LeARning—An AR Approach

Mihai Albu, Scott Helm, Thanh-Dat Cao, George Miltenburg(✉), and Angelo Cosco(✉)

Mohawk College, 135 Fennell Ave W. Hamilton, Hamilton ON L9C 0E5, Canada
{mihai.albu, scott.helm, thanh-dat.cao, george.miltenburg, angelo.cosco}@mohawkcollege.ca

Abstract. A remote, instruction-based approach allows the students and teachers to interact in an AR environment that does not require their physical presence in the classroom, nor the physical presence of an object of study. In many laboratory environments, students are present in the same physical location and are instructed to interact with real devices. The aim of this research is to create a general development software tool that allows importing 3D models of laboratory devices; set up specific script actions that students must perform; and allow AR 3D interaction and communication with the instructor. The system allows one-to-one or one-to-many AR communication that will eliminate the barriers of audio calls or manual video-instructed actions. The application has a natural extension in any industrial environment that requires remote assistance with visual interpretation of problems and solutions.

Keywords: Augmented reality · Distance-based learning

1 Introduction

With a general agreement that online learning is an advantageous method over in-class teaching [1], experiential learning needs additional resources. Typical apprentice training will also require in-class presence and exposure to possible failures [2]. Moreover, many experimental lab activities cannot be practiced at home, not even talking about having virtual assistance at all. Many debates propose to make it easier for apprentices to access their in-class learning, with a standard wait time for training, online training options, and support for apprentices in rural and remote communities. Studies showed that augmented reality (AR) can benefit from mental models, spatial cognition, and social constructivist learning theories to improve learning outcome [3]. AR is described as "the fusion of any digital information with physical world settings, being able to augment one's immediate surroundings" [4].

Hands-on augmented reality (AR) classroom experiences are rare. Current technological advances in AR provide massive opportunities for distance-based teaching with hands-on experimental learning. Various AR tools allow loading augmented reality objects into the view scene and interactively manipulating them. Referring to AR learning, [5] described pointed that "there is evidence that specific skills can be improved, that learners were motivated and challenged through the interactive

M. E. Auer and T. Tsiatsos (eds.), *Mobile Technologies and Applications for the Internet of Things*, Advances in Intelligent Systems and Computing 909,
https://doi.org/10.1007/978-3-030-11434-3_18

problem-solving activities and that the technology offered many opportunities for collaboration". While AR viewers range from basic mobile phones that can load AR applications, to smart glasses systems, to advanced AR devices that allow user interaction, the applicability in-class activities are still reduced. The AR implementation of a class lesson benefits from the lack of physical presence but still includes student–teacher interaction, with the "gaming" manipulation of objects belonging to the lesson without physical danger, plus the correct estimation of a student's performance. Taking advantage of the Unity3D scripting environment, projects can be implemented to allow manipulation of augmented objects that are part of a laboratory/lesson, check that all interactions were performed correctly, and finally display and report the results of the work. On the same approach, audio instructions can be recorded and offered as responses to user interactions. One significant advantage of AR learning is the possibility of "learning by mistakes" (Fig. 1).

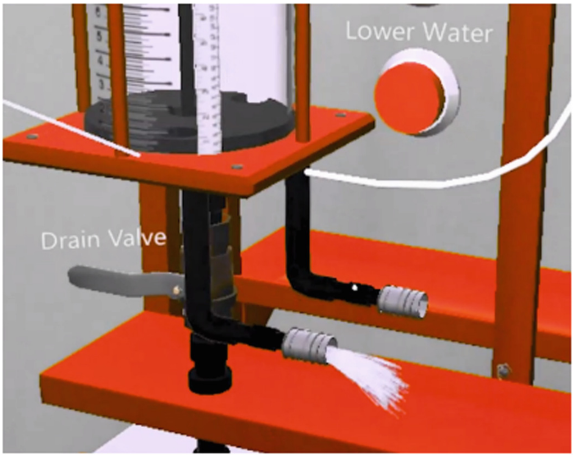

Fig. 1. Simulated failures in a lab experiment.

A student can try as many times to perform the tasks, without the fear of failing, which will increase their confidence and effectiveness. Moreover, several laboratories or lessons may result in dangerous or life-threatening situations. An AR simulation will avoid those, and may even offer examples of such cases to showcase what could happen in negative circumstances.

2 Approach

Electrical Engineers and Instrumentation Technicians study Instrumentation and Control in several courses and in many labs. In a specific laboratory experiment, students are required to follow step-by-step instructions to complete a laboratory experiment, where they may be assisted by an instructor. This project uses Microsoft's HoloLens AR development kit and implements the interactivity with Unity3D, to replace the traditional lab with an AR version. The system requires the import of a 3D model, together with a list of actions to perform in the form of a text-based script. This allows for a multifunctional, interchangeable, and universal approach that can be applied to any 3D model and teaching experiment (Fig. 2).

