USING HAND RECOGNITION IN TELEROBOTICS

by

SAAD HAFIANE

FINAL PROJECT REPORT

Submitted to the Department of Electrical & Electronic Engineering
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronic Engineering)

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

USING HAND RECOGNITION
IN TELEROBOTICS

by

Saad Hafiane

A project dissertation submitted to the
Department of Electrical & Electronic Engineering
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronic Engineering)

Approved:

_____

Dr Aamir Saeed Malik
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

May 2012

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

_____

Saad Hafiane

ABSTRACT

The objective of this project is to recognize selected hand gestures and imitate the recognized hand gesture using a robot. A telerobotics system that relies on computer vision to create the human-machine interface was build. Hand tracking was used as an intuitive control interface, as it represents a natural interaction medium. The system tracks the hand of the operator and the gesture it represents, and relays the appropriate signal to the robot to perform the respective action, in real time. The study focuses on two gestures, open hand, and closed hand, as the NAO robot is not equipped with a dexterous hand. Numerous object recognition algorithms were compared and the SURF based object detector was used. The system was successfully implemented, and was able to recognise the two gestures in 3D space using images from a 2D video camera.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

2D	Two dimensions
2NN	Two nearest neighbours
3D	Three dimensions
Fps	Frames per second
FLANN	Fast library for approximate nearest neighbour
NAO	Autonomous, programmable humanoid robot developed by Aldebaran Robotics
OpenCV	Open Source Computer Vision library
RANSAC	RANdom SAmple Consensus
QVGA	Quarter Video Graphics Array (320x240)
VGA	Video Graphics Array (640x480)

CHAPTER 1
INTRODUCTION

This study implements a telerobotics system that relies on computer vision to create the human-machine interface. 3D Hand tracking through a 2D video camera is used as a natural and intuitive control interface. The system tracks the hand of the operator in the 3D space and the gesture it represents using monocular vision, and relays the appropriate signal to the robot to perform the respective action, in real time.

1.1 BACKGROUND OF STUDY

In robotics, the subdomain that covers using a mainly wireless connection to remote control a robot is called telerobotics. [1] [6] [7] [8-11]. It is the marriage of two disciplines, teleoperation and telepresence [1] [5].

Teleoperation aims to give the control of a robot to an operator from a distance in cases where the physical presence of the person is cumbersome or hazardous and creating an autonomous robotic system intricate. In order for teleoperation to be a viable option, it has to offer a level of manipulation close to the intuitive human hand motion. To deal with this area, telemanipulation is needed. Telemanipulation, is the combination of two words tele which means remote, and manipulation. Therefore telemanipulation can be defined as teleoperation where a human operator remotely controls a mechanism manually, in order to manipulated the environment where the robot is present [1] [6] [7] [8-12]. In recent years, many researchers have focused on creating telerobotics system using robots equipped with dexterous robotic hands in order to allow the operator to manipulate the remote environment intuitively. Different strategies were followed in order to control the robotic arm, as some systems used joysticks or space ball to operated them [2][3], when others used trackers fixed to wrist of the operator, in order to map the motion of the arm of the operator to the robotic arm [4] [12] [13]. Computer vision was used in other telerobotics systems to allow the operator to manipulate the robot without requiring a physical tool [6]. Often the hand gesture is mapped from the operators hand to the robot using a dataglove, as it is easy and robust [1] [12][13].

1.2 PROBLEM STATEMENT

Hand gestures can be used as a more natural and convenient way for human robot control.

The direct interface of hand gestures provides us a new way for communicating with the remote environment through the robotic medium.

1.3 OBJECTIVES

- Recognize selected hand gestures.
- Imitate recognized hand gesture using a robot.

1.4 SCOPE OF STUDY

We will be considering Monocular vision only, as it is compatible with all robots with vision, either 2D or 3D.

Initially only planar hand gesture and position will be studied. The study will focus on two gestures, open hand, and closed hand, as the NAO robot is not equipped with a dexterous hand.

1.5 RELEVANCY OF THE PROJECT

With the development of robotics, more and more focus is put on human machine interaction.

Not all robots with vision use stereo vision, therefore for a broader implementation, monocular vision is important.

With the increase in performance of embedded processors, and Systems-On-Chip, one of the main drawbacks of computer vision is being reduced.

CHAPTER 2
LITERATURE REVIEW

2.1 A ROBOT ARM/HAND TELEOPERATION SYSTEM WITH TELEPRESENCE AND SHARED CONTROL [1]

The study investigates how effective telepresence is in application by describing a teleoperation based on a master slave paradigm. The sensorial feedback system consists of sensors mounted on the fingers of the robotic arm, that measure the forces of contact exerted and sends them to the force feedback device in order to reproduce them on the hand of the human operator. A visual feedback was also used to immerse the operator in the remote environment through a 3D display helmet. [1].

The components used in this system were:

Component	Usage
CyberGrasp	Force feedback device
3D display helmet	Vision feedback device
Staubli RX60	Robot arm
HIT/DLR	Dexterous hand
Dataglove	Finger position input device

The position and velocity of the joints were computed by mapping the input from the dataglove. The Space Mouse is used as the 3D input device to control the Staubli RX60 robotic arm. The results of the experimental study are that the teleoperation system built can be intuitively controlled in addition to being productive [1].

2.2 DATA FUSION FOR 3D GESTURES TRACKING USING A CAMERA MOUNTED ON A ROBOT [3]

This study aims to track 3D gestures of both arms by using monocular vision equipped mobile robot, combined with an assisting particle filter using fused data from several features [3].

The arms are modeled by geometrical shapes approximations, where degenerate quadratics were used. The systems projects the shape and contour of the model in

addition to local surface features. Based on the projection the particles of the particle filter are confirmed, in order to compute a better mathematical model. [3].

In the case of monocular vision, 3D pose is usually estimated using geometrical volumes models. The authors state that particle filtering produces suitable results in this situation. In addition moving edges are favored in the case of busy backgrounds [3].

Clothes of particular color distributions were considered, as they natural markers. To do so histogram matching was used [3].

2.3 GAME INTERFACE USING HAND GESTURE RECOGNITION [4]

The authors use difference image entropy using a stereo camera to achieve real-time hand gesture recognition [4].

Existing systems use hand detection primarily with some type of marker. The system discussed in this study, however, uses a real-time hand image recognition system. In the detection step, a depth map using a sum of absolute differences was implemented based on the acquired right-left image using a stereo camera. This system detects a foreground object and perceives it as a hand.

The hand gesture recognition system uses the difference image entropy of the input image and the average image. To test the performance of the system, an experiment using the hand gesture 240 database was conducted. The proposed method shows an average recognition rate of 85%. Using the proposed method, a Chinese chess game based on hand gesture recognition was implemented [4].

2.4 SOM-BASED HAND GESTURE RECOGNITION FOR VIRTUAL INTERACTIONS [5]

The paper proposes a new hand gesture recognition method using self-organizing map (SOM) with datagloves. The SOM method is a type of machine learning algorithm. It deals with the raw data sampled from datagloves as input vectors, and builds a mapping between these uncalibrated data and gesture commands. The results show the average recognition rate and time efficiency when using SOM for dataglove-based hand gesture recognition. A series of tasks in virtual house illustrate the performance of our interaction method based on hand gesture recognition [5].

The open and close hand gestures are easily found using the SOM method. On the other hand, the grasp is more difficult to recognize, the system is always confused by the grasp and hold. Although this system does not recognize the hold with good precision, it is powerful enough to work as a navigation or interface for virtual interactions [5].

2.5 REAL-TIME HAND GESTURE TELEROBOTICS SYSTEM USING FUZZY C-MEANS CLUSTERING [6]

The study aims to teleoperate a remote robot through TCP/IP in order to push a block remotely. Hand gestures are classified using C-Means as commands. The results of the tests that the system was subjected to, using 20 trials of a 12-gesture set of commands showed an effectiveness of 99.6% of the gesture described adequately, and a recognition accuracy of 100% of the gesture successfully classified. [6].

2.6 SUMMARY

Table 1 Comparison of reviewed systems

System	Key points	Limitations
A Robot Arm/Hand Teleoperation System with Telepresence and Shared Control	• Mechanical input sensor to measure hand gesture [1] • Helmet is used as the stereo video display device [1].	• Requires a mechanical input devices in order to operate • Hand gesture mapping to robot is often unnatural
Data Fusion for 3D Gestures Tracking using a Camera mounted on a Robot	• Combines particle filter tracking with edge cues, motion cues and Local color distributions [3].	• Computation expensive, "a PentiumIV-3GHz requires about 1s per frame" [3].
Game Interface using Hand Gesture Recognition	• Hand detection using sum of absolute differences and depth map [4]. • Hand recognition using difference image entropy [4].	• Uses Stereo vision [4], which can be limiting as most robots use single cameras as input.
SOM-based Hand Gesture Recognition for Virtual Interactions	• Uses dataglove as input device [5]. • Uses Self-Organizing Map Description to detect hand gesture [5].	• Requires the operator to wear a dataglove [5]
Real-time hand gesture telerobotics system using fuzzy c-means clustering	• Hand detection using threshold [6]. • Hand gesture recognition using feature vectors [6].	• Frame must contain only the hand of the operator [6].

