A Framework for Intelligent Virtual Reality Tutoring System Using Semantic Web Technology

Victor Häfner[a], Tengyu Li*[b], Felix Longge Michels[c], Polina Häfner[d], Haoran Yu and Jivka Ovtcharova

Institute for Information Management in Engineering (IMI), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

Keywords: Virtual Reality Training, Personalized Tutoring, Semantic Web Technology, Intelligent Tutoring System, Engineering Education, Virtual Flow Rig Training.

Abstract: Adaptive and personalized immersive virtual reality (VR) training is a current research focus, aiming to make training more efficient and let users have a better learning experience. But there is currently an unmet need for flexible VR training that can adapt to the user's abilities and performance to provide a personalized learning process. The main problem is the high cost of setting up and maintaining virtual environments, resulting in a lack of flexibility to easily update training content. To solve this problem, we incorporate Semantic Web technology into VR training. Using the heterogeneous knowledge integration and reasoning capabilities of the Semantic Web, VR training content can be easily created and updated. In addition, due to the advantages of the intelligent tutoring system (ITS) in tracking the student performance and providing automated and personalized tutoring, we explore the integration of the ITS into the VR learning environment. Therefore, this paper proposes a framework for a Semantic Web-Enabled Intelligent VR Tutoring System, especially for engineering education, and demonstrates its feasibility by a proof of concept application. The system is exemplified using a virtual flow rig application for training, facilitating the education of students and professionals on fluid flow principles, control strategies, and measurement techniques.

1 INTRODUCTION

With the emergence of consumer hardware of immersive virtual reality (VR), the application range of VR technology continues to expand, involving many fields such as education, training and entertainment (Häfner, 2020; Zahabi and Abdul Razak, 2020). Among them, the application of VR in the field of training has attracted much attention. It can simulate expensive or dangerous scenarios, thereby reducing the cost of training, eliminating risks for trainees, and greatly improving the flexibility of training in time and location (Ruthenbeck and Reynolds, 2015). In addition, immersive and contextual experiences provide the best basis for understanding knowledge and skills (Dale, 1969). Due to the significant advantages of VR training, it has been widely used in industry (Richard et al., 2021b), medical (Vaughan et al., 2016), military (Bhagat et al., 2016), construction (Schiavi et al., 2021) and other fields. In order to make VR training more effective, current research focuses on adaptation and personalization in VR training, and many achievements have been made (Zahabi and Abdul Razak, 2020). The U.S. National Academy of Engineering (U.S. National Academy of Engineering, 2017) has also listed "Enhancing Virtual Reality" and "Advancing Personalized Learning" among the major challenges in Engineering in the 21st century. Nonetheless, there is an unmet need for flexible VR training that can adapt to the user's abilities and performance to provide a personalized learning process. This is because most studies concentrate on the adaptation of features or their combinations in a given training scenario (such as display features, controlled elements, difficulty level determined by other features, etc.) (Zahabi and Abdul Razak, 2020), and pay less attention to the adaptation of training content. The main problem is the high cost of setting up and maintaining virtual environments, resulting in a lack of flexibility to easily update training content. To address this problem, it is promising to utilize Se-

[a] https://orcid.org/0000-0001-8682-1122
[b] https://orcid.org/0009-0008-8174-1009
[c] https://orcid.org/0000-0001-6533-4886
[d] https://orcid.org/0000-0003-4534-351X
*Corresponding author

Häfner, V., Li, T., Michels, F., Häfner, P., Yu, H. and Ovtcharova, J.
A Framework for Intelligent Virtual Reality Tutoring System Using Semantic Web Technology.
DOI: 10.5220/0012746400003693

In *Proceedings of the 16th International Conference on Computer Supported Education (CSEDU 2024) - Volume 2*, pages 141-152
ISBN: 978-989-758-697-2; ISSN: 2184-5026

