Real-Time Guitar Chord Recognition System Using Stereo Cameras for Supporting Guitarists

Chutisant Kerdvibulvech[1] and **Hideo Saito**[2], Non-members

ABSTRACT

This paper proposes a novel approach for recognizing chords played on a guitar by using stereo cameras. This process is done in real time by recognizing the patterns of fingers' pushing positions detected from input images. The program detects the position of a player's fingers by using on-line adaptation of color probabilities and a Bayesian classifier. Therefore the system is able to cope easily with considerable illumination changes and a dynamic background. ARTag is utilized to calculate the extrinsic parameters as an online process, so that the system can recognize the correct chord, even if the guitar is moving. By utilizing a triangulation method on stereo cameras, the 3D positions of fingers are recognized when a guitar string is pressed. It used the advantages found in PCA (Principal Component Analysis) to efficiently deal with complex chord classification. The experimental results have revealed that the proposed system is effective in real time also.

Keywords: Guitar Chord Recognition, Bayesian Classifier, Augmented Reality Tag, Principal Component Analysis

1. INTRODUCTION

Learning to play the guitar usually involves tedious lessons in the correct fingering positions for the left hand (assuming the guitarist is right-handed). It is sometimes difficult for beginners to recognize whether they are accurately positioning their fingers on the strings to make the correct guitar chords.

In this paper, we propose a guitar teaching aid that will assist guitar players by using computer vision and pattern recognition technologies. This system detects the players' fingers in real time and determines the guitar chord which the player's left hand is using. To calculate the positions of fingers, colored finger markers are detected using a Bayesian classifier that is bootstrapped with a small set of training data and refined through an off-line iterative training procedure [1] [2]. By using an on-line adaptation of finger markers-color probabilities the classifier is able to effectively cope with considerable illumination changes, and therefore it is able to robustly track colored markers even when there are cluttered and dynamic background conditions. We utilize ARTag (Augmented Reality Tag) [3] to compute the extrinsic parameters for estimating the guitar position. Then we calibrate the cameras for calculating the projection matrix at online process. To assist the recognition of correct chords while the guitar is moving, we define the world coordinate on the guitar neck as the guitar coordinate system. We utilize a triangulation [4] technique to estimate 3D position of the player's fingers using stereo cameras, and then recognize if a guitar string is being pressed or not. Next, we apply PCA (Principal Component Analysis) [5] to reduce the input dimension which allows each guitar chord to be classified more accurately. As a result, players can recognize the guitar chords they are playing during the song in real time.

Obviously it would be of great assistance to learners if they are able to recognize the chord being played by the left hand. The system improves accuracy because it can identify whether the finger positions are correct according to the finger positions required for the piece of music they are playing (i.e., the guitar teacher inputs the correct fingering for each guitar chord and these are then incorporated into the system that we have provided). Applying the proposed method, beginners can automatically identify whether their fingers are in the correct position which makes this a guitar application. The proposed teaching system would be invaluable to teachers and have a wide application to supporting people learning to play the guitar.

2. RELATED WORK

Research about guitars is one of the popular topics in the field of computer vision for musical applications. We found a number of valuable research papers that focus on assisting guitar players with computer vision driven applications.

Maki-Patola et al. [6] proposed a system called VAG (Virtual Air Guitar) using computer vision. Their aim is to create a virtual air guitar which does not require a real guitar (e.g., by using only a pair of colored gloves), but can produce music as similar as the player is playing the real guitar.

Manuscript received on February 28, 2007 ; revised on June 30, 2007.

[1,2] The authors are with Department of Information and Computer Science, Keio University 3-4-1 Hiyoshi, Kohoku-ku, Yokohama, 223-8522, JAPAN, Emails: chutisant@ozawa.ics.keio.ac.jp and saito@ozawa.ics.keio.ac.jp

Liarokapis [7] proposed an augmented reality system for a guitar learner. The aim of this work is to show the augmentation (e.g., the positions where the learner should place their fingers to play the chord) on an electric guitar to guide the player as a self-learning system.

Motokawa and Saito [8] built a system called Online Guitar Tracking that supports a guitarist using augmented reality. This is done by showing a virtual model of the fingers on a stringed guitar as an aid to learning to play the guitar.

In these above systems, they do not aim to recognize the chord which a player is playing. Their systems cannot determine whether the player is playing chord correctly or not. A similar research goal as ours can be found in [9]. Their goal is to develop an electric bass guitar system to evaluate if the user is pressing his/her forefinger on the correct fret of the bass guitar. However, their system has a constraint that only one finger is used for playing. Our aim is to propose a system for helping the player to play the ordinary guitar (6 strings) that uses all four fingers for playing.

There has been a lot of work published focusing on the retrieval of pianist fingering using computer vision (e.g., a fingering detection system for pianists [10], a Handel system [11], a video recognition tool for supporting pianists [12], etc).

However, in the case of retrieval of guitarist fingering as an aid to recognize the chords using computer vision, the situation is different. Guitar chord recognition using fingering retrieval information is more challenging than detection on piano keys. The first reason is because self-finger occlusion usually occurs while playing the guitar and this rarely happens when playing the piano. Secondly, it is more difficult to detect the guitar chord because the guitar neck is always moving whereas both the piano and the cameras used in this system are fixed to the ground.

Basically, to retrieve the guitarist's fingering, fingering information must be deduced at several points in the music production process as described in [13]. Three main strategies are: Pre-processing using score analysis; Real-time using Midi guitars; Post-processing using sound analysis.

Radicioni et al. [14] retrieve fingering information through score analysis. The score is fragmented in phrases, and the optimum fingering for each phrase is determined by finding the shortest path in an acyclic graph of all possible fingering positions. The problem with this approach is that it cannot account for all the factors influencing the choice of a specific fingering, namely philological analysis, physical constraints due to the musical instrument, and biomechanical constraints in the musician-instrument interaction.

Other systems retrieve the fingering while (or after) a musician is playing a piece. One of these approaches uses a Midi guitar. Theoretically, a Midi guitar with separate Midi channel assigned to each string provides real-time pitch recognition and thus determines fret position. However, Midi guitar users report several problems, including a variation in the recognition time from one string to another and the necessity to adapt their playing technique to avoid glitches or false note triggers [15]. Moreover, a Midi guitar will appear as a keyboard guitar, which does not have any real strings. Therefore, it cannot be applied to recognize chords played on a real string guitar.

