Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2009, Article ID 724947, 10 pages
doi:10.1155/2009/724947

Research Article

Model-Based Hand Tracking by Chamfer Distance and Adaptive Color Learning Using Particle Filter

Chutisant Kerdvibulvech and Hideo Saito

Department of Information and Computer Science, Keio University, 3-14-1 Hiyoshi, Kohoku-ku 223-8522, Japan

Correspondence should be addressed to Chutisant Kerdvibulvech, chutisant@ozawa.ics.keio.ac.jp

Received 31 January 2009; Revised 23 June 2009; Accepted 3 October 2009

Recommended by Amit Roy-Chowdhury

We propose a new model-based hand tracking method for recovering of three-dimensional hand motion from an image sequence. We first build a three-dimensional hand model using truncated quadrics. The degrees of freedom (DOF) for each joint correspond to the DOF of a real hand. This feature extraction is performed by using the Chamfer Distance function for the edge likelihood. The silhouette likelihood is performed by using a Bayesian classifier and the online adaptation of skin color probabilities. Therefore, it is to effectively deal with any illumination changes. Particle filtering is used to track the hand by predicting the next state of three-dimensional hand model. By using these techniques, this method adds the useful ability of automatic recovery from tracking failures. This method can also be used to track the guitarist's hand.

1. Introduction

Acoustic guitars are currently very popular and as a consequence, research about guitars is a popular topic in the field of computer vision for musical applications.

Maki-Patola et al. [1] proposed a system called "Virtual Air Guitar" using computer vision. Their aim was to create a virtual air guitar which does not require a real guitar but produces music similar to a player using a real guitar. Liarokapis [2] proposed an augmented reality system for guitar learners. The aim of his work is to show the augmentation (e.g., the positions where the learner should place the fingers to play the correct chords) on an electric guitar as a guide for the novice player. Motokawa and Saito [3] built a system called "Online Guitar Tracking" that supports a guitarist by using augmented reality. This is done by showing a virtual model of the fingers on a stringed guitar as a teaching aid for anyone learning how to play the guitar.

These systems do not aim to detect the fingering and handing which a player is actually using (a pair of gloves are tracked in [1], and graphics information is overlaid on captured video in [2] and [3]). We have developed a different approach from most of these researches.

In our previous work [4], we proposed a method that accurately locates the positions of the fingertips of a guitar player by employing computer vision aid. However, it cannot track the whole structure of guitarist's hand. In order to improve the previous work, we present a model-based hand tracking which will be further applied to the recovery of three-dimensional hand motion of guitar player from an image sequence, without the use of markers. Example input and output images are given in Figure 1.

A challenge for tracking the hand of a guitar player is that, while playing the guitar, the fingers are not stretched out separately. Thus the existing model-based hand tracking methods such as [5], [6], and [7] are not directly applicable to the guitarist's hand tracking as the fingers are usually bent while playing the guitar. Moreover, the background is dynamic and nonuniform (e.g., guitar neck and natural scene) which makes it more difficult for background segmentation. Also, for many classic guitars, the colors of frets and strings are very similar to skin color. As a result it is not an easy task to track the hand correctly.

To begin with, we construct a three-dimensional hand model [8] using truncated quadrics as building blocks, approximating the anatomy of a real human hand [9]. A

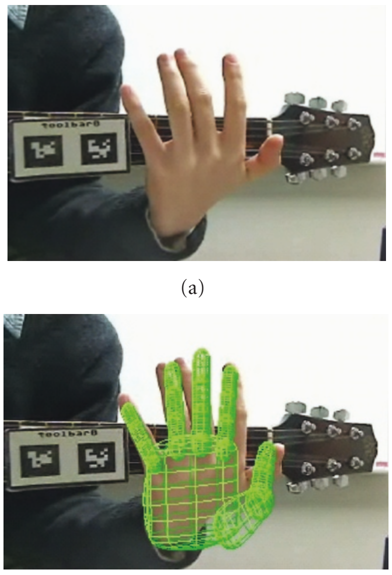


Figure 1: Model-based guitarist's hand tracking: (a) sample input image and (b) sample output image (projected corresponding three-dimensional hand model).

hierarchical model with 27 degrees of freedom (DOF) is used. The DOF for each joint correspond to the DOF of a real hand. Then we extract corresponding features (edges and silhouette) between three-dimensional hand model and input image. The Canny edge detection is used to extract the edge. Then, the Chamfer Distance function [10] is used for edge likelihood. The silhouette is determined by using a Bayesian classifier and the online adaptation of skin color probabilities [11, 12]. By using the online adaptation, this method is able to cope well with illumination changes. Following this, the particle filter [13] is applied to track the hand by predicting the next state of three-dimensional hand model. As a result, the system enables us to visually track the hand, which can be applied to track the hand of a guitarist.

The advantage of particle filter is that the tracker is able to initialize and recover. Particle filter uses a lot of state vectors (particles) to represent possible solution in each time instance. If the hand moves fast until lost tracks, it can automatically recover from tracking failures. Also, in [9] they use template-based method. In their work, the range of allowed hand motion is limited by the number of templates that need to be stored. Thus, their method requires creating many templates manually enough at the first time, unless its coverage will not be sufficient for hand's movements. In contrast, by applying particle filter, it can avoid the cumbersome process of manually generating a lot of templates.

