

# Hybrid Machine Vision Control

Andre Senior, and Sabri Tosunoglu  
Florida International University  
Department of Mechanical Engineering  
Miami, Florida 33174

305-348-6841

[mail@andresenior.info](mailto:mail@andresenior.info)

## ABSTRACT

The tracking of objects in real time lends itself to vast industrial and practical applications; of interest is the tracking of objects within a manipulator's workspace by using machine vision. This paper presents the basics for control by vision of a 3-DOF robotic manipulator using artificial intelligence. The object or objects are first identified, next the object's movement or trajectory is estimated and then the robot's actuators are controlled to track the object of interest. The system is hierarchical with the identification process at the top of the algorithm, being carried out by a neural network (classifier) capable of identifying objects of interest and passing on the object's position within the viewing frame to the tracking process. Within the tracking process a hybrid combination of two vision processes interact to estimate the orientation and position of the object within the viewing frame, after which the hybrid process passes on appropriate information to the manipulator's control module. The manipulator control module utilizes the information from the tracking process to determine appropriate manipulator configurations in light of the ultimate tracking aim. The vision-based control is then illustrated via several experimental tests to verify its use and further improve the method.

## Keywords

Hybrid Vision Control, Vision System, CAMSHIFT, MHI, Neural Network, OpenCV.

## 1. INTRODUCTION

Intelligent robotic machines depend on realizations in artificial intelligence (AI); of particular interest are the modes of receiving and analyzing sensory data. The common steps [1] involved in the realization are listed as

- (i) Perception,
- (ii) Analysis,
- (iii) Determination, and
- (iv) Operation.

Perception is provided by a sensory system attached to the robot, analysis and determination is achieved by an information/data processing unit and operation is accomplished by controlling the robots actuators. Figure 1 shows a flow diagram representing data throughput within the system. Sensory information takes many forms and can prove overwhelming considering the computing process effort required.

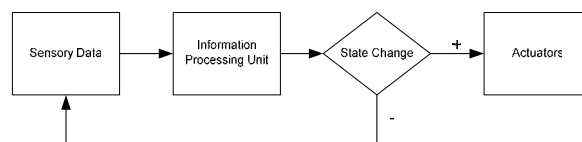


Figure 1: Block diagram of dynamic robotic system

Vision is viewed as one of the more important senses a human possess [2]. The ability to perceive images translates into more information than would normally be perceived by the other human senses. Likewise, machine vision has been seen as a popular method to perceive sensory data, even though viewed simply as a computer vision system able to do image capturing and processing. As the technology changes, machine vision systems have provided potential applications in many industries such as human computer interaction and robotic guidance. This paper will develop the latter, by developing a hybrid algorithm in robotic control demonstrated by using a robotic manipulator controlled by vision. A combination of open computer development methods found in Intel's OpenCV [8] and ANSI C++ will be used to demonstrate its implementation.

## 2. BASICS

The objective of the algorithm is to sense, plan and act in a continuous loop until there is a desired change in the environment or user input. If we consider moving an object with machine vision control and a manipulator arm, the aim is to first identify the desired object. If the object is moving, the system will then need to estimate the object(s) trajectory and then match the robot's link actions to intercept the object within its trajectory. The computer processing power required differs along the different stages and provides interesting challenges.

One technique used to lower computing power requirements is to pass the required data processing efforts to robust computing modules. Each module will employ methods that are capable of running at a lower computer operating system tread level or CPU usage level. To illustrate this, if we consider physically walking from one point to the next, our walking action is done innately, but much more thought is given to determining the starting and ending points of our path. The computing power used should match the processing power necessary to accomplish each process. Once the objective is identified, the other processes can be carried out using more robust computing methods which require less computing<sup>1</sup> effort as illustrated in figure 2. In

<sup>1</sup> The test bed computer used in determining the computing effort (CPU Usage) was a Pentium 4 HT Computer with 1Gb of memory, Operating System computing effort at idle was 5%.

identifying an object with regards to human walking, the computing effort is greater in determining (identification) where to go, whereas sight and walking is done innately.