CHAPTER 3
METHODOLOGY:

3.1 RESEARCH METHODOLOGY & RESULTS

The project is split in two major parts, the computer vision system responsible of recognizing and tracking a hand gesture, and the robotic control system.

3.1.1 Gesture Recognition:

In order to recognize a gesture the shape of the hand and the hand itself must be detected in the frame, to do so multiple object recognition algorithms exist. These algorithms generally fall in the following categories:

- Appearance-based methods: Use example images (called templates or exemplars) of the objects to perform recognition
- Feature-based methods: Interesting features in the object are extracted and described, then a search is used to find feasible matches between object features and image features.

The algorithms tested for gesture recognition are:

- Speeded-Up Robust Features (SURF) - (*Feature based method*)
- Template matching - (Appearance based method)

3.1.1.1 ALTERNATIVES COMPARISON

The results were obtained using a Core2 Quad 2.67GHz Intel processor, running OpenCV2.1 on Windows 7 64bits.

a) SURF [14]:

SURF (Speeded-Up Robust Features) is a scale- and rotation-invariant detector and descriptor. To extract the points of interest, the method relies on integral images in the summation for image convolution in order to approximate the determinant of hessian, in addition to 2D Haar wavelets. For describing the interest points, SURF uses the distribution of the intensity neighboring the point the Haar wavelet of first order [14].

Due to the texture nature of the hands, the number of features extracted from the hand is not always sufficient to identify it in the frame, furthermore during our testing the frame rate was between 7 to 0.8 FPS, depending of the Hessian threshold and the size of the frames.

The surf program was tested, at different hessian thresholds, 300, 500, 1000, and on the two gestures that we are studying, the open hand gesture, and the closed hand gesture.

To match the features extracted between the gesture template and the frame, the naïve nearest neighbor method based on the squared distance was used.

Once the matched points are found, the RANSAC (RANdom SAmple Consensus) algorithm is used to remove the erroneous matches before computing the 3 by3 homography matrix that describes the relation between the template and the match in the frame.

Using the homography, a bounding box in drawn around the match.

i. Open hand gesture:

In the figure on the left, the surf feature extraction and description was subjected to a hessian threshold of 300, therefore allowing more points for relatively weaker features to be selected. The relatively large number of features extracted and described requires computing power, therefore yielding a frame rate of 2pfs.

In figure on the right, the surf feature extraction and description was subjected to a hessian threshold of 500, therefore dropping some points that are relatively weaker features. This can be seen when comparing the center of the palm, which contains weaker features, between the left and right figures. The resulting frame rate was 5fps.

When using a hessian threshold of 1000, most of the feature points were dropped, resulting in the decrease of the number of matches, causing the gesture not to be recognized in the frame.

The size of the frames taken from the camera is QVGA 320x240 @ 30 fps.

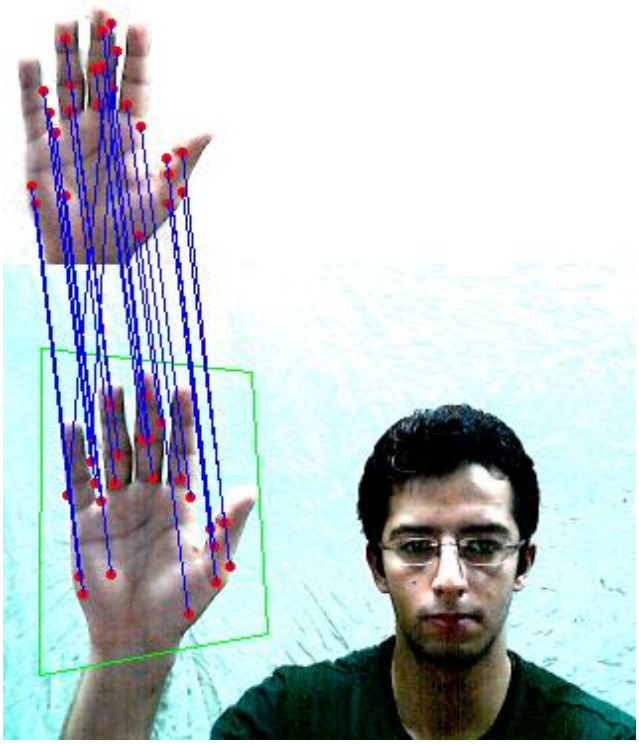


Figure 2. Open hand gesture recognition using SURF, hessian threshold=500, QVGA@5fps

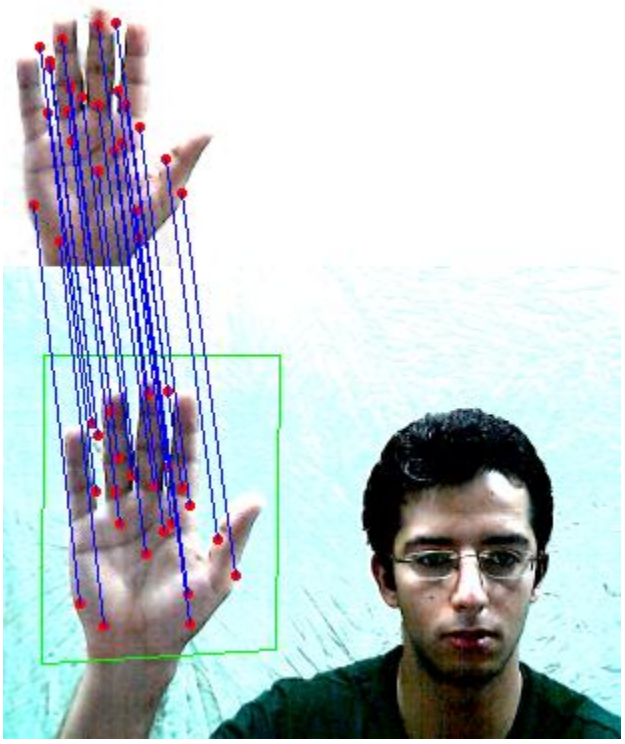


Figure 1. Open hand gesture recognition using SURF, hessian threshold=300, QVGA@2fps

ii. Closed Hand Gesture:

In the figures above, the surf feature extraction and description was subjected to a hessian threshold of 500, therefore allowing more points for relatively weaker features to be selected. The relatively large number of features extracted and described requires computing power, therefore yielding a frame rate of 6pfs.

The hessian threshold of 300, caused the number of feature points extracted to be large, however most of the points extracted were erroneous matches.

When using a hessian threshold of 1000, most of the feature points were dropped, resulting in the decrease of the number of matches, causing the gesture not to be recognized in the frame.

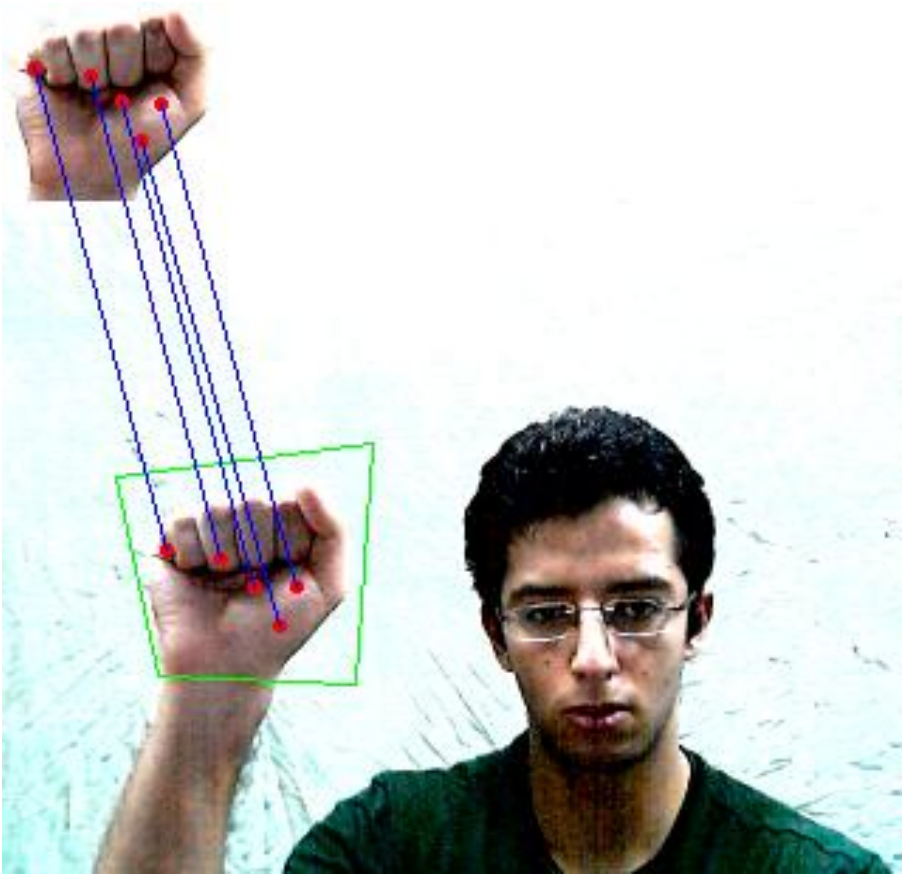


Figure 3. Closed hand gesture recognition, hessian threshold=500, QVGA@6fps

b) Template matching:

Template matching works by sliding the template over the searched image and computing the error between the shifted template and the image. This method can be computing expensive, and suffers from lighting variance, scale variance, rotation variance.

