Al-Khwarizmi Engineering Journal, Vol. 17, No. 2, June, (2021)
P. P. 18- 30

Al-Khwarizmi Engineering Journal

The Application of Virtual Reality to (Mechatronics Engineering) by Creating an Articulated Robotic Work Cell Using EON Reality V9.22.24.24477

Aiman Al-Allaq * **Nebojsa Jaksic** **
Hussein Al-Amili** **Dhuha Mohammed Mahmood*****
*, ** *Colorado State University/ Pueblo/Colorad/ USA*
, * *Al-Khwarizmi College of Engineering/ University of Baghdad/ Iraq*
*Email: ah.alallaq@pack.csupueblo.com
**Email: n.jaksic@csupueblo.com
***Email: alamilihussein@gmail.com
****Email: duhamohammed55@gmail.com

(Received 10 November 2020; Revised 1 March 2021; Accepted 7 March 2021)
https://doi.org/10.22153/kej.2021.04.001

Abstract

Virtual reality, VR, offers many benefits to technical education, including the delivery of information through multiple active channels, the addressing of different learning styles, and experiential-based learning. This paper presents work performed by the authors to apply VR to engineering education, in three broad project areas: virtual robotic learning, virtual mechatronics laboratory, and a virtual manufacturing platform. The first area provides guided exploration of domains otherwise inaccessible, such as the robotic cell components, robotic kinematics and work envelope. The second promotes mechatronics learning and guidance for new mechatronics engineers when dealing with robots in a safe and interactive manner. And the third provides valuable guidance for industry and robotic based manufacturing, allowing a better view and simulating conditions otherwise inaccessible.

Keywords: *EON, MATLAB, Robotics, Simulation, Virtual Reality.*

1. Introduction

Virtual Reality can be described as the field of recreating three-dimensional images and textures using computer software that utilizes hardware capabilities.

VR procedures offer truly important assignment representation that helps for arranging and reviewing robotic systems and function, for foreseeing mechanical activities, preparing mechanical frameworks administrators, and for a visual impression of non-noticeable occasions like contact powers in mechanical errands [1]. According to dos Santos [2], utilizing Virtual Reality on Professional Robotics Instruction and training will help in reinforcing tasks, improving the conception of images, narrowing the time spent on learning and education, and increasing the precision. At present, there are numerous VR advancement instruments, including World Toolkit, Vega, VirTools and EON Studio. Age Studio is more mainstream for its solid thorough capacity and perception [3].

1.1. EON Virtual Reality

Starting the system is to start the EON Studio software. Eon releases updates to their software frequently so there may be more than one version of the software installed on the computer. Find the latest release of the EON Studio in the program menu or on the desktop. The Icon

should be similar to the image in Figure 1

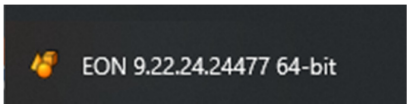

Fig. 1. EON Studio Icon.

Start the EON studio by clicking the icon. Figure 2 shows what the default EON studio layout looks like.

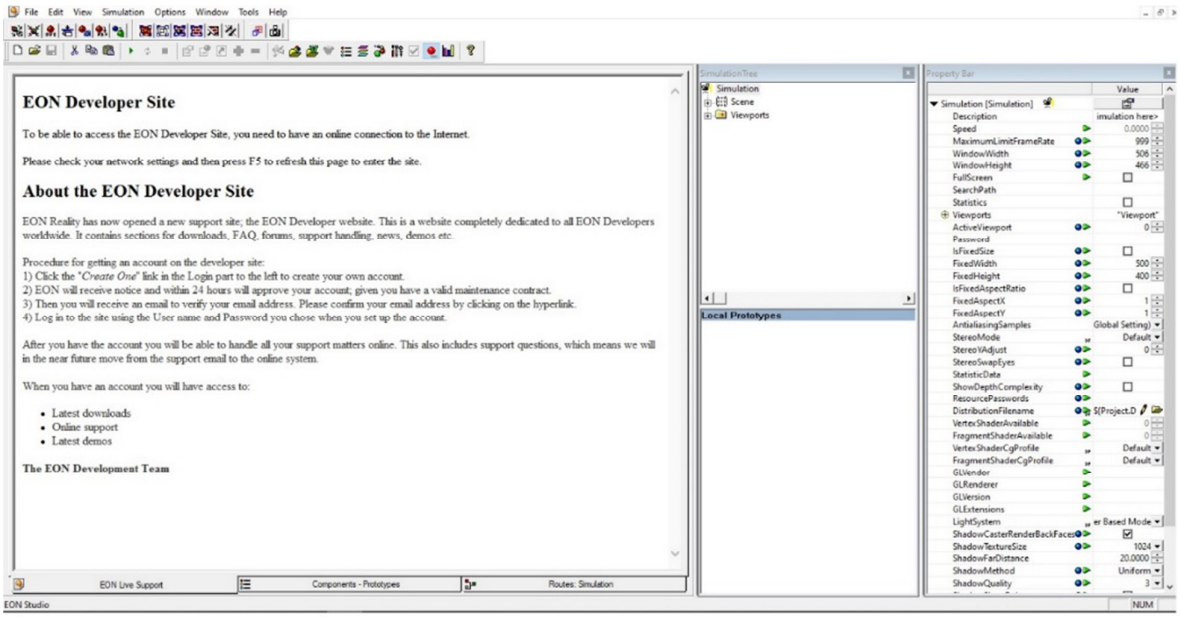

Fig. 2. Default EON Studio Layout.

