Hand Gesture Recognition for Robot Hand Teleoperation

Adilson Gonzaga, agonzaga@sc.usp.br
Italo M. Neto, italomneto@hotmail.com
Fabricio Nazareno, blutod@hotmail.com
Department of Electrical Engineering
Engineering School of São Carlos
University of São Paulo,

Glauco A. P. Caurin, gcaurin@sc.usp.br
Leonardo M. Pedro, lmpedro@sc.usp.br
Valdinei L. Belini, valdineibelini@gmail.com
Department of Mechanical Engineering
Engineering School of São Carlos
University of São Paulo,

Marius Stücheli, mariusstuecheli@gmx.ch
Eldgenössische Technische Hochschule Zürich

Abstract. *This paper describes a new methodology for hand gesture recognition and its corresponding application to a real-time Human Robot Hand Interaction system. The gesture segmentation of the video sequence is done by background subtraction through a time-adaptive mixture of Gaussians and a skin-color filter based on RGB model. The hand position and attributes of moment invariants extracted from the hand image boundary are tracked per frame for generating the feature vector. Extracting moment invariants from image outline reduce computational cost, assuring real-time operation. The minimal Euclidean distance is calculated between the input vector and the ten pattern feature vectors previously stored to do recognition of hand gestures. Experiments were carried out using both computer simulations and a five fingered anthropomorphic robot hand. Results show the method efficiency for real-time applications achieving 94.43% of True Positives at 60-frames/second rates. Errors in recognition within the 10 established gestures are very low, showing the method's applicability in Human Robot Hand Interaction systems.*

Keywords: *telerobotics, hand gesture recognition, human robot interaction, anthropomorphic robot hand*

1. Introduction

The purpose of this paper is to develop a methodology to recognize hand gestures from video images to interact with systems, mainly in Human - Robot Hand Interaction (HRHI). Hand gestures are a natural way of communication between people, hence can provide a more natural way to interact with a robot. The gesture recognition based on computer vision has the advantages of being intuitive and requiring less specialized hardware for teleoperation.

This article proposes a system to recognize some hand gestures that can be used to recognize more complex gestures in conventional work environments. The system is presented as a low-level layer that guarantees the recognition of a basic gesture set to be used in the remote operation of robot hands. The further development of this capacity is intended for equipment maintenance applications related to deep water oil exploration.

Human-machine interaction researchers have been attracted to gestures as inputs for the robot behavior for some time. In recent years, this has become a very important research area for computer hardware, vision system development and robotic teleoperation.

Teleoperation of robot hands using gesture information may be implemented in different forms. Teleoperation using haptic devices represent a strong and successful research area for itself Butterfass *et al.* (2004). Important concepts of the field may be found in Bejczy (1992), and a picture of the current achievements is presented by Finger (2009) and Glassmire *et al.* (2004). Another interesting possibility for retrieving information from the gesture is the use of inertial measuring units Miller *et al.* (2004).

This work focus on gesture inputs achieved with the aid of image based approaches. The use of markers in the master hand (glove) simplifies the image processing complexity Li *et al.* (2004) at the cost of a reduced number of gesture possibilities. Nevertheless, a more ambitious and attractive strategy extracts information from normal hand images. Stark et. al. Stark *et al.* (1995) created a system that recognizes eight hand poses. The poses were used as a 3D input device for a visualization program and in a presentation system where the user could control a computer display using gestures. Cui et.al. Cui *et al.* (1995) presented a system that recognizes different hand gestures in front of a complex background. It was able to achieve 93.1% of correct recognition for 28 different gestures, but the system was not used independently and had a relative slow segmentation speed.