Template matching works by sliding through image, comparing the overlapped patches of size $w \times h$ against the gesture template, using the normalized squared difference method, and storing the comparison results to a result matrix.

Here is the formula for the different normalized squared difference method:

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 . \sum_{x',y'} I(x+x',y+y')^2}}$$

Where:
$x = 0..W - w \;\; ; \;\; y = 0..H - h$
$x' = 0..w - 1 \;\; ; \;\; y' = 0..h - 1$

$I = (W \times H)\ image\ matrix$

$T = (w \times h)\ template\ matrix$

$R\ = (W - w \times H - h)\ result\ matrix$

The matches are the minimums of the result matrix, which denotes the location of the smallest difference between the template and the image.

The sizes of the frames taken from the camera are VGA 640x480 and QVGA 320x240 @ 30 fps.

i. Open hand gesture:

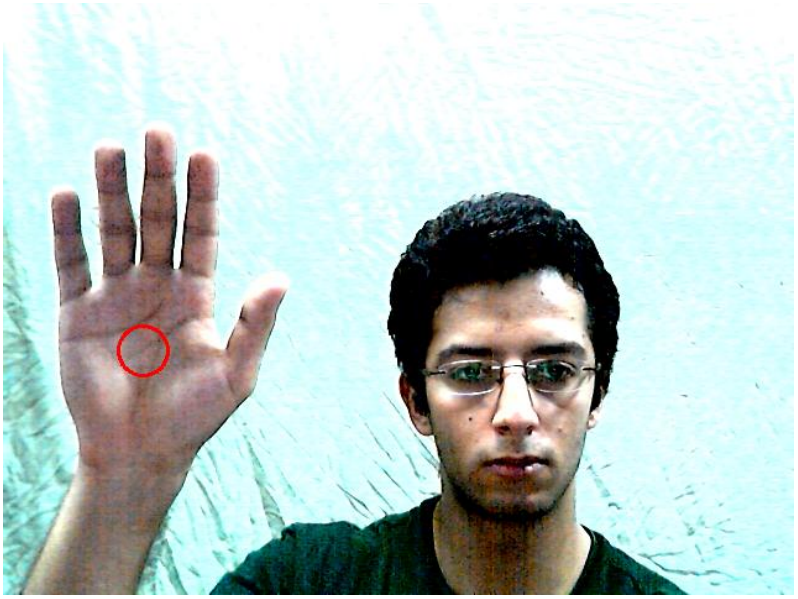


Figure 4 Open hand gesture recognition using template matching, threshold=0.2, QVGA@30fps

Figure 5 Open hand gesture recognition using template matching, threshold=0.2, VGA @14.7fps

In the figure above, template matching was used to locate the gesture template inside the video frame. Only the results with a normalized squared difference less the 0.2 were considered to indicate a positive match.

At QVGA resolution, the resulting frame rate was 30fps, which is the maximum frame rate of the camera used, which means that the frame rate of the camera is the bottle neck of system.

The average execution time of the matching routine is 14ms, which translates in theoretical maximum frame rate of 71.4 fps.

At VGA resolution, the resulting frame rate was 14.7fps.

ii. Closed hand gesture:

Figure 6 Closed hand gesture recognition using template matching, threshold=0.2, QVGA@30fps

Figure 7. Open hand gesture recognition using template matching, threshold=0.2, VGA @17.1fps

In the figure above, template matching was used to locate the gesture template inside the video frame. Only the results with a normalized squared difference less the 0.2 were considered to indicate a positive match.

At QVGA resolution, the resulting frame rate was 30fps, which is the maximum frame rate of the camera used, which means that the frame rate of the camera is the bottle neck of system.

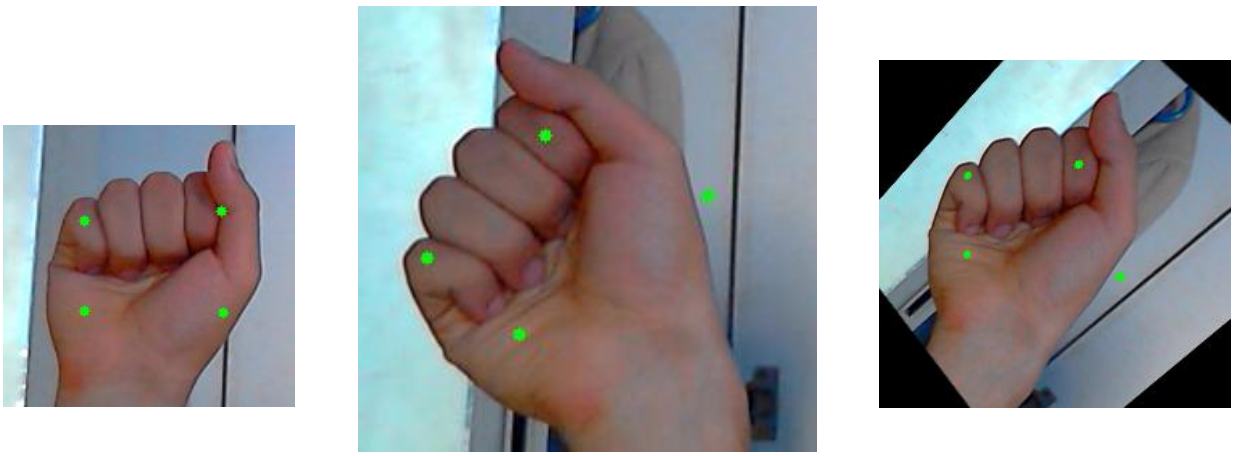
The average execution time of the matching routine is 12.1ms, which translates in theoretical maximum frame rate of 82.6 fps.

At VGA resolution, the resulting frame rate was 17.1fps.

The higher frame rate of the closed hand recognition compared to the closed hand is largely due to the smaller size of the template.

d) Template matching stabilized using optical flow tracking:

The aim in stabilizing the image before using template matching is to attempt eliminate the effect of size rotation and perspective change which considerably reduce the performance of the template matching method performance.

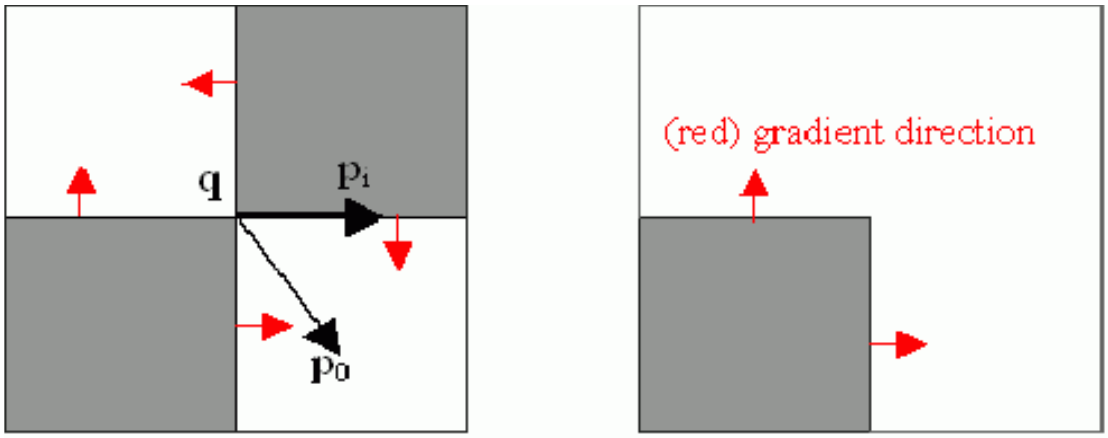


Original Tracked Stabilized

Figure 8. Stabilized closed hand

The figure above shows the original image with the points tracked on the right and the corresponding stabilized image on the left. The stabilization was used to correct the scale and orientation change.

Initially the template is selected and matched in the frame using template matching. Once a match is found, four points are selected in a rectangular pattern inside the match area. The position of the points is fine tuned using iterates to find the sub-pixel accurate location of corners, or radial saddle points, as shown in on the picture below.



The optical flow of the four tracked points is calculated using the Lucas-Kanade iterative method [17].

Once the new position of the tracked points is determined, a homography matrix, between the original points and the new tracked points is computed.

$$s_i \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim H \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$$

Using the homography matrix, the frame is transformed in order to recover the original matched template and therefore get an image where the area to be matched in of the same size, and orientation of the originally matched area.

Using the stabilized image, template matching is performed. In case of a match, the location of the match is confirmed. On the other hand if the template is not located, the original frame is used to try to find a match until the template is located.

i. Tracking with corrections

Using the stabilized image, template matching is performed. In case of a match, the location of the match is confirmed, and the tracked points are corrected and placed following the same pattern used to place the initial points.

During experimentation, the points were found to converge in a single point quickly, and therefore throwing the tracker off course. This is suspected to be due to the tracked points fine tuning needed each time the points are updated after a successful.

ii. Tracking without corrections

On the other hand if the template is not located, the original frame is used to try to find a match until the template is located. And the whole process is repeated again.

During experimentation, this version was found to be more stable; tracking lasted longer and was more reliable.