Some views support context-sensitive help. This means you can bring up the correct Help page of a specific node/prototype by pressing F1 while the node/prototype is selected. The Components view, Simulation Tree window, and Property Bar support context-sensitive help. The information that is displayed depends on which view you are in when you press the F1 key:

- If you are on the Nodes tab of the Components window, pressing F1 will present a Help page describing how to use the selected node.
- If you are on the Prototypes tab of the Components window, pressing F1 will present a Help page describing how to use the selected prototype.
- If you are in the Simulation Tree window or the Property Bar, pressing F1 will present a Help page describing the node and the fields found in the Property Bar for the selected node.

Another avenue for obtaining help is through the EON Help window. This can be accessed by clicking the Help Topics button on the Toolbar or by selecting Help Topics from the Help menu. This opens the same window that is accessed using the F1 key. Rather than opening a specific topic (based on what is selected), however, it opens the "home" page from which you can generate your own search based on keywords, or you can browse through the entire Help system. When EON is opened, one can see several windows, each of which serves a certain function, those windows are classified as follows:

- Simulation Tree window
- Components window (Nodes or Prototypes)
- Routes window
- Property Bar

There are also a number of windows of lesser importance (because they are used less frequently or used only in certain circumstances). These include:

- Local Prototypes window
- Log window
- Find window
- Butterfly window

Additionally, there are numerous toolbars that put commonly used features and functionality in a convenient

location for ease of use. Figure 3 illustrates

many of these child windows and tools within the EON Studio interface, in their default locations.

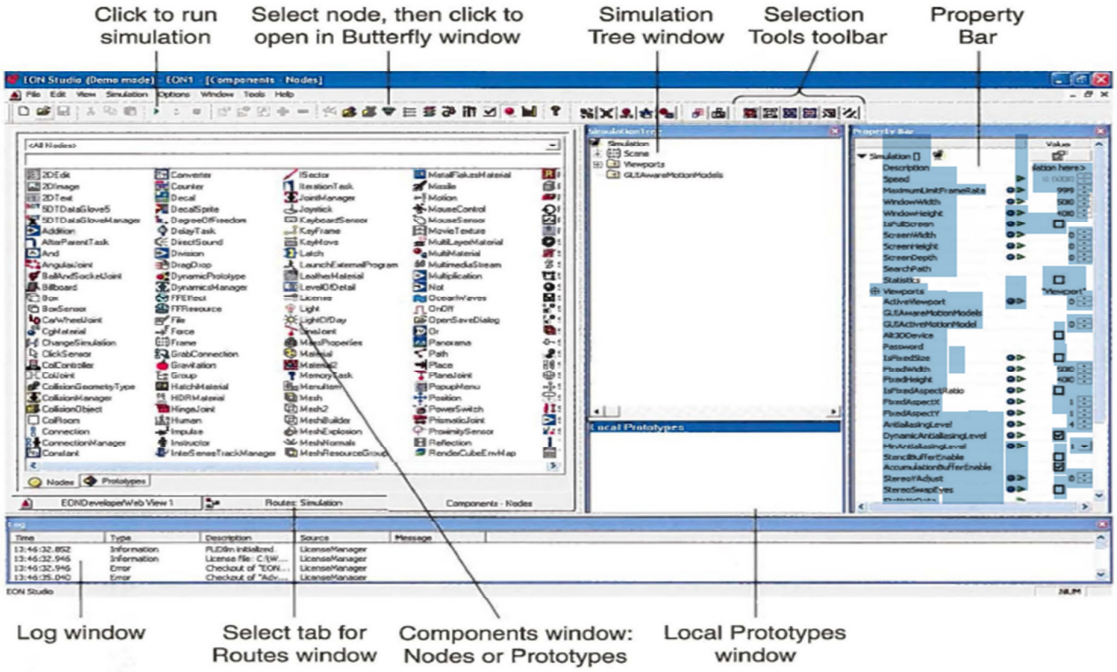

Fig. 3. EON Studio default workspace.

There are four basic types of windows in EON Studio. They are:

- Construction windows
- Library window
- Help windows
- Simulation window

An EON simulation is built by forming and connecting base node and prototype components together. This will create a certain function scheme that will produce a (or constructing) a simulation, and therefore it makes sense that some of the windows are viewed as "construction" windows. The construction windows are the Simulation Tree window and the Routes window. The Simulation Tree window is where EON applications are built by arrangement nodes into subtree orders. Each node is represented within that tree with an identifying text and a small icon. Every hub has a Property page that can be shown by double tapping the hub. The Routes window is a realistic proofreader for characterizing intelligent, occasion driven connections between hubs. You characterize how data should be sent between hubs by making sequences between nodes. Courses are made by hauling hubs from the recreation tree progression into the Routes window, where the hub symbols are associated with lines or, all the more explicitly, courses. After hubs are associated, they can trade data or trigger occasions while the reproduction is running.

1.2 System Components

There are several different components that can comprise the EON software. The fundamental programming segments, which run on Microsoft Windows frameworks are: EON studio, EON Dynamic Load, EON SDK, and EON Raptor. Age Studio is a finished graphical UI based creating device for growing constant 3D sight and sound applications zeroed in on deals/advertising, preparing/uphold [4, 5], and perception. The advancement cycle incorporates bringing in 3D objects, as a rule beginning from different demonstrating apparatuses like 3D Studio Max (otherwise called 3ds Max) and Light Wave 3D, or from PC helped plan (CAD) frameworks, for example, Solid-Works, ArchiCAD, or AutoCAD. Age Studio can import information in various record designs. After the model is imported, practices can undoubtedly be related with the models through EON's instinctive graphical programming interface, reenactment tree, hub properties, occasion steering, scripting, or even accumulated C++ code, utilizing the EON SDK along with EON Studio. The Vicon tracking system is what tracks your movements as you

interact with the simulation. The system is comprised of the three tracking cameras that are mounted above the screen, the reflective balls mounted on the controllers and 3D glasses, and the Vicon tracking software. This system has been calibrated so that the simulation will track the movements of the 3D glasses to determine the user's position and perspective. It will also track the controller so the user can use it to interact with the simulation. Double click the icon to start the Vicon Tracker software. Figure 2 shows the interface to the Vicon tracking software. Opposite to what has been mentioned by Crespo, García, and Quiroz [6], this system and its application is simpler and uses less hardware and software addons, thus making it much more efficient in terms of ease of use, complexity, functionality, and even cost.