mantic Web technology to facilitate ontology-driven VR training. Ontologies can support the integration and management of heterogeneous knowledge such as virtual three-dimensional (3D) scenes, interactions and tasks (Benferdia et al., 2021). Once implemented, the training content can be easily created and updated with the support of the reasoner. However, there are few literatures on this aspect at present. An Intelligent Tutoring System (ITS) is a computer-based tutoring agent engineered to track and analyze a "student model", which encompasses the student's progress, performance, and learning patterns. The primary objective of an ITS is to provide automated and personalized tutoring, comprising task selection, error detection and rectification, step guidance, and other adaptive learning approaches (Häfner, 2021; Herbert et al., 2018). Integrating Intelligent Tutoring Systems (ITSs) into VR learning environments represents an emerging research field, yet the extent of exploration to date remains limited (Laine et al., 2022). As a response to this gap, we propose the concept of a Semantic Web-Enabled Intelligent VR Tutoring System, a fusion of Semantic Web technology, ITS, and VR training applications. Semantic Web technology, an artificial intelligence (AI) approach based on knowledge representation, offers an alternative pathway to implement an ITS. Hence, the integration of these three components is viable, and currently, there exists no established framework to describe such a comprehensive system.

In view of the above development trends and challenges of VR training, this paper proposes a framework of Semantic Web-Enabled Intelligent VR Tutoring System. To illustrate its practical application, we specifically focus on the development of an intelligent VR tutoring system for a flow rig. The flow rig is a valuable tool utilized extensively in research, experimentation, analysis, testing, and validation within engineering, fluid dynamics, thermodynamics, and related domains. While the flow rig serves a wide array of applications, our particular emphasis is on its pivotal role in education and the creation of VR training application to showcase its effectiveness in teaching control principles, methods, and operational procedures for fundamental process parameters such as flow rate, level, volume, and differential pressure within piping systems. The application caters to a diverse range of users including students in educational settings, vocational training programs, sales and technical support personnel, as well as professionals seeking continuing education and research and development teams in engineering and manufacturing industries. The flow rig use case serves as an example training application and a proof of concept for the proposed framework. The implementation methods of the three core modules within the framework are detailed, aiming to provide readers with a clear understanding of the system's functionality and potential. Among them, in the implementation of immersive VR learning environment, we focus on the high-fidelity simulation of complex machines in the field of mechanical engineering. Furthermore, we point out the need and importance of automating the virtualization process of machines, aiming to facilitate further the process of creation of immersive VR trainings.

The subsequent sections of this paper are outlined as follows. In Section 2, we present a comprehensive review of related research. Section 3 elaborates on the framework of the proposed ITS system. Following that, Section 4 provides an introduction to the implementation approaches for the three core modules. Finally, in Section 5, we summarize the paper and offer insights into potential avenues for future research.

2 RELATED WORK

Here, we mainly review the prior work on the application of Semantic Web technology in virtual training and ITSs. Due to the advantages of the Semantic Web in knowledge representation, integration, sharing, reuse, reasoning, etc., there is growing interest in using the Semantic Web in virtual training. Häfner (Häfner, 2017) pointed out that connecting the virtual world with semantic models to build a smart virtual environment has great prospects for training applications, especially in the engineering field. Walczak et al. (Walczak et al., 2020) proposed an ontology-based semantic modeling method for VR training scenarios, and developed a scenario editor based on this method, which supports users without advanced programming and modeling skills to easily create training scenarios. Elenius et al. (Elenius et al., 2016) proposed a framework for users to interact with virtual environments based on ontologies and rules, and applied it to the training of weapon skills. Filho et al. (Filho and Vieira, 2014; Filho et al., 2015) used an ontology approach to facilitate the development of virtual scenarios in a training simulator for the operation of electrical systems. Havard et al. (Havard et al., 2017) proposed an industrial ontology describing operational guidelines for integration with information systems. Gorecky et al. (Gorecky et al., 2014; Gorecky et al., 2017) used an ontology-based semantic modeling method to solve the integration problem of heterogeneous data in order to promote the existing enterprise data (such as production line structure, process description, etc.) to be used in the setting of vir-