The work that is similar to ours is by De la Gorce et al. [14] They focused on recovery of geometric and photometric pose parameters of a skinned hand model from monocular image sequences. However, they do not aim to apply hand model-based to track the hand of guitarist which includes with guitar neck in the background. In their work, their

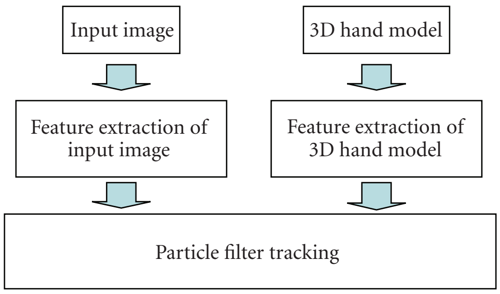

Input image
3D hand model
Feature extraction of input image
Feature extraction of 3D hand model
Particle filter tracking

Figure 2: Method overview.

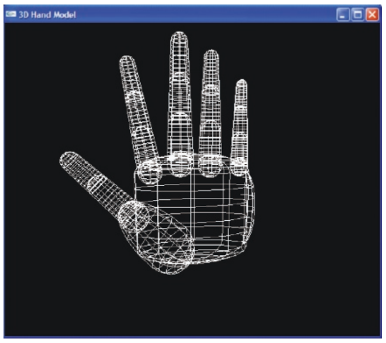
assumption is that the background image is basically static and was also obtained from a frame where the hand was not visible. In other words, when they define the image synthesis process, they assume that the background image is known (if their background is not static, they relax this constraint by assuming that the color distribution associated to the background is available). In the case of guitarist's hand tracking, the background is dynamic because the guitar neck is not fixed to the camera which makes the situation different. Due to the dynamic movement of guitar position, it cannot simply use background subtraction from a frame where the hand is invisible at the first time. In addition, for many classic guitars, the colors of frets and strings are similar to skin color which makes it difficult for segmenting the hand from the guitar neck. The method of robust hand segmentation from the guitar neck is undoubtedly needed. For this reason, in this paper we apply a Bayesian classifier and the online adaptation of skin color probabilities [11, 12] to robustly segment the hand region from the guitar neck. This approach can also deal well with illumination changes.

2. Method

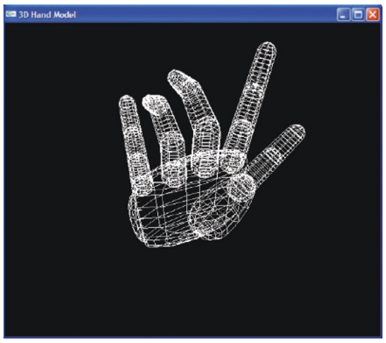
Figure 2 shows the schematic of the implementation. We firstly build the three-dimensional hand model using truncated quadrics. Following this, the feature extraction algorithm is performed for the hand model. At the same time, we also extract the feature of captured image. As the next step, we utilize a particle filter to track the hand.

2.1. Construct 3D Hand Model. This section explains the method we used for building the hand model [8, 15]. The hand model is constructed using a set of quadrics $\{Q_i\}_{i=1}^{N_Q}$, approximately representing the anatomy of a real human hand, as represented in Figure 3. The 27 DOF hand model is constructed from 39 truncated quadrics. A hierarchical model with 27 DOF is used: 6 for the global hand position, 4 for the pose of each finger, and 5 for the pose of the thumb [16]. Starting from the palm and ending at the tips, the coordinate system of each quadric is defined relative to the previous one in the hierarchy.

Truncated quadrics are joined together in order to design parts of a hand, such as a finger. There may be several ways in which to define the model shape. However, it is desirable to make all variables dependent on a small number of values



(a)



(b)

Figure 3: Our geometric hand model, built by a set of quadrics and drawn by OpenGL.

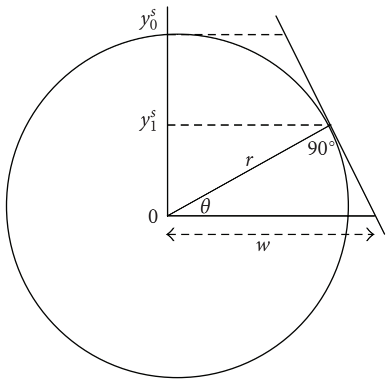


Figure 4: Building a finger model from quadrics. The joints are modeled with spheres. This shows a view of bottom joint [8].

that can be easily obtained, such as the height, width, and depth of a finger. These values are then used to set the parameters of the quadrics. Each finger is built from cone segments and truncated spheres, as shown in Figure 4. The width parameter w of the cone is uniquely determined by the height of the segment h and the radius r of the sphere, which models the bottom joint:

$$w = \frac{hr}{\sqrt{h^2 - r^2}}. \tag{1}$$

In order to create a continuous surface, the parameter y_1^s of the bottom clipping plane for the sphere is found as

$$y_1^s = \frac{r^2}{h}, \tag{2}$$

and from this the clipping parameters for the cone are found as

$$\begin{aligned} y_1^c &= h - y_1^s, \\ y_0^c &= y_1^c - f_1 H, \end{aligned} \tag{3}$$

where H is the sum of the length of all three cone segments, and f_1 is the ratio between the length of the bottom segment to the total length.