1- Hardware

- Vicon 3D Laser Tracking System.
- Tower Workstation Computer, 9th Gen Intel Core i7, 32GB DDR4 RAM, 1TB HDD, GeForce GTX 1650 Graphics.
- Wireless KB.

2- Process Software

- EON Reality V9.22.24.24477.
- Autodesk 3D MAX.
- EON Raptor Addon.

These features make a wider range of opportunities of use in vast range of applications, such as in engineering education, manufacturing, and in the health sector.

2. Implementation

Following what has been previously stated. The implementation of the application followed the structure of the available system components as can be seen in Figure 4 below. Starting with the CAD model of the chosen robotic arm, the model was of a six-axis articulated robotic arm. The arm was constructed first Autodesk 3D MAX. The model was then imported to EON Raptor. EON Raptor is a 3D MAX plug-in tool that allows for simple interaction and manipulation of CAD models. Some interactions can be performed within 3D MAX. The completed application models can then be exported and imported to EON for further modification, the models are exported as .EOZ files.

Once the model is imported to EON, it can be seen in EON application tree with the main scene frame, this allows for the addition and modification of the imported model with the main frame of the application. For the robotic arm, several node interactions are needed to establish the relationship between the imported model parts and define the model relationship with the scene environment. Several node properties are depended upon the robotic arm forward and reverse kinematics calculations, which are solved separately using MATLAB software.

After the node properties are set, and the range of each link within the cell work envelope is properly simulated. Physics nodes are introduced to finalize the application and its node interactions within a realistic frame that is depended upon real physical properties introduced by physics node. Each frame within the robotic arm model and the whole environment scene will be provided with its own physics node that reflects its real physical properties. The resulting physics interaction between the frames will be verified and the adjusted accordingly until a suitable result is obtained. A suitable result is one where the model interaction and node simultaneous properties (such link weight, position, and speed) matches what is analytically expected.

Lastly, the final model GUI menu is designed and then built. A teach pendant of what most robotic arms available have, and contains most the of intuitive functions, such as link selector, axis movement direction and speed. This is followed by designing the final application interface window. In conclusion, the finished application can be exported either as standalone application, web-based application, advanced VR display systems, and/or imbedded systems.

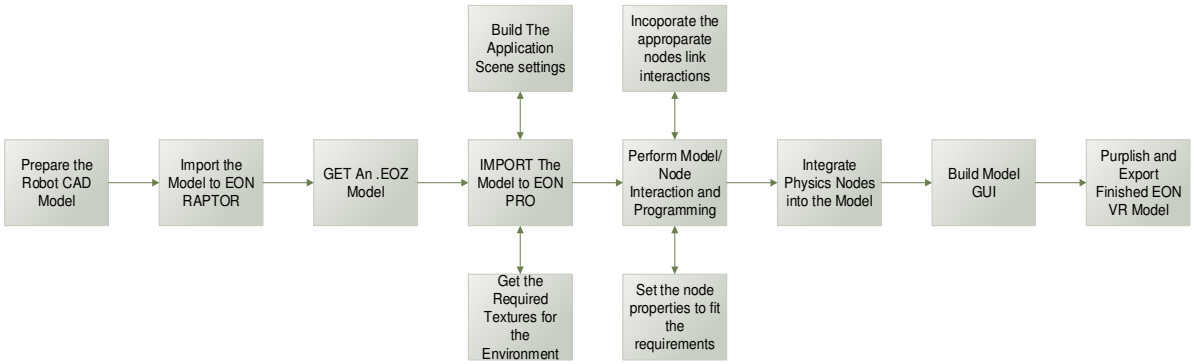

Fig. 4. EON Application Process Flow.

2.1. Modeling

The CAD model of the six-axis robotic arm was the first step in the process of building the virtual reality application. the model was of a six-axis articulated robotic arm. Robots have various axis configurations. By far most of verbalized robots, notwithstanding, highlight six tomahawks, likewise called six levels of opportunity. Six hub robots take into consideration more prominent adaptability and can play out a more extensive assortment of utilizations than robots with less tomahawks.

Axis 1

This pivot, situated at the robot base, permits the robot to turn from left to right. This broad movement stretches out the work territory to remember the region for one or the other side and behind the arm. This pivot permits the robot to turn up to an entire 180 degree range from the middle point.

Axis 2

This pivot permits the lower arm of the robot to stretch out forward and in reverse. It is the hub controlling the development of the whole lower arm.

Axis 3

The pivot broadens the robot's vertical reach. It permits the upper arm to raise and lower. On some expressed models, it permits the upper arm to reach behind the body, further extending the work envelope. This hub gives the upper arm the better part access.

Axis 4

Working related to the hub 5, this hub helps in the situating of the end effector and control of the part. Known as the wrist move, it pivots the upper arm in a round movement moving parts between level to vertical directions.

Axis 5

This pivot permits the wrist of the robot arm to tilt here and there. This pivot is answerable for the pitch and yaw movement. The pitch, or twist, movement is all over, much like opening and shutting a crate top. Yaw moves left and right, similar to an entryway on pivots.

Axis 6

This pivot permits the wrist of the robot arm to tilt here and there. This pivot is answerable for the pitch and yaw movement. The pitch, or twist, movement is all over, much like opening and shutting a crate top. Yaw moves left and right, similar to an entryway on pivots.

