Real-Time Robot Control Using Leap Motion

A Concept of Human-Robot Interaction

Trinadh Venkata Swamy Naidu Venna
Department of Computer Science, RISC Laboratory
University of Bridgeport
Bridgeport, CT
tvenna@my.bridgeport.edu

Sarosh Patel
Department of Computer Science, RISC Laboratory
University of Bridgeport
Bridgeport, CT
saroshp@bridgeport.edu

***Abstract* — With the advent of robots in various industries, countless tasks that are complex for humans are made easier than ever before. Thanks to their functionality and accuracy we are able to achieve high productivity while investing less cost. As we see into our future of manufacturing industries, we also see that robots will replace most of the human workers to achieve faster production. All these areas use automated robotic arms which do certain tasks assigned to them with amazing speeds and pin point accuracy, because of the mathematical calculations done by the computer and are not manually controlled by humans. However, no robot can match the dexterity of a human hand. In order to control a robotic arm manually, the operator should carefully manipulate every joint in the arm to a perfect angle. Just as it sounds, it is very tedious to control them manually, that's why they are left to the computers. But, areas such as medicine, space research, and military robotics require robot arms to be manually controlled to operate with objects that cannot be dealt with human hands. Existing systems provide traditional controllers that are not efficient to handle a robotic arm and is time consuming. To achieve speed and accuracy like automated robots, we need a new approach that can bridge this gap. Here comes the Leap Motion Technology, a latest invention in Human-Computer interaction area. Using this device we track a human hand in air accurate to millimeter. The position of hand is then used to calculate the joint angles that in turn help us to rotate the robotic arm joints by the computer with blazing speeds.**

Keywords—Robotic arm; Leap Motion; Gesture Control; Inverse Kinematics; IR Sensor;

I. INTRODUCTION

"*We're fascinated with robots because they are reflections of ourselves*" –said Ken Goldberg, Professor and scientist from University of California, Berkeley. It's been 2300 years since the invention of the first so-called robot named "*Steam powered pigeon*", designed by Greek Mathematician, *Archytas.* The steam-powered pigeon was able to fly only 200 meters before it ran out of steam. However advancement in technology leads to design of robots that can even explore an entire planet themselves without relying on humans. As ken said, robots are the reflections of us, they are designed to replace humans in extreme working conditions, the area is so widespread, *and they* are almost in same proportion with humans in a manufacturing industry. There are robots for every task that is complicated or dangerous for a human to do. Productivity has increased at humongous rate after robots are being used in manufacturing. Also *they* cut cost of product drastically since *they* are not paid for the work. Almost all areas of technology use robots these days. Starting from the popular field of robotics i.e., manufacturing industries to the advanced researches on planet, space missions, we see robots everywhere taking place of humans to tackle complex tasks quicker and easier. Most of the articulated robots that we see in manufacturing of cars and electronics are automated, that means they are preprogrammed to do specific set of tasks repeatedly. They don't require human interaction in their operation. However, in some areas robots are to be manually controlled to do certain tasks that are not programmed earlier. Space and military missions require manual robotic arms to do the real-time operations. Here the need of a controller arises to manipulate the arm. A robotic arm is made up for numerous joints that replicate human arm joints i.e., shoulder, elbow, wrist, and fingers. All these joints are driven by motors inside which are operated by the controller. The controller is made up of set of joysticks to manipulate each joint of the robotic arm. In order to bring a robot arm to a position, we need to carefully set the joints to specific angles. This is a tedious task. We have inverse kinematics to calculate the joint angles based on the position of end effecter. But still there is need for interactive system that is far better than using a joystick to control a robotic arm. With the invention of the Leap Motion we are able to track a whole human hand including joints (wrist, fingers) using infrared rays, accurate to millimeter. This paper walks through the research carried out on a robotic arm integrated with Leap Motion Sensor to mimic the action of a human hand which makes real-time controlling more user friendly than ever before.

II. UNDERSTANDING ROBOTIC ARM

A. Degree of Freedom [1]

In mechanics, Degree of freedom refers to the number of parameters that define the workspace of a machine. Higher the degree of freedom, the more flexible the robotic arm is. Each joint is a DoF in robotics. In order to reach any point in space, we require a robot with 6 DoF. 6 DoF is the highest number that a robotic arm can have, however few-advanced robotic arms have 7 DoF for special tasks. We will deal with 4 DoF Robotic arm in this research paper.

---

they- Robotic arms

DoF- Degree of Freedom

B. Types of Joints

Prismatic- These joints in robot are for linear motion. They can only move in two directions such as front-back, up down, forward backward, left right.