2.2. Feature Extraction. This section explains the feature extraction which is used within the algorithm. The methods explained in this section are applied for both the hand model image and the captured image. The likelihood p(z | x) relates observations z in the image to the unknown state x. The likelihoods we used are based on the edge map z^{edge} of the image (edge) as well as pixel color values z^{sil} (silhouette). These features have proved useful for detecting and tracking hands in previous work [17]. Therefore, the joint likelihood of $z = (z^{\text{edge}}, z^{\text{sil}})^T$ is approximated as

$$\begin{aligned} p(zx) &= p\left(z^{\text{edge}}, z^{\text{sil}} x\right), \\ p\left(z^{\text{edge}}, z^{\text{sil}} x\right) &\approx p\left(z^{\text{edge}} x\right) p\left(z^{\text{col}} x\right). \end{aligned} \tag{4}$$

2.2.1. Edge Likelihood. We first extract feature by considering the edge likelihood. The edge likelihood term $p(z^{(\text{edge})} \mid x)$ is based on the chamfer distance function [10]. Given the set of template points $A = \{a_i\}_{i=1}^{N_a}$ and the set of Canny edge points $B = \{b_i\}_{i=1}^{N_b}$, a quadratic chamfer distance function is given by the average of the squared distances between each point of A and its closest point in B:

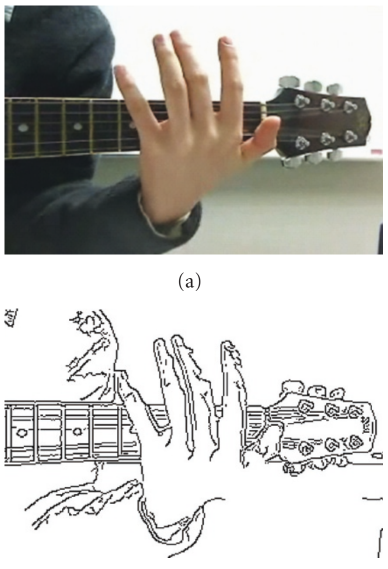
$$d(A, B) = \frac{1}{N_a} \sum_{a \in A} \min_{b \in B} \|a - b\|^2. \tag{5}$$

Example edge extraction is shown in Figure 5. The chamfer function can be computed efficiently for many model templates by using a distance transform (DT) of the edge image. This transformation takes the set of feature points B as input and assigns each location the distance to its nearest feature; that is, the DT value at location u contains the value $\min_{\mathrm{b} \in \mathrm{B}} \|\mathrm{u} - \mathrm{b}\|$. The chamfer function for a single template can be computed by correlating its points with the DT image.

Given the shape template P and the observed edge image z^{edge}, the likelihood function is defined as

$$p\left(z^{\text{edge}} \mid x\right) = \frac{1}{Z} \exp\left(-\lambda d\left(A(x), B\left(z^{\text{edge}}\right)\right)\right), \tag{6}$$

where $A(x)$ denotes the set of template points. A is generated by projecting the model using the state vector x, and B is the set of edge points obtained from the edge image z^{edge}. In our implementation, we set $\lambda = 0.03$.


(a)

(b)

Figure 5: Edge extraction: (a) input image and (b) edge extraction result by Canny edge detector.

2.2.2. Silhouette Likelihood. We also perform feature extraction by determining the silhouette likelihood. The silhouette is calculated by using a Bayesian classifier and the online adaptation of skin color probabilities [11, 12].

The learning process has two phases. In the first phase, the color probability is obtained from a small number of training images during an offline preprocess. In the second phase, we gradually update the probability automatically and adaptively from the additional training data images. The adapting process can be disabled as soon as the achieved training is deemed sufficient.

Therefore, this method allows us to get accurate color probability of the skin from only a small set of manually prepared training images. This is because the additional skin region does not need to be segmented manually. Also, because of the adaptive learning, it can be used robustly with changing illumination during the online operation.

In this way, because our method can learn color probability of hand adaptively, the background of testing images does not have to be the same. When the background is suddenly changed, the segmentation result might become error prone in the beginning but as soon as several frames are learned adaptively, the segmentation will be recovered and becomes good again.

Learning from Training Data Set. During an offline phase, a small set of training input images (20 images) is selected on which a human operator manually segments skin regions. The color representation used in this process is YUV 4 : 2 : 2 [18]. However, the Y-component of this representation is not employed for two reasons. Firstly, the Y-component corresponds to the illumination of an image pixel. By omitting this component, the developed classifier becomes less sensitive to illumination changes. Secondly, compared to a 3D color representation (YUV), a 2D color representation (UV) is lower in dimensions and, therefore, less demanding in terms of memory storage and processing costs.

Assuming that image pixels with coordinates *(x,y)* have color values $c = c(x,y)$, training data are used to calculate the following.

(i) The prior probability $P(s)$ of having skin s color in an image. This is the ratio of the skin-colored pixels in the training set to the total number of pixels of whole training images.

(ii) The prior probability $P(c)$ of the occurrence of each color in an image. This is computed as the ratio of the number of occurrences of each color c to the total number of image points in the training set.

(iii) The conditional probability $P(c \mid s)$ of a skin being color c. This is defined as the ratio of the number of occurrences of a color c within the skin-colored areas to the number of skin-colored image points in the training set.

By employing Bayes' rule, the probability $P(s \mid c)$ of a color c being a skin color can be computed by using

$$P(s \mid c) = \frac{P(c \mid s)P(s)}{P(c)}. \tag{7}$$

This equation determines the probability of a certain image pixel being skin-colored using a lookup table indexed with the pixel's color. The resultant probability map thresholds are then set to be threshold$T_{\max}$ and threshold $T_{\min}$, where all pixels with probability $P(s \mid c) > T_{\max}$ are considered as being skin-colored—these pixels constitute seeds of potential skin-colored blobs—and image pixels with probabilities $P(s \mid c) > T_{\min}$ where $T_{\min} < T_{\max}$ are the neighbors of skin-colored image pixels being recursively added to each color blob. The rationale behind this region growing operation is that an image pixel with relatively low probability of being skin-colored should be considered as a neighbor of an image pixel with high probability of being skin-colored. The values for $T_{\max}$ and $T_{\min}$ should be determined by test experiments (we use 0.5 and 0.15, resp., in the experiment in this paper). A standard connected component labelling algorithm (i.e., depth-first search) is then responsible for assigning different labels to the image pixels of different blobs. Size filtering on the derived connected components is also performed to eliminate small isolated blobs that are attributed to noise and do not correspond to interesting skin-colored regions. Each of the remaining connected components corresponds to a skin-colored blob.