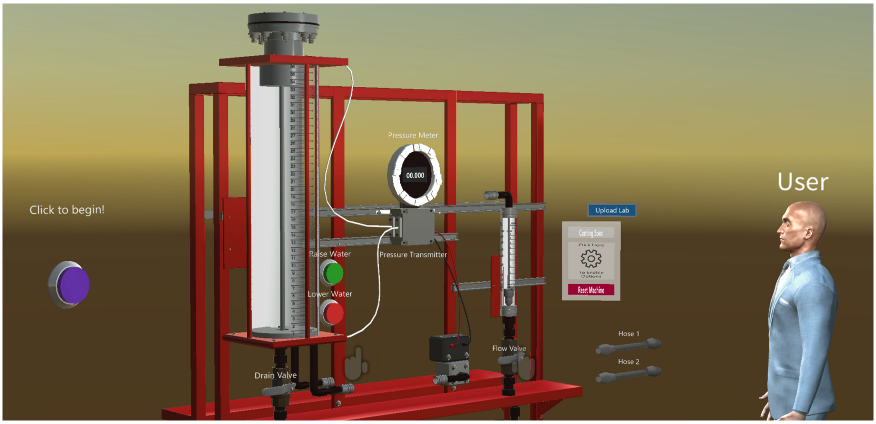

Fig. 2. Typical experimental AR view

This approach allows not only script-based teaching, but also for remote advising while students perform the laboratory assignments, thereby allowing for distance education. The student can be away from the school, and/or the professor can be away from the class. Users set up the application for students to perform. They may request AI or remote assistance as instructions for the completion of the project. Student interaction allows click/press, click/drag/connect, click/record measurements, and visualize results. Typical mistakes are flagged by the system automatically. Reports are generated upon completion of the experiments, allowing for easier examination of the results. Remote assistance allows for "physical" pointing to parts of the laboratory device. In addition, the students receive audio advice on specific actions to perform.

3 Methodology

The basic approach follows the below pattern of actions:

1. Loading model

The model is loaded as the scene is entered (Fig. 3), but how things are displayed deviates at this point. If working in the sandbox, the model appears right as the scene has finished loading. Given that this was our test environment, where the user's height is not being accounted for, we approximated a default height for rendering the model.

The second path involves spatial perception from within Microsoft's own Mixed Reality Tool Kit; their scripts are intended to help developers make use of the HoloLens. However, we found their tutorials cumbersome and often quite outdated, leading to the creation of custom object placement scripts. Using their default settings for spatial perception/understanding, a custom placement script was written to ensure that game objects were rendered at a specified height relative to the floor, so that it would be consistent every time (in retrospect, it may have been better to render it as a percentage distance from the floor to the user's eye line rather than static numbers). Once the user's environment has been scanned by the device, one of two things occurs. We either match the sandbox's default height (if the environment was deemed unsuitable, or was not scanned enough), or every object shifts into its new suitable position and becomes visible to the user at this point.

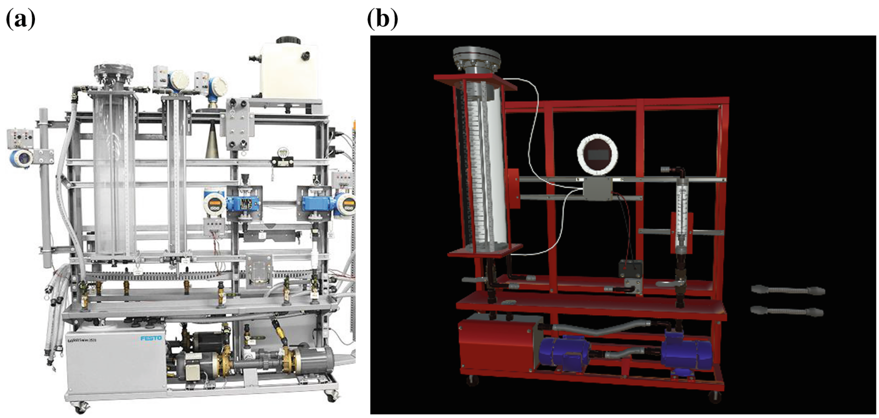

Fig. 3. **a** LabVolt 353x series. **b** 3D model

2. Interactions

We created code that would control which part of the machine could advance through a set of instructions once the current active goal was met. Housed in an array, each instruction was its own object. When the step was advanced, information was pulled from the target object at the current point in the array: text for the user with an objective, and a target object to enable if required for that stage of the experiment. These instructions were generated from a text file (Table 1) that was read through a script in the editor that created a specific "asset" resource on demand. This asset would then be loaded depending on the required experiment to be run (Fig. 4).

