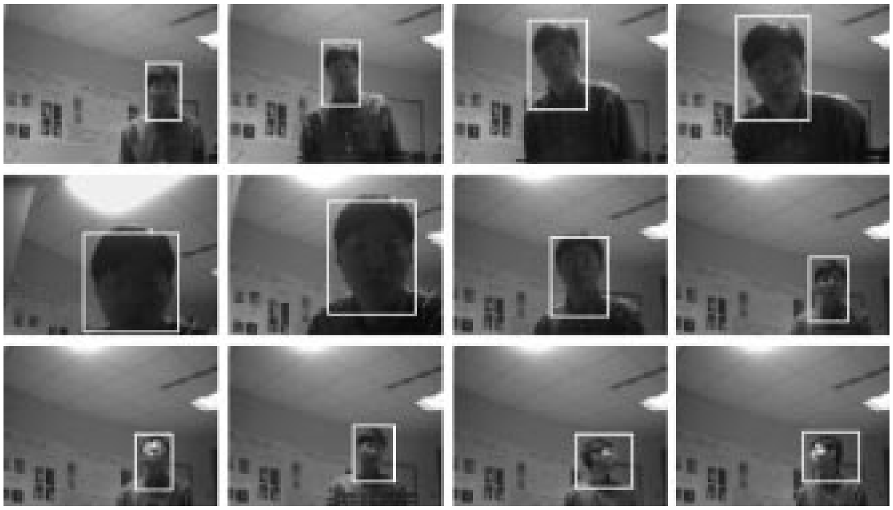


Fig. 10. Face localization by D-EM.

We use two images (resolution 100 × 75), where I_1 is a segmented image, and I_2 has the same content as I_1 except that the color distribution of I_2 is transformed by shifting the R element of every pixel by 20 such that I_2 looks like adding a red filter to I_1. A color classifier is learned for I_1 with error rate less than 5%. In this simple situation, this color classifier would fail to correctly segment target regions out from I_2, since the skin color in I_2 is much different. Actually, it has error rate of 35.2% on I_2 in our experiments.

Fig. 7 shows the comparison between EM and D-EM. In this experiment, both EM and D-EM converge after several iterations, but D-EM gives a lower classification error rate (6.9% versus 24.5%). To investigate the effect of the unlabeled data used in D-EM, we feed the D-EM algorithm a different number of labeled and unlabeled samples. The number of labeled data is controlled by the confidence level. In this experiment, confidence level is the same as the size of the labeled set. In general, combining unlabeled data can largely reduce the classification error when labeled data are very limited. When using 20% (1500) unlabeled data, the lowest error rate achieved is 27.3%. When using 50% (3750) unlabeled data, the lowest error rate drops to 6.9%. The transduced color classifier gives around 30% more accuracy. Fig. 8 shows the effect of different sizes of labeled and unlabeled data sets in D-EM.

2) Hand and Face Localization Based on D-EM: Based on the D-EM algorithm, we implemented a nonstationary color tracking system, which is also applied to a gesture interface, in which hand gesture commands are localized and recognized to provide inputs to a virtual environment application. These experiments ran at 15–20 Hz on a single processor SGI O2 R10000 workstation.

Figs. 9 and 10 show two examples of hand and face localization in a typical lab environment. Both cases are difficult for static color models. In Fig. 9, the skin color in different parts of hand are different. The camera moves from downwards to upwards and the lighting conditions on the hand are different. Hand becomes darker when it shadows the light sources in several frames. In Fig. 10, skin color changes a lot when the head moves back and forth, and turns around. We also observed that D-EM failed under dramatic lighting changes such as turning on/off lights.

VII. Conclusion and Future Work

Computer vision techniques provide promising ways to human–computer interaction through understanding human movements from visual data. An important step to achieve this goal is the robust and accurate tracking of the human body such as hand and face. However, cluttered backgrounds, unknown lighting conditions and multiple moving objects make the tracking tasks challenging. This paper mainly concentrated on color-based image segmentation and color-based target tracking by addressing these difficulties.

This paper presented a new representation of color model based on the proposed SASOM neural network, in which the structure of the SOM could be learned in training. This SASOM representation could afford efficient image segmentation through a competition process of the neurons in SASOM. Then we investigated the nonstationary color-based tracking problem. A challenge of this task lies in the fact that the lighting condition and the background may not be static, such that the color distributions in the image sequence is not stationary. In order to capture the nonstationary color distributions, our method, i.e., SASOM transduction, transduces the SASOM over time by combing supervised and unsupervised learning paradigms. Based on the SASOM model, we achieved a robust real-time tracking system that has been widely used in our further research.

We notice that the SASOM transduction is not mature, and it needs more efforts to find a better way to combine supervised and unsupervised learning schemes. In addition, since the process of competition among all neurons is essentially parallel, the tracking system can be made much faster by parallel implementation of the competition process. Currently, our localization system outputs a bounding box of the target. Shape analysis of localized target will be extended to estimate its 3-D motion.