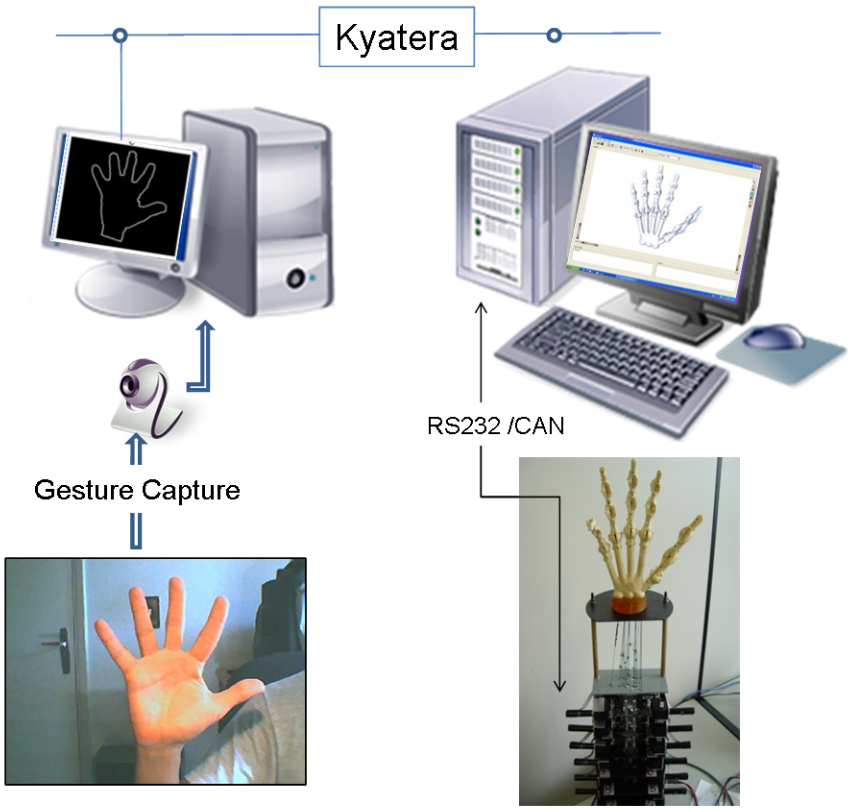

Figure 1. Hand gesture recognition for HRHI system.

Friedmann et. al. Friedmann *et al.* (1992) developed a hand gesture recognition system introducing the extended Kalman filters for structure and motion estimation of video-based systems. Methods for hand pose recognition vary from silhouette orientation histograms Finlayson and Schaefer (2001) to template matching Shimada and Shirai (1996) and table lookup Athitsos and Sclaroff (2001).

This work has a proposal that is similar to the work presented by Hoshino *et al.* (1743-1761), i.e. the use of simple camera images for gesture recovery. Nevertheless, the approach presented here has the advantages of being able to work with arbitrary background conditions, and hands from different users.

2. System Description

The complete system (Fig. 1) was conceived to test teleoperation concepts related to a human robot hand interaction. The system kernel is a management environment based on Matlab m-files that is responsible for relating the received hand gesture data to the experimental robot hand poses. Input data uses a special high speed network called Kyatera. This network is a collaborative environment based on a Fiber-to-the-Lab network that connects companies, research institutes, universities, and funding agencies to develop technological innovations and to generate scientific knowledge Kyatera (2009).

The programs implemented in Matlab receive the pattern reference information (Fig. 8) from the remote gesture subsystem and establish a correspondence between the human hand posture and the corresponding robot hand joint angles. These joint angles are then released to both the GraspIt! grasp simulator Miller and Allen (2000) and Kanguera robot hand Benante *et al.* (2007). In the current development stage, the simulation environment GraspIt! is used just as an auxiliary visualization tool.

2.1 The GraspIt! Environment

A 3D model version of the Kanguera hand was implemented for the GraspIt! robot simulation environment Miller and Allen (2000). The data of the hand links were extracted from their corresponding CAD models and converted to the specific GraspIt! file input format. An additional configuration file describes the relative poses of the single model links in dependence of the joint angles.

During initialization another Matlab program establishes the communication with both the GraspIt! server and the

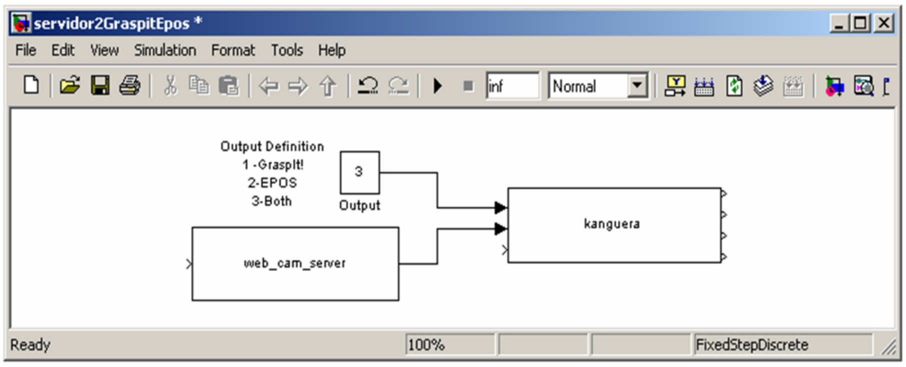

Figure 2. Server implemented in Simulink for the communication with GraspIt! and the Kanguera robot hand.

EPOS controllers. The communication with these controllers is done through a 38.4 kbps serial connection based the RS-232 standard.

Depending on the incoming gesture pattern a specific m-file program in Matlab chooses a predefined hand configuration determined by the joint angles, which reproduce the requested gesture. From a previously established configurations scheme the motor positions are calculated corresponding to the joint angles. Subsequently these positions are sent to the Kanguera Hand. Using a different data protocol, the same joint angles are simultaneously sent to the GraspIt! server that changes stepwise the link positions of the 3D model in the simulator reaching the new joint values.