Following that, the general robotic arm concept was then molded in Autodesk 3D MAX. Many industries use 3ds Max for generating graphics that are mechanical or even organic in nature. The engineering, manufacturing, educational, and medical industries all make use of 3ds Max for visualization needs as well. 3ds Max utilizes polygon displaying which is a typical method in game plan. With polygonal displaying craftsmen have a serious level of authority over individual polygons which gives them a more noteworthy scope of detail and exactness in their work. After a model is finished, 3ds Max would then be able to be utilized to produce the materials and surfaces important to truly rejuvenate things. Adding surface subtleties, for example, shadings, slopes, and surfaces will prompt greater delivers and game resources..

2.2. Import to EON

The model was then imported to EON Raptor. EON Raptor permits to show and associate 3ds Max content in real time and with natural controls. The completed application models can then be exported and imported to EON for further modification, the models are exported as .EOZ files.

Once the model is imported to EON as shown in Figure 5, it can be seen in EON application tree with the main scene frame, this allows for the

addition and modification of the imported model with the main frame of the application. For the robotic arm, several node interactions are needed to establish the relationship between the imported model parts and define the model relationship with the scene environment. Several node properties are depended upon the robotic arm forward and reverse kinematics calculations, which are solved separately using MATLAB software.

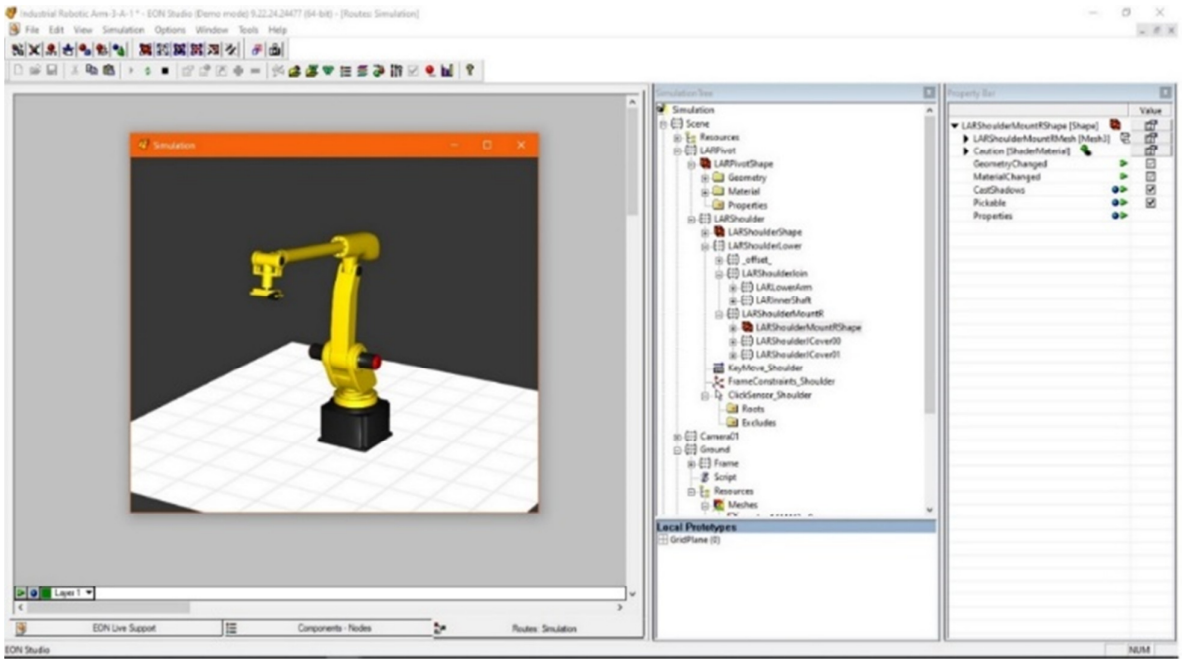

Fig. 5. CAD Model Imported into EON.

2.3. Robot kinematics

2.3.1 Forward Kinematics Modeling

For the robot arm model used in EON application. Forward kinematics is the transformation between joint space and the cartesian space to solve the position and orientation of the robot end effector. Denavit Hartenberg (DH) convention computes the forward kinematic by attaching a coordinate frame system at each joint and specifying the four parameters of DH: αi-1, ai-1, θi and di where:

ai: (link length) is the distance between zi-1 and zi axes along the xi axis.

αi: (link twist) is the required rotation of zi-1 to zi axes about the xi axis.

di: (joint offset) is the distance between xi-1 and xi axes along the zi-1 axis.

θi: (joint angle) is the required rotation of xi-1 to xi axes about the zi-1axis.

The transformation matrix of frame {i} relative to previous frame {i-1} is:

$$\begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad ...(1)$$

And the transformation matrix of nth coordinate frame to base coordinate frame is:

$${}^{0}_{n}T = {}^{0}_{1}T\,{}^{1}_{2}T \ldots {}^{n-1}_{n}T \quad ... (2)$$

The first (3x3) matrix represents rotation matrix of frame {i} relative to frame {i-1} and the fourth column represents the origin of the frame {i} position in frame {i-1}. DH parameters of the robot manipulator are defined according to the assigned frames that shown in Fig.2 and they are listed in Table 1.

