Generalized Structure for Adaptable Immersive Learning Environments

Erik Coltey*, Yudong Tao†, Tianyi Wang*, Shahin Vassigh‡, Shu-Ching Chen*, Mei-Ling Shyu†
*Knight Foundation School of Computing and Information Sciences
Florida International University
Miami, FL, United States
†Department of Electrical and Computer Engineering
University of Miami
Coral Gables, FL, United States
‡College of Communication, Architecture and the Arts
Florida International University
Miami, FL, United States
ecolt003@cs.fiu.edu, yxt128@miami.edu, wtian002@cs.fiu.edu, svassigh@fiu.edu, chens@cs.fiu.edu, shyu@miami.edu

***Abstract*—Immersive Learning Environments (ILEs) developed in Virtual and Augmented Reality (VR/AR) are a novel professional training platform. An ILE can facilitate an Adaptive Learning System (ALS), which has proven beneficial to the learning process. However, there is no existing AI-ready ILE that facilitates collecting multimedia multimodal data from the environment and users for training AI models, nor allows for the learning contents and complex learning process to be dynamically adapted by an ALS. This paper proposes a novel multimedia system in VR/AR to dynamically build ILEs for a wide range of use-cases, based on a description language for the generalizable ILE structure. It will detail users' paths and conditions for completing learning activities, and a content adaptation algorithm to update the ILE at runtime. Human and AI systems can customize the environment based on user learning metrics. Results show that this framework is efficient and low-overhead, suggesting a path to simplifying and democratizing the ILE development without introducing bloat.**

***Index Terms*—virtual reality, augmented reality, content generation, immersive learning, 3D environments**

I. Introduction

Some of the first compelling applications of VR/AR hardware involve training users to perform tasks. Given VR/AR's spatial nature, task-driven training offers a realistic and valuable alternative to traditional teaching by providing a secure and low-risk learning environment [1]–[4]. The increase in available stimuli within an ILE is significant, as perception, sound, light, and mobility can play a critical role in the learning process [5]. ILEs also provide greater interactivity compared to traditional content delivery methods. When combined with high-fidelity environments, they can create psychological experiences described as a sense of presence, shown to improve learning [6].

Despite the positive aspects of ILEs, their development presents three challenges: (1) Creating an environment for immersive learning is very complicated. Most existing methodologies consist of one-off environments with hard-coded rules, leading to lengthy development times, usually not transferable to other ILEs in a different domain. (2) Users need to be evaluated on their actions in an environment in a structured way. With so many possible assets and environments to use, it is challenging to account for every case. Most 3D engines are precompiled, so changing the conditions for events to occur dynamically is not immediately possible. (3) Implementing future advancements in ILEs could further increase the development time. For example, utilizing AI techniques for education is promising, but their practical implementations require an advanced infrastructure.

To address these challenges, in this paper, we propose a novel multimedia system for learning in an immersive environment, creating ILEs and dynamically adapt their content. The main contributions of this work are the following:

- A unified and extensible description language for learning processes used for immersive applications in various domains.
- A dynamic ILE adaption system based on the description language which updates the content in the immersive environment in runtime. Several test cases were implemented to validate that the system can achieve real-time performance for content adaption. An optimization method to balance the latency of updating dynamic content and user's experience. The method aims to update as many contents in the ILE as possible while outside the user's view.
- Integration with a multimodal data collection module. With this, the proposed ILE can seamlessly interface with a broad range of AI systems, collecting data for model training and manipulating the environment with AI-based decision-making.

The rest of the paper is organized as follows. Section II discusses the related work in ILEs, adaptive learning, and relevant educational theories. In Section III, the proposed framework for creating ILEs is presented, including the description language of the learning process and a dynamic content adaption framework based on the proposed description

