A PROJECT REPORT

on

DEVELOPMENT OF VIRTUAL ENVIRONMENT FOR AN ASSEMBLY OF MACHINE COMPONENTS

Submitted in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Mechanical Engineering

of

SREE VIDYANIKETHAN ENGINEERING COLLEGE, TIRUPATI

by

A. ARAVIND **17125A0302**

Under the esteemed guidance of

Dr. K. L. NARASIMHAMU

Professor

Department of Mechanical Engineering

DEPARTMENT OF MECHANICAL ENGINEERING

SREE VIDYANIKETHAN ENGINEERNIG COLLEGE

(Affiliated to J. N. T. University Anantapur, Anantapuramu)

Sree Sainath Nagar, A.Rangampet, Tiruapti-517102, Andhra Pradesh

2016-2020

SREE VIDYANIKETHAN ENGINEERING COLLEGE

(Autonomous)

A.Rangampet, Tirupati-517102

DEPARTMENT OF MECHANICAL ENGINEERING

CERTIFICATE

This is to certify that the project titled " **DEVELOPMENT OF VIRTUAL ENVIRONMEMNT FOR AN ASSEMBLY OF MACHINE COMPONENTS** "submitted by the following students in the Department of Mechanical Engineering, Sree Vidyanikethan Engineering College (Autonomous), A.Rangampet, Tirupati and is submitted in partial fulfillment for the award of B.Tech. in Mechanical Engineering to the Jawaharlal Nehru Technological University Anantapur, Anantapuramu is a record of bonafide work carried out by them under our guidance and supervision.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

S MOHITH REDDY	**16121A03D7**
Y VIJAY SAI NATH	**16121A03F8**
A ARAVIND	**17125A0302**
B SAI LOHITH	**17125A0305**
D VASIF HUSSAIN	**17125A0314**
G JAYA PRAKASH REDDY	**17125A0317**

GUIDE **HEAD OF THE DEPARTMENT**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep sense of gratitude, appreciation and indebtedness to our guide Dr.K.L.NARASIMHAMU for his valuable guidance. His sincere interest, support and constant encouragement during the project work have helped us in completing the thesis.

We wish to express our deepest gratitude and thanks to Dr.K.C.VARAPRASAD Professor & Head, Department of Mechanical Engineering, SVEC.

We express our profound gratitude to the Principal and Management of SVEC for permitting us to do our project work.

Last but not least, thanks to our parents for their numerous support in many different ways throughout the study.

We also thank all who have helped directly or indirectly in completing the project work.

Students Name with Signatures

A.ARAVIND

SREE VIDYANIKETHAN ENGINEERING COLLEGE

VISION

To be one of the Nation’s premier Engineering Colleges by achieving the highest order of excellence in Teaching and Research.

MISSION

- To foster intellectual curiosity, pursuit and dissemination of knowledge.
- To explore students’ potential through academic freedom and integrity.
- To promote technical mastery and nurture skilled professionals to face competition in ever increasing complex world.

DEPARTMENT OF MECHANICAL ENGINEERING

VISION

To become and be recognized as a nationwide center of excellence in Mechanical Engineering and allied areas for acquiring self-reliance through education, engagement and research.

MISSION

- Department of Mechanical Engineering is established to provide students with a sound Mechanical Engineering education, advance the understanding and application of Mechanical Engineering principles to work in multicultural and multidisciplinary environment.
- Engage and impart knowledge to the students for innovative, high-impact and leading edge research and development of modern Mechanical Engineering science through contemporary curriculum.
- Maintain a collegial, supportive, and diverse environment that encourages students, faculty, and staff to achieve to the best of their abilities.
- Serve our students by teaching them problem solving, leadership and teamwork skills, and the value of a commitment, quality and ethical behavior for their employability.
- Serve the community and industry through proactive knowledge exchange.

PROGRAM EDUCATIONAL OBJECTIVES

Within few years of graduation, B. Tech. (ME) Program, graduates would have:

1. Higher education in mechanical engineering, business administration, or other disciplines.
2. Career in mechanical engineering and allied industry, software industry, or managerial positions, and ability to start entrepreneurial ventures related to Mechanical Engineering.
3. Ability to recognize the importance of, and engage in life-long learning through self-study for solving problems related to Mechanical Engineering.

PROGRAM OUTCOMES

On successful completion of the Program, the graduates of B. Tech. (ME) Program will be able to:

1. Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems. (Engineering knowledge)
2. Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences. (Problem analysis)
3. Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations. (Design/development of solutions)

4. Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the (Conduct investigations of complex problems) information to provide valid conclusions.
5. Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations. (Modern tool usage)
6. Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice. (The engineer and society)
7. Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development. (Environment and sustainability)
8. Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice. (Ethics)
9. Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings. (Individual and team work)
10. Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions. (Communication)
11. Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and

leader in a team, to manage projects and in multidisciplinary environments. (Project management and finance)

12. Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change. (Life-long learning)

PROGRAM SPECIFIC OUTCOMES

On successful completion of B. Tech. (ME) Program, graduates will be able to:

1. Apply principles and concepts pertaining to Fundamental sciences, Humanities and Mechanical Engineering in solving problems of practical interest in sub domains of Manufacturing, Thermal, Design and Management Sciences.
2. Analyze problems of practical importance in research and industry by integrating engineering sciences.
3. Design mechanical components, systems or processes to meet desired functionality with realistic constraints.
4. Conduct investigations on complex engineering problems to obtain plausible solutions in the sub domains.
5. Apply appropriate analytical, experimental and computational tools and resources by integrating multifaceted requirements from various sub domains.

IV B. Tech – II Semester

(16BT80331) PROJECT WORK

Int. Marks	Ext. Marks	Total Marks	L T P C
100	100	200	- - - 12

PRE-REQUISITES: All the courses of the program.

COURSE DESCRIPTION: Identification of topic for the project work; Literature survey; Collection of preliminary data; Identification of implementation tools and methodologies; Performing critical study and analysis of the topic identified; Time and cost analysis; Implementation of the project work; Preparation of thesis and presentation.

COURSE OUTCOMES: Completion of the project work enables a successful student to demonstrate:

CO1. Knowledge on the project topic.

CO2. Analytical ability exercised in the project work.

CO3. Design skills applied on the project topic.

CO4. Ability to investigate and solve complex engineering problems faced during the project work.

CO5. Ability to apply tools and techniques to complex engineering activities with an understanding of limitations in the project work.

CO6. Ability to provide solutions as per societal needs with consideration to health, safety, legal and cultural issues considered in the project work.

CO7. Understanding of the impact of the professional engineering solutions in environmental context and need for sustainable development experienced during the project work.

CO8. Ability to apply ethics and norms of the engineering practice as applied in the project work.

CO9. Ability to function effectively as an individual as experienced during the project work.

CO10. Ability to present views cogently and precisely on the project work.

CO11. Project management skills as applied in the project work.

CO12. Ability to engage in life-long leaning as experience during the project work.

ABSTRACT

Virtual Reality (VR) is a high-end user computer interface that involves real time simulation and interaction through multiple sensorial channels. From the perspective of technology, near-eye display, rendering technology, perception and interaction, network transmission, and content creation are forming the primary system. In recent years, the level of realism in virtual environments (VE) has increased dramatically, from 2D drawings to highly detailed 3D visualizations. In addition, VE have also been developed as a potential solution for teaching practical and technical skills. This interface enables to train the users and improve their skills. The engine assembly is designed in CatiaV5 and virtual environment is developed by Unity. The interaction is made by using Oculus Go. This project is inclined to experience richer and more engaging learning experiences that can be provided through books, websites, or even videos. It immerses the user in a 3d interactive experience that allows interactive learning in a controlled and safe environment.

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

VR	Virtual Reality
HMD	Head Mounted Display
CAI	Computer Assisted Instruction
CBT	Computer Based Training
CAD	Computer Aided Drawing
CAM	Computer Aided Manufacturing
CAE	Computer Aided Engineering
STEP	Stand for the Exchange of Product model data
IGES	Initial Graphics Exchange Specification
FBX	Film Box
OBJ	3D Object
2D	Two Dimensional
3D	Three Dimensional
BMP	Bitmap
TIF	Tagged Image File
JPG	Joint Photographic Experts
PSD	Photoshop Document
IDE	Integrated Development Environment
USB	Universal Serial Bus
WI-FI	Wireless Fidelity
PC	Personal Computer
GB	Gigabyte
LPDDR	Low Power Double Data Rate

GPU	Graphics Processing Unit
GFLOPS	Gigaflops
API	Application Program Interface
XR	Extended Reality
SDK	Software development kit
JT	Jupiter Tessellation

CONTENTS

1. INTRODUCTION

Human history is marked by a progression of media used to convey and experience ideas. Perhaps the most recent step is the use of virtual reality. The term "virtual" has been used in the computer sense of "not physically existing but made to appear by software".

The term "virtual reality" was initially coined by Jaron Lanier, founder of VPL Research.

Virtual Reality is a term used to describe a computer-generated virtual environment that may be moved through and manipulated by a user in real-time. A virtual environment may be displayed on a head-mounted display, a computer monitor, or a large projection screen. Head and hand tracking systems are employed to enable the user to observe, move around, and manipulate the virtual environment.

A virtual world is a real representation of some world that may or may not exist in the physical world. Virtual reality (VR) provides the ultimate level of immersion, creating a sense of physical presence in real or imagined worlds. VR accomplishes this by focusing on the three pillars of immersion-visual quality, sound quality, and intuitive interactions-to simulate our human senses with realistic feedback.

VR technology can be characterized by the following main aspects: Firstly, VR stands for a realistic rendering of the product appearance (material, surface, colors) and behavior. Secondly, VR makes use of advanced display technologies that allow the engineers to experience the virtual prototype like a real one.

1.1 FOUR KEY ELEMENTS OF VIRTUAL REALITY EXPERIENCE

The key elements in experiencing virtual reality are a virtual world, immersion, sensory feedback, and interactivity.

Key Element 1: Virtual World

A virtual world is the content of the given medium. It is the description of objects within a simulation.

Key Element 2: Immersion

Imagination is where virtual worlds begin and how numerous virtual worlds are experienced. In virtual reality, the effect of entering the world begins with immersion.

Key Element 3: Sensory Feedback

Sensory feedback is an ingredient essential of virtual reality. The VR system provides direct sensory feedback to the participants based on their physical position. VR allows the participant to select their vantage point by positioning their body and to affect events in the virtual world. These features may help to make reality more compelling.

Key Element 4: Interactivity

Immersion within a virtual environment is one thing, but for a user to feel truly involved there must also be an element of **interaction**. Another necessary component in the full definition of virtual reality is interactivity. The player can interact with objects, characters, and places in the virtual world. Each world responds to commands typed by the player, giving the player the sense of being involved with these worlds.

1.2 THE THREE I'S OF VIRTUAL REALITY:

It is clear from the foregoing description that virtual reality is both interactive and immersive. these features are the two I's that most people are familiar with. There is, however, a third feature of virtual reality that fewer people are aware of. Virtual reality is not just a medium or a high-end user interface, it also has applications that involve solutions to real problems in engineering, medicine, the military, etc. These applications are designed by virtual reality developers. The extent to which an application can solve a particular problem, that is, the extent to which a simulation performs well, depends therefore very much on the human imagination, the third "I" of VR. Virtual reality is, therefore, an integrated trio of immersion-interaction-imagination, as shown in the below diagram.

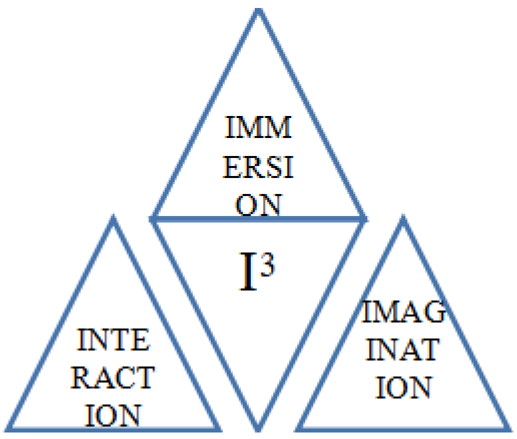


FIG: 1.1 THREE I’S OF VR

The imagination part of VR refers also to the mind's capacity to perceive non-existent things. The triangle is, for example, is easily "seen" by the reader, yet it only exists in his or her imagination.

1.3 THE FIVE CLASSIC COMPONENTS OF VR SYSTEM :

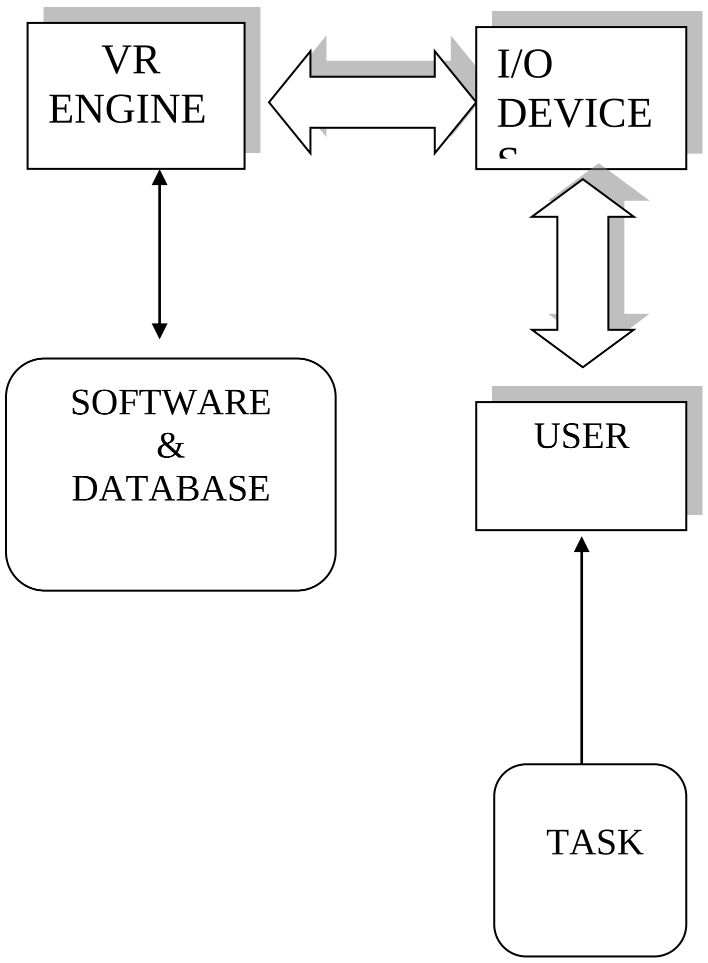


FIG: 1.2 VR SYSTEM ARCHITECTURE

1.4 VR IS COMMUNICATION

Normally, communication is thought of as interaction between two or more people. Communication can also be between humans and technology—an essential component and basis of VR. VR design is concerned with the communication of how the virtual world works, how that world and its objects are controlled, and the relationship between user and content: ideally where users are focused on the experience rather than the technology.

Well-designed VR experiences can be thought of as collaboration between human and machine where both software and hardware work harmoniously together to provide intuitive communication with the human.

Communication between the human and system is achieved via hardware devices. These devices serve as input and/or output. A transfer function, as it relates to interaction, is a conversion from human output to digital input or from digital output to human input. Input is considered information traveling from the user into the system and output is the feedback that goes from the system back to the user. This forms a cycle of input/output that continuously occurs for as long as the VR experience lasts.

Input collects data from the user such as where the user's eyes are located, where the hands are located, button presses, etc. The application includes non-rendering aspects of the virtual world including updating dynamic geometry, user interaction, physics simulation, etc. Rendering is the transformation of a computer-friendly format to a user-friendly format that gives the illusion of some form of reality and includes visual rendering, auditory rendering (called Auralization), and haptic (the sense of touch) rendering. The output is the physical representation directly perceived by the user (e.g., a display with pixels or headphones with sound waves).

True virtual environments are artificially created without capturing any content from the real world. The goal of virtual environments is to completely engage a user in an experience, present in another world such that the real world is temporarily forgotten while minimizing any adverse effects.

1.5 APPLICATION OF VIRTUAL REALITY:

VR can simulate real workspaces for workplace occupational safety and health purposes, educational purposes, and training purposes. It can be used to provide learners with a virtual environment where they can develop their skills without the real-world consequences of failing. It has been used and studied in primary education anatomy teaching, military, astronaut training, flight simulators, miner training, architectural design, driver training, and bridge inspection. Immersive VR engineering systems enable engineers to see virtual prototypes before the availability of any physical prototypes. Supplementing training with virtual training environments has been claimed to offer avenues of realism in military and healthcare[training while minimizing cost. It also has been claimed to reduce military training costs by minimizing the amounts of ammunition expended during training periods.

2. LITERATURE REVIEW

Recorded history begins with people paintings on cave walls. These paintings were a primitive medium for conveying the artist's concept.

It all started in 1989 when Jaron Lanier, founder of VPL Co. came up with the phrase "Virtual Reality", which became the title for this scientific technology field [1].

In the 19th Century, the first 360-degree art through panoramic murals began to appear. Nearly one hundred years later, a mechanical device, the Sensorama, engaged multiple senses to create an immersive VR. The system provided a multisensory experience of riding a motorcycle, including three-dimensional, full-color film together with sounds, smells, and the feeling of motion, as well [2].

The virtual environment will be characterized by the immersion of learners in highly communicative, multi-sensory, interactive, involving simultaneous visual, auditory, and haptic feedback [3].

In the past, there were major difficulties in using HMDs or similar technologies. They cause a feeling of aversion to their users due to the mismatch between the movements of the head and the corresponding change in the scene [4].

There are Cave Automatic Virtual Environments (CAVE), where the user is in a room where all the walls, as well as the floor, are projection screens (or flat displays). The user, who can wear 3D glasses, feels floating in the projected world where he can move around freely. CAVE environments are still rather expensive, they need to have a specific space dedicated to them and they cannot be moved easily. All these characteristics make it difficult for them to have widespread use in education and training [5].

Duncan et al. described benefits that virtual worlds can have in teaching and learning, as they can provide a laboratory for collaborative work, socialization, and entertainment [6].

