Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

# Intuitive Control of Virtual Robots using Transformed Objects as Multiple Viewports

Rajeevlochana G. Chittawadigi
*Department of Mechanical Engineering*
*Amrita School of Engineering, Bengaluru*
*Amrita Vishwa Vidyapeetham, India*
rg_chittawadigi@blr.amrita.edu

Takafumi Matsumaru
*Graduate School of Information, Production and Systems*
*Waseda University, Kitakyushu*
*Fukuoka, Japan*
matsumaru@waseda.jp

Subir Kumar Saha
*Department of Mechanical Engineering*
*Indian Institute of Technology Delhi*
*New Delhi, India*
saha@mech.iitd.ac.in

***Abstract* - In this paper, the integration of Leap Motion controller with RoboAnalyzer software has been reported. Leap Motion is a vision based device that tracks the motion of human hands, which was processed and used to control a virtual robot model in RoboAnalyzer, an established robot simulation software. For intuitive control, the robot model was copied and transformed to be placed at four different locations such that the user watches four different views in the same graphics environment. This novel method was used to avoid multiple windows or viewports and was observed to have a marginally better rendering rate. Several trials of picking up cylindrical objects (pegs) and moving them and placing in cylindrical holes were carried out and it was found that the manipulation was intuitive, even for a novice user.**

***Index Terms –Virtual Robot, Intuitive Control, Multiple viewports, Transformed Objects, RoboAnalyzer.***

## I. INTRODUCTION

Serial robots are used extensively in industries, and other applications such as health, space, entertainment, etc. Generally, the robots have a manipulator arm being controlled through a computer or a dedicated controller. The robots can be controlled using joystick like device known as a Teach Pendant. The Teach Pendant has buttons using which a robot can be moved in joint-space or Cartesian-space, referred to as joint jogging and Cartesian jogging, respectively. Robots can also be controlled using offline and online programming. The former deals with simulation of robot motion in a virtual environment and upon successful execution, the program can be transferred to the robot controller for uninterrupted execution. The latter deals with the physical robot in the loop and various sensors connected to the robot and in the robot's workspace. Here, the current status of the sensors and robot configuration are used to plan motion of the robot. Generally, vision based sensors are used in such applications.

Another intuitive approach to control the motion of a robot is through master-slave manipulation. Devices such as exo-skeleton can be worn by user on his/her arm and motion of the human arm is tracked by sensors. The motion of the wrist is mapped to the motion of the robot arm, thus mimicking the motion of the human. By far, this is one of the good approaches to control a robot located remotely and can be used in applications such as handling hazardous items. Though an exo-skeleton is an intuitive method of controlling a robot, it has its own disadvantages. The major one being the fatigue on the human who has to wear the exo-skeleton.

To counter this issue, one can use new sensors or devices that have emerged in the market such as Microsoft Kinect, Leap Motion, etc., which can also act as input device to a robot's motion. These devices generally have cameras which determine the position and orientation of objects in its image workspace, using various image processing techniques. An example of Kinect based control of robot was presented in [1], where the position of index-finger and thumb of the user was tracked by Kinect sensor to control a virtual robot and then an actual robot located remotely. Similar work on usage of Leap Motion to track human hand and control of virtual and actual robots are reported in [2-3]. A recent work of the second author is the usage of Leap Motion controller to detect virtual grasping of a holographic image [4] and its application in controlling virtual model of an industrial robot [5]. It is reported that the devices mentioned above are better and more intuitive to control a robot, than using a traditional teach pendant.

In this paper, an integration of Leap Motion sensor and controller, and the Virtual Robot Module (VRM) of RoboAnalyzer software has been presented. RoboAnalyzer [6] is a 3D model based robot simulation software developed by the first and third authors since 2009 and has been used actively for teaching robotics concepts.