Besides the nonparametric SASOM model, we also looked into a parametric approach based on the Gaussian mixture model. Since the nonstationary color tracking could be formulated as a model transduction problem, our study focused on the problem of learning a new Gaussian mixture model based on an old mixture model and a set of unlabeled training data, e.g., unsegmented color pixel data. Integrating discriminant analysis and the EM framework, the proposed D-EM algorithm offers a means to relax the assumption of probabilistic structures of data distribution. In addition, the proposed D-EM algorithm is able to select a good color space automatically. Some promising color-based tracking results were also achieved by the D-EM approach.

One of the future research directions of the D-EM algorithm is to explore the nonlinear case of MDA. In addition, the convergence and stability analysis should be studied in the future work. Currently, the confidence level is an important parameter in the transduction to control the size of labeled set. It needs further studies.

Acknowledgment

The authors would like to thank the anonymous reviewers for their comments.

References

[1] S. C. Ahalt, A. Krishnamurthy, P. Chen, and D. Melton, “Competitive learning algorithms for vector quantization,” *Neural Networks*, vol. 3, pp. 277–290, 1990.

[2] P. J. Angeline, G. Saunders, and J. Pollack, “An evolutionary algorithm that constructs recurrent neural networks,” *IEEE Trans. Neural Networks*, vol. 5, pp. 54–65, Jan. 1994.

[3] A. Blake and M. Isard, *Active Contours*. London, U.K.: Springer-Verlag, 1998.

[4] D. Choi and S. Park, “Self-creating and organizing neural networks,” *IEEE Trans. Neural Networks*, vol. 5, pp. 1072–1085, Nov. 1994.

[5] D. Comaniciu and P. Meer, “Robust analysis of feature spaces: Color image segmentation,” in *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, San Juan, Puerto Rico, June 1997, pp. 750–755.

[6] M. Cottrell and J. C. Fort, “A stochastic model of retinotopy: A self-organizing process,” *Biol. Cybern.*, vol. 53, pp. 405–411, 1986.

[7] T. Darrell and A. Pentland, “Active gesture recognition using partially observable Markov decision processes,” in *Proc. IEEE Int. Conf. Pattern Recognition*, vol. 3, 1996, pp. 984–988.

[8] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

[9] E. Erwin, K. Obermayer, and K. Schulten, “Self-organizing maps: Ordering, convergence properties and energy functions,” *Biol. Cybern.*, vol. 67, pp. 47–55, 1992.

[10] ——, “Self-organizing maps: Stationary states, metastability and convergence rate,” *Biol. Cybern.*, vol. 67, pp. 35–45, 1992.

[11] B. Fritzke, “Growing cell structures—A self-organzing network for unsupervised and supervised learning,” *Neural Networks*, vol. 7, pp. 1441–1460, 1994.

[12] ——, “A growing neural gas network learns topologies,” in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. Cambridge, MA: MIT Press, 1995, pp. 625–632.

[13] A. Gersho, “On the structure of vector quantizers,” *IEEE Trans. Inform. Theory*, vol. 28, pp. 157–166, 1992.

[14] K. Imagawa, S. Lu, and S. Igi, “Color-based hands tracking system for sign language recognition,” in *Proc. Int. Conf. Face Gesture Recognition*, 1998, pp. 462–467.

[15] M. Isard and A. Blake, “ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework,” in *Proc. European Conf. Comput. Vision*, vol. 1, 1998, pp. 767–781.

[16] S. Jockusch and H. Ritter, “Self-organizing maps: Local competition and evolutionary optimization,” *Neural Networks*, vol. 7, pp. 1229–1239, 1994.

[17] M. Jones and J. Rehg, “Statistical Color Models With Application to Skin Detection,” Compaq Cambridge Research Lab., Cambridge, MA, CRL-98-11, 1998.

[18] R. Kjeldsen and J. Kender, “Finding skin in color images,” in *Proc. 2nd Int. Conf. Automat. Face Gesture Recognition*, 1996, pp. 312–317.

[19] T. Kohonen, “Self-organized formation of topologically correct feature maps,” *Biol. Cybern.*, vol. 43, pp. 59–69, 1982.

[20] ——, “Learning vector quanitizer,” *Neural Networks*, vol. 1, p. 308, 1986.

[21] T.-C. Lee, *Structure Level Adaptation for Artificial Neural Networks*. Boston, MA: Kluwer, 1991.

[22] Z. P. Lo and B. Bavarian, “On the rate of convergence in topology preserving neural networks,” *Biol. Cybern.*, vol. 65, pp. 55–63, 1991.

[23] Z. P. Lo, Y. Yu, and B. Bavarian, “Analysis of the convergence properties of topology preserving neural networks,” *IEEE Trans. Neural Networks*, vol. 4, pp. 207–220, 1993.

[24] F. M. Muiler, “Statistical analysis of self-organization,” *Neural Networks*, vol. 8, pp. 717–727, 1995.

[25] V. Pavlović, R. Sharma, and T. S. Huang, “Visual interpretation of hand gestures for human computer interaction: A review,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 677–695, July 1997.