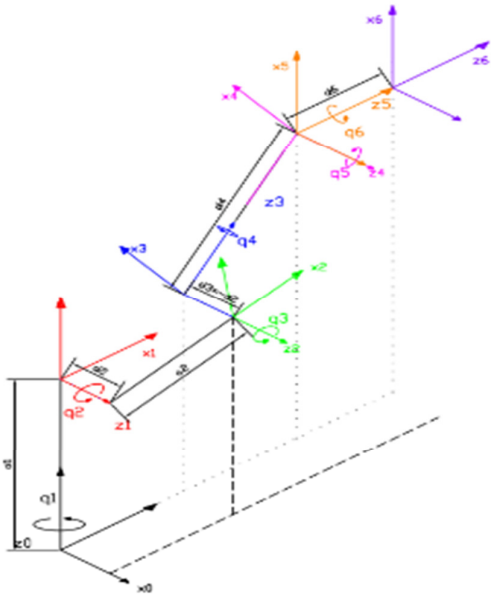

Fig. 6. The Attached Coordinate Frame Systems for EON Robot Model

Table 1,
DH Parameters of the EON Robot Manipulator

i	ai (mm)	αi (deg)	di(mm)	Θi
1	0	90	201	q1
2	390	0	65	q2
3	0	90	-65	q3
4	0	-90	380	q4
5	0	90	0	q5
6	0	0	81	q6

2.3.2 Inverse Kinematics

The reverse kinematics issue of the chronic controllers has been read for a long time. It is required in the control of controllers. Illuminating the opposite kinematics is computationally sweeping and by and large takes quite a while in the continuous control of controllers. Undertakings to be performed by a controller are in the Cartesian space, while actuators work in joint space. Cartesian space incorporates direction network and position vector. Be that as it may, joint space is spoken to by joint points. The change of the position and direction of a controller end-effector from Cartesian space to joint space is called as reverse kinematics issue. There are two arrangements approaches in particular, mathematical and arithmetical utilized for inferring the converse kinematics arrangement, logically.

2.3.3 Kinematic Jacobian

Jacobian gives the relationship between the joint's velocities and the corresponding end effector linear and angular velocities. The end effector linear and angular velocities can be defined as:

$$\begin{bmatrix} v_e \\ \omega_e \end{bmatrix}_{(6x1)} = \begin{bmatrix} J_p \\ J_o \end{bmatrix}_{(6xn)} \begin{Bmatrix} \dot{q} \end{Bmatrix}_{(nx1)} \quad ...(1)$$

Where:
ve : (3×1) matrix represents the end effector linear velocity in cartesian space.
ωe : (3×1) matrix represents the end effector angular velocity in cartesian space.
Jp : (3×n) jacobian matrix relates the end effector linear velocity to joints velocities.
Jo : (3×n) jacobian matrix relates the end effector angular velocity to joints velocities.

The geometry ith column of jacobian matrix for revolute joint is:

$$\begin{bmatrix} J_{p_i} \\ J_{o_i} \end{bmatrix} = \begin{bmatrix} z_{i-1} \times (p_e - p_{i-1}) \\ z_{i-1} \end{bmatrix} \quad ...(2)$$

Where:
zi-1: unit vector in z-direction is given by third column of the rotation matrix ${}^{0}R_{i-1}$.
pe: end effector position vector is given by the first three elements of fourth column of transformation matrix ${}^{0}T_{e}$.
pi-1: is given by the first three elements of the fourth column of transformation matrix ${}^{i-1}T_{i}$.
These calculations helped in the following steps in EON modeling. Where the values calculated used to determine the range of movement for each link and will be used in the nodes used in EON.

2.4 Nodes interaction

Simulations are implicit EON Studio, yet you will require different projects to make and alter 3D items, surfaces, and sounds. Albeit some fundamental item altering is conceivable inside EON Studio (for instance, you can change an article's tones or apply various surfaces), the "crude materials" needed for your reenactments, for example, 3D models, surfaces, and sound records, must be set up ahead of time. Building an EON recreation includes four principle activities:
Adding (bringing in) math to a scene and sorting out the scene chain of importance

Applying materials and improving 3D objects in a scene Characterizing conduct properties for these items Indicating how clients will interface with objects in the recreation Figure (7) shows hubs window. to additional improve the

authenticity of a reproduction scene, you may likewise need to add sound, lighting, and embellishments. The last advance in the process is dispersing the application. Most EON applications are planned to be shown outside of EON Studio. since EON clings to Microsoft's ActiveX standard, you can see full-include EON applications in archives made in ActiveX-agreeable projects

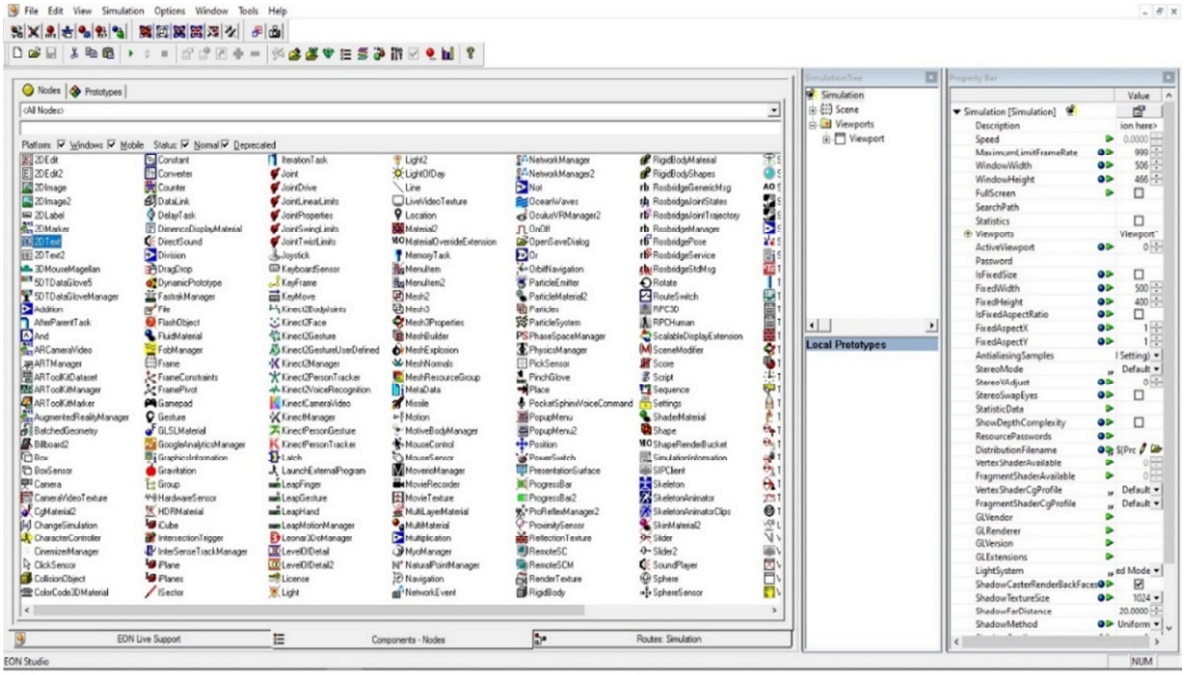

Fig. 7. Nodes window.

