*Ivanna Sandyk, Margus Müür, Vladimir Kuts, Yevhen Bondarenko, Tiia Rüütman*

*Vol. m, No.n, pp. a-b (to be filled by editorial desk)*

*Date of Publication: (to be filled by editorial desk)*

*DOI: (to be filled by editorial desk)*

*This paper can be cited as: (to be filled by editorial desk)*

*This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.*

**Pneumatics Laboratory Interactive Educational Experience Development**

**Ivanna Sandyk**

*Department of Mechanical and Industrial Engineering, Tallinn University of Technology, Tallinn, Estonia;* *ivanna.sandyk@taltech.ee*

**Margus Müür**

*Department of Mechanical and Industrial Engineering, Tallinn University of Technology, Tallinn, Estonia;* *margus.muur@taltech.ee*

**Vladimir Kuts**

*Department of Mechanical and Industrial Engineering, Tallinn University of Technology, Tallinn, Estonia;* *Department of Electronic and Computer Engineering, University of Limerick, Limerick, Ireland;* *vladimir.kuts@taltech.ee*

**Yevhen Bondarenko**

*Department of Mechanical and Industrial Engineering, Tallinn University of Technology, Tallinn, Estonia;* *yevhen.bondarenko@taltech.ee*

**Tiia Rüütman**

*Department of Mechanical and Industrial Engineering, Tallinn University of Technology, Tallinn, Estonia;* *tiia.ruutman@taltech.ee*

**Abstract**

*In the recent past, gamification has been promptly developing as well as actively applied in the educational sector. Nowadays, it figures prominently on account of making the learning process more engaging and motivating which leads to enhancement of the quality of gained knowledge. In addition to that, due to the relevance of contactless learning throughout the COVID-19 pandemic, the paper “Pneumatics Laboratory interactive educational experience development” is dedicated to the development of Laboratory works on Pneumatics. An essential aspect is the preservation of the quality of the existing level of education or, moreover, its strengthening. The redeployment of Laboratory works into the gamification experience format allows students to perform tasks without the necessity to attend the designated Laboratory room as well as reduce its utilization. The main objectives of the gamification experience are the establishment and contribution of the virtual tool into the Educational Game Project that is being developed by the Virtual and Augmented Reality Laboratory at Tallinn University of Technology.*

*The primary software tool implemented in the project is the Unity game engine due to its broad functionality while collaborating 2D and 3D environments.*

*Each Laboratory work requires schematic composition, namely, the selection of the necessary components from the library of symbols as well as drawing connections between them. In addition to that, the specific algorithm for ascertaining schematics for correctness is included in the project and directly when a schematic is assembled accurately, a student has an opportunity to observe a video tutorial. The project involves the Main menu environment along with 9 Laboratory works on Pneumatics with corresponding educational videos. By parity of reasoning the Laboratory works on Electro-Pneumatics, Hydraulics, and Electro-Hydraulics can be developed, which would enable students to comprehensively perform Laboratory works on the “Hydraulics and Pneumatics” course remotely.*

**Keywords**

Gamification experience, Pneumatics, Remote laboratory, Interactions.

**1. Introduction**

During recent years, gamification has been rapidly developing and applied in the educational process. The term “gamification” is generally used to denote the application of game mechanisms in non‐gaming environments with the aim of enhancing the processes enacted and the experience of those involved. In recent years, gamification has become a catchword throughout the fields of education and training, thanks to its perceived potential to make learning more motivating and engaging (Caponetto, Earp & Ott, 2014). In today’s digital generation gamification has become a popular tactic to encourage specific behaviours and increase motivation and engagement. Though commonly found in marketing strategies, it is now being implemented in many educational programs as well, helping educators find the balance between achieving their objectives and catering to evolving student needs (Mertala, 2019). As a result, the elements of novelty in performing learning tasks are among the most critical factors for this development (Shevtshenko et al., 2017).