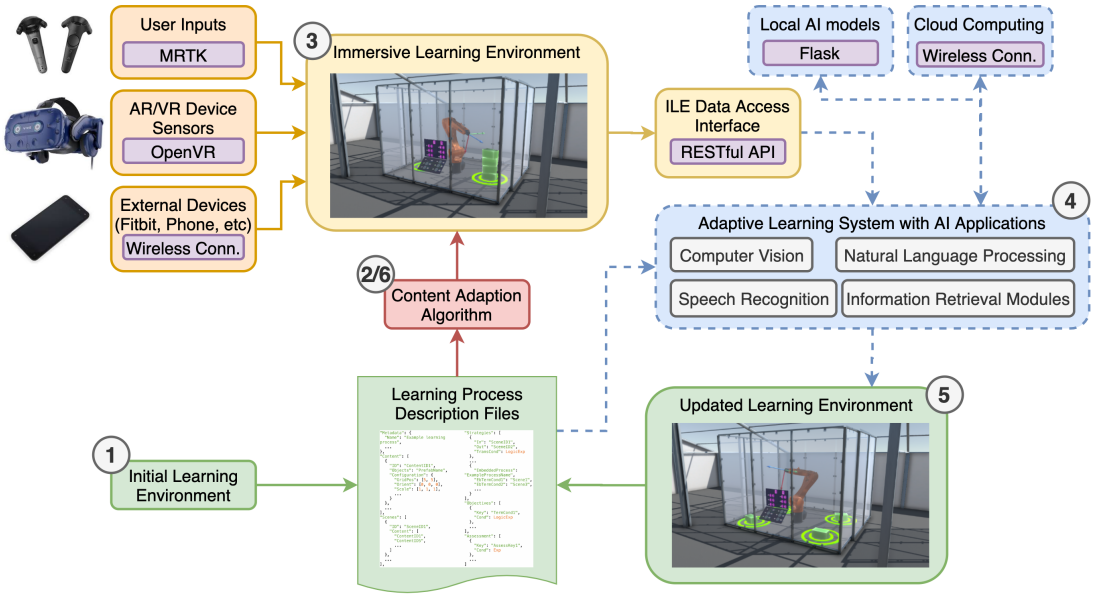


Fig. 1: System architecture of the proposed multimedia system for ILEs. Solid lines refer to the data flow in the proposed multimedia system and dashed lines refer to the data flow involving the external ALS.

language. Section IV explores some experiments with three different use cases and the system performance evaluation. Finally, Section V summarizes our contributions and discusses future work in this domain.

II. Related Work

A. Immersive Learning Environments

Many industries find VR/AR training very valuable for their employees, and several schools have begun to introduce VR/AR into their curriculum. In [2], the authors have designed a VR application for engineering coursework and found that realistic and interactive ILEs can enhance teaching and learning effectiveness. Even a primarily static VR environment, such as in [3], has led to improved retention compared to 2D content measured after a few days. Software used to train nurses in [4] was effective, showing that such technology could improve learner confidence in practical skills essential for operational tasks. Such applications can also help in non-verbal communication, with [7] showing novice music conductors quickly improve their skills within an ILE.

Although these ILEs are successful in training within their specific use-case, they are not generalizable to other domains. To address the issue, a few generalizable learning systems have been proposed. For example, MASCARET defines learning content using Unified Modeling Language (UML) and describes how ILEs should be created [8]. Furthermore, a dynamic learning environment, called VRLE, was proposed [9], which integrates content meta-modeling and an intelligent tutoring system (ITS) to manipulate the UML files created with MASCARET. However, these dynamic learning systems are not developed for VR/AR, and user behavioral data are not utilized to influence learning in the system.

B. Adaptive Learning

It is imperative for ILEs to adapt the content to user's needs as they complete tasks. Content adaptation provides an efficient, practical, and customized learning experience, where the Adaptive Learning System (ALS) is incorporated to adjust the learning content based on the user's objectives, preferences, and knowledge [10]. For example, quality of service (QoS) metrics were used to generate content in an ILE which account for cognitive learning styles [11]. User's emotions and experiences in games have shown to be important [12], and the environment's properties can be changed to maximize users' desired behaviors. It has been shown that learning is most effective when the learners are aware of their weaknesses [13], so meaningfully updating the environment requires that an ALS understands the environment and determines how the next learning iteration should be adjusted. As a result, a generalizable ILE needs to provide access to all the data regarding the learning environment and user behaviors to the ALS to obtain feedback. Moreover, most current ALSs are based on manually-crafted rules and do not take advantage of the recent advances in data-driven Artificial Intelligence (AI) techniques.

C. Educational Underpinning

ILEs provide interactivity, that when combined with efficient representational fidelity, create a psychological experience described as a sense of presence, which has shown to improve learning [6]. The increase in stimuli is significant as perception, sound, light, and mobility, typical to ILEs, are considered critical in learning theories such as the Dunn and Dunn Learning-Style Model [5]. Another well-established learning theory is the Kolb Experiential Learning Cycle model, which describes learning as a cycle with four stages. These

include concrete experience, reflective observation, abstract conceptualization, and active experimentation [14]. According to Kolb, different types of learners enter the cycle in different ways, making it essential for an ILE to accommodate users entering the cycle at different points. It is also imperative for ILEs to adapt the content to users' needs as they complete tasks. Other theories such as Metacognitive Theory state that learning is most effective when learners are aware of their weaknesses [13]. Deliberate Practice theory states that when learners know their weaknesses, they can better address their specific challenges [15]. Therefore, meaningfully updating the environment using instructor or AI feedback is another significant step to improve learning in ILEs.