2.2 The experimental platform - The Kanguera Anthropomorphic Hand

As stated in more details in Benante *et al.* (2007) the Kanguera hand is a five fingered anthropomorphic hand, which adds adduction and abduction capacities to its previous development versions.

In these experiments only three fingers and the thumb are actuated. Further each distal interphalangeal joint has a equivalent four linkage-bar dependency relation to the middle interphalangeal joint. As a result, each finger has 3 DOF and consequently uses 3 servo-actuators. Therefore, in these experiments the Kanguera Hand presents is provided with 12 independent DOF.

New to this version is also the control architecture, as the motor control is performed independently by compact Maxon EPOS controllers. These units receive set-point values as input from the management environment through a serial link and using an internal PID control law, they impose the required voltage to the servomotors. The individual PID parameters for each motor were empirically adjusted for better friction compensation. The motors, which are located outside the hand body, drive thin cables transmitting the movement to the finger links.

In the current Kanguera version the motor encoders are the only sensors available for information feedback. This sensor information is used locally in each controller and globally by the trajectory planner.

3. Gesture Recognition

Our methodology is based on background subtraction by a Gaussian Mixture Model (GMM) described by Stauffer and Grimson Stauffer and Grimson (1999), and an auxiliary segmentation method, associated to the GMM, using a skin color classifier. It explicitly defines the boundaries for skin cluster in a RGB color space by mean of Peer, Kovac and Solina rules Peer *et al.* (2003). So, pixels belonging to the hand are separated from the background image based on background extraction and skin-color segmentation.

Image post-processing is applied before contour detection. Noise and segmentation fault are eliminated by Gaussian filtering and morphological functions (dilatation and erosion). The recognition algorithm uses only extracted borders, that is, the outline or the hand boundary. Therefore, our algorithm is quick enough for real-time applications. The largest extracted boundary from the segmented image will be considered the region of the hand that represents the gesture.

The detected contour is analyzed generating the hand position and orientation for each frame. Position and other hand attributes are tracked per frame distinguishing a hand movement from the background and other moving objects; contour information is also extracted for gesture recognition. A distance classifier recognizes gestures using a feature vector with invariant moments and some hand contour attributes.

The proposed methodology (Fig. 3) was developed to run on Microsoft Windows Operational System and the program was implemented using Borland C++ Builder 6.0 programming language. The OpenCV (Open Source Computer Vision Library) functions Library (2006) were used to compute image filters, image conversions, matrix operations, transpositions, image handling and image visualization. A web cam captures a 320 x 240 color video image and the program runs on a PC (AMD Athlon 64 Processor and 1 GB of RAM memory).

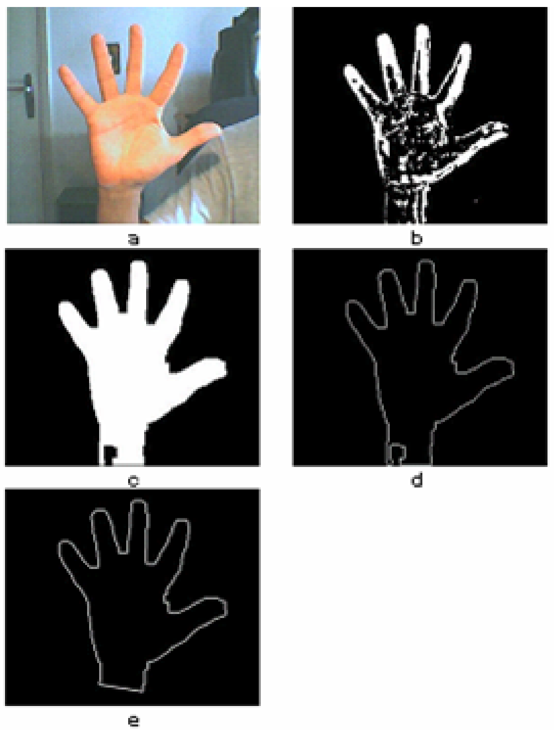

Figure 3. a) Original Image. b) Segmented Image. c) Post-processed Image. d) Hand boundary. e) Hand without forearm.

3.1 Background Model Estimation

Background estimation is related to segment an image region, delimiting the pixels of interest. To do so, we propose modeling the color attribute of each pixel, in an image sequence, through an adaptive mixture of Gaussian distribution. This mixture is updated for each new captured observation, reducing the influence of past observations and improving the adaptation model according to a gradual variation of illumination.

However, Gaussian distributions represent both image foreground and background. It would be necessary to define the distribution of the subsets to describe the background model. The subsets are defined at each observation, according to the weights associated to every distribution, which indicate the frequency that the distribution better represents the pixel. After pixel processing by GMM, foreground pixels are submitted to skin color segmentation.