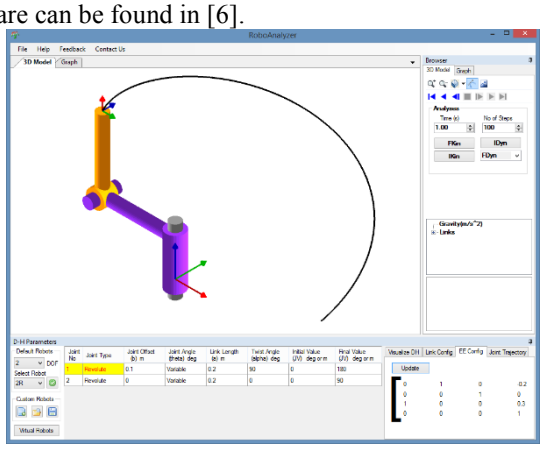
The robot shown in a simulation software can usually be viewed from one of the standard views such as top view, right side view, isometric view, etc. However, from only one particular view, controlling the robot can be confusing as the depth of the environment is not felt by the user. Hence, one can resort to having multiple views or viewports of the robot so that different views such as front, top, isometric, etc. can be shown to the user at a time and the user can use them to understand the depth factor better. In this paper, a novel methodology of transformed copies of the virtual robot is proposed that can be used for intuitive control. Another alternative is the usage of Virtual Reality (VR) and Augmented Reality (AR) based methods to visualize and teach robot programming, as reported in [7]. In this paper, the AR/VR aspects are not considered.

The rest of the paper has a brief overview of RoboAnalyzer, VRM and the new features added to it in Section II. Existing and novel proposed methods to display multiple views are explained along with implementation details in Section III, followed by the integration of Leap Motion and VRM in Section IV. The results of the trials conducted are discussed in Section V, followed by the conclusions.

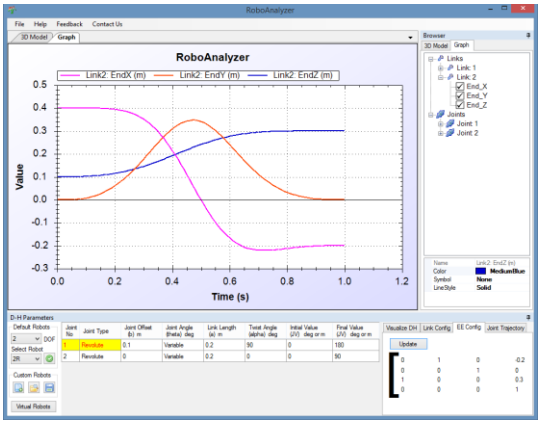
978-1-7281-6321-5/19//$31.00 ©2019 IEEE

## II. ROBOANALYZER

RoboAnalyzer is a 3D model based robot simulation software developed by the first and third authors since 2009. The topics of homogeneous transformation, Denavit-Hartenberg (DH) parameters, forward and inverse kinematics of serial robots can be demonstrated and taught effectively to the students. Similarly, the concepts of forward and inverse dynamics can be explained using the plots and animations obtained in the software. Screenshots of two main interfaces of the software are shown in Fig. 1(a-b). More details about the software can be found in [6].



(a) Skeleton model of robots using DH parameters and its forward kinematics




(b) Plots of kinematic analysis

Fig. 1 RoboAnalyzer interfaces for effective teaching of robotics concepts.

Virtual Robot Module (VRM) is a standalone application in RoboAnalyzer software that has CAD models of more than 20 industrial manipulators. It has ability to perform joint-jogging, Cartesian jogging and Cartesian motion [8]. It also has a record and playback feature that can be used to teach robot and playback its motion.

VRM application supports server-client architecture where the VRM application acts as a server and any client application connected to it can update the values of the robot shown to the user. It can be connected either through COM (Component Object Model) interface or as a .NET framework DLL (Dynamic Linked Library). The COM based interface was used to integrate VRM with Robotics Toolbox for MATLAB and Microsoft Excel, whose details are given in [9].

### A. *CellObjects in Virtual Robot Module*

In this paper, a new feature added to the VRM is reported. Additional 3D objects can be imported into the virtual environment. These are referred to as *CellObjects*. They can be further classified as *Fixed* and *Movable*. As the names suggest, the former cannot be moved whereas the latter can be moved, programmatically. A robot with several *CellObjects* is shown in Fig. 2. Here, a table is kept in the robot work cell and on the table are two peg-stands. Eight pegs are placed in the holes of one of the peg-stands. Here, the table and peg-stands are *Fixed CellObjects,* and the pegs are *Movable* ones.