Adaptive Learning. The success of the skin color detection crucially depends on whether or not the illumination conditions during the online operation of the detector are similar to those during the acquisition of the training data set. Despite the fact that using the UV color representation model has certain illumination independent characteristics, the skin color detector may produce poor results if the illumination conditions during online operation are considerably different to those used in the training set. Thus, a means of adapting the representation of skin-colored image pixels according to the recent history of detected colored

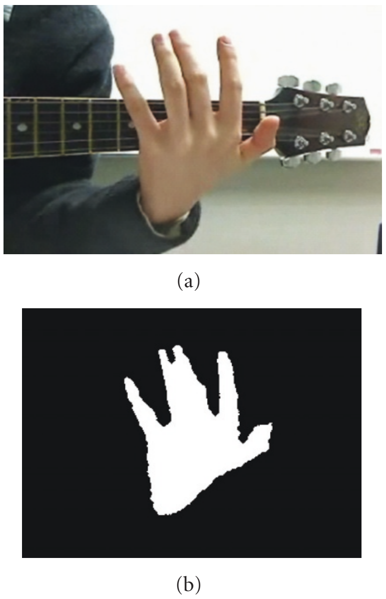


Figure 6: Hand extraction by silhouette likelihood: (a) input image and (b) hand extraction result by adaptive skin color learning.

pixels is required. To solve this problem, skin color detection maintains two sets of prior probabilities. The first set consists of $P(s)$, $P(c)$, and $P(c \mid s)$ that have been computed offline from the training set. The second is made up of $P_W(s)$, $P_W(c)$, and $P_W(c|s)$ corresponding to the $P(s)$, $P(c)$, and $P(c|s)$ that the system gathers during the W most recent frames respectively. Obviously, the second set better reflects the "recent" appearance of skin-colored objects and is therefore better adapted to the current illumination conditions. Skin color detection is then performed based on the following weighted moving average formula:

$$P_A(s \mid c) = \gamma P(s \mid c) + (1 - \gamma) P_W(s \mid c), \tag{8}$$

where γ is a sensitivity parameter that controls the influence of the training set in the detection process, $P_A(s \mid c)$ represents the adapted probability of a color c being a skin color, and $P(s \mid c)$ and $P_W(s \mid c)$ are both given by (1) but involve prior probabilities that have been computed from the whole training set (for $P(s \mid c)$) and from the detection results in the last W frames (for $P_W(s \mid c)$). In our implementation, we set $\gamma = 0.8$ and $W = 5$.

Thus, the hand skin color probability can be determined adaptively. By using online adaptation of skin color probabilities, the classifier is easily able to cope with considerable illumination changes. Example hand segmentation is illustrated in Figure 6.

In this way, given the shape template P and the observed silhouette image z^{sil}, the likelihood function $p(z^{\text{sil}} \mid x)$ is defined from the ratio difference of overlapped areas, calculated from adaptive hand segmentation algorithm.

The skin color (or the luminance) in the testing images does not have to be similar to the skin color in the training images. This is because during online process we can learn the input color adaptively, so that the skin color probability will converge automatically. In case that the skin color (or the luminance) is suddenly changed, the segmentation result might become error prone in the beginning but as soon as several frames are learned, and the segmentation will be recovered. In other words, the training images are just the initial value (seed). When the real input hand enters the scene, the probability will converge automatically to the new values matching with the real hand.

2.3. Particle Filter Tracking. Particle filtering [13] is a useful tool to track objects in a clutter, with the advantage of performing automatic recovering from tracking failures. We apply particle filter to compute and track the hand. In our method, one particle represents each DOF of hand model. We determine the probability-density function by calculating from edge likelihood and silhouette likelihood, as explained in Section 2.2.1 and Section 2.2.2, respectively. The calculation is based on the following analysis.

Given that the process at each time-step is an iteration of factored sampling, the output of an iteration will be a weighted, time-stamped sample-set, denoted by $\{s_t^{(n)}, n = 1, \ldots, N\}$with weights $\pi_t^{(n)}$, representing approximately the probability-density function $p(X_t)$ at time t, where N is the size of sample sets, $s_t^{(n)}$ is defined as the position of the *nth* particle at time t, and X_t represents the position of hand model at time.

The iterative process can be divided into three main stages:

(i) selection stage,

(ii) predictive state,

(iii) measurement stage.

In the first stage (the selection stage), a sample $s_t'^{(n)}$ is chosen from the sample-set $\{s_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}\}$ with probabilities $\pi_{t-1}^{(j)}$, where $c_{t-1}^{(n)}$ is the cumulative weight. This is done by generating a uniformly distributed random number $r \in [0, 1]$. We find the smallest j for which $c_{t-1}^{(j)} \geq r$ using binary search, and then $s_t'^{(n)}$ can be set as follows: $s_t'^{(n)} = s_{t-1}^{(j)}$.