A mixture of K Gaussian distribution models each pixel in the scene. The history of pixels can be defined as a temporal series whereof values are vectors in RGB space:

$$\overrightarrow{X}_{i,t} = \{R_{i,t}, G_{i,t}, B_{i,t}\} \tag{1}$$

For each time t and for every pixel $i = \{x_0, y_0\}$, the history of pixels can be represented using equation 1, where $\overrightarrow{I}$ is the frame sequence.

$$\{\overrightarrow{X}_{i,1}, \ldots, \overrightarrow{X}_{i,t-1}\} = \{\overrightarrow{I}(x_0, y_0, j) : 1 \leq j \leq t-1\} \tag{2}$$

Therefore, the pixel values are samples of some random variable $\overrightarrow{X}_{i,t}$, which includes the behavior of the mixture K.

The variable $\overrightarrow{X}_{i,t}$ can be 1-dimensional (monochrome intensity), 2-dimensional (normalized color space or intensity-plus-range), 3-dimensional (color), or D-dimen-sional in general (represented as column vectors). The probability that a given pixel has a value $\overrightarrow{X}_{i,t}$ at the time t is shown by:

$$P(\overrightarrow{X}_{i,t}|\overrightarrow{X}_{i,1}, \ldots, \overrightarrow{X}_{i,t-1}) = \sum_{k=1}^{K} \omega_{i,t-1,k} \cdot \eta(\overrightarrow{X}_{i,t}; \overrightarrow{\mu}_{i,t-1,k}, \Sigma_{i,t-1,k}) \tag{3}$$

In this equation, η is a Gaussian probability density function, and $\omega_{i,t-1,k}$ is the weight parameter of the K_{th} Gaussian component that indicates the relative proportion of past observations modeled for every Gaussian distribution. The η_k factor denotes the Gaussian distribution of a mixture and it is represented using the equation 3, where D is the D-dimensionality of vector $\overrightarrow{X}_{i,t}$.

$$\eta(\overrightarrow{X}_{i,t}; \overrightarrow{\mu}_{i,t-1,k}, \Sigma_{i,t-1,k}) = \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma_{i,t-1,k}|^{\frac{1}{2}}} \cdot exp\left(-\frac{1}{2}(\overrightarrow{X}_{i,t} - \overrightarrow{\mu}_{i,t-1,k})^T \Sigma_{i,t-1,k} (\overrightarrow{X}_{i,t} - \overrightarrow{\mu}_{i,t-1,k})\right) \tag{4}$$

In our work, D = 3 because we adopted the RGB color space.

In Eq. 4, $\overrightarrow{\mu}_{i,t-1,k}$ and $\Sigma_{i,t-1,k}$ are, respectively, the mean-vector and covariance matrix of the k_{th} Gaussian component. For computational reasons, the covariance matrix is assumed to be:

$$\Sigma_{i,t-1,k} = \sigma^2_{i,t-1,k} I \tag{5}$$

Therefore, the covariance matrix for the 3-D space is $\Sigma_k = [\sigma^2_R\ \sigma^2_G\ \sigma^2_B]$, where $\sigma^2_R\ \sigma^2_G\ \sigma^2_B$ are the variances of the RGB components. This premise allows red, green, and blue pixel values to be independent and have the same variances. Traditionally, the k value is the same for every pixel, in a range among 3 and 5. When more distributions are used, the model better represents more complex scenes; however, this will increase computational costs.

A Gaussian mixture characterizes the distribution of recently observed values for each pixel in the scene. A new pixel value is represented by one of the major components of the mixture model and used to update the model. For every new observation we update the mixture of Gaussian distribution used to model the observation history of each pixel. Ideally, at each time step t, the parameters of pixel mixture would be re-estimated applying an exact Expectation Maximization algorithm Dempster *et al.* (1977) on some recent observation window, including the last observation. This is a very costly procedure, so, we use an on-line *K-means* approximation for reducing computational time.

The matching is done by a distance metric - normally Euclidean - and a deviation threshold parameter (β) that indicates the deviation among the current observation and the K Gaussian distributions. The parameter β is typically 2.5, so the boundary of the matching zone in RGB-space for η_k encompasses over 95% of the data points that would be drawn from the true Gaussian probability density. The algorithm integrates the new observed data into the background model so that every pixel value of the actual frame, $\overrightarrow{X}_{i,t}$, is checked against the existing K Gaussian distributions.

The pixel will be considered a matched pixel if the following criterion, $|Xt - \mu| \leq 2.5\sigma$, is true for all RGB channels. If a match is found for some distribution, this distribution will be updated.