Probably the most effortless ways for the client to interface with a recreation is by utilizing the mouse. With essential methods, the simulation can be configured so that users can navigate through the simulation environment and manipulate simulation objects by moving and clicking the mouse. You can also add more complex interactions that will enhance the user's experience with the simulation. In EON, nodes such as Mesh2, Texture2, and MovieTexture use external resources. These nodes have a field named Embedded that indicates whether the resource is to be placed inside or outside the resulting distribution file. The Embedded flag has no effect on project files (.eoz or .eop) because all resources except dynamic prototypes are always embedded in the project files. However, when creating distribution files, the resources that are marked as external (Embedded = False) will be placed outside of the distribution file. These resources will be compiled like the embedded ones, but, as mentioned previously, they will be placed in a subfolder relative to the distribution file. Figure 8 shows the nodes interaction window. An item's situation in 3D space is controlled by its interpretation and revolution esteems. Interpretation alludes to development along a hub a positive or negative way and is communicated in (X, Y, Z) values. On the x axis (right to left}, positive is to the right and negative is to the left. On the y axis, a positive coordinate is into the screen also, a negative facilitate is out of the screen. On the z hub, positive is up and negative is down.

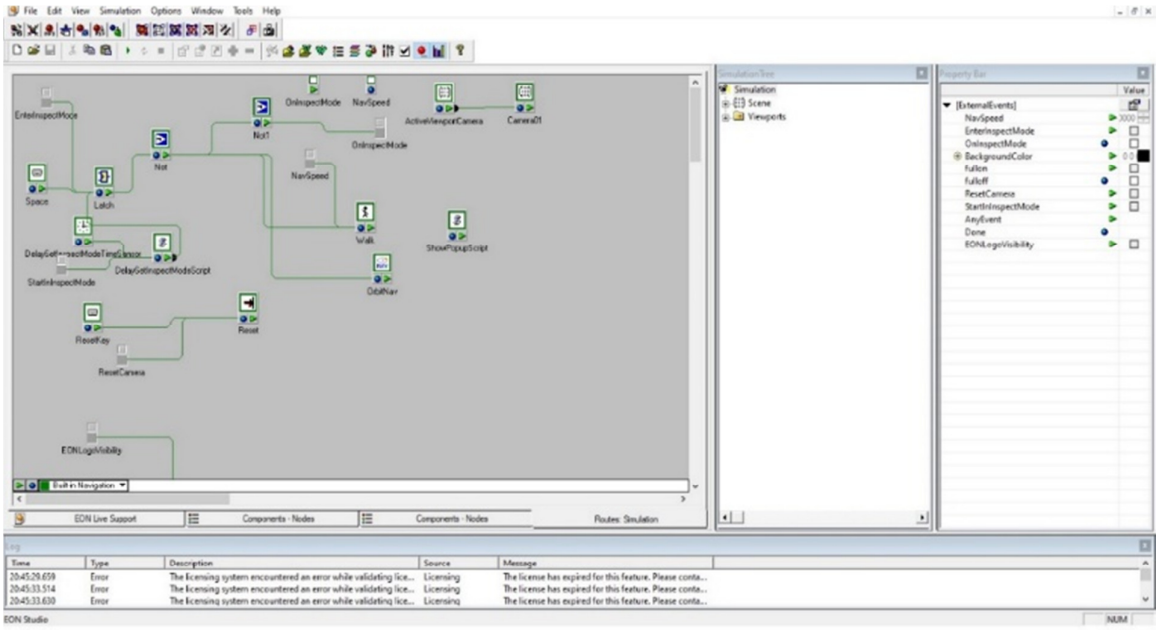

Fig. 8. Nodes interaction.

Rotation refers to rotation around an axis and is communicated in H, P, and R esteems, speaking to heading, pitch, and roll. Heading (H) is the revolution around the z pivot, pitch (P) is the turn around the x hub, and move (R) is the turn around the y hub. Revolution is indicated in degrees inside the 0° to 360° territory. A section of - 90° is likewise substantial and is dealt with equivalent to 270°. In the event that you had the option to remain at the beginning of the facilitate framework and sight along the different interpretation tomahawks, you would experience positive rotation in a counterclockwise direction. Figure 9 shows nodes properties window.