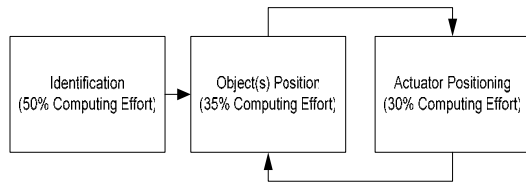


Figure 2: Computing Effort for Different Stages

### 2.1 Identification

Without having experiences like humans, AI has limitations with respect to analysis and hardware; limitations will be developed later. Humans have the innate ability to recognize objects based on experience and the cognitive ability to merge information. Machine vision systems on the other hand have to presently rely on relatively pseudo methods to identify objects [4, 5]. Pattern recognition is the most fundamental method of identifying objects. There are numerous techniques that are employed in recognizing patterns that lend to data mining [11, 12].

Data mining techniques are relevant to machine vision as they develop machine learning specific techniques such as neural networks [13]. Of interest is the ability to train a neural network to recognize simple features of an object within an image [6, 7]. The neural network feature used in this paper works by matching simple noticeably connected patterns in an image. Such as edges, lines, center-surrounded features or a combination features as shown in figure 3 below.

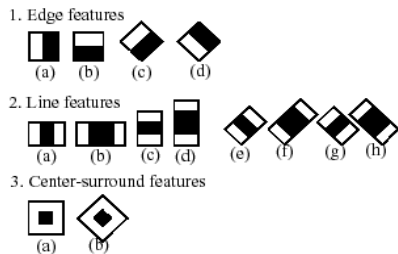


Figure 3: Neural Network (Haar-like) features

Application of a neural network is attractive as it presents modularity, allowing different trained networks to be implemented within the same system. The aim of the neural network is to identify the desired object and pass on its coordinates to the tracking module that is pass on information about the object's size and position within a captured frame.

The neural network (also known as the classifier) is trained by examining positive samples, negative samples and classifying them. The positive samples are views of the desired object which are scaled to the same size. The negative samples on the other hand are views of arbitrary images of the same size as the positive samples. After training, the neural network can classify outputs within a region of interest as being positive or negative. Within the OpenCV library there is a variant of the Adaboost classifier called Haar boost classifier.

The principles the Adaboost classifier uses are as follows:

1. Given sample images  $(x_1, y_1), \dots, (x_n, y_n)$  where  $y_i = 0$  or  $1$  for negative and positive samples respectively
2. Initialized weights  $\omega_{1,i} = 2^{m-1} 2^{t-1}$  for  $y_i = 0$  or  $1$  respectively, where  $m$  and  $t$  represents the number of negative and positive samples
3. For  $t = 1, \dots, T$ 
  - Normalize the weights

$$\omega_{t,i} \leftarrow \frac{\omega_{t,i}}{\sum_{j=1}^n \omega_{t,i}}$$

where  $\omega_t$  is a probability distribution

- Classifier  $h_j$  is trained for each feature. The associated error with respect to  $\omega_t$  is  $\epsilon_j = \sum \omega_i |h_j(x_i) - y_i|$
  - Use the classifier with the lowest error
  - Update weights,  $\omega_{t+1,i} = \omega_{t,i} \beta_t^{1-e_i}$  where  $e_i = 0$  or  $1$  for positive and negative classifications
4. The ultimate classifier is

$$h(x) = \begin{cases} 1 & \sum_{t=1}^T \log(\frac{1}{\beta_t}) h_t(x) \geq 0.5 \sum_{t=1}^T \log(\frac{1}{\beta_t}) \\ 0 & \text{otherwise} \end{cases}$$

Figure 4 shows the algorithm of the Haar identification module using OpenCV.