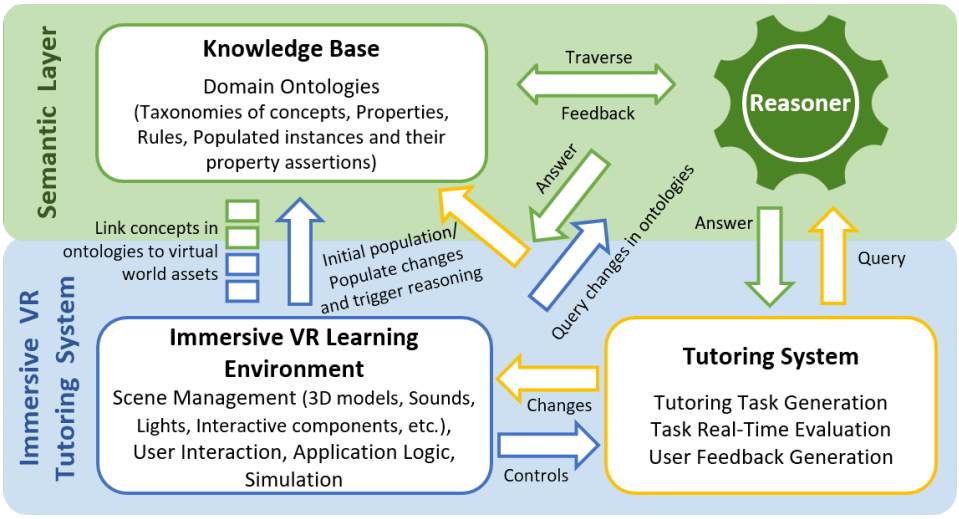

Figure 1: Framework of the Semantic Web-Enabled Intelligent VR Tutoring System (adapted from (Häfner, 2021)).

tual training scenarios. It can be seen that most studies use Semantic Web technology to solve the problems of scenario creation and heterogeneous knowledge integration in virtual training, while few studies focus on how to use Semantic Web to drive the VR training process, including user interaction, training content update, and evaluation.

Some achievements have also been made in the application of Semantic Web in ITS. Chang et al. (Chang et al., 2020) used ontology to build a tutoring model in an ITS, and explored the use of data mining technology to accelerate this process. Jacinto and Oliveira (Jacinto and Oliveira, 2008) presented an architecture of an ITS based on multiple ontologies. Muñoz Merino and Kloos (Muñoz Merino and Kloos, 2008) combined the ITS with Semantic Web technology to provide students with personalized prompts adaptively. Vesin et al. (Vesin et al., 2012) used semantic reasoning to identify students' learning styles and recommend personalized courses for students. Wen et al. (Wen et al., 2022) developed an ontology-driven dialogue-based ITS. However, few studies have explored the combination of Semantic Web, ITS and VR training.

3 FRAMEWORK OVERVIEW

The framework of the Semantic Web-enabled intelligent VR tutoring system is shown in Figure 1. The whole is divided into three core modules, namely the semantic layer, the immersive VR learning environment and the tutoring system.

The semantic layer is a module that provides intelligence based on Semantic Web technology. It includes an ontology-based knowledge base and a reasoner. The ontology defines the data model of the knowledge base (Häfner et al., 2013; Wicaksono et al., 2013), which includes several small ontologies for different domains describing the virtual world, user interactions and tutoring situations, such as the ontology characterizing the state of the visual interactive object like machine virtual twin ontology, student behavior ontology, tutoring task ontology, student skill ontology, etc. These ontologies are comprised of domain-related concept (class) taxonomies, properties, reasoning rules, concept instances and their property assertions. Concepts from different ontologies are linked to each other through rules. Virtual world assets are linked to the corresponding concepts. In the stage of system initialization, relevant object instances are generated according to the initial training requirements and populate the knowledge base, such as specific virtual scene instances, initial state parameter instances for both the user and the system, tutoring task instances, among other relevant elements. During the tutoring phase, the system continuously adds or updates instances in the knowledge base in reaction to changes within the VR environment and tutoring system, like populating user behavior instances, updating tasks or virtual scenes, and so on. The semantic layer tracks the knowledge of the virtual learning environment and the user in real time, which is very important for the dynamic update of the visual scene and the function implementation of the tutoring system.

Reasoner is a module in the semantic layer that implements intelligence. It engages in automatic reasoning and logical deduction using ontology rules to derive new knowledge from the existing knowl-

edge within the knowledge base, ultimately facilitating decision-making and query responses. Scenarios that trigger the reasoner include user interactions with the virtual scene and queries from the application logic in the VR environment or the tutoring system. The reasoner uses predefined ontology rules to infer the changes that should be made in the virtual world based on the user interactions and their interpreted behaviors, or to execute prescribed tutoring actions. The inference results are reflected as modifications to the instance information in the knowledge base. The reasoner cannot directly change the virtual world, which requires the application logic to achieve it. Therefore, the application logic should send queries to the reasoner to learn about changes in the knowledge base. In addition, the tutoring system can also issue queries to the reasoner to check the progress of task execution, the user's wrong behavior, or the improvement of the user's skills, etc. The reasoner will traverse all the knowledge of the virtual world and the user in the knowledge base, and make corresponding feedback based on all relevant rules.

