FINAL YEAR PROJECT II

DISSERTATION

Hand Commanded Machine Interface (Data Glove)

Supervisor: Dr. Nidal Kamel
Co Supervisor: Dr. Aamir Saeed Malik

by

Amr Magdy Lotfy

11057

UNIVERSITI TEKNOLOGI PETRONAS

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

June, 2011

ABSTRACT

Human-Robot interface is a methodology that is heavily utilized in industrial and commercial applications such as to fulfill user needs. Different types of robotic frameworks are designed to fulfill those needs. The aim of the work presented in this paper is to simulate a robotic movement that emulates the movement of a human hand (i.e. arm and fingers section). The robotic hand comprises of three fingers. The main objective of this robotic hand simulation is to design a link between the pre designed data glove (FYPI) to a robotic hand and imitates the movement of a human hand.

This project presents a new simulation methodology for the human-machine interface of robotic frameworks and control systems. The interface uses dynamic hand gestures, relative arm position estimation in order to have control of the Humanoid Robot used (NAO) and provides a control and visualization interface between a human and NAO.

All the Important aspects to develop such an interface; image processing techniques, object tracking, colour tracking, motion detection and software filters; contrast, brightness and saturation have been explored. Seeking high ratio of faster and correct tracking have been achieved from the experiments.

Using "Python" and some other programming IDEs (integrated developing environments), Software simulation and hardware implementation to show the behavior of the Data Glove will be carried.

CERTIFICATION OF APPROVAL

Hand Commanded Machine Interface

by

Amr Magdy Lofty

A project dissertation submitted to the
Electrical Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL ENGINEERING)

Approved by,

____

(Dr. Nidal Kamel - Dr. Aamir Saeed Malik)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

December 2011

CERTIFICATION OF ORIGINALITY

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

AMR MAGDY LOTFY ELSAYED

ACKNOWLEDMENT

Praise be to ALLAH, the most gracious and the most merciful for his endless blessings throughout my life and the success HE granted me during this final year project. My appreciation to Universiti Teknologi Petronas for providing me with all the necessary assets and resources, not only to accomplish my tasks, but to enrich my character and knowledge.

My utmost gratitude and appreciation to my supervisors for guiding me, and the endless effort they exerted toward the achievement of this project despite their many other obligations.

My appreciation would be incomplete without giving credit to my family for their sacrifices coupled with their continuous encouragement and support and heading me towards the stars. Finally my appreciation is to my friends and those who I love.

Table of Contents

List of Figures

Code Snippets

CHAPTER 1: INTRODUCTION

1.1 Background of the study

This project involved in hardware and software parts. For hardware it is using The pre designed Data Glove (FYP I), robotic framework (NAO) humanoid robot, ATmega microcontroller with capacity of 32K implemented on an arduino developing kit and RF module for transmission and reception of data.

Figure 1 data glove

Figure 2 NAO

NAO

Nao is a humanoid robot developed by Aldebaran Robotics with huge number of actuators, motors and sensors. NAO has 27 degree of freedom (DOF), the robot can be programmed using C, C++ and python programming languages.

Arduino Microcontroller

Arduino ATmega is an open-source microchip integrated circuit that is intended for designers, letting them sense environment parameters (input) using sensors such as temperature pressure and flex, send commands to actuators ,leds and motors (output) according to uploaded sketch (script) using certain programming language.

RF Module

RF module is being used in order to send data wirelessly from surrounding to pc and vice versa using Radio frequency waves (Bluetooth or infrared) after configuration of certain parameters; parity, BUAD rate and flow control.

Flex sensors

A number of 5 flex sensors have been used in order to estimate the relative position of every finger in the user hand through bending force exerted by the user.

For software it is consisting of two main parts; server side and client side,

Server Side

The server is acting as an event listener to every change that occurs in the client side by the client including; hand motion and fingers motion. In order to move the robotic arm as synchronization with client action

Client Side

The client is recording an estimated value of the relative motion that is being done by the client hand as well as recording the gestures being done by the client fingers

- Recording the motion of the client hand is being done by image processing techniques including motion recognition, object detection and filtering in order to calculate the estimated relative position of the hand.

- Recording the gestures of the client fingers is being done using the pre designed Data Glove (FYP Part I) by using the flex sensors and the ATmega micro controller mounted on the glove.

1.2 Problem Statement

Wearable computing, this technology hasn't taken off yet because it is not applicable enough, but it's definitely coming. With accurate sensing capabilities, it can be used where remote movement emulation is necessary or helpful.

1.3 Objectives

The objective of this project is to use pre-designed data glove to control the motion of robotic arm by developing a link In between in order to deliver a new way of emulating the motion of human hand to a robotic (NAO) hand.

1.4Scope of Study

The scope of this project consists of research, simulation and analysis. The research is essential for better understanding on controlling and communicating. Machine Interface must be developed and Simulation must be carried out to determine the behaviour of the motion controller; exploring different types of functions and rules. Furthermore, analysis of the results as well as seeking high accuracy is included in the scope of this project.