The main objectives of the project are the creation and implementation of a virtual tool into the Educational Game Project in order to digitalize methods of learning Pneumatics. The project is being developed by the Virtual and Augmented Reality Laboratory at Tallinn University of Technology. In addition to that, the gamification experience is conducted in a 2D environment thus it is essential to develop and employ a specific algorithm for the Laboratory works execution. A fundamental aspect is collecting a library of symbols that represent Pneumatics components and elaborating the design of the gamification experience, which is inclusive of the “Main Menu” and 9 Laboratory works on Pneumatics. The task of each Laboratory work is to compose a Pneumatic schematic, namely, the selection of the necessary components and the construction of connections between them. The algorithm for performing and ascertain the schematic for the correctness of each Laboratory work is included in this work.

The gamification experience is conducted directly on individual Scenes that can be defined as assets that contain all or part of a game or application (Unity, 2021). Each Scene contains GameObjects, which represent various formats and perform predefined functions, e.g., interact with other objects within the Scene. Such operations can be performed by dint of the Event System. The Event System is a way of sending events to objects in the application based on input, be it keyboard, mouse, touch, or custom input. The Event System consists of a few components that work together to send events (Unity, 2021). In addition to that, GameObjects can represent colliders, which are used in this work to access the virtual Laboratory from the Educational Game Project. Colliders allow for the creation of events that occur in the case of an interaction with the object to which it is assigned.

The primary software tool selected for the project realization is the Unity game engine. Unity was chosen among similar in functionality platforms, e.g., Godot and Unreal Engine 4 due to the fact that Unity is competent while rendering 2D and 3D Scenes and superb in multiplatform gamification experience development. The game engine involves a wide range of possibilities and allows to accomplish the designated task entirely. The work consists of numerous C# scripts that are written using the Rider by JetBrains tool.

**2. Methodology**

The present study is designed to be implemented into the Educational Game Project conducted by the Virtual and Augmented Reality Laboratory at Tallinn University of Technology. The Educational Game Project is based on the digitalization of the educational process by dint of gamification experience development that represents the itemized model of Tallinn University of Technology. To accomplish the task the Unity platform is implemented. Unity is mainly used to create games and interactive 3D and 2D experiences, e.g., training simulation, medical and structural visualization (Li & Tang, 2019). The current study is inclusive of a 2D space creation that student accesses from the virtual Pneumatics Laboratory. The gamification experience provides an opportunity to conduct Pneumatics Laboratory works remotely leading to the elimination of the necessity for contact learning and utilization diminution of Pneumatics components. Pneumatics Laboratory tasks are transferred into the gamification experience and a student is presumed to construct a schematic in conjunction with a step diagram. In order to endow the virtual environment with an ergonomic interface, the gamification experience is inclusive of the “Main Menu” for Laboratory work selection and advancement observation.

The main objective of the study is to develop the gamification experience that permits a student to construct the schematics independently with an unlimited number of attempts. On the occasion of the gamification experience initiation, a user is suggested to select the required subject and subsequently the Laboratory work number. In addition to that, the algorithm that allows access to the Laboratory works exclusively after successful completion of the previous task is involved, implying the Laboratory works are obliged to be accomplished consistently. The transition between the Laboratory works is initiated automatically and immediately after the total amount of Laboratory works is completed, a student relocates to the “Main Menu” in order to either withdraw the gamification experience or revise Laboratory works. Representation of the gamification experience system can be seen in Figure 1.



**Figure 1:** *Gamification experience system diagram*

**2.1 Algorithm of Laboratory work execution**

In the event of a Laboratory work execution, a student observes a specific task, the Inventory of the Pneumatics symbols, empty space for the schematics construction as well as buttons that are responsible for connections removal, reset of the work to the initial state, reverse to the “Main Menu” and check the composed schematics for correctness. An illustration of the first Laboratory work’s initial state can be seen in Figure 2.



**Figure 2:** *Initial state of the first Laboratory work*

In order to achieve the main objective of the study, the sequence of actions must be determined and established into an algorithm of the Laboratory work completion that is illustrated in Figure 3.