Real-time interactivity is a critical feature of VR which allows learners to more effectively engage with the learning system. The learner can directly interact with virtual objects, test their ideas, and observe the results in real-time. Reworking a physical model can take weeks, but reworking a virtual prototype can take only minutes.[7]

Veronica S. Pantelidis(1993), said that the use of virtual reality (VR) in education can be considered as one of the natural evolutions of computer-assisted instruction (CAI) or computer-based training (CBT). At every level of education, virtual reality has the potential to make a difference, to lead learners to discoveries, to motivate and encourage and excite. The learner can participate in the learning environment with a sense of presence, of being part of the environment [8].

Michael Rorke (1997), has explained that Immersive virtual reality (VR) places the user inside of the computing environment, blurring the distinction between the environment itself and the user interface to access that environment non-obvious. This lack of distinction between environment and interface makes it difficult to place menus and other interface elements where the user is both able to access them easily and where they do not obscure large parts of the user's field of view [9].

Sandra Dutra Piovesan et al. (2000), the author explains the virtual reality is being more and more used in the education, enabling the student to find out, to explore and to build his knowledge. The results reached with the use of the developed software show the attributes that make the ideal Virtual Reality for situations of research and learning taking
the discipline as a reference of the classroom to the computer labs and making more interesting to the student, making the learning easy [10].

Michael Rorke et al. (2000), has suggested that the Virtual reality is no longer hampered by the absence of appropriate hardware, but rather the absence of understanding about the medium and how to deal with its shortcomings. The sensory input missing from current virtual reality systems (e.g. the lack of haptic feedback) must be compensated for, to make these systems more accessible and usable to the general public [11].

Stacy Kluge (2002), has been described the virtual worlds provide a mechanism to incorporate constructivist, experiential, and student-centered learning practices into the classroom. The authors also discuss the challenges and the benefits of using virtual worlds in education as well as some implications for the future of education [12].

Kami Hanson et al. (2008), have explained that there exists an increasingly attractive lure of using virtual reality applications for teaching in all areas education, but perhaps the largest detriment to its use is the intimidating nature of VR technology for nontechnical instructors. This addresses the issues regarding identifying the appropriate techniques for integrating VR into traditional instructional design, and the considerations for development for non-technical educators [13].

Chwen Jen Chen (2009), have suggested that the VRID an instructional design and development model guides the design and development of educational virtual environments. Research should focus on identifying the advantages of virtual reality methods, devising innovative methods that employ the unique features of this technology, and figuring out the approaches to implement this technology that can help to improve the quality of education as well as to direct the proper use of virtual reality [14].

Enrico Gobbetti et al. (2010), have explained that more and more research has demonstrated its usefulness both from the evolutionary perspective of providing a better user interface and from the revolutionary perspective of enabling previously impossible applications. Examples of application areas that have benefited from VR technology are virtual prototyping, simulation, and training, telepresence, and teleportation, and augmented reality. Virtual reality has thus finally begun to shift away from the purely theoretical and towards the practical. The complexity of VR, the importance of human factors, and the lack of standard solutions, the secret of successfully implementing professional VR applications is to set realistic expectations for the technology [15].

B**obbie Ticknor et al. (2011)**, have suggested the term virtual reality (VR) is believed to have originated in 1938 with author Antonin Artaud, who referred to the theatre as “la realite virtuelle” (Davis, 1998; Zorn, 2010). Although used in this sense to describe more abstract concepts, visionaries were already beginning to create devices to mimic reality. In 1929, inventor Edward Link created the first flight simulator [16].

Chris Cocking (2011), has proposed that the use of VLEs in education is increasing exponentially and this is opening up exciting new opportunities, as learners interact with each other more and more in virtual worlds. This may even result in changing the very way in which we view education. "The definition of learning as information regurgitation is giving way to a notion of learning as centering upon immersive learning experiences that are inherently social and collaborative" [17].

E. KirubaNesamalar et al. (2012), has described as Virtual Reality is a computer system used to create an artificial world in which the user has the impression of being in that world and with the ability to navigate through the world and manipulate objects in the world. Most current virtual reality environments are primarily visual experiences, displayed either on a computer screen or through special stereoscopic displays, but some simulations include additional sensory information, such as sound through speakers or headphones. Some advanced, haptic systems now include tactile information, generally known as force feedback, in medical and gaming applications. Virtual reality is used to describe a wide variety of applications commonly associated with immersive, highly visual, 3D environments [18].

Jinzhou Wang (2012), has explained Virtual reality technology is becoming perfecter and perfecter with the aid of computer hardware, software, and virtual world integration technology, which can simulate the real world dynamically. The dynamic circumstance can make reactions according to people's form, language, and so on immediately, by which real-time communication is formed between people and the virtual world [19].

Nooriafsharet al. (2014), has proposed the importance and effectiveness of visual features in teaching and learning materials. virtual reality multimedia could enhance learning by providing much more realistic images and visual features [20].

Chris Christou (2015), has explained that A key feature of VR is that it allows multi-sensory interaction with space visualized. This combination of multi-sensory visualization and interactivity make VR ideally suited for effective learning [21].

Xueqin Chang et al. (2016), have been described that Virtual reality technology as a new kind of remote education media technology, with its powerful teaching advantages and potential, is bound to vigorously promote the informatization development process of distance education [22].

Jorge Martín-Gutiérrez et al. (2016), have represented that, Virtual reality captures people's attention. This technology has been applied in many sectors such as medicine, industry, education, video games, or tourism. Educational institutions will benefit from better accessibility to virtual technologies; this will make it possible to teach in virtual environments that are impossible to visualize in physical classrooms, like accessing virtual laboratories, visualizing machines, industrial plants, or even medical scenarios. The huge possibilities of accessible virtual technologies will make it possible to break the boundaries of formal education [23].

A. Pantelić, et al. (2017), explain the purpose of using VR and AR technologies in education is to achieve better learning effects than by using traditional means of teaching and learning. Recognition will be better if the image has a tonal variation and contrast, if it's large enough and printed on matte finished paper instead of glossy paper which might reflect the light to the camera [24].

Having a VR simulation is always something that is very cool and motivates students to give it their best. As well, VR training is more economical, ecological, and efficient [25].

3. EXPERIMENTAL INVESTIGATION

3.1 DESIGN- CATIA V5

CATIA is the World's Leading Solution for Product Design and Experience. It is used by leading organizations in multiple industries to develop the products. It is a multi-platform software suite for computer-aided design (CAD), computer-aided manufacturing (CAM), computer-aided engineering (CAE), and 3D, developed by the French company Dassault Systems. It delivers the unique ability not only to model any product but to do so in the context of its real-life behavior: design in the age of experience.

Scope of application

Commonly referred to as a 3D Product Life cycle Management software suite, CATIA supports multiple stages of product development, including conceptualization, design (CAD), engineering (CAE) and manufacturing (CAM). CATIA facilitates collaborative engineering across disciplines around its 3DEXPERIENCE platform, including surfacing & shape design, electrical, fluid and electronic systems design, mechanical engineering, and systems engineering.

Mechanical engineering

CATIA enables the creation of 3D parts, from 2D sketches, sheet metal, composites, molded, forged, or tooling parts up to the definition of mechanical assemblies. CATIA supports multiple stages of product design whether it started from scratch or 2D sketches(blueprints).

Automotive

Many automotive companies use CATIA for car structures – door beams, bumper beams, roof rails, side rails, body components because of CATIA's capabilities in Computer representation of surfaces.

CATIA USER INTERFACE

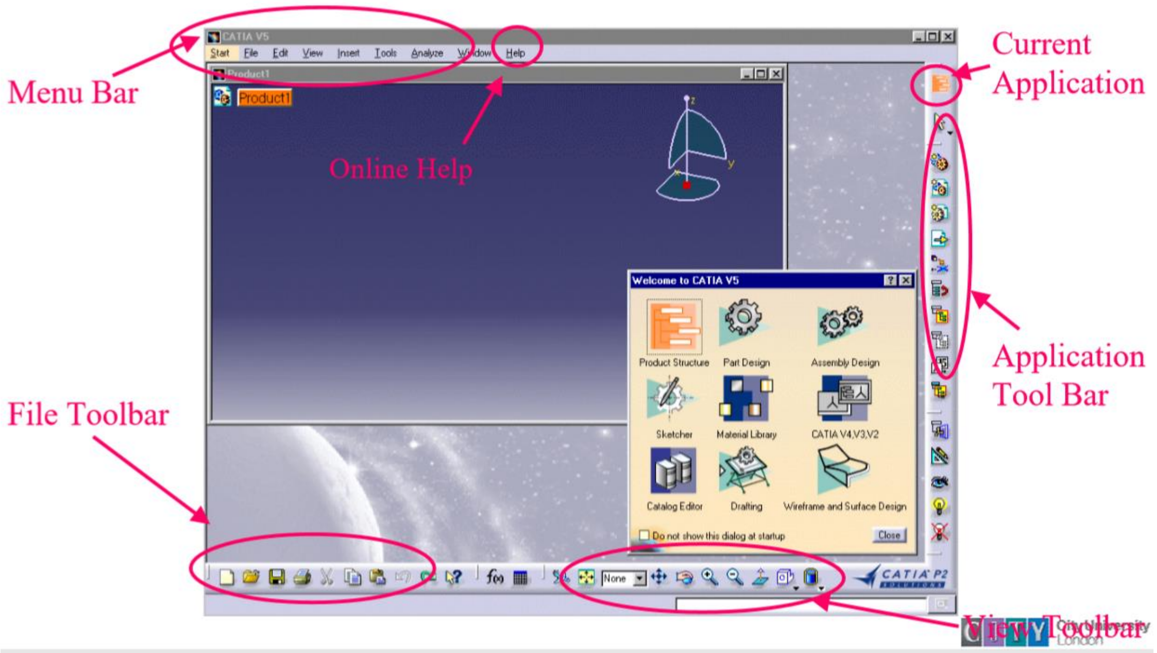


FIG: 3.1 GRAPHICAL USER INTERFACE

SPECIFICATION TREE

- The Specification Tree is displayed on the left side of the screen while you are working
- Provides access to the history of how a part was constructed, and shows the product structure
- Product entities can be selected from the spec. tree or in the geometry area
- Parts can be modified by selecting them from the spec. tree.
- Click on + to open a tree branch
- Solid Parts are stored in the PartBody branch of the Part tree

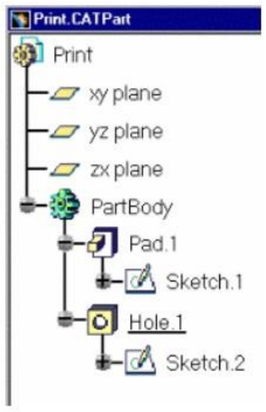
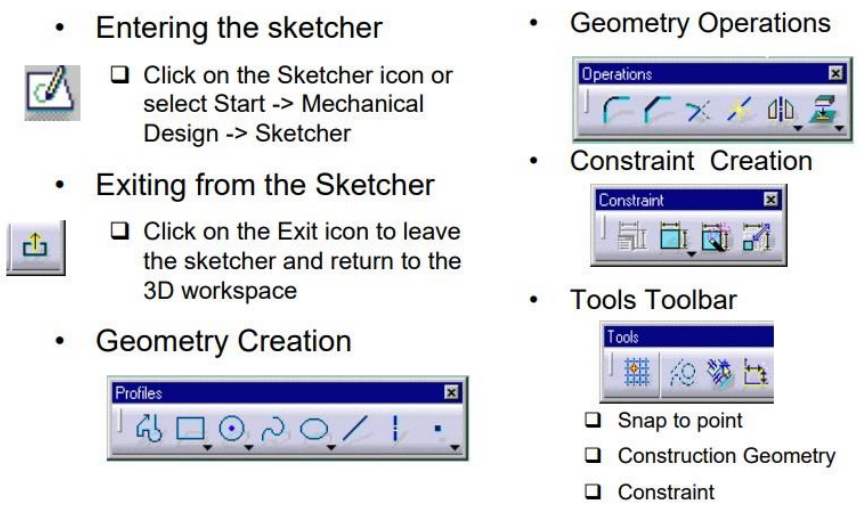


FIG: 3.2 SPECIFICATION TREE

Step 1: Sketcher Module

The Sketcher is a parametric design tool, to draw the approximate shape of a design, and then assign constraints to complete the shape definition. Solids can only be created from sketches that form a single boundary.

The Sketcher workbench contains the following standard workbench specific toolbars.



- Entering the sketcher
  - ❑ Click on the Sketcher icon or select Start -> Mechanical Design -> Sketcher
- Exiting from the Sketcher
  - ❑ Click on the Exit icon to leave the sketcher and return to the 3D workspace
- Geometry Creation
- Geometry Operations
- Constraint Creation
- Tools Toolbar
  - ❑ Snap to point
  - ❑ Construction Geometry
  - ❑ Constraint

FIG: 3.3 SKETCHER WORKBENCH SPECIFIC TOOLBARS

• **Profile toolba**r: The commands located in this toolbar allow you to create simple geometries (rectangle, circle, line, etc...) and more complex geometries (profile, spline, etc...).

• **Operation toolbar:** Once a profile has been created, it can be modified using commands such as trim, mirror, chamfer, and other commands located in Operation toolbar.

• **Constraint toolbar:** Profiles may be constrained with dimensional (distances, angles, etc...) or geometrical (tangent, parallel, etc...) constraints using the commands located in the Constraint toolbar.

• **Sketch tools toolbar:** The commands in this toolbar allow you to work in different modes which make sketching easier.

Step 3: Part Design Module

The part design application is used to create 3D models from the basic 2D sketches created in the sketcher tool. Parts are stored in files with the extension.CATPART

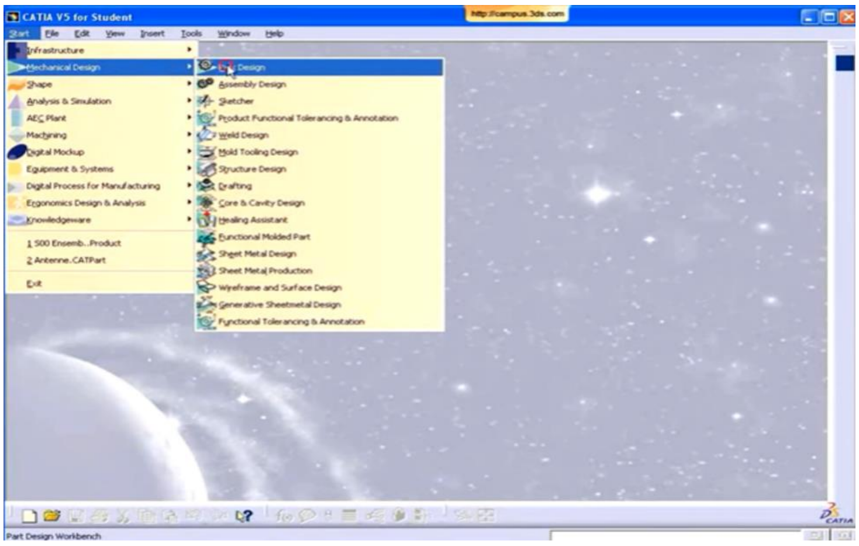
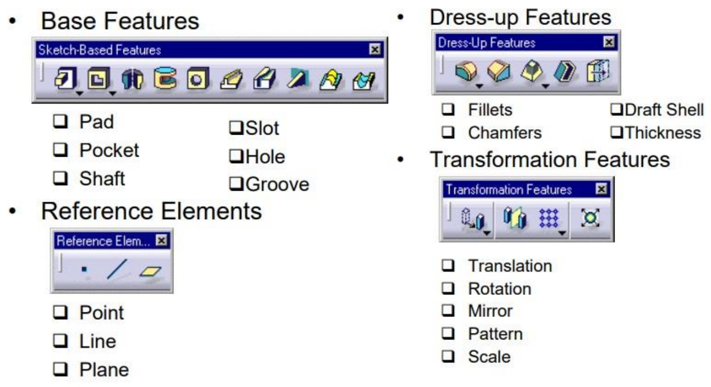


FIG: 3.4 PART DESIGN MODULE WINDOW

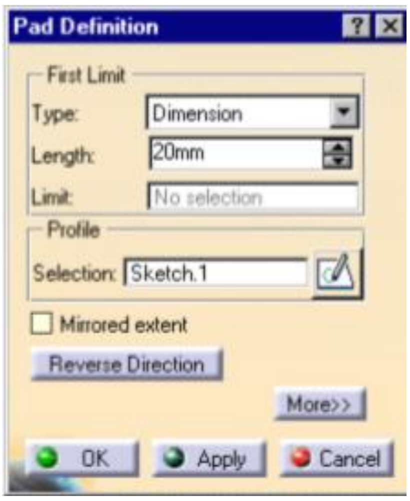


- Base Features
  - ❑ Pad
  - ❑ Pocket
  - ❑ Shaft
  - ❑Slot
  - ❑Hole
  - ❑Groove
- Reference Elements
  - ❑ Point
  - ❑ Line
  - ❑ Plane
- Dress-up Features
  - ❑ Fillets
  - ❑ Chamfers
  - ❑Draft Shell
  - ❑Thickness
- Transformation Features
  - ❑ Translation
  - ❑ Rotation
  - ❑ Mirror
  - ❑ Pattern
  - ❑ Scale

FIG: 3.5 FEATURES OF PART DESIGN

Creating a solid part from a sketcher

1. Click on the pad icon to create an extruded part.
2. Select the sketch containing the profile you want to extrude.
3. The pad definition window will appear


Pad Definition

First Limit

Type: Dimension

Length: 20mm

Limit: No selection

Profile

Selection: Sketch.1

☐ Mirrored extent

FIG: 3.6 PAD DEFINATION

4. Select the limit type from:
    - Dimension
    - Up To Next
    - Up To Last
    - Up To Plane
5. Type in the length if required.
6. Check the extrude direction arrow.
7. Click on OK to create the part.