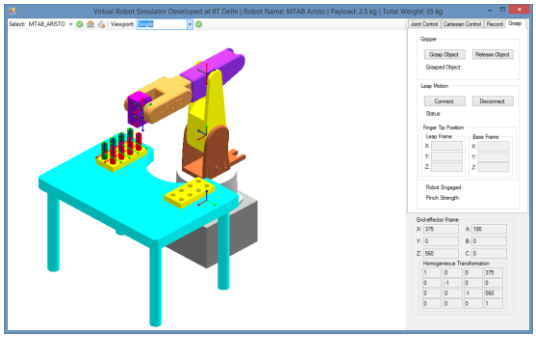
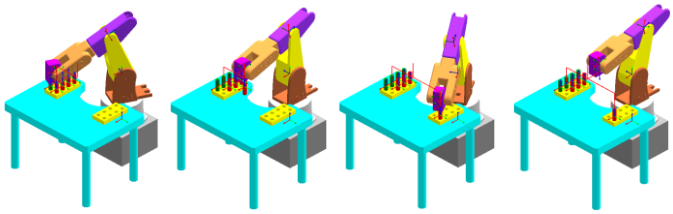


Fig. 2 *CellObjects* in Virtual Robot Module (VRM) in RoboAnalyzer.

The robot in the environment can be jogged (joint or Cartesian) such that its end-effector (EE) is near one of the pegs. At that configuration, a new button "Grasp Object" can be pressed, which will determine the relative transformation between the nearest peg and the EE frames. If the distance is below an acceptable value, the emulation of a vacuum gripper is achieved by ensuring that the peg and the EE do not have relative motion. Hence, any robot motion with the grasped object is achieved. There after the peg can be moved to one of the holes of the empty peg-stand. Sample Cartesian motion to move a peg between two peg-stands is shown in Fig. 3.



(a) Grasp (b) Transit (c) Release (d) After release

Fig. 3 Grasping and Release of movable *CellObject* (peg) in VRM.

Here, the exact location of each peg was known and hence the motion was achieved without any issue. However, if one has to use Cartesian jogging (incremental motion of EE along the three principle directions), the depth information will not be visualized properly that may result in improper motion and pickup of the objects. To overcome this issue, the authors propose a multi-view approach as explained next.

## III. MULTI-VIEW VISUALIZATION

A CAD (Computer Aided Design) software such as Autodesk Inventor, SpaceClaim, etc. has a 3D environment which has 3-dimensional objects in it. The object(s) can be

transformed (translated or rotated) using the specific tools. Also, the camera position and the look at angle of the camera can be altered to emulate looking at the object from different viewpoints, for effective detailing or manipulation of shape. The standard orthographic views available are front, top and right/left which have standard treatment for the positioning of the camera and where it has to be looking at. Similarly, the camera position and orientation for isometric or any other views can be determined and set.

Similar views are possible to be implemented in custom graphics software, such as RoboAnalyzer or VRM. Most of the robot simulation software allow only one 3D environment which has the virtual robot in it. Using the mouse, keyboard or by using special buttons/functionalities, the view of the camera can be changed.

However, it still results in one view of the robot. In the standard version of VRM developed using OpenGL (OpenTK: a C# wrapper for OpenGL), the single view was not sufficient to have intuitive control of the virtual robot. The depth information was required at times and swapping between the views was causing more hindrance than help. Hence, the authors explored ways of showing four views of the robot, so that by combining more than one view, the user can get the depth information and move the robot easily. The options available are explained, followed by the novel method proposed for multi-view environment.

### *A. Multiple OpenGL Controls*

The VRM application was made to show four independent instances of OpenGL controls (3D canvas on which rendering takes place). Each control was supplied with a robot model which was drawn on it. As the drawing happened on four different canvases and due to individual swapping of buffers, the application was very slow to react. Hence it was not considered further.

### *B. Multiple Viewports using OpenGL*

OpenGL has a provision to define *Viewport* for a specific region inside the canvas/control, using which, any number of viewports can be displayed. This was used to display four views of the robot in the VRM software. The output and the representative code are shown in Fig. 4 and Fig. 5, respectively.