**Figure 3:** *Laboratory work completion algorithm*

Following the current algorithm, a student is suggested to construct a schematic based on the selection and transferring appropriate components from the present Inventory of the Pneumatics symbols as well as draw corresponding connections. Immediately upon the schematics is composed and a student is prepared to ascertain it for correctness, the “Check” button is required to be pressed. In event of inaccuracy detection, the window with the corresponding notification appears on the screen and the student should continue with the construction. Directly when the schematic is built correctly, the prerecorded educational video of the detailed illustration of the process of construction and execution of the circuit comes into sight.

**2.2 Inventory of Pneumatics symbols**

In order to construct a schematic, the corresponding components selection from the presented Inventory of Pneumatics symbols is required to be performed. The Inventory is composed in a method that is inclusive of the components’ symbols that a student would have in a real Pneumatics Laboratory at Tallinn University of Technology. The illustration of the symbols was obtained from the educational materials on the MES0085 “Hydraulics and Pneumatics” course. To endow the symbols with the ability to be interactable and represent not an image but a UI object, the texture type of each component is substituted by “Sprite and UI”. The term “Sprite” is generally understood to mean a 2D Graphic object.

The Inventory is inclusive of single and double-acting cylinders, directional control valves, various control methods, flow control valves, filter, manometer, silencer, compressed air service unit, and pressure source. Every component represents a GameObject to which a C# script that is defining it as an “Entity” is attached.

By the dint of the C# script, it becomes possible to interact with each symbol subsequently, e.g., transfer it to a specially designated location for schematic composition. Since the composition is based on components connection, hence, each symbol has nodes that represent inputs and outputs.

The “Entity” script consists of numerous functions that will be cycled for subsequent interactions, but the fundamental aspect is storing the list of nodes that the component involves. Whereby the “Init” function which is called once the program is initiated, every component acquires the list of nodes which will be used further in order to join them and ascertain the connections for correctness.

Since every component represents a GameObject, therefore, it is inclusive of a child object. A child object can be loosely described as a GameObject which has a lookup field designating to the parent's object data. The child object of each “Entity” is titled “Nodes” and it has a corresponding C# script attached. Depending on the polarity type, the nodes can be distinguished by In, Out, or In-Out which creates a possibility for further deviation between the nodes in order to compose connections. E.g., In nodes of two different components cannot be connected due to the rule that input connects to output exclusively.

The “Node” script is designed in consequence in the Unity Inspector it is possible to assign each node to any polar type which in turn changes the title of the node. Considering the following logic, by the dint of the “Init” function, if the polarity type is selected as output, then the node is renamed to “Out Node” immediately. In addition to that, once the program is initiated, depending on the type of polarity for each “Entity” the list of nodes is created.

Every node contains a point that is defined as a control point for drawing a connection line. Representation of the connection line can be seen in Figure 4.



**Figure 4:** *Screen capture of the connection line*

The objective of each control point consists of aesthetic appearance improvement. In the present study, the points are titled “BezierControlPoint” taking as a basis Bezier curves that are used in computer graphics to draw shapes and each curve is defined by control points. The minimum amount of control point for one curve is 2 but in the case of components connection, it is set to be 4 namely output of first component, Bezier point of the first component, Bezier point of the second component, and input of the second component.

**2.3 Canvas administration and locomotion**

Construction of the connection lines takes place for drawing on the Canvas and cannot be performed without access to its administration. Canvas is a parent object which children objects are inclusive of the objects that are implemented to perform a Laboratory work including the task, Pneumatics components, and a designated space for the circuit composition. In addition to that, the “Manager” script is attached to the Canvas, by dint of which it becomes possible to accustom in the Unity Inspector the width of the line, its default color, and type. In this study, three types of lines are implicated: Spline, Z-Shape line, and regular line. In order to construct a schematic, the Z-Shape line type is preferred from among choices which is defined by 4 control points, two of which are Bezier Control Points that were mentioned earlier.