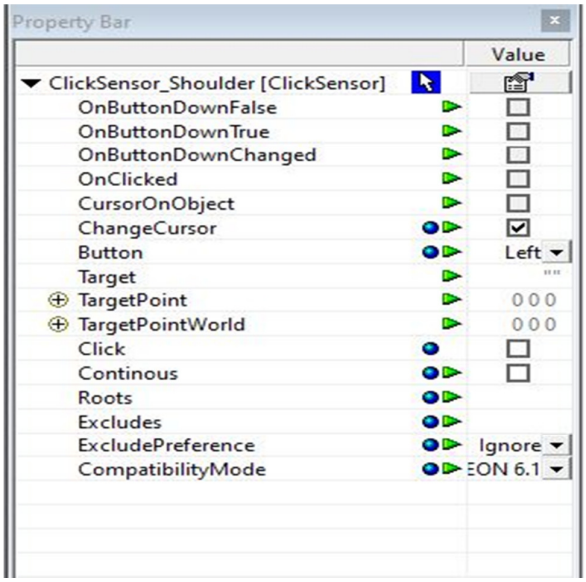

Fig. 9. Nodes properties

The KeyMove node is an extremely useful node for simulation development. It provides an easy way to place an object correctly in the 3D world while the simulation is running. This allows

you to visually position your model exactly where it should be and then determine the exact coordinates for that position-in translation (the Position property in EON) and rotation (the Orientation property in EON). The click sensor allows you to cause action to happen when an object is clicked on by either the right or left mouse button. The following example will demonstrate the use of the click sensor node.
The Rotate hub pivots its parent around one hub (X, Y, or Z) or a mix of these. The beginning for the revolution is the rotate point characterized by parent hub, typically an edge. The parent hub must help turn. Turn is indicated by three qualities (X, Y, or Z). These decide the portion of a 360° turn around the individual pivot to be finished during the LapTime esteem. The LapTime value is specified by the number of seconds it takes to make one rotation.

2.5 Physics interaction

The EON Professional Physics module gives admittance to highlights like gravity, mass, contact, kinematics and actual requirements between objects. The Physics module is based on the effective Vortex material science motor that is a ground-breaking elements motor, in view of principal Newtonian physical science. It additionally incorporates another hearty impact identification calculation with huge speed upgrades just as a high-constancy vehicle elements motor. We will now cover the various physics related nodes in EON Professional and how they are used. The physics manager defines an independent physics world and have properties that affect all physics entities in that world. Physics entities managed by different managers cannot interact with each other. It is best to only have one physics manager per simulation to avoid conflicts. Most of the time you do not have to add a manager since physics entities automatically create and connect to a manager when you add them to the simulation. The manager will be created in Simulation/Scene/Managers.

The Physics Manager node controls some of the high-level physics parameters for the simulation. The main parameter that will be adjusted will be the force of gravity. The RigidBody node defines an object as a rigid body and hold physical properties such as mass and volume. As opposed to deformable objects, they do not change their shape during the simulation. A Rigid body can be static, kinematic or dynamic.

- Dynamic – Is affected by gravity
- Kinematic – can be move other objects.
- Static – Will not move

Rigid body dynamics enable objects to move realistically when influenced by forces, joints or the interaction with other rigid bodies. To define an object as a dynamic rigid body, put a RigidBody node in the frame that represent the object. The geometries in the object frame will be used to generate collision geometry for the rigid body. By default, convex collision geometries will be generated for each geometry. The type of collision geometry to generate can be specified in the GeometryType field. It is important to choose the appropriate geometry type for you object so you do not use. The box type uses a lot less processing power than the triangle mesh. The CharacterController node provides the standard collide and slide character behavior that is often used in games. The CharacterController makes it possible to walk on the ground, climb stairs and small obstacles and slide along walls. The CharacterController geometry approximate a character as an upright cylinder with a rounded top and bottom. The name character controller is a bit misleading; it is not really controlling the character, it detects and handles collisions. When using a CharacterController, a RigidBody or CollisionObject is also needed to define the collision geometry that the character will interact with. A typical use is to make the camera a controlled character, put a CharacterController node in the camera and a CollisionObject node in the frame with the environment geometry and it will be possible to walk around in the environment in a first-person view.

2.6 GUI design

The main focus was to what the VR application users might need to do and ensuring that the interface has elements that are easy to access, understand, and use to facilitate those actions. Users have become familiar with interface elements acting in a certain way. Doing so will help with task completion, efficiency, and satisfaction. There are times when multiple elements might be appropriate for displaying content.

Provide 2D graphical user interface (GUI) elements that overlay the 3D simulation window. They are positioned with 2D pixel screen coordinates. Because they are not part of the 3D space, they do not render inside a viewport. It does not matter what frame they are placed in because they will behave the same regardless of where they are. This lesson will cover the three most common 2D GUI control nodes. The

2DText2 node displays text wherever you want in a box whatever size you want and with whatever colors you want. The text can be updated constantly to show real-time data if you want. Figure 10 shows what the default 2Dtext2 node looks like.

Fig. 10. 2Dtext2 node.

One important thing to note for many of the GUIControl nodes is that there are position/size and position/size offset fields. You specify the position parameter as a percentage of the screen. You specify the Position offset as number of pixels. If you want the node to scale with the window use the position/size fields. If you want the nodes to stay the same size regardless of the size of the simulation window use the offset fields. Don't use both fields at the same time; i.e. if you use position make sure the offset field is "0 0" and if you use the offset field make sure the position is "0 0". It is usually best to use the position/size field instead of the offset fields for the 2Dtext2 node.

3. Simulation

Before the final Simulation. The application structure can be seen in the Tree window, and by expanding the Scene frame, one can notice the order and the structure of the frames and nodes under the Scene frame. Since the model for the robot arm was imported, the Resources group was automatically created with Materials, Textures, and Meshes folders. The user will have the option to use the GUI interface or use EON intuitive controls.

Running the simulation by clicking the simulation button on the menu bar and getting

the simulation window. A Right-click and holding the mouse while dragging upward, this allows the user to zoom out and downward to zoom in. Pressing Ctrl and Shift at the same time (or, alternatively, pressing P). This will display the x, y, and z axes for the simulation. A Left-click and moving the mouse to rotate and pivot around the robotic arm to view it from different angles. Clicking the robotic arm and then use the X, Y, Z, H, P, and R keys with the up and down arrow keys allows for the verification that each part of the arm is moving correctly. The addition of clickseneor and different rotation nodes (Dragging and dropping these nodes under the DOF node located under the selected robotic arm frame) gives the functionality of allowing the user to click on the robotic link on the simulation screen and then use the GUI to perform the required action as shown in figure 11. Following this logic, all the robot links can be actuated to perform user functions.