III. Proposed Multimedia System

To enable the interaction between ILEs and ALSs and facilitate building the AI models for these applications, it is critical to develop a general framework to adapt the ILE dynamically and monitor the environment in real-time. This is critical for preparing the data collected from the environment and user for the AI applications [16]–[18]. Built on this idea, our proposal for a novel multimedia system for immersive learning is shown in Figure 1. This system can create ILEs based on a structured description, adapt contents in the ILE at runtime based on an extensible and unified description language of the learning processes. It will also collect both event-based and time-series data from the ILE and the user. The workflow of the proposed multimedia system to create ILEs is: (1) the initial learning environment is provided based on the proposed learning process description language; (2) the content adaption algorithm will load it and create the ILE; (3) the user can interact with the ILE in VR/AR while the user and environment data are collected in real-time; (4) the external ALS and AI applications will then request the data via the ILE data access interface using Restful API and process them using local AI model shipped by Flask library and cloud computing infrastructure; (5) when it is appropriate, the description of newly updated learning environment will be generated by ALS; and (6) the dynamic content adaption algorithm will periodically check the description files for updates, and the changes of contents in the learning environment will be adjusted accordingly with minimum influence to the users. The system ends when the user completes the learning process in the ILE and exits the system. In addition to addressing the identified contents in creating ILEs, our proposed framework bridges the gap between data-driven AI applications and VR/AR technologies, which is expected to accelerate the research and development of novel AI techniques for immersive learning.

A. Learning Process Description Language

To meaningfully structure an ILE, it is necessary to create a learning process that is essentially a description of the ordering of the learning content, different tasks that the users must complete, and the conditions for their success. There are four primary components to a learning process: learning content, learning objectives, assessment, and instructional strategies [19]. These components provide flexibility when defining an ILE, as they are domain-agnostic. Therefore, we formulate a learning process by defining each of its primary components as a series of elements in the immersive environment and define a learning process description language using the JSON format [20].

1) Learning Content: Learning content is the set of course materials that will help users gain the necessary knowledge and skills during the learning process. In the ILE, learning content is embodied by virtual objects, how they are placed in the environment and how they are expected to interact with the user. Thus, we formally define the learning content in an ILE as a list of triplets $C = [c_1, \ldots, c_{N_C}]$, where $N_C > 0$ is the number of models in the environment, $c_i = (c_i^o, c_i^c, c_i^s)$ is the i-th virtual object, c_i^o is its model, c_i^c is its configuration (e.g. location, orientation), and c_i^s is its behavior description defined by a script associated with the object. Specifically, c_1 is used to represent the user in the ILE since the user always has a virtual presence in the VR/AR environment.

2) Learning Objective: Any learning process should have objectives, which determine success in the ILE. This can be extended with various termination conditions (TCs) to provide more flexibility, which is formally defined by $O = [(O_1^{key}, O_1^e), \ldots, (O_{N_O}^{key}, O_{N_O}^e)]$, where $N_O > 0$ is the number of TCs, O_j^{key} is the key/name of the j-th TC, and O_j^e is its corresponding condition statement which is a logical expression based on any combinations of configurations of learning content c_i^c in the environment. To avoid confusion among TCs, the keys should be unique, i.e., $\forall j_1, j_2 \in [1, N_O]$, $j_1 \neq j_2$, $O_{j_1}^{key} \neq O_{j_2}^{key}$. System-level metrics, such as process time duration, can also be used in learning objectives as well.

3) Assessment: In the ILE, the assessment is determined via quantitative metrics calculated based on the user's performance during the learning process. The assessment A can thus be defined as $A = [(A_1^{key}, A_1^e), \ldots, (A_{N_A}^{key}, A_{N_A}^e)]$, where $N_A \geq 0$ is the number of assessments, A_p^{key} is the key/name of the p-th assessment defined as the same way of the learning objective, and A_p^e is an arbitrary expression based on any combinations of configurations of learning content c_i^c in the environment. Specifically, when $N_A = 0$, there is no assessment required in the learning process and A is an empty list. Unlike the learning objectives, assessments do not determine if the learning process ends or not, as they only evaluate the user's status after the learning process.

4) Instructional Strategy: Instructional strategy defines the sequence and organization of the learning content for the user to achieve the objectives [19]. Within a learning process, a subset of learning content in C can form a scene $s = \{c_{i_1}, c_{i_2}, \ldots, c_{i_{N_s}}\}$, where N_s is the number of learning content in this scene, and $i_k \in [1, N_C]$. A learning process might contain various scenes $S = [s_1, s_2, \ldots, s_{N_S}]$ designed by instructors, where N_S is the number of scenes in the learning process. The transition conditions from one scene to another $t(s_{k_1}, s_{k_2}), k_1 \neq k_2$ can be defined as a logical expression based on any combinations of configurations of learning content in the scene. Therefore, the instructional

strategy, IS, can be formally defined as a directed graph without self-loops.

$$IS = G(S, T) \tag{1}$$

S is the set of scenes that form the graph's vertices, and T is the set of transition conditions of the learning process that form the graph's edges. Furthermore, since the transition conditions are very similar to the learning objectives and to facilitate the modularization of the learning process, a scene can reuse any existing learning process defined within the system. So, the definition of the instructional strategy should be extended as a directed hypergraph, where a hyper-edge can link more than two vertices. For example, let the scenes in the embedded learning process be $S' = [s'_1, s'_2, \ldots, s'_{N_S}]$ and one of its termination conditions be $O_1^{e\prime}$. If this termination condition leads to scene s_1 in the main learning process, the hyper-edge can be defined as $t(S', s_1) = O_1^{e\prime}$. Both vertices of the edge can be contained in the embedded learning processes. Based on the definition of the instructional strategy, a well-defined instructional strategy should satisfy all the following requirements. This ensures the integrity of the learning process descriptions, which is important to avoid system crashing caused by partially updated or incorrect descriptions when the learning process descriptions are dynamically updated. Note: $deg_+(s)$ and $deg_-(s)$ refer to the indegree and outdegree of vertex s, respectively.

