DEVELOPMENT OF HUMAN MOTION GUIDED ROBOTIC ARM USING KINECT V2 SENSOR AND MATLAB

[1]Vashista Bhati
[1]Lecturer
[2]Lecturer, Department of Electrical Engineering, Govt. Polytechnic College, Barmer, Rajasthan, India

Abstract*—*The main aim of this paper is to build a Robotic arm which reflects the action of human arm. The human arm is tracked by Microsoft Kinect V2 sensor. The Kinect V2 sensor provides reliable and sufficient color and depth information of human skeleton. Image acquisition toolbox of MATLAB simulator is used to extract the joint coordinates of human skeleton from color and depth information. Joint coordinates of human arm is separated from joint coordinates of human skeleton. Now by applying inverse kinematics to coordinates, angle of shoulder and elbow joint can be find out. These angles are used to drive servo motor. This model can work near nuclear radiation area and in any type of circumstances where human cannot easily work. Robot soldiers can be developed which will work as real human soldiers. A camera can be placed on robot so that far sitting robot handler can see what robot can see. This Robot soldier will copy actions of real human soldier standing in front of Kinect V2 sensor. So this robot loaded with high technique weapons will work with the fighting experience of real human soldier.

Index Terms*—*Robotic Arm, Human Motion, MATLAB, Microsoft Kinect V2

I. INTRODUCTION

A In the last few years, Kinect sensor has attracted a great deal of invention because of their unique advantages for Image processing. Some of these advantages include extracting human skeletal data and tracking human body actions. Kinect is very affordable, which makes it an attractive tool for many researchers and companies interested in 3D modelling. However, Kinect was originally designed for tracking human body movements in video gaming. Since the tracking has to be done in real time, Kinect has been designed such that it could capture the depth frames with high frequency, being maximum 30 Hz. Kinect combines an HD color camera, a depth processor and an infrared sensor to effectively track up to 6 human bodies.

The interfacing between Man and machine is the beginning of a new era. Humans and machines no longer run parallel to each other, but instead, they go hand in hand. This new technology is helping to improve lifestyles. This motivates the creation of a better technology for tomorrow. This is called, the digital circle. The technology used in robotics earlier was of a joystick, then came the touchscreen, and now it is the advent of gestures.

This research paper work helps in developing a model which can work near nuclear radiation area and in any type of circumstances where human cannot easily work. For example in year 2008 Mumbai attack our soldiers were martyred. To save our soldiers using this technology, Robot soldiers can be developed which will work as real human soldiers. A camera can be placed on robot so that far sitting robot handler can see what robot can see. This Robot soldier will copy actions of real human soldier standing in front of Kinect V2 sensor. So this robot loaded with high technique weapons will work with the fighting experience of real human soldier.

This model can be used in many other applications like Roboarm in medical field, in Teleimmersive conferencing for creating real 3D environment in Video conferencing and in home monitoring for ill people alone at home etc.

II. DEVELOPMENT AND METHODOLOGY

Development of Model

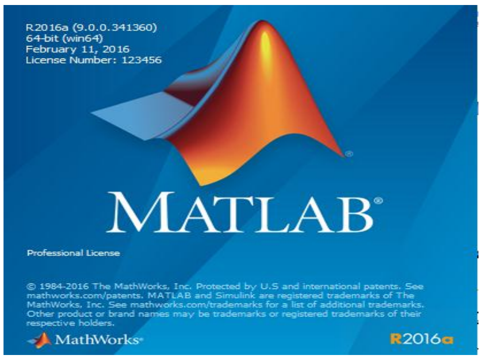
Kinect V2 sensor is the heart of this model. In development of this model I also used Arduino board, Bluetooth device and Robotic arm. Robotic arm is manufactured using wood and servo motors.

Simulator

Simulation is an important aspect in the development of robotics, especially for mobile autonomous robots. With good simulations more of the development can be done in the simulation environment, lowering cost as the need for hardware and prototypes are reduced.

Here for developing robotic arm Auto-CAD software can be used. But due to sort of time I have made it from wood.

For programming MATLAB 2016a software is used. Matlab is used for extracting joint coordinates from Kinect v2 sensor and interfacing laptop with Bluetooth device.



R2016a (9.0.0.341360)
64-bit (win64)
February 11, 2016
License Number: 123456

© 1984-2016 The MathWorks, Inc. Protected by U.S and international patents. See mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Fig. 1 Matlab R2016a Simulator

Prior to work

- Gathered the information about the project via Internet, journals, magazines, published work and reference books.
- Study of the software implementations (Matlab 2016a).
- Purchased the used Kinect V2 sensor which is compatible with the Matlab 2016a version.