The primary roles of the tutoring system are to generate tutoring tasks, evaluate tasks in real time, and generate user feedback. These three functions are implemented via interacting with the semantic layer, which will be described in detail in Section 4.3. The results produced by the tutoring system will change the virtual environment, for example, changes in tasks may affect changes in the virtual scene, and user feedback is visually presented in real time on the graphical interface. In addition, in order to give users the option to control their own learning, it is also necessary to design an interactive interface for the tutoring system in the virtual environment.

The immersive VR learning environment consists of scene management, simulations, user interaction and application logic. A virtual scene encompasses assets such as 3D models, materials, textures and sounds, as well as artefacts like virtual cameras, lights and virtual sound sources. The arrangement of these virtual assets and their topological relationships can be managed using a hierarchical structure diagram. In each frame, the hierarchical structure diagram is traversed to compute the transformation of the virtual world. Simulations are approximations of various behaviors in the real world, and are also the core of building an immersive VR learning environment. Usually, VR engines have been embedded with physics engines to simulate the basic physical behavior of objects, such as collisions, gravity, vehicle dynamics, and so on. User interaction, a key characteristic of immersive VR applications, depends on the user behavior and the choice of interactive devices. Users can virtually select, grab, and move virtual objects, as well as implement different navigation modes as needed. Additionally, the realistic manipulation of virtual objects can be realized through the simulation of physical reactions of these objects like deformations and the implementation of force feedback. Finally, developers should add application logic to the virtual assets as needed to implement the necessary actions in the virtual environment. The application logic is programmatically created using scripting languages. Within this ITS system framework, an important function that the application logic should perform is to populate various instance data into the semantic layer during system initialization and run-time. In addition, the application logic is also responsible for dynamically generating some ontology rules, issuing queries, and activating the reasoner.

In the following sections, we will provide a detailed description of the implementation of the three core modules of this system.

4 IMPLEMENTATION

4.1 Concept of Automating the Virtualization to Build up Machine Virtual Twins

The core and difficulty of constructing an immersive VR learning environment is the simulation of the real world. Since our long-term work focuses on VR applications in the field of mechanical engineering, this section mainly discusses the simulation of real machines in the VR environment, which will be used for virtual training of machines.

The virtual twin of a machine is its virtual representation, which includes all dynamic and functional aspects (Häfner, 2019). The creation of the virtual twin involves knowledge from multiple domains, such as mechanical, electrical, and machine programming. Integrating planning data from various domains to generate an interactive virtual twin with realistic machine behavior is highly complex, time-consuming, and resource-consuming. Automatic aggregation of overall knowledge from planning data is desired, in particular automatic association of component data in Electrical Computer-Aided Design (ECAD) and Mechanical Computer-Aided Design (MCAD) (Michels and Häfner, 2022; Häfner et al., 2020). On the other hand, if the semantic information required for the simulation of kinematics, dynamics, hydrodynamics, etc. is lacking in the MCAD data, then this information should be de-