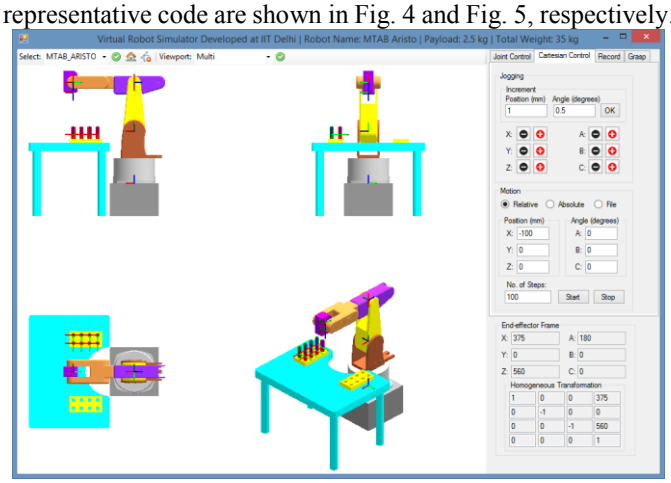


Fig. 4 Multiple Viewports: Top-left: Right view; Top-right: Front view; Bottom-left: Top view; Bottom-right: Isometric view

```
GL.Clear(ClearBufferMask.ColorBufferBit |
         ClearBufferMask.DepthBufferBit);

GL.LoadIdentity();

GL.Viewport(0, 0, width/2, height/2);
GL.PushMatrix();
SetView(LookAtView.Top);
    DrawRobot();
GL.PopMatrix();

GL.Viewport(width/2, 0, width/2, height/2);
GL.PushMatrix();
SetView(LookAtView.Iso);
    DrawRobot();
GL.PopMatrix();

GL.Viewport(0, height/2, width/2, height/2);
GL.PushMatrix();
SetView(LookAtView.Right);
    DrawRobot();
GL.PopMatrix();

GL.Viewport(width/2, height/2, width/2, height/2);
GL.PushMatrix();
SetView(LookAtView.Front);
    DrawRobot();
GL.PopMatrix();
```

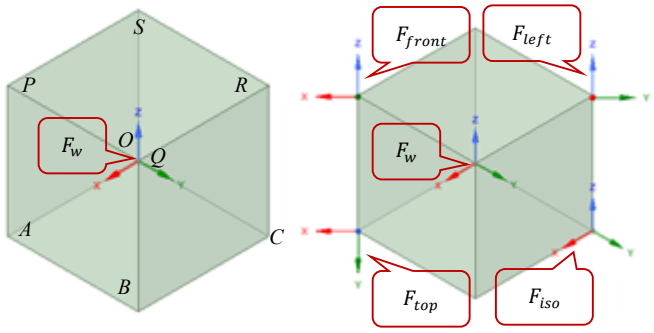
Fig. 5 Representative code for multiple viewports in OpenTK.

In the above implementation, the four robot views are useful to control the robot as the depth information is better understood by the user. An alternative method is proposed next.

### *C. Multiple Transformed Objects/Robots*

The authors explored ways of representing multiple views in the same OpenGL environment, rather than using four different controls or four separate viewports. In this paper, they propose a scenario where four copies of the robot (or any object) can be positioned such that when viewed from a particular position, the objects appear to be from different positions.

Consider a cube of a known length ($a$), positioned such that one of its vertices is at the origin ($O$) of the World coordinate system, as shown in Fig. 6(a). A frame attached there is referred to as $F_w$. The camera can be assumed to be in the direction of unit vector along the body diagonal ($OQ$) of the cube, starting from point $O$. The unit vector along $OQ$ is $\left[\frac{1}{\sqrt{3}} \quad \frac{1}{\sqrt{3}} \quad \frac{1}{\sqrt{3}}\right]^T$, which is very important in the proposed methodology.




(a) Cube with $F_w$ at origin of the World frame

(b) Cube with four frames at points $A$, $C$, $P$ and $R$

Fig. 6 Cube of length '$a$' used for derivation of proposed method.