By using the optical flow data we were able to limit the effect of rotation, scale and perspective change on the performance of the template matcher while maintaining a frame rate similar to the original frame rate obtained when using the template matching alone.

e) Summary

The results were obtained using a Core2 Quad 2.67Ghz Intel processor, running OpenCV2.4.2 on Windows 7 64bits.

Algorithm	Resolution	Parameters	Frame rate (fps)		
			Open Hand	Closed Hand	Average
SURF	QVGA @30fps	hessian threshold = 300	9.08	9.17	9.125
		hessian threshold = 500	10.2	10.5	10.35
Template matching	QVGA @30fps	Uniform squared distance threshold = 0.2	30 (71.4)*	30 (82.6)*	30(77)*
	VGA @30fps		14.7	17.1	15.9
Stabilized template matching	VGA @30fps	Corrected	13.2	15.3	14.25
	VGA @30fps	Uncorrected	13.4	15.4	14.4

* [actual (theoretical)] where the theoretical is the result of the matching alone, independently from the camera frame rate which limits the frame rate of the system

3.1.2 Robot control

The control of the NAO robot is facilitated by the Naoqi library, in this project we are mainly focused on controlling the arm and grip of NAO.

This can be done in manual mode by sending the updated joints angles to NAO. This method is very low level and does not take into account the balancing and the physics involved.

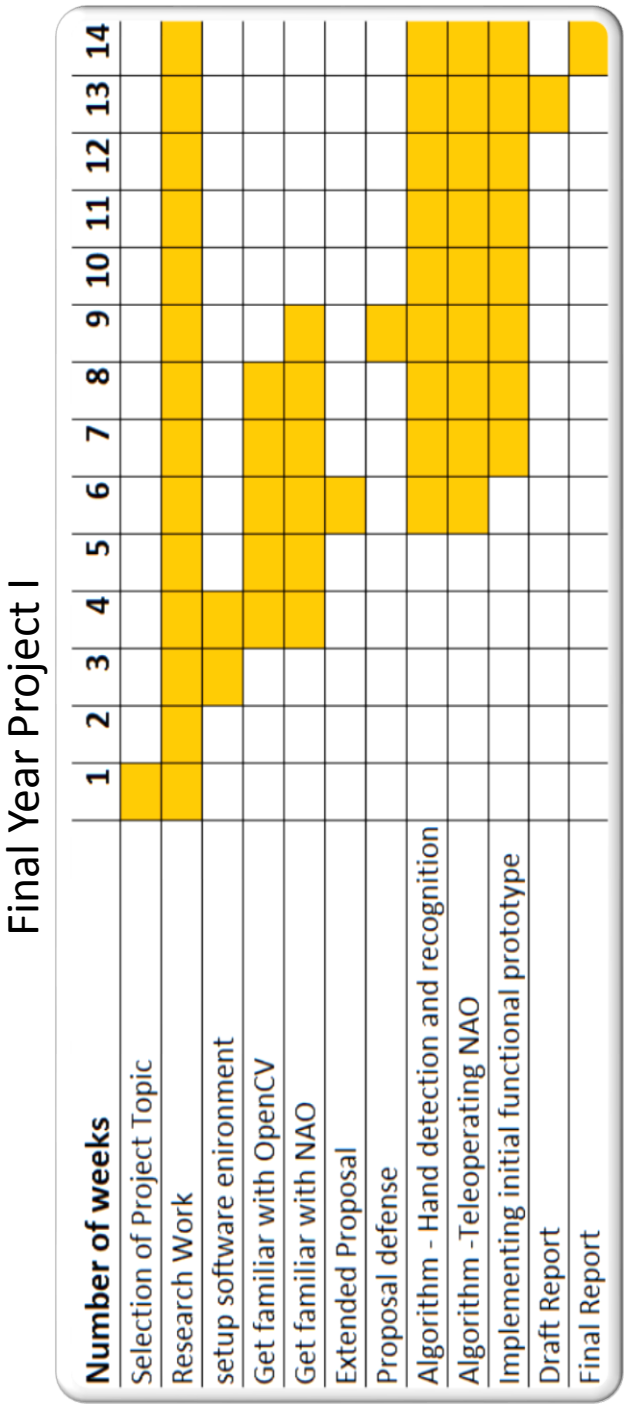
Another way to implement the motion control of NAO, is by enabling the whole body effector control, which basically takes the absolute Cartesian coordinates of the effector (in our case the arm) and controls the whole body to balance it and be able to perform the action needed.

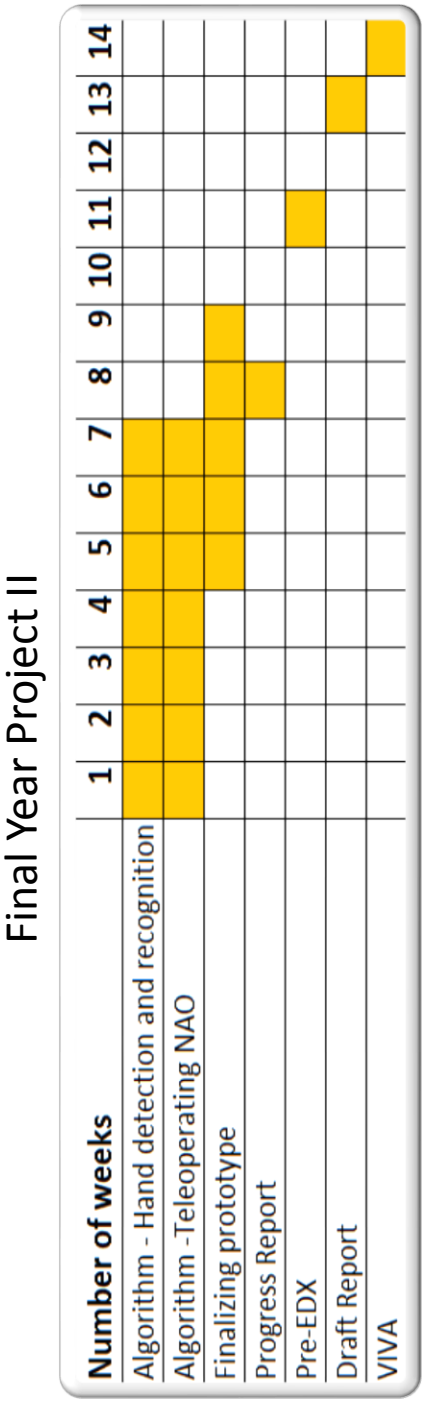
During experimentation, an initial single threaded program was developed, where the match was found and the motion was sent to the robot to be executed. Due to the Blocking nature of the whole body effector control of NAO, the performance dropped considerably and the frame rate dropped to 0.4fps. This was caused by the fact that the program waits for the motion to finish before processing the new frame.

To solve this problem a new multithreaded version was implemented where the gesture recognition and the robot telecontrol were implemented in separate threads that communicate through queues. The results were improved as the frame rate returned to normal, however the update speed of the robot motion was still slow, as the robot has to finish the motion before being able to execute a new one, even if the motion that is being performed is obsolete.

To counter this problem, the whole body effector control of NAO will be replaced by a manual version that is non-blocking and will therefore remove the lag created by the whole body effector control of NAO.

3.2 GANTT CHART





3.3 PROJECT ACTIVITIES

Final Year Project I

Documentation:

- Perform research and study
- Literature review
- Prepare extended proposal

Practical work:

- Select multiple alternatives of hand gesture recognition algorithm
- Analyse performance and fitness of algorithms
- Implement initial prototype

Final Year Project II

Documentation

- Progress report and technical report
- Final report

Practical work:

- Improve recognition and telecontrol algorithms.
- Finalize prototype
- Conduct experiments to test the performance of the system

3.4 TOOLS USED

- Monocular camera
- NAO Robot – See appendix
- The Open Computer Vision library (OpenCV)
- The Naoqi Framework – NAO's programing API

Python programming language – to implement the computer vision algorithm and control the NAO robot

CHAPTER 4
IMPLEMETATION AND TESTING

During initial experimentation, and alternatives comparison, the focus was to implement each alternative and to be able to choose the fittest alternative for our system. The NAO documentation recommends sending motion commands to the robot every 0.2 seconds at least, due to the mechanical inertia of the robots limbs and the transient response of the joints' actuators. Furthermore sending motion data to the robot at rate equivalent to 0.2 seconds between each command caused the robot's motion to jitter and the whole robot to shake. To insure a smooth motion of the robot multiple rates of motion data sending were tested and a rate of a command every 0.3 seconds was chosen, as it represented a good compromise between performance and smoothness of motion.

Based on a motion data sending rate of a command every 0.3 seconds, a minimum frame rate of 3.33 frames per second must be achieved by the computer vision system.

SURF was chosen as the method to use for the hand gesture recognition, as it achieves frames rates higher than 3.33 frames per second, in addition to being scale invariant, rotation invariant, and robust against lighting changes compared to template matching and stabilized template matching.

Furthermore the scale invariance of SURF features allows extracting more information from the frames, and moving the motion for the 2D plane to a 3D space.

4.1 THE RESULTING ALGORITHM

The results of the experiments conducted were used to create the algorithm and the structure of the system that will be used to detect and recognize hand gestures in the 3D space.

The system consists of two threads running in parallel, the first thread runs the computer vision program while the second controls the NAO robot.