Each element chosen from the new set is now subjected to the second stage (the predictive step). We propagate each sample from the set s'_{t-1} by a propagation function, $g(s_t'^{(n)})$, using

$$s_t^{(n)} = g\left(s_t'^{(n)}\right) + \text{noise}, \tag{9}$$

where noise is given as a Gaussian distribution. The form of $g(s_t'^{(n)})$ is a propagation function we used. We have tried different propagation functions (e.g., constant velocity motion model and acceleration motion model), but our experimental results have revealed that constant velocity motion model and acceleration motion model do not give a significant improvement. A possible reason is that the motion of hand is usually changing directions while tracking. Therefore the calculated velocities or accelerations in previous frame do not give accurate prediction of the next

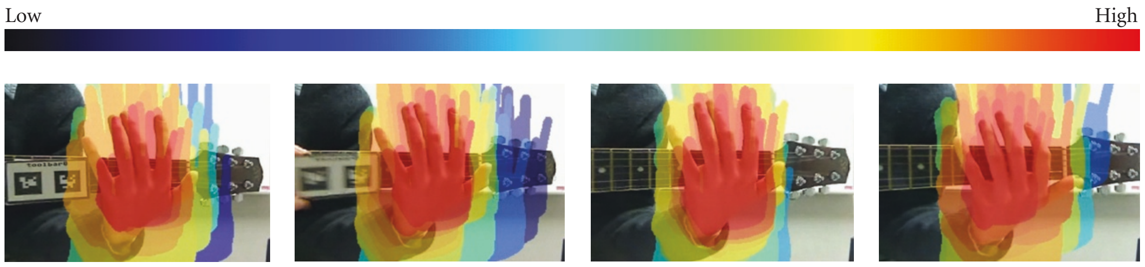


Figure 7: Particle filter behaviour in our system.

frame. In this way, we use only the noise information by defining $g(x) = x$ in (9).

In the last stage (the measurement stage), we generate weights from the probability-density function $p(X_t)$ to obtain the sample-set representation $\{(s_t^{(n)}, \pi_t^{(n)})\}$ of the state-density for time t using

$$\pi_t^{(n)} = p\left(X_t = s_t^{(n)}\right) = p(z \mid x), \tag{10}$$

where $p(z \mid x)$ is the joint likelihood of z, obtained from (4).

Next, we update the cumulative probability, which can be calculated from normalized weights using

$$c_t^{(0)} = 0, \qquad c_t^{(n)} = c_t^{(n-1)} + \pi_t^{(n)}{}_{\text{Total}}, \tag{11}$$

where $\pi_t^{(n)}{}_{\text{Total}}$ is the total weight and $n = 1, \dots, N$.

Once the N samples have been constructed, we estimate moments of the tracked hand model at time-step t as using

$$\varepsilon[f(X_t)] = \Sigma_{n=1}^{N} \pi_t^{(n)} s_t^{(n)}. \tag{12}$$

The hand can then be tracked, enabling us to perform automatic track recovering.

Figure 7 shows how the particle filter visually behaves in our system. The color in these hand model images illustrates the probabilities of the hand which is used in the particle filter step (left-to-right means low-to-high probability in color scale). For example, at each moment if the particle has a low normalized probability of being the correct hand pose by considering edge and silhouette likelihoods as described in Section 2.2, the color of the hand model shown will be black. In contrast, if the particle has high probability of the hand, the color will be red. The sum over all probabilities of every particle is 1. The scale of color used is also shown in Figure 7.

If the tracking certainty is lower than the threshold we set, it will return to the initial state again (as described in Particle Filter Tracking). This means that the particles will be initially distributed again for tracking. Thus, if the recent tracking results are not perfect, the hand can still be tracked.

3. Results

Representative results from our experiments are shown in this section. Figure 8 provides some representative results of the tracking experiments from a single camera with captured resolution 320 × 240 pixels. The reported experiment was acquired and processed on an Intel(R) Core (TM2) Duo CPU T7300 laptop computer running MS Windows at 2.0 GHz 778 MHz. One thousand particles are used for each experiment, that is, $N = 1000$. The execution time for these sequences is about 84 seconds per frame.

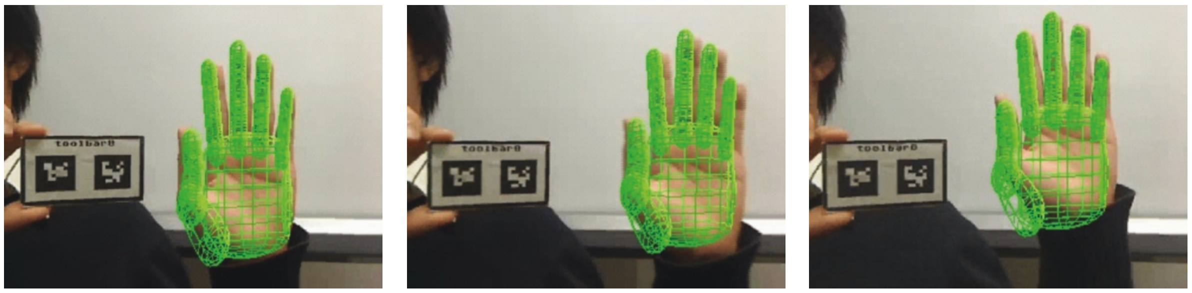
It is possible to use all 27 DOF in tracking, but it will take too long time to compute the result in each frame (a lot of particles are required for tracking in those high dimensions). Thus, in our experiments we limit the movement of the hand in the input video to a small number of DOF. Then, we track the hand movement in this reduced dimension.

Figures 8 (a) and (b) show the results of tracking global hand motion together with finger articulation. The images are shown with projected corresponding three-dimensional hand models (green color). For these sequences, the three-dimensional hand model has 11 DOF.