- **One Starting Scene Requirement**: There must be no more than one node in IS with zero indegree, i.e.,

$$\begin{aligned} &\forall k_1, k_2 \in [1, N_S], \nexists k_1 \neq k_2, \\ &deg_+(s_{k_1}) = deg_+(s_{k_2}) = 0 \end{aligned} \tag{2}$$

- **No Deadlock Scene Requirement**: For all scenes with zero outdegree in IS, there must exist at least one combination of configurations of learning content that satisfies the termination condition, i.e.,

$$\begin{aligned} &\forall s \in S, deg_-(s) = 0, \exists c^c_{i_1}, \ldots, c^c_{i_{N_s}}, \\ &\exists j \in [1, N_O], O^e_j(c^c_{i_1}, \ldots, c^c_{i_{N_s}}) = \text{True} \end{aligned} \tag{3}$$

- **Process Reuse Completeness**: For all embedded learning processes, each learning objective defined in the process should be linked to a succeeding scene and the embedded learning processes should also satisfy **Process Reuse Completeness**.

5) Learning Process: Based on the formulation of learning content, learning objective, assessment, and instructional strategy, the learning process can be defined as a triplet, $P = (G, O, A)$. For the proposed learning processes to be lightweight, portable, and easily understood, the JSON format was chosen due to its human readability and ubiquitous support in most modern programming languages. The directed hypergraph structure is represented using multiple JSON files. Additionally, since the model and scripts of a virtual object can form a prefabricated (prefab) object in a 3D engine such as Unity, the scripts are ignored, assumed to be included in the prefab. This allows flexibility with simulated objects spawned using a JSON string reference. The collections of prefabs can be stored as packages to be added to the ILE and loaded at runtime. Such prefabs should mimic physical objects, much like supplies in a classroom. An example description of the learning process is shown in Figure 2, where the Metadata section is used to include information such as the process name. The first scene is the initial scene to initialize the learning process, and a unique ID is assigned to each content and scene for future referencing.

Fig. 2: Example of a potential JSON file defining an ILE hypergraph.

B. Dynamic Content Adaption

ALSs have been broadly used in professional education to improve training effectiveness within an ILE [11], [12]. In the Kolb Experiential Learning Cycle [14], it is important that an ILE can allow users with different learning styles to enter at their preferred point in the learning cycle. It is also desired for the ILE to integrate ALSs, and the learning content can be adapted to the user's performance and learning style dynamically. Since the proposed learning process description language provides a standardized way to describe the ILE, it allows the system to interact with the ALS to adapt the contents in the ILE dynamically via the learning process description files while the system remains agnostic to the ALS implementation. As shown in the visual example of dynamic content adaption in Figure 1, when the user has completed the previous task (i.e., moving the blocks from one location to the other), the ALS increases the complexity of the task to challenge the user. Similarly, when the user has difficulty completing the tasks, ALS can reduce the task difficulty and provide corresponding tutorials.

The dynamic content adaption module will periodically check the description files of the main learning process and all its dependencies. If any updates are detected, the ILE will dynamically adapt to the new description as shown in Figure 3. The files are checked in the topological order of the file

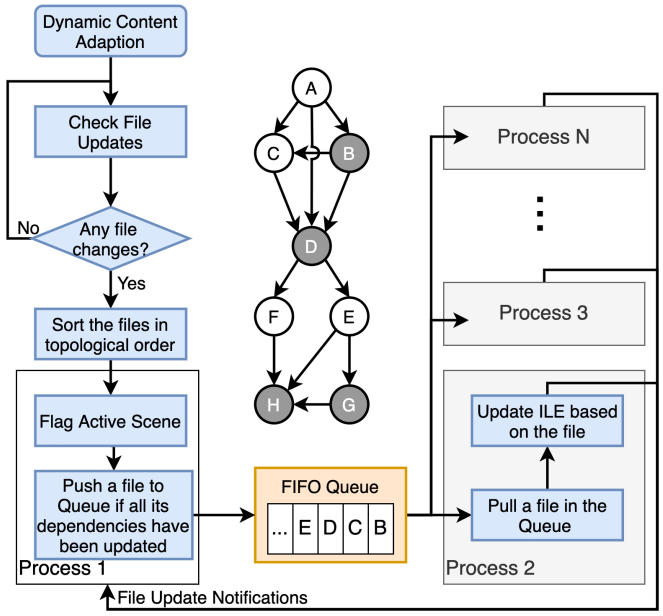


Fig. 3: The asynchronous and parallel dynamic content adaption based on the file dependency graph (white nodes: unchanged files, gray nodes: updated files).