CHAPTER 2: LITERATURE REVIEW

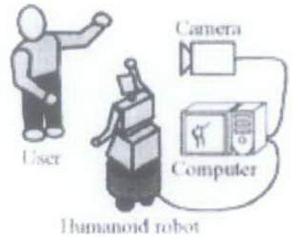
Controlling humanoid robotic frameworks using captured human motion data has been highly investigated in robotics fields, huge number of methodologies and algorithms have been explored, including gesture based human robotic interaction, computer vision techniques (object position estimation, blob detection, color detection and motion detection) to provide a vision based control HMI (human machine interface) for users, seeking high accuracy control over robotic frameworks.

A paper by Katsu Yamane and Jessica hodgins at Carnegie Mellon University[1] has presented an algorithm for modifying the joint angle, trunk and center of mass of humanoid model so that motion can be captured and emulated to a robotic framework

Another paper entitled "motion emulation system with humanoid robot and image processing"[2] by Hideaki Mitsugami, Kazuhiko Yamamoto, Kunihito Kato at Faculity of engineering, Gifu university, Japan has presented a similar way of transferring the motion from a human to a humanoid robot, using "YAMATO" humanoid robot , and they have been able to let it follows and emulate the user's hand motion making use of some image processing techniques as follows

Using the internal humanoid robot cameras to capture the image, estimating the relative position of the human arm after extracting the background image and sending commands to the humanoid robot accordingly

The user doesn't have to wear sensors or put markers, but he has to stay in a very clear background



Another related work has been conducted lately by cboirum on "December 7, 2011" entitled "real-time robot arm control"[3], he used a combination of blender(3d modeling software), python(programming language), processing(programing language) and arduino(micro controller) in order to control motion of a robotic arm Using "4 DOF" robotic arm with 4 RC servo motors for joint angle calculations, he has been able to control the robotic arm by blender commands in a real time simulation

Figure 3 robotic arm emulation

A paper entitled "Real-Time Path Planning Procedure for a Whole-Sensitive Robot Arm Manipulator"[4] by Edward Cheung and Vladimir Lumelsky at Yale university has considered the problem of real time sensor based robotic motion emulation techniques.

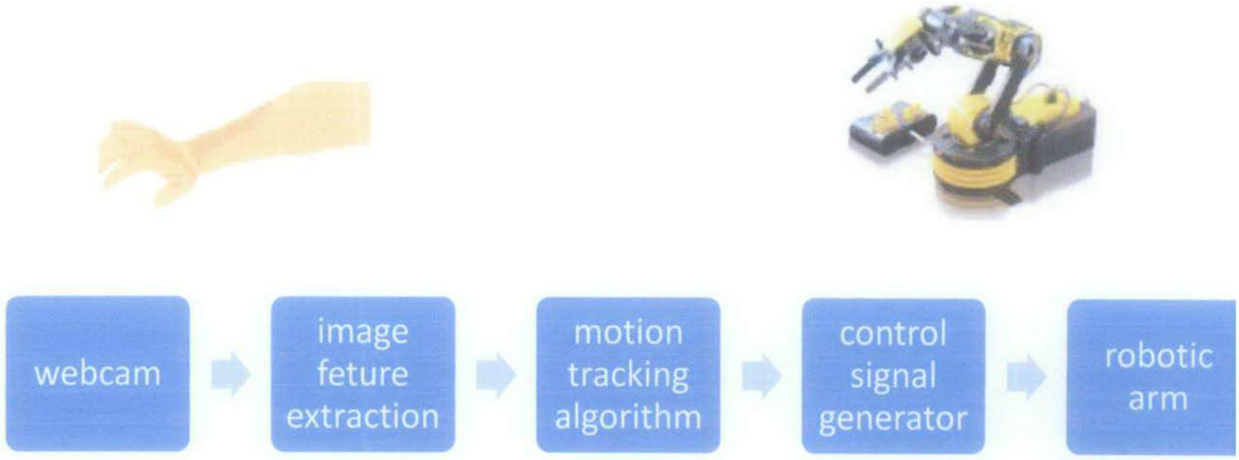
Using a developed proximity sensitive skin to transfer the human hand motion to a target robotic framework, at the same time, this developed methodology is still using sensory based techniques.

Another highly considerable approach has been done by IPI soft, IPI soft[4] company has developed a new methodology to capture the motion conducted by user's hand using "MICROSOFT KINECT " camera for high accuracy 3d object detection with depth calculation.