Step 4: Assembly Module

Assembly design application allows creating a product model from several separate parts. The parts in a product assembly are not joined together but assembled as they would be in a physical assembly. The product assembly structure is hierarchical and allows us to model complex product relationships. Constraints can be applied between the parts in an assembly to define relationships between them.

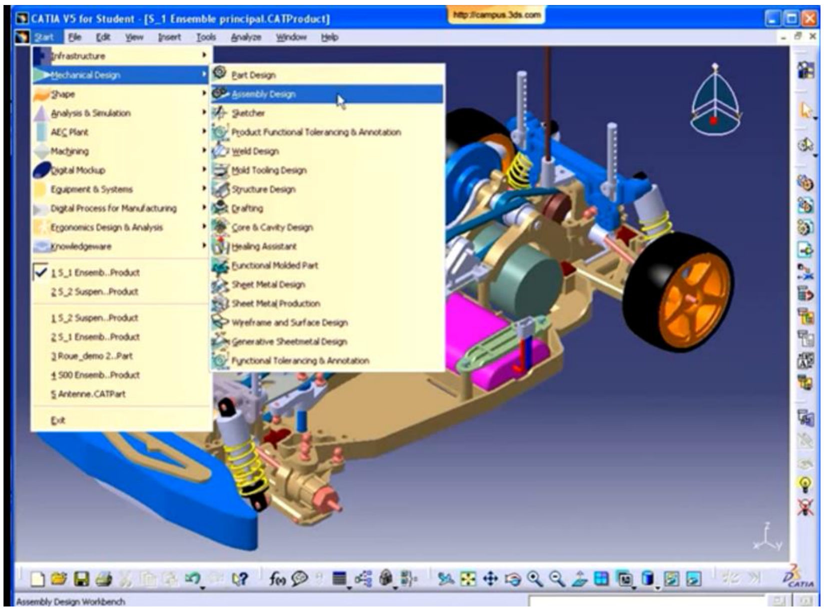
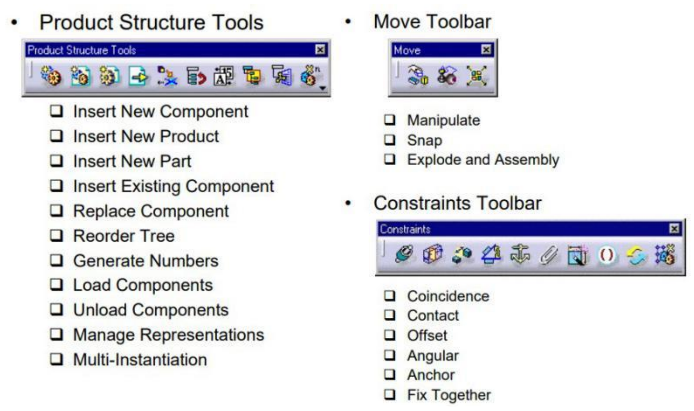


FIG: 3.7 ASSEMBLY MODULE WINDOW



- Product Structure Tools
  - ❑ Insert New Component
  - ❑ Insert New Product
  - ❑ Insert New Part
  - ❑ Insert Existing Component
  - ❑ Replace Component
  - ❑ Reorder Tree
  - ❑ Generate Numbers
  - ❑ Load Components
  - ❑ Unload Components
  - ❑ Manage Representations
  - ❑ Multi-Instantiation
- Move Toolbar
  - ❑ Manipulate
  - ❑ Snap
  - ❑ Explode and Assembly
- Constraints Toolbar
  - ❑ Coincidence
  - ❑ Contact
  - ❑ Offset
  - ❑ Angular
  - ❑ Anchor
  - ❑ Fix Together

FIG: 3.8 FEATURES OF ASSEMBLY DESIGN

Steps for creating an assembly

1. Create a new catproduct using File -> New -> Product.
2. Use the product structure tools to layout the main assembly structure.
3. Use Insert Existing Component or Insert New Part to create geometry in the assembly.
4. Use constraints to capture the design relationships between the various parts of the assembly.
5. Assembly information is stored in a file with the extension.CATProduct.
6. It contains only information relating to the product assembly.
7. The detailed geometric information about the parts in the assembly is referenced to the original.CATPart files.

NOTE: We have two approaches in assembly design, namely:

1. Top-down approach

2. Bottom-up approach

- In the top-down approach, the entire design structure will be created in the production environment.
- In the bottom-up approach, parts will be created separately and will be mated using mating or constraint tools.

3.2 3D DATA CONVERSION-PIXYZ STUDIO

Traditional 3D staging & Digital content creation tools (Unity3D, 3DS Max, Maya, Blender, Unreal Engine...) can natively read 3D files containing tessellated objects (meshes) like .fbx, .dae, .3ds and .obj files.

However native CAD files, created for engineering, manufacturing, and interoperability purposes in professional CAD solutions (CATIA, SIEMENS NX, SolidWorks, CREO, Inventor, STEP,...), cannot be easily imported in these tools.

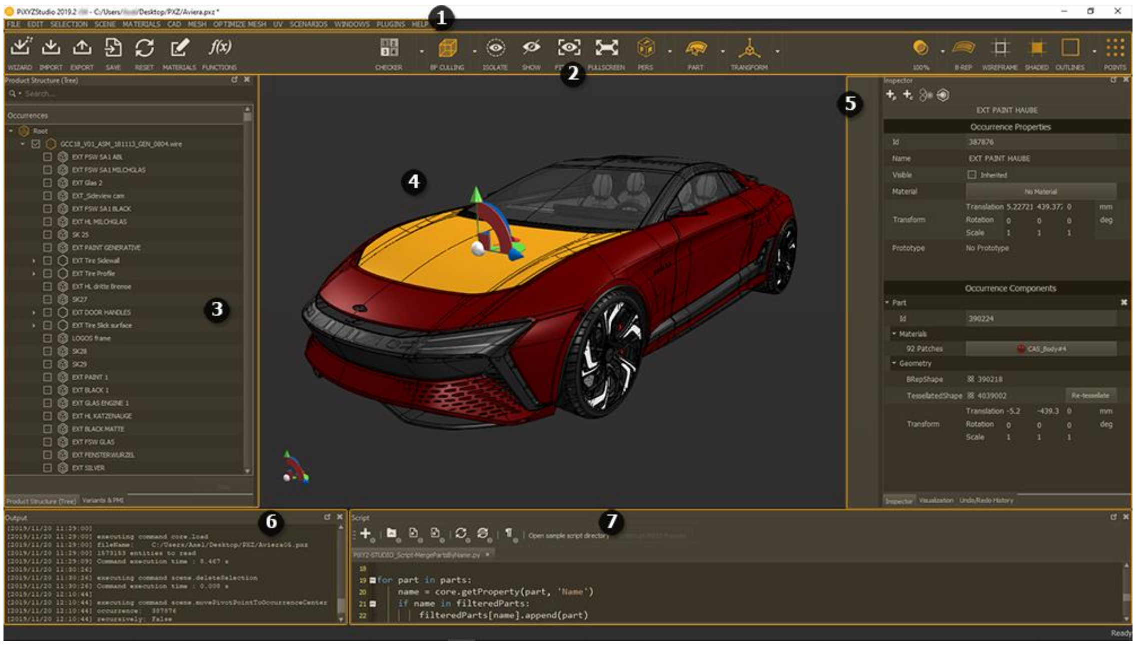
PiXYZ STUDIO is a unique 3D Data Preparation tool providing best-in-class tessellation, enabling the transformation of CAD data coming from almost all industry-leading CAD solutions into a lightweight, optimized mesh, ready to be exported.

Also, when importing and processing CAD data with PiXYZ STUDIO, the integrity of all the information contained in the original CAD file is preserved: product structure (hierarchy), original CAD surfaces, materials assignment, and appearance. PiXYZ STUDIO is the perfect tool for bringing complex data from the CAD-engineering world to the visualization world, allowing effortless integration for creating new and powerful visualization experiences.

Supported File Formats

PiXYZ STUDIO can import a wide range of CAD file formats from industry-leading solutions and standards (CATIA, NX, SolidWorks, Alias, STEP, IGES...), and tessellated/mesh file formats (FBX, OBJ, JT…).

User Interface


1
2
3
4
5
6
7
FILE EDIT SELECTION SCENE MATERIALS CAD MESH OPTIMIZE MESH UV SCENARIOS WINDOWS PLUGINS HELP
WIZARD IMPORT EXPORT SAVE RESET MATERIALS FUNCTIONS
CHECKER BF CULLING ISOLATE SHOW FIT FULLSCREEN PERS PART TRANSFORM
100% B-REP WIREFRAME SHADED OUTLINES POINTS
Product Structure (Tree)
Inspector
Output
Script

```
[2019/11/20 11:29:00]
[2019/11/20 11:29:00] executing command core.load
[2019/11/20 11:29:00] fileName:     C:/Users/Axel/Desktop/PXZ/Aviera06.pxz
[2019/11/20 11:29:00] 1573153 entities to read
[2019/11/20 11:29:09] Command execution time : 8.467 s
[2019/11/20 11:30:26]
[2019/11/20 11:30:26] executing command scene.deleteSelection
[2019/11/20 11:30:26] Command execution time : 0.000 s
[2019/11/20 12:10:44]
[2019/11/20 12:10:44] executing command scene.movePivotPointToOccurrenceCenter
[2019/11/20 12:10:44] occurrence:  387876
[2019/11/20 12:10:44] recursively: False
```

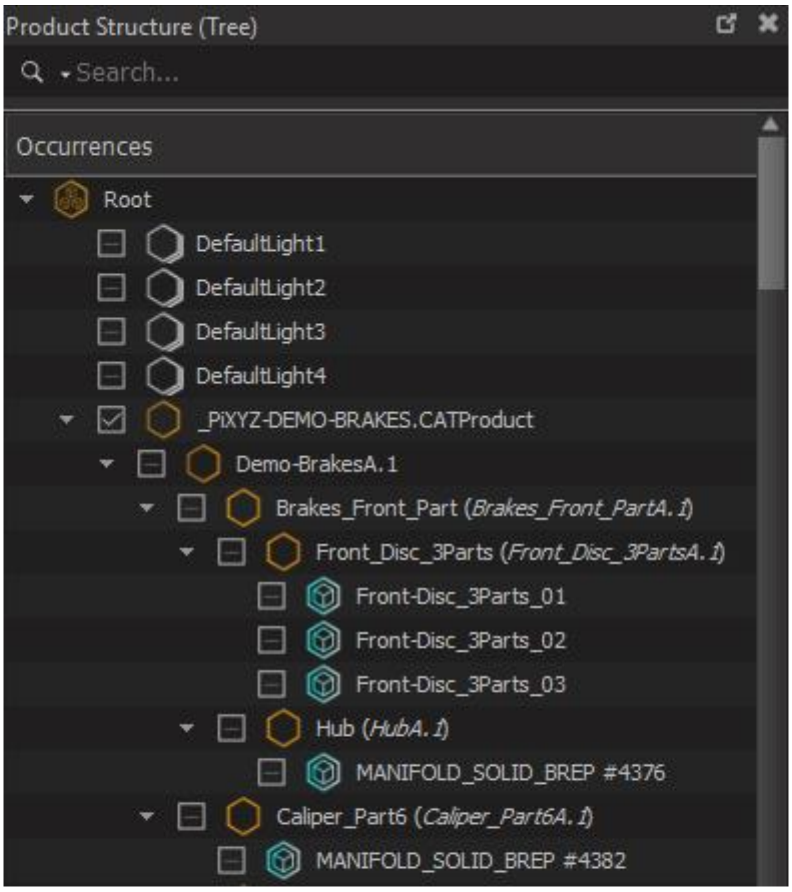
```
for part in parts:
    name = core.getProperty(part, 'Name')
    if name in filteredParts:
        filteredParts[name].append(part)
```

FIG: 3.9 PiXYZ USER INTERFACE WINDOW

PiXYZ STUDIO user interface is divided into several panels and windows:

- (1) Menu Bar
- (2) Main Toolbar
- (3) Product Structure
- (4) Viewport
- (5) Inspector
- (6) Log output
- (7) Script editor

About the Product Structure (Tree)



```
Product Structure (Tree)
Search...
Occurrences
Root
    DefaultLight1
    DefaultLight2
    DefaultLight3
    DefaultLight4
    _PiXYZ-DEMO-BRAKES.CATProduct
        Demo-BrakesA.1
            Brakes_Front_Part (Brakes_Front_PartA.1)
                Front_Disc_3Parts (Front_Disc_3PartsA.1)
                    Front-Disc_3Parts_01
                    Front-Disc_3Parts_02
                    Front-Disc_3Parts_03
                Hub (HubA.1)
                    MANIFOLD_SOLID_BREP #4376
            Caliper_Part6 (Caliper_Part6A.1)
                MANIFOLD_SOLID_BREP #4382
```

FIG: 3.10 PRODUCT STRUCTURE WINDOW

Like many other CAD solutions, the **Product Structure** in PiXYZ STUDIO designates the **scene tree** (or scene hierarchy), that provides a structured view of the occurrences of the scene. When a CAD/3D model is imported in PiXYZ STUDIO, the Product Structure reflects its hierarchy as it was designed in the CAD/3D solution that produced it. Then, this hierarchy can be modified to fit the user's need, interactively, or through Python scripting.

DIFFERENCES BETWEEN CAD MODELS AND MESH MODELS

CAD models inherited from CAD software (CATIA, NX, SolidWorks, Alias, STEP) are not tessellated. They contain exact parametric/mathematical surfaces. A CAD body (closed volume) or CAD surface (open-shell surface) is composed of CAD faces (or patches), delimited by boundaries.

To be displayed in a 3D application, these CAD's faces need to be translated into meshes. A mesh is composed of multiply connected polygons, or triangles (1 polygon = 2 coplanar triangles), forming a mesh surface that is understandable by a Graphics Card, to be rendered in a 3D application.

Moreover, CAD models can contain additional engineering and design data (metadata, PMI,...), that can be very useful to perform a targeted Data Preparation process based on targeted properties.

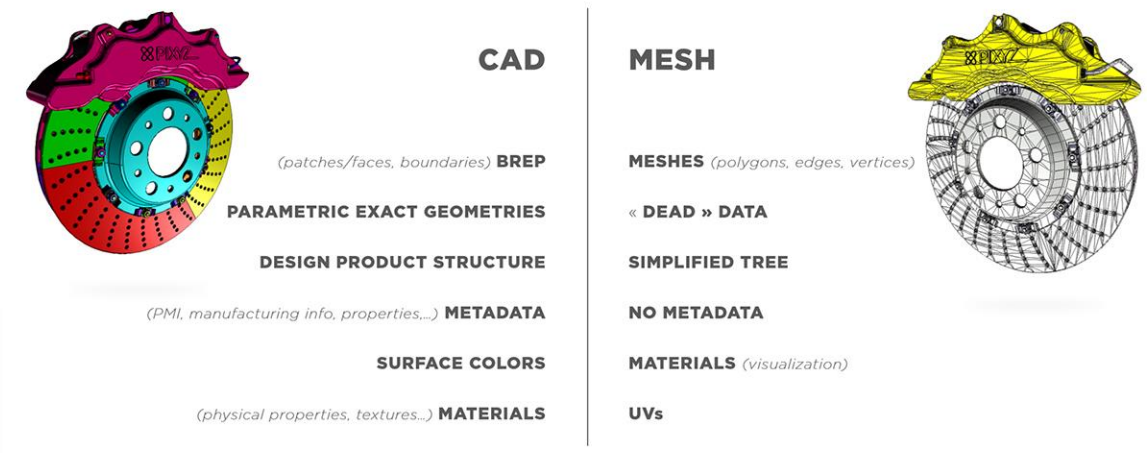

CAD
MESH
(patches/faces, boundaries) BREP
MESHES (polygons, edges, vertices)
PARAMETRIC EXACT GEOMETRIES
« DEAD » DATA
DESIGN PRODUCT STRUCTURE
SIMPLIFIED TREE
(PMI, manufacturing info, properties,...) METADATA
NO METADATA
SURFACE COLORS
MATERIALS (visualization)
(physical properties, textures...) MATERIALS
UVs

FIG: 3.11 DIFFERENCE BETWEEN CAD AND MESH MODELS

DCC software (Maya, 3DSMax, Blender, ...) natively create tessellated geometries, or meshes, than can be exported as FBX files to be re-imported in Unity3D using PiXYZ PLUGIN. Already tessellated meshes are meant to be optimized to create LODs by performing efficient and conservative decimation on them

ABOUT TESSELLATION AND DECIMATION

TESSELLATION

In surface modeling and solid modeling, Tessellation is the method used to represent 3D objects as a collection of triangles or other polygons. All surfaces, both curved and straight, are turned into triangles either at the time they are first created or in real-time when they are rendered. The more triangles used to represent a surface, the more realistic the rendering, but the more computation is required.

When creating a mesh out of a CAD model, PiXYZ tessellation algorithm uses several parameters to create a surface mesh:

MAX SAG: The maximum distance between the geometry and the tessellation. This parameter ensures that the mesh is similar enough to the original analytical surface (exact geometry).

A low value means that a very fine mesh is created. The distance values are expressed in millimeters.

MAX ANGLE: The maximum allowed angle between normals of two adjacent polygons (on the same face). It allows adding more precision on short radius fillets.

Adjust the «Max angle» parameter to keep enough polygons in high curvature areas whose radius is lower than the «Max sag» value: fillets and chamfers for example.

MAX LENGTH: Used to control the length of a polygon (edge). For a rendering usage, it is often not recommended to use the «Max Length» parameter. It increases the polygons count without significant improvement in the visual aspect. But in case of very long objects (a plane body, a train cabin), this setting can avoid lighting artifacts caused by too long polygons.

DECIMATION When optimizing a tessellated model, PiXYZ PLUGIN uses this algorithm to reduce the mesh polygon density by smartly deleting vertices. It allows precise control, preserving from bad smoothing and topological irregularities.

The algorithm uses a combination of the 3 following parameters to obtain the lighter model possible while keeping an acceptable quality:

•**SURFACIC TOLERANCE**: the maximum distance between vertices of the original model and resulting in simplified surfaces.

•**LINEIC TOLERANCE**: the maximum distance between lineic vertices of the original model and resulting in simplified lines.