dependencies. Since the update is performed in runtime, it is desired that the system can update the ILE efficiently so that the updated contents can be seamlessly integrated into the system. When large amounts of multimedia contents are presented, efficiently indexing the contents in the learning environment can be incorporated to increase the speed in which content is presented [21]. The ILE will asynchronously process all the updated description files in parallel without violating the file dependency by implementing a FIFO Queue and message passing mechanism. For the file dependency graph shown in the middle of Figure 3, files B, D, G, and H are updated. File A will be checked first but it will not be pushed to the queue since it is not updated. File C will not be pushed into queue until Process 1 is notified that file B has been updated since file C depends on both files A and B. Although file C is not updated, file C needs to be pushed into the queue for integrity validation to avoid integrity issues in file C caused by the changes in file B. Once file D is updated, both files E and F will be pushed into the queue so they can be processed in parallel. Since our proposed system implements asynchronous parallel computing, file G can be pushed into the queue without being blocked, which further improves ILE update efficiency when file E is smaller than file F and it is updated quicker.

To update a modified file with integrity assurance, the dynamic content adaption takes the following six steps: **(1)** loads the modified description file; **(2)** compares the modified description (D_n) with the previous one (D_{n-1}) and finds the differences; **(3)** updates all the changes in the learning content, scenes, and assessments and validates that no reference to these contents and scenes exists if any learning contents or scenes are removed; **(4)** updates the instructional strategies and validates that all the scenes and embedded processes in the strategies exist; **(5)** validates that the instructional strategies are still well-defined based on Eq. (2) and Eq. (3) if there is any modification in scenes and instructional strategies; and **(6)** updates the objectives.

Furthermore, it is possible that learning content viewed by the user needs to be changed. To minimize the influence on user's experience, learning content is expected to be updated as soon as possible while they are outside of the user's view. The following optimization problem is solved to balance the latency and user's experience when updating the active scene in the ILE to resolve the conflicting objectives.

$$\begin{aligned} \min_{\tau_{i_k}} \quad & \sum_{k=1}^{N_{s^*}} [-\lambda_{i_k}(\tau_{i_k}) + \alpha \cdot \tau_{i_k}] \\ \text{s.t.} \quad & \lambda_{i_k}(\tau_{i_k}) = \arg\max_{\lambda} \sum_{i=1}^{N_D(\tau_{i_k})} \ln \left[\frac{e^{-\lambda} \lambda^{n_i}}{n_i!} \right] \end{aligned} \tag{4}$$

where τ_{i_k} is the expected waiting time for learning content c_{i_k} in the active scene, N_{s^*} is the number of contents in the active scene, $\alpha > 0$ is the weighting factor to configure the tolerance to wait, and $\lambda_{i_k}(\tau_{i_k})$ is the expected duration of content c_{i_k} being visible within time τ_{i_k}. We assume that the content being visible follows Poisson distribution and thus $\lambda_{i_k}(\tau_{i_k})$ is obtained by maximum likelihood estimation based on user's behavioral data in the scene. $N_D(\tau_{i_k})$ is the number of τ_{i_k}-length samples in the data, and n_i is the time when the content is visible. Once the optimal τ_{i_k} is computed, when the waiting time of any content c_{i_k} exceeds its expectation τ_{i_k}, a notification will be sent to the user and all the contents will be updated.

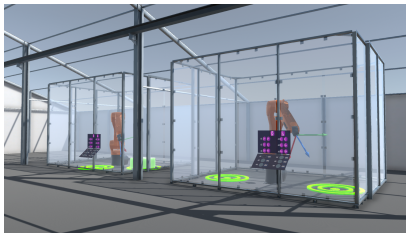
C. Multimodal Content Monitoring

Another main component of the proposed ILE is monitoring the environment and allowing a seamless integration with AI systems to retrieve and use the data from the ILE in real-time. To this end, an event driven data collection system was incorporated [22], consisting of two main components: Data Writers (DWs) and the Data Collection System (DCS).

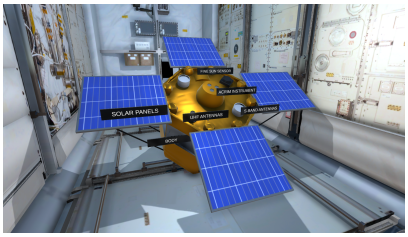
A DW controls each different type of data (e.g., eye tracking, head pose, heart rate, etc). The DCS controls each DW instance. This system works well for time-series data, in which by default, the DCS collects data at a user-specified time-step, but can also work for event-based data. Each DW accesses a separate service driver to acquire data. Data collection can be stopped and uploaded at any time using CSV files to quickly start testing and integrating with AI models such as an ALS. DWs can also store a user-specified amount of data in a first-in first-out (FIFO) buffer to enable inferences across multiple frames if necessary.