Now, four frames are attached at points $A$, $C$, $P$ and $R$ of the cube such that the frames when looked from the camera ($OQ$) appear as shown in Fig. 6(b). These points are chosen deliberately so that these four frames appear below or beside each other, to represent orthographic views.

Frame $F_{iso}$ is attached at point $C$, such that it remains parallel to the $F_w$. Therefore, the homogenous transformation matrix (HTM) of $F_{iso}$ with respect to $F_w$ is given by

$$T_{iso} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & a \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Frame $F_{top}$ at point $A$ has its $Z$ axis along the body diagonal ($OQ$). Its $X$ axis is along the line joining $CA$, which is given by unit vector $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \end{bmatrix}^T$. Using the cross-product rule, the $Y$ axis of $F_{top}$ can be determined as

$$Y_{top} = Z_{top} \times X_{top} = \begin{bmatrix} \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{bmatrix} \times \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \\ \frac{-2}{\sqrt{6}} \end{bmatrix} \quad (2)$$

Therefore, the HTM corresponding to $F_{top}$ with respect to $F_w$ can be determined as

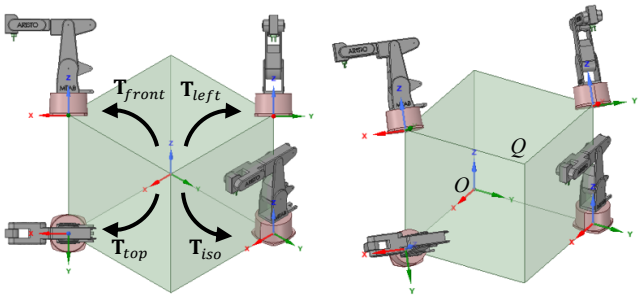
$$T_{top} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} & a \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} & 0 \\ 0 & \frac{-2}{\sqrt{6}} & \frac{1}{\sqrt{3}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

For frame $F_{front}$, its $Y$ axis is along the body diagonal and its $X$ axis is along line $RP$. Similarly for $F_{left}$, its $X$ axis is along the body diagonal and its $Y$ axis is along line $PR$. Their homogenous transformations with respect to $F_w$ can be similarly determined as

$$T_{front} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{6}} & a \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{6}} & 0 \\ 0 & \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{6}} & a \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$$T_{left} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{2}} & \frac{-1}{\sqrt{6}} & 0 \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{6}} & a \\ \frac{1}{\sqrt{3}} & 0 & \frac{2}{\sqrt{6}} & a \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

Note that in the above transformations, the factor '$a$' is the only variable and hence, depending on the size of the object to be displayed, it can be assigned. Copies of a robot model after transformation using the above HTMs are shown in Fig. 7(a). The same scenario when viewed from a direction other than the body-diagonal ($OQ$) is shown in Fig. 7(b).




(a) Camera along body-diagonal (b) Camera at other location

Fig. 7 Demonstration of novel method to display multiple views of same object in SpaceClaim CAD software.

The proposed method for display of multiple views in the same 3-dimensional environment was implemented VRM, as shown in Fig. 8, whose representative code is given in Fig. 9.

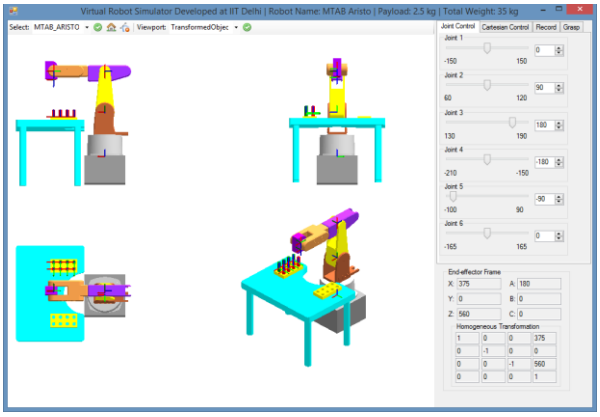


Fig. 8 Multiple views using transformed objects in VRM.

```
GL.Clear(ClearBufferMask.ColorBufferBit |
         ClearBufferMask.DepthBufferBit);

GL.PushMatrix();
GL.MultMatrix(ref transIso);
    DrawRobot();
GL.PopMatrix();

GL.PushMatrix();
GL.MultMatrix(ref transTop);
    DrawRobot();
GL.PopMatrix();

GL.PushMatrix();
GL.MultMatrix(ref transFront);
    DrawRobot();
GL.PopMatrix();

GL.PushMatrix();
GL.MultMatrix(ref transLeft);
    DrawRobot();
GL.PopMatrix();
```

Fig. 9 Representative code for transformed objects in OpenTK.

