Vision-Based Detection of Guitar Players' Fingertips Without Markers

Chutisant Kerdvibulvech, Hideo Saito
Department of Information and Computer Science, Keio University
{chutisant@ozawa.ics.keio.ac.jp, saito@ozawa.ics.keio.ac.jp}

Abstract

This paper proposes a vision-based method for detecting the positions of fingertips of a hand playing a guitar. We detect the skin color of a guitar player's hand by using on-line adaptation of color probabilities and a Bayesian classifier which can cope with considerable illumination changes and a dynamic background. The results of hand segmentation are used to train an artificial neural network. A set of Gabor filters is utilized to compute a lower-dimensional representation of the image. Then an LLM (Local-Linear-Mapping)-network is applied to map and estimate fingertip positions smoothly. The system enables us to visually detect the fingertips even when the fingertips are in front of skin-colored surfaces and/or when the fingers are not fully stretched out. Representative experimental results are also presented.

Keywords--- ***Fingertip Detection of Guitar Player, Bayesian Classifier, Gabor Filter, Local Linear Mapping Network***

1. Introduction

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.

Cakmakci and Berard [4] developed a system that assists beginner level musicians to learn the electric bass guitar using augmented reality. This system placed an ARToolKit (Augmented Reality Toolkit) marker on the bass guitar and another marker on a forefinger of the player for tracking the bass guitar position and the forefinger position respectively. Then, they computed the distance between the target note and the current forefinger position for deciding whether to move on to the next note or wait on the current note which makes this a bass guitar application.

Maki-Patola et al. [10] proposed a system called VAG (Virtual Air Guitar) using computer vision. They combined "air guitar playing" and a guitar synthesizer, including sound effects, thus creating a virtual instrument with sensors that follow the hand movements. In this system, a pair of colored gloves was used to track the hand positions for controlling the pitch of the sound.

Liarokapis [8] proposed an augmented reality system capable of simulating and superimposing 3D audio and 3D visual information using ARToolKit marker. The aim of this work is to teach the basics of electric guitar by using a prototype augmented reality interface toolkit.

Motokawa and Saito [11] 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 real stringed guitar as an aid to learning to play the guitar.

However, we have different goals from most the above systems we examined. We propose a system to visually detect and recognize fingering gestures of the left hand of a guitar player (assuming the guitarist is right-handed). Retrieval of guitar player fingering can be applied to use in various purposes such as music education, music theory, physical modeling and automatic music generation.

More recently, Burns and Wanderley [3] 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, and therefore they use the circular Hough transform to detect fingertips. However, this assumption is sometimes difficult to use because the shapes of fingertips in some positions do not resemble semicircular shapes.

In our previous work [6], we proposed a system for recognizing chords played on a guitar. This process was done in real time by recognizing the patterns of fingers' pushing positions detected from input images. ARTag (Augmented Reality Tag) was utilized to detect the guitar position and to define world coordinate relative to guitar neck for dynamic camera calibration. To calculate the positions of fingertips, four colored fingertip markers

Computer Graphics, Imaging and Visualisation (CGIV 2007)
0-7695-2928-3/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

were placed on the fingertips. By utilizing a triangulation method on stereo cameras, the 3D positions of fingertips were recognized when a guitar string was pressed. The results of the guitar and finger detection were used to compute the guitar chord. Applying this system, guitar players can automatically identify whether their fingers are in the correct position. However, although using the markers placed on the fingertips can be easily detected, this sometimes makes it unnatural for playing guitar in real life.

This paper presents a method for retrieval of the fingering information from a guitar player without any fingertip markers. To segment the hand, colored skin objects 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 skin-color probabilities the classifier is able to cope with considerable illumination changes, and therefore it is able to track colored skin even when there are cluttered and dynamic background conditions. Following this, we utilize an artificial neural network to locate the positions of the fingertips in the image [12]. Gabor filters are used to compute a lower-dimensional representation of the image. Then, we use an LLM (Local-Linear-Mapping)-network [9] to generate a smooth mapping for locating the fingertip positions. As a result, the system can recognize the fingering gestures of a guitar player even when the fingertips are in front of the skin area, when the fingers are not stretched out separately, and when using in the complex background. All these conditions occur when playing guitar in a real life.