IV. Experimental Results

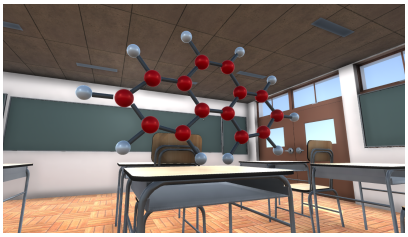
VR/AR Environments can, by nature, be very computationally intensive. Hence, it is imperative that any further abstraction in the form of a framework such as the one proposed uses as little overhead as possible while being performant.



(a) Pick and place with a robot arm.



(b) Labeling satellite components.



(c) Building molecules.

Fig. 4: Rendered view of each type of test scene.

TABLE I: System performance metrics for various test scenes.

Scene	FPS					Used Memory (MB)					Rendering Execution (ms)					Script Execution (ms)				
	Min	Max	Mean	Median	Std	Min	Max	Mean	Median	Std	Min	Max	Mean	Median	Std	Min	Max	Mean	Median	Std
Baseline	82.9	95.1	89.1	89.3	2.4	145.5	146.2	145.8	145.8	0.2	0.5	14.3	9.9	10.1	0.6	0.4	9.8	0.5	0.4	0.2
Simple *Pick and Place*	81.6	97.0	89.4	89.4	2.7	185.3	214.1	197.7	196.6	7.9	0.7	25.7	7.8	8.0	1.0	1.1	8.8	2.0	1.8	0.8
Simple *Satellite Labeling*	82.3	96.9	89.6	89.4	2.7	149.8	162.9	156.6	156.6	4.1	4.1	11.4	9.3	9.3	0.4	0.3	1.5	0.7	0.6	0.1
Simple *Molecule Building*	82.9	96.0	89.5	89.4	2.5	148.3	156.4	153.7	154.0	1.7	0.8	21.9	9.5	9.3	1.7	0.3	10.3	0.7	0.6	0.8
Complex *Pick and Place*	85.2	93.8	89.5	89.4	1.7	191.0	217.3	203.4	202.8	6.1	1.0	12.9	7.7	8.1	1.3	2.0	10.7	2.5	2.1	1.2
Complex *Satellite Labeling*	82.0	97.0	89.5	89.4	2.8	156.7	158.8	157.7	157.6	0.4	2.4	22.1	9.3	9.3	0.6	0.4	3.6	0.7	0.7	0.1
Complex *Molecule Building*	84.2	94.8	89.4	89.0	2.0	148.8	157.9	153.1	154.3	2.9	0.3	13.3	9.4	9.4	0.5	0.3	1.8	0.6	0.6	0.1

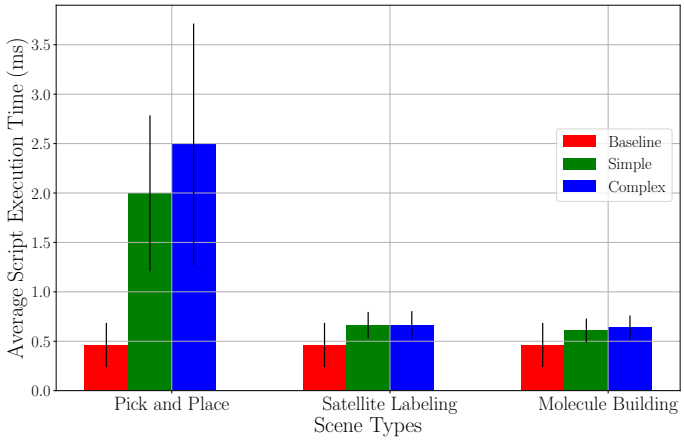


Fig. 5: Average script execution time within each test scene.

TABLE II: Scene properties for each test scene.

Scene	Triangle Count	Objectives
Baseline	0	0
Simple *Pick and Place*	348,916	8
Simple *Satellite Labeling*	185,032	8
Simple *Molecule Building*	135,864	3
Complex *Pick and Place*	697,832	16
Complex *Satellite Labeling*	370,064	16
Complex *Molecule Building*	357,912	3

Several different ILEs were created to explore this, and a diverse set of performance metrics were collected to gauge how this framework would perform to potential end-users.