•**NORMAL TOLERANCE**: the maximum angle between the original normals and those interpolated on the simplified surface.

The **Normal Tolerance** setting preserves the quality of how the light reacts on a surface/mesh. Combined with the **Surface Tolerance** setting, it will act as a quality controller, keeping polygons where the surface curvature is important, preserving the visual quality of the model.
The **Lineic Tolerance** is meant to preserve the boundaries of the original surface (where the edges are sharp).

3.3 DEVELOP -UNITY 3D

Unity is a game engine developed by Unity Technologies, used to create three-dimensional, two-dimensional, virtual reality, and augmented reality games, as well as simulations and other experiences. The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, engineering and construction.

Installing the Unity Hub

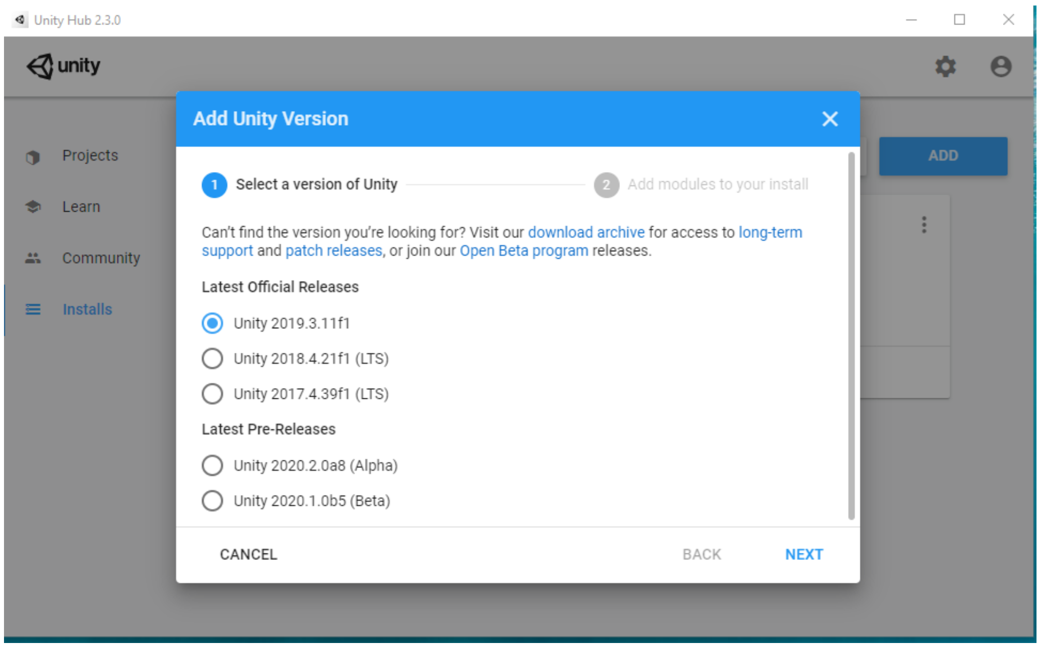
The Unity Hub is a management tool that can be used to manage all Unity Projects and installations. Use the Hub to manage multiple installations of the Unity Editor along with their associated components, create new Projects, and open existing Projects.

Installing the Unity Editor

To install the Editor:

Click the **Installs** tab.

Click the **Add** button and select a specific version of the Editor.



Add Unity Version

1 Select a version of Unity — 2 Add modules to your install

Can't find the version you're looking for? Visit our download archive for access to long-term support and patch releases, or join our Open Beta program releases.

Latest Official Releases

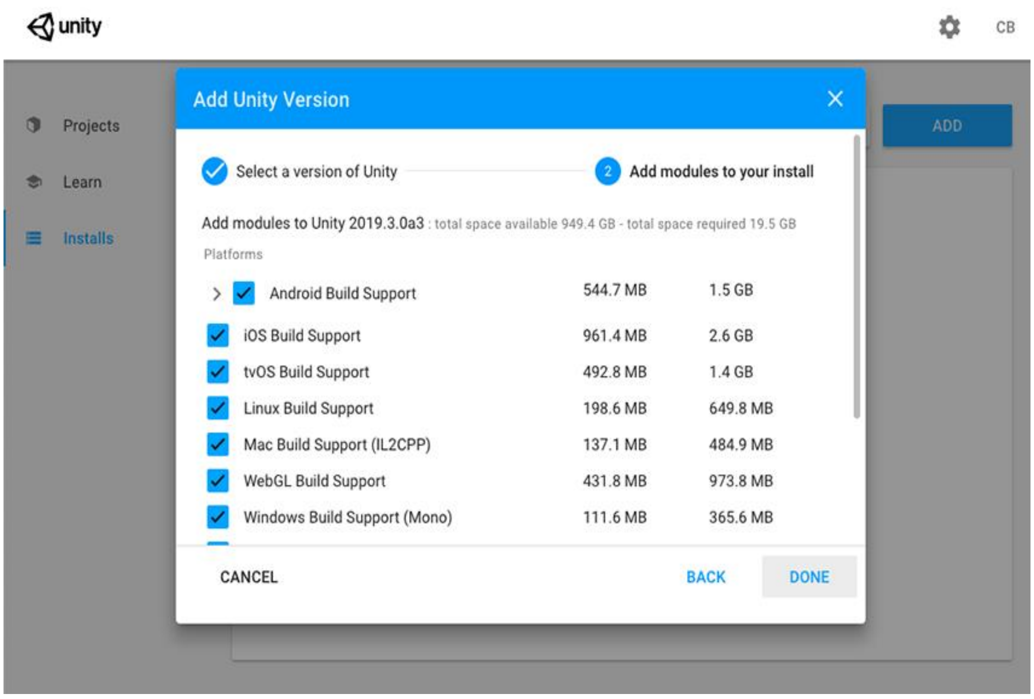
[✓ Unity 2019.3.11f1]
Unity 2018.4.21f1 (LTS)
Unity 2017.4.39f1 (LTS)

Latest Pre-Releases

Unity 2020.2.0a8 (Alpha)
Unity 2020.1.0b5 (Beta)

FIG: 3.12 EDITOR INSTALL SCREEN

Click the **Next** button and select the modules you want to install with the Editor. If you don't install a component now, you can add it later if you need to. When you've selected all the modules you need, click **Done**.



Add modules to Unity 2019.3.0a3 : total space available 949.4 GB - total space required 19.5 GB

Platforms

Android Build Support	544.7 MB	1.5 GB
iOS Build Support	961.4 MB	2.6 GB
tvOS Build Support	492.8 MB	1.4 GB
Linux Build Support	198.6 MB	649.8 MB
Mac Build Support (IL2CPP)	137.1 MB	484.9 MB
WebGL Build Support	431.8 MB	973.8 MB
Windows Build Support (Mono)	111.6 MB	365.6 MB

FIG: 3.13 MODULES INSTALL SCREEN

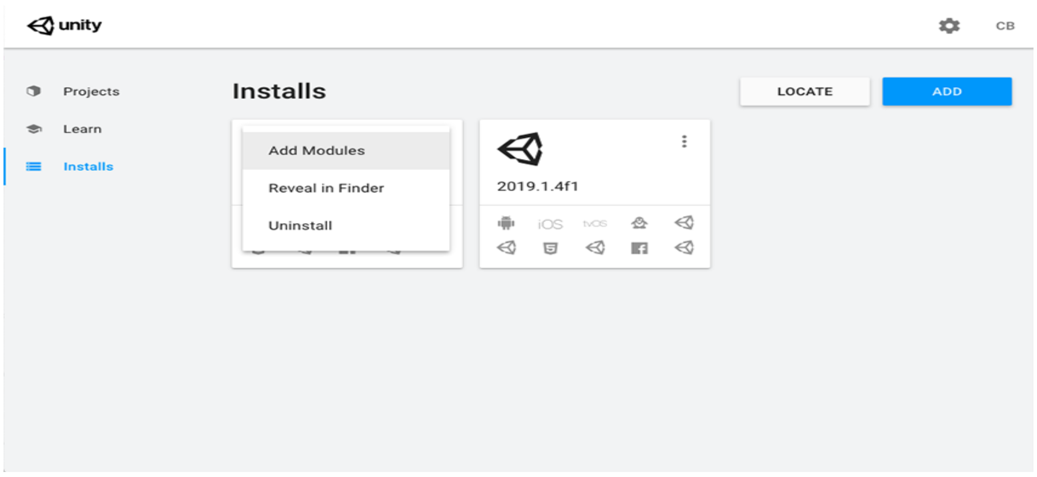


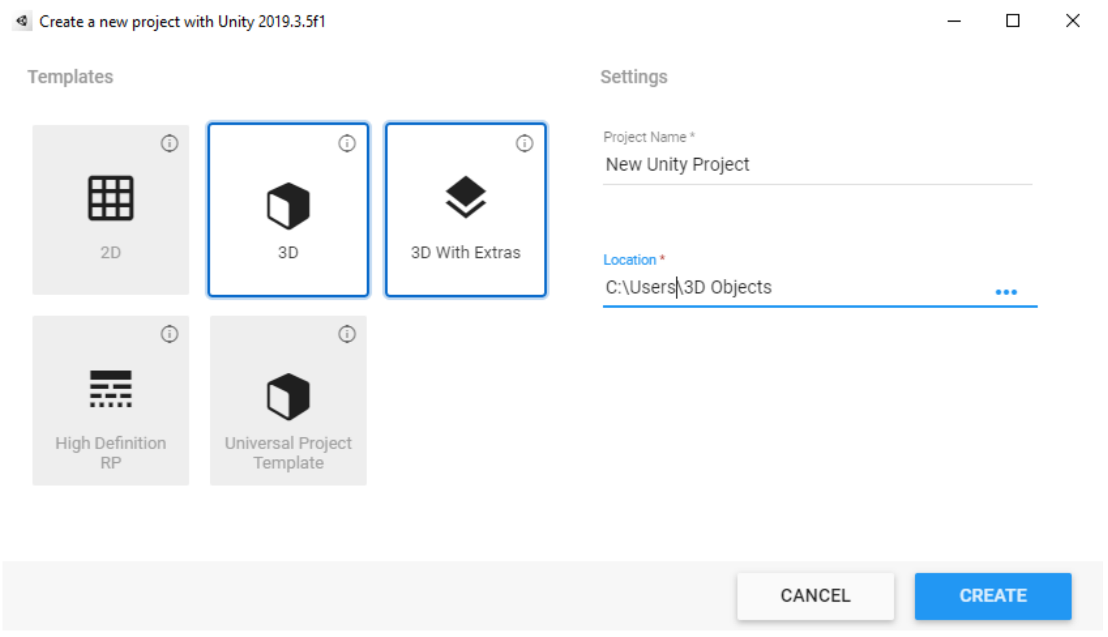
FIG: 3.14 MODTFYING AN EXISTING EDITOR

Adding modules to the Unity Editor

1. Click the three dots to the right of the version label, then select **Add Modules**.
2. In the **Add Modules** dialog, locate the module to add and tick its checkbox.
3. When you have selected all the modules to add, select **Done**.

Creating a new project

When you create a Project, you select a template with which to initialize your Project.


Create a new project with Unity 2019.3.5f1

Settings

Project Name *
New Unity Project

Location *
C:\Users\3D Objects

FIG: 3.15 PROJECT TEMPLATE DROP- DOWN SELECTION

Opening existing Project

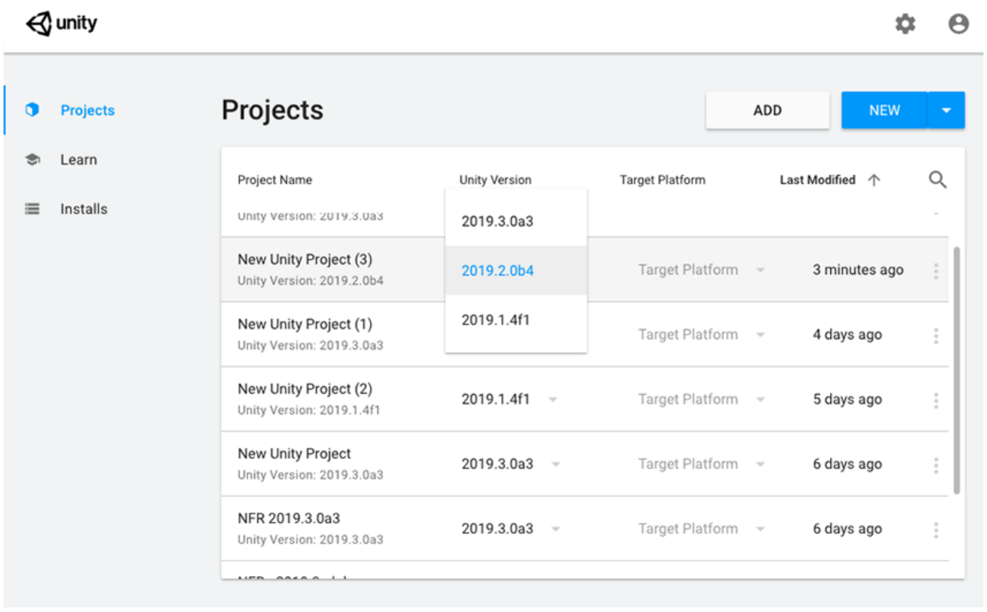
You have several options when opening an existing Project.

In the Unity Hub, click on the Project to open it with the assigned Unity Editor version. If the Hub can't find a matching Editor version for the Project, it displays a warning message asking you to download that version.

You can also:

- Click the drop-down arrow next to the **Unity Version** to select a different version of the Editor. If you try to open a project with an older version of the Editor than the one it was created in, Unity warns you that downgrading the Project might result in data loss and asks you to confirm your selection.
- Click the drop-down arrow next to the **Target Platform** option to select a different platform.

In the Editor, click File > Open Project to open the Hub and access the list of available Projects.



unity

Projects

Learn

Installs

Projects

ADD

NEW

Project Name	Unity Version	Target Platform	Last Modified
Unity Version: 2019.3.0a3	2019.3.0a3		
New Unity Project (3) Unity Version: 2019.2.0b4	2019.2.0b4	Target Platform	3 minutes ago
New Unity Project (1) Unity Version: 2019.3.0a3	2019.1.4f1	Target Platform	4 days ago
New Unity Project (2) Unity Version: 2019.1.4f1	2019.1.4f1	Target Platform	5 days ago
New Unity Project Unity Version: 2019.3.0a3	2019.3.0a3	Target Platform	6 days ago
NFR 2019.3.0a3 Unity Version: 2019.3.0a3	2019.3.0a3	Target Platform	6 days ago

FIG: 3.16 AVAILABLE PROJECTS WINDOW

Unity's interface

This section provides a detailed tour of the most common editor windows, and how to make full use of them.

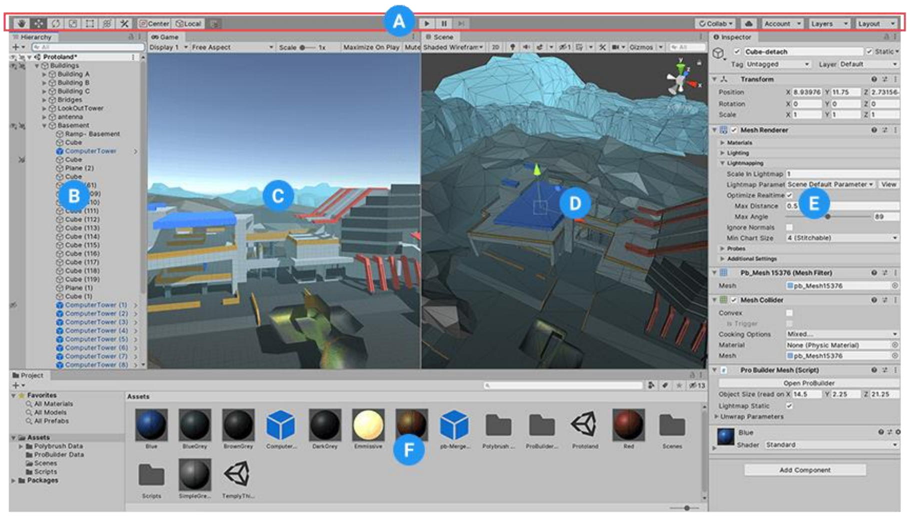


FIG: 3.17 UNITY INTERFACE WINDOW

(A) The **Toolbar** provides access to the most essential working features. On the left, it contains the basic tools for manipulating the **Scene view** and the **Game**

Objects within it. In the center are the play, pause, and step controls. The buttons to the right give you access to Unity Collaborate, Unity Cloud Services, and your Unity Account, followed by a layer visibility menu, and finally the Editor layout menu.

(B) The **Hierarchy** window is a hierarchical text representation of every Game Object in the **Scene**. Each item in the **Scene** has an entry in the hierarchy, so the two windows are inherently linked. The hierarchy reveals the structure of how Game Objects attach.

(C) The **Game view** simulates what your final rendered game will look like through your Scene **Cameras**. When you click the Play button, the simulation begins.

(D) The **Scene view** allows you to visually navigate and edit your Scene. The Scene view can show a 3D or 2D perspective, depending on the type of project you are working on.

(E) The **Inspector Window** allows you to view and edit all the properties of the currently selected Game Object. Because different types of Game Objects have different sets of properties, the layout and contents of the **Inspector** window change each time you select a different Game Object.

(F) The **Project window** displays your library of Assets that are available to use in your Project. When you import assets into your project, they appear here

Positioning Game Objects:

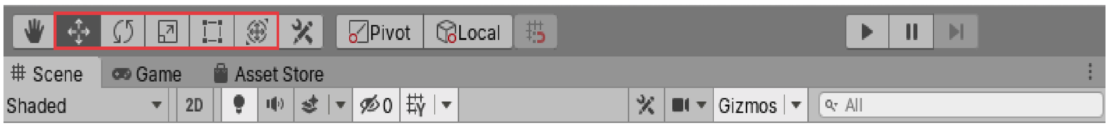


FIG: 3.18 TOOL BAR FOR POSITIONING THE GAME OBJECTS

To alter the Transform component of the **GameObject**, use the mouse to manipulate any **Gizmo** axis, or type values directly into the number fields of the **Transform** component in the **Inspector. In the inspector, the transform** tool combines the **Move**, **Rotate**, and **Scale** tools. Its Gizmo provides handles for movement and rotation. When the **Tool Handle Rotation** is set to **Local**, the Transform tool also provides handles for scaling the selected Game Object.