A third approach using guitar timbre provides chords fingering. Traube [16] used a method relying on the recording of a guitarist. However, the main drawback of this method concerns real-time application.

More recently, Burns and Wanderley [13] presented a real-time prototype system to visually recognize fingering gestures for retrieval of guitarist fingering using computer vision algorithm. This system affixes the camera to the guitar neck, and thereby eliminates the motion of the neck caused by ancillary gestures. However, a limitation of this system is that it is sometimes difficult to fix the camera mount onto the guitar neck. Also, in this system, they assume that the fingertip shape can be approximated with a semicircular shape. However, this assumption is sometimes difficult to use because the shapes of fingertips in some positions do not resemble semicircular shapes. Furthermore, we can note another limitation concerning the difficulty to recognize whether a guitar string is pressed by the player's finger because they use only one camera.

This paper presents a novel system for real-time retrieval of the fingering information needed to recognize a guitar chord from a guitarist playing a musical excerpt using computer vision and pattern recognition. This system tracks the players' fingers and then calculates the guitar chord played by the player's left hand. By applying the proposed method, as an online process, it estimates whether a player is holding the guitar correctly and is therefore a valuable guitar teaching aid.

3. PROPOSED SYSTEM

3.1 System Configuration

Fig. 1 illustrates the proposed system configuration. The system is composed of two USB cameras and a display connected to the PC for the guitar players. The two USB cameras capture the left player's fingers positioning and the guitar neck from two different positions.

The important aspect in constructing this guitar teaching aid was developing accurate 3D positions of the guitar and the fingers. We attach a 4.5cm $\times$ 8cm ARTag fiducial marker onto the top right corner of guitar neck and use it to compute the position of the guitar. To make the system robust enough, we use

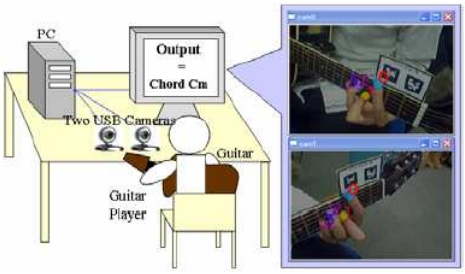


Fig.1: *Proposed System Configuration*

four color markers to calculate the pose of the fingers which are attached to the four fingertips: forefinger, middle finger, ring finger and little finger. Utilizing this method, we can implement a practical system for recognizing the guitar chords with just two USB cameras.

3.2 System Overview

Fig. 2 shows schematically the outline of the method. After grabbing the images, we firstly apply a Bayesian classifier and an on-line adaptation of color probabilities for finger markers segmentation. At the same time, we calculate the projection matrix in each frame by utilizing ARTag. As the next step, we apply the triangulation to calculate the 3D positions of the guitar and fingers. Then, we calculate the guitar chords which the guitar player is playing by using PCA.

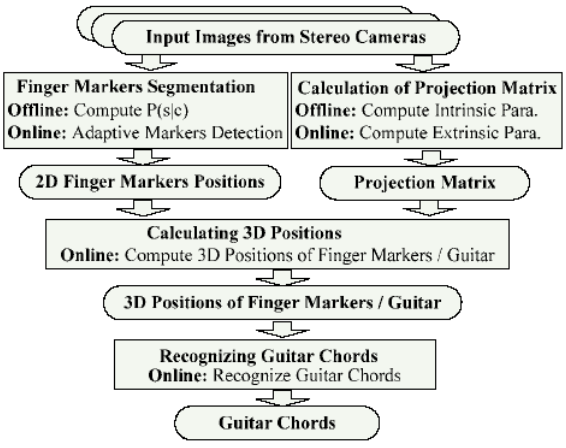


Fig.2: *System Overview*

3.3 Finger Markers Segmentation

We describe in this section a method that tracks the player's fingers (i.e., the left hand) and estimates the 2D fingers positions. To track the fingering required to recognize a guitar chord, it is important to detect the correct fingertips positions.

Recently, many methods in computer vision for fingertip detection have been proposed.

However, when applying the methods proposed previously, problems usually arise in retrieving the guitar fingering. The first reason is that the lack of contrast between fingertips and background skin adds complication (i.e., when the fingertips are in front of the palm). The next reason is that, while playing the guitar, the fingers are not stretched out separately, so it is difficult to detect the fingertips correctly. Moreover, to detect the fingertips while playing the guitar, the background is usually dynamic and non-uniform (e.g., guitar neck and natural scene) which makes it difficult to locate the fingertip positions. These conditions often occur when playing the guitar in a real environment.

The existing method (used in [17] [13]) for retrieval of guitarist fingering is to use the circular Hough transform [18] to detect fingertips, whose ends can be approximated with a semicircular shape. It takes advantage of the quasi-circular shape of the fingers while the rest of the hand is roughly straight. An edge image is obtained by applying a Canny edge detector [19] on the silhouette images. In our system, we attempted to utilize the circular Hough transform to detect the fingertips when the players are playing the guitar. Unfortunately, our experiments have revealed that we could not detect fingertips accurately enough - even when carefully changing the radius of a circle in the Hough transform step. This is because the fingertip shape does not appear as the circular shape in some views. For this reason, this assumption is sometimes difficult to use practically.

In this way, though it is possible to detect the positions of fingertips without markers, the detection accuracy is not as high as we required. For this reason, to recognize the chord output robustly enough, we decided to utilize four colored markers which are placed on the four fingertips (because the aim of this paper is not to emphasize on how to locate the fingertip positions, but to focus on the overall view of guitar system which can make this a guitar application). We detect these four fingertip markers from background using color detection algorithm, but at the same time a well known problem of colored marker tracking is the control of lighting. Changing levels of light and limited contrasts disable correct registration, especially in the case of a cluttered background.

A survey [20] provides an interesting overview of color detection. A major decision towards deriving a model of color relates to the selection of the color space to be employed. Once a suitable color space has been selected, one of the commonly used approaches for defining what constitutes color is to employ bounds on the coordinates of the selected space.

In our previous work [21] we used this approach (i.e., HSV model) to differentiate the four color markers from the background by converting the RGB picture into the HSV space and finding the proper threshold values in each marker. However, by using

this technique, it is difficult to deal with illumination changes and a cluttered background.