Table 1. Script-based elements

Action	Element	Command
Connect pipes	Pipe 1	Drag and drop pipe 1 to low left hose
Connect pipes	Pipe 2	Drag and drop pipe 1 to up right hose
Turn on valve	Valve	Click and rotate left valve 1

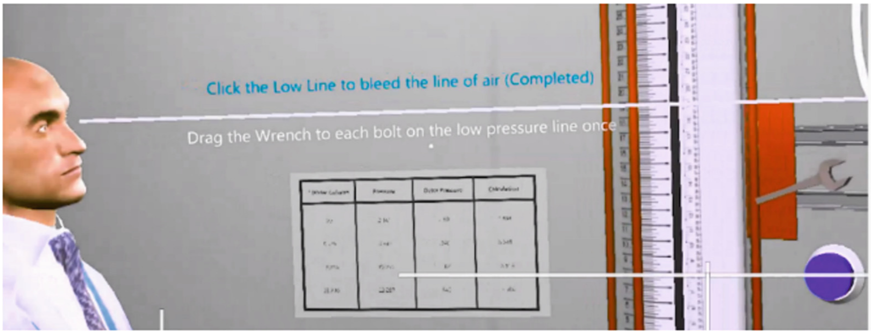

Fig. 4. Instruction-based activities from a script

In the future, this would allow for flexibility where a series of experiments could be generated for the same machine while being housed within one scene. Data is recorded, analyzed, and reported back to the user (Fig. 5), highlighting errors or successful completion of the laboratory task.

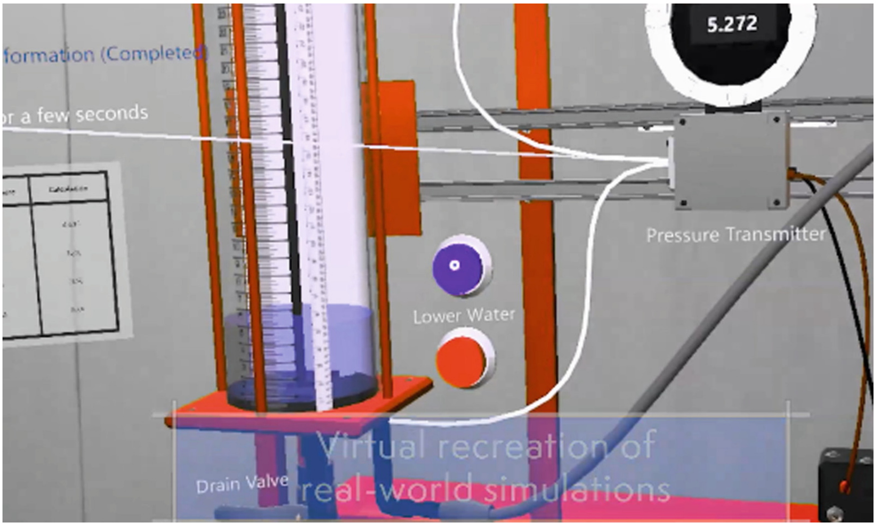

Fig. 5. Measurements and reports of lab experiments

Each object that the user could interact with for the experiment had (at minimum) two scripts on them.

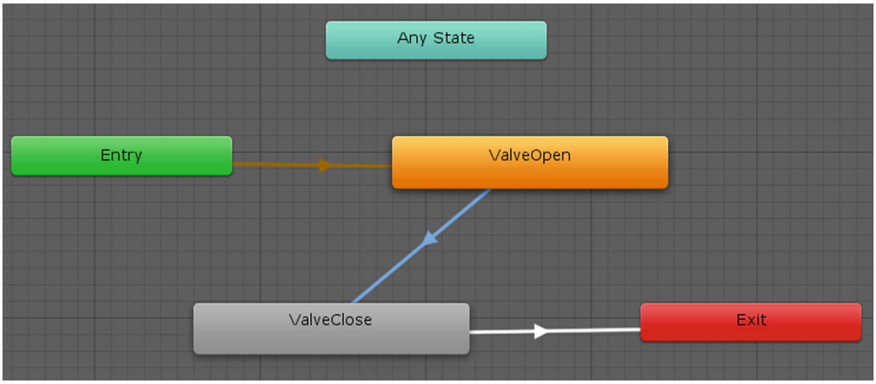

Fig. 6. Animation pane for valve opening

One user altered the color of the part in the question and played a sound when the correct object was interacted with. The other user was responsible for feeding information back to the manager who handled which instruction the user was currently working on, updating it with a "(Correct)" beside it while also advancing one step further in the list, or an "(Incorrect)" mark that kept the user in their current stage (Fig. 6, Sample code 1).

```
 /// Activate light source and change the col-
 our of the material when the pointer enters
/// the trigger.
public void Enter()
{
    targetMat.color = newMatColour;
    targetMat.EnableKeyword("_EMISSION");
    HoverSounds(); // play the audio clip once
    StartCoroutine(HideText());
}
```

Sample code 1. Allow color interaction to object

We wanted failure states to play an important part of the lab process, both because it made the experience feel more "game-like" but also because we believe making mistakes is an important part of learning—and our choice of technology allows the user to fail in a way that results in no serious consequences.