2. Proposed system

Recently, many methods in computer vision for fingertip detection have been proposed (e.g., include correlation with predefined templates, model fitting, skin color detection with edge merging technique, cylindrical model based tracking, image-division-based decision tree recognition, fingertip shape analysis, a suitable kinematic hand model, etc).

However, when applying the methods proposed in the research cited above, problems usually arise in retrieving the guitar fingering. The first reason is that the lack of contrast between fingertips and background skin adds complication. 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 [3]) for retrieval of guitarist fingering is to use the circular Hough transform 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 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 doesn't appear as the circular shape in some views. For this reason, this assumption is sometimes difficult to practically use.

2.1. System overview

For this reason, we propose an alternative method for detecting the fingertips while playing the guitar. After grabbing the image, we firstly apply a Bayesian classifier and an on-line adaptation of color probabilities for hand segmentation. As the next step in artificial neural network, we apply the global processing algorithm and then the local processing algorithm for each fingertip separately. These steps are schematically shown in Fig.1.

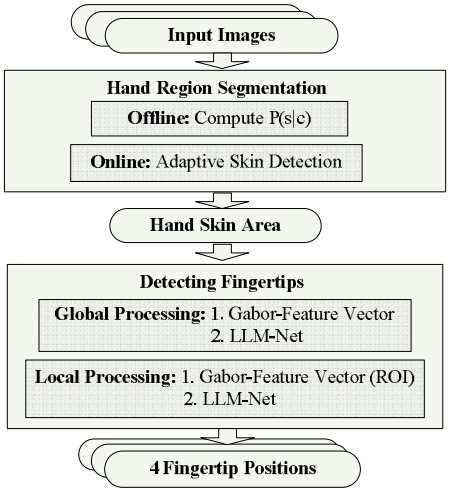

Figure 1: System overview.

2.2. Hand region segmentation

We describe in this section a method that detects and segments the player's hand from the input image. To retrieve the guitar fingering, it is important to segment the hand from the background robustly. We segment the hand from the background using a color detection algorithm. However, a well known problem associated with this is the control of lighting. Changing levels of light and limited contrasts disable correct registration, especially in the case of a complex background (e.g., natural scene) and a dynamic background (e.g., guitar neck, etc).

A survey [13] provides an interesting overview of color detection. Several color spaces have been proposed including RGB, normalized RGB, HSV, YCrCb, YUV,

IEEE COMPUTER SOCIETY

etc. Once a suitable color space has been selected, one of the basically used approaches for defining what constitutes color is to employ bounds on the coordinates of the selected space. We attempted to use this approach (i.e., HSV model) to differentiate the skin-colored region from the background by converting the RGB picture into the HSV space and finding the proper threshold values. However, by using this technique, it is very difficult to deal with illumination changes and a cluttered background.

Therefore, we utilize 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 skin-colored regions. The color representation used in this process is YUV 4:2:2 [5]. However, the Y-component of this representation is not employed for two reasons. First, the Y-component corresponds to the illumination of an image pixel and therefore, by omitting it, the developed classifier becomes less sensitive to illumination changes. Second, compared to a 3D color representation (i.e. YUV), a 2D one (i.e. UV) is of lower dimensionality and is, therefore, less demanding in terms of memory storage and processing costs.

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 skin color, **(ii)** the prior probability *P(c)* of the occurrence of each color and **(iii)** the prior probability *P(c|s)* of a skin being color *c*. By employing Bayes' rule, the probability *P(s|c)* of a color *c* being a skin color can be computed as described in Equation (1).

$$P(s \mid c) = P(c \mid s)P(s) / P(c) \qquad (1)$$

Equation (1) determines the probability of a certain image pixel being skin-colored using a lookup table indexed with the pixel's color. All pixels with probability $P(s|c) > T_{\max}$ are considered as being skin-colored. These pixels constitute seeds of potential skin-colored blobs. Also, image pixels with probabilities $P(s|c) > T_{\min}$ where $T_{\min} < T_{\max}$ that are immediate neighbors of skin-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 skin-colored should be considered as such in the case that it is a neighbor of an image pixel with high probability of being skin-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 skin-colored regions (hand). Each of the remaining connected components corresponds to a skin-colored blob.