[26] Y. Raja, S. McKenna, and S. Gong, “Color model selection and adaptation in dynamic scenes,” in *Proc. European Conf. Comput. Vision*, 1998, pp. 460–475.

[27] H. Ritter and K. Schulten, “On the stationary state of Kohonen’s self-organizing sensory mapping,” *Biol. Cybern.*, vol. 54, pp. 99–106, 1986.

[28] M. J. Swain and D. H. Ballard, “Color indexing,” *Int. J. Comput. Vision*, vol. 7, pp. 11–32, 1991.

[29] K. Toyama and Y. Wu, “Bootstrap initialization of nonparametric texture models for tracking,” in *Proc. European Conf. Comput. Vision*, Ireland, 2000.

[30] C. Wren, A. Azarbayejani, T. Darrel, and A. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 9, pp. 780–785, July 1997.

[31] Y. Wu and T. S. Huang, "Capturing articulated human hand motion: A divide-and-conquer approach," in *Proc. IEEE Int. Conf. Comput. Vision*, Corfu, Greece, Sept. 1999, pp. 606–611.

[32] ——, "Human hand modeling, analysis and animation in the context of HCI," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, Kobe, Japan, Oct. 1999, pp. 6–10.

[33] ——, "Robust visual tracking by co-inference learning," in *Proc. IEEE Int. Conf. Comput. Vision*, vol. II, Vancouver, July 2001, pp. 26–33.

[34] Y. Wu, Q. Liu, and T. S. Huang, "An adaptive self-organizing color segmentation algorithm with application to robust real-time human hand localization," in *Proc. Asian Conf. Comput. Vision*, Taipei, Taiwan, Jan. 2000, pp. 1106–1111.

[35] J. Yang, W. Lu, and A. Waibel, "Skin-color modeling and adaptation," in *Proc. Asian Conf. Comput. Vision*, 1998, pp. 687–694.

[36] Y. Zheng and J. Greenleaf, "The effect of concave and convex weight adjustment in self-organizing maps," *IEEE Trans. Neural Networks*, vol. 7, pp. 1458–1471, Nov. 1996.

Ying Wu (M'01) received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 1994, the M.S. degree from Tsinghua University, Beijing, China, in 1997, and the Ph.D. degree in electrical and computer engineeringfrom the University of Illinois at Urbana-Champaign (UIUC), Urbana, in 2001.

From 1997 to 2001, he was a Graduate Research Assistant at the Image Formation and Processing Group of the Beckman Institute for Advanced Science and Technology, UIUC. During summer 1999 and 2000, he was a Research Intern with the Vision Technology Group, Microsoft Research, Redmond, WA. Since 2001, he had been on the faculty of the Department of Electrical and Computer Engineering at the Northwestern University, Evanston, Illinois. His current research interests include computer vision, computer graphics, machine learning, human–computer intelligent interaction, image–video processing, multimedia, and virtual environments.

Dr. Wu received the Robert T. Chien Award from UIUC in 2001.

Thomas S. Huang (S'61–M'63–SM'76–F'79–LF'01) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, China, and the M.S. and Sc.D. degrees in electrical engineering from the Massachusetts Institute of Technology (MIT), Cambridge.

He was on the Faculty of the Department of Electrical Engineering at MIT from 1963 to 1973, and on the faculty of the School of Electrical Engineering and Director of its Laboratory for Information and Signal Processing at Purdue University, West Lafayette, IN, from 1973 to 1980. In 1980, he joined the University of Illinois at Urbana-Champaign, where he is now William L. Everitt Distinguished Professor of Electrical and Computer Engineering, Research Professor at the Coordinated Science Laboratory, and Head of the Image Formation and Processing Group at the Beckman Institute for Advanced Science and Technology. During his sabbatical leaves, he has worked at the MIT Lincoln Laboratory, the IBM Thomas J. Watson Research Center, and the Rheinishes Landes Museum in Bonn, Germany, and held Visiting Professor positions at the Swiss Institutes of Technology in Zurich and Lausanne, University of Hannover, Germany, INRS-Telecommunications of the University of Quebec, Montreal, QC, Canada, and the University of Tokyo, Japan. He has served as a Consultant to numerous industrial firms and government agencies both in the United States and abroad. His professional interests include information technology, especially the transmission and processing of multidimensional signals. He has published 12 books and more than 300 papers in network theory, digital filtering, image processing, and computer vision.

Dr. Huang is a Fellow of the International Association of Pattern Recognition and the Optical Society of America; and has received a Guggenheim Fellowship, an A.V. Humboldt Foundation Senior U.S. Scientist Award, and a fellowship from the Japan Association for the Promotion of Science. He received the IEEE Acoustics, Speech, and Signal Processing Society's Technical Achievement Award in 1987, and the Society Award in 1991. He is a Founding Editor of the *International Journal Computer Vision, Graphics, and Image Processing*, and Editor of the Springer Series in Information Sciences, published by Springer-Verlag.