The K distributions are ordered based on the value ω/σ_k and the first B distributions are used as the background model of the scene, where B is estimated using Eq. 6:

$$B = arg_b\ min \left(\sum_{K=1}^{b} \omega_{i,k} > T \right) \tag{6}$$

The threshold T is the minimum fraction of the background model. In other words, it is the minimum prior probability of the background scene. If the matched Gaussian component model is one of any B distributions, the model must be updated. The prior weights of the K distributions $\omega_{i,t,k}$ at time t are adjusted as shown by Eq. 7, where α $(0 < \alpha \leq 1)$ is the learning rate and the time constant $1/\alpha$ defines the speed at which the distribution parameters change.

$$\omega_{i,t,k} = (1-\alpha)\omega_{i,t-1,k} + \alpha M_{i,t,k} \tag{7}$$

Where $M_{i,t,k}$ is 1 for the matched model and 0 for the remaining models. After this approximation, the weights from both matched and unmatched models need to be renormalized.

The parameters of the distribution that match the new observation are updated using Eq. 8 to 12:

$$\overrightarrow{\mu}_{i,t,k} = (1-\rho)\overrightarrow{\mu}_{i,t-1,k} + \rho\overrightarrow{X}_{i,t} \tag{8}$$

$$\sigma^2_{R,i,t,k} = (1-\rho)\sigma^2_{R,i,t-1,k} + \rho(R_{i,t} - \mu_{R,i,t-1,k})^2 \tag{9}$$

$$\sigma^2_{G,i,t,k} = (1-\rho)\sigma^2_{G,i,t-1,k} + \rho(G_{i,t} - \mu_{G,i,t-1,k})^2 \tag{10}$$

$$\sigma^2_{B,i,t,k} = (1-\rho)\sigma^2_{B,i,t-1,k} + \rho(B_{i,t} - \mu_{B,i,t-1,k})^2 \tag{11}$$

$$\rho = \alpha \cdot \eta(\overrightarrow{X}_{i,t}; \overrightarrow{\mu}_{i,t-1,k}, \sigma_{i,t-1,k}) \tag{12}$$

The parameter ρ is the second learning rate estimated using Eq. 12. If none of the K distributions matches that pixel value, the least probable component will be replaced by a new distribution with a mean equals to the current value of $\overrightarrow{X}_{i,t}$, an initially high variance, and a low weight parameter.

After pixel processing, the segmented foreground pixels are submitted to skin color classification. This segmentation method establishes a human skin classifier that explicitly defines the boundaries of a skin cluster in RGB color space model as defined by Peer, Kovac and Solina's rules Peer *et al.* (2003). According to them, a pixel will be classified as human skin in RGB color space if some rules are accepted, as shown by Eq. 14:

$$\begin{aligned}&(R>95)\cap(G>40)\cap(B>20)\cap(R>95)\cap(G>40)\cap(B>20)\cap max\{R,G,B\}-min\{R,G,B\}\cdots\\&\quad\cdots>15\cap(R-G>15)\cap(R>G)\cap(R>B)\end{aligned} \tag{13}$$

The main advantage of this classifier is the simplicity of skin threshold rules that builds a very fast classifier. The goal of this skin color threshold is too easily discriminate, in real-time, the skin color from the non-skin color pixels. Figure 4b presents segmented frames using GMM and skin color filter applied on the original frames shown at Fig. 4a.

3.2 Post-processing

After pixel processing and skin color filter, the result will be a binary image containing foreground pixels (1's) and background pixels (0's). The segmented binary image must be optimized. Since a statistically significant portion of the input samples will lie in the tails of the distributions, the output image will inevitably contain small noise-generated blobs.

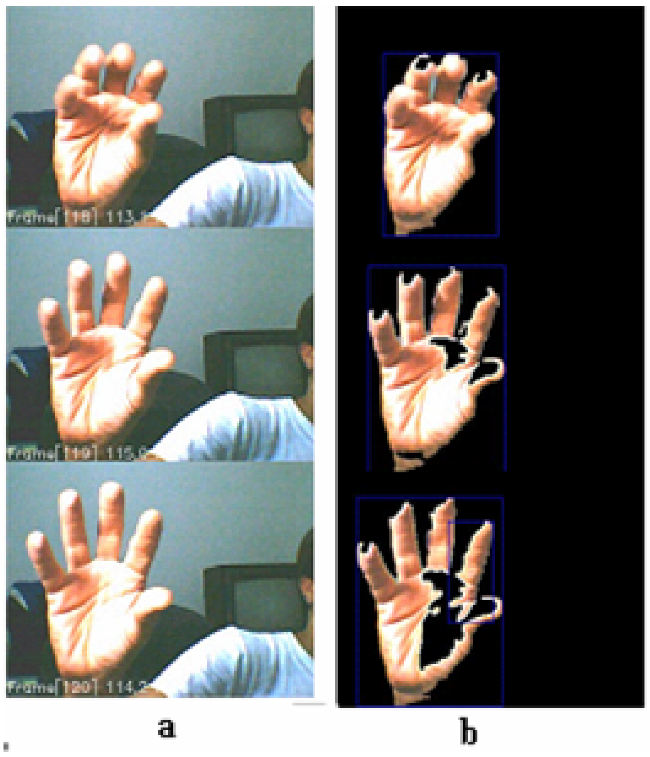

Figure 4. a) Original frames. b) GMM segmentation and skin color filter.