Interactions with objects on Canvas occur by the dint of mouse movements and specific commands. In order for a user to observe alterations during the gamification process, a hand pointer is implemented which represents an open hand icon in the default state and a hand with fingers folded when interacting with an object. To achieve the task, the GameObject is supplemented as a child object of the Canvas titled “Pointer” to which a C# script with the identical name is attached. By dint of the script attached to the “Pointer”, a hand pointer overlaps the original mouse icon and follows its movements comprehensively. The pointer moves depending on the Render Mode that is configured for the Canvas in the Unity Inspector. In the current study, there is a place to be merely two Render modes namely Screen Space Overlay and Screen Space Camera due to the fact that the gamification experience on Pneumatics Laboratory works is presented in a 2D space. Since camera movements are not involved in this case, the Render Mode of the Canvas is set to Screen Space Overlay.

For the Screen Space Overlay, it is enough to use the transform position declaration exclusively and equate it to the mouse position. As for the Screen Space Camera, it is essential to set the position of the pointer on the Z-axis since the mouse movement, in this case, depends on the position of the predefined camera. Z-axis value defines the distance of the plane from the camera that is equal to 10.0 Unity units. Onwards by parity of reasoning of the Screen Space Overlay, the transform position declaration is implemented which follows the “screenPoint” variable using the “ScreenToWorldPoint” function that transforms a point from screen space into world space, where world space is defined as the coordinate system (Unity, 2021).

**2.4 Drag and drop of the Pneumatics components**

The main objective of the Pneumatics Laboratory works is a schematic composition by dint of necessary components selection and connection. The component selection occurs from the aforementioned Inventory of symbols that subsequently being transferred to the designated place. The displacement of GameObjects on Canvas is not possible to be performed without drag and drop functions. In the current study, the method of transferring a component from one parent GameObject to another is implemented. The “Pointer” script involves a “sortingOrder” function that equals 999 leading to placing the pointer on top of all GameObjects meaning a component being dragged will also appear on top.

In order for a GameObject to be transferred, the first priority is to find and select it. The public “FindObjectCloserToPointer” function allows finding the object that is closest to the pointer using the list of ordered objects.

In addition to that, various C# scripts that involve an interface function exclusively are developed. Interface function can be broadly described as a representation of a complex of operations an object can perform without specific instructions. Through the agency of the “UI\_Clickable” function, it becomes possible to determine whether the GameObject is clickable or not.

For the present study, a computer mouse is a primary tool for performing all operations within a virtual environment. The “Pointer” script involves a public “KeyCode” variable “clickKey” meaning it is possible to adjust the click button in the Unity Inspector. As soon as the “clickKey” is pressed and the pointer is being relocated, the “OnDrag” function is activated which allows the GameObject to be grabbed.

In the case when the GameObject is selected and ready for transfer, the “OnPointerDown” function declared in the “Pointer” script is enabled. By dint of the function, two public UnityEvents “e\_OnPointerDownFirst” and “e\_OnPointerDownLast” are defined where First and Last mean the priority of the action on the event. Once the “clickKey” is pressed, the corresponding Events in the Event System are created, the cursor icon changes to “iconHold”, the pointer detects a component that can be grabbed and the cursor position transforms following the movements of the computer mouse.

The subsequent step is to discharge the GameObject at the position where the “clickKey” is released. The task is accomplished by the “OnPointerUp” function which is a part of the “Pointer” script. This function declares two public UnityEvents “e\_OnPointerUpFirst” and “e\_OnPointerUpLast” where First and Last by parity of reasoning mean the priority of the action on the event. On the occasion when the “clickKey” is released, the corresponding Events in the Event System are created, the pointer icon changes to “iconDefault”, and if the GameObject is a clickable object, it is released.

Releasing the GameObject means defining a new position for it on the Canvas. The function “InstantiateEntityAtPosition” includes the vector and access to the GameObject of the “Entity” type. The function supports creating new coordinate values for the “Entity” that is subject to transfer.