Revolute- These joints in robot are for angular motion, they can be turned to a specific angle in its range. Ex. 45.6^0

Most of the industrial robotic arms are designed with revolute joints to have dexterity that prismatic joints lack.

III. EXISTING CONTROLLERS

Controllers that are available now to manipulate robot arms are clustered with numerous buttons and controls, and are very hard to learn by the operator. Since humans cannot calculate the required joint angles as fast as computer, it will be a time consuming process to bring arm to a position using the controller. Figure 1 shows the traditional controller to handle a robotic arm.

Figure 1. A traditional robotic arm controller.

It is evident from above figure that existing controllers are quite complex to manipulate robotic arm. We need much simpler interactive system to handle this task. The best way could be making the robotic arm mimic a human hand. And this can be done using an IR sensor called Leap Motion.

IV. LEAP MOTION SENSOR

The Leap Motion Sensor is a small USB peripheral device, designed to place on a physical desktop, facing upward. Using two monochromatic IR cameras and three infrared LEDs, the device observes a roughly hemispherical area to a distance of about 1 meter. The data that is obtained from the Leap Motion Sensor contains various details of hands such as position, orientation, and frame ids, finger bone information. Leap Motion Sensor can track a human hand 300 times a second with millimeter accuracy. This outstanding functionality is what makes it a good choice to integrate with a robotic arm. The two monochromatic IR sensors used in the device detect the heat signature from the hand while other infrared LED's detect the shape structure of the hand. Together they provide us the data of positions of each joint in hand including bone of fingers. *Figure 2* shows the top view layout of Leap Motion Sensor.

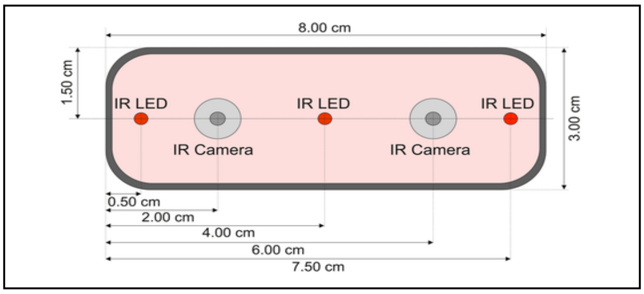

Figure 2. Top view of Leap Motion Sensor.

The two monochromatic IR cameras keep track of the movement of the hand where as remaining three IR LED's concentrate on the joints of hand. Figure 3 shows how a Leap Sensor tracks your hand when placed above it. It is the screen shot of the leap motion visualizer tool that is used to monitor the hand movements as detected by the sensor.

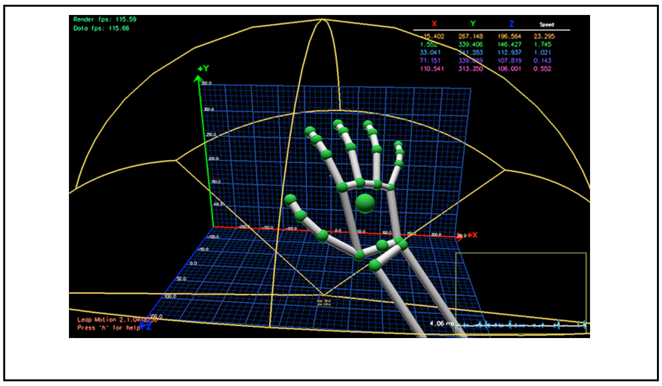

Figure 3. Leap Motion visualizer.

The sensor provides the position of our hand and fingers represented in 3D coordinates i.e. (X, Y,Z). After obtaining the position of the hand from the sensor we cannot directly send them to the robotic arm, because the arm doesn't know how to reach the position until its joints are turned to specific angles. Here comes inverse kinematics [2] that calculates the respective joint angles when the position of the end-effecter is given.

V.INVERSE KINEMATICS

Mostly used in game design and robotics, Inverse kinematics refers to the use of the kinematic equations of a robot to determine the joint parameters that provide a desired position of the end-effecter. It can also be defined as specification of the movement of a robot so that its end-effecter [4] achieves a desired position known as motion planning [3,5]. Inverse kinematics transforms the motion plan into joint actuator trajectories [6] for the robot. Inverse Kinematics plays a prominent role in calculating the path of the robotic arm by solving mathematical equations of that robot. Equations of inverse kinematics vary from robot to robot. Length of the joints, degree of freedom and end point are the parameters that are used to calculate the inverse kinematics equations of a particular robot. Hence they vary from each other. Figure 4 shows the criteria to calculate inverse kinematics when an end point in space is given.