Compared to the code in Fig. 5, the code here has less calls to the underlying graphics library and also has reduced number of matrix multiplications.

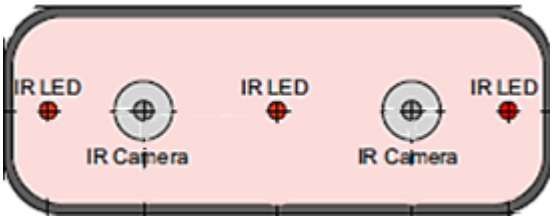
During various trials of same robot motion on Intel core i5 laptop running Windows 8.1 operating system, the multiple viewports consumed around 14 ms for every step, whereas the proposed methodology consumed around 13 ms. Though the time saved in the proposed methodology is marginal, it may be beneficial for a very large system, which will be reported in the future.

## IV. Integration with Leap Motion

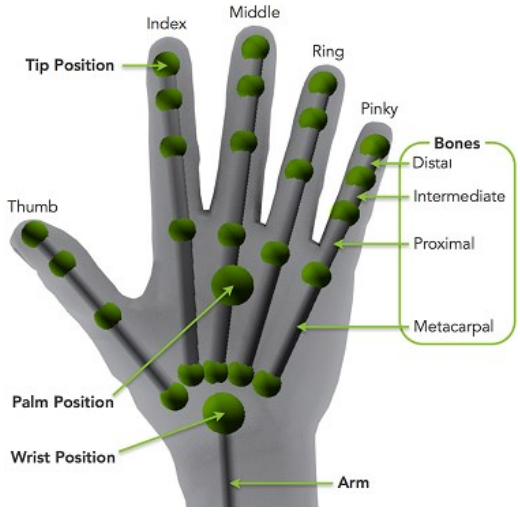
Leap Motion is an inexpensive sensor which has three infrared (IR) emitters and two IR cameras and can track 10 fingers of the two human hands. The device sends the data to the computer connected to it through its controller software which can be accessed in a variety of programming languages. The device can measure a range between 25mm and 800mm, above the device, in a hemispherical workspace. In this paper, Leap Motion has been integrated using Visual C# language. The layout of sensors on it are shown in Fig. 10(a).

### A. Motion Capture of Human Hand

Leap Motion Controller software has to be installed on a computer and a reference of it has to be added in Visual C# project in Visual Studio environment. The Leap Motion device is connected using a USB cable to the target computer. Controller object is the gateway to the Leap Motion device and through it, a *Frame* can be captured which has information of the current position of all the 10 fingers of the two human hands, which are in its visible range. The nomenclature of the fingers, joints and bones of a hand, as used in Leap Motion, is illustrated in Fig. 10(b).




(a) Sensors and cameras on Leap Motion




(b) Nomenclature of fingers in human hand

Fig. 10 Leap Motion device for tracking hand movement [10].

For every frame, the program can read the hand's palm or any finger's *StabilizedTipPosition*, with respect to the coordinate frame attached at the center of the device. The stabilized tip position of the current frame and the previous frame can be used to determine the incremental motion of the palm or fingers. More details about the working of the device and the details on the Software Development Kit (SDK) to be used for integration with other applications are available at [10]. The orientation of the hand can also be obtained using suitable library calls.

### B. Tracking of Hand in Virtual Robot Module

The VRM application was included with references to the DLLs provided for Leap Motion Controller. Upon connecting to the Leap Motion, the data is received in the application for every frame. To manipulate the virtual robot, the origin of the Leap Motion coordinate frame was assumed 300 mm below the home position of the EE frame. This is to ensure that pegs are in the visible workspace of Leap Motion.