**2.5 Composition of nodes connection**

The sequential step in schematic construction is to produce connections between Pneumatics components. In order to accomplish the task, the “UI Connect” Asset is purchased from the Unity Asset Store and applied as a pattern. Connections are created in the following consistency: first, a line emerging from the first node is drawn, then the node with which the user desires to create a connection is selected and the connection line is being created, which exists as a separate GameObject. The construction originates in a way that the user places the cursor over the necessary node, presses and holds the left mouse button until a connection line occurs. The “FollowMouse” function that is declared in the C# script attached to the “Pointer” GameObject returns the location of the cursor, which is additionally used to compose connections. As described earlier, line appearance is adjustable in the Unity Inspector, since variables, e.g., line type, width, and color are declared as public units. Since the Z-shape line type is applied, the connection line is defined by 4 points which are the output of the first component, two Bezier control points of the first and the second components, as well as the input of the second component. In the “DrawLine” function, 4 position variables of type Vector2 are declared, denoting a vector that is defined by two points on the X and Y-axis. In addition to that, the method of drawing lines by calculating the sine and cosine is implemented. Positions of the connection lines are determined by generating new X and Y-axis values by adding or subtracting the corresponding cosine and sine values.

The definition of the input of the second component occurs using the method of the program automatic detection of the closest located node to the mouse position. In the case when the program discovers components of type “Entity” that resides in the Entity List, the list of nodes is read and stored. Furthermore, if the node has no existing connections and represents the opposite polarity type, the input point of the second component is attached to the given node that is nearest to the mouse position at the given time. An illustration of automatic detection of the closest node can be seen in Figure 5. The left side of the illustration exemplifies that the present node cannot be selected as an input of the second component due to polarity type incompatibility. On the opposite side, it may be observed the method the program automatically determines the nearest node of the opposite polarity type to the output of the first component.



**Figure 5:** *Screen capture of automatic closest node detection*

The conclusive stage of the connection construction occurs the moment the program has identified the closest node and the user releases the left mouse button. Once two nodes are selected and a connection line is composed, it is stored in the list of existing connections, which holds the information regarding the reference to nodes of every component. Moreover, the connection line represents a GameObject that is created by nodes connection and appended to the list of objects that belong to the Scene.

**2.6 Destruction of connection lines**

In the present study, the ability to remove connection lines, if necessary, is involved. In order to acquire the task, a UI button titled “DeleteButton” is created, which is located in the designated place for schematic construction. The “ButtonRemoveSelected” C# script is attached to the button that endows it with the ability to perform an operation. The “Action” function that is activated at the moment of pressing the button with the left mouse button is inclusive of a “for” loop. The loop accomplishes the task of selecting a connection line, which is a GameObject and removing it handling a Destroy declaration. The object is destroyed immediately after the current Update loop, or t seconds from now if a time is specified. If the object is a GameObject, it destroys the GameObject, all its components, and all transform children of the GameObject (Unity, 2021). In addition to that, once a GameObject is eliminated, the “ContextMenu” is refurbished namely the list of existing connections.

**2.7 Resetting the accomplished progress**

In a designated place for schematic composition on the right side of the “DeleteButton”, the UI button is situated titled “ResetButton” for discarding the progress accomplished in schematic composition. The button is developed to allow a student to reverse the circuit to its original state at the moment the button is activated, instead of manually destructing connection lines and returning every component to its initial position. The method selected for resetting the process is reloading the active Scene. At the moment the program executes, the Scene variable is read by virtue of the SceneManager applying the “GetActiveScene” function that stores a currently active Scene. Furthermore, a separate “Reset” function is developed, which is inclusive of a single declaration namely a LoadScene function defined by two variables. The declaration allows loading the Scene by the name acquired initially. It should also be indicated that the SceneManager is able to interact exclusively with Scenes that are arranged in the Unity Build Settings. Build Settings intend for occasions when more than one loaded Scene exists, in order to switch between them and adjust the sequence.

The following stage is the configuration of the On Click event of the button. In this case, the action executes exclusively on the Runtime and the operation is performed based on the “ResetButton” script. Therefore, at the moment of pressing the “ResetButton”, the “Reset” function is initiated, which serves to reload the Scene.