A. Experiment Configuration

1) System Configuration: For this evaluation, the proposed multimedia system was developed in the Unity Engine. The Windows 10 test machine has an Intel i7-9750H CPU, NVIDIA GeForce RTX 2060 GPU and 16GB of RAM. Three distinct tasks were developed to showcase different use-cases for ILEs. Users can use the robot arms to pick and place objects, label the different sensors and antennas on NASA's ACRIMSAT satellite, and build the structure of alkane molecules. To test the learning process structure, process and scene JSON files were loaded locally at runtime. These JSONs were written manually, using unity prefab assets. For objectives and assessments, the MoonSharp Lua interpreter was used for code execution within an ILE using a Lua API to C# functions. Lua code is embedded within the description JSONs. DWs for pointer and head pose were also enabled for data collection testing. Initial AI integration was implemented with the confidence estimation model detailed in [22], which runs inferences using the original dataset during the assessment portion of learning scene traversal. The model

was deployed on a GPU server that houses an Intel Xeon Silver 4116 CPU, Tesla V100-SXM2 GPU, and 128 GB of RAM. The HTC VIVE VR headset was used for testing.

2) ILE Test Cases: To test ILE performance under different loads, three ILEs were created: baseline, simplified, and complex. The baseline ILE provides the performance of an idle, empty scene. Both of the simplified and complex ones contain three scenes, with their learning processes organized as a linear graph. Each scene focuses on one learning task, containing a linear sub-graph of objectives and assessments within, similar to a traditional laboratory course. For *pick and place* and *satellite labeling*, stress testing is done with duplicates, while in *molecule building*, the user builds increasingly complex alkane molecules. Each ILE was tested using the HTC VIVE VR headset, simulating users completing objectives, and traversing the learning process.

3) Profiling the ILE: Performance profiling was conducted using Unity's native profiling functions. This allows for the direct access to various metrics such as frames per second (FPS) across the entire lifetime of the application and collecting data on the most recent frame every half a second.

B. Performance Evaluation

1) Render and Script Execution Time: When looking at render execution times (RET) in Table I, they are the longest in the baseline (mean: 9.9ms; median: 10.1ms), while the *pick and place* scenes are the shortest (mean: 7.7ms; median: 8.1ms) despite being more computationally intensive. This is due to Vertical Synchronization (VSync), causing the render loop to wait if the scene is rendering too quickly. Because the *pick and place* scenes have a much higher script execution time (SET), the render loop does not wait as long to hit 90 FPS, which is the VSync limit for the HTC VIVE. Viewing the actual script execution time (SET) for each scene as in Figure 5 and Table I can provide a better understanding of actual scene performance than the raw RET. In this case, the slowest scene is the complex *pick and place* because it includes multiple simulated 6DoF robots with control panels, while the other scenes consist of static objects. Even this scene only uses about 22.5% of the 11.1 ms necessary for keeping a consistent 90 FPS on script execution. The slowest SETs happen when interpreting assessments and objectives, which leads to higher than usual SET at specific frames, such as the high maximum in the simple *molecule building* scene.

2) Memory Usage: The simple and complex scenes use more memory on average than the baseline. However, the simple and complex scenes for each task have much less variance in memory consumption (e.g., the complex *pick and place* uses on average 5.7MB more memory than the simple one). Doubling the number of assets does little to the used memory, and most of the memory usage is shared overhead related to the loading learning processes or engine overhead. The simple and complex scenes used less memory during initialization to load the content from the process files. Both the used memory and reserved memory are within margins for even the VR/AR all-in-one headsets, which will usually allow RAM usage up to 1 to 2 GB.

3) Discussion: In our experiment, all the scenes stayed around 90 FPS, ideal for most VR/AR headsets. Memory usage on all processes was also low, with a large margin for all-in-one headsets, such as the Oculus Quest or Microsoft HoloLens. The scenes in each process were also created with detailed assets, such as the 6DoF robot, which have 80,000 triangles. One example of this is the tile models used for generating the floor grid, which was default assets from Unity, with each plane containing 200 triangles, whereas an optimized plane would only consist of 2 triangles. Optimizing this can allow for more objectives and assets to be loaded. With predicted lower render times from optimized assets, the processor could spend more time executing code related to (1) the evaluation of objectives and assessment conditions; and (2) learning processes with very large directed hypergraphs.

V. Conclusion and Future Work

In this paper, we propose a multimedia system for immersive learning that leverages a novel learning process description language to represent the adaptable ILEs built using directed hypergraphs. Experimental results show that this system is performant enough to enable the simplified creation of ILEs and dynamic adaption of contents in the ILE without introducing excessive overhead. In the future, more testing could be done directly with all-in-one VR headsets such as the Oculus Quest to gauge performance on lower-cost devices, which would likely be more useful when deploying such a multimedia system for training purposes. Future work would also include developing and integrating novel methods into the system to continue simplifying the ILE creation, such as directly converting traditional educational content into an ILE.

VI. Acknowledgments

This research is partially supported by NSF OIA-1937019, NSF OIA-2029557, and NSF CNS-1952089.

References

[1] F. Osti, A. Ceruti, A. Liverani, and G. Caligiana, "Semi-automatic design for disassembly strategy planning: An augmented reality approach," *Procedia Manufacturing*, vol. 11, pp. 1481–1488, 2017.