Research Methodology

Every project is not all of the sudden. It always has some predecessors to it through which several attempts are being made to obtain the desired results. Similarly, there were prior attempts to make the robotic arm simulate the human gestures using Kinect sensors. But the prior attempts were not as much as successful as the given work because of following reasons:-

1. The Kinect sensors used in the prior project was the version-1 of the Kinect sensor which is able to detect only 20 human joints. But the Kinect sensor used in this project is Kinect sensor version-2 which is able to detect 24 human joints which includes thumb etc. thus the detection of the human gestures is made more effective using this sensor and hence simulation of those gestures through robotic arm is much more accurate and effective.
2. The programming of converting the human gesture to their respective coordinates was done using Visual studios in most of the earlier works. This made the program quite lengthy and complex. Here we have used Matlab programming language to get the resulting coordinates which made the programming quite easy to work with. Thus the results obtained are much more accurate as the probability of tracking the errors and correcting them is better than the earlier versions.
3. The algorithm used in this project to get the coordinates of the joints of the human gesture is based on inverse kinematics which allows us to get the angles from the coordinates which is much easier than that of used in prior works.

Figure 2 presents a block diagram to understand the improvements in designing the robotic arm which mimics the actions of human arm.

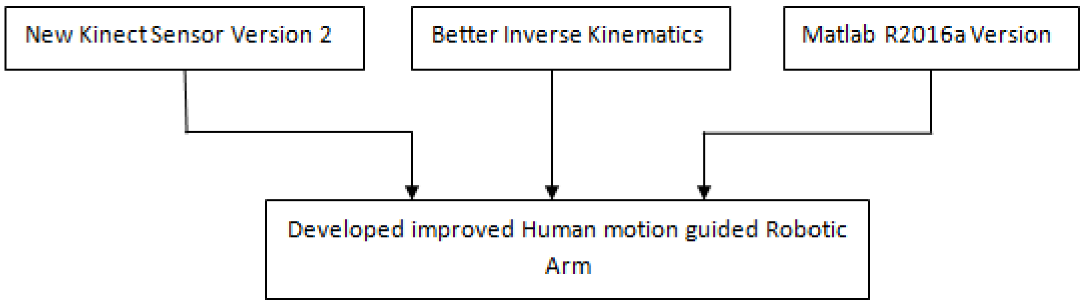

New Kinect Sensor Version 2
Better Inverse Kinematics
Matlab R2016a Version
Developed improved Human motion guided Robotic Arm

Fig. 2 Block diagram of Project Methodology

III. Hardware and Software

Kinect V2

The second-generation Kinect (Version 2) for Windows, based on the same core technology as Kinect for Xbox One, including a new sensor, was first released in 2014. Kinect on Xbox One features a number of changes in comparison to the first-generation, Xbox 360 version. The physical design of the new Kinect is aligned with that of the Xbox One console itself, with a more rectangular appearance, a two-toned matte-grey/glossy black color scheme, diagonal vents, and a white status light. The new Kinect no longer contains a motor for automatic angle adjustment, and must be adjusted manually.