These blobs are eliminated by Gaussian filtering and mathematical morphology functions (dilatation and erosion).

3.2.1 Boundary Detection

Image blobs are detected from the binary image by polygonal approximation. This function creates a boundary list that is examined to verify the biggest one related to its area. Only areas bigger than 5% from the total image area are accepted.

3.2.2 Forearm Filtering

While previous filters are needed for reducing noise, the aim of the forearm filter is avoiding forearm effects in the image. The hand parameters are mistaken or even dominated by the forearm's parameters when the detected boundary is too big.

The hand and the forearm together appear as a region that changes with the arm movement, affecting feature extraction procedure and decreasing recognition rates. Therefore, it is necessary to "delete" the forearm from the hand image to ensure the correct gesture classification.

An offset circumference can be traced for detecting the maximum region that comprises the palm hand, so, detecting a reference region for deleting the forearm (Fig. 5b). Another circumference is drawn, named palm circumference that encompasses the palm hand. The center and radius of the palm circumference are calculated by the Euclidean Distance Transform (EDT) Borgefors (1996).

The following algorithm shows a simple way to "delete" the forearm. It is based on simplifications from the Deimel and Schröter's work B and Schröter (1999):

- Find the palm circumference center and radius by Euclidean Distance Transform.
- Calculate the offset circumference whose radius is 1.65 times the palm circumference radius (Fig. 6a).
- Determine the lower intersection points between the offset circumference and the hand boundary.
- All pixels below the lower intersection points are assumed to represent the forearm and are removed (Fig. 6b).

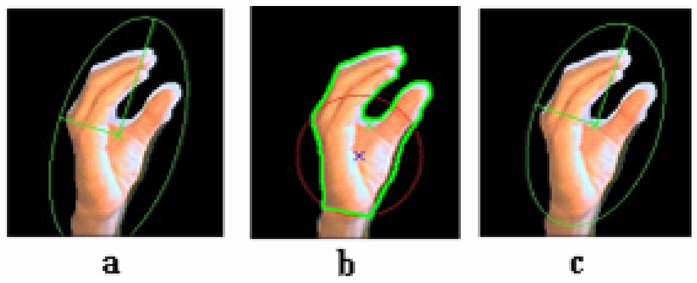

Figure 5. (a) Equivalent ellipse including the forearm. (b) New boundary without the forearm and palm offset circumference. (c) New equivalent ellipse without the forearm.

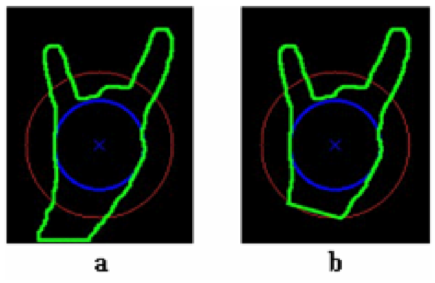

Figure 6. a) Segmented image boundary, offset circumference and palm circumference. b) Hand boundary without the forearm.

After forearm deletion, only the hand boundary will be the image considered for gesture recognition. Figure 7 shows the hand boundary with and without the forearm.

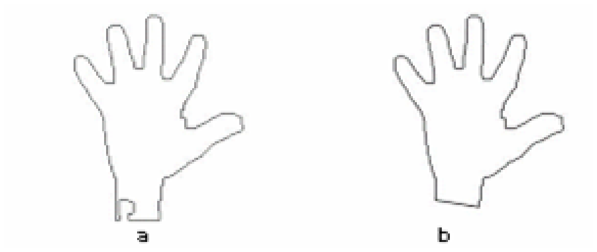

Figure 7. a) Hand boundary with the forearm. b) Hand boundary without the forearm.

3.3 Hand Gesture Recognition

Gesture recognition consists in comparing a new gesture with pattern models to determine whether the given new gesture matches any previously known pattern gesture or not.

The recognition task is based on the hand boundary format at a certain time. The goal is to recognize some basic hand gestures in a video sequence.

The extracted features for pattern recognition are based on moment invariants and other attributes, derived from these moments and calculated over the hand boundary. The Euclidean distance is applied for searching similar boundaries between the captured frame and all boundary patterns stored in the system.

3.3.1 Feature Vector

For gesture classification, hand features must be extracted from the hand boundary image. Two boundaries with dissimilar features should have very different or even orthogonal feature vectors.