Therefore, we decided to replace the finger markers detection method from the previous approach with a Bayesian classifier that is bootstrapped with a small set of training data and refined through an off-line iterative training procedure (proposed recently in [1] [2]). During an off-line phase, a small set of training input images is selected on which a human operator manually delineates markers-colored regions. The color representation used in this process is YUV 4:2:2 [22].

Following this, assuming that image pixels with coordinates (x, y) have color values $c = c(x, y)$, training data are used to calculate (**i**) the prior probability $P(s)$ of marker color, (**ii**) the prior probability $P(c)$ of the occurrence of each color and (**iii**) the prior probability $P(c|s)$ of a marker being color c. By employing Bayes' rule, the probability $P(s|c)$ of a color c being a marker color can be computed as described in Equation (1).

$$P(s|c) = P(c|s)P(s)/P(c) \tag{1}$$

Equation (1) determines the probability of a certain image pixel being marker-colored using a lookup table indexed with the pixel's color. All pixels with probability $P(s|c) > T_{max}$ are considered as being marker-colored. These pixels constitute seeds of potential marker-colored blobs. Also, image pixels with probabilities $P(s|c) > T_{min}$ where $T_{min} < T_{max}$ that are immediate neighbors of marker-colored image pixels are recursively added to each blob. The rationale behind this region growing operation is that an image pixel with relatively low probability of being marker-colored should be considered as such in the case that it is a neighbor of an image pixel with high probability of being marker-colored. Indicative values for the thresholds T_{max} and T_{min} are 0.5 and 0.15, respectively. A standard connected components labeling algorithm 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 marker-colored regions. Each of the remaining connected components corresponds to a marker-colored blob.

The success of the marker-color detection depends crucially on whether illumination conditions during the on-line operation of the detector are similar to those during the acquisition of the training data set. Despite the fact that the UV color representation model used has certain illumination independent characteristics, the marker-color detector may produce poor results if the illumination conditions during on-line operation are considerably different compared to the ones represented in the training set. Thus, a means for adapting the representation of marker-colored image pixels according to the recent history of detected colored pixels is required. To solve this problem, marker color detection maintains two sets of prior probabilities. The first set consists of $P(s), P(c), P(c|s)$ that have been computed off-line from the training set while the second is made up of $P_W(s), P_W(c), P_W(s|c)$, corresponding to the evidence that the system gathers during the W most recent frames. Clearly, the second set better reflects the "recent" appearance of marker-colored objects and is therefore better adapted to the current illumination conditions. Marker color detection is then performed based on the following weighted moving average formula:

$$P(s|c) = \gamma P(s|c) + (1 - \gamma)P_W(s|c) \tag{2}$$

where $P(s|c)$ and $P_W(s|c)$ are both given by Equation (1) but involve prior probabilities that have been computed from the whole training set and from the detection results in the last W frames, respectively. In Equation (2), γ is a sensitivity parameter that controls the influence of the training set in the detection process. By using on-line adaptation of finger markers-color probabilities the classifier is able to cope with considerable illumination changes effectively, and also it is able to robustly track colored markers even in the case of a complex cluttered and dynamic backgrounds. Though this mechanism is implemented to track four colored markers, experimental evaluation has indicated that processing is performed fast enough to keep up with completely real-time acquisition.

3.4 Calculation of Projection Matrix

In this section, we describe the method we used to calculate the projection matrix for estimating the position of the guitar. Because the guitar neck always moves while playing guitar, it is difficult to recognize the guitar chord accurately in real time.

Burns and Wanderley [13] attempted to solve this problem by affixing the camera mount on the guitar neck. Therefore this method can eliminate the motion of the neck caused by ancillary gesture because both the guitar neck and the camera are fixed together. In practice however, it is sometimes difficult to affix the camera mount on the guitar neck for playing guitar in real life.

In recent years, several guitar systems (an electric bass guitar system [9], an augmented guitar learning system [7] and *Online Guitar Tracking* [8]) attempted to use an ARToolkit (Augmented Reality Toolkit) [23]'s marker to estimate the pose of the guitar. Even though ARToolkit is a powerful tool for tracking the guitar position, it cannot effectively cope with the large occlusion, a high rate of false positive detections and lighting variation problems. Therefore, we have decided to utilize ARTag's marker to track the guitar position. ARTag uses a different method for

identifying the borders that does not need a threshold that is much more robust to lighting variation, occlusion - plus robust digital communication algorithms are used so it never gets false detections or confusion between markers (see details of comparing between ARTag and ARToolkit in [24]). Hence, in our system, the correct guitar position can successfully be tracked even despite large ARTag occlusions and lighting variation conditions.

Because the guitar neck is not fixed to the ground while the cameras are fixed, the projection matrix changed at every frame. In this way, it is important to compute the projection matrix relative to the world coordinate that is attached to the guitar neck at online process. The world coordinate is defined on the top right corner of guitar neck as a guitar coordinate system.

$$P = \begin{bmatrix} \alpha_u & -\alpha_u \cot\theta & u_0 \\ 0 & \alpha_v / \sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{21} & R_{31} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \end{bmatrix} \quad (3)$$

In the camera calibration process, the relation by projection matrix is generally employed as the method of describing the relation between the 3D space and the images. The important camera properties, namely the intrinsic parameters that must be measured, include the center point of the camera image, the lens distortion and the camera focal length. We compute the intrinsic parameters during preprocess the first time. The matrices R and t in the camera calibration matrix describe the position and orientation of the camera with respect to world coordinate system. The extrinsic parameters include three parameters for the rotation, and another three for the translation. Using the online process, the ARTag algorithm computes the extrinsic parameters in every frame, and therefore we can compute in real time the projection matrix from both the intrinsic parameters and the extrinsic parameters as shown in Equation (3). For this reason, because we calibrate the cameras at online process, we can track the correct guitar's position effectively even in the case of obvious changing guitar position.

3.5 Calculating 3D Positions

In this section, we utilize triangulation to compute 3D position of fingers in the guitar coordinate system. Recently, Burns and Wanderley [13] attempted to recognize chords played on a guitar by using a single camera in 2D space. Nonetheless, using only one camera, it is very difficult to recognize whether a guitar string is pressed by fingers.