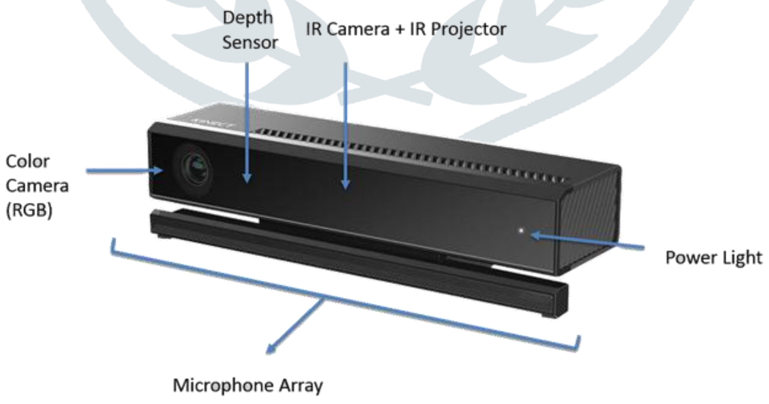


Fig. 3 Kinect V2 Sensor

Arduino Uno

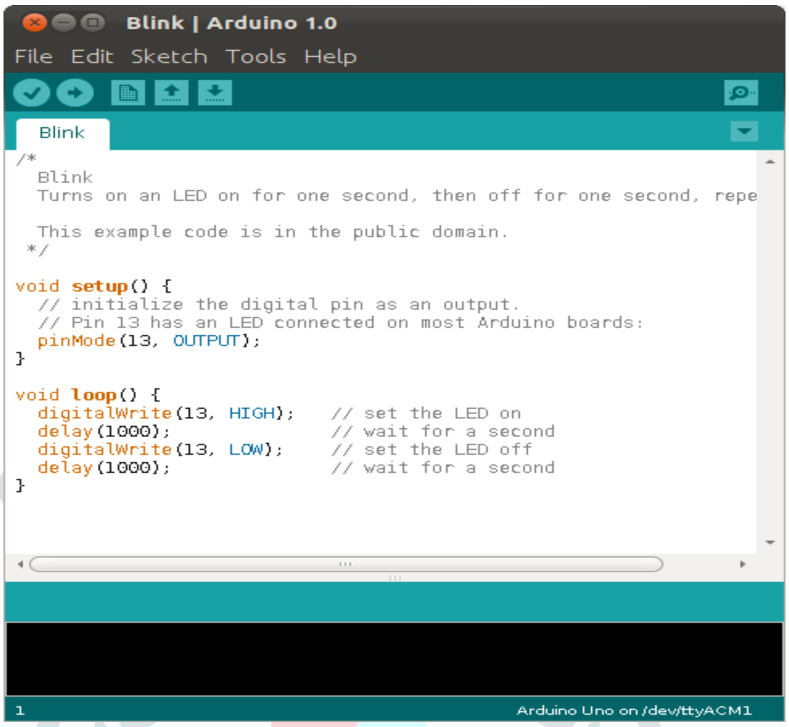
Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language, and the Arduino Software (IDE).

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT (Internet of Things) applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software too, is open-source, and it is growing through the contributions of users worldwide.

The Arduino Uno is based on the popular Atmel ATmega 328P microcontroller. Older Arduino Uno Boards utilized the ATmega8 or ATmega 168. However, boards featuring those microcontrollers are no longer available. The ATmega 328P is a low power 8-bit AVR RISC-based microcontroller. The ATmega supports operating frequencies up to 20 MHz, however it is clocked at 16 MHz on Arduino Uno boards. The 328P features a variety of useful peripherals to interact with its environment.

Software

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and provides simple one-click mechanism to compile and load programs to an Arduino board. A program written with the IDE for Arduino is called a "sketch".



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repe

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

Fig. 4 Arduino 1.0 Simulator

Bluetooth Device HC-05

HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. The HC-05 Bluetooth Module can be used in a Master or Slave configuration, making it a great solution for wireless communication. This serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH (Adaptive Frequency Hopping Feature).

The Bluetooth module HC-05 is a MASTER/SLAVE module. By default the factory setting is SLAVE. The Role of the module (Master or Slave) can be configured only by AT COMMANDS. The slave modules cannot initiate a connection to another Bluetooth device, but can accept connections. Master module can initiate a connection to other devices. The user can use it simply for a serial port replacement to establish connection between MCU and GPS, PC to your embedded project, etc.

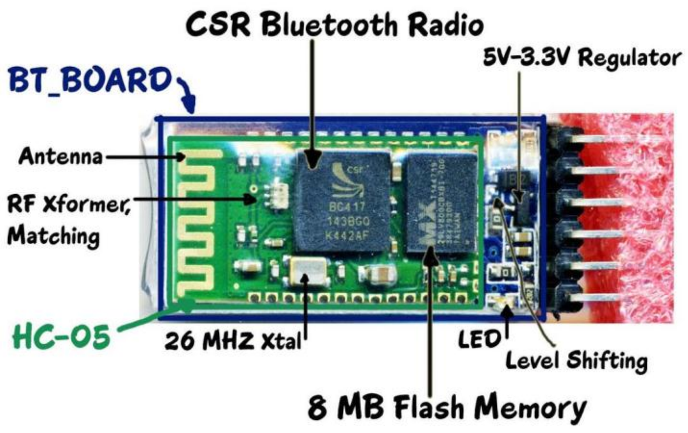


Fig. 5 Bluetooth Module HC-05

Servo Motor

Servo motors have been around for a long time and are utilized in many applications. They are small in size but pack a big punch and are very energy-efficient. These features allow them to be used to operate remote controlled or radio-controlled toy cars, robots and airplanes. Servo motors are also used in industrial applications, robotics, in-line manufacturing, pharmaceutics and food services.