The computer vision thread starts by loading the hand gestures' templates from the respective folders. The templates are easily created a program written specifically for that purpose, which automates capturing, creating, configuring, and managing the gestures' templates database.

Once the templates are loaded, the extraction of the SURF features of each template begins, and all SURF features are categorized by the gesture they indicate.

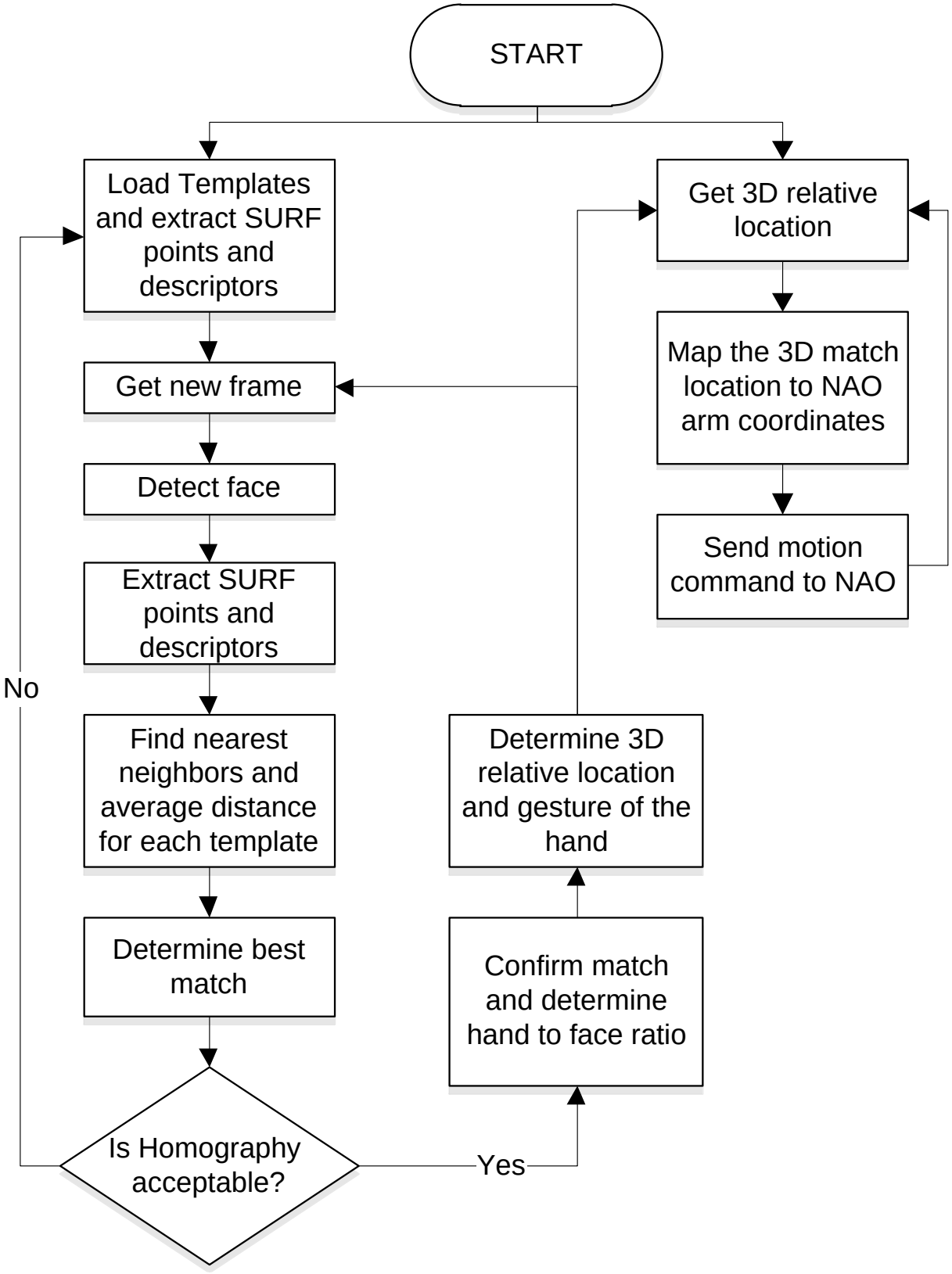
This concludes the initialization of the computer vision thread and initiates the recognition routine.

The system grabs a frame from the video camera, and then searches for a face inside it by using a frontal face Haar classifier. If a match is found, its size and location on the frame is stored.

Next the SURF features of the frame are extracted and stored. The hessian threshold for the SURF extraction was selected after testing a range of values, as will be shown in tests results.

4.2 FLOWCHART

The System proposed is described in the following flowchart:



4.3 SPEEDED-UP ROBUST FEATURES EXTRACTION

SURF (Speeded-Up Robust Features) [14] is a scale- and rotation-invariant detector and descriptor. To extract the points of interest, the method relies on integral images in the summation for image convolution in order to approximate the determinant of hessian, in addition to 2D Haar wavelets. For describing the interest points, SURF uses the distribution of the intensity neighboring the point [14]. The OpenCV implementation of the SURF algorithm was used in this system.

4.3.1 Hessian Threshold

The hessian value represents the local curvature of the image intensity, which describes the strength of the feature extracted and therefore is a good indicator on its robustness. The surf extraction was tested at different hessian thresholds, 50, 100, 200, 300, 500, and 1000. In other words the higher the hessian threshold the fewer and the stronger the features are.

The tests showed a direct relation between the hessian threshold and the frame rate achieved by the system.

Due to the fact that extraction of the SURF feature descriptors from the templates is only done during the initialization of the system, a hessian threshold of 100, with an octave value of 2 (with each next octave the feature size is doubled), and using 2 layers per octave was chosen, in order to extract a larger number of points.

On the other hand for the extraction of the surf points from the camera frames, a hessian threshold of 300 was selected as a compromise between the number of points extracted and the time needed for extraction.

4.4 HOMOGRAPHY EXTRACTION

4.4.1 Nearest Neighbor Matching

In order to determine the location of the match between the gesture template and the frame, a 2NN (2 nearest neighbor) search was used determine the two closest points to each point of the gestures templates. The FLANN library was used to perform the nearest neighbor search, and compute the respective distances [15].

During the 2NN search, all points with different Laplacians were ignored in order to speed up the search.

Using the 2NN points, the nearest neighbor was confirmed as a plausible match if the distance from it is smaller than 60% of the distance from the second nearest neighbor. The value of 60% was determined by trial and error.

The matched gesture is determined by the least average distance of the confirmed points.

4.4.2 Finding the Homography

Relying on the matched points using the nearest neighbor search, the RANSAC (RANdom SAmple Consensus) algorithm is used to remove the erroneous matches while computing the 3 by 3 homography matrix *H* that describes the affine transformation between the gesture's template and the match in the frame.

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = H \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{1}$$

4.4.3 Homography testing

Once the homography between the template points and the frame points is computed, four points that represent a square of length 100, are projected using the homography matrix.

Due to the fact that the appearance of the hand gesture changes significantly with its orientation, due to shadows and luminance variance, a practical match orientation range of ±30º was found to be acceptable in this situation. Based on that, the homography is tested by projecting four points that represent a square of side length 100, and testing the projected square.

An acceptable transformation is expected to keep the general form of the square; hence, the acceptable maximum ratio between the lengths of opposite sides of the projected square was set to 3.Furthermore the acceptable angle of rotation of the square was limited to the range ±30º.

Finally if any criteria is not satisfied the homography is not accepted, and no match is declared to be found.

4.4.4 Relative 2D Location

Using the confirmed homography, the center point of the matched gesture template is projected over the frame, giving therefore the absolute location of the match.

In order to send a location that will be easily translated to a motion that uses the full range of motion of NAO, a normalized location is computed and then sent.

$$x_{norm} = \frac{x}{x_{\max}}; y_{norm} = \frac{y}{y_{\max}} \tag{2}$$

4.5 DEPTH EXTRACTION

Using a 2D camera, the distance between an object and the camera can be easily determined if the size of the object and characteristics of the camera are available. However in our system the location of the hand relatively to the camera is invaluable. In order to extract the 3rd spatial coordinate of the hand, the location of the hand must be determined relatively to the body of the operator.

4.5.1 Face Detection

To determine the location of the hand relatively to the body, the location of the later must be determined. In this system the face was chosen as an approximated reference to the body.

In order to detect the face in the frame, Haar classifiers were used. We relied on the "*haarcascade_frontalface_alt.xml*" Haar cascade provided with the OpenCV library, to detect the frontal side of the face of the operator. Once the face is detected, its size is saved for later use.

4.5.2 Hand to Face Distance

In order to determine the hand to face distance, some approximations and assumptions have to be made. Numerous studies were conducted in the field of human body proportions, for centuries artists relied on approximation and calculated proportions, to describe the human body.

One of them is the fact that the actual length of the hand can be approximated with the length of the face.

Figure 1 shows the size difference of the hand and face in the frame due to the distance between them relatively to the camera lens.