Figure 4 ipi motion capture using KINECT

Another similar work has been devolped at Cilicon Valley Campus entitled "real time imitative robotic arm control for home robot applications"[6] by Heng-tze Cheng, Zheng Sun and Pei Zhang

It aimed a remotly controlled low-cost system using Motion detection algorithm that is applied to the image captured using the webcam in order to detrmine the relative position of the human arm and transfer the motion to be emulated by the robotic arm



In Comparison with all related previous works and papers;

- our system directly controls the robotic arm without any kind of physical interaction between user hand and target object
- our system directly detects user's motion without using highly priced technologies or any additional accessories
- our system directly detects user's motion without using complicated hardware eg. Kinect by Microsoft
- no spoken commands have been used
- no physical markers have been used
- no specific clear background needed while capturing the image

CHAPTER 3: METHODOLOGY

3.1 Methodology

Firstly, All the important aspects to develop such an environment must be studied and explored, including the API (application programming interface) ,the microcontroller IDE (interface developing environment), sensing instruments ,serial socket communication, wireless socket communication ,microcontroller interfacing,

This project is divided into four major Steps which are;

Developing the configuration file for the Humanoid Robot (NAO)

- using Python programming language

Developing the hand motion listener

- using image processing techniques

Developing fingers motion listener

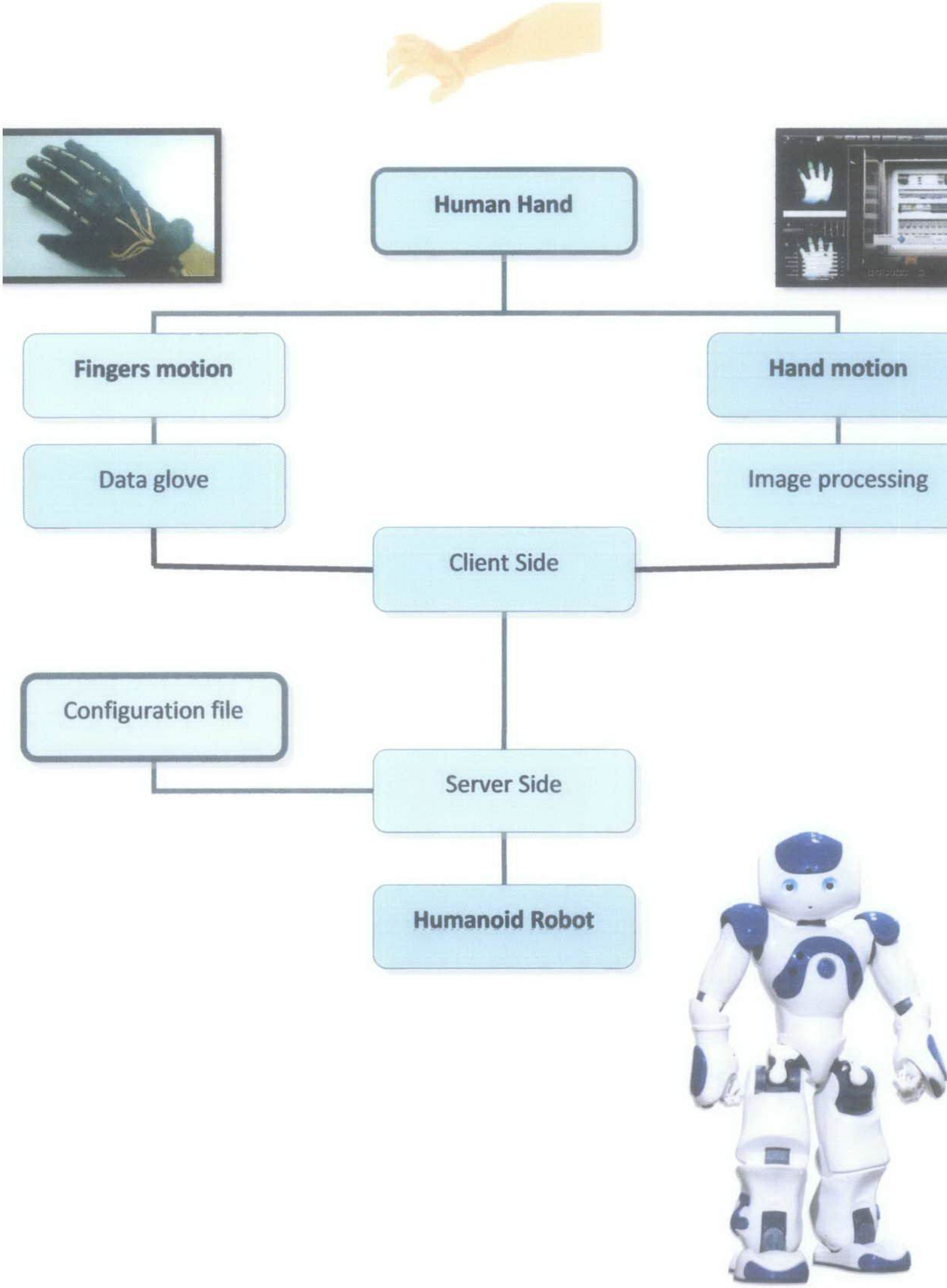
- using the pre designed Data Glove (FYP I)

Developing the server side

- in order to transmit the translated commands to the Humanoid Robot

This project will follow the following methodology to meet the objectives.