The success of the skin-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 skin-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 skin-colored image pixels according to the recent history of detected colored 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)*, *P(c|s)* that have been computed off-line from the training set while the second is made up of $P_W(s)$, $P_W(c)$, $P_W(c \mid s)$, corresponding to the evidence that the system gathers during the W most recent frames. Clearly, 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:

$$P(s \mid c) = \gamma P(s \mid c) + (1-\gamma) P_W(s \mid c) \qquad (2)$$

where $P(s \mid c)$ and $P_W(s \mid c)$ 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 skin-color probabilities the classifier is able to cope with considerable illumination changes effectively, and also it is able to segment hand even in the case of a dynamic background (e.g., moving guitar neck, etc).

2.3. Detecting fingertips

In this section, we describe the method we used to detect the positions of the fingertips. To enable us to locate the fingertips even in areas with low contrast (i.e., when the fingertips are in front of the skin colored region) and in areas of the non-stretched out fingers, we make use of a sequence of 2 neural networks (LLM-networks), which gradually specialize in finding the fingertips [12]. The global network receives a feature-vector which is computed from the whole image (by use of Gabor-filters) and finds a rough approximation for the specific fingertip-position. This point becomes the center of a smaller sub-image, from which the next feature-vector is extracted, and a local neural network finds the fingertip again with improved accuracy.

IEEE COMPUTER SOCIETY

2.3.1. Global processing In this process, we calculate a lower-dimensional representation of the hand image (from the preceding step) by a set of Gabor filters, and apply an artificial neural network to the resulting feature vector.

2.3.1.1. Gabor-feature-vector The input vector of 76800 pixel intensities of an image (320x240 pixels) is too high-dimensional for a direct use as a feature vector. To get a low-dimensional representation of the image, we apply Gabor filters at five positions in the image (arranged like the "Five" on a dice). The 2-D Gabor function is a harmonic oscillator, composed of a sinusoidal plane wave of a particular frequency and orientation, within a Gaussian envelope as represented in Equation (3):

$$g(x,y) = \exp(-\frac{x^2+y^2}{2\sigma^2}) f(u_0 x + v_0 y) \qquad (3)$$

with f(x) = sin(x) or f(x) = cos(x), where σ is the width of the Gaussian and u_0, v_0 are parameters to specify the modulation that has spatial-frequency.

Figure 2: Filter kernels of a cosine-filter, a sine-Gabor filter and an isotropic Gaussian.

To compute the feature-vector, we perform a cross-correlation of the filter with the image. At five points in each image, we apply a Gaborjet consisting of Cosine- and Sine-Gabor filters of 3 different orientations each (0°, 60° and 120° respectively) plus an additional isotropic Gaussian as shown in Fig. 2. The central point is placed on the centroid of the hand pixels. The frequencies as well as the width σ of the Gaussian are adjusted to the size of the image, so that the filters cover the area of the hand. In this way, a 35-dimensional feature vector can be computed in each image.

2.3.1.2. Local-Linear-Mapping-network An LLM-network is a useful tool for the learning of non-linear mappings, which is related to the self-organizing maps [7]. The basic concept of the LLM-network is to perform a vector quantization of the input space combined with an adaptive, locally valid, piecewise linear approximation of the output values. We utilize the LLM-network to generate a smooth mapping from the input variables (a 35 dimensional feature vector $\in \Re^{35}$) onto the output variables (2D position $\in \Re^2$ in the image).

An LLM-network has a fixed number *n* of nodes, each of which consists of a reference vector $\vec{\omega}_i^{\,in} \in \Re^L$ in the input space and an associated reference vector $\vec{\omega}_i^{\,out} \in \Re^M$ in the output space (i = 1, …, n) together with a matrix A_i defining a local linear mapping from $\Re^L$ to $\Re^M$. The task of the matrices A_i is to approximate the mapping linearly in the vicinity of the prototype vectors $\vec{\omega}_i^{\,in}$.