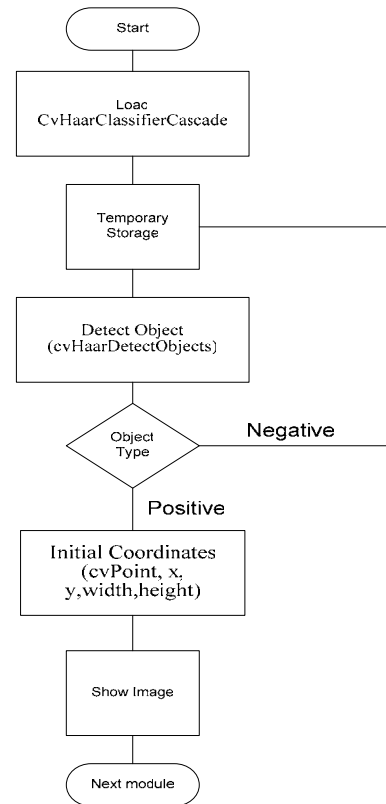


Figure 4: Neural Network Algorithm

The classifier being robust as it is, has limitations, the training process takes a relatively long time and requires tremendous

computing power for the large training set. The latest trend is to have AI train the classifier that is for the classifier to learn on its own. For example the classifier can identify body positions in a newscast by capturing and analyzing each frame. Most newscast presenters assume the same general orientation within a frame, thus giving the classifier ample samples. This method can be further enhanced by determining early in the training of the classifier what makes a good feature, how many features to use and how to compute variations in the features.

The detail principles involved in designing the neural network are beyond the scope of this paper. Commercial neural networks are available, that can be easily implemented. The classifier is the principal method for identification within the developed hybrid vision system. A less autonomous method would be to use a human operator to physically tag the desired object(s) either physically or on a video image.

## 2.2 Tracking

Tracking estimation is achieved by utilizing two techniques namely Motion Segmentation with motion history image gradients (MHI) and Continuously Adaptive Mean Shift (CAMSHIFT). Both techniques were developed by Davis and Bratski [9, 10]. Both techniques are also intended for human feature tracking and interaction; however the implementation presented in this paper is geared at general machine vision systems with industrial applications. Within the system the neural network passes on a set of coordinates that define the position and size of a search area.

### 2.2.1 Motion History Image (MHI)

Within the motion history image algorithm temporary layers create residual silhouettes of moving object(s) within a captured frame. From the residual images directional bearings can be computed to give relative direction (degrees) of objects within the frame. The accuracy of the algorithm is depended on the frame rate of the video input device and the rate of computing motion using the temporary silhouettes.

The MHI algorithm shown in figure 5 generates silhouettes of the entire frame. It generates the silhouettes from the movement within the frame that is to say that only with a difference in one frame to the next can a silhouette be generated. After a few silhouettes are generated it calculates the motion gradient and motion segmentation (differences from one frame silhouette to the next). Having both the motion gradient and segmentation allows for the orientation and direction of the object to be computed. If the object tends in a y direction relative to the video input device, a relative bearing is calculated. Directions of smaller movements of the general object are computed as well. MHI works by using a floating point value for each timestamp silhouette, t. The MHI is represented with respect to time as:

$$tMHI\delta(x, y) = \begin{cases} \tau & \text{if current silhouette at } (x, y) \\ 0 & \text{else if } tMHI\delta(x, y) < (\tau - \delta) \end{cases}$$

Where  $\tau$  is the current timestamp and  $\delta$  is the time interval for calculating silhouettes. Using the MHI timestamp technique allows computational tolerance within the system with respect to the capture device hardware specifications.