[2] D. Vergara, M. P. Rubio, and M. Lorenzo, "On the design of virtual reality learning environments in engineering," *Multimodal Technologies and Interaction*, vol. 1, no. 2, pp. 11:1–11:12, 2017.

[3] S. K. Babu, S. Krishna, U. R., and R. R. Bhavani, "Virtual reality learning environments for vocational education: A comparison study with conventional instructional media on knowledge retention," in *18th IEEE International Conference on Advanced Learning Technologies*. Washington, DC: IEEE Computer Society, 2018, pp. 385–389.

[4] Z. Taçgın, "The perceived effectiveness regarding immersive virtual reality learning environments changes by the prior knowledge of learners," *Education and Information Technologies*, vol. 25, pp. 2791—-2809, 2020.

[5] R. Dunn, "Learning styles: Theory, research, and practice," in *National Forum of Applied Educational Research Journal*, vol. 13, no. 1, 2000, pp. 3–22.

[6] C. Fowler, "Virtual reality and learning: Where is the pedagogy?" *British journal of educational technology*, vol. 46, no. 2, pp. 412–422, 2015.

[7] E. K. Orman, H. E. Price, and C. R. Russell, "Feasibility of using an augmented immersive virtual reality learning environment to enhance music conducting skills," *Journal of Music Teacher Education*, vol. 27, no. 1, pp. 24–35, 2017.

[8] J. Saunier, M. Barange, B. Blandin, R. Querrec, and J. Taoum, "Designing adaptable virtual reality learning environments," in *Proceedings of the 2016 Virtual Reality International Conference*. New York, NY: ACM, 2016, pp. 5:1–5:4.

[9] L. Oubahssi, O. Mahdi, C. Piau-Toffolon, and S. Iksal, "A process of design and production of virtual reality learning environments," in *International Conference on Interactive Collaborative Learning*. Berlin, Germany: Springer, 2018, pp. 353–364.

[10] J. R. Carbonell, "AI in CAI: An artificial-intelligence approach to computer-assisted instruction," *IEEE Transactions on Man-Machine Systems*, vol. 11, no. 4, pp. 190–202, 1970.

[11] A. Karadimce and D. Davcev, "Personalized multimedia content generation using the QoE metrics in distance learning systems," in *Fourth International Conference on Adaptive and Self-Adaptive Systems and Applications*. Wilmington, DE: IARIA XPS Press, 2012, pp. 1–6.

[12] N. Shaker, G. N. Yannakakis, and J. Togelius, "Towards automatic personalized content generation for platform games," in *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 6, no. 1. Cambridge, MA: The AAAI Press, 2010, pp. 63–68.

[13] A. K. Upadhyay and K. Khandelwal, "Artificial intelligence-based training learning from application," *Development and Learning in Organizations*, vol. 33, no. 2, pp. 20–23, 2019.

[14] L. Richlin, *Blueprint for learning: Constructing college courses to facilitate, assess, and document learning*. Sterling, VA: Stylus Publishing, LLC., 2006.

[15] G. D. Posner, M. L. Clark, and V. J. Grant, "Simulation in the clinical setting: towards a standard lexicon," *Advances in Simulation*, vol. 2, no. 1, pp. 15:1–15:5, 2017.

[16] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. E. P. Reyes, M. Shyu, S. Chen, and S. S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Computing Surveys*, vol. 51, no. 5, pp. 92:1–92:36, 2019.

[17] H. Tian, Y. Tao, S. Pouyanfar, S.-C. Chen, and M.-L. Shyu, "Multimodal deep representation learning for video classification," *World Wide Web*, vol. 22, no. 3, pp. 1325–1341, 2019.

[18] S. Pouyanfar, Y. Yang, S.-C. Chen, M.-L. Shyu, and S. S. Iyengar, "Multimedia big data analytics: A survey," *ACM Computing Surveys*, vol. 51, no. 1, pp. 10:1–10:34, 2018.

[19] L. D. Fink, *Creating significant learning experiences: An integrated approach to designing college courses*. Hoboken, NJ: John Wiley & Sons, 2013.

[20] F. Pezoa, J. L. Reutter, F. Suárez, M. Ugarte, and D. Vrgoc, "Foundations of JSON schema," in *Proceedings of the 25th International Conference on World Wide Web*. New York, NY: ACM, 2016, pp. 263–273.

[21] S.-C. Chen and R. Kashyap, "A spatio-temporal semantic model for multimedia database systems and multimedia information systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 4, pp. 607–622, 2001.

[22] Y. Tao, E. Coltey, T. Wang, M. A. Jr., M.-L. Shyu, S.-C. Chen, H. Alhaffar, A. Elias, B. Bogosian, and S. Vassigh, "Confidence estimation using machine learning in immersive learning environments," in *IEEE Conference on Multimedia Information Processing and Retrieval*. Piscataway, NJ: IEEE, 2020, pp. 247–252.