**2.8 Predefining correct schematic composition**

Each Laboratory work implies the fulfillment of the assigned task and in the case of correct assembly of the circuit, the transition to the subsequent task is performed. In order to create a list of nodes to be connected, the “SetupInitialConnectionsEditor” script is developed. Initially, the size of the list is selected, which cannot exceed 50 items. The list of connections is designed in a way that it stores the identification number of each connection, as well as information about the correspondence of components’ nodes that are connected. Immediately after a connection line is drawn, a corresponding input is created for the node that represents the output.

Taking as an example the first Laboratory work on Pneumatics, in which, in order to achieve the assigned task, it is essential to carry out 6 connection lines in total.

**2.9 Verification of the diagram assembly**

As mentioned above, checking the schematic for correctness is performed by attaching and configuring the “SetupInitialConnections” script. In this study, the script is attached to the UI (User Interface) button titled “CheckButton”. In addition to that, two supplementary GameObjects titled “TryAgain” and “Correct” were created previously. The “TryAgain” GameObject is the child object of the “Canvas Manager” that represents a pop-up window in case of incorrect completion of the assigned task. The “Correct” GameObject is a separate parent object in order to endow it with the ability to not be obscured on the screen by other GameObjects. It involves a prerecorded educational video that can be observed by the student in order to understand how the circuit is assembled in a real Laboratory.

The “SetupInitialConnections” script is inclusive of a Boolean “CheckCorrectConnections” function denoting that the function ultimately returns a Boolean parameter that can be broadly defined as either a true or false statement. Primarily, the integer “connCount” variable is declared, which represents the total number of required connections. If “connCount” is not equal to the number of drawn connections, then the circuit is considered to be composed incorrectly. However, if the total number coincides, the revision for the identity of two lists occurs, the first of which is the list of correct connections, and the second is the list of connections created by the student. Otherwise, if the right connections are not connected, by parity of reasoning, the schematic is considered to be assembled incorrectly.

Additionally, the “SetupInitialConnections” script includes the “CheckConnections” function that allows to activate and deactivate necessary events. The function is inclusive of two outcomes. The first one exists once the schematic is assembled correctly, and the “Canvas Manager” is deactivated as well as the educational video appears. Vice versa, once the schematic is assembled incorrectly, the “Canvas Manager” remains active, but a “TryAgain” window appears.

**2.10 Pneumatics Laboratory work selection**

Once all Pneumatics Laboratory works are developed pertaining to a separate Scene, it becomes necessary to create an algorithm for accessing them. Reasonably, the Laboratory works should be streamlined, and since the level of complexity increases afterward, it is rational to supply access to subsequent Laboratory work exclusively in case of successful completion of the previous Laboratory work. The “PneumaticsMenuSelection” window, which is the child object of the main Canvas, contains 9 numbered buttons that provide access to the corresponding Laboratory work.

In order to initially reject access to the Laboratory works that are supervenient to the first one, it is essential to define them as not interactable. This determines if this component will accept input. When it is set to false interaction is disabled and the transition state will be set to the disabled state (Unity, 2021). The priority is to implement all the necessary Scenes including the “Main Menu” and Laboratory works to Build Settings. In the next stage, the “LevelSelection” C# script is developed and attached to the “PneumaticsMenuSelection”. The script is inclusive of a public array variable of type Button declaration, the size of which is adjustable in the Unity Inspector. Once the program is initiated, the number of the levels to which the student has reached is read. In case the index number of a Scene is not available, subsequent buttons are set not interactable.

**2.11 Accessing the laboratory**

Numerous games deliberately use the Raycast approach in order to execute interactions with GameObjects. In broad terms, "Raycasting" can be defined as a method of releasing from a specific point an invisible for user ray, e.g., main camera, to detect intersections with colliders.

The Raycast function's first parameter contains the origin position of the ray, while the second parameter corresponds to the path of the ray. The vector stores information about an interaction with a collider. In addition to that, the function involves two optional parameters: distance and layer mask. The distance variable represents the ray's length, while the layer mask specifies how many specific layers in the Unity Layer System can be prevented by the ray in the event of a collision.