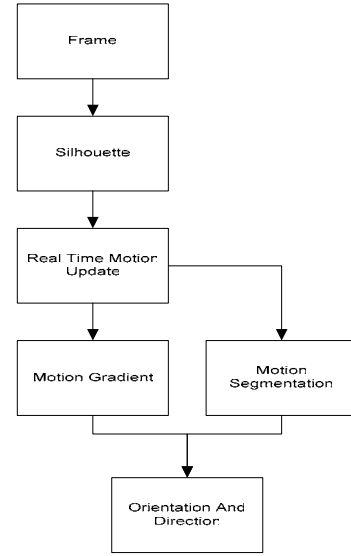


Figure 5: Motion History Image Algorithm

Figure 6 shows an object moving to the left. Smaller features that have movement in the frame are given and shown direction within the smaller circles with lines.

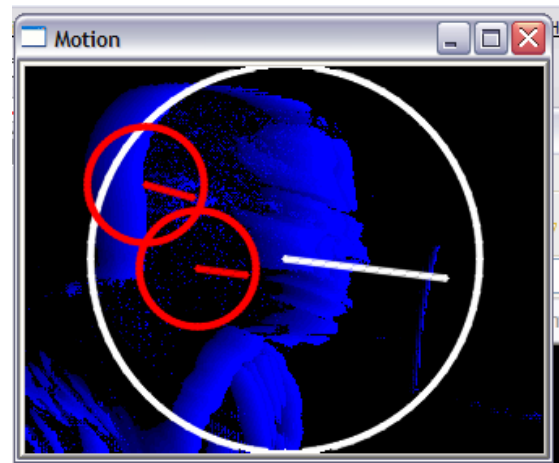


Figure 6: MHI Demonstration

In figure 6 it will be noticed that the object appears to be faded from the left to the right, the silhouettes allow gradients of the MHI to be calculated. The gradients are represented by the circles with the lines as shown in figure 6. For each MHI calculation at the maximum time interval, the global object representations can be seen in figure 6 as the large white circle with the white line. The global weighted orientation can be calculated from the following equation:

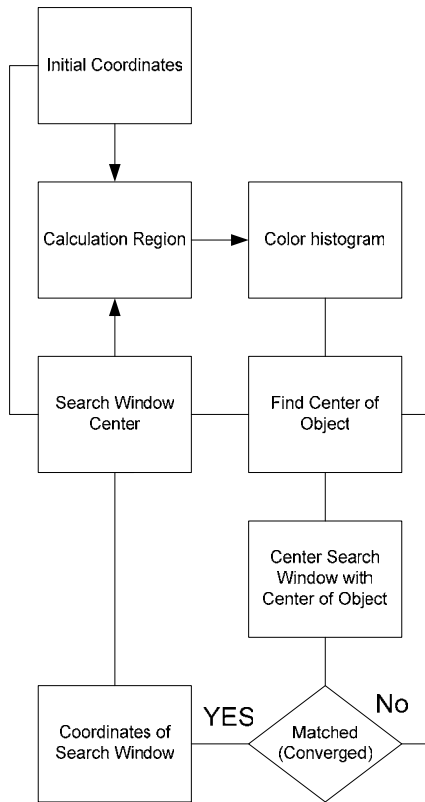
$$\phi = \phi_{reference} + \frac{\sum_{x,y} D(\phi(x, y), \phi_{ref}) \times \text{norm}(\tau, \delta, MHI \delta(x, y))}{\sum_{x,y} \text{norm}(\tau, \delta, MHI \delta(x, y))}$$

Where  $\phi$  is the global motion orientation,  $\phi_{reference}$  is the base reference angle,  $\phi(x, y)$  is the motion orientation map found from

gradient convolutions,  $norm(\tau, \delta, MHI\delta(x, y))$  is a normalized MHI value and D represents the difference between the current orientation and the reference orientation angle [15].

### 2.2.2 Continuously Adaptive Mean Shift

The Continuously Adaptive Mean Shift (CAMSHIFT) algorithm works from the basic principle of continuously calculating the mean changes in color within a search area with respect to an initial tracking area color. Figure 7 shows the algorithm flow.