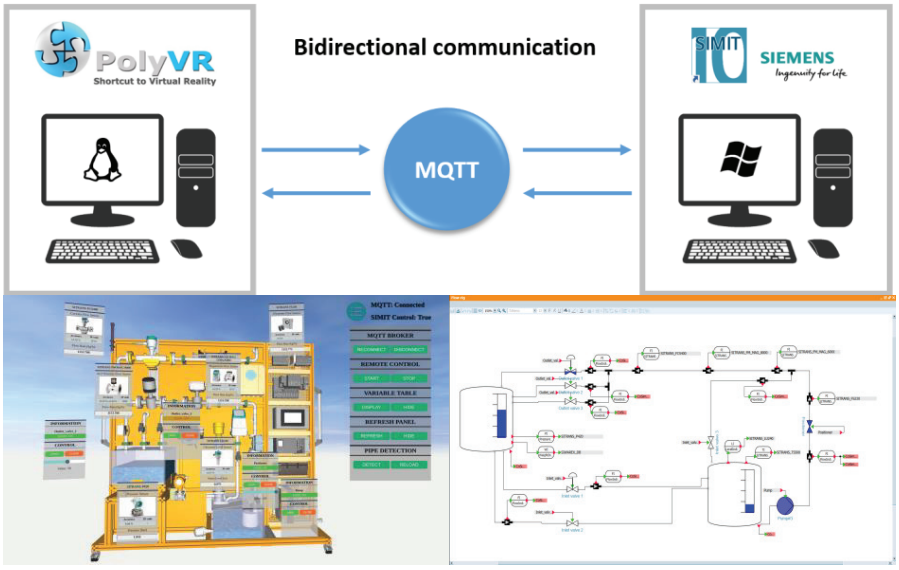

Figure 2: Architecture of flow rig virtual twin.

termined elsewhere. The development of automatic analysis algorithms for geometric objects helps to quickly capture as much inherent knowledge as possible from MCAD data to support parameter settings for various simulations. Specifically, the choice of the first data source in the virtualization process often comes to MCAD data as the entry point. These geometries are often exported from native engineering software tools in the exchange format *STEP*, which is the status quo due to its general compatibility with other Computer-Aided Design (CAD) software vendors. The exports lack further semantic information, which either is stored in the native design tool and cannot be exchanged digitally or rests in the non-digital knowledge of the engineers and domain experts. This semantic data can be for example mechanical relevant data such as rotation axis, radius and pitch of gears for the calculation of ratios in gearboxes to virtually generate a kinematic chain for the kinematic simulation (Michels and Häfner, 2022) or pipe parameters for generating a flow simulation, as shown in the example of flow rig below. Another regular difficulty with MCAD data exports is the inconsistent data quality, depending on the CAD system and the modelling methods used by the construction engineer. It may be that the product structure is bad, lacking meaningful groups of parts into assemblies. For complex machines, it is more urgent to realize the automation of MCAD data structure optimization.

The automated virtualization process forms the foundation for expediting the implementation of diverse virtual engineering applications, including but not limited to machine training, virtual commissioning, material flow simulation, and more (Michels and Häfner, 2022; Häfner et al., 2020). This approach holds significant promise and potential to streamline the authoring process of immersive learning applications. Furthermore, embedding the semantic layer in the virtual environment not only helps integrate the heterogeneous data needed to build a virtual twin, but also paves the way for the automatic parameterization of the simulation model by linking semantic information to virtual 3D assets, as well as expanding the range of possibilities for machine behavior simulation and user interaction options.

Let's take the VR tutoring system for a specific flow rig, which we are currently working on, as an illustrative example to further elucidate our concept. During the creation of a flow rig virtual twin, the pre-existing MCAD data was directly imported and deployed, while the system simulation for the hydrodynamics was coupled through the externally running simulation tool. In order to save the time and cost of creating the virtual twin, the existing *SIMIT* simulation was combined with the MCAD model to obtain an interactive virtual twin of the flow rig, as shown in Figure 2. The software *SIMIT* from *SIEMENS* is a simulation environment used for simulating the behavior of the flow rig, as shown on the right side of Figure 2. The virtual scene of the flow rig was generated in *PolyVR*, an open-source VR authoring software (Häfner, 2019), as shown on the left side of Figure 2. The communication between the virtual scene running and the hydrodynamic simulation was facilitated via *MQTT* (OASIS Standard, 2015), a well established protocol for Industry 4.0 applications. Over

this *MQTT* interface we sent sensor data such as flow rate, or tank level towards the virtual twin as well as action commands input from the user for individual valves and pumps towards *SIMIT*. One of the typical issues we encountered was that in the MCAD data, the scene graph generated from the product tree exhibited a flat structure without any hierarchical group concepts and the names of components were missing. A lack of semantic information to identify the components makes it difficult to map MCAD components to the *SIMIT* simulation model, which is necessary for visualizing and interacting with the virtual flow rig. To solve this problem, it is necessary to classify the components of the piping system and identify their topological position to map them to the simulation model. Moreover, for pipes, in addition to their type, semantic information such as length and inner radius are particularly important for their parametric simulation and analysis of flow characteristics such as throughput or pressure loss due to friction. Finally, we adopted a geometric analysis method to realize the classification of the components and the extraction of the basic pipe parameters, and further implemented the automatic generation of the piping system topology. Since this part is not the focus of this paper, the specific classification implementation method will not be covered in detail here.