Therefore, in our proposed system, we utilize triangulation to compute 3D position of fingers using stereo cameras to cope with this problem. Fig. 3 depicts the triangulation method which demonstrates

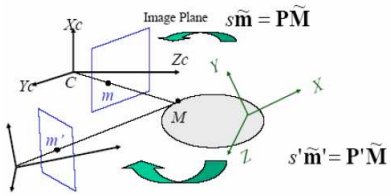


Fig.3: *Triangulation Method*

how to estimate 3D positions from two cameras using the projection matrix results. In this way, using the data gathered from the 2D finger detection and the projection matrix results, we can estimate the 3D position of the finger markers in the guitar coordinate system. Because it is able to estimate the 3D position of fingers at online process, we can successfully recognize if the player is pressing a string in real time.

3.6 Recognizing Guitar Chords

In this section, we propose two methods to compute the guitar chords from the guitar and 3D fingers tracking results. Although we can identify the guitar position and fingers positions, it remains difficult to accurately recognize chords played on a guitar because, in some chords, the four fingering positions are very close to each other.

The first method is called the Interval Threshold Technique. At every finger marker, this method selects both of the minimum value and the maximum value of the 3D positions, and then utilizes these minimum and maximum values as the interval threshold positions of each fingers marker in each chord. However, our experimental results have revealed that the Interval Threshold Technique is not enough accurate to recognize if the four finger positions are not obviously different, especially in some chords which two fingers play at the same fret and also in some paired chords which the four fingers positions are very similar.

For this reason, we propose the second method by applying PCA's advantage to classify chords played on a guitar in more accurate ways. Since patterns in data (i.e., the complexity of four fingers positions in each guitar chord) can be hard to find in data of high dimension, PCA is a good tool for analyzing data in each chord.

With a PCA process, we reduce the dimensionality of the data set by transforming to a new set of variables (the principal components) to summarize the features of the data. During a training phase, we collect the four fingers positions in x-axis, y-axis and z-axis of each guitar chord by playing a sample of the notes by the guitar teacher. In other words, the guitar teacher inputs the correct fingering information for each chord (using 300 training data sets in each chord). Each data set composes of twelve dimensions (i.e., the four fingers positions in x-axis, y-axis and

z-axis). Then, after the eigenvectors and eigenvalues are computed from the four fingers' positions in each axis, we construct a feature vector by taking the eigenvectors and forming a matrix as shown in Equation (4). Next, we derive the new data set from this feature vector. We utilize Cattell's Scree Test [25] to determine the number of factors to be retained from a factor analysis. The idea of the Scree test is the number of retained dimensions corresponds to the number of eigenvalues preceding the Scree. Therefore, we plot the eigenvalues as a line plot (the horizontal axis represents the number of eigenvalues, while the vertical axis is the eigenvalues), and then we find the place where the smooth decrease of eigenvalues appears to level off. According to this test, six dimensions (from twelve dimensions) are retained in our case. Also, to optimize the quality of chord classification, isolating outliers is an important step in preparing a data set for classifying the guitar chords. Hence, we remove the outliers (i.e., use Mild outliers) to improve the quality of training data sets and the impact of outlying values.

$$FeatureVector = (eig_1, eig_2, eig_3, \ldots, eig_n) \quad (4)$$

During a testing phase, we compute the distance between the unknown pattern and the desired set of known patterns and determine which known pattern is closest to the unknown. Finally, the unknown pattern is placed under the known pattern to which it has minimum distance. This is called Minimum Distance Classification. In Minimum Distance Classification process, two types of distance-based classifiers can be considered depending upon sample statistics: The first-order statistics classifiers (e.g., Euclidean distance) and second-order statistics classifiers (e.g., Mahalanobis classifier). In the case of guitar chords classification, our experimental results have examined that the Euclidean distance can classify the guitar chords more accurate than the Mahalanobis distance because of the data distribution format (i.e., four fingers 3D positions). For this reason, we decided to utilize the Euclidean distance for classifying the guitar chords as represented in Equation (5).

$$ED(x, m_j) = (x - m_j)^T (x - m_j) = \sum_{l=1}^{L} (x_l - m_{jl})^2 \quad (5)$$

Therefore, we calculate the Euclidean distance of the retained dimensions (i.e., six dimensions per chord) and then determine which known pattern is closest to the unknown. Then, we experimentally select the threshold value of the Euclidean distance. If the closest distance is less than this threshold value, the system would consider as the guitar chord is being played.

In this way, if the guitar player does not play any chord (e.g., the strings are not depressed but simply held over the correct strings, etc), the system is able to determine as no chord is being played (however, the results depend on this threshold value of the Euclidean distance we set). As a result, once the players have played the guitar, our system can immediately compute which the guitar chord is being played.

3.7 Bar Chord Problem

In a real guitar sequence, there are some guitar chords which require the guitarists to push all six strings down with the one finger (the so called bar chord). This usually only involves the forefinger. In our proposed method, we utilize only four color markers which are placed on the four fingertips, so that it could be impossible to identify whether the forefinger pushed all six strings down. We called this the Bar Chord Problem. However, we do not have to solve this problem directly because there are no two guitar chord in all the 144 guitar chords (e.g., see [26]) when the three fingers (the middle finger, the ring finger and the little finger) push exactly the same frets and strings, however the forefinger from first chord pushes all six strings down while the forefinger from second chord pushes the E string (i.e., the top string) on the same fret as the first chord. As a result, there are no two chords which we cannot differentiate, and therefore the information of the four fingertips is sufficient to classify each chord.

4. RESULTS

In order to show the efficiency of our method, we evaluate stability and accuracy of our system by testing 15 samples guitar chords (i.e., 11 general chords and 4 bar chords) in various different conditions, some of which are challenging. The reported experiment was acquired and processed on-line and in real-time on a Pentium M laptop computer running MS Windows at 1.6 GHz with 1 GB RAM, and two USB cameras with resolution 320 $\times$ 240 have been used for capture.