Alternatively, you can select each of the four **Transform** modes with a hotkey: **W** for Move, **E** for Rotate, **R** for Scale, **T** for Rect Transform, and **Y** for Transform.

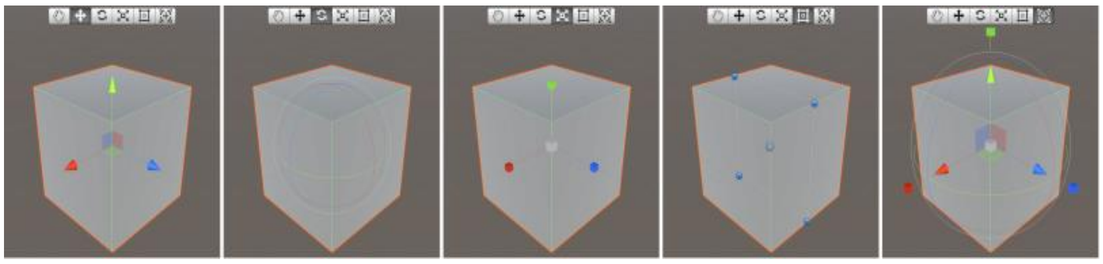


FIG: 3.19 MOVE, ROTATE, SCALE, RECT TRANSFORM AND TRANSFORM GIZMOS

Vertex snapping: Snap any vertex from a given **Mesh** to the position of another **Mesh**'s vertex or surface. You can snap vertex to vertex, vertex to the surface, and pivot to the vertex. Use **vertex snapping** to quickly assemble your Scenes.

Follow the steps below to use vertex snapping:

1. Select the Mesh you want to manipulate and make sure the **Move** tool is active.

2. Press and hold the **V** key to activate the vertex snapping mode.
3. Move your cursor over the vertex on your Mesh that you want to use as the pivot point.
4. Hold down the left mouse button once your cursor is over the vertex you want and drag your Mesh next to any other vertex on another Mesh.
5. To snap a vertex to a surface on another Mesh, add and hold down the **Shift+Ctrl** (Windows) or **Shift+Command** (macOS) key while you move over the surface you want to snap to.
6. To snap the pivot to a vertex on another Mesh, add and hold the **Ctrl** (Windows) or **Command** (macOS) key while you move the cursor to the vertex you want to snap to release the mouse button and the **V** key when you are happy with the results (**Shift+V** acts as a toggle of this functionality).

Look-at rotation

While using the **Rotate** tool, hold **Shift** and **Control** (**Command** on Mac) to rotate the Game Object towards a point on the surface of any **Collider**.

Common types of Assets

Image files

Unity supports the most common image file types, such as BMP, TIF, TGA, JPG, and PSD. If you save your layered Photoshop (.psd) files into your Assets folder, Unity imports them as flattened images.

FBX and Model files

Since Unity supports the FBX file format, you can import data from any 3D modeling software that supports FBX.

Note: You can also save your 3D files from the most common 3D modeling software in their native format (for example, .max, .blend, .mb, .ma).

Meshes and animations

Whichever 3D modeling software you are using, Unity imports the Meshes and animations from each file. Your **Mesh** file does not need to have an animation to be imported. If you do use animations, you can import all animations from a single file, or import separate files, each with a single animation.

Audio files

If you save uncompressed audio files into your Assets folder, Unity imports them according to the **compression** settings specified.

Standard Assets

Unity ships with multiple **Standard Assets**. These are collections of Assets that most Unity customers use. These include 2D, **Cameras**, Characters, CrossPlatformInput, Effects, Environment, ParticleSystems, Prototyping, Utility, and Vehicles.

To transfer **Standard Assets** in and out of Projects, Unity uses **Asset packages**, available on the Unity **Asset Store**. Asset packages allow you to share and re-use Unity Projects and collections of Assets.

Using the Asset Store

The Unity Asset Store has a library of free and commercial Assets that Unity Technologies, as well as members of the community, create. A wide variety of Assets are available, covering everything from Textures, Models, and animations - to whole Project examples, tutorials, and Editor extensions. You can access the Assets from an interface that is built into the Unity Editor which allows you to download and import Assets directly into your Project.

Asset Store access and navigation

To open the Asset Store window, go to **Window** > **Asset Store** in the Editor. In the first visit, create a free user account that can use to log into the Store.

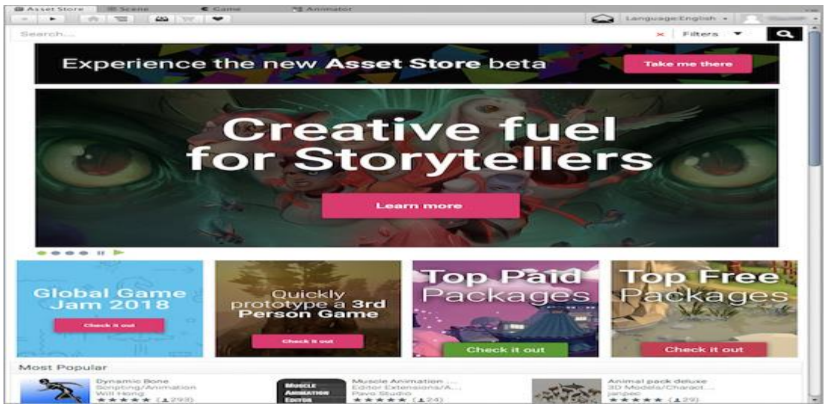
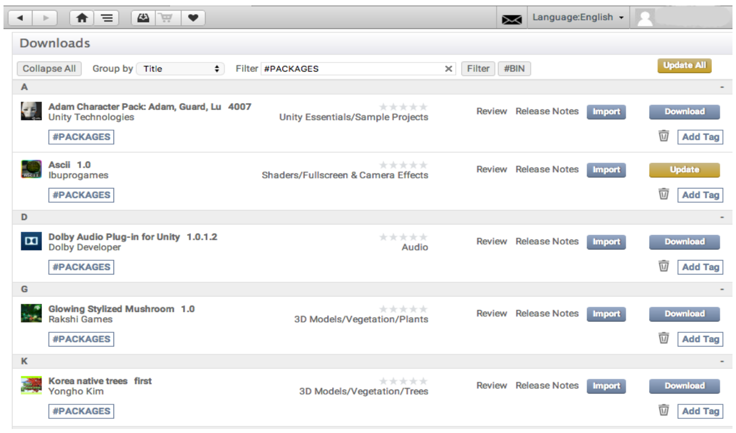


FIG: 3.20 ASSET STORE FRONT PAGE

Use the Download Manager to view the packages that already bought as well as find and install any updates.



Downloads

Collapse All | Group by Title | Filter #PACKAGES | Filter | #BIN | Update All

A				
Adam Character Pack: Adam, Guard, Lu 4007 Unity Technologies #PACKAGES	Unity Essentials/Sample Projects	Review Release Notes	Import	Download Add Tag
Ascii 1.0 Ibuprogames #PACKAGES	Shaders/Fullscreen & Camera Effects	Review Release Notes	Import	Update Add Tag
D				
Dolby Audio Plug-in for Unity 1.0.1.2 Dolby Developer #PACKAGES	Audio	Review Release Notes	Import	Download Add Tag
G				
Glowing Stylized Mushroom 1.0 Rakshi Games #PACKAGES	3D Models/Vegetation/Plants	Review Release Notes	Import	Download Add Tag
K				
Korea native trees first Yongho Kim #PACKAGES	3D Models/Vegetation/Trees	Review Release Notes	Import	Download Add Tag

FIG: 3.21 DOWNLOAD MANAGER WINDOW

Scenes

Scenes contain the environments and menus of the game. When creating a new Unity project, the **scene view** displays a new Scene. This Scene is *untitled* and *unsaved*. The Scene is empty except for a **Camera** (called **Main Camera**) and a Light (called **Directional Light**).

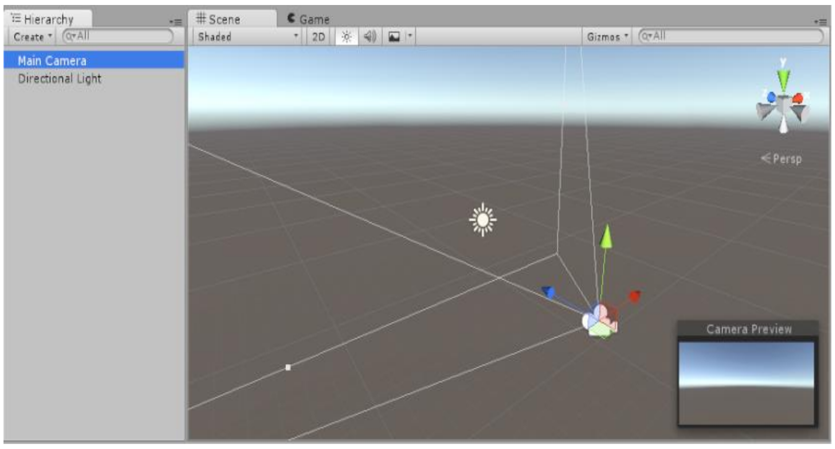


FIG: 3.22 SCENE VIEW WINDOW

Lights

Lights are an essential part of every **scene**. While meshes and textures define the shape and look of a **scene**, lights define the color and mood of your 3D environment. You'll likely work with more than one light in each **scene.**

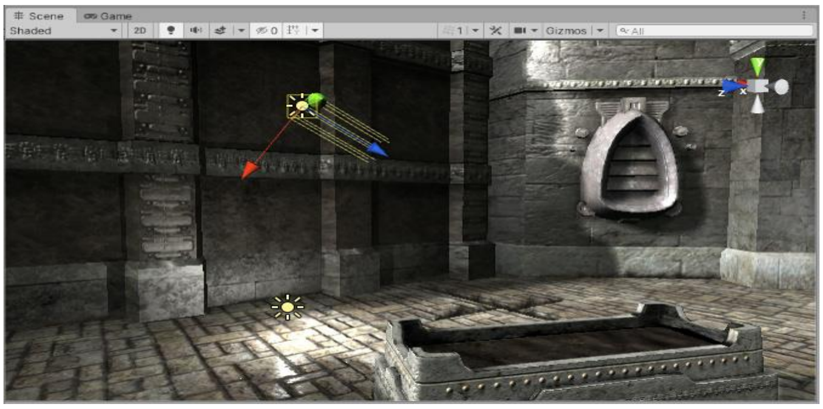
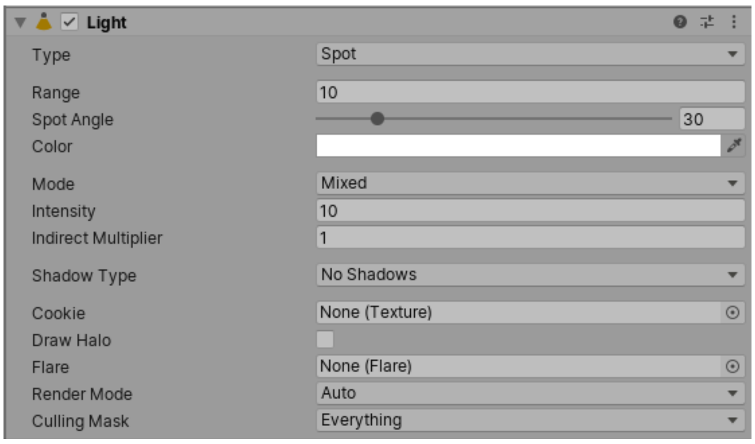


FIG: 3.23 A SIMPLE TWO LIGHT SET UP

Lights can be added to the scene from the **GameObject->Light** menu. Once a light has been added, you can manipulate it like any other GameObject. Additionally, you can add a Light Component to any selected GameObject by using **Component->Rendering->Light**. There are many different options within the Light Component in the **Inspector**.



Light

Type	Spot
Range	10
Spot Angle	30
Color	⬜
Mode	Mixed
Intensity	10
Indirect Multiplier	1
Shadow Type	No Shadows
Cookie	None (Texture)
Draw Halo	☐
Flare	None (Flare)
Render Mode	Auto
Culling Mask	Everything

FIG: 3.24 LIGHT COMPONENTS PROPERTIES IN INSPECTOR

Cameras

Just as cameras are used in films to display the story to the audience, **Cameras** in Unity are used to display the game world to the player. You will always have at least one camera in a **scene**, but you can have more than one. Multiple cameras can give you a two-player split-screen or create advanced custom effects. You can animate cameras, or control them with physics. Practically anything you can imagine is possible with cameras, and you can use typical or unique cameras to fit your game's style.

Saving Scenes

To save the Scene you're currently working on, choose **File** > **Save Scene** from the menu, or press Ctrl + S (Windows) or Cmd + S (macOS).

Unity saves Scenes as Assets in your project's *Assets* folder. This means they appear in the Project window, with the rest of your Assets.

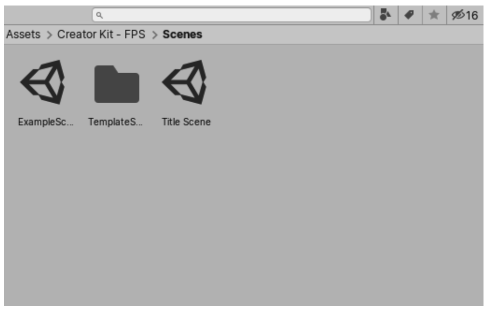
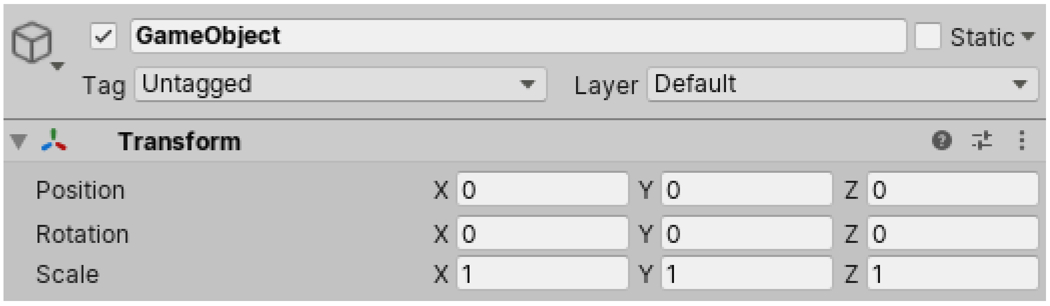


FIG: 3.25 SAVED SCENES IN ASSETS

Using Components

Components are the nuts & bolts of objects and behaviors in a game. They are the functional pieces of every **GameObject**. A GameObject is a container for many different Components. By default, all GameObjects automatically have a **Transform** Component. This is because the Transform dictates where the GameObject is located, and how it is rotated and scaled. Without a Transform Component, the GameObject wouldn't have a location in the world. Try creating an empty GameObject now as an example. Click the **GameObject->Create Empty** menu item. Select the new GameObject, and look at the**Inspector**).

.

GameObject Static
Tag Untagged Layer Default
Transform

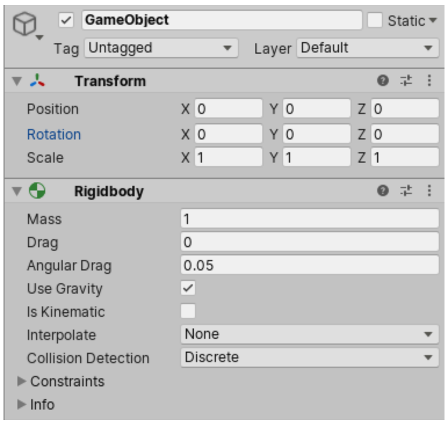
Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

FIG:3.26 EVEN EMPTY GAME OBJECT HAVE TRANFORM COMPONENT

Remember that you can always use the Inspector to see which Components are attached to the selected GameObject. As Components are added and removed, the Inspector will always show you which ones are currently attached. You will use the Inspector to change all the properties of any Component (including scripts).

Adding Components

You can add Components to the selected GameObject through the Components menu. We'll try this now by adding a **Rigidbody** to the empty GameObject we just created. Select it and choose **Component->Physics->Rigidbody** from the menu.



✓ GameObject ☐ Static

Tag Untagged Layer Default

Transform

Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Rigidbody

Mass	1
Drag	0
Angular Drag	0.05
Use Gravity	✓
Is Kinematic	☐
Interpolate	None
Collision Detection	Discrete

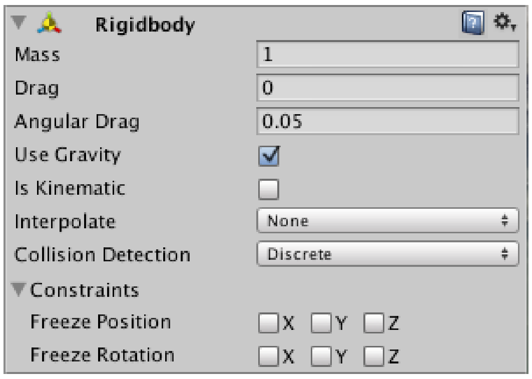
▶ Constraints

▶ Info

FIG:3.27 AN EMPTY COMPONENT WITH A RIGID BODY ATTACHMENT

Rigidbody

Rigidbodies enable your **GameObjects** to act under the control of physics. The Rigidbody can receive forces and torque to make your objects move realistically. Any GameObject must contain a Rigidbody to be influenced by gravity, act under added forces via scripting, or interact with other objects through the NVIDIA PhysX **physics engine**.