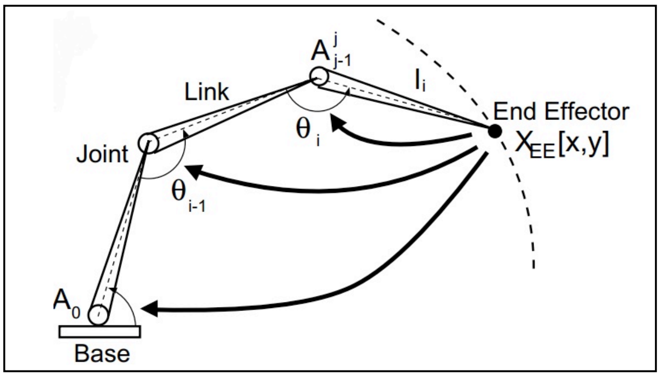

Figure 4. Calculating inverse kinematics for 3DoF Robotic Arm.

For a typical Stamford manipulator (Most used robotic arm), the following are the inverse kinematics equations that calculate joint angles.

$$c1[c2(c4c5c6 - s4s6) - s2s5c6] - s1(s4c5c6 + c4s6) = r11$$

$$s1[c2(c4c5c6 - s4s6) - s2s5c6] + c1(s4c5c6 + c4s6) = r21$$

$$-s2(c4c5c6 - s4s6) - c2s5s6 = r31$$

$$c1[-c2(c4c5s6 + s4c6) + s2s5s6] - s1(-s4c5s6 + c4c6) = r12$$

$$s1[-c2(c4c5s6 + s4c6) + s2s5s6] + c1(-s4c5s6 + c4c6) = r22$$

$$s2(c4c5s6 + s4c6) + c2s5s6 = r32$$

$$c1(c2c4s5 + s2c5) - s1s4s5 = r13$$

$$s1(c2c4s5 + s2c5) + c1s4s5 = r23$$

$$-s2c4s5 + c2c5 = r33$$

$$c1s2d3 - s1d2 + d6(c1c2c4s5 + c1c5s2 - s1s4s5) = Ox$$

$$s1s2d3 + c1d2 + d6(c1s4s5 + c2c4s1s5 + c5s1s2) = Oy$$

$$c2d3 + d6(c2c5 - c4s2s5) = Oz$$

Where rij are rotation parameters and Oij are orientation parameters. Together they provide us the joint angles that are necessary to keep end effecter in position. Solving these equations for every given position is a very complex task, hence after finding the required expressions from the equations, we us it in computer program to find respective joint angles. The inverse kinematics program keeps finding the joint angles for every 100ms and sends the information to the robotic arm which helps us to achieve seamless synchronization of the human arm and robotic arm.

VI. INTEGRATION

For the research, we have used OWI robotic arm, which is a typical 4 DoF robotic arm. OWI can be connected to a computer using its USB interface circuit. It can be controlled manually using commands from the computer for each joint. We have created a java program that sends commands to the OWI to set a joint to a given angle. This is done using a technique called timed angle turn, in which joint motors are made to run for a specific period of time until it reaches an angle. So, finally all the angles that are calculated using inverse kinematics are converted into timed angle turns. The 4 DoF robotic arm OWI is shown in Figure 5.

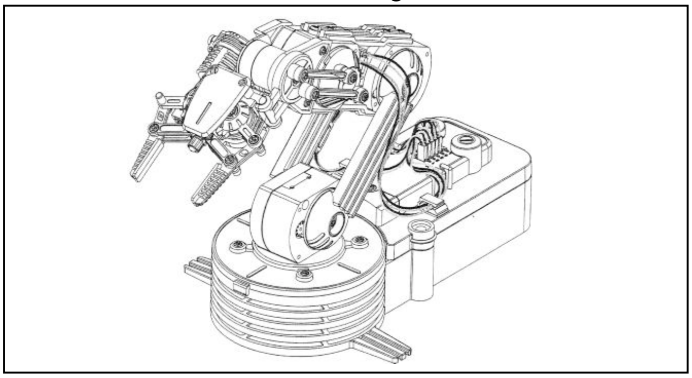

Figure 5. OWI Robotic Arm Edge.