- As clarified in the following chart



- Client Side programming is being done using **Action Script programming language**
- Server Side programming is being done using **Python programming language**

As the application initiates, it captures the user image using the PC integrated webcam with a resolution of **420 x 320**, after that it analyses the image captured every frame (frame rate of **60** frame per second), then it tries to extract the pixels which have the same colour as the user skin from the image with a certain margin of error (range of **RGB** combinations), with the help of applying software filters to the image (brightness, contrast, saturation and blur) in order to ease the skin colour detection process, after determining the pixels containing user skin, the application compares every frame with the previous frame (with rate of **30** frames per second) and determines which pixels of the extracted image have changed in order to define which portion of the image had the motion (in order to determine the approximate position of the user hand that is continuously moving), then the application converts the portion detected into two variables; **X** and **Y** (ranging from **0 to 420** for **X** and from **0 to 320** for **Y**), and accordingly the application sends those two variables every frame (with frequency of **10** command per second) to the server side through the pc **TCP/IP** protocol on the Local Host (**IP** address: **127.0.0.1** , port no. **8000**), on the other hand, the server side is listening to those commands posted by the application (client side), accordingly, it converts values received for **X** and **Y** into angles (ranging from **20 to 60** for Shoulder Roll Angle and from **0 to 60** for Shoulder Pitch Angle), then it converts those angles into Radians, then it sends those values to the Robot (through the Ethernet port on IP address: **192.168.0.101** , port no. **9559**) in order for it to move its hand to the target angle at **10%** of maximum speed, at the same time, the server is also listing to changes occurred on the flex sensors mounted on the data glove, AT the glove side, the flex sensors readings are transmitted to the microcontroller mounted on the data glove, then the micro controller send those variables accordingly to the **RF** module (radio frequency Bluetooth module) then the module transmit those readings, then received at the PC through the Bluetooth communication port (COM PORT **5**, BUAD rate of **9600**), after those readings are received on the server side, the server analyses those readings and predict whether the user has closed his hand or opened it and sends commands accordingly (Close Hand and Open Hand) to the Humanoid robot.

CHAPTER 4: RESULTS AND DISCUSSION

Hand motion listener has been written using action script programming language in order to record an estimated value to human hand relative position

- capture user image through PC webcam at resolution of 420 x 320
- applying filters (brightness, contrast, saturation and blur) to the captured video feed in order to ease detection
- use image processing techniques to extract the pixels which have the same colour as the user skin from the image with a certain margin of error (RGB combinations)
- compare every frame with the previous frame (30 frames per second) and determine which pixels of the extracted image have changed in order to define which portion of the image is moving
- convert the portion detected into two variables; X and Y

Client side message transmitter has been written using action script programming language in order to transmit recorded values to the server

- send those two variables (X and Y) every frame (10 command per second) to the server side through the pc TCP/IP protocol on the Local Host

Server listener for the Humanoid Robot (NAO) has been written using python programming language in order to receive recorded values from the client side

- the server is listening to those commands posted by the application (client side)
- convert values received (X and Y) into angles (Shoulder Roll Angle and Pitch Angle)
- convert those angles into Radians
- Configuration file for the Humanoid Robot (NAO) has been written using python programming language in order for the server to be able to connect to the humanoid robot

Command handler for the Humanoid Robot (NAO) has been written using python programming language in order to send motion commands to be done by the humanoid robot

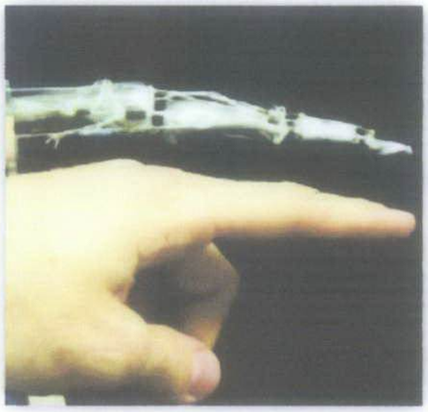
- send those converted values (radians) to the Robot
- The Robot moves its hand to the target angle at 10% of maximum speed

The server is also listening to changes occurred on the flex sensors mounted on the data glove, AT the glove side, **Fingers motion listener** has been written using action script programming language in order to record an estimated value to human hand fingers gestures

- flex sensors readings are transmitted to the microcontroller mounted on data glove
- micro controller send those variables accordingly to the RF module
- the RF module transmit those readings to the running server on the PC
- server receives readings at the PC through the Bluetooth communication port
- server analyses those readings and predict whether the user has closed his hand or opened it
- server sends commands accordingly (Close Hand and Open Hand) to the Humanoid robot (NAO).