The scene is designed to be adjustable/scalable for easier user manipulation and to accommodate various interaction styles (Sample code 2).

```
public void AdjustInteractiveSize(float size)
  {
     Vector3 sizeVector = new Vector3(size, size, 1);
    foreach (GameObject thing in exampleManipulable)
    {
       thing.transform.localScale = sizeVector;
     }
    canChange = true;
  }
```

Sample code 2. Resize objects

3. Networking

One of the primary goals of this project is allowing students and instructors to connect to the same lab. This will aid the learning process, where students can work on their unique lab instance, and instructors can switch between labs, providing assistance to the students as needed. As the result, the Water Lab application supports networking capability, which enables connectivity between users.

The networking backend follows a client–server model. The server infrastructure is comprised of a Windows server running Sharing Service from Microsoft's Mixed Reality Toolkit. A server can host many sessions, each of which may contain one or more rooms. The lobby functionality allows clients to select and hop around sessions and rooms, in addition to creating their own room. Once connected to the server, each running instance of Water Lab is registered as a client. In term of scalability, real-life testing shows that a server can support up to five active clients at a time.

In the context of this project, each room can be considered an instance of the lab. When a client creates a new room, a new lab instance is initialized. Each lab instance contains data about the state of that lab, such as water level and valve position. Those data are distributed to all clients in that room to ensure that they all interact with a synchronized lab (Sample code 3).

```
public virtual XString GetRemoteAddress() {
     global::System.IntPtr cPtr = SharingClient-
  PINVOKE.NetworkConnection_GetRemoteAddress(swigCPtr);
     XString ret = (cPtr == global::System.IntPtr.Zero) ? null : new
  XString(cPtr, true);
    return ret;
  }
```

Sample code 3. Retrieving remote address procedure

To reduce the load on the server and prevent conflict errors among users, changes made by the user must be submitted manually, via a designated button in the lab. The lifecycle of a room ends when the last client (not necessarily the host) exits the room.

To simulate a real-life collaboration experience, Water Lab assigns every remote client a 3D model as their avatar. Visually, the 3D model shows the client's name, their position and their line of sight (in the form of a white line starting at their head). Based on that, each client in the room has the following data: position of the client within the room, head rotation and cursor position (i.e., where the client is looking).

These data are created when a client joins or creates a room, along with the avatar. While the application is running, data are sent to the server, which processes, and subsequently distributes them to other users. Once it receives the data, Water Lab will update the 3D model of the respective client to reflect the changes. Upon leaving the room, the avatar and all related data to the user are discarded. Since the process happens in real time and continuously, everything is seamless from the user's perspective.

4 Conclusions and Future Work

The implementation of the system allows for remote experimental teaching of laboratory material. The project clearly eliminates any laboratory experiment-based risks while providing a "close to reality" experimental approach to learning. In addition, it allows for "gamification of learning," which is learning by making mistakes, but doing so in a safe, controlled manner.

The final goal of this research is to create a system that allows any laboratory experiment to be implemented in augmented reality in a user-friendly environment.

Acknowledgements. The research was supported by Mohawk College School of Engineering Technology and the Marshall School of Skilled Trades and Apprenticeship.

References

1. Reuter, R. (2009). Online versus in the classroom: Student success in a hands-on lab class. *American Journal of Distance Education, 23*(3), 151–162.
2. Cosco, A. (2018). Mohawk college pilots innovative instructional technology. The Canadian Apprenticeship.
3. Cheng, K.-H., & Tsai, C.-C. (2013). Affordances of augmented reality in science learning: Suggestions for future research. *Journal of Science Education and Technology, 22*(4), 449–462.
4. Fitzgerald, E., Ferguson, R., Adams, A., Gaved, M., Mor, Y., & Thomas, R. (2013). Augmented reality and mobile learning: The state of the art. *International Journal of Mobile and Blended Learning, 5*(4).
5. Luckin, R. (2011). Limitless or pointless? An evaluation of augmented reality technology in the school and home. *International Journal of Technology Enhanced Learning, 3*(5).