VII. LEAP MOTION API

API stands for Application program interface. It is a collection of methods in a programming language that can be used by developer to access the functionality. Leap Motion API provides us all the required methods to connect with the sensor and retrieve the hand information for processing inverse kinematics. API for Leap Motion Sensor is available in many languages such as C, JAVA, Python, and Objective-C etc., Here we use Java API to bridge the gap between the sensor and robotic arm. In short, using API we can create a communication channel between sensor and the robotic arm using a JAVA program that calculates inverse kinematics. Figure 6 shows a section of java code that is used to track the human hand using Leap Motion sensor.

```
    for (Finger finger : frame.fingers()) {
             Hand hand=Frame.hands().frontMost();

if(finger.type()==Finger.Type.TYPE_INDEX)
                   {
 for(Bone.Type boneType : Bone.Type.values()) {
Bone bone = finger.bone(boneType);
 if(bone.type()==Bone.Type.TYPE_DISTAL)
                      {
    x1=(int)bone.center().getX();
    y1=(int)bone.center().getY();
    z1=(int)bone.center().getZ();
    robotArm.moveTo(x1(-1*z1),y1);
                      }
                    }
                  }
                }
```

Figure 6. Java code to track the hand using Leap Motion API

The code shown above gets the X, Y,Z position of the tip of index finger, which is called Distal Phalanx[7] in medical terms and it is forwarded to the method called "moveTo" to calculate the inverse kinematics for that position and perform timed angle turns on OWI robotic arm. The above code repeatedly gets the position and keeps calculating inverse kinematics 100 times in a second. This helps to achieve smooth synchronized movements while mimicking human arm. This approach can be used for robots with any degrees of freedom. It is quite obvious that this is the simplest way to handle a robotic arm in real time.

VIII. CONCLUSION

Gesture control is undoubtedly the simplest way to control a complex robot, with the help of a sensor such as Leap Motion and with few mathematical equations we are able to achieve seamless synchronization between human hand and a robotic arm. However we cannot achieve the dexterity of a human hand completely, this system can be used in all the areas where robotic hands are manually controlled and reduces lot of effort and being used while handling with traditional controllers. Statistics from International Federation of Robotics – IFR reveal that 200,000 units of industrial robotic arms are shipped every year and expected to increase 3 times by the end of 2020. Also new researches on planetary objects like mars require robotic arms to carry experiments on their surfaces in extreme conditions. This rapid increase in usage of robotic arms also increases the necessity of simpler and interactive environment to control them. Gesture control could be the answer for it and Leap Motion can be the revolution in this industry.

REFERENCES

[1] Sawhney, D.; Khera, P.; Keskar, A.G., "Robotic Arm with Four Degrees of Freedom," Emerging Trends in Engineering and Technology, 2008. ICETET '08. First International Conference on, vol., no., pp.802, 806, 16-18 July 2008

[2] Ying-Shieh Kung; Ming-Kuang Wu; Bui Thi Hai Linh; Tz-Han Jung; Shin-Hon Lee; Wen-Chuan Chen, "Design of inverse kinematics IP for a six-axis articulated manipulator," Automatic Control Conference (CACS), 2013 CACS International , vol., no., pp.300,305, 2-4 Dec. 2013

[3] Jianqing Peng; Wenfu Xu; Zhiying Wang; Deshan Meng, "Analytical inverse kinematics and trajectory planning for a 6DOF grinding robot," Information and Automation (ICIA), 2013 IEEE International Conference on , vol., no., pp.834,839, 26-28 Aug. 2013

[4] Mehmood, N.; Ijaz, F.; Murtaza, Z.; Ali Shah, S.I., "Analysis of End-Effector position and orientation for 2P-3R Planer Pneumatic Robotic Arm," Robotics and Emerging Allied Technologies in Engineering (iCREATE), 2014 International Conference on , vol., no., pp.47,50, 22-24 April 2014

[5] Gochev, K.; Narayanan, V.; Cohen, B.; Safonova, A.; Likhachev, M., "Motion planning for robotic manipulators with independent wrist joints," Robotics and Automation (ICRA), 2014 IEEE International Conference on, vol., no., pp.461, 468, May 31 2014-June 7 2014

[6] Werner, A.; Lampariello, R.; Ott, C., "Trajectory optimization for walking robots with series elastic actuators," Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on, vol., no., pp.2964, 2970, 15-17 Dec. 2014

[7] Hashimoto, Hiroshi; Sasaki, Akinori; Makino, Koji; Mitsuhashi, Kaoru, "Reduced DoFs of digital hand based on anatomy for real time operation," Informatics in Control, Automation and Robotics (ICINCO), 2014 11th International Conference on , vol.01, no., pp.764,768, 1-3 Sept. 2014