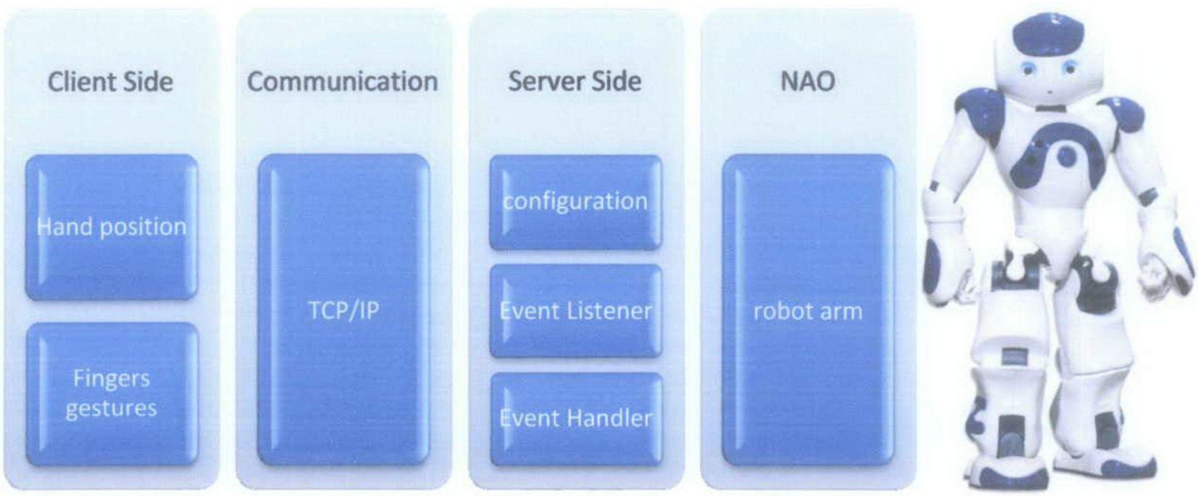
All the important aspects to develop such an interface has been successfully explored

- image acquisition
- image processing
- adaptive object tracking
- Software and hardware filtering
- motion detecting
- Sensors
- Communication



Data Glove has been connected to NAO and programmed successfully.

Proposed layout has been successfully implemented.



The Data Glove has been designed successfully and used to acquire fingers gestures.

Image processing techniques have been used to estimate position of the hand.

Server is responding correctly to changes done by the human hand.

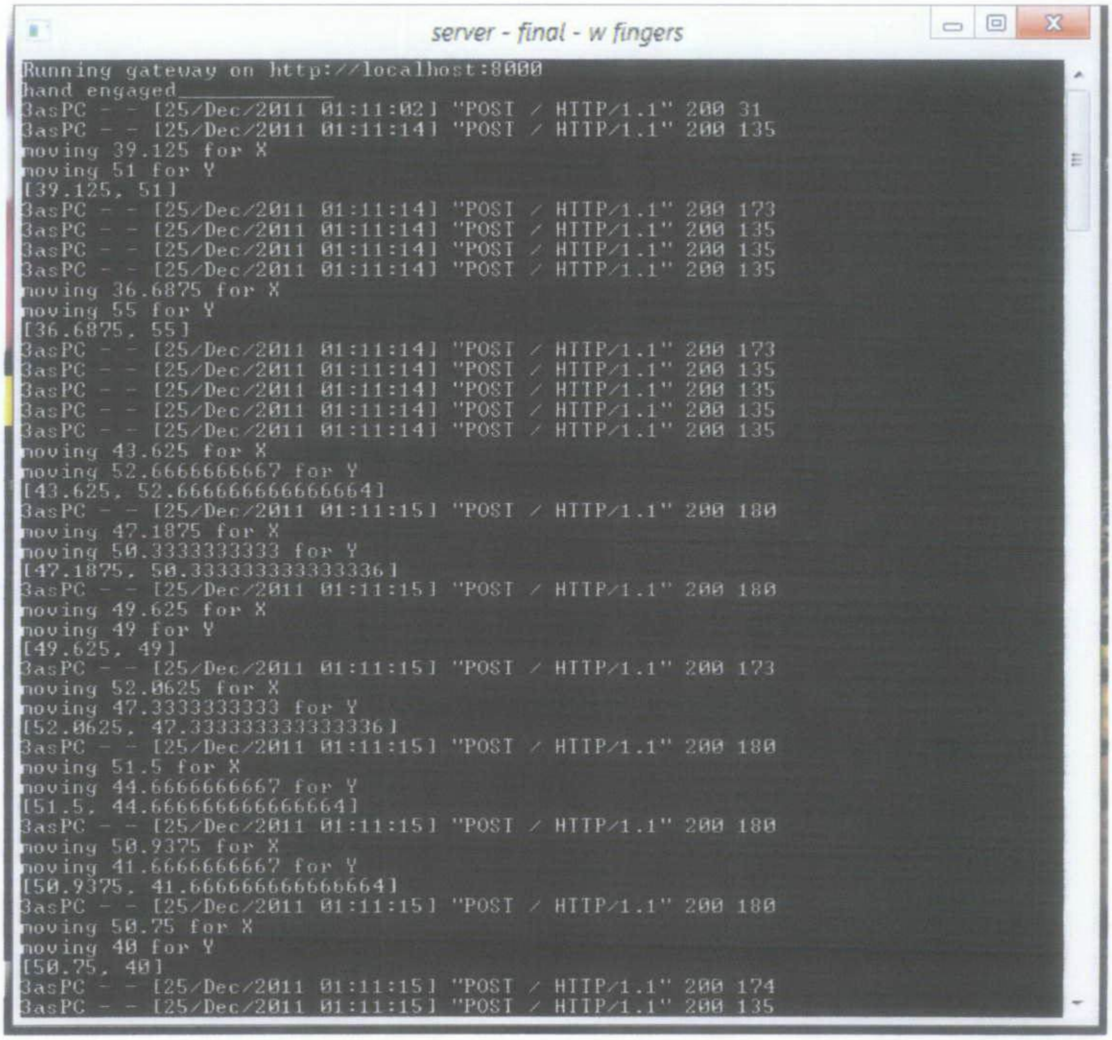
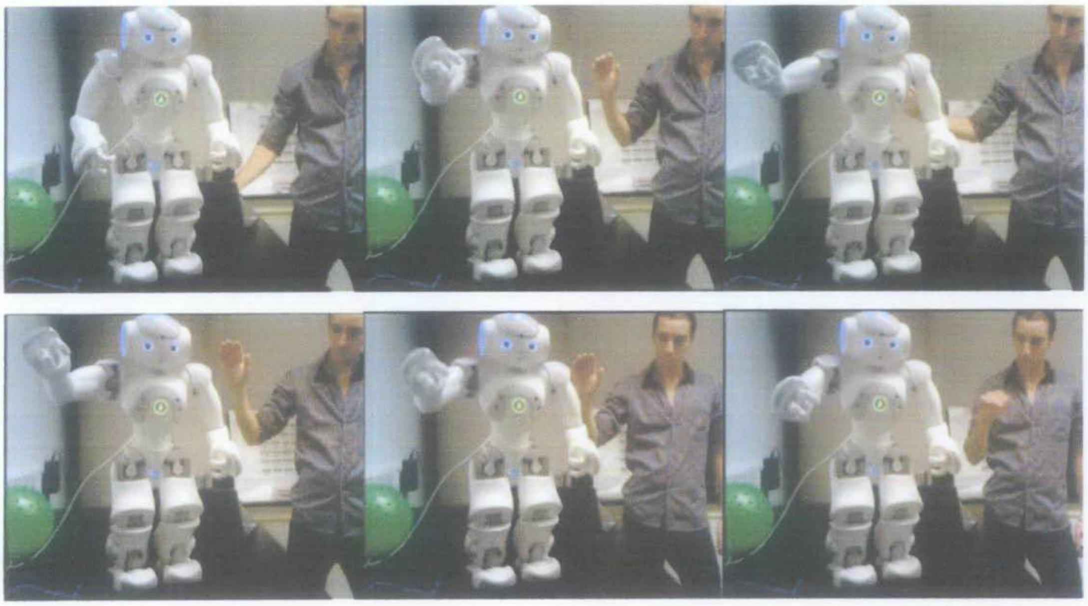