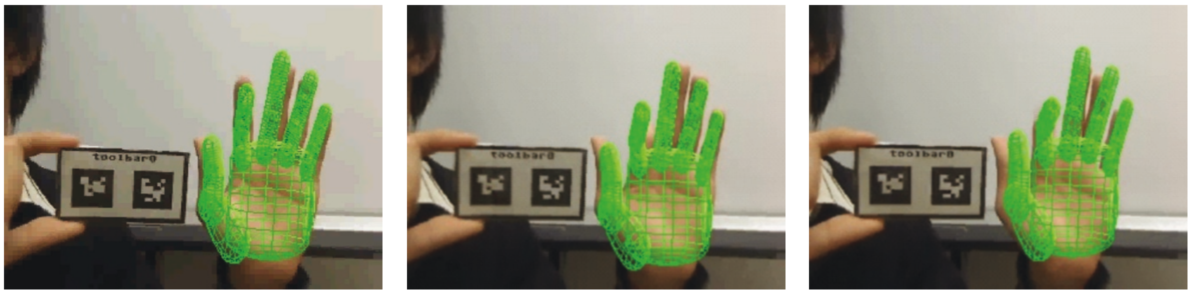
As seen in Figure 8 (a), at the commencement of the experiment, the hand starts moving from left-to-right and then moving back from right-to-left, respectively. It can be seen that the global hand motions are tracked successfully in this sequence.

Figure 8 (b) shows the result of hand tracking, when bending down the forefinger toward the palm. For this sequence, the range of global hand motion is restricted to a smaller region, but it still has 11 DOF. It can be observed that the proposed tracker successfully recovers these motions.

In the next experiment (Figure 9), a 13-DOF model is used to track the hand of guitarist, while holding the guitar. In this case, it is more challenging to track the hand because edge extraction of the frets and strings is more complicated than other normal backgrounds. Also the colors of frets and strings are quite similar to skin color. However, due to adaptive learning of the color probabilities from online input images as described in Section 2.2.2, it can robustly segment the hand region from the guitar neck background. This allows the hand to be accurately tracked. Figures 9(a) and 9(b) show the difference between without and with using adaptive leaning for tracking, respectively. It can be seen that when adaptive color learning is not used, the system cannot successfully track the hand, as shown in Figure 9(a), because of problem of background segmentation. In contrast, Figure 9(b) illustrates that when we apply adaptive learning of the color probabilities, the system is able to track the hand correctly.

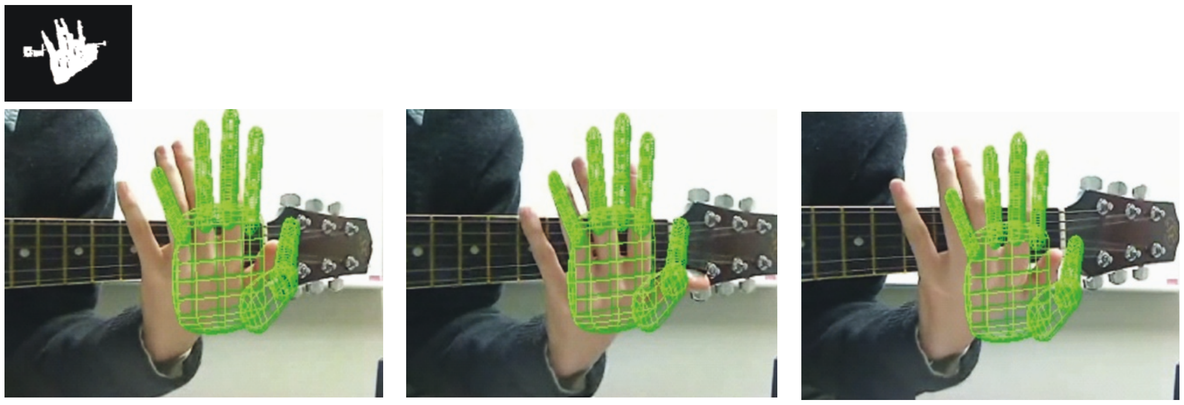


(a) Results of hand tracking, when moving the global position of hand (11-DOF model)

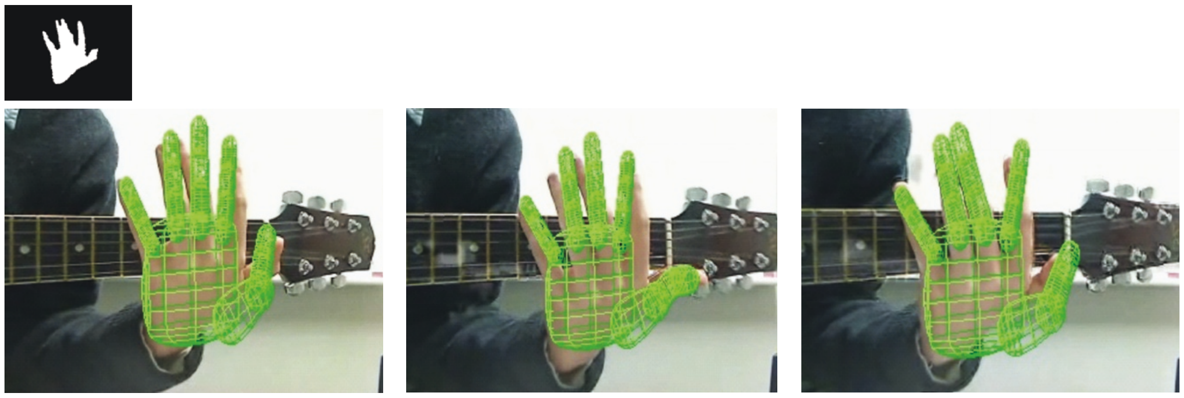


(b) Results of hand tracking, when bending the forefinger down toward the palm (11-DOF model)

Figure 8: Representative results of 3D model-based hand tracking.