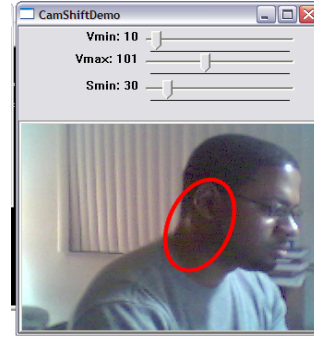


**Figure 7: Continuously Adaptive Mean Shift Algorithm**

Figure 8 demonstrates an implementation of the CAMSHIFT algorithm. The eclipse in the captured frame tracks that portion of the user's face within the frame. If the user moves out of the viewing path of the camera, the references are retained and tracking continues once the user comes back into the viewing frame. Of particular interest is that once the coordinates of the object of interest are given to the CAMSHIFT module, the module has the ability to track an object in the event the object changes orientation with respect to the camera that is if the object rolls.

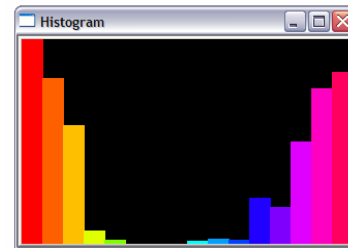
The principles the CAMSHIFT algorithm uses are as follows:

1. Get Initial Coordinates
2. Calculate the Mean Shift and store the center moment
3. Set the search area size and center moment size the same
4. Calculate average location movement with respect to defined threshold, that is until convergence



**Figure 8: Camshift Demonstration**

In figure 9, a color probability histogram is shown representing the region to be tracked in Hue space. The algorithm first computes the color probability histogram based on the initial search area of the object to be tracked. Upon motion within the captured frame, the algorithm determines the motion by using an estimation or gradient approach within the neighborhood of the original search area.



**Figure 9: CAMSHIFT Histogram**

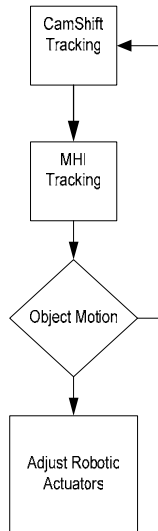
The algorithm recomputes the probability histogram in a continuous and adaptive fashion, allowing elements within the original search area to always be the point of interest irrespective of the objects orientation. The CAMSHIFT algorithm is attractive because it employs ideas from statistics and probability distributions, which allows the algorithm to ignore points that are not in the area of interest.

### 2.2.3 Manipulator Control

After the initial coordinates and size for the search area are passed on the CAMSHIFT module from the neural network, the manipulator algorithm follows as shown in figure 10. The CAMSHIFT module first determines the speed and orientation of the object within the captured frame and then the MHI tracking module determines the objects general direction within the frame. The robot actuators are adjusted to reflect a path or position inline with the objects trajectory for interception.

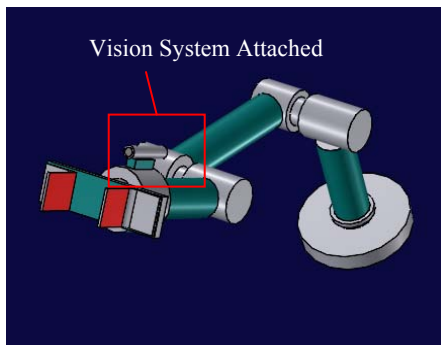
The analysis associated with a 3-DOF robotic arm will not be presented in this paper; however, trajectory planning will be discussed. The aim of the vision system is to ultimately provide coordinate data to the robotic arm in order for the arm to accomplish its objective. The robotic arm has hardware limitations that will restrict it in its workspace. For example, in a packaging environment, if the system is given a palletizing task, the aim is to pack objects in a defined order, the robot might be fixed at the base in one position. Furthermore, the objects could initially be randomly moving in the workspace along with undesired objects, making the task difficult.

The robotic arm movement is essentially mimicked within the vision system algorithm, defined within subroutines that describe forward, reverse, up, down and rotational motions. The limiting positions and payload capacity of the robotic arm are also provided to the algorithm. The vision system first needs to identify and lock on to one object, and then actuate the robotic arm to intercept the object. The vision system has a complete camera system that scans the entire workspace. The robotic system is illustrated in figure 11.