The servo circuitry is built right inside the motor unit and has a shaft, which usually is fitted with a gear. The motor is controlled with an electric signal which determines the amount of movement of the shaft.

To fully understand how the servo works, you need to take a look under the hood. Inside there is a pretty simple set-up: a small DC motor, potentiometer, and a control circuit. The motor is attached by gears to the control wheel. As the motor rotate, the resistance of potentiometer changes, so the control circuit can precisely regulate how much movement there is and in which direction.

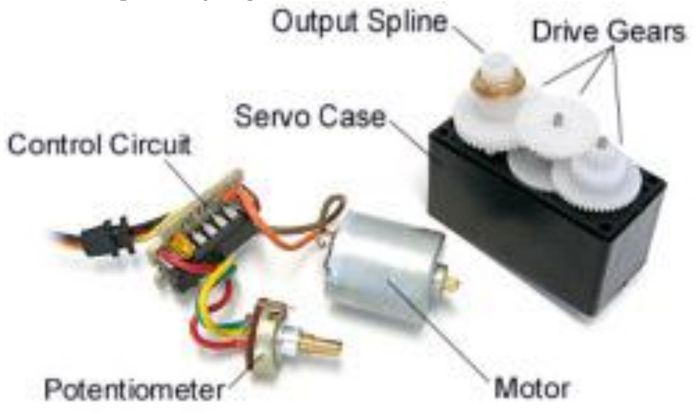


Fig. 6 The Guts of a servo motor (L) and An Assembled Servo Motor (R)

MATLAB

The MATLAB system consists of five main parts:

The MATLAB language

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

The MATLAB working environment

This is the set of tools and facilities that you work with as the MATLAB user or programmer. It includes facilities for managing the variables in your workspace and importing and exporting data. It also includes tools for developing, managing, debugging, and profiling M-files, MATLAB's applications.

Handle Graphics

This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete Graphical User Interfaces on your MATLAB applications.

The MATLAB mathematical function library

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix Eigen values, Bessel functions, and fast Fourier transforms.

The MATLAB Application Program Interface (API)

This is a library that allows you to write C and Fortran programs that interact with MATLAB. It include facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

MATLAB R2016a is used in this dissertation. It is updated version of MATLAB. The main fact about version is that the image acquisition toolbox of this version and later version(R2016b) of MATLAB only supports Kinect V2 sensor.

IV. Design and Implementation

This is everything about the work done during dissertation work. From interfacing Kinect sensor to moving of Robotic arm is divided in four parts. These are explained one by one according to work progress.

Image Acquisition using Kinect version 2

The Kinect Adaptor lets you acquire images using a Kinect for Windows device. Kinect devices are often used in automotive IVS, robotics, human-computer interaction (HCI), security systems, entertainment systems, game design, and civil engineering. Uses of Kinect devices include body analysis, 3-D mapping, gesture recognition, human travel patterns, and sports and games. Image acquisition is performed in two steps.

Detect the Kinect V2 device

Typically, each camera or image device in the Image Acquisition Toolbox has one DeviceID. Because the Kinect for Windows camera has two separate sensors, the color sensor and the depth sensor, the toolbox lists two DeviceIDs. Use imaqhwinfo on the adaptor to display the two device IDs.[22]

```
>>info = imaqhwinfo('kinect');
>>info
  info =

            AdaptorDllName: '<matlabroot>\.......\win64\mwkinectimaq.dll'
            AdaptorDllVersion: '5.0 (R2016a)'
            AdaptorName: 'kinect'
            DeviceIDs: {[1]  [2]}
            DeviceInfo: [1x2 struct]
```

If we look at each device, we will see that they represent the color sensor and the depth sensor. The following shows the color sensor.

```
info.DeviceInfo(1)
```

```
ans =

        DefaultFormat: 'RGB_1920x1080'
         DeviceName: 'Kinect V2 Color Sensor'
   DeviceID: 1
        VideoInputConstructor: 'videoinput('kinect', 1)'
        VideoDeviceConstructor: 'imaq.VideoDevice('kinect', 1)'
   SupportedFormats: 'RGB_1920x1080'
```

The following shows the depth sensor, which is Device 2.

```
info.DeviceInfo(2)
ans =

        DefaultFormat: 'Depth_512x424'
   DeviceName: 'Kinect V2 Depth Sensor'
   DeviceID: 2
        VideoInputConstructor: 'videoinput('kinect', 2)'
        VideoDeviceConstructor: 'imaq.VideoDevice('kinect', 2)'
   SupportedFormats: 'Depth_512x424'
```