The adopted feature vector (v) has fourteen attributes for pattern matching:

$$v = (p1, p2, p3, p4, p5, p6, p7, \ldots, p14) \tag{14}$$

where:

$[p1, \ldots, p7]$: Are the moment invariants of Hu Hu (1962).
$[p8]$: Is the major semi-axis orientation of the equivalent ellipse.
$[p9]$: Is the boundary circularity.

$[p10]$: Is the boundary eccentricity.
$[p11]$: Is the boundary radius of gyration.
$[p12]$: Is the contour spreadness.
$[p13, p14]$: Are the maximal and minimal inertial moment.

3.3.2 Recognition

The gesture recognition is based on Euclidean distance between the current feature vector and the pattern feature vectors. The recognition is done taking into account the smallest distance. The ten reference pattern boundaries chosen for the matching process are presented at Fig. 8.

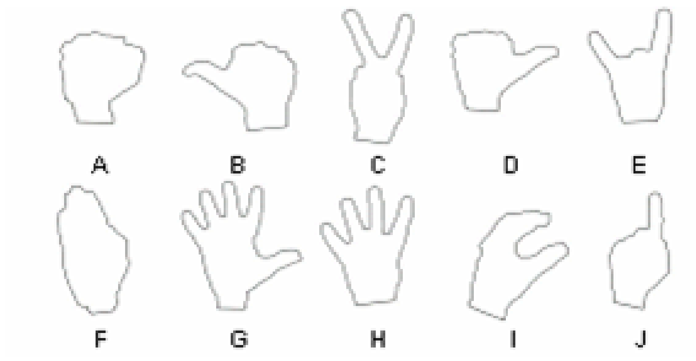

Figure 8. The pattern reference boundaries.

Since it is desired that all feature vector attributes have the same weight and do not dominate the distance calculation, it is necessary to normalize their values. Therefore, all values are normalized using the mean and standard deviation from the feature vectors.

4. Results

Table 1 shows the methodology performance comparing only one gesture to the ten different hand gestures. The method performance can be analyzed as follows:

The B gesture achieved the highest positive recognition rate (100%), and the I gesture achieved 1095 true positives (98.92%) and only 9 false positives (1.08%). The selected features based on moment invariants discriminate better this type of gesture from the others due its shape and spatial distribution. The D gesture was recognized one time as A gesture and one time as I gesture. Visually comparing D type with A and I type we can observe similarities in the shape and spatial distribution. The recognition performance for D gesture was 95.83% of true positives (TP). The H gesture achieved the most false positives (FP), resulting in 50 incorrect results considered as C and C was confused 35 times with the I gesture due to open fingers. Moment invariants do not discriminate well among gestures with one, two or more open fingers.

Table 1. Method performance for 100 gesture videos.

Gesture	Frames/60	TP	FP	TP(%)	FP(%)
A	3748	208	8	92.71	7.29
B	3727	246	0	100	0
C	3642	437	44	92.48	7.52
D	3701	70	2	95.83	4.17
E	3701	1000	9	98.63	1.37
F	3720	166	30	83.45	16.54
G	3697	583	12	98.24	1.76
H	3624	505	59	89.79	9.64
I	3740	1095	9	98.92	1.08
J	3741	628	33	94.2	5.8

The F gesture achieved the smallest TP rate with 83.45% of TP. It was confused with C, E, G, H, and J gestures, all of which show one or more fingers.

5. Conclusions

The GMM algorithm is a fast way to provide hand gesture segmentation in video sequences. The experimental results using the Kanguera Hand shown in this work demonstrate that this approach is suitable for real-time robotic applications. The GMM generates a robust background model for complex background scenes with different geometric shapes, textures and colors, gradual variations of illumination and shading. Furthermore, GMM is a robust technique for scene motion segmentation, both in external or indoor environments. Although the model is robust for these different situations, it fails under high light variations, when a great amount of shades is detected in the foreground, or when the moving object is static. The cause of the late is that, if the hand movement stops, the object is recognized as part of the background and is removed from the scene. Some problems occur when light changes abruptly and backgrounds have skin-like colors.

The hand image boundary used to compute the moment invariants is different from the ones in other works that use the segmented region Chaumette (2004). In our work, the number of considered points is quite small, drastically reducing the processing time and consequently increasing the performance. Besides, this method is easy to implement and fast.

The proposed methodology was applied for recognizing a single hand gesture in a video sequence, but it can also be modified for recognizing a sequence of gestures implying in a dynamic gesture. The achieved high recognition rate allows applications in real-time tasks.

The experimental results demonstrate further that the chosen features are able to correctly discriminate shapes, sometimes even from deformed boundaries. The proposed methodology offers a high performance hand gesture recognition system, at low cost, using a real-time video stream.

Demonstrations videos of the system are available at:
http://www.mecatronica.eesc.usp.br/wiki/upload/b/b2/Mov01672.rar

6. Acknowledgements

The authors would like to thank the State of SŸao Paulo Research Foundation - FAPESP (2006/06597-5) and The National Council for Scientific and Technological Development - CNPq (301417/2007-5) for the financial support of this research.