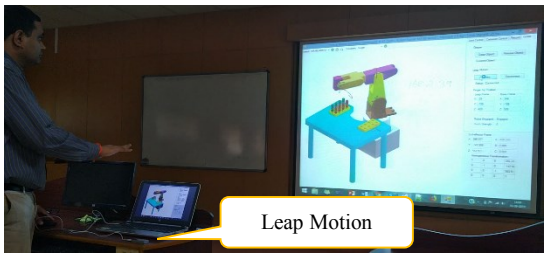
For every frame captured, the coordinates of palm position of the visible hand are determined in the Leap Motion frame. The position is then obtained in the frame attached to the base link of the robot, which is also used for all robot motion calculations. If the palm position is within a sphere of 50mm with respect to the origin of the EE frame, the robot is engaged and further motion of the palm is provided as incremental motion to the EE. The correlation between the incremental motion of the EE and the corresponding incremental motion of the joint angles is determined using Jacobian inverse method, discussed in [8]. Thus, the robot's EE mimics the motion of the human hand. The scaling of the motion can be decided by the user to suit the size of the robot and the workspace requirements. If the palm position goes beyond the virtual sphere, the robot is disengaged and the user has to again reach near the EE to start controlling it.

### C. Grasping and Release of CellObject

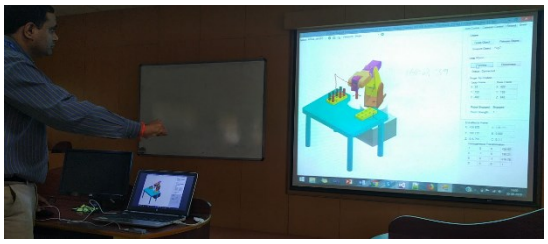
Leap Motion tracks the strength of the grasp of the human hand. If the hand is closed to form a fist, the grasp strength has a value of 1 and opened fist has a value of 0. A partial closed fist has a value in between. In the implementation of the VRM, the open fist was used to manipulate the EE and once it reaches closer to a peg (*CellObject*), the user has to close the fist. This calls the function to grasp the nearest *CellObject*. With the closed fist, the user moves the peg to desired location and then opens the fist to release the object. With the opened fist, the user continues the pick-and-place tasks.

## V. Results of Trials

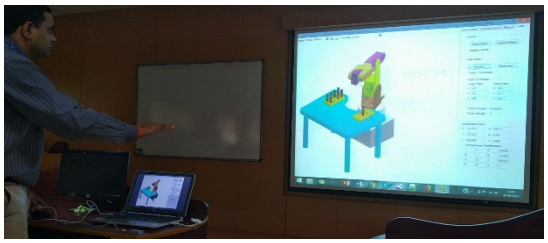
Several trials were conducted on the pick-and-place operations using VRM and Leap Motion. First, the users were subjected to the manipulation tasks by showing only one isometric view of the robot. The users were able to pick pegs with some approximate motion, as the depth information was not perceived correctly. A demonstration of the user interacting with single view is shown in Fig. 11(a-c). Here, the user is asked to look at a projector screen, displaying the VRM software. Similar tests with multiple views, as proposed by the authors, were conducted, as illustrated in Fig. 12(a-c).




(a) Manipulation with open fist

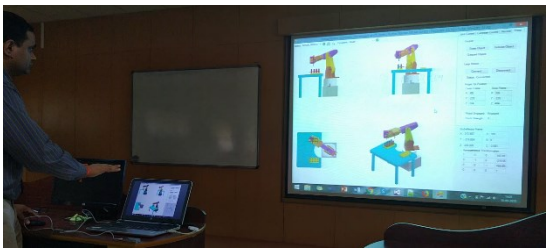


(b) Grasped manipulation with closed fist

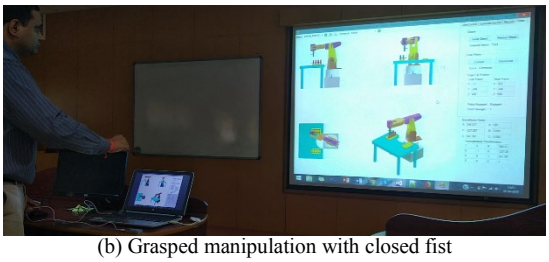


(c) After release of peg (open fist)

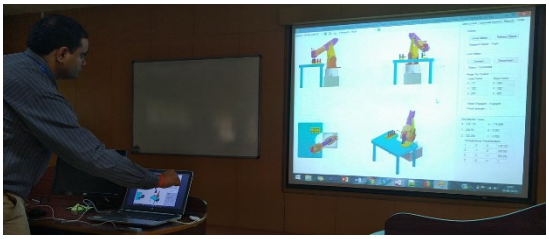
Fig. 11 Manipulation of robot using single view in VRM.