As compared to the work in [7, 8], the work described here is technically less complicated, in terms of the modules and addons used, programming approach, and ease of application.

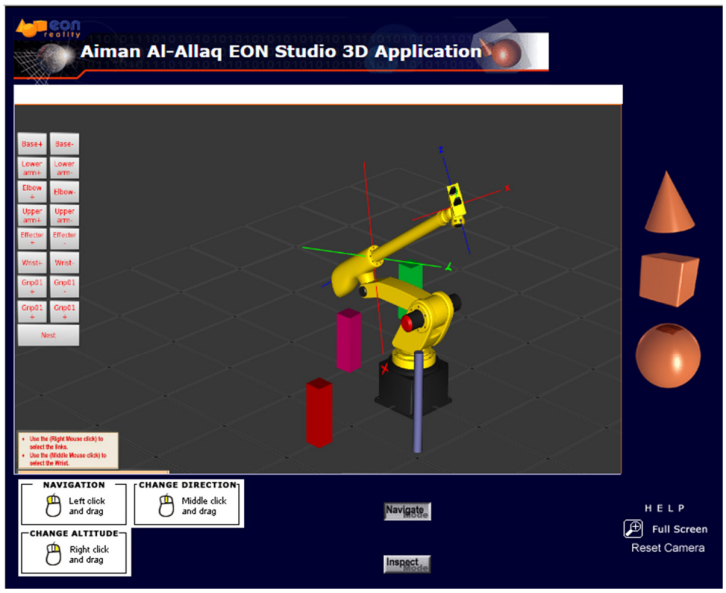

Fig. 11. System application.

4. Conclusions

A virtual reality of 6-axes robotic has been implemented. The presented system is fully configurable, expandable and immersive. Based on the obtained results and tests, it can be determined that the application has a high potential to become a very reliable engineering learning tool. The system is fully functional and it is presented in a stage from which it can be modified. Learning users can make use of the virtual reality system and interact with it, chances to make mistakes are allowed, since simulation prevents user or equipment damage.

5. References

[1] L. Pérez, E. Diez, R. Usamentiaga, and D. F. García, "Industrial robot control and operator training using virtual reality interfaces," Computers in Industry, vol. 109, pp. 114-120, 2019/08/01/ 2019.

[2] M. C. C. d. Santos, V. A. Sangalli, and M. S. Pinho, "Evaluating the Use of Virtual Reality on Professional Robotics Education," in 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC), 2017, vol. 1, pp. 448-455.

[3] C. V. Hurtado, A. R. Valerio, and L. R. Sánchez, "Virtual Reality Robotics System for Education and Training," in 2010 IEEE Electronics, Robotics and Automotive Mechanics Conference, 2010, pp. 162-167.

[4] F. Wang, "Research on Virtual Reality Based on EON Studio," in 2010 Fourth International Conference on Genetic and Evolutionary Computing, 2010, pp. 558-561.

[5] J. He and W. Hou, "Key Technologies of the Virtual Driving Platform Based on EON," in 2011 International Conference on Virtual Reality and Visualization, 2011, pp. 263-266.

[6] R. Crespo, R. García, and S. Quiroz, "Virtual reality simulator for robotics learning," in 2015 International Conference on Interactive Collaborative and Blended Learning (ICBL), 2015, pp. 61-65.

[7] J. Feuser, O. Ivlev, and A. Graser, "Collision prevention for rehabilitation robots with mapped virtual reality," in 9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005., 2005, pp. 461-464.

[8] B. Volmer, A. Verhulst, A. Drogemuller, B. H. Thomas, M. Inami, and M. Sugimoto, "Towards Robot Arm Training in Virtual Reality Using Partial Least Squares Regression," in 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), 2019, pp. 1209-1210.

تطبيق الواقع الافتراضي على (هندسة الميكاترونكس) عن طريق نمذجة ذراع روبوت باستخدام منصة ايون (EON Reality V9.22.24.24477)

ايمن العلاق* نيبوسا جاكسك
حسين العاملي*** ضحى محمد محمود ****
*، ** جامعة كولورادو الحكومية بويبلو/ الولايات المتحدة الامريكية
،* كلية الهندسة الخوارزمي/ جامعة بغداد/ العراق
*البريد الالكتروني: ah.alallaq@pack.csupueblo.com
**البريد الالكتروني: n.jaksic@csupueblo.com
***البريد الالكتروني: alamilihussein@gmail.com
**** البريد الالكتروني: duhamohammed55@gmail.com

الخلاصة

يقدم الواقع الافتراضي، VR، العديد من الفوائد للتعليم التقني ، بما في ذلك توصيل المعلومات من خلال قنوات متعددة، ومعالجة أنماط التعلم المختلفة، والتعلم القائم على التجربة. يعرض هذا البحث العمل الذي قام به المؤلفون لتطبيق الواقع الافتراضي على التعليم الهندسي، في ثلاثة مجالات واسعة: التعلم الآلي الافتراضي، ومختبر الميكاترونيك الافتراضي، ومنصة التصنيع الافتراضية. يوفر المجال الأول استكشافًا موجهًا للمجالات التي يتعذر الوصول إليها بطريقة أخرى مثل مكونات الخلايا الروبوتية والحركية الروبوتية وظروف العمل. والثاني يعزز تعلم الميكاترونكس وتوجيه مهندسي الميكاترونكس الجدد عند التعامل مع الروبوتات بطريقة آمنة وتفاعلية. والثالث يوفر إرشادات قيمة للصناعة والتصنيع القائم على الروبوت، مما يسمح برؤية أفضل ومحاكاة الظروف التي يتعذر الوصول إليها.