(a) Results of hand tracking, while holding the guitar (13-DOF model) without adaptive color learning



(b) Results of hand tracking, while holding the guitar (13-DOF model) with adaptive color learning

Figure 9: Comparing results of 3D model-based hand tracking without/with adaptive color learning.

Table 1: Error performance in terms of fingertip localization measured against manually labeled ground truth with different numbers of particles.

Number of particles	Execution time (seconds per frame)	Mean distance errors (pixels)
750	69	11.78
1000	84	8.46
1250	126	6.18

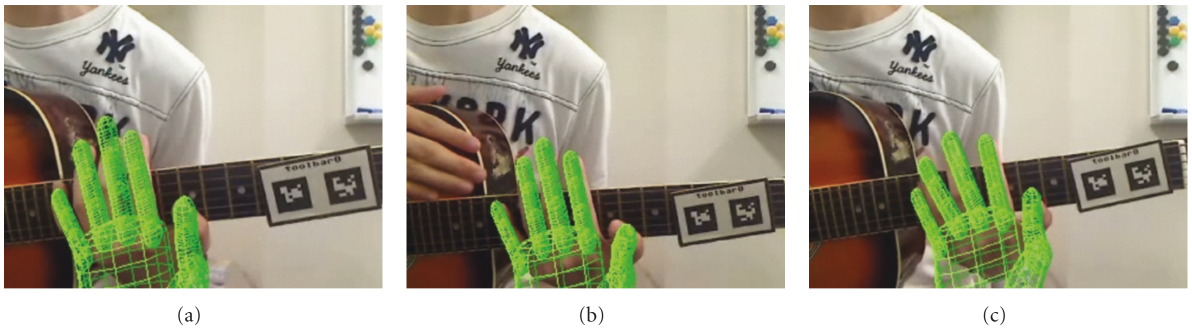

(a) (b) (c)

Figure 10: Results of hand tracking, when there is more than one hand or other skin areas rather than hand in the scene (13-DOF model).

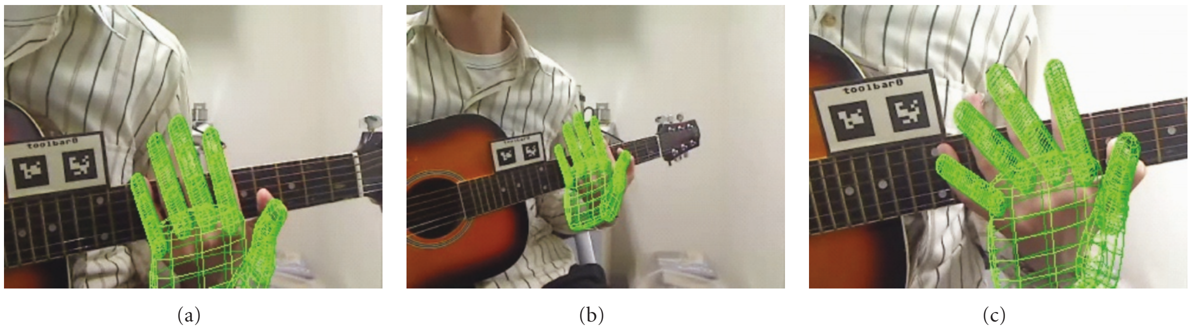

(a) (b) (c)

Figure 11: Results of hand tracking, when the scale of hand changes (13-DOF model).

Camera is calibrated, so that the intrinsic parameters are known before starting the system. Then the camera is registered to the world coordinate through ARTag (Augmented Reality Tag) [19]. Hence, when the projection matrix is known, we can track by using particle filter to register to the three-dimensional hand models onto the input images. It converges to the hand automatically by fitting model based on silhouette and edge clues using particle filter. Because ARTag's marker is placed to the guitar neck, the origin of the world coordinate is defined on the guitar neck. In this way, we know the projection matrix of camera at every frame, and so we can track the three-dimensional hand model using the proposed method.

In our method, we assume that the biggest region of a skin color found in the images is a hand blob. When we segment the skin area from the images, we remove small noise by size filtering. We determine that if noise is smaller than the threshold, we remove it. After that, we choose the biggest area as a hand blob. Therefore, our assumption is that the hand region has to be the largest area of skin found in the images. In this way, even though there is some other regions with color similar to skin color appear in the scene, our method can deal with. Similarly, if there is more than one hand or having faces in the images (but the hand of interest is the biggest skin area), the hand can still be tracked.

Figure 10 shows representative results when there is more than one hand or other skin area rather than hand in the image. As described, if there is more than one skin area in the image, our assumption is that the biggest area of a skin color is a hand blob. Thus, although there are other areas of skin color, we can differentiate them by deciding the size of appearing skin-color areas.

Figure 11 represents some results when changing the scale of hands. In the dimensions of DOF of the hand model, the X-axis, Y-axis, and Z-axis are all considered. Because we distribute randomly particles in Z-axis too, if the scale of hands changes (i.e., Z-axis is changed), our system can still track the hand correctly.