4.1 Robustness

Fig. 4 shows the scene that a guitar player uses our system in various conditions. The cam0 and cam1 windows depict the input images which are captured from two USB cameras. These two USB cameras capture the player's fingers in the left hand positioning and the guitar neck from two different views. The guitar player is holding the input chord, and then the four color numbers in each fingertip depict four 2D tracking results from the fingers' markers (forefinger [number0], middle finger [number1], ring finger [number2] and little finger [number3] ordered from right to left accordingly). The 3D reconstruction window, which is drawn by OpenGL, represents both of the detected 3D positions of the four finger markers and the guitar chord output. In 3D space, the four dif-

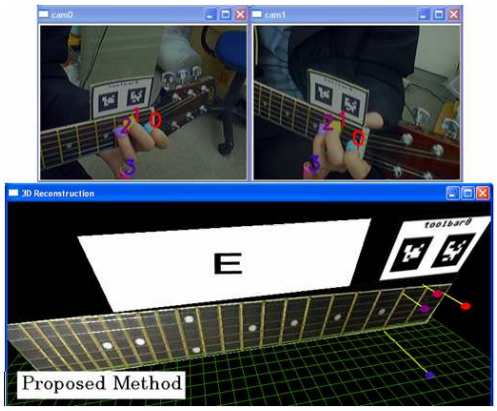
ferently color 3D cubes (i.e., forefinger [bright red], middle finger [crimson], ring finger [violet] and little finger [blue] ordered from right to left respectively) show each 3D detected result of the finger markers, and then the guitar chord output represents the recognized chord in real time.

Fig. 4(a) represents the general condition where our system can successfully recognize the guitar chord E output which is the same input chord as the player is holding. Next, as seen in Fig. 4(b), the system can cope with Bar Chord Problem because the four fingertips information is adequate to classify all guitar chords, and therefore we can estimate the guitar chord Cm (Bar Chord) accurately which is the same input chord.

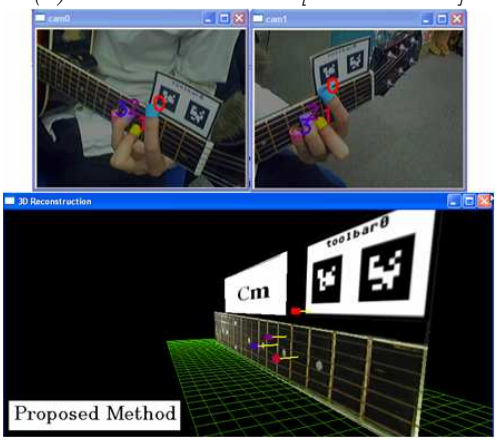
Moreover, in poor lighting conditions, by finding the threshold values in HSV space for detecting the fingertip's markers in the Finger Markers Segmentation step, it is not able to locate the player's fingertips correctly, while the Bayesian classifier can robustly track the fingertips in more effective ways. The system has been comparably tested under considerable changes in ambient lighting conditions (by turning lights off) using the earlier method (HSV model) and the proposed method (Bayesian classifier) as shown in Fig. 4(c) and 4(d) respectively. By using thresholds in HSV model, the number of color detection errors significantly increased during changing lighting condition as seen in Fig. 4(c). As a result, this previous method cannot recognize the chord properly (i.e., the input chords is Em, while the recognized chord output is chord F). However, as shown in Fig. 4(d), our proposed system can successfully recognize the correct guitar chord Em which is the same input chord under varying illumination condition.

Besides, the self-finger occlusion problem usually occurs in some commonly used chords (e.g., chord D7, A, Am, E, G, etc.), and at the same time fingertips (i.e., especially little fingertip) on the background palm condition frequently happen when capturing from two cameras. Fortunately, the system works well under the self-finger occlusion condition and also fingertips on the background palm condition because we use finger markers. As seen in Fig. 4(e), the middle fingertip, the ring fingertip and the little fingertip in window cam1 are partially occluded by each other, while the fingertip of little finger is located above the palm in the background, as seen in window cam0. The system can successfully continue to recognize the accurate output of a D7 chord as the same as that which the player is holding.

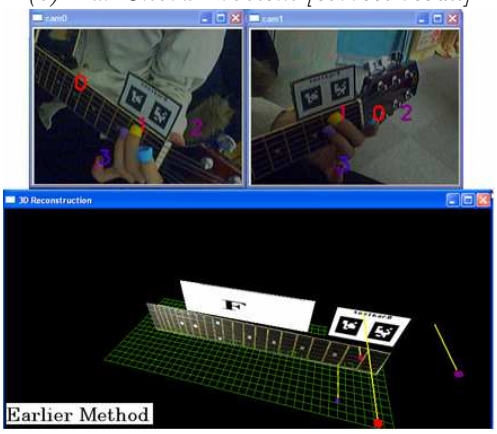
Furthermore, the system has been tested under challenging conditions from a sequence containing a dynamic background with a person walking around as depicted in Fig. 4(f). Because the system is robust enough for the dynamic background condition, the system correctly recognizes the chord F which is the same input chord as the player is producing.



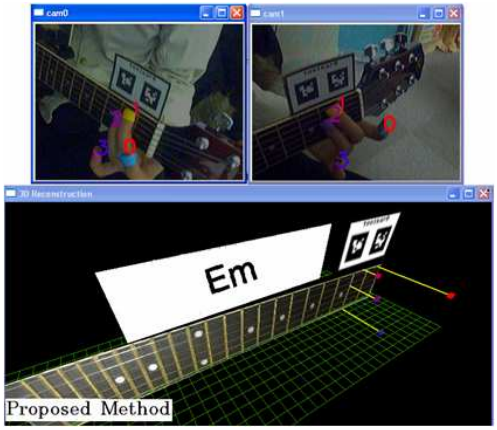
(a) General Condition [correct result]



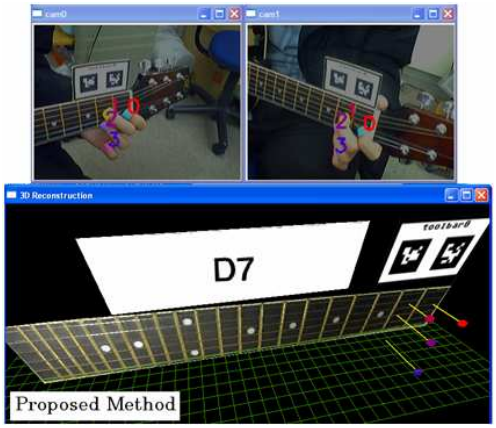
(b) Bar Chord Problem [correct result]