In the training process, the neural network gets new examples each of which consists of a feature vector $\vec{x} \in \Re^L$ and the associated output vector $\vec{y} \in \Re^M$. By comparison of $\vec{x}$ and the reference vectors of the input space, the reference vector $\vec{\omega}_s^{\,in}$ with the smallest Euclidean distance d_s is determined as shown in Equation (4).

$$d_s = \min_i \left\| \vec{x} - \vec{\omega}_i^{\,in} \right\| \qquad (4)$$

The smallest unit *s* determines the answer $\vec{y}^{\,net}$ in the output space for an input feature $\vec{x} \in \Re^L$ as represented in Equation (5).

$$\vec{y}^{\,net} = \vec{\omega}_s^{\,out} + A_s(\vec{x} - \vec{\omega}_s^{\,in}) \qquad (5)$$

At the end of each training step, an adaptation of the reference vectors $\vec{\omega}_s^{\,in}$, $\vec{\omega}_s^{\,out}$ and the matrix A_s is performed in the Equation (6), (7) and (8) respectively:

$$\Delta\vec{\omega}_s^{\,in} = \varepsilon_1(\vec{x} - \vec{\omega}_s^{\,in}) \qquad (6)$$

$$\Delta\vec{\omega}_s^{\,out} = \varepsilon_2(\vec{y} - \vec{y}^{\,net}) + A_s\Delta\vec{\omega}_s^{\,in} \qquad (7)$$

$$\Delta A_s = \varepsilon_3 (d_s^{\,2})^{-1}(\vec{y} - \vec{y}^{\,net})(\vec{x} - \vec{\omega}_s^{\,in}) \qquad (8)$$

where the learning step sizes $\varepsilon_1, \varepsilon_2, \varepsilon_3$ decrease gradually from 0.8. The update of $\vec{\omega}_s^{\,in}$ leads to a change of the network output $\vec{y}^{\,net}$ via the linear interpolation term. This must be compensated in the update of $\Delta\vec{\omega}_s^{\,out}$ by the additional term $+ A_s\Delta\vec{\omega}_s^{\,in}$.

2.3.2. Local processing The rough fingertip position estimates from the global processing stage are now used to define smaller ROI (Regions Of Interest)'s for the subsequent local processing stage. Each ROI is centered at one of the estimated fingertip positions and is a sample with the original, full pixel resolution. Again, we first compute a 35-dimensional feature vector from which then a second artificial neural network (one for each ROI) computes a correction to the initial position estimate. As a result, we can locate the positions of fingertips with improved accuracy even where the contrast is low.

IEEE COMPUTER SOCIETY

3. Results

Fig. 3 provides some results of the experiment. The 'cam' window (left window) depicts the input images which are captured from USB camera with resolution 320x240. This USB camera captures the scene that a guitar player uses our system. The 'detect hand' window (right window) represents the output results (i.e., the hand segmentation result and the fingertip detection result) by using our method. The four color numbers in each fingertip depict four 2D detecting results of fingertips (i.e., little finger [red], ring finger [green], middle finger [blue] and forefinger [yellow]).

In our experiment, we used various kinds of background to test our system. Fig. 3(a) represents some example results when the guitar player is playing the guitar. It can be seen that the system can successfully segment the hand region and, at the same time, the fingertip positions can be located. Furthermore, we tested our system while the fingers of the hand are not obviously stretched out. It can be demonstrated in Fig. 3(b) that the system is able to detect the fingertips even in area of the non-stretched out fingers which is difficult to locate correctly. Moreover, in the case that the fingertip is in front of other skin and the fingers are not clearly stretched out, the system is also able to locate the fingertip positions correctly. Fig. 3(c) depicts some example results of this case which the fingertip of little finger is in front of the skin colored area, but the fingertip position of little finger can be located correctly.

3.1. Accuracy

Then, we evaluate accuracy of our system by using 100 samples images for testing (use 400 images for LLM-network training) in different poses. Fig. 4 shows the accuracy of our experimental results when detecting fingertip positions. All errors are measured in pixels (320x240 image size). With respect to the manually measured ground truth positions, the mean Euclidean distance error and standard derivation error are shown in the table in Fig. 4. The networks for ring finger and middle finger achieve better results, whereas the position of the little finger and forefinger seem to be more difficult to ascertain. A possible reason may be that the fingertips of little finger and forefinger are usually in front of the skin area (area with low contrast) while playing the guitar, and therefore this may effect in the feature vector than the other fingers.