Rigidbody

Mass	1
Drag	0
Angular Drag	0.05
Use Gravity	✓
Is Kinematic	☐
Interpolate	None
Collision Detection	Discrete

▼ Constraints

Freeze Position	☐ X	☐ Y	☐ Z
Freeze Rotation	☐ X	☐ Y	☐ Z

FIG: 3.28 RIGIDBODY COMPONENT IN INSPECTOR

Details

The biggest difference between manipulating the Transform versus the Rigidbody is the use of forces. Rigidbodies can receive forces and torque, but Transforms cannot. Transforms can be translated and rotated, but this is not the same as using physics. Adding forces/torque to the Rigidbody will change the object's position and rotation of the **Transform component**.

Parenting

Parenting is one of the most important concepts to understand when using Unity. When a GameObject is a **Parent** of another GameObject, the **Child** GameObject will move, rotate, and scale exactly as its Parent does. You can think of parenting as being like the relationship between your arms and your body; whenever your body moves, your arms also move along with it. Child objects can also have children of their own and so on. So your hands could be regarded as “children” of your arms and then each hand has several fingers, etc. Any object can have multiple children, but only one parent. These multiple levels of parent-child relationships form a **Transform** hierarchy. The object at the very top of a hierarchy (i.e, the only object in the hierarchy that doesn’t have a parent) is known as the **root**.

You can create a Parent by dragging any GameObject in the **Hierarchy View** onto another. This will create a Parent-Child relationship between the two GameObjects.

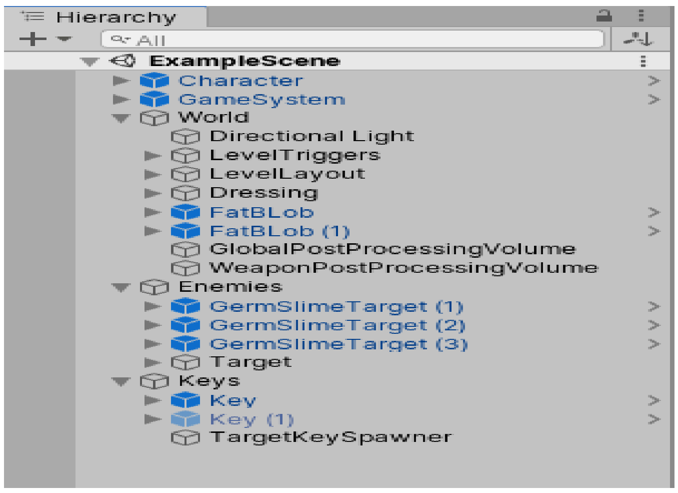


FIG: 3.29 PARENT CHILD HIERARCHY

Note that the Transform values in the Inspector for any child GameObject are displayed relative to the Parent’s Transform values. These values are referred to as **local coordinates**. For scene construction, it is usually sufficient to work with local coordinates for child objects but in gameplay, it is often useful to find their exact position in world space or **global coordinates**.

Box Collider

The **Box Collider** is a basic cuboid-shaped **collision** primitive.

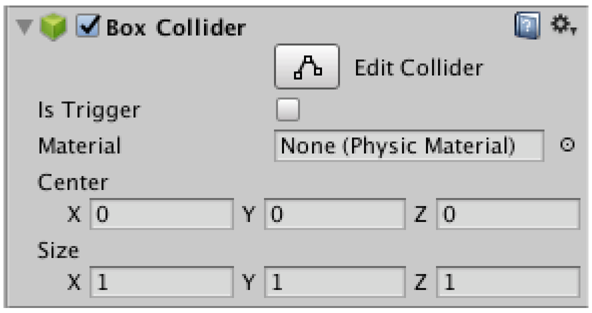

Box Collider
Edit Collider
Is Trigger ☐
Material: None (Physic Material)
Center: X 0 Y 0 Z 0
Size: X 1 Y 1 Z 1

FIG: 3.30 BOX COLLIDER IN INSPECTOR

Details

Box Colliders are rectangular cuboids and are useful for items such as crates or chests. However, you can use a thin box as a floor, wall, or ramp. The Box Collider is also a useful element in a Compound Collider. A vertex appears in the center of each face of the Box Collider in Edit Mode. To move the vertices, drag them when the mouse is over the vertex to make the Box Collider bigger and smaller.

Mesh Collider

The **Mesh Collider** takes a Mesh Asset and builds its **Collider** based on that Mesh. It is more accurate for **collision detection** than using primitives for complicated Meshes. Mesh Colliders that are marked as **Convex** can collide with other Mesh Colliders.

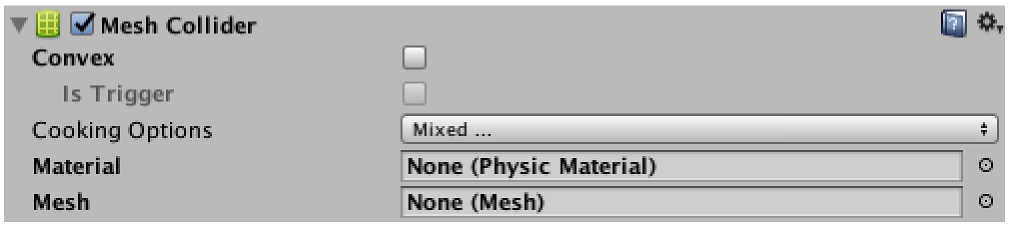

Mesh Collider
Convex ☐
Is Trigger ☐
Cooking Options: Mixed ...
Material: None (Physic Material)
Mesh: None (Mesh)

FIG: 3.31 MESH COLLIDER IN INSPECTOR

Details

The Mesh Collider builds its collision representation from the Mesh attached to the **GameObject** and reads the properties of the attached Transform to set its position and scale correctly.

The benefit of this is that you can make the shape of the Collider the same as the shape of the visible Mesh for the GameObject, which creates more precise and authentic collisions.

Primitive and placeholder objects

Unity can work with 3D models of any shape that can be created with modeling software. However, several primitive object types can be created directly within Unity, namely the **Cube**, **Sphere**, **Capsule**, **Cylinder**, **Plane**, and **Quad**.

These objects are often useful in their own right (a plane is commonly used as a flat ground surface, for example) but they also offer a quick way to create placeholders and prototypes for testing purposes. Any of the primitives can be added to the scene using the appropriate item on the **GameObject > 3D Object** menu.

Deactivating GameObjects

Unmark a **GameObject** as inactive to temporarily remove it from the **Scene**. To do this, navigate to the **Inspector** and uncheck the checkbox next to the **GameObject**'s name (see image below), or use the active self property in the script.

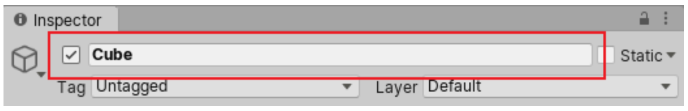


FIG: 3.32 A GAMEOBJECT'S ACTIVATION CHECK BOX

Prefabs

In Unity's **Prefab** system, **Prefab Assets** act as templates. Create **Prefab** Assets in the Editor, and they are saved as an Asset in the Project window. From **Prefab Assets**, create any number of **Prefab instances**. **Prefab** instances can either be created in the editor and saved as part of the **Scenes** or instantiated at runtime.

Tags

A **Tag** is a reference word which can assign to one or more **GameObjects**.

Creating new Tags

The **Inspector** shows the **Tag** and **Layer** drop-down menus just below any GameObject's name.

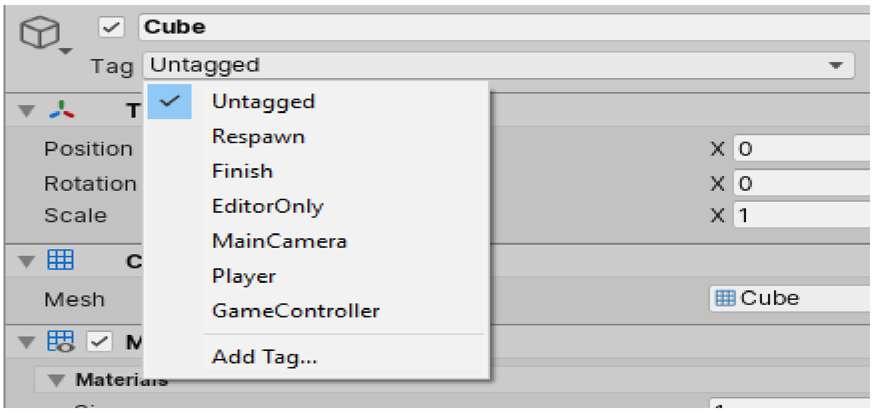


FIG: 3.33 TAGGING A GAMEOBJECT

To create a new Tag, select **Add Tag….** This opens the Tag and Layer Manager in the Inspector. Note that once a tag is named, it cannot be renamed later. Layers are similar to Tags but are used to define how Unity should render GameObjects in the scene.

Applying a Tag

To apply an existing Tag to a GameObject, open the **Tags** drop down and choose the Tag you want to apply. The GameObject is now associated with this Tag.

Static GameObjects

If a **GameObject** does not move at run time, it is known as a **static GameObject**. If a **GameObject** moves at run time, it is known as a **dynamic GameObject**.

Many systems in Unity can pre compute information about static GameObjects in the Editor. Because the GameObjects do not move, the results of these calculations are still valid at runtime. This means that Unity can save on run time calculations, and potentially improve performance.

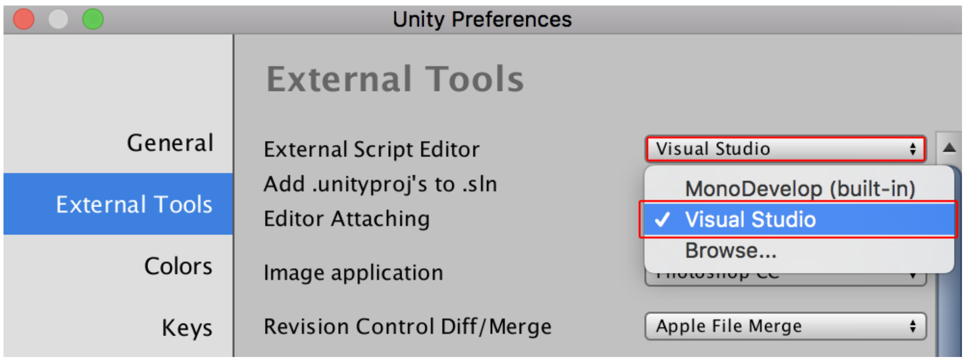
Visual Studio C# integration

Visual Studio is an Integrated Development Environment (IDE) tool from Microsoft. A more sophisticated C# development environment, think smart autocompletion, computer-assisted changes to source files, smart syntax highlighting, and more.

Unity’s Visual Studio integration allows you to create and maintain Visual Studio project files automatically. Also, it will open when you double click on a script or an error message in the Unity console.

Using Visual Studio with Unity

Follow these steps to configure the Unity Editor to use Visual Studio as its default IDE: In Unity, go to **Edit** > **Preferences**, and make sure that Visual Studio is selected as your preferred external editor.



External Tools

External Script Editor: Visual Studio
Add .unityproj's to .sln
Editor Attaching
Image application
Revision Control Diff/Merge: Apple File Merge

FIG: 3.34 EXTERNAL TOOL SETTINGS

Next, double-click a C# file in your project. Visual Studio should automatically open that file. You can edit the file, save, and switch back to Unity to test your changes. Unity automatically creates and maintains a Visual Studio .sln and .csproj file.

Creating and Using Scripts

The behavior of **GameObjects** is controlled by the **Components** that are attached to them. Unity allows creating own Components using **scripts**. It supports the C# programming language natively. C# (pronounced C-sharp) is an industry-standard language similar to Java or C++.

Creating Scripts

Create a new script from the Create menu at the top left of the Project panel or by selecting **Assets > Create > C# Script** from the main menu.

The new script will be created in whichever folder selected in the Project panel. The new script file's name will be selected, prompting you to enter a new name.

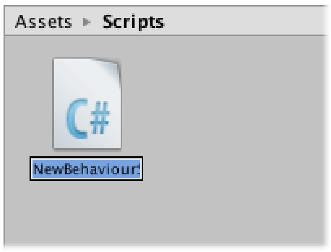


FIG: 3.34 CREATING SCRIPT NAME IN ASSETS

It is a good idea to enter the name of the new script at this point rather than editing it later. The name that you enter will be used to create the initial text inside the file, as described below.

Saving the work

Unity categorizes most save the information into Scene changes and Project-wide changes.

To save all Scene changes and Project-wide changes, go to **File** > **Save** (or **Save as**). This is the fastest way to save everything at once.
To save Project-wide changes, but not Scene changes, go to **File** > **Save Project**.

Scene changes

Scene changes include modifications to **GameObjects** in the **Scene**.

Project-wide changes

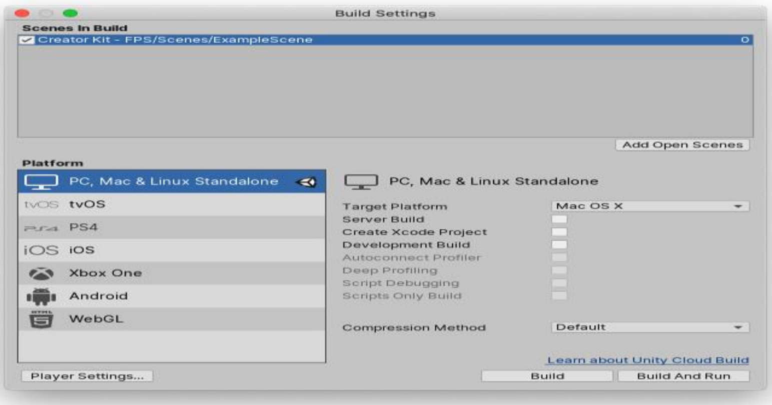
Project-wide changes in Unity apply to the whole project, rather than a specific Scene. It might be useful when a temporary Scene is created to test some changes

Build Settings

To configure and build apps for Android, access the Build Settings window (**File** > **Build Settings**). In **Platform**, select **Android**.

To set Android as your default build platform, click the **Switch Platform** button.

After you specify your build settings, click the **Build** button to create your build. To build the app, click **Build And Run** to create and run your build on the platform you have specified.


Build Settings

Scenes In Build

✓ Creator Kit - FPS/Scenes/ExampleScene 0

Platform

PC, Mac & Linux Standalone

Target Platform: Mac OS X
Server Build: ☐
Create Xcode Project: ☐
Development Build: ☐
Autoconnect Profiler: ☐
Deep Profiling: ☐
Script Debugging: ☐
Scripts Only Build: ☐

Compression Method: Default

FIG: 3.35 BUILD SETTINGS WINDOW

Tip: Specify the output path for the packages and then use the **Ctrl+B** (Windows) or **Cmd+B** (macOS) keyboard shortcut to **Build and Run** using the saved output path.

3.4 DEPLOY-OCULUS GO

The Oculus Go is a standalone virtual reality headset developed by Facebook Technologies in partnership with Qualcomm and Xiaomi. The Go is an all-in-one headset, meaning it contains all the necessary components to provide virtual reality experiences and doesn't require a connection to an external device to use.

FIG: 3.36 OCULUS GO HEAD SET

Software

For setting up the Oculus Go, the Oculus application must be downloaded on a mobile phone running Android or iOS. With the Oculus application, the headset is paired to the phone, connected to an account, and the controller is paired with the headset.

While pictures and videos can be transferred to the Go from a PC via USB and applications can be sideloaded to the device, many applications require a Wi-Fi connection to function.

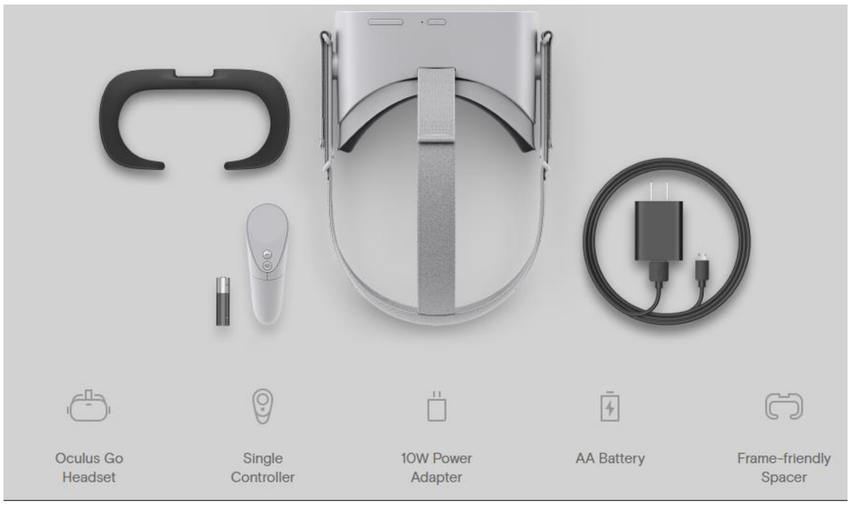


FIG: 3.37 INSIDE THE BOX

The Oculus Gallery app can be used to access media stored on the Go's internal memory or from external websites or services such as Facebook, Instagram, Dropbox, or a Wi-Fi-connected media server such as Plex. The Go's Casting functionality allows the headset's view to be streamed to the phone paired with the Go.

Hardware

The headset uses the Qualcomm Snapdragon 821 system on a chip, 3 GB of LPDDR4 RAM, and either 32 or 64 GB of internal storage. The integrated Adreno 530 GPU provides roughly 500 GFLOPS of graphics performance. The device is equipped with a 2600 mAh battery, which provides up to two hours of gaming or 2.5 hours of video playback according to Oculus. The Go has non-positional 3-degrees-of-freedom tracking, a proximity sensors for detecting when the headset is being worn, and a small controller that functions as a laser pointer in the virtual environment.

The headset is outfitted with a fast-switching 5.5-inch LCD with a 2560 x 1440 (1280 × 1440 pixels per eye) resolution in an RGB-stripe subpixel arrangement. The headset has a field of view of about 101 degrees, which gives it a horizontal pixel density of 12.67 pixels per degree. The headset doesn't feature physical interpupillary distance (IPD) adjustment and has a fixed lens distance of 63.5 mm, which according to Oculus best accommodates users with an IPD between 61.5 and 65.5 mm.