4.2 Implementation of the Semantic Layer

Here, we give the general procedure for implementing the semantic layer, as shown in Figure 3. Ontology design is subjective, which means that the ontology designed by different developers has different definitions of concepts and properties. A good ontology taxonomy should be intuitive, and its internal relationships should be easy to grasp and explore (Häfner, 2019). In order to standardize the ontology design as much as possible to facilitate the subsequent merging and mapping of different ontologies, the guidelines proposed by Häfner (Häfner, 2021) can be referred to. The Web Ontology Language (OWL) is a prominent ontology description language, widely utilized in various domains. Specifically, OWL ontologies can be effectively modeled using tools like *Protégé*, an open-source graphical ontology development tool that significantly simplifies the process, enhancing accessibility for researchers and practitioners (Musen, 2015). In addition, the tool has a built-in reasoner, whose main function is to infer the hierarchical structure of classes, i.e. concepts, and to check the consistency of ontologies in the design phase.

Analyzing and determining all the domains

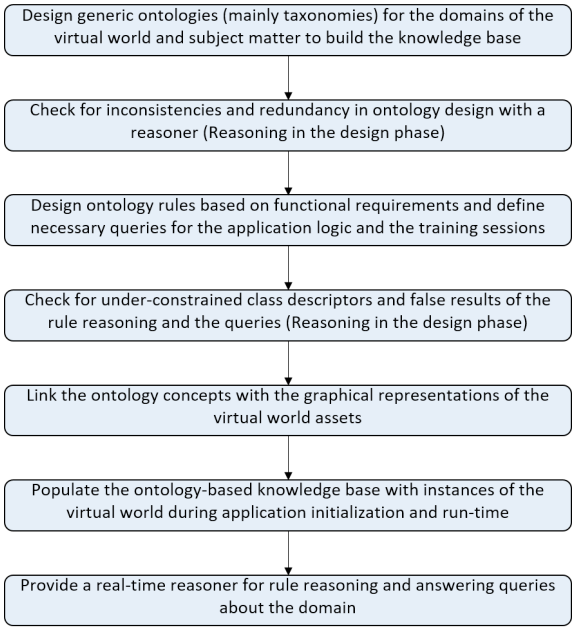

Figure 3: General procedure for implementing the semantic layer (adapted from (Häfner, 2021)).

needed to implement the tutoring system is the premise of constructing ontologies in the semantic layer, and it is also the first step in constructing the semantic layer. This paper takes the intelligent VR tutoring system of the flow rig mentioned in the previous section as an example, and briefly analyzes which domain ontologies are required for its semantic layer. First, the content and objectives of the tutoring should be clearly defined. While the flow rig is a versatile tool with applications in research, experimentation, analysis, testing, and validation across engineering, fluid dynamics, thermodynamics, and more, its primary function remains in education. It helps teach students the control principles and methods of basic process parameters such as flow rate, level, volume and differential pressure in the piping system, as well as the operation methods of common control components, equipment and measuring instruments, such as the starting steps of the pump, the measurement method of the flow meter, etc. More complexly, it can be used to teach the setting of a single closed-loop level/flow control system, the test method of the step response characteristics of the tank level, and the test method of the flow characteristics of the valve, etc. This type of learning is called skill training. The vast majority of skill training tasks for machines are procedural, which means that the task can be represented as an ordered sequence of behaviors or actions.

Therefore, based on the above analysis and the content of the preceding sections, we preliminarily concluded that the semantic layer of the ITS ought