The distance between the hand and the face can be calculated as follows:

$$\tan(\theta) = \frac{h}{D-d} = \frac{h'}{D} \Rightarrow \therefore d = D\left(1 - \frac{h}{h'}\right) \tag{3}$$

Where:

θ: angle of view of the camera occupied by the hand

h: actual height of the hand

h': apparent height of the hand

D: distance face to camera

d: distance hand to face

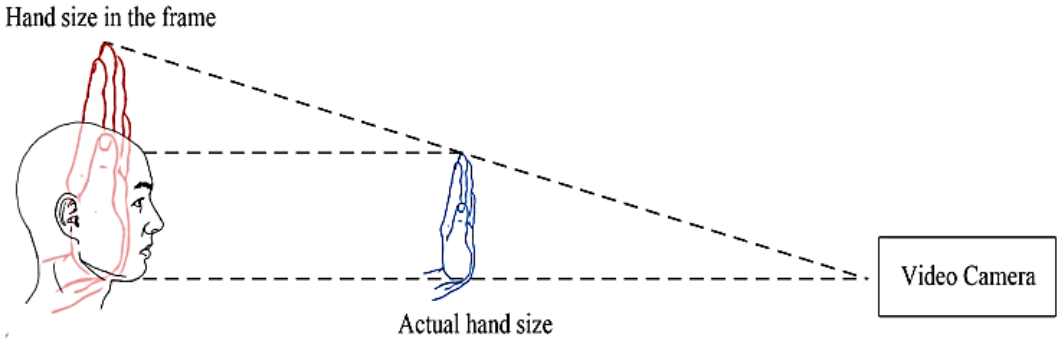


Figure 9. Optical Projection of the Hand

Equation 3 describes the absolute distance from hand to face, however in our system, the ratio of the hand to face distance to the maximum hand to face distance, in order to use the full range of motion of NAO even if the operator's size changes. By setting the operator at twice his arms' length from the camera, the normalized hand to face distance (depth) becomes:

$$D = 2d_{\max} \Rightarrow \therefore depth = \frac{d}{d_{\max}} = 2\left(1 - \frac{h}{h'}\right) \tag{4}$$

4.6 PERFORMANCE OF THE SYSTEM

In order to test the performance of the system created, two aspects were taken into consideration. First the objective performance of the system, which consist of the gathering the data about the system's frame rate performance, positive matches rate, the robotic system's lag compared to the motion of the operator and others.

Finally the subjective performance of the system was tested which consists of the perceived responsiveness of the system, the ease of operation, the learning curve faced while using the system for the first time, and how intuitive is it to perform a practical task.

In order to measure the objective and subjective performance of the system two types of tests were conducted:

a. Synthetic tests to get the processing performance of the system.
b. Task oriented tests where the robot had to successfully perform a practical task.

The results were obtained using a Core2 Quad 2.67Ghz Intel processor, running OpenCV2.4.2 on Windows 7 64bits.

4.6.1 Synthetic test results

Table 2 shows the frame rates achievable by the system for different hessian thresholds. It can be clearly seen that the frame rate increase with the increase of the hessian threshold

Table 2 Effect of the hessian threshold on the frame rate performance

Algorithm	**Resolution**	**Hessian threshold**	**Frame rate (fps)**		
			Open Hand	***Closed Hand***	***Average***
SURF	VGA @30fps	100	6.15	6.05	6.10
		300	9.17	9.08	9.125
		500	13.35	13.20	13.275
		1000	16.10	16.24	16.17

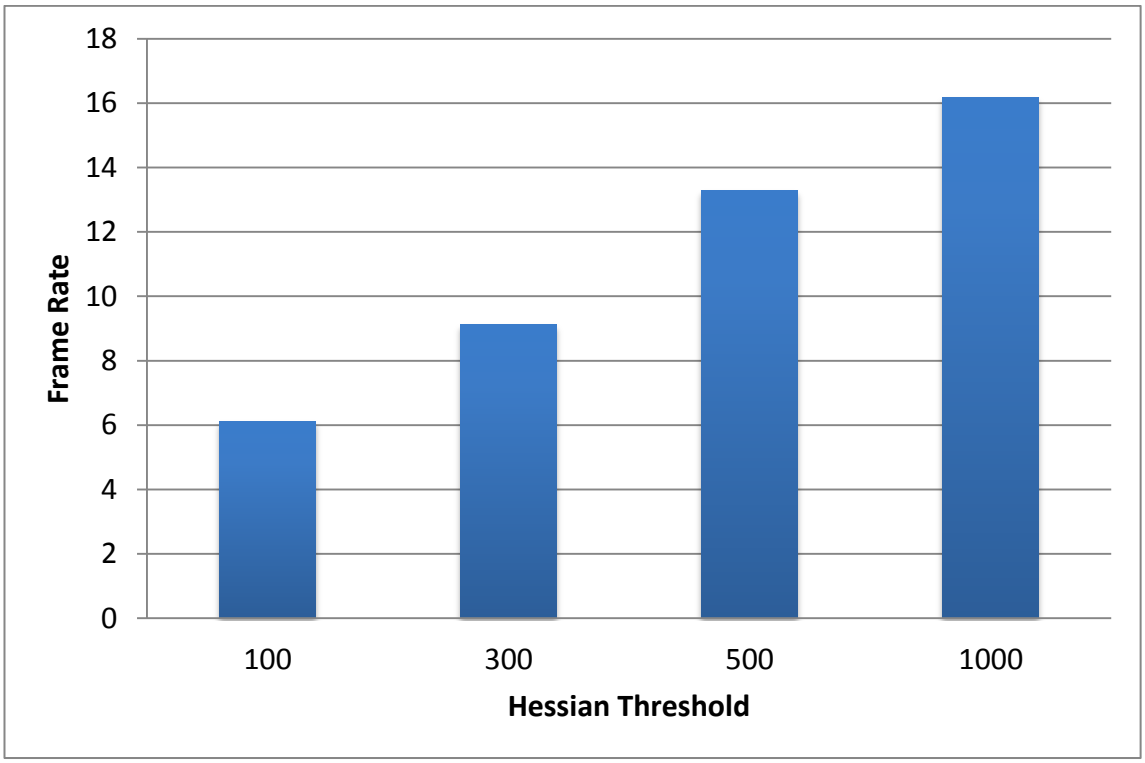


Figure 10 Effect of the Hessian Threshold on the Frame Rate Preformance

Table 3 shows the repeatability rates achievable by the system for different hessian thresholds. It can be clearly seen that the repeatability rate decrease with the increase of the hessian threshold.

Is this experiment, repeatability refers to the ability to repeat the procedure of extraction and matching on different frames and still get the same results, in other words it represents the robustness of the system.

Table 3 Effect of the hessian threshold on the matching repeatability

Gesture	Hessian threshold	Number of frames with positive match	Repeatability (%)	Average frame rate (fps)
Open hand	100	725	72.5	6.15
	300	678	67.8	9.17
	500	586	58.6	13.35
	1000	385	38.5	16.10
Closed hand	100	705	70.5	6.05
	300	667	66.7	9.08
	500	529	52.9	13.20
	1000	332	33.2	16.24

*Tests conducted at VGA@30fps over 1000 frames

Figure 11 Repeatability results summary

Hessian threshold	Average Repeatability (%)
100	71.5
300	67.25
500	55.75
1000	35.85