Raycasting with a camera is possible to accomplish by virtue of a C# script attached to the Main Camera and is in charge of controlling the Raycast. Initially, the boundaries are defined by the minimum and maximum values for Raycast motions on the X and Y-axis that are declared as public float values, allowing them to be calibrated in the Unity Inspector eliminating the prerequisite to access the script. Furthermore, it is essential to take into consideration the sensitivity of both X and Y-axis displacements in order to achieve a smooth image for a user.

The access to the Laboratory may be accomplished by dint of interaction of the avatar with a specified GameObject. In this study due to the absence of a 3D model of the Pneumatics and Hydraulics Laboratory, the interactable GameObject is temporarily arranged to be a Personal Computer in the Virtual and Augmented Reality Laboratory. Representation of Raycasting when detecting an interactable GameObject can be seen in Figure 6.



**Figure 6:** *Raycasting once detecting an interactable GameObject*

The PC is endowed with an Interactable Tag titled “InteractiveObject” denoting that the avatar is able to collaborate with the PC that is inclusive of the Interactable Tag exclusively. The Raycast is represented by a white dot that is located exactly in the center of the Game view that turns red in case of encountering an interactable PC, indicating it is possible to perform the predefined action while the “Open Laboratory” inscription appears at the bottom giving a prompt to the user.

**3. Results**

For the purpose of flaws and faults determination as well as agenda development for future advancement, a questionnaire was compiled. Students have performed the Laboratory works in the form of gamification experience as a part of the MES0085 “Hydraulics and Pneumatics” course. Consequently, the total number of responses is equal to 37, yet 43.2% of respondents have never experienced the usage of platforms similar to Unity. The results of the questionnaire revealed that 10 participants could not pass the 5th Laboratory work and 5 students accomplished all 9 Laboratory works. The prime obstacle was the algorithm of circuit composition, namely in current version the sequence of components’ connection lines is essential, and a Laboratory work is not marked correct if the schematic is not assembled in ascending order.

The main privileges mentioned by the students were as follows.

* opportunity to execute Laboratory works remotely particularly during pandemic time;
* instant ascertainment of the schematic correctness;
* link of theory with practice by dint of combination of the exercises and the educational videos;
* ability to walk around the virtual university;
* smaller size and better performance in comparison with other platforms;
* detailed visualization and intuitive interface for the educational purposes.

General level of satisfaction on variety of aspects is presented in Figure 7.



**Figure 7:** *Raycasting once detecting an interactable GameObject*

The subject of the future development is related mainly to the algorithm of a circuit verification which affects such aspects as ease of use, overall reliability as well as performance. In the final analysis, the interface serves as great preparation before actual laboratory and the gamification experience is overall professional and helpful.

**4. Limitations of software and Future development**

Currently, the gamification experience is inclusive of one shortcoming which is verification of a schematic. Generally, the functionality of the gamification experience may be expanded, e.g., on the occasion of hovering over a component located in the Inventory, its title may be displayed, which would make it possible to remember the designations of the components more effectively. In addition to that, prompts about what exactly is not correct on the schematic would give the student more insight into the progress. Such prompts may represent notifications that either the components are not selected correctly, or the connection lines are not drawn correctly. As an additional feature, it is possible to add navigation to the educational video that would endow the student with the ability to delve deeper into the subject and be more aware of the circuit composition in the contact Laboratory on Pneumatics and Hydraulics. Furthermore, by parity of reasoning the Laboratory works on Hydraulics can be developed, which would enable students to completely perform Laboratory works on the MES0085 “Hydraulics and Pneumatics” course remotely.

Henceforth, the Laboratory work on Hydraulics should be included in the project, following the modified and finalized algorithm. Additionally, the preservation of the existing progress should be saved at each withdraw from the gamification experience in order not to mislay the progress obtained.