We can use multiple Kinect devices together, but only one Kinect V2 at a time. Multiple Kinect sensors are enumerated as DeviceIDs [1] [2] [3] [4] and so on. For example, if we had two Kinect devices, a Kinect V1 and a Kinect V2, the first one would have Kinect Color Sensor with DeviceID 1 and Kinect Depth Sensor with DeviceID 2 and the second Kinect device would have Kinect V2 Color Sensor with DeviceID 3 and Kinect V2 Depth Sensor with DeviceID 4.

Acquire Image and Body Data Using Kinect V2

To acquire image and body data using Kinect v2 sensor we need to create a videoinput object for the color sensor to acquire RGB images and then an object for the depth sensor to acquire body data.[24]

Create the videoinput object for the color sensor. DeviceID 1 is used for the color sensor.

```
Video1 = videoinput('kinect',1);
```

Create the videoinput object for the depth sensor. Note that a second object is created (vid2), and DeviceID 2 is used for the depth sensor.

```
Video2 = videoinput('kinect', 2);
```

To achieve body tracking using image acquisition tool we need to change a device specific property i.e. change EnableBodyTracking from off state to on state.

So now body tracking data can be accessed using getdata function. Data returned from body index are as follows.

```
metaData =

11x1 struct array with fields:
   IsBodyTracked: [1x6 logical]
   BodyTrackingID: [1x6 double]
   BodyIndexFrame: [424x512 double]
   ColorJointIndices: [25x2x6 double]
   DepthJointIndices: [25x2x6 double]
   HandLeftState: [1x6 double]
   HandRightState: [1x6 double]
   HandLeftConfidence: [1x6 double]
   HandRightConfidence: [1x6 double]
   JointTrackingStates: [25x6 double]
   JointPositions: [25x3x6 double]
```

The detail of given data is shown in table below.

Table 1 Details of Data extracted from Kinect

MetaData	Description
Is Body Tracked	A 1 x 6 Boolean matrix of true/false values for the tracking of the position of each of the six bodies. A 1 indicates the body is tracked, and a 0 indicates it is not. See step 9 below for an example.
Body Tracking ID	A 1 x 6 double that represents the tracking IDs for the bodies.
Color Joint Indices	A 25 x 2 x 6 double matrix of x- and y-coordinates for 25 joints in pixels relative to the color image, for the six possible bodies.
Depth Joint Indices	A 25 x 2 x 6 double matrix of x- and y-coordinates for 25 joints in pixels relative to the depth image, for the six possible bodies.
Body Index Frame	A 424 x 512 double that indicates which pixels belong to tracked bodies and which do not. Use tis metadata to acquire segmentation data.

MetaData	Description
Hand Left State	A 1 x 6 double that identifies possible hand states for the left hands of the bodies. Values include: 0 unknown 1 not tracked 2 open 3 closed 4 lasso
Hand Right State	A 1 x 6 double that identifies possible hand states for the right hands of the bodies. Values include: 0 unknown 1 not tracked 2 open 3 closed 4 lasso
Hand Left Confidence	This is a 1 x 6 double that identifies the tracking confidence for the left hands of the bodies. Values include: 0 low 1 high
Hand Right Confidence	A 1 x 6 double that identifies the tracking confidence for the right hands of the bodies. Values include: 0 low 1 high
Joint Tracking States	A 25 x 6 double matrix that identifies the tracking states for joints. Values include: 0 not tracked 1 inferred 2 tracked
Join tPositions	This is a 25 x 3 x 6 double matrix indicating the location of each joint in 3-D space. See the **Joint Positions** section for a list of the 25 joint positions.

The Enable Body Tracking property indicates whether body metadata is collected. When set to on, this list displays the order of the joints returned by the Kinect V2adaptor in the Joint Positions property.