Because it is difficult to measure directly the ground truths of the parameters of every joint of 3D hand model, we evaluate the error performance by measuring the projection of fingertips in 2D. Table 1 shows the error performance in terms of fingertip localization of five fingers measured against manually labeled ground truth with different numbers of used particles. We randomly select 25 images from 200 images from the sequence to measure the error performance. We measure the distance errors by measuring the differences of the distance between the ground truth positions and the results of fingers tracking positions in each fingertip. We obtain the ground truth positions manually by human eyes to locate the correct positions of five fingertips, while the fingertips of tracked hand model are estimated by our system at each time. The distance errors in each fingertip are measured by Euclidean distances in pixels (320×240 total image size) with respect to the ground truth positions. After obtaining the distance errors from five fingertips, we calculate the mean of the distance errors, as shown in Table 1. The execution time in each of the number of used particles is also shown in seconds per frame. We believe that these errors are low enough to make the proposed algorithm presented in this paper a suitable method for guitarist's hand tracking.

4. Conclusions

In this paper, we have developed a system that tracks the hand by using a model-based approach. We construct the three-dimensional hand model by using a set of quadrics. After that, we utilize a quadratic chamfer distance function to calculate the edge likelihood, and then the online color learning adaptation is utilized. Following this, particle filter is performed to track the hand. This implementation can be used to further improve the hand tracking application of guitarist such as [20]. Although we believe that we can successfully produce accurate output from our system, the current system has the limitation with finger occlusion and guitar neck occlusion. This sometimes happens when playing the guitar in real life. Another limitation is about the high dimension of the state space (DOF). The number of particles required increases with the dimension of the state space. Therefore, some improvement or optimization should be considered to make the system faster. In the future, we intend to further refine these problems.

Acknowledgments

This work is supported in part by a Grant-in-Aid for the Global Center of Excellence for High-Level Global Cooperation for Leading-Edge Platform on Access Spaces from the Ministry of Education, Culture, Sport, Science, and Technology in Japan. The work presented in this paper is partially supported by CREST, JST (Research Area: Foundation of technology supporting the creation of digital media contents).

References

[1] T. Maki-Patola, J. Laitinen, A. Kanerva, and T. Takala, "Experiments with virtual reality instruments," in *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME '05)*, pp. 11–16, Vancouver, Canada, May 2005.

[2] F. Liarokapis, "Augmented reality scenarios for guitar learning," in *Proceedings of the International Conference on Eurographics UK Theory and Practice of Computer Graphics*, pp. 163–170, Canterbury, UK, 2005.

[3] Y. Motokawa and H. Saito, "Support system for guitar playing using augmented reality display," in *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '06)*, pp. 243–244, Santa Barbara, Calif, USA, October 2006.

[4] C. Kerdvibulvech and H. Saito, "Vision-based detection of guitar players' fingertips without markers," in *Proceedings of the IEEE International Conference on Computer Graphics, Imaging and Visualization (CGIV '07)*, pp. 419–428, Bangkok, Thailand, August 2007.

[5] A. Imai, N. Shimada, and Y. Shirai, "Hand posture estimation in complex backgrounds by considering mis-match of model," in *Proceedings of the Asian Conference on Computer Vision (ACCV '07)*, pp. 596–607, Tokyo, Japan, November 2007.

[6] Y. Iwai, Y. Yagi, and M. Yachida, "A system for 3D motion and position estimation of hand from monocular image sequence," in *Proceedings of the U.S.-Japan Graduate Student Forum in Robotics*, pp. 12–15, Osaka, Japan, November 1996.

[7] Y. Iwai, Y. Yagi, and M. Yachida, "Estimation of hand motion and position from monocular image sequence," in *Proceedings of the Asian Conference on Computer Vision (ACCV '95)*, pp. 230–234, Singapore, December 1995.

[8] B. Stenger, *Model-based hand tracking using a hierarchical bayesian filter*, Ph.D. thesis, University of Cambridge, Cambridge, UK, 2004.

[9] B. Stenger, A. Thayananthan, H. S. P. Torr, and R. Cipolla, "Model-based hand tracking using a hierarchical bayesian filter," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1372–1384, 2006.

[10] G. Borgefors, "Hierarchical chamfer matching: a parametric edge matching algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 849–865, 1988.

[11] A. A. Argyros and M. I. A. Lourakis, "Tracking skin-colored objects in real-time. Invited contribution to the "Cutting edge robotics book"," in *Advanced Robotic Systems International*, pp. 77–90, 2005.

[12] A. A. Argyros and M. I. A. Lourakis, "Tracking multiple colored blobs with a moving camera," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 2, San Diego, Calif, USA, June 2005.

[13] M. Isard and A. Blake, "Condensation—conditional density propagation for visual tracking," *International Journal on Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[14] M. De la Gorce, N. Paragos, and D. J. Fleet, "Model-based hand tracking with texture, shading and self-occlusions," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 1–8, Anchorage, Alaska, USA, June 2008.

[15] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, UK, 2004.

[16] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "A review on vision-based full DOF hand motion estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 3, p. 75, San Diego, Calif, USA, June 2005.

[17] R. Lockton and A. W. Fitzgibbon, "Real-time gesture recognition using deterministic boosting," in *Proceedings of the British Machine Vision Conference (BMVC '02)*, vol. 2, pp. 817–826, Cardiff, UK, September 2002.

[18] K. Jack, *Video Demystified: A Handbook for the Digital Engineer*, Elsevier Science and Technology Books, 4th edition, 2004.

[19] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR '05)*, pp. 590–596, San Diego, Calif, USA, June 2005.

[20] C. Kerdvibulvech and H. Saito, "Real-time guitar chord estimation by stereo cameras for supporting guitarists," in *Proceedings of the International Workshop on Advanced Image Technology (IWAIT '07)*, pp. 256–261, Bangkok, Thailand, January 2007.