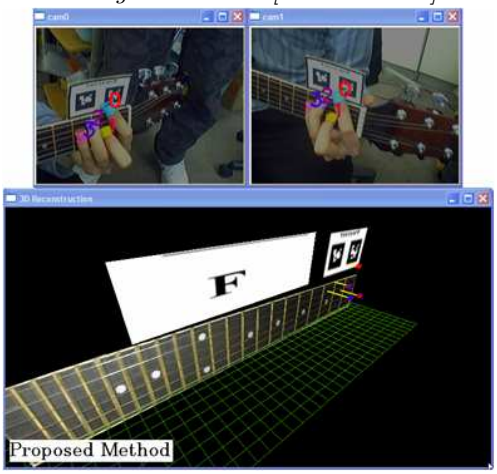
(c) Changing Lighting: Earlier Method [wrong result]



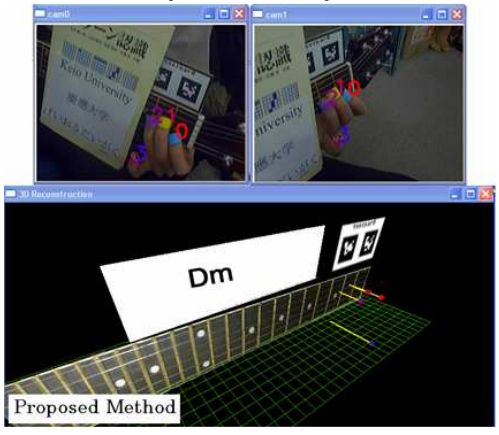
(d) Changing Lighting: Proposed Method[correct result]



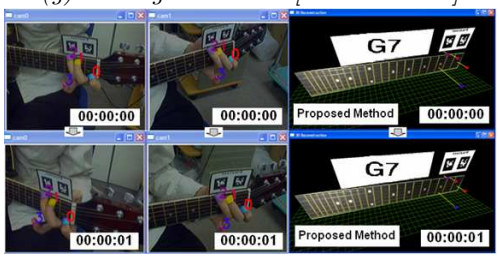
(e) Self-finger Occlusion and Fingertips on the Background Palm [correct result]



(f) Dynamic Background: person walking nearby [correct result]



(g) ARTag Occlusion [correct result]



(h) Moving Guitar [correct result]

Fig.4: *Experimental Results*

The system has also been tested under ARTag occlusion conditions. Fig. 4(g) shows how the guitar chord Dm can be precisely recognized even despite a marker occlusion. The main reason why the system can cope with ARTag occlusion effectively is because the ARTag algorithm uses digital coding theory to get a very low false positive, employing an edge linking method to give occlusion immunity.

The system also works well under situations where the player moves the guitar. As illustrated in image sequence in Fig. 4(h), because we define the world coordinate on the guitar neck (i.e., on the ARTag's marker), even though the guitar is moving, the detected 3D positions of the four finger markers from different guitar positions (i.e., but the same input chord) are closely the same positions. As a result, the system can produce the accurate guitar chord G7 output, although the player is moving the guitar position.

Indeed, the proposed method performs well in all the above cases, some of which are challenging. The experimental results revealed that the proposed system is robust in varied conditions even when used in real time.

4.2 Accuracy

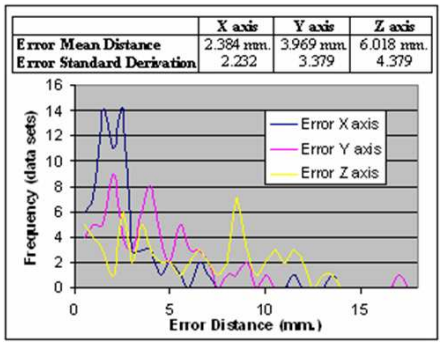


Fig.5: *Accuracy of Detected 3D Finger Positions*

Fig. 5 shows the accuracy of our experimental results when detecting finger positions in 3D. With respect to the manually measured ground truth positions, the mean error in distance on the x-axis is 2.384mm and on the y-axis it is 3.969mm. These errors in distance are small and reasonably acceptable if we compare them with the distances of each fret and string respectively (e.g. the distance in the first fret is 27mm; the distance between the first and second strings is 8mm). However, on the z-axis, the mean error distance is 6.018mm, which is slightly large. The z-axis (perpendicular to the guitar board) is used to recognize if any guitar strings are pressed by fingers. For this reason, our system is limited: the distance between the fingers and the strings must be great enough in the z-direction whenever the fingers are not pressing. If not, the system cannot recognize whether the string is pressed or not.

Table 1: *Accuracy of Chords Recognition Rates Using Interval Threshold Technique*

		ACTUAL (data sets)														
		A	Am	C	D	D7	Dm	E	Em	G	G7	F#m	F	Cadd9	Cm	F#
RECOGNITION	A	250	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Am	-	240	-	-	-	13	-	-	-	-	-	-	-	-	-
	C	-	-	241	-	-	-	-	-	-	4	-	-	23	-	-
	D	-	-	-	250	-	-	-	-	-	-	-	-	-	-	-
	D7	-	-	-	-	250	-	-	-	-	-	-	-	-	-	-
	Dm	-	-	-	-	-	237	-	-	-	-	-	-	-	-	-
	E	-	10	-	-	-	-	250	-	-	-	-	-	-	-	-
	Em	-	-	-	-	-	-	-	250	-	-	-	-	-	-	-
	G	-	-	-	-	-	-	-	-	249	-	-	-	-	-	-
	G7	-	-	9	-	-	-	-	-	-	246	-	-	-	-	-
	F#m	-	-	-	-	-	-	-	-	-	-	250	-	-	-	-
	F	-	-	-	-	-	-	-	-	-	-	-	250	-	-	-
	Cadd9	-	-	-	-	-	-	-	-	1	-	-	-	227	-	-
	Cm	-	-	-	-	-	-	-	-	-	-	-	-	-	250	-
	F#	-	-	-	-	-	-	-	-	-	-	-	-	-	-	250
Total		250	250	250	250	250	250	250	250	250	250	250	250	250	250	250
Accuracy		100 %	96.0 %	96.4 %	100 %	100 %	94.8 %	100 %	100 %	99.6 %	98.4 %	100 %	100 %	90.8 %	100 %	100 %
Mean Accuracy = 3690 / 3750 = 98.4 %																