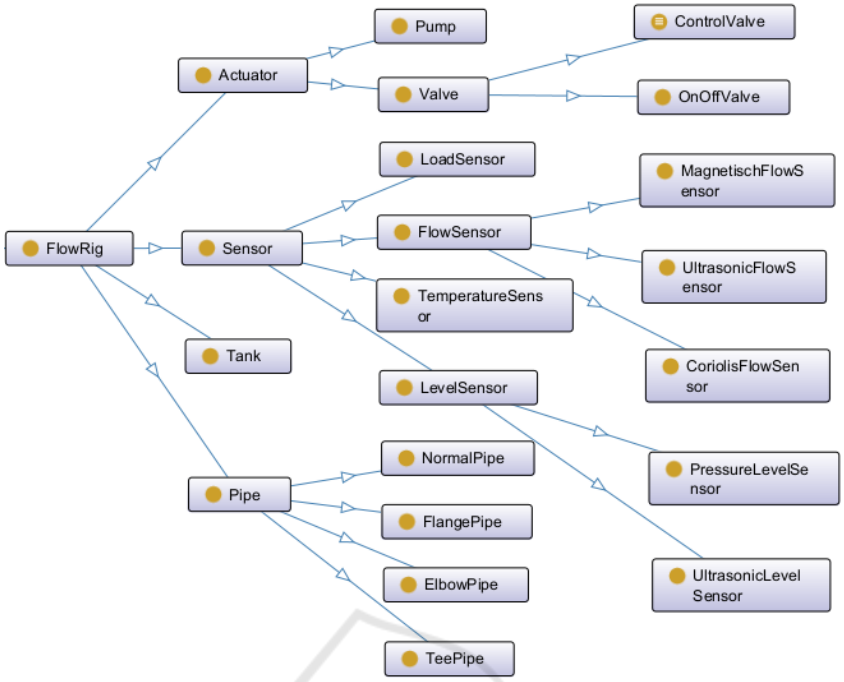

Figure 4: Flow rig virtual twin ontology.

to comprise four key domain ontologies. First among these is the virtual twin ontology, serving to elucidate information about each component within the flow rig and the interrelationships between these components. The second is the behavior ontology, describing the interactions available to students, along with the corresponding execution rules for these interactions on the virtual twin components. The third, the tutoring task ontology, is used to describe the procedural task content, task evaluation and user feedback content. Lastly, there is a skill ontology, responsible for recording the learning intensity and mastery of various skills, which is used to support the adaptive generation of tutoring tasks.

The component class hierarchy defined in the flow rig virtual twin ontology is shown in Figure 4. Define a set of properties for each component class, which can be divided into static properties to record basic component parameters such as the volume of the tank, the orifice diameter of the valve, etc., and dynamic properties to capture the current state of the component such as the opening degree of the valve, the level of the tank, etc. In the system initialization stage, the component instances of the flow rig virtual twin established in the previous section and their initial semantic information are populated into the virtual twin ontology knowledge base, where the basic structure of representing facts is a triple based on Resource Description Framework (RDF), that is, the format of subject (instance of class) - predicate (property of class) - object (property value) (Pan, 2009). Take the on-off valve class as an example, which includes the orifice diameter property and the switch position property to indicate the state of the on-off valve. Their range classes are *Decimal* class and *Boolean* class respectively. After system initialization, several fact assertions about specific on-off valves will be generated in the knowledge base. Take one of them as an example and use the RDF graph to represent it as shown in Figure 5. Each entity in the ontology (including classes, properties, and instances) has a unique Internationalized Resource Identifier (IRI) used for reference purposes. After mapping the flow rig virtual twin to the ontology, an automatic parametric simulation model is formed. Driven by the reasoner and behavior ontology rules, the flow rig virtual twin enables automatic responses to user interactions.

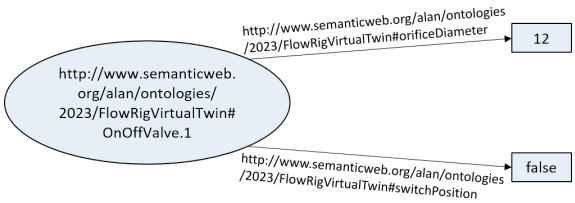

Figure 5: RDF graph describing the on-off valve instance OnOffValve.1.

The purpose of constructing the behavior ontology is, on the one hand, to endow the basic interface inter-