**5. Summary**

The present work was designed to transfer the educational process, namely, the conduct of Laboratory works on Pneumatics, into the form of a virtual gamification experience. To accomplish this task, the work is integrated into the Educational Game Project developed by the Virtual and Augmented Reality Laboratory at Tallinn University of Technology. Originally, the gamification experience was inclusive of a 3D model of the University and a customizable avatar that could move around in a virtual environment. The model of the Pneumatics and Hydraulics Laboratory has not yet been built, thus a student can acquire access to the Laboratory works in the Virtual and Augmented Reality Laboratory.

On the occasion of entering the virtual environment, which represents a 2D gamification experience, a user observes the “Main Menu”, which consists of subject selection between Pneumatics and Hydraulics, as well as a return to the 3D Laboratory. At the time of subject selection, a list of numbered Laboratory works appears, access to which opens exclusively in case the previous task is completed successfully. At the opening of the Laboratory work, the student observes the task in text format on the top of the screen, the Inventory of Pneumatics symbols, and the designated place for the schematic composition. The objective is to select the necessary components, transfer them to the schematic and build connection lines. For building connection lines, the “UI Connect” Asset is purchased from the Unity Asset Store and applied as a pattern. The connection line represents a Z-shape line, which is determined by 4 control points in order to improve the appearance of the circuit, and each connection line stores information about the nodes between which the connection is created. Laboratory works are developed hereby two UI buttons are located in the left corner of the schematic that allows to destruct connection lines or revert the work to its initial state. In addition to that, two buttons are placed at the bottom, one of which allows the withdrawal of the gamification experience, while the second is utilized to ascertain the composed schematic for correctness. The verification algorithm is that for each Laboratory work, a list of correct connections is initially created in the format of connected nodes, and once each connection is built, the student will be able to observe the prerecorded educational video. The educational videos are inclusive of an illustration of the components that are necessary to assemble a circuit as well as the process of composition and execution. These videos are implemented thereby the student cannot merely compose a theoretical schematic, but also study the construction process in a real Laboratory. In case the schematic is not assembled correctly, the user observes a window with the corresponding information. Upon successful completion of the designated task, access to the subsequent Laboratory work acquires and the student can revise the progress accomplished.

**Acknowledgment**

This project was supported by the next project: Practicing Mechanical Engineering Online (PRAMECO) under the Grant Agreement n° 2020-1-DE01-KA226-HE-005760 funded by Erasmus+ Programme.

**REFERENCES**

Caponetto, I., Earp, J., & Ott, M. (2014). Gamification and education: A literature review. *Proceedings of the European Conference on Games-Based Learning*, 1(October), 50–57. (Caponetto, Earp & Ott, 2014)

Li, C., & Tang, B. (2019). Research on the Application of AR Technology Based on Unity3D in Education. *Journal of Physics*: Conference Series, 1168(3). (Li & Tang, 2019)

Mertala, P. (2019). Wonder children and victimizing parents–preservice early childhood teachers’ beliefs about children and technology at home. *Early Child Development and Care*, 189(3), 392–404. (Mertala, 2019)

Shevtshenko, E., Karaulova, T., Igavens, M., Strods, G., Tandzegolskiene, I., Tutlys, V., et al. Dissemination of engineering education at schools and its adjustment to needs of enterprises. In *Proceedings of the 28th DAAAM International Symposium on Intelligent Manufacturing and Automation, November 8–11, 2017, Zadar, Croatia. DAAAM International, Vienna*, 2017, 44–53. (Shevtshenko et al., 2017)

Unity. (2021). *Camera.ScreenToWorldPoint.*

 <https://docs.unity3d.com/ScriptReference/Camera.ScreenToWorldPoint.html>.

Unity. (2021). *Event System.*

 [https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/EventSystem.html](https://docs.unity3d.com/Packages/com.unity.ugui%401.0/manual/EventSystem.html).

Unity. (2021). *Object.Destroy.*

 <https://docs.unity3d.com/ScriptReference/Object.Destroy.html>.

Unity. (2021). *Scenes.*

 <https://docs.unity3d.com/Manual/CreatingScenes.html>.

Unity. (2021). *Selectable Base Class.*

[https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/script-Selectable.html](https://docs.unity3d.com/Packages/com.unity.ugui%401.0/manual/script-Selectable.html).