3.2. Speed

The reported experiment was acquired and processed on a Pentium M laptop computer running MS Windows at 1.6 GHz with 1 GB RAM. The speed of skin colored detection (hand segmentation) is approximately 9 fps, while the current speed of fingertips detection is about 1 fps which is not real time. However, the system should achieve higher speed if we decrease the input image size.

3.3. Discussion

In this section, a limitation of the proposed system will be discussed. The accuracy of the fingertips detection depends critically on whether conditions during the on-line operation of the detector are similar to those during the acquisition of the training data set in LLM-network process. Though the use of LLM-network has been shown as a useful tool for the learning of non-linear mappings, the constraint of this method is that the training data set should be nearly similar to the testing data set. If not, the system can not detect the fingertip positions of the guitar player correctly. Fig. 5 depicts an example result of wrong detection because the training data set in LLM-network does not cover enough similarly to this testing data set. Therefore, the system detects the fingertip position of little finger incorrectly. However, the system will probably obtain more accuracy on increasing the number of training data sets.

4. Conclusion

In this paper, a method to detect the fingertips of guitar player without any fingertip markers has been presented. We segment the skin colored hand region of guitar player by using on-line adaptation of color probabilities and a Bayesian classifier which can deal with considerable illumination changes and a dynamic background. Then, we use the results of the hand segmentation to detect the fingertips by utilizing an artificial neural network.

By applying the proposed method, one of the possible applications (proposed in our previous work [6] by using fingertip markers) 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. Although the current speed and accuracy are not enough to make this a guitar application without fingertip markers, we intend to make technical improvements to our system which should result in it becoming a guitar application.

References

[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. In *Proceeding of IEEE Computer Vision and Pattern Recognition Conference (CVPR 05)*, Vol.2, No.2, 1178, 2005.

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

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

[5] K. Jack. Video Demystified. Elsevier Science, 2004.

0-7695-2928-3/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

Figure 3: Example results in various poses.

	Little Finger	Ring Finger	Middle Finger	Forefinger
Mean Distance Error	10.756	6.131	5.490	10.388
Standard Derivation Error	11.989	3.396	3.478	7.799

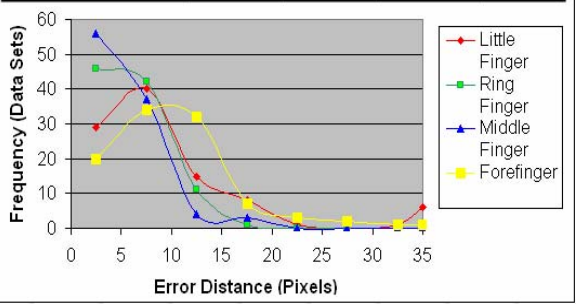

Figure 4: Accuracy of fingertip detection.

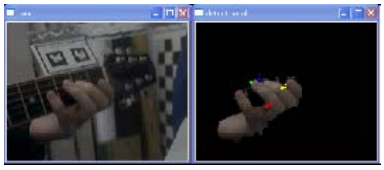

Figure 5: Example of wrong detection result.

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

[7] T. Kohonen. The Self-Organizing Map. In *Proceedings of IEEE*, Vol. 78, 1464-1480, 1990.

[8] F. Liarokapis. Augmented Reality Scenarios for Guitar Learning. In *Proceeding of Eurographics UK Theory and Practice of Computer Graphics*, 163-170, 2005.

[9] E. Litmann and H. Ritter. Adaptive Color Segmentation - A Comparison of Neural and Statistical Methods. *IEEE Transaction on Neural Networks*, Vol.8, No.1, 175-185, 1997.

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

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

[12] C. Nolker and H. Ritter. Visual Recognition of Continuous Hand Postures. *IEEE Transaction on Neural Networks*, Vol.13, No.4, 983- 994 2002.

[13] 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, 34-58, 2002.

IEEE COMPUTER SOCIETY