action between users and objects (such as click, drag and drop, etc.) with specific semantics, which helps the tutoring system meaningfully understand user behavior and make evaluations, and on the other hand, to realize the impact of user behavior on the virtual environment by using ontology rules. In addition, there may be more than one semantic operation that can be performed on an object, which should be explicitly presented and chosen by the user when manipulating the object. For example, when operating the pump of the flow rig, in addition to the operation to start the pump, it should also include the operation to check the submergence degree of the pump, which should be performed before starting the pump. By classifying various object operations, we consider constructing a behavior ontology composed of general behavior classes and object-specific behavior classes. The definition of general behavior classes helps the behavior ontology to be reused in other machine training, as well as to keep the ontology simple. General behaviors include: *Open*, *Close*, *Adjust*, *Press*, *Check*, *Record*, etc. The object-specific behavior can be adding water to the tank, checking the submergence depth of the pump, etc. Based on the needs of rule reasoning, define a set of properties for the behavior class to capture information associated with the behavior instance. For example, the *Open* class has an *openedObject* property, whose value can be a certain on-off valve or the pump; in addition to having an *adjustedObject* property, the *Adjust* class should also have a *value* property that records the final value of the adjustment, such as the opening adjustment of the control valve. When the user performs a behavior on a component object of the virtual twin, a new behavior instance and its property assertions will be created, which will be populated into the knowledge base in the form of RDF triples. Finally, the ontology rules describe the effects of different behaviors on the virtual twin. Ontology rules are also a kind of formalized knowledge in essence, which is represented based on *if-then* logic. Semantic Web Rule Language (SWRL) (Horrocks et al., 2004) is used to model ontology rules. Taking the adjustment of the control valve as an example, the behavior rule is modeled as follows:

- `adjustedObject(adjustInstance, ?cv)^`
  `value(adjustInstance, ?val)^`
  `openingDegree(?cv, ?opd)^`
  `swrlb:notEqual(?val, ?opd)->`
  `openingDegree(?cv, ?val)`

The prerequisite for the above rule to be true is that the expression on the left side of the arrow "->", that is, the *if* statement, is true. The part on the right side of the arrow represents the result of the rule, that is, the *then* statement. While generating an instance of the *Adjust* class (in this case replacing the instance name with the parameter *adjustInstance*), the above rule is created dynamically and the reasoner is triggered to reason based on the rule. The symbol "^" represents the logical operation "*and*", linking multiple subexpressions. The expressions *adjustedObject*, *value*, and *openingDegree* are all property expressions. Their first parameter represents the individual being tested, and the second parameter represents the property value of the individual. The SWRL syntax uses the form of a question mark plus a character to represent a wildcard, which is used to dynamically bind each property value of the individual. The *swrlb:notEqual* is a built-in function of SWRL to determine whether two parameters are unequal. On the result side of the rule, the property expression *openingDegree* is used as a property assertion to set the *openingDegree* property value of the individual bound to the first argument (that is, the control valve) to the value bound to the second argument (that is, the adjusted value of the *Adjust* behavior). Due to the monotonic reasoning characteristics of SWRL, SWRL rules can only add new reasoning information to the ontology but cannot modify the existing information of the ontology. Therefore, SPARQL, short for "SPARQL Protocol and RDF Query Language"(World Wide Web Consortium, 2013) can be used to delete old axioms. The overall meaning of this behavior rule is that if the user's adjustment behavior does change the opening degree of the control valve, then update the value of the opening degree property of the control valve in the virtual twin ontology. Once the application logic learns through queries that the information in the virtual twin ontology has changed, it will re-run the flow rig parametric simulation model to update the virtual environment.

The details and functions of the other two ontologies will be explained in conjunction with the content of the next section.

4.3 Intelligent Tutoring System

This section summarizes specific requirements for setting up an ITS and delivers some insights into their realization. As mentioned in Section 3, in our proposed framework, the functions of an ITS mainly include generating tutoring tasks, evaluating tasks in real time, and generating user feedback. These three functional modules are described in turn as follows.

First, for the tutoring task generation module, it should be able to adaptively generate the most suitable tutoring task considering the student's skill level, proficiency in skills and forgetting curve. In order to evaluate the skill level of the students, a simple

method is proposed, which is to divide the skills required by students into different levels through a top-down analysis. The learning of a higher-level skill is based on the mastery of all lower-level skills. Taking the training of the flow rig as an example, the premise for students to learn process parameter control strategies is that they have been able to skillfully operate various control components, equipment and measuring instruments. With this approach, a student's skill level can be obtained by analyzing his learning situation of all tasks at each level, which is recorded in the user profile. As long as there are tasks at a certain level that are not fully mastered, meaning that the student is still at that skill level, new tasks generated by the tutoring system should also be at that level. The tasks generated based on the student's skill level prevent students from being frustrated by the excessive burden during the training process or losing interest in learning due to tasks that are too easy.

For the evaluation of skill proficiency, it can be realized