**Figure 10: Manipulator Control Algorithm**

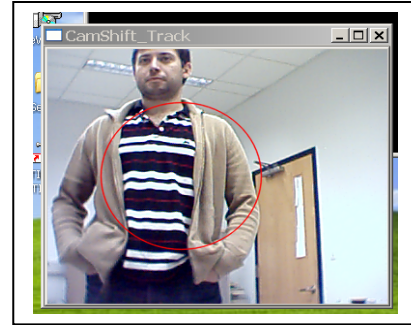
As can be seen in figure 11, the vision system is placed near the gripper, having the ability to rotate around the gripper link to scan the workspace. The algorithm passes instruction from the vision system to the robotic arm control system. The interaction is of the master-slave type with the vision system being the master and the robotic arm system being the slave, interpreting instructions.



**Figure 11: Robotic Arm with Vision System**

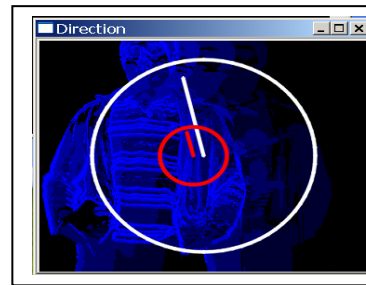
### 3. IMPLEMENTATION

The following implementation shows a converged effort to control a manipulator by vision. First the system identifies the object. The neural network would first identify the object of interest. Next as seen in figure 12 the CamShift tracking module, is initialized around the object of interest and then robust tracking begins.



**Figure 12: CamShift Tracking**

The CamShift module along with the MHI module determines the direction of the overall frame movement as shown in figure 13.



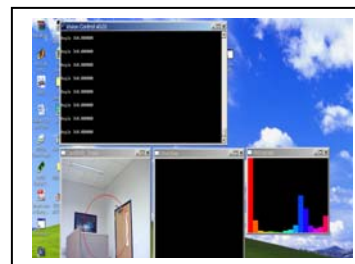
**Figure 13: MHI Tracking**

The complete implementation can be appreciated as shown in figure 14. In the background of figure 14, the black screen represents data that is being correlated with the system to map appropriate movement for the camera and robotic arm motion



**Figure 14: Vision Control Layout**

If the object that the initial coordinates selected move out of frame the histogram model remains the same as can be appreciated in figure 15, where the object is not in the viewing frame. Once the same object comes back into view the tracking module resumes from wherever the object comes unto the frame.



**Figure 15: Histogram Hue Model with no Object**

The direction and position of robotic manipulator system is determined by computing the orientation and general direction explained before - the aim being that the calculated coordinates correspond to the robotic manipulator and the camera orientations in order to keep the object within view or until interception.

#### 4. LIMITATIONS

We are strictly considering frame by frame action, changes in the environment and hardware settings will ultimately affect the system, such as changes in camera focus, changes in environmental lighting, changes in the difference between camera frame rate and object movement and events in the robots working space such as incurred obstacles. The hardware plays a critical role in determining the effectiveness of system. Within the Camshift tracking module, if an object has the same mean color, then the algorithm may veer off and track a new object. Likewise with the MHI module, it can be appreciated that with multiple objects the only successful method of dealing with tracking one object is to focus and minimize the viewing path, which would ultimately restrict the robot workspace.

The hardware sensitivity to light will affect the overall system influencing the performance of the entire system. If there are fluctuations in lighting within the workspace, the tracking modules will require additional elements to counter, such as infrared cameras or the ability of the overall system to direct light on the object of interest. The software itself can also help to compensate for the light limitations by calculating a significant mean history image over a wider range of images, which would ultimately slow the entire system down.