The Go is available in two storage configurations: a model with 32 GB of internal storage for US$149 or 64 GB of internal storage for US$199.

FIG: 3.38 OCULUS GO TOUCHPAD, TWO FRONT FACING BUTTONS AND A TRIGGER

The Oculus Go controller is a wireless, orientation-tracked remote controller with pointer capabilities that are used to interact with applications and games. In addition to a touchpad, the controller has three buttons that can be used to select things, go back to a previous screen or menu, and return to the Go's home screen. The controller is powered by a single AA battery and allows for either hand to be chosen to function through the Oculus Go.

HEADSET LIGHT INDICATIONS

Colour	**State**	**Indication**
Red	Solid	Low battery (less than 15%)
White	Solid	Screen is on
White	Blinking	Oculus Go is rebooting
Blue	Blinking	Connecting to app
Purple	Blinking	Factory reset
Orange	Solid	Headset is starting up
Headset is plugged in		
Green	Solid	Battery is charging (above 95%)
Orange	Solid	Battery is charging (below 95%)

TABLE: 3.1 HEADSET LIGHT INDICATIONS

EXPERIMENTAL SETUP : FLOW CHART

CATIA V5

⬇

PIXYZ STUDIO

⬇

UNITY 3D & VISUAL STUDIO

⬇

OCULUS GO

FIG: 3.39 EXPERIMENTAL SET UP FLOW CHART

METHODOLOGY

Objective:

To develop a VR application that gives complete learning, training, and finally self -evaluation of the student.

The progress of this project shows in three stages namely: 1. Design
2. Develop
3. Deploy

Tools used:

1. Catia v5 – Cad Design Software.
2. Pixyz Studio – Digital Content Creation (3D) Tool.
3. Unity3D – Game Development Platform.
4. Visual Studio – Integrated Development Environment.
5. Oculus Go – A standalone VR headset.

This application is in three modes namely: 1. Learning
2. Training
3. Self- Evaluation

1. DESIGN:

For any VR experience, we have to create an environment and game objects in the scene. For that, we create the engine parts in CATIA-v5 software. Using part design we create individual parts of the engine and using assembly design we make the whole engine assembly in CATIA format as shown in below figures.

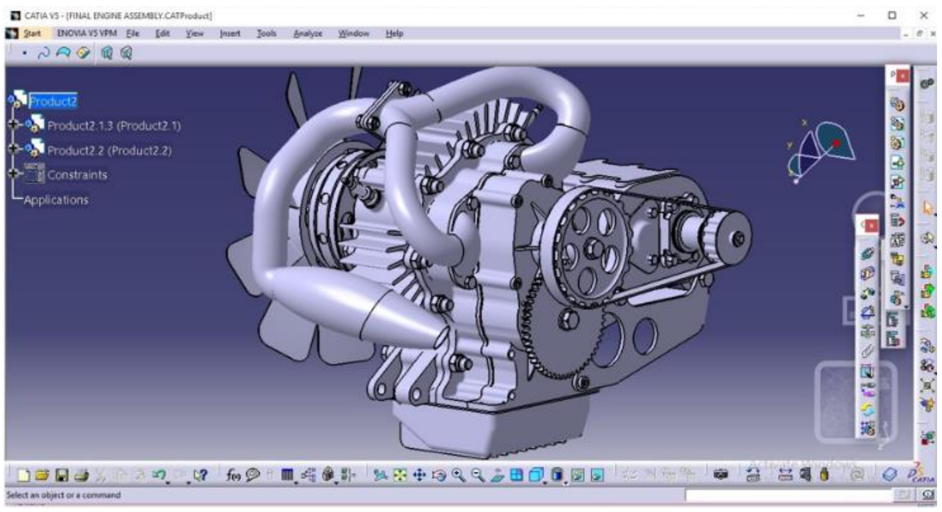

CATIA V5 - [FINAL ENGINE ASSEMBLY.CATProduct]
Start ENOVIA V5 VPM File Edit View Insert Tools Analyze Window Help
Product2
Product2.1.3 (Product2.1)
Product2.2 (Product2.2)
Constraints
Applications
Select an object or a command

FIG: 3.40 ENGINE ASSEMBLY

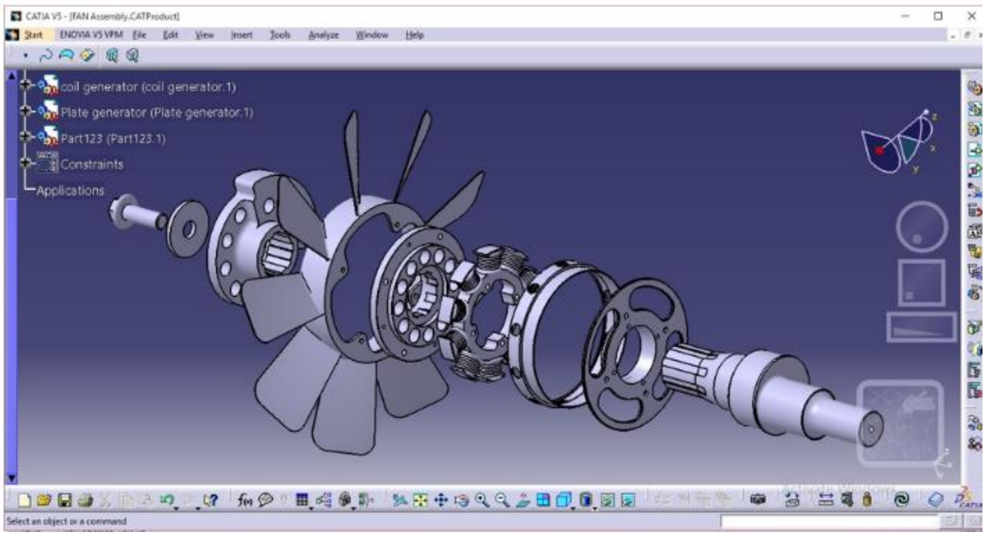

CATIA V5 - [FAN Assembly.CATProduct]
Start ENOVIA V5 VPM File Edit View Insert Tools Analyze Window Help
coil generator (coil generator.1)
Plate generator (Plate generator.1)
Part123 (Part123.1)
Constraints
Applications
Select an object or a command

FIG: 3.41 EXPLODED VIEW OF FAN SUB ASSEMBLY

3D DATA PREPARATION :

Objective: Obtain a high-quality tessellation from a native CAD file while respecting original geometry.

Pixyz Studio creates high quality and low-density meshes from almost any CAD model. It provides efficient and very fast tessellation.

STEP 1: CAD file import

Pixyz Studio supports both tessellated (mesh) and native CAD (exact geometry) formats.

To import your model :

- Go to the « File » menu and choose « Import Model » or press the following button from the Main Toolbar ;
- Select the file to import. Multiple files (several formats) can be imported at the same time.

STEP 2: Tessellation

CAD models are not tessellated, they contain exact mathematical surfaces. To create meshes or polygonal surfaces :

- Go to the « CAD » menu and choose the « Tessellate» function or click the button from the Essential toolbar

- Set the following parameters for a high-quality tessellation :

  Max Sag : 0 .2 mm

  Max Angle: 10
- Click on « Execute » (do not select anything or select the Root node to execute the algorithm on the whole scene)

STEP 3: Model Repair

PiXYZ STUDIO offers to mesh repair features, such as polygons reconnection, normals/faces orientation unification, topology correction.

The «Repair Mesh» function can automatically correct the defaults.

To repair your model:

- Go to the «Optimize mesh» menu and choose «Repair Mesh» option or press the following button from the Essential toolbar:
- Set 0.1 millimeter as tolerance value
- Click on «Execute» (do not select anything or select the Root node to execute the algorithm on the whole scene)

The «Orient» and «Invert Orientation» functions are also available, but they only correct orientation issues on selected surfaces.

STEP 4: Remove holes

Once tessellated and repaired, the original model contains around 1.2M triangles. This amount can be optimized. Pixyz can remove specific features on a CAD model, such as holes.

To reduce the triangles count, start by removing through holes with a diameter lower than 10mm

• Go to « Reduce mesh » menu and choose « Remove holes » option

• Check « Through Holes » option

• Set the Maximum Diameter to 10 millimeters

• Click on « Execute » (do not select anything or select the Root node to execute the algorithm on the whole scene)

STEP 5: Hidden Removal

The original model contains hidden patches*, parts*, and polygons which are not necessary for the visualization,

To remove automatically all hidden parts*, patches* or polygons not visible from a sphere around the scene:

• Go to the « Optimize » menu and choose « Hidden removal » function

• Choose to remove patches (for instance)

• Click on « Execute » (do not select anything or select the Root node to execute the algorithm on the whole scene)

You can choose to remove parts*, patches*, or polygons.

If a portion of a patch* or a part* is visible, this patch or this part won't be removed. For further reduction of the number of polygons, choose

« polygons ». Thus the portion of the patch*/part* not visible will be removed.

STEP 6: **Remove the link with CAD patches**

Before reducing the triangles count with the decimation* function, we need to delete the CAD patches* to allow the re-meshing overpatches*boundaries Go to « Optimize mesh » menu and choose « Delete Patches» function.

STEP 7: Decimation

For a lower-memory usage, the model needs to be transformed to a low-poly object. Pixyz Studio reduces the polygon density of a mesh by deleting vertices. It allows precise control, preserving normal distortion and texture coordinates.

To remove vertices:

• Go to the « Reduce mesh» menu and choose the « Decimate » function, or push on the following button from the Essential toolbar

• Use the default parameters

• Click on « Execute » (do not select anything or select the Root node to execute the algorithm on the whole scene)

STEP 5: Export the model

The model has been reduced from 1M to 76k triangles, almost 95 % reduction, and the visual aspect remains the same as the original. The model has been optimized for real-time usages, such as Virtual or Augmented Reality.

It can now be exported in one of the supported formats. To do so, go to the « File » menu
and choose « Export Model » or click on the following button to open the export window.

2. DEVELOP:

- In unity (version:2019.3.5f1) we created the required environment.

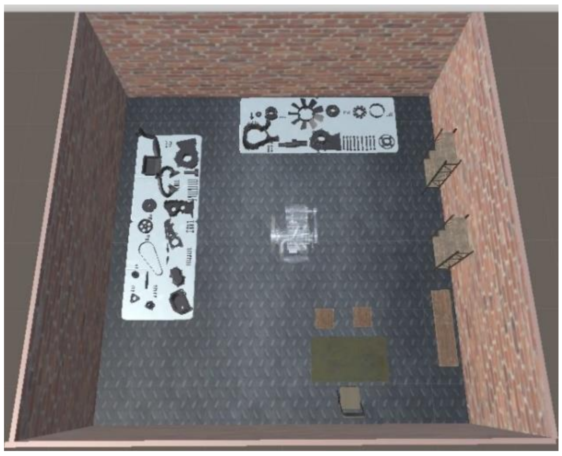


FIG: 3.42 TOP VIEW OF ENVIRONMENT

- In learning mode, it will be shown both the engine assembly and individual components of the engine. Whenever we hold the trigger in the controller on a game object it will select that individual part and resembles that in the main engine assembly and also is shows the description of that part. It will give a complete understanding of the construction of an engine. For this, we wrote a select script for this operation.

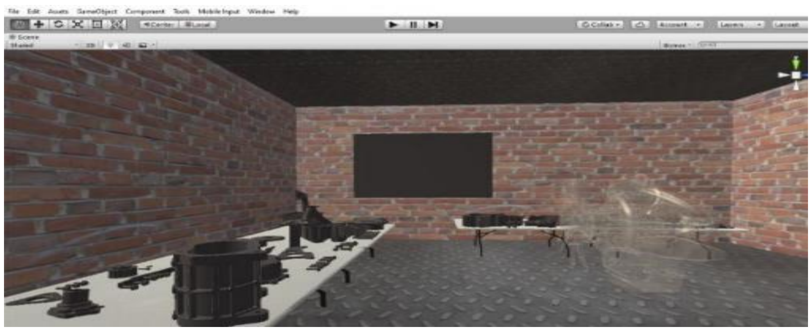


FIG: 3.43 INSIDE ENVIRONMENT

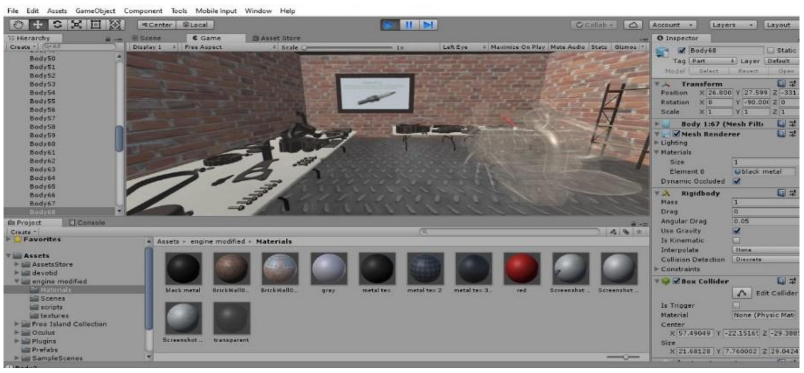


FIG: 3.44 INSIDE ENVIRONMENT WITH UNITY INTERFACE WINDOW

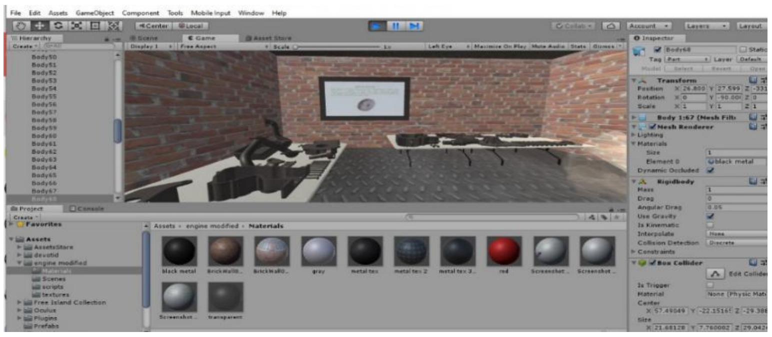


FIG: 3.45 DISPLAYING THE FUNCTION OF SELECTED PART

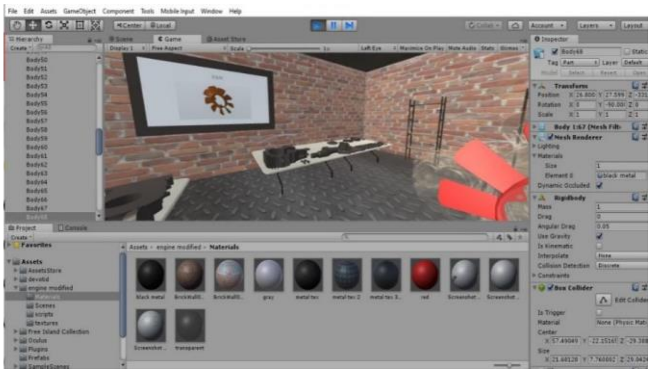
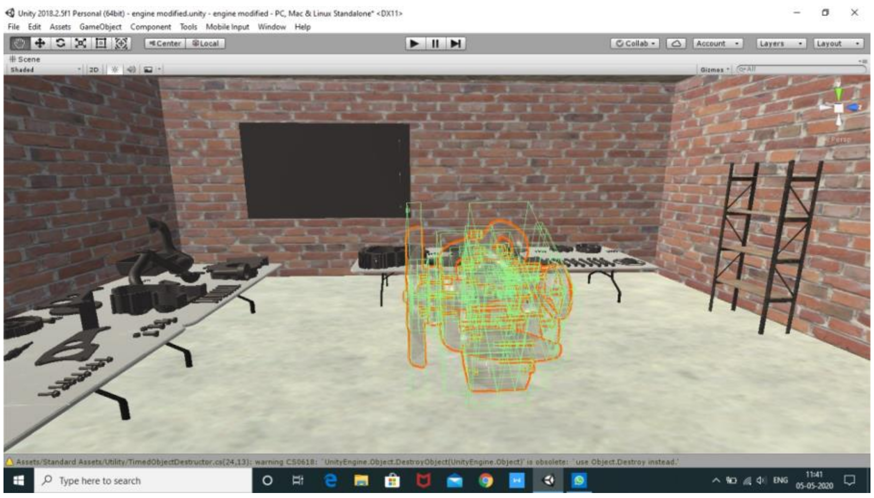


FIG: 3.46 SELECTED PART ON TRANSPARENT ENGINE WITH HIGHLIGHTED COLOUR

- In training mode, we changed the select script to drag-and-drop script. With this, whenever we hold the trigger on a game object it will select and drag the part from the respective position and dropped in the required position. Like this, all the parts should be placed in the correct position. This operation should be done sequentially as like in manual assembly operation by giving conditions in the script.