(a) Manipulation with open fist



(b) Grasped manipulation with closed fist



(c) During release of peg (closed fist)

Fig. 12 Manipulation of robot using mutli-view in VRM.

The feedback collected from a few users can be summarised by the following statements:

- The Single View task was confusing, but the robot motion was without any observable lag or jerk.
- The Multi-View task was intuitive as the depth feeling was better. However, it requires some practice to get used to merging information from multiple views in the user's mind to act on the motion. Moreover, there was a slight lag or jerks at times.

The tests were conducted on a regular laptop. In future, the tests will be conducted on workstations with better computing power and RAM to provide improved performance, especially in multi-view scenarios. Also, time based study and accuracy and usefulness of the master-slave manipulation would be performed and reported.

## VI. Conclusions

In this paper, the Virtual Robot Module (VRM) of RoboAnalyzer has been improvised to have multi-views of the robot and the objects in the workspace. The multi-views were implemented using a novel methodology of transformed replica of the objects in a single environment, as compared to four separate environments or viewports in OpenGL. The proposed methodology has elegance and can be used in other similar applications. Leap Motion sensor, an inexpensive camera (around 80 USD) has been integrated with the VRM software to track the position of human hand and its grasp and suitable control a virtual robot to perform pick-and-place operations, in an intuitive manner. The VRM version is available at http://www.roboanalyzer.com for download and usage.

## References


[1] G. Du, P. Zhang, J. Mai and Z. Li., "Markerless kinect-based hand tracking for robot teleoperation." *International Journal of Advanced Robotic Systems*, vol. 9(2), Aug 2012.

[2] D. Bassily, C. Georgoulas, J. Guettler, T. Linner and T. Bock, "Intuitive and adaptive robotic arm manipulation using the leap motion controller," in *Proc of 41st International Symposium on Robotics (ISR Robotik)*, Munich, Germany, 2014.

[3] U. Agell, "Robot manipulation based on Leap Motion – for small and medium sized enterprises," M. S. Thesis, Department of Engineering Science, University West, Trollhattan, Sweden, 2016.

[4] M. Jiono, T. Matsumaru, "Interactive Aerial Projection of 3D Hologram Object," in *Proc IEEE International Conference on Robotics and Biomimetics*, Qingdao, China, 2016.

[5] T. Matsumaru, A. I. Septiana and K. Horiuchi, "Three-dimensional Aerial Image Interface, 3DAII," *Journal of Robotics and Mechatronics*, vol. 31(5), pp. 657-670, Oct 2019.

[6] R. S. Othayoth, R. G. Chittawadigi, R. P. Joshi and S. K. Saha, "Robot kinematics made easy using RoboAnalyzer software," *Computer Applications in Engg Education*, vol. 25(5), pp. 669-680, Sept 2017.

[7] J. Wassermann, A. Vick, and J. Krüger, "Intuitive robot programming through environment perception, augmented reality simulation and automated program verification," *Procedia CIRP*, vol. 76, pp.161-166, Aug 2018.

[8] R. O. M. Sadanand, R. G. Chittawadigi, and S. K. Saha, "Virtual Robots in RoboAnalyzer Software," in *Proc The 1st Int & 16th Nat Conf on Machines and Mechanisms*, Roorkee, India, 2013.

[9] R. O. M. Sadanand, R. G. Chittawadigi, R. Joshi, and S. K. Saha, "Virtual Robots Module: An effective visualization tool for Robotics Toolbox," in *Proc 2nd Int Conf on Advances in Robotics*, Goa, India, 2015.

[10] Website: https://www.leapmotion.com/, accessed in Nov 2018.