SpineBase = 1;	SpineMid = 2;	Neck = 3;
Head = 4;	ShoulderLeft = 5;	ElbowLeft = 6;
WristLeft = 7;	HandLeft = 8;	ShoulderRight = 9;
ElbowRight = 10;	WristRight = 11;	HandRight = 12;
HipLeft = 13;	KneeLeft = 14;	AnkleLeft = 15;
FootLeft = 16;	HipRight = 17;	KneeRight = 18;
AnkleRight = 19;	FootRight = 20;	SpineShoulder = 21;
HandTipLeft = 22;	ThumbLeft = 23;	HandTipRight = 24;
ThumbRight = 25;		

Skeletal Joint Indices Extraction

Depth Joint Indices is an array of 25x2x6. It contains the coordinates of the 25 skeletal joint of the entire 6 tracked skeleton. The joint coordinates of a tracked skeleton can be easily extracted from the Depth Joint Indices using some MATLAB programming (Appendix 1). As I only required joint coordinates of Right Shoulder, Right hand elbow and Right hand wrist in my dissertation work. So again using MATLAB programming these coordinates can be separated from the 25 joint coordinates.

Inverse Kinematics

Kinematic analysis is one of the first steps in the design of most industrial robots. Kinematic analysis allows the designer to obtain information on the position of each component within the mechanical system. This information is necessary for subsequent dynamic analysis along with control paths. Basically Kinematic analysis is of two types. One is Inverse kinematic and another one is Forward kinematic. In short, determining the angles from co-ordinates is an Inverse Kinematics problem and determining co-ordinates from angles is a Forward Kinematics problem.
In a two dimension Cartesian system angle of a line to the horizontal axis can be calculated as shown below.

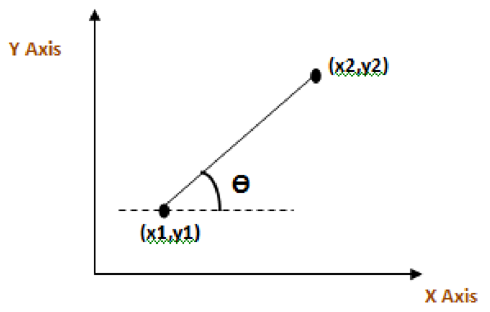


Fig. 7 Two Dimension Cartesian system angle problem

$$\theta = \tan^{-1}\frac{(y2-y1)}{(x2-x1)}$$

Using this formula we can find the angles at the shoulder and elbow joints using co-ordinates of shoulder, elbow and wrist joints. The diagram shown below presents position 1 and position 2 of human arm. So change in the arm position can be calculated in terms of angles at shoulder and elbow joints.

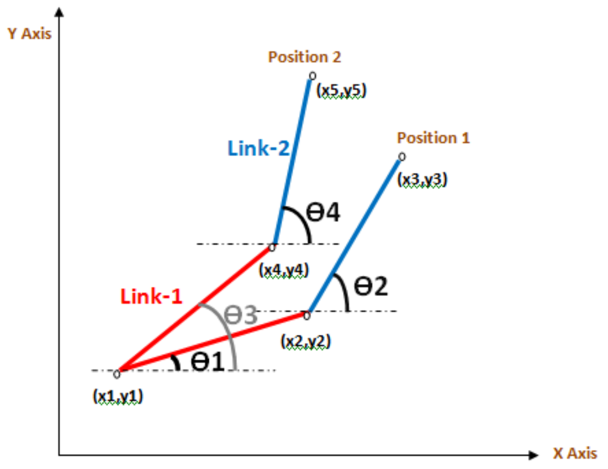


Fig. 8 Two dimension Link 1 and Link 2 angle problem

$$\theta 1 = \tan^{-1}\frac{(y2-y1)}{(x2-x1)}$$
$$\theta 2 = \tan^{-1}\frac{(y3-y2)}{(x3-x2)}$$
$$\theta 3 = \tan^{-1}\frac{(y4-y1)}{(x4-x1)}$$
$$\theta 4 = \tan^{-1}\frac{(y5-y4)}{(x5-x4)}$$

If there is any change in position of Link-1, it can be easily find out by calculating$(\theta 3 - \theta 1)$. Change in Link-2 is related to Link-1. So we must take care of angle $\theta 1$ and $\theta 3$ while calculating angle of movement of Link-2.

If Link-1 is moved by some angle while Link-2 doesn't move, then

$$(\theta 2 - \theta 1) = (\theta 4 - \theta 3)$$

Angle of movement at shoulder joint is = $(\theta 3 - \theta 1)$.

Angle of movement at elbow joint is = $(\theta 4 - \theta 3) - (\theta 2 - \theta 1)$.

Kinect-Robot Interfacing

The joint co-ordinates received from Kinect sensor with the help of MATLAB are then used to find angles of movement at shoulder and elbow joint between position 1 and position 2. This calculated angle will be given to servo motors which are attached to robotic arm, so that the robotic arm can reflect the actions of human arm. The transmission of angle between MATLAB and Robot can be either wireless or wired.