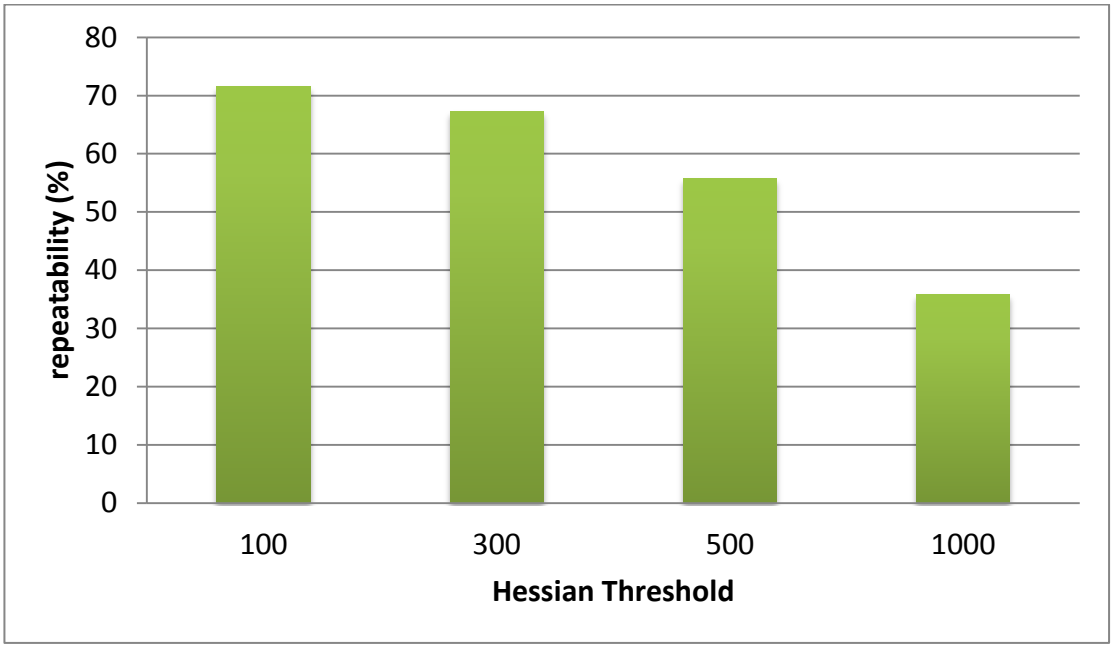


Figure 12 Effect of the hessian threshold on the matching repeatability

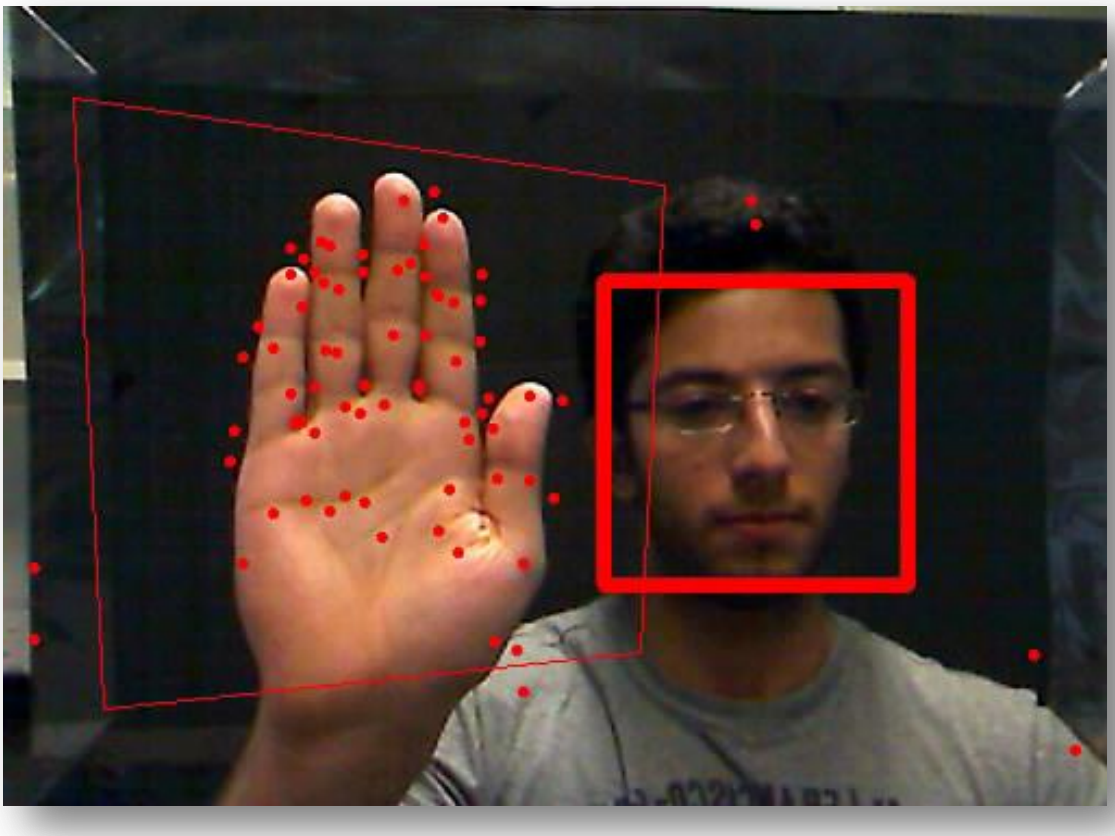


Figure 13 Open hand Gesture Recognized

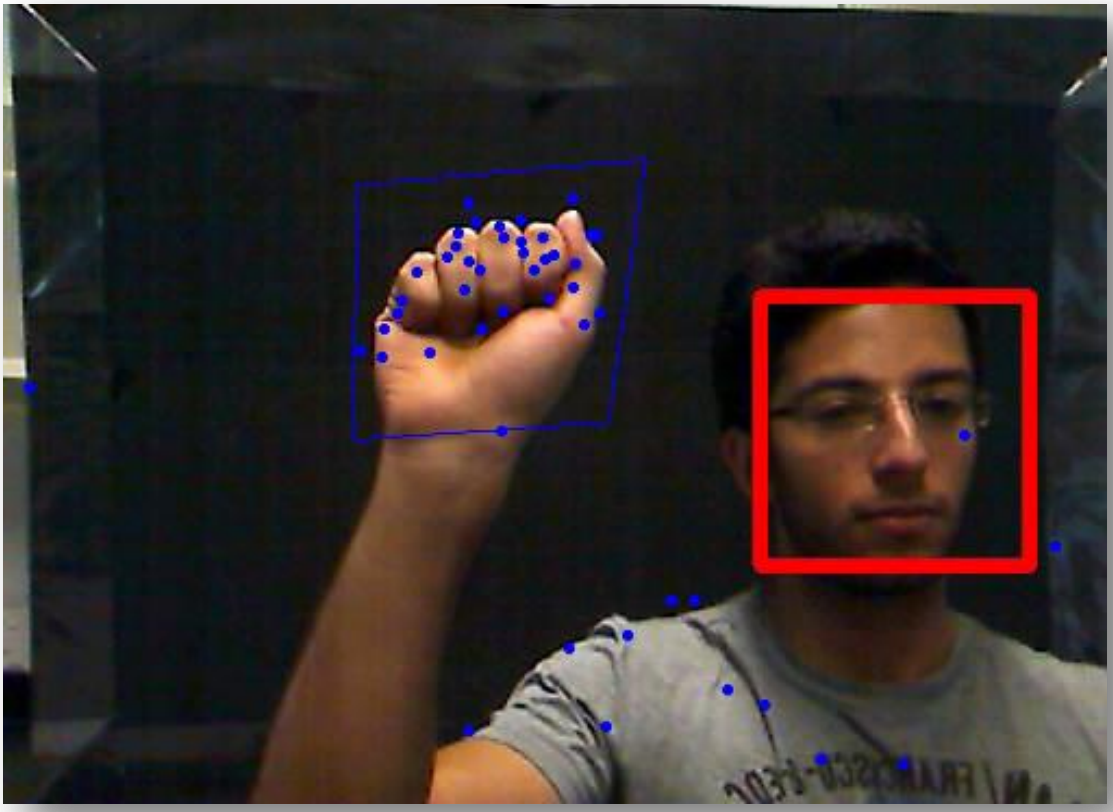


Figure 14 Closed hand Gesture Recognized

4.6.2 Task Oriented Tests

Multiple tasks were conducted in order to test the telerobotics system created.

Tasks included picking up an object that was handed to the robot, grasping it, and placing it in a boxed.

The test was conducted by an untrained operator over multiple trial runs in order to investigate: the performance of the system in a real world application, the intuitiveness of the system to an untrained operator, and the ease of learning and familiarizing with the system.

Five trial runs were conducted and were timed until the object was successfully transported to the box.

The chart in Figure 15 clearly shows a decrease in the time required to accomplish the task by an untrained operator.

The time required to grab the object and put it in the box took on the fourth trial a fifth of the time it took for the first trial. This clearly shows a very quick familiarization with the system.

These results demonstrate the ease and intuitiveness of the system developed, in addition to a good responsiveness.

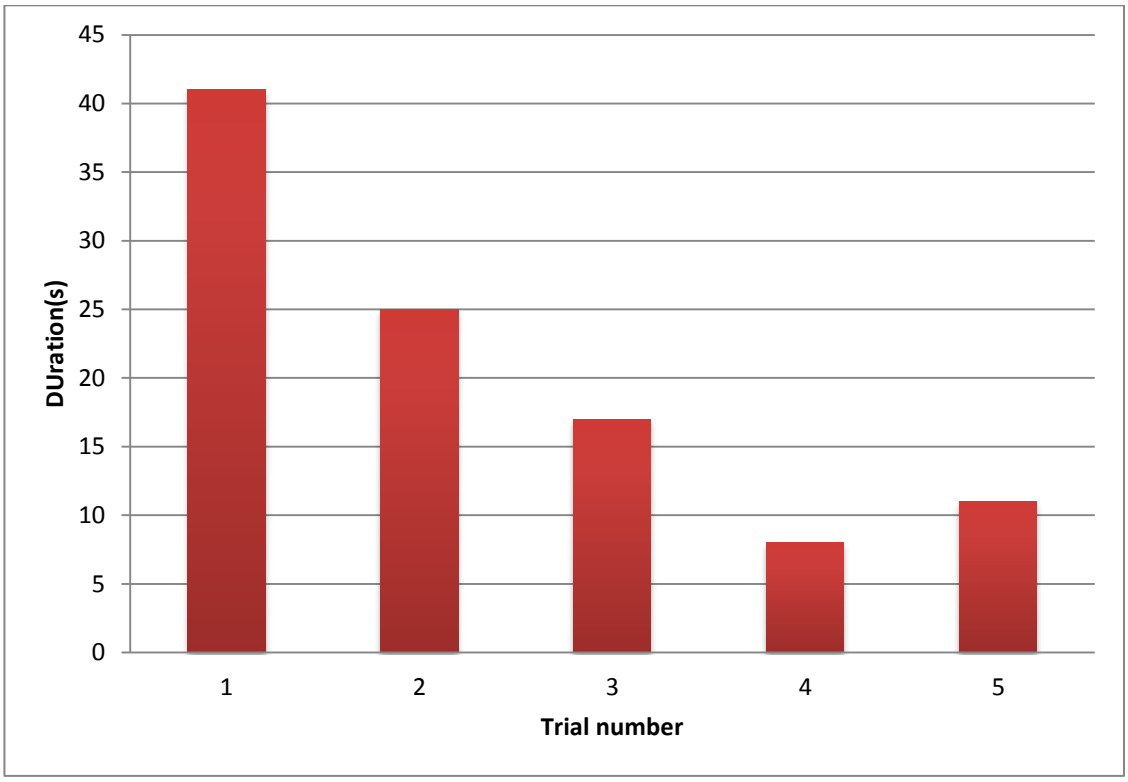


Figure 15 Successive trials durations

Figure 16 NAO grabbing and object and placing it in a box

CHAPTER 5
CONCLUSIONS AND RECOMMENDATIONS

The system was implemented using SURF (Speeded-Up Robust Features), the feature points of the template and the frames were matched using squared difference nearest neighbor, and unwanted points were removed using RANSAC during the homography computation. This resulted in a 9.125fps and a repeatability of 67.25%, using a hessian threshold of 300 on a VGA sized frame.