Unity 2018.2.5f1 Personal (64bit) - engine modified.unity - engine modified - PC, Mac & Linux Standalone* <DX11>
File Edit Assets GameObject Component Tools Mobile Input Window Help
Center
Local
Collab
Account
Layers
Layout
Scene
Shaded
2D
Gizmos
Type here to search
ENG
11:41
05-05-2020

```
Assets/Standard Assets/Utility/TimedObjectDestructor.cs(24,13): warning CS0618: `UnityEngine.Object.DestroyObject(UnityEngine.Object)' is obsolete: `use Object.Destroy instead.'
```

FIG: 3.47 SCENE VIEW IN TRAINING MODE

- In self-evaluation mode, we hide the main engine assembly, by the prior two modes student itself can assemble the engine. According to the number of parts assembled incorrect position can evaluate the student's performance and understanding of the concept. Finally, it will give a score, this will motivate the students to perform well in the examination

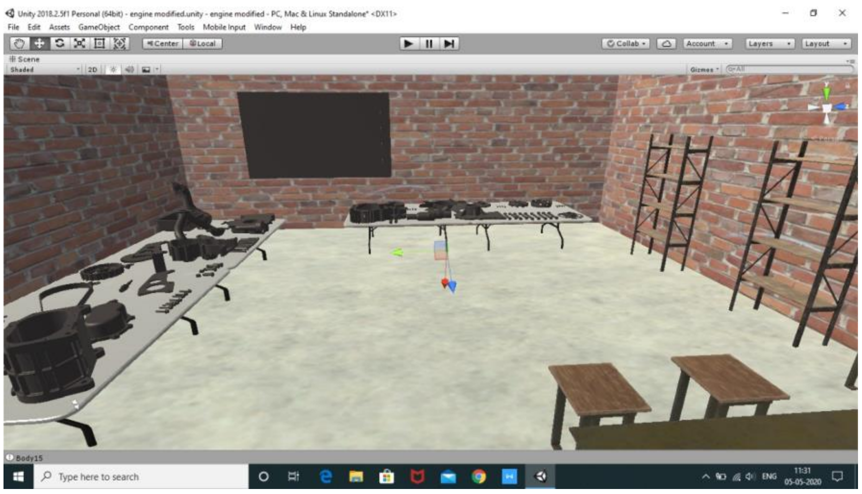

Unity 2018.2.5f1 Personal (64bit) - engine modified.unity - engine modified - PC, Mac & Linux Standalone* <DX11>
File Edit Assets GameObject Component Tools Mobile Input Window Help
Center
Local
Collab
Account
Layers
Layout
Scene
Shaded
2D
Gizmos
Body15
Type here to search
ENG
11:31
05-05-2020

FIG: 3.48 SCENE IN SELF-EVALUATION MODE (MAIN ENGINE ASSEMBLY IS HIDED)

VISUAL STUDIO
Scripts:

1. **Select Script:** Used for Learning Mode.

Select script

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;

public class Select : MonoBehaviour
{
    Renderer rend;
    Renderer rend1;
    Renderer rend2;
    Material orgMat;
    GameObject[] parts;
   // GameObject nameofpart;
    public Camera MainCamera;
    public Material[] mat;
    GameObject[] orgparts;
    GameObject[] slides;
    // Start is called before the first frame update
    void Start()
    {
        MainCamera = Camera.main;
        rend = GetComponent<Renderer>();
        //nameofpart = GameObject.FindGameObjectWithTag("PN");
        parts = GameObject.FindGameObjectsWithTag("Part");
        orgparts = GameObject.FindGameObjectsWithTag("orgpart");
        slides = GameObject.FindGameObjectsWithTag("slide");

    }

    // Update is called once per frame
    void Update()
    {
    }

    void OnMouseDown()
    {
        Debug.Log("M.Down");
        foreach (GameObject part in parts)
    {
        rend1 = part.GetComponent<Renderer>();
        rend1.enabled = true;
        rend1.sharedMaterial = mat[2];
        part.SetActive(true);
        Debug.Log(part.name);
    }

    Debug.Log(name);
    rend.sharedMaterial = mat[1];
        this.gameObject.SetActive(false);
    //nameofpart.GetComponent<TextMesh>().text = this.name;
    // Change color of original part
    foreach (GameObject S in slides)
    {
        if (S.name == this.name)
            S.SetActive(true);
       else
            S.SetActive(false);
    }
    // Change color of original part
    foreach (GameObject part in orgparts)
    {

        rend2 = part.GetComponent<Renderer>();
        rend2.enabled = true;

        if (part.name == this.name)
            rend2.sharedMaterial = mat[1];
        else
            rend2.sharedMaterial = mat[0];
    }
}

void OnMouseUp()
{
    }
```

2. **Drag and Drop Script:** Used for Training Mode.

DRAG AND DROP SCRIPT

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class pickandplace : MonoBehaviour
{
  public Transform theDest;
      void OnMouseDown()
   {
      GetComponent<BoxCollider>().enabled = false;
      GetComponent<Rigidbody>().useGravity = false;
      this.transform.position = theDest.position;
      this.transform.parent = GameObject.Find("Destination").transform;
   }
   void OnMouseUp()
   {
      this.transform.parent = null;
      GetComponent<Rigidbody>().useGravity = true;
      GetComponent<BoxCollider>().enabled = true;
   }
```

3. DEPLOY-OCULUS GO:

Set Up Oculus Go Device

1. Download the Oculus companion app for Android or iPhone and set up the user account.
2. Go to settings in your app and select the type of Oculus device you want to pair from the list.
3. Follow the on-screen instructions to pair the device and its controllers.
4. Follow the in-VR instructions to complete the generic device settings.

Project Settings

Project settings surface a variety of elements that drive the app's quality and performance. For example, it bundles audio, input, graphics, rendering, physics, XR support, and many more.

Add Product Details

You can add basic product details that uniquely identifies your company and the app in Unity.

1. From the menu, go to **Edit** > **Project Settings** > **Player Settings**.
2. In **Company Name**, type the name of your company.
3. In **Product Name**, type the name of your app or product that you want it to appear on the menu bar when your app is running.
4. In **Version**, type the version number that identifies the iteration. For subsequent iterations, the number must be greater than the previous version number.

Add Package Identification Details

When you build your app into the application package (APK), the Android build tools use the package name as the package attribute in the AndroidManifest.xml file. Instead of manually adding the package attribute and its value in the manifest file, Unity lets you add it from the package set.

1. From the menu, go to **Edit** > **Project Settings**, and select the Android tab.
2. Under **Identification**, do the following:

   a. In **Package Name**, enter a unique package name. It must be unique within the entire Oculus ecosystem, and it should be similar to the app's in-store title.

   b. For example, if your company's name is Jane Doe Inc. and the app's title is Move Along with Colors, the package name can be com.JaneDoeInc.colorgame.

   c. In **Version**, type the version number that identifies the iteration. For subsequent iterations, the number must be greater than the previous version number.

   If you've set the version number in product details, Unity automatically populates the version number.

   d. In **Bundle Version Code**, increment the existing version code. This version is used internally to determine whether one version is more recent than another, with higher numbers indicating more recent versions.

   d. In **Minimum API Level**, set the minimum Android version to Android 5.0 Lollipop (API level 21).

   e. In **Target API Level**, select **Automatic (highest installed)** to compile the app.

   f. In **Install Location**, select **Automatic**.

Enable Virtual Reality (VR) Support

To build a Virtual Reality (VR) app that provides the immersive and interactive user experience, you need to enable the VR support in Unity.

Depending on the Unity version you’re using, there are two ways you can enable the VR support:

a. Through the XR plug-in framework, which is introduced in Unity version 2019.3 and higher

b. Through built-in XR platform integration, which is the legacy method available in Unity version 2019.3 and prior.

With the Unity version 2019.3 and higher, Unity has introduced the XR plug-in framework, which allows Oculus to integrate with Unity's engine and make full use of its features. As a result, for these versions, the built-in XR platform integrations are marked as deprecated. However, for version 2019.3, both the methods are available and it's up to you to select the appropriate method.

Use XR Plugin Framework (only available in Unity versions 2019.3 and higher):

1. From the menu, go to **Edit** > **Project Settings**.
2. From the left navigation pane, select **XR Plugin Management**, and click **Install XR Plugin Management**.
3. Click the Android tab and select **Oculus** to install the Oculus XR plugin, which enables VR support.
4. From the left navigation pane, under **XR Plugin Management**, click **Oculus** to open and modify the default settings.

Use built-in XR platform integration:

1. From the menu, go to **Edit** > **Project Settings** > **Player Settings** > Android tab.
2. Expand **XR Settings** and select **Virtual Reality Supported**. Under Virtual Reality SDKs, you should be able to view Oculus.

Set Rendering Preferences

There are a variety of options and settings that let you optimize rendering. You can define a specific set of Graphics APIs, choose color space property, or enable multithreaded rendering to optimize performance.

1. From the menu, go to **Edit** > **Project Settings**, and select the Android tab.
2. Expand **Other Settings** and under **Rendering**, do the following:

For more information about selecting the correct color space for your project, Unity has outlined the gamma and linear color space workflow to understand the differences.

a. Set the Color Space property to Linear for realistic rendering. It lets colors supplied to shaders within your scene brighten linearly as light intensities increase.

b. Clear **Auto Graphics API** to manually pick and set the order in which the Graphics APIs are consumed. The Linear color space requires OpenGL ES 3.0 or Vulkan. Currently, Oculus Quest supports Vulkan but it is under the experimental phase.

Therefore, we highly recommend that you use OpenGL ES 3.0 and set that as the first Graphics API in the order. Oculus Go does not support Vulkan.

c. Select **Multithreaded Rendering** to move graphics API calls from the main thread to a separate worker thread. d. Select **Low Overhead Mode** to skip error checking in release versions of an app. This applies to apps that use OpenGL ES API.

Define Quality Settings

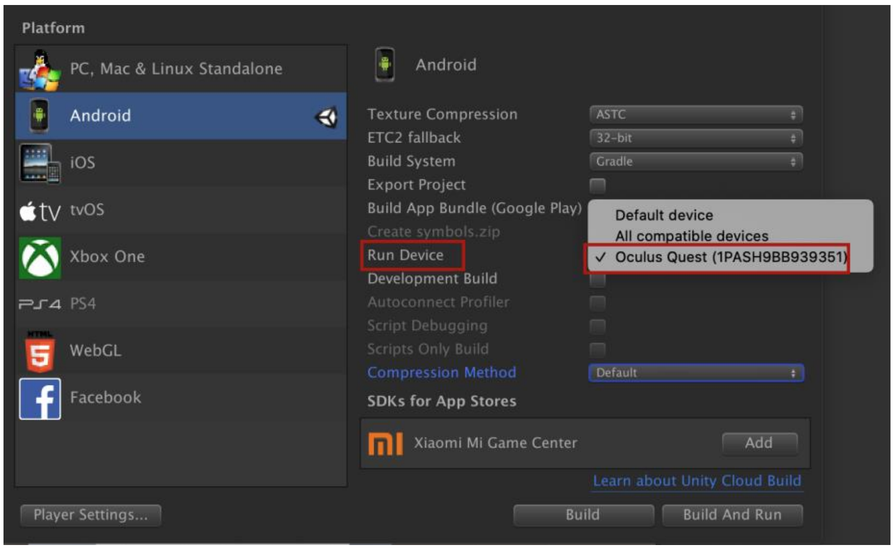
Several quality options let you define the graphical quality.

1. In the menu, go to **Edit** > **Project Settings** > **Quality**.
2. In **Pixel Light Count**, set the maximum number of pixel light count to one.
3. In **Texture Quality**, select **Full-Res** to display textures at maximum resolution.
4. In **Anisotropic Textures**, select **Per Texture**.
5. Clear the **Soft Particles** checkbox.
6. Select **Realtime Reflections Probes** to update reflection probes during gameplay.
7. Select **Billboards Face Camera** to force billboards to face the camera while rendering instead of the camera plane.

BUILD SETTINGS

You can run your app in the Oculus device to test its functionality, layout, and overall user experience. The Oculus device is connected to the computer over a USB cable. Before connecting the device and your computer, you need to set your device into developer mode.

1. Open the companion Oculus app on your phone and go to **Settings**.
2. From the list of paired Oculus devices, tap the device you want to set up and tap **More Settings** > **Developer Mode**.
3. Slide the toggle to turn on the developer mode.
4. Connect the device with your computer by using a USB-C cable.
5. Put on the device, and when prompted for permission to allow the connected computer to access the device, click **Allow**.
6. To verify the connection, open any Unity project that can build, go to **File** > **Build Settings**.
7. From the **Platform** list, select Android, and click the **Switch Platform**. If the target platform is already set to **Android**, skip to the next step.
8. In the **Run Device** list, you should be able to view the Oculus device.



Texture Compression	ASTC
ETC2 fallback	32-bit
Build System	Gradle
Export Project	
Build App Bundle (Google Play)	
Create symbols.zip	
Run Device	Oculus Quest (1PASH9BB939351)
Development Build	
Autoconnect Profiler	
Script Debugging	
Scripts Only Build	
Compression Method	Default

FIG: 3.47 BUILD SETTINGS WITH OCULUS CONNECTED

9. Click **Build And Run** to run the app in your Oculus device.

The scene loads in the connected Oculus device.

Finally, We can experience the VR environment for the assembly of machine components.

6. EXPERIMENTAL RESULTS

Practical in science and engineering areas require high technology, high-cost machines; and with the advancement in virtual reality, it is possible to have virtual labs in remote areas accessed sitting in the office or home of the learner. VR applications are also designed for training requirements like aviation and vehicle simulators, medical and military training.

Virtual reality can assist students to become immersed in a learning environment where they can participate in their learning in a technology-based environment.

Virtual reality is the basis for functional virtual prototyping, which enables engineers to analyze the shape, form, and functional behavior of products in an immersive and interactive virtual environment. Applying this technology greatly improves the communication in product design and product development: It helps to identify and avoid design errors in the early stages of the development process, it reduces the number of physical prototypes and saves time and cost. VR is considered a valuable tool for improving and accelerating product and process development in many industrial applications.

7. SCOPE FOR FUTURE WORK

- Virtual labs aim to provide remote-access to laboratories and to develop a complete learning management system where the students can avail of the various tools for learning, including additional web-resources, animated demonstrations, and self-evaluation.
- In Learning mode, we can integrate Augmented reality and make more interactive and allow learners to focus more on practice instead of just theory, by adding virtual objects to the environment.
- In Training mode, to provide more interaction we can add virtual hand gloves, tools (wrench, pullers etc.) which are used in conventional assembly tasks.
- In Self-evaluation mode, add options like giving score at the end of the complete assembly, which gives more game approach to the user.

8. CONCLUSION

The use of Immersive VR can offer great advantages for learning:

1. It allows a direct feeling of objects and events that are physically out of our reach.
2. It supports training in a safe environment avoiding potential real dangers and, thanks to the game approach, it increases the learner's involvement and motivation while widening the range of learning styles supported.
3. This VR based learning could help the learners to understand and memorize them better.

9. REFERENCES

1. Gilson Giraldi, R. S. (n.d.), “Introduction to virtual reality”,https://doi.org/10.1109/VR.2003.119117
2. Jean Thilmany, ”Walkabout in another world".(2000)
3. V. Kovalcík, J. Chmelík, M. Bezdeka, and J. Sochor, “Virtual Reality System as a Tool for Education”, 20th International Conference on Computer Graphics, Visualization and Computer Vision, p. 15-18, 2012.
4. Vafadar, M. (2013), "Virtual Reality : Opportunities and Challenges”, 3(2), 1139–1145.
5. Chen, C. J. (2006). The design, development, and evaluation of a virtual reality-based learning environment. Australasian Journal of Educational Technology, 22(1), 39-63
6. L. Ducan, A. Miller (2012). “ A taxonomy of virtual world usage in education”.
7. A.C. Boud, C. Baber, S.J. Steiner, “Virtual Reality: A Tool for Assembly”,[11].
8. Pantelidis, V. S. (1993). Virtual reality in the classroom. Educational Technology, 33(4), 23-27.
9. Michael Rorke (1997), “Virtual reality Interaction Techniques”, Computer Science Department Rhodes University.
10. Piovesan, Sandra Dutra; Passerino, Liliana Maria; Pereira, Adriana Soares [2000], “Virtual Reality as a Tool in the Education", International Association for Development of the Information Society.
11. Michael Rorke (2000), “Virtual reality Interaction Techniques”, Computer Science Department Rhodes University
12. Kluge, S., & Riley, L. (2008), “Teaching in Virtual Worlds: Opportunities and Challenges”, Issues in Informing Science & Information Technology, 5, 127–135
13. Hanson, K., & Shelton, B. E. (2008), “Design and development of virtual reality: Analysis of challenges faced by educators. Educational

Technology and Society", 11(1), 118–131. https://doi.org/10.2307/jeductechsoci.11.1.118 ISSN no -1436-4522

14. Jen Chen Chen, C. (2009), "Theoretical Bases for Using Virtual Reality in Education", Themes in Science and Technology Education Special Issue, Special Is, 71–90. Retrieved from https://files.eric.ed.gov/fulltext/EJ1131320.pdf
15. Gobbetti, E., &Scateni, R. (2010), "Virtual Reality: Past, Present, and Future", 1–31. https://doi.org/10.3233/978-1-60750-902-8-3
16. Bobbie Ticknor, Sherry Tillinghast (2011), "Virtual Reality and the Criminal Justice System: New Possibilities for Research, Training, and Rehabilitation: Vol 4, no 2.
17. Cocking, C. (2011), "Using VR in Learning and Teaching ", Introduction : Conceptual background to VR : The development of Immersive Virtual Reality (IVR): VLEs and Constructivist Learning : 7(1995), 113–124 ISSN no 1740-5106.
18. E Kiruba Nesamalar, G Ganesan. (2012), "An Introduction To Virtual Reality Techniques And Its Applications", International Journal of Computing Algorithm.
19. Wang J.(2012)," Research on Application of Virtual Reality Technology in Competitive Sports", International and electronics Engineering,p-3659-3662, ISSN no 1877-7058.
20. Nooriafshar, M., Williams, R., &Maraseni, T. N. (2014), "The use of virtual reality in education", The American Society of Business and Behavioral Sciences (ASBBS) 2004 Seventh Annual International Conference, (September).
21. Chris Christou. (2010), " Affective, interactive and cognitive methods for e-learning design: creating an optimal education experience".
22. Xue-qin Chang, Dao-hua Zhang, Xin-xin Jin. (2016), "Application of Virtual Reality Technology in Distance Learning": Vol 11, International Journal of Emerging Technologies in Learning (iJET) – eISSN: 1863-0383

23. Martín-Gutiérrez, J., Mora, C. E., Añorbe-Díaz, B.,& González-Marrero, A. (2016),"Virtual technologies trends in education", Eurasia Journal of Mathematics, Science and Technology Education, 13(2), 469–486. https://doi.org/10.12973/eurasia.2017.00626 ISSN No 1305-8223

24. Pantelić, A., & Vukovac, D. P. (2017), "The Development of Educational Augmented Reality Application: A Practical Approach",10th International Conference of Education, Research and Innovation, (November), 8745–8752.

25. Smutny, P., Babiuch, M., & Foltynek, P. (2019). A Review of the Virtual Reality Applications in Education and Training. 2019 20th International Carpathian Control Conference (ICCC). DOI:10.1109/carpathiancc.2019.8765930