Here in this dissertation work I preferred to use Bluetooth technology for interfacing MATLAB and Robot. Bluetooth module HC-05 is used for this interfacing. Bluetooth communication is free from lots of channel wires and also it has a communication range of around 10 meters. In section we have seen the connection diagram to connect Bluetooth module to Arduino Uno board. Now we have to understand "how to interface Bluetooth module to MATLAB?"

MATLAB have some functions for transmitting data over the Bluetooth interface, looking device specific properties of Bluetooth device and also change properties of that Bluetooth device.

To Create a Bluetooth object called bt using channel 1 function used is

```
bt = Bluetooth('HC-05', 1);
```

To connect to the device function used is

```
fopen(bt);
```

To send a message to the remote device the fwrite function is used.

```
fwrite(bt, angle);
```

To clean up by deleting and clearing the object fclose function is used.

```
fclose(bt);
clear('bt');
```

V. Results and Discussion

When we tested the system on many circumstances, we obtained a very good result and the identification rate of human motions by Robotic arm was about 96%. Figure 9 shows the snapshots of human motion guided robotic arm movements. The error in its movement is very less. We have used a simple MATLAB simulator which makes our project simple and efficient.

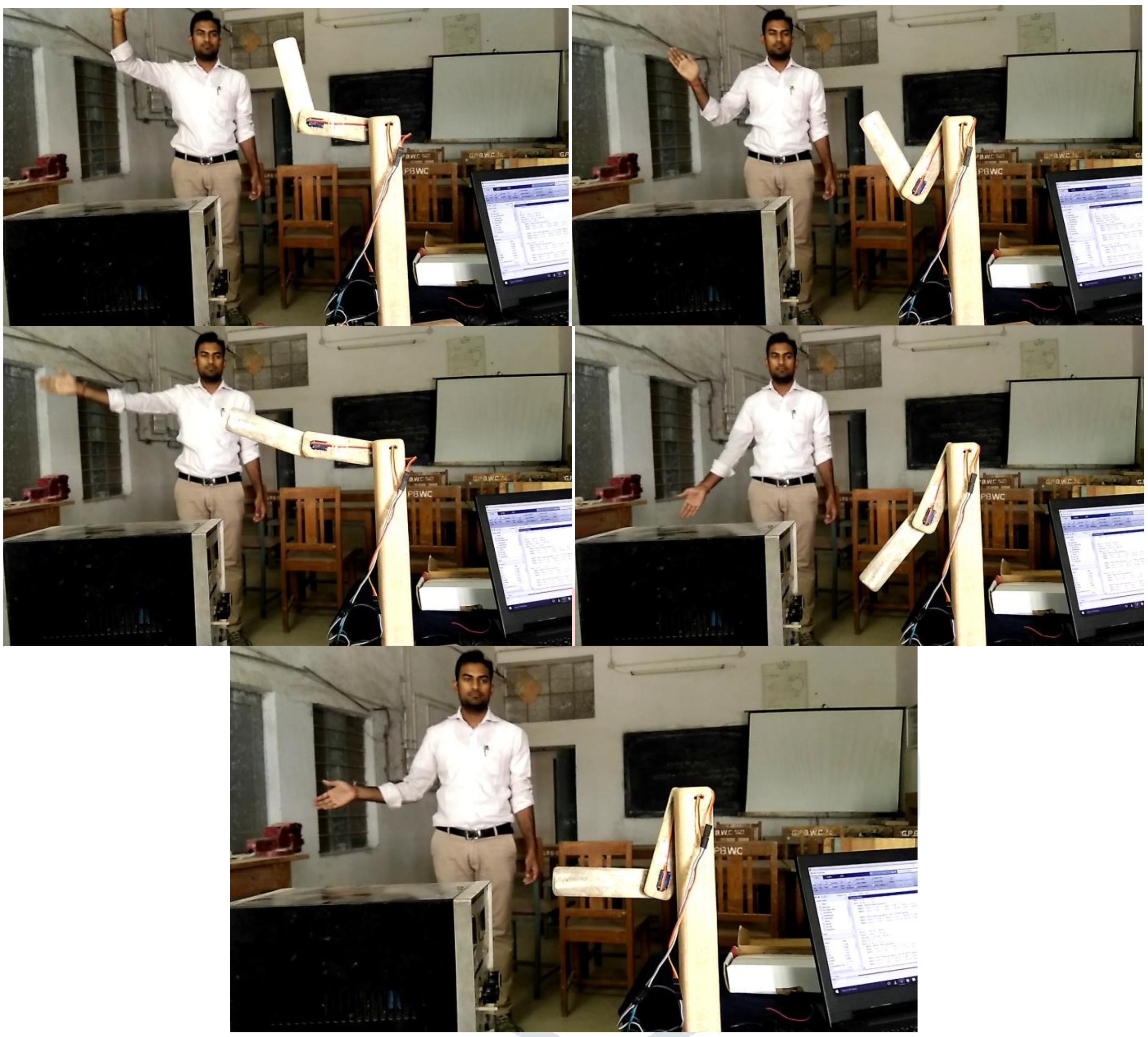


Fig. 9 Snapshots of human motion guided robotic arm movements

VI. Conclusions

This thesis has dealt with the design, implementation and integration of the human motion guided Robotic arm. The work was divided into three parts.

The first part of the paper was dealing with the design of the Human-Kinect interface and extracting skeleton data with the help of Matlab 2016a simulator. This Skeleton data can be used for common tasks.

In the second part of the paper, the Bluetooth interface was established between Matlab and Arduino. Arduino board has a microcontroller attached to it, so that it can be programmed to achieve a desired task.

The third part of the paper was dealing with the reduction in errors of Robotic arm. These errors include delay in Robotic arm movements and jerk of Robotic arm.

This final implemented architecture gives simple, low cost and robust solution for controlling humanoid with human gestures.