Table 2: *Accuracy of Chords Recognition Rates Using PCA Method*

		ACTUAL (data sets)														
		A	Am	C	D	D7	Dm	E	Em	G	G7	F#m	F	Cadd9	Cm	F#
RECOGNITION	A	250	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Am	-	240	-	-	-	13	-	-	-	-	-	-	-	-	-
	C	-	-	241	-	-	-	-	-	-	4	-	-	23	-	-
	D	-	-	-	250	-	-	-	-	-	-	-	-	-	-	-
	D7	-	-	-	-	250	-	-	-	-	-	-	-	-	-	-
	Dm	-	-	-	-	-	237	-	-	-	-	-	-	-	-	-
	E	-	10	-	-	-	-	250	-	-	-	-	-	-	-	-
	Em	-	-	-	-	-	-	-	250	-	-	-	-	-	-	-
	G	-	-	-	-	-	-	-	-	249	-	-	-	-	-	-
	G7	-	-	9	-	-	-	-	-	-	246	-	-	-	-	-
	F#m	-	-	-	-	-	-	-	-	-	-	250	-	-	-	-
	F	-	-	-	-	-	-	-	-	-	-	-	250	-	-	-
	Cadd9	-	-	-	-	-	-	-	-	1	-	-	-	227	-	-
	Cm	-	-	-	-	-	-	-	-	-	-	-	-	-	250	-
	F#	-	-	-	-	-	-	-	-	-	-	-	-	-	-	250
Total		250	250	250	250	250	250	250	250	250	250	250	250	250	250	250
Accuracy		100 %	96.0 %	96.4 %	100 %	100 %	94.8 %	100 %	100 %	99.6 %	98.4 %	100 %	100 %	90.8 %	100 %	100 %
Mean Accuracy = 3690 / 3750 = 98.4 %																

Table 3: *PCA Classification Results (where strings are not depressed but fingers are held over correct strings)*

Average distance where strings are not pressed but held over the correct strings	Output: Recognition results (data sets)			
	Chord being played		No chord being played	Total
	Same as input chord	Not same as input chord		
5 mm	485 (97.0%)	10 (2.0%)	5 (1.0%)	500
10 mm	180 (36.0%)	2 (0.4%)	318 (63.6%)	500
15 mm	9 (1.8%)	5 (1.0%)	486 (97.2%)	500
20 mm	3 (0.6%)	0 (0.0%)	497 (99.4%)	500

Next, we conducted a chord recognition test. We used a testing set of 3750 data sets from 15 samples chords. As shown in Table 1, using the first method (Interval Threshold Technique) in the Recognizing Guitar Chords step, it cannot recognize accurate enough if the finger positions are not observably different, particularly in some chords which two fingers play at the same fret (e.g., chord A and Am) and in some paired chords which fingers positions are similar (e.g. paired chord C and G7). Nevertheless, the second method (i.e., applying a PCA process) can cope with these problems more efficiently as represented in Table 2. It is generally observed that this method can estimate the 13 guitar chords (chord A, Am, C, D, D7, E, Em, G, G7, F#m, F, Cm and F#) from total 15 testing chords correctly with over 240 data sets (i.e., from total 250 testing data sets), while the other 2 chords (chord Dm and Cadd9) can be estimated accurately with over 227 data sets. Its mean accuracy reaches 98.4%.

Finally, Table 3 shows the PCA classification re-

sults where the strings are not depressed, but simply held over the correct strings perpendicular to the guitar board. It can be seen that if the average distance between the correct fingering positions and the shifted fingering positions on the z-axis is greater than 10mm, the system performs well to recognize correctly (i.e., no input chord is indeed being played and the system feeds back correctly as same as the input). However, if the distance where the strings are not pressed is lower than 10mm (almost pressing), the system is difficult to classify properly (i.e., the chord is not actually being played, but the system determines as chord is being played).

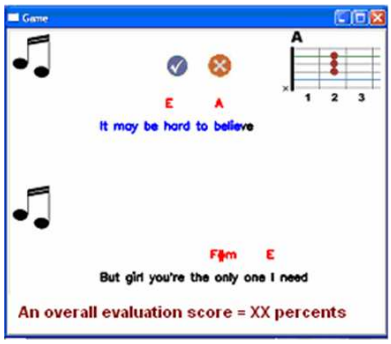


Fig.6: *User Interface of the Guitar Application*

5. APPLICATION

After we recognize the guitar chord which the player's left hand is using, we apply the proposed method to utilize the guitar application. By applying the proposed method, one of the example applications is to identify whether the finger positions are correct and in accord with the finger positions required for the piece of music that the players are playing.

Fig. 6 represents the guitar application for recognizing the guitar chord whenever the player is holding it correctly. This guitar application contains the lyrics, guitar chord charts and voice information that can be used to assist student guitarists to play a song. The lyrics are displayed on the screen in color which changes and is synchronized with the music. Most importantly, this application recognizes each successive chord used by the song, whenever the player is holding it correctly and this provides greater user friendliness. It is considered that this would be of great assistance to guitarists because they are able to automatically identify if their own finger positions are correct and if they match the correct chords required by the musical piece and all the information is provided in real time. Finally, this application will show an overall evaluation score, in percentages, indicating the user's accuracy when the performance has been completed. This application would be invaluable as a teaching aid for guitar players.

6. CONCLUSIONS

In this paper, a vision-based method to recognize the guitar chord played by the left hand has been presented. We detect the position of a player's finger markers by using on-line adaptation of color probabilities and a Bayesian classifier which can deal effectively with considerable illumination changes and a cluttered background. We utilize ARTag's marker information to track the guitar neck, therefore the system can produce the correct chord output while players are moving the guitar, and at the same time it can cope with ARTag occlusion robustly. The triangulation method is utilized to compute the 3D positions of fingers to recognizing whether a guitar string is pressed. We use the results of the guitar and finger detection to compute the guitar chord by utilizing the PCA method for more accurate results. By applying this methodology to different conditions, it has been experimentally demonstrated that the proposed methodology gives robust and accurate results even when used in real time.