The SURF point were extracted using the OpenCV implementation of the surf algorithm. The FLANN library was used to perform the nearest neighbor search, and compute the respective distances [15].

Once the points were matched, points were selected based on their distances.

Next a 3x3 homography matrix was computed while ignoring outliers using the RANSAC algorithm. Further testing of the homography matrix was set up to check that the homography describes a direct affine transformation. Once the homography is confirmed to be correct, the matches are projected using the homography and the location of the matche is identified within the 2D plane.

Using Haar classfiers for the frontal face, the face is detected in the frame and, the approximate actual hand size is found. By using the relation between the actual hand size and the apparent hand size, the distance between the face and the hand is estimated. And therefore the depth data is extracted.

By combining the depth data with the 2D location, the hand gesture is recognized and located in the 3D space

The NAO robot was teleoperated using a TCP connection over Wi-Fi, and its body motion was set to self-balance during the arm motion in order to maximize the arm motion range. Due to the blocking nature of the whole body effector control of NAO, it will be replaced by a manual non-blocking method to remove the lag and improve the performance.

The program was implanted in a single thread fashion initially, but was later made into a multithreaded version in order to improve the performance.

As a recommendation regarding the methods used to recognize the gestures we note the following; due to the high computation requirement of the SURF algorithm it would be recommended to use a GPU implementation of the algorithm as the SURF algorithm is highly parallelizable [16], as the frame rate can reach 105fps [16]. In addition a new feature extractor/descriptor called FREAK (Fast Retina Keypoint), can provide better results [30] compared to SURF.

REFERENCES

[1] Haiying Hu, Jiawei Li, Zongwu Xie, Bin Wang , Hong Liu and Gerd Hirzinger , "A Robot Arm/Hand Teleoperation System with Telepresence and Shared Control", Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics Monterey, California, USA, 24-28 July, 2005

[2] David Gouaillier, Vincent Hugel, Pierre Blazevic Chris Kilner, Jérôme Monceaux, Pascal Lafourcade, Brice Marnier, Julien Serre and Bruno Maisonnier "Mechatronic design of NAO humanoid", 2009 IEEE International Conference on Robotics and Automation Kobe International Conference Center Kobe, Japan, May 12-17, 2009

[3] Paulo Menezes, Frédéric Lerasle and Jorge Dias`, "Data Fusion for 3D Gestures Tracking using a Camera mounted on a Robot", Proceedings of the IEEE 18th International Conference on Pattern Recognition (ICPR'06), 2006

[4] Doe-Hyung Lee, Kwang-Seok Hon , "Game Interface using Hand Gesture Recognition", School of Information and Communication Engineering, Sungkyunkwan University, 300 Chunchun-dong, Jangan-gu, Suwon, Kyungki-do, 440-746, Korea

[5] Shuai Jin, Yi Li, Guang-ming Lu1, Jian-xun Luo, Wei-dong Chen, Xiao-xiang Zheng1, "SOM-based Hand Gesture Recognition for Virtual Interactions", IEEE International Symposium on Virtual Reality Innovation 2011, 19-20 March, Singapore

[6] Juan Wachs, Uri Kartoun, Helman Stern, Yael Edan, "Real-time hand gesture telerobotics system using fuzzy c-means clustering", WAC 2002, Fifth Biannual World Automation Congress, June 9-13, 2002, Orlando, Florida

[7] R. Marín, J. S. Sánchez, P. J. Sanz, "Object Recognition and Incremental Learning Algorithms for a Web-based Telerobotic System", Proceedings of the 2002 IEE International Conference on Robotics & Automation, Washington DC, May 2002

[8] Paolo Fiorini , Roberto Oboe . "Internet-based telerobotics: problems and approaches". Proceedings of the 8th International Conference on Advanced Robotics, 1997. ICAR '97.

[9] Liu Jianbang, Lai Xuzhi, Wu Min, Chen Xin. "Design of embedded Telerobotics system". Proceedings of the 27th Chinese Control Conference July 16-18, 2008, Kunming,Yunnan, China

[10] Costas S. Tzafestas, Nektaria Palaiologou, Manthos Alifragis. "Virtual and remote robotic laboratory: comparative experimental evaluation". IEEE transactions on education, vol. 49, no. 3, august 2006, p360 – 369.

[11] Bartlomiej Stanczyk , Martin Buss . "Development of a telerobotic system for exploration of hazardous environments". Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems September 28. October 2,2004, Sandal, Japan

[12] M.L. Turner, R.P. Findley, W.B Griffin, M.R. Cutkosky, and D.H. Gomez, "Development and Testing of a Telemanipulation System with Arm and Hand Motion", Proc. ASME Dynamic Systems and Control Division (Symposium on Haptic Interfaces for Virtual Environments and Teleoperators), DSC-Vol. 69-2, 2000, pp. 1057-1063.

[13] M. A. Diftler, R. Platt, Jr, C. J. Culbert, R.O. Ambrose, W. J. Bluethmann. "Evolution of the NASA/DARPA Robonaut Control System". Proceedings of the 2003 IEEE International Conference on Robotics & Automation. Taipei,Taiwan, September 14-19, 2003. 2543-2548

[14] H Bay, A Ess, T Tuytelaars, L Vangool. "Speeded-Up Robust Features (SURF)". Computer

Vision and Image Understanding (2008)

[15] Marius Muja and David G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration", in International Conference on Computer Vision Theory and Applications (VISAPP'09), 2009.

[16] N. Cornelis and L. Van Gool "Fast scale invariant feature detection and matching on programmable graphics hardware", Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshops, pp.1 -8 2008

[17] B. D. Lucas and T. Kanade (1981), "An iterative image registration technique with an application to stereo vision". Proceedings of Imaging Understanding Workshop, pages 121--130

[18] R. Marín, J. S. Sánchez, P. J. Sanz, "Object Recognition and Incremental Learning Algorithms for a Web-based Telerobotic System", Proceedings of the 2002 IEE International Conference on Robotics & Automation, Washington DC, May 2002

[19] Jesús Martínez del Rincón, G. Rogez, Carlos Orrite Uruñuela, "2D Shape-Skeleton Models for Human Tracking in Unconstrained Monocular Sequences", BMVA technical meeting on Articulated Human Motion, BMVA technical meeting Edited by Makris, D. and Galata, A., BMVA, October, British Computer Society, 5 Southampton Street, London, UK, (2009)

[21] Guo, J.-ming, & Nguyen, H.-son. (2011). "Hybrid Hand Tracking System". 18th IEEE International Conference on Image Processing (Vol. 127, pp. 557-560).

[22] Mott, M., & Rajaei, H. (2010). "Hand Detection and Tracking for Virtual Training Environments". The 2010 Spring Simulation Multiconference (pp. 1-5).

[23] Wen, J., & Zhan, Y. (2009). "Vision-Based Two Hand Detection and Tracking". ICIS '09 Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human. ACM New York, NY, USA ©2009

[24] Stergiopoulou, E., & Papamarkos, N. (2006). "A New Technique for Hand Gesture Recognition". 2006 IEEE International Conference on Image Processing, 2657-2660.

[25] Rokade, R., Doye, D., & Kokare, M. (2009). "Hand Gesture Recognition by Thinning Method". 2009 International Conference on Digital Image Processing, 284-287. IEEE.

[26] Coogan, T., Awad, G., Han, J., & Sutherland, A. (2006). "Real Time Hand Gesture Recognition Including Hand Segmentation and Tracking". (G. Bebis & R. Boyle, Eds.)Advances in Visual Computing, 1(April), 495-504. Springer Berlin / Heidelberg.

[27] Medrano, C., Igual, R., Martinez Del Rincon, J., & Orrite-Urunuela, C. (2008). "Multi-target tracking with occlusion management in a mean field framework". The Eighth International Workshop on Visual Surveillance - VS2008 (2008)

[28] Petersen, N., & Strieker, D. (2009). "Fast Hand Detection Using Posture Invariant Constraints". KI 2009 Advances in Artificial Intelligence 32nd Annual German Conference on AI Paderborn Germany September 1518 2009 Proceedings (pp. 106-113). Springer.

[29] Yuan, Q., Sclaroff, S., & Athitsos, V. (2005). "Automatic 2D Hand Tracking in Video Sequences". Computer (Vol. 1, pp. 250-256). IEEE.

[30] A Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast Retina Keypoint. In IEEE Conference on Computer Vision and Pattern Recognition, 2012. CVPR 2012 Open Source Award Winner.