For future direction to this project a complete humanoid can be developed which will copy all actions of human skeleton, develop a Robotic arm which can reflect actions of human arm in three dimension and Tele-immersive conference can be arranged using more than one Kinect sensors. Audio recognition capability will be added to this dissertation work, so that it can be controlled with voice also.

References

[1] Tanner Bryce Blair, Chad Eric Davis, “Innovation Engineering Outreach: A Special Application of the Xbox 360 Kinect Sensor.” Frontiers in Education Conference, 2013 IEEE Pages 1279-1283.

[2] M. Samir, E. Golkar and A. A. A. Rahni, "Comparison between the Kinect V1 and Kinect V2 for respiratory motion tracking," 2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), Kuala Lumpur, 2015, pp. 150-155.

[3] P. Fankhauser, M. Bloesch, D. Rodriguez, R. Kaestner, M. Hutter and R. Siegwart, "Kinect v2 for mobile robot navigation: Evaluation and modeling," Advanced Robotics (ICAR), 2015 International Conference on, Istanbul, 2015, pp. 388-394.
[4] L. Liu and S. Mehrotra, "Patient walk detection in hospital room using Microsoft Kinect V2," 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Orlando, FL, USA, 2016, pp. 4395-4398.
[5] L. Müller, M. Mohammed and J. W. Kimball, "Using the Arduino Uno to teach digital control of power electronics," 2015 IEEE 16th Workshop on Control and Modeling for Power Electronics (COMPEL), Vancouver, BC, 2015, pp. 1-8.
[6] D. Sunehra and M. Yeena, "Implementation of interactive home automation systems based on email and Bluetooth technologies," 2015 International Conference on Information Processing (ICIP), Pune, 2015, pp. 458-463.
[7] M. Alsayegh, F. I. Mues, J. Malzahn and T. Bertram, "Analytic Forward and Inverse Kinematics of a Multi-Elastic-Link Robot Arm," Proceedings of ISR 2016: 47st International Symposium on Robotics, Munich, Germany, 2016, pp. 1-6.
[8] http://www.thocp.net/reference/robotics/robotics2.htm#26.
[9] http://www.computerhistory.org/revolution/arti%20cial-intelligence-%20robotics/13/289
[10] Shimon Y. Nof. Handbook of Industrial Robotics. Wiley, 2nd edition,1999.
[11] https://en.wikipedia.org/wiki/Telerobotics
[12] https://en.wikipedia.org/wiki/Autonomous_robot
[13] Kinect for Windows. [Online] MICROSOFT. [Cited: 04 19, 2013.] http://www.microsoft.com/en- us/kinectforwindows/
[14] http://www.xbox.com/en-US/xbox-one/accessories/kinect
[15] L. G. Wiedemann, R. Planinc, I. Nemec and M. Kampel, "Performance evaluation of joint angles obtained by the kinect V2," Technologies for Active and Assisted Living (TechAAL), IET International Conference on, London, 2015, pp. 1-6.