7. References

Athitsos, V. and Sclaroff, S., 2001. “3d hand pose estimation by finding appearance based matches in a large database of training views.” *Proc. of the IEEE Workshop on Cues in Communication*.

B, B.D. and Schröter, S., 1999. “Improving hand-gesture recognition via video based methods for the separation of the forearm from the human hand”. *GW’99 - 3rd Int. Gesture Workshop*, pp. 17–19.

Bejczy, A.K., 1992. “Teleoperation: the language of the human hand.” *Proc. of the IEEE Int. Workshop on Robot and Human Communication*, pp. 32–43.

Benante, R.C., Pedro, L.M., Massaro, L.C., Belini, V.L., Araujo, A.F.R. and Caurin, G.A.P., 2007. “A self-organizing state trajectory planner applied to an anthropomorphic robot hand”. *Intelligent Robots and Systems. IEEE/RSJ Int. Conf. on Robotic Systems*, Vol. 3082-3087.

Borgefors, G., 1996. “Distance transformations in digital images”. Vol. 34, pp. 344–371.

Butterfass, J., Fischer, M., Grebenstein, M., Haidacher, S. and G, G.H., 2004. “Design and experiences with dlr hand ii”. *Proc. of the Automation Congress*, Vol. 15, No. 105-110.

Chaumette, F., 2004. “Image moments: A general and useful set of features for visual servoing”. *IEEE Transactions on Robotics*, Vol. 20, No. 4, pp. 731–723.

Cui, Y., Swets, D. and Weng, J., 1995. “Learning-based hand sign recognition using shoslif-m”. *Proc. of 5th Int. Conf. on Computer Vision*, pp. 631–636.

Dempster, A., Laird, N. and Rubin, D., 1977. “Maximum likelihood from incomplete data via the em algorithm”. *J. Royal Statistical Soc.*, Vol. 39, pp. 1–38.

Finger, E., 2009. URL `http://lims.mech.northwestern.edu/projects/finger_exo/`.

Finlayson, G. and Schaefer, G., 2001. “Hue that is invariant to brightness and gamma”. *Proc. of the 12th British Machine Vision Conf.*, pp. 303–312.

Friedmann, M., Starner, T. and Pentland, A., 1992. “Device synchronization using an optimal linear filter”. *Proc. of the Symp. on Interactive 3D Graphics, Special Issue of Computer Graphics*, Vol. 26, pp. 57–62.

Glassmire, J., O’Malley, M., Bluethmann, W. and R, R.A., 2004. “Cooperative manipulation between humans and teleoperated agents.” *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 12th Int. Symp. on Haptics*, pp. 114–120.

Hoshino, K., Tamaki, E. and Tanimoto, T., 1743-1761. “Copycat hand - robot hand imitating human motions at high speed and with high accuracy”. *Advanced Robotics*.

Hu, M.K., 1962. “Visual pattern recognition by moment invariants”. *IRE Trans. Info. Theory*, Vol. IT-8, pp. 179–187.

Kyatera, 2009. URL `http://kyatera.incubadora.fapesp.br/portal/weblabs-en`.

Li, J., Su, W., Zhang, Y. and Guo, W., 2004. "Vision-based grasp planning system for dexterous hands". *Proc. of the Int. Conf. on Intellligent Manipulation and Grasping*.

Library, O.O.S.C.V., 2006. URL `http://www.intel.com/research/mrl/research/opencv/`.

Miller, A.T. and Allen, P.K., 2000. "Graspit!: A versatile simulator for grasp analysis." *Proc. ASME Int. Mechanical Engineering Congress & Exposition*, pp. 1251–1258.

Miller, N., Jenkins, O.C., Kallmann, M. and Mataric, M.J., 2004. "Motion capture from inertial sensing for untethered humanoid teleoperation." *4th IEEE/RAS Int. Conf. on Humanoid Robots*, Vol. 2, pp. 547–565.

Peer, P., Kovac, J. and Solina, F., 2003. "Human skin colour clustering for face detection". *EUROCON - Int. Conf. on Computer as a Tool*.

Shimada, N. and Shirai, 1996. "3d hand pose estimation and shape model refinement from a monocular image sequence". *Int. Conf. on Virtual Systems and Multimedia*, pp. 423–428.

Stark, M., Kohler, M. and Zyklop, P.G., 1995. "Video based gesture recognition for human computer interaction". *Modeling-Virtual Worlds-Distributed Graphics*.

Stauffer, C. and Grimson, W.E.L., 1999. "Adaptive background mixture models for real-time tracking". *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, Vol. 2, pp. 246–252.

8. Responsibility notice

The author(s) is (are) the only responsible for the printed material included in this paper.