Figure 5 server is running

Humanoid Robot (NAO) is responding correctly to server commands.

- ✓ By making use of implemented hardware as an indicator to glove condition, Including hand relative position and fingers gestures recognition, we can design robotic applications desired to be remotely controlled by the actual human hand in an autonomous flow or by human interpretation
- ✓ Robotic Actuators have responded correctly to the desired sequence within the desired timing.
- ✓ Results have been verified, output variables have successfully matched the estimated results and the target sequence with no errors detected



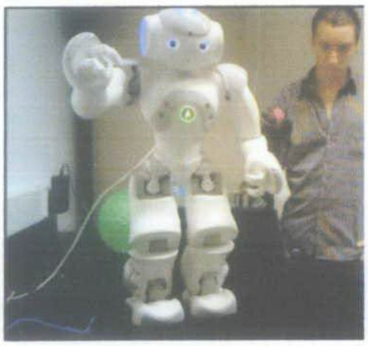


Figure 6 hand is closed

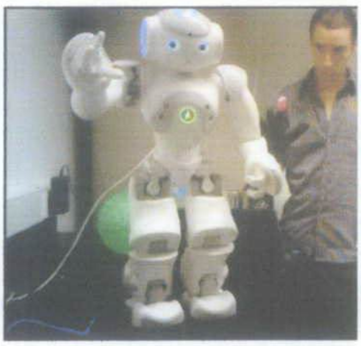


Figure 7 hand is opened

CHAPTER 5: CONCLUSION

Nowadays, robots are the perfect modification of electro mechanical instruments used in performing fabrication, production, surveillance and medication tasks. Using robots boosts economy as work done needs to be highly efficient, besides remote controlled operations are highly desired in certain applications, new potentials will be available as Robotic technologies improves.

Wearable Computing Technology is definitely coming. Accurate sensing capabilities will enhance robotics motion emulation techniques; it can be used where remote control is necessary or helpful.

Yet To be done: Reliable desired applications must be carried making use of achieved concept of robotic arm movement emulation.