The proposed method would be of assistance to the guitarists by providing real time feedback by checking it against the preloaded accurate finger positions required by each musical piece. As a result, it would allow the player to gain a higher level of enjoyment during the lesson. As future work, we are planning to further refine the problem of the finger markers by removing these markers which may result in even greater user friendliness.

References

[1] [1] A. A. Argyros and M. I. A. Lourakis, "Tracking Skin-colored Objects in Real-time," Invited Contribution to the "Cutting Edge Robotics Book", ISBN 3-86611-038-3, Advanced Robotic Systems International, 2005.

[2] A. A. Argyros and M. I. A. Lourakis, "Tracking Multiple Colored Blobs with a Moving Camera," *Proceeding of IEEE Computer Vision and Pattern Recognition Conference (CVPR 05)*, Vol.2, No.2, pp.1178, 2005.

[3] A. A. Argyros and M. I. A. Lourakis, "Tracking Multiple Colored Blobs with a Moving Camera," *Proceeding of IEEE Computer Vision and Pattern Recognition Conference (CVPR 05)*, Vol.2, No.2, pp.1178, 2005.

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

[5] I. Jollife, *Principal Component Analysis*, Springer-Verlag, New York, 1986.

[6] T. Maki-Patola, J. Laitinen, A. Kanerva and T. Takala, "Experiments with Virtual Reality Instruments," *Proceeding of International Conference on New Interfaces for Musical Expression*, pp.11-16, 2005.

[7] F. Liarokapis, "Augmented Reality Scenarios for

Guitar Learning," *Proceeding of Eurographics UK Theory and Practice of Computer Graphics*, pp.163-170, 2005.

[8] Y. Motokawa and H. Saito, "Support System for Guitar Playing using Augmented Reality Display," *Proceeding of 5th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 06)*, pp.243-244, 2006.

[9] O. Cakmakci and F. Berard, "An Augmented Reality Based Learning Assistant for Electric Bass Guitar," *Proceeding of 10th International Conference on Human-Computer Interaction (HCI 03)*, 2003.

[10] Y. Takegawa, T. Terada, and S. Nishio, "Design and Implementation of a Real-time Fingering Detection System for Piano Performances," *Proceeding of International Computer Music Conference (ICMC 06)*, pp.67-74, 2006.

[11] L. Cheng and J. Robinson, "Personal Contextual Awareness through Visual Focus," *IEEE Intelligent Systems*, Vol.16, pp.16-20, 2001.

[12] D. Gorodnichy and A. Yogeswaran, "Detection and Tracking of Pianist Hands and Fingers," *Proceeding of 3rd Canadian Conference on Computer and Robot Vision (CRV 06)*, pp.63, 2006.

[13] A. M. Burns and M. M. Wanderley, "Visual Methods for the Retrieval of Guitarist Fingering," *Proceeding of International Conference on New Interfaces for Musical Expression*, pp.196-199, 2006.

[14] D. Radicioni, L. Anselma, and V. Lombardo, "A Segmentation-based Prototype to Compute String Instruments Fingering," *Proceeding of Conference on Interdisciplinary Musicology (CIM 04)*, 2004.

[15] J. A. Verner, *Midi Guitar Synthesis Yesterday, Today and Tomorrow*, An Overview of the Whole Fingerpicking Thing, Recording Magazine, Vol.8, pp.52-57, 1995.

[16] C. Traube, *An Interdisciplinary Study of the Timbre of the Classical Guitar*, Ph.D. Thesis, McGill University, 2004.

[17] A. M. Burns and B. Mazzarino, "Finger Tracking Methods Using Eyesweb," *Proceeding of Gesture Workshop, Vol. LNAI 3881*, pp.156-167, 2006.

[18] R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Communications of the ACM*, Vol.15, pp.11-15, 1972.

[19] J. A. Canny, "Computational Approach to Edge Detection," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol.8, pp.679-698, 1986.

[20] M. H. Yang, D. J. Kriegman and N. Ahuja, "Detecting Faces in Images: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.24, pp. 34-58, 2002.

[21] C. Kerdvibulvech and H. Saito, "Real-Time Guitar Chord Estimation by Stereo Cameras for Supporting Guitarists," *Proceeding of 10th International Workshop on Advanced Image Technology (IWAIT 07)*, pp.256-261, 2007.

[22] K. Jack, *Video Demystified*, Elsevier Science, 2004.

[23] H. Kato and M. Billinghurst, "Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System," *Proceeding of 2nd IEEE and ACM International Workshop on Augmented Reality*, pp.85-94, 1999.

[24] M. Fiala, "Fiducial Marker Systems for Augmented Reality: Comparison between ARTag and ARToolKit," *In Computer Vision/Computer Graphics Collaboration for Model-based Imaging, Rendering, image Analysis and Graphical special Effects (MIRAGE 05)*, 2005.

[25] R. B. Cattell, "The Scree Test for the Number of Factors," *Multivariate Behavioral Research*, pp.245-276, 1966.

[26] C. Lopez, *The First Stage Guitar Book - Learn How to Play Guitar Easily & Quickly*, Publisher: First Stage Concepts, 2006.

Chutisant Kerdvibulvech received the B.E. degree (Hons) in computer engineering from Chulalongkorn University, Bangkok, Thailand, in 2005 and M.E. degree in information and computer science from Keio University, Yokohama, Japan, in 2007. Currently, he is a Ph.D. student in the Department of Information and Computer Science, Keio University. His work has mainly focused on the research area of computer vision, image processing, and pattern recognition.

Hideo Saito received the B.E., M.E., and Ph.D. degrees in electrical engineering from Keio University, Yokohama, Japan, in 1987, 1989, and 1992, respectively.

He has been on the faculty of Department of Electrical Engineering, Keio University, since 1992. From 1997 until 1999, he was a Visiting Researcher with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA. Since 2006, he has been a Professor in the Department of Information and Computer Science, Keio University. Since 2000, he has also been a Researcher with Precursory Research for Embryonic Science and Technology (PRESTO), Japan Science and Technology Corporation (JST), Tokyo, Japan. He has been engaging in the research areas of computer vision, image processing, and human-computer interaction.

Dr. Saito is a member of The Institute of Electronics, Information and Communication Engineers (IEICE), Information Processing Society Japan (IPSJ), The Society of Instrument and Control Engineers (SICE), and The Virtual Reality Society of Japan (VRSJ).