The system can further be enhanced by using recursive means such as the Kalman filter to subtract the background data from the data of interest which will increase the robustness of the system but it would also slow the system down. The vision system is suited for case by case studies, optimizing the entire system based on individual workspaces and robot configurations. For example with variations in lighting the system at hand must be modified in both hardware and software. For systems that work underwater the system has to reduce the noise by calculating a mean history image over a narrower range of images to compensate for the fluidity of underwater images.

#### 5. CONCLUSIONS

As discussed in this paper, a hybrid vision system will ultimately reduce computational load making the entire system more efficient and perhaps more robust. The hybrid machine vision system presented in this paper works by first identifying the object of interest with a neural network module. Next, the system passes on the object of interest coordinates within a set frame to a combined robust tracking module. Within the tracking module there is a variation of a CAMSHIFT algorithm and a variation of a MHI module, presented together to track the desired object with respect to orientation and general direction. While tracking the object may tend to move towards the edges of the frame at which the robotic system and machine vision system manipulators interact to intercept or carry out a desired task.

#### 6. FUTURE CONSIDERATIONS

Future work involves building a hardware prototype for testing purposes in order to tweak the algorithm to work within

acceptable variations of lighting scenarios. Also work may be carried out on implementing another camera into the system to have stereo vision or infrared vision. The training of the neural network will be modified to ease training with respect to time.

#### 7. ACKNOWLEDGEMENTS

Our special thanks to Can Dede, Ph.D. candidate at Florida International University, Department of Mechanical Engineering. Special thanks also extend to the Open Computer Vision team at Intel Corporation as well as the OpenCV Yahoo! group (opencv@yahoogroups.com), for providing tools and insight into applying the OpenCV technology.

#### 8. REFERENCES

- [1] Jorge Angeles J. *Fundamentals of Robotic Mechanical Systems*, Second Edition Springer Press, 2002.
- [2] Murphy R. *Introduction to AI Robots*, MIT Press, 2000.
- [3] Kulkarni A. *Computer Vision and Fuzzy-Neural Systems*, Prentice Hall, 2001.
- [4] Gonzalez and Woods. *Digital Image Processing*, Second Edition, Prentice Hall, 2002.
- [5] Duda R. and Hart P. *Pattern Classification and Scene Analysis*, John Wiley & Sons Inc., 1973.
- [6] Viola P. and Jones M. *Rapid Object Detection using a Boosted Cascade of Simple Features*, IEEE, 2001.
- [7] Lienhart R. and Maydt J. *An Extended Set of Haar-like Features for Rapid Object Detection*, IEEE, Vol. 1, pp. 900-903, September 2002.
- [8] Hilton Head, *Vision Technology Research Could Lead to Profound Changes in Human-Computer Interaction in the Home and Workplace*, South Carolina, <http://www.intel.com/research/mrl/news/videosoftware.htm>, June 13, 2000.
- [9] Bradski G. and Davis J. *Motion segmentation and pose recognition with motion history gradients*, *Machine Vision and Applications*, 2002.
- [10] Bradski G. *Computer vision face tracking as a component of a perceptual user interface*. In *Workshop on Applications of Computer Vision*, pages 214-219, Princeton, NJ, October 1998.
- [11] Shawe-Taylor J. and Cristianini N. *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.
- [12] MacKay D., *Information Theory, Inference and Learning Algorithms*, Cambridge University Press, 2004.
- [13] Dayan P. and Abbott L., *Theoretical Neuroscience, Computational and Mathematical Modeling of Neural Systems*, MIT Press, 2001.
- [14] Barczak A. and Dadgostar F. *Real-time hand tracking using a set of cooperative classifiers based on Haar-like features*, *Institute of Information and Mathematical Sciences, Massey University at Albany, Auckland, New Zealand, Lett. Inf. Math. Sci.*, Vol. 7, pp 29-42, 2005.
- [15] Davis J. *Recognizing Movement using Motion Histograms*, MIT Media Laboratory Perceptual Computing Section Technical Report No. 487, April 30, 1998.