REFERENCES

Papers and publications

[1] Carnegie Mellon University research paper http://www.disneyresearch.com/research/projects/humanoids_mocap_iros10.pdf
[2] motion emulation system with humanoid robot and image processing http://joa.csee.org.cn/Public/DownloadFile.aspx?FileStorageId=0b437650-a1ae-4c75-a0bf-54908bb3bea0
[3] real-time-robot-arm-control http://www.blendernation.com/2011/12/07/real-time-robot-arm-control/
[4] Real-Time Path Planning Procedure http://www.edcheung.com/job/sbr/TR89.pdf
[5] ipi soft Ref. http://www.ipisoft.com/
[6] imitative robotic arm control http://repository.cmu.edu/cgi/viewcontent.cgi?article=1019&context=silicon_valley

Internet sources

[1] iRobot Corporation. [http://www.irobot.com/]
[2] nao humanoid robot [http://users.aldebaran-robotics.com/]
[3] python documentation http://www.python.org/doc/
[4] NAO robotics http://www.cyberbotics.com/nao/
[5] Baxes, Gregory A (1994) DIGITAL IMAGE PROCESSING, Wiley *IT*
[6] Castleman, Kenneth (1979), Digital image Processing Prentice-Hall, --TA1632.C37--(recommended by UTHSCSA Image Tool)
[7] Foley, J.D., VanDam, A. (1982) FUNDAMENTALS OF INTERACTIVE COMPUTER GRAPHICS, Reading Massachusetts, Addison-Wesley, chapter 17.

APPENDIX

- **hand motion listener** has been written
- in order to record an estimated value to human hand relative position
- using image processing techniques
  - colour detection
  - motion detection

✓ using action script programming language

```
public function track( b : BitmapData ) : void
{
    _now = new BitmapData(320, 240, false, 0);

    _now.draw(b);
    _now.draw(_old, null, null, BlendMode.DIFFERENCE);

    _now.applyFilter(_now, _now.rect, new Point(), _col);
    _now.applyFilter(_now, _now.rect, new Point(), _blr);

    _now.threshold(_now, _now.rect, new Point(), '>', 0xFF333333, 0xFFFFFFFF);

    _old = new BitmapData(320, 240, false, 0);
    _old.draw(b);

    var area : Rectangle = _now.getColorBoundsRect(0xFFFFFFFF, 0xFFFFFFFF, true);
    _act = ( area.width > ( 320 / 100) * _min || area.height > (240 / 100) * _min );

    if ( _act )
    {
        _box = area;
        x = _box.x + (_box.width / 2);
        y = _box.y + (_box.width / 2);
    }
}
```

Code Snippet 1 : Hand Motion detection

- **fingers motion listener** has been written (Data Glove)
- in order to record an estimated value to human hand fingers gestures

✓ using action script programming language

```
var arduino:Arduino = new Arduino("127.0.0.1", 9001);
// Arduino initialization
function onArduinoStartup(e:ArduinoEvent):void {
    arduino.setAnalogPinReporting(analogPin0, Arduino.ON);
    arduino.setAnalogPinReporting(analogPin1, Arduino.ON);
    arduino.setAnalogPinReporting(analogPin2, Arduino.ON);
    arduino.setAnalogPinReporting(analogPin3, Arduino.ON);
    arduino.setAnalogPinReporting(analogPin4, Arduino.ON);
    log.appendText("Data Glove initialized"+ "\n");
}

function onEnterFrame(evt:Event):void {
    update(vals0,analogPin0,graph0,label0);
    update(vals1,analogPin1,graph1,label1);
    update(vals2,analogPin2,graph2,label2);
    update(vals3,analogPin3,graph3,label3);
    update(vals4,analogPin4,graph4,label4);

    for(m=1; m<wordsArray.length+1; m++){
    if(vals0[vals0.length-1]>database[(5*(m-1))+0]-tolerance.value&&
       vals0[vals0.length-1]<database[(5*(m-1))+0]+tolerance.value&&
       vals1[vals1.length-1]>database[(5*(m-1))+1]-tolerance.value&&
       vals1[vals1.length-1]<database[(5*(m-1))+1]+tolerance.value&&
       vals2[vals2.length-1]>database[(5*(m-1))+2]-tolerance.value&&
       vals2[vals2.length-1]<database[(5*(m-1))+2]+tolerance.value&&
       vals3[vals3.length-1]>database[(5*(m-1))+3]-tolerance.value&&
       vals3[vals3.length-1]<database[(5*(m-1))+3]+tolerance.value&&
       vals4[vals4.length-1]>database[(5*(m-1))+4]-tolerance.value&&
       vals4[vals4.length-1]<database[(5*(m-1))+4]+tolerance.value)

        {if (namesArray[m-1].toString()!=test){
        displayText.appendText(namesArray[m-1]+ "    ");
        test=namesArray[m-1].toString();
        displayText.verticalScrollPosition=pos;pos+=1}}
    }
}
```

Code Snippet 2 : Fingers gestures recognition

- **Client side message transmitter** has been written
- in order to transmit recorded values to the server

✓ using action script programming language

```
import flash.net.NetConnection;
import flash.net.Responder;

public class client_connection extends MovieClip
{
    // Gateway connection object
    private var gateway:NetConnection;

    public function gateway_connection()
    {
        // Setup connection
        gateway = new NetConnection();

        // Connect to gateway
        gateway.connect( "http://localhost:8000" );

        // This var holds the data we want to pass to the remote service.
        var param:Array = [99,11];

        // Set responder property to the object
        var responder:Responder = new Responder( onResult, onFault );

        gateway.call( "echo.echo", responder4, param4 );
    }

    // Result handler method
    private function onResult( result:Object ): void {
        var myData:String = result.toString();
        trace( result );// prints "Hello World!"
        status_txt.text = myData;
    }
    // Fault handler method displays error message
    private function onFault( error:Object ): void {
        // Notify the user of the problem
        status_txt.text = "Remoting error: \n";
        for (var d:String in error) {
            status_txt.appendText( error[d] + "\n" );
        }
    }
}
```

Code Snippet 3 : Message Transmitter

- **configuration file** for the Humanoid Robot (NAO) has been written
- in order to connect to the humanoid robot

✓ using python programming language

```
import naoqi
import motion
from naoqi import ALProxy

IP = "192.168.0.198" # setting Ip adress

PORT = 9559  # setting the port

if (IP == ""):
  print "IP address not defined, aborting"
  print "Please define it in " + __file__
  exit(1)

def loadProxy(pName):
  proxy = ALProxy(pName, IP, PORT)
  return proxy

def StiffnessOn(proxy):
  #We use the "Body" name to signify the collection of all joints
  pNames = "Body"
  pStiffnessLists = 1.0
  pTimeLists = 1.0
  proxy.stiffnessInterpolation(pNames, pStiffnessLists, pTimeLists)

def PoseInit(proxy):
  # Define The Initial Position
  HeadYawAngle        = 0
  HeadPitchAngle      = 0

  ShoulderPitchAngle = 80
  ShoulderRollAngle  = 20
  ElbowYawAngle      = -80
  ElbowRollAngle     = -60
  WristYawAngle      = 0
  HandAngle          = 0

  HipYawPitchAngle   = 0
  HipRollAngle       = 0
  HipPitchAngle      = -25
  KneePitchAngle     = 40
  AnklePitchAngle    = -20
```

Code Snippet 4 : Configuration file

- **server listener** for the Humanoid Robot (NAO) has been written
- in order to receive recorded values from the client side

✓ using python programming language

```
import os
from pyamf.remoting.gateway.wsgi import WSGIGateway
from wsgiref import simple_server

gw = WSGIGateway(services)

httpd = simple_server.WSGIServer(
    ('localhost', 8000),
    simple_server.WSGIRequestHandler,
)

def app(environ, start_response):
    if environ['PATH_INFO'] == '/crossdomain.xml':
        fn = os.path.join(os.getcwd(), os.path.dirname(__file__),
           'crossdomain.xml')

        fp = open(fn, 'rt')
        buffer = fp.readlines()
        fp.close()

        start_response('200 OK', [
            ('Content-Type', 'application/xml'),
            ('Content-Length', str(len(''.join(buffer))))
        ])

        return buffer

    return gw(environ, start_response)

httpd.set_app(app)

print "Running AMF gateway on http://localhost:8000"

try:
    httpd.serve_forever()
except KeyboardInterrupt:
    pass
```

Code Snippet 5 : Server Listener

- **command handler** for the Humanoid Robot (NAO) has been written
- in order to send motion commands to be done by the humanoid robot

✓ using python programming language

```
import config
import motion

 proxy = config.loadProxy("ALMotion")

    #Set NAO in stiffness On
    config.StiffnessOn(proxy)

    # send robot to Pose Init
    config.PoseInit(proxy)

    leftarm    = "LArm"
    rightarm   = "RArm"
    space       = motion.SPACE_NAO
    axisMask    = 7            # just control position
    isAbsolute = True

def echo(data):

    print "moving %s for Y"%(data[0])
    print "moving %s for Y"%(data[1])

    # define the changes relative to the current position
    dx          =  data[0]   # translation axis X (meters)
    dy          =  data[1]   # translation axis Y (meters)
    dz          =  0.00      # translation axis Z (meters)
    dwx         =  0.00      # rotation axis X (radians)
    dwy         =  0.00      # rotation axis Y (radians)
    dwz         =  0.00      # rotation axis Z (radians)
    targetPos   = [dx, dy, dz, dwx, dwy, dwz]

    # go to the target and back again
    path        = [targetPos] # ,currentPos
    times       = [2.0] # seconds  # ,4.0

    proxy.positionInterpolation(leftarm, space, path,
                                axisMask, times, isAbsolute)

    print "%s "%(data)
    return data
```

Code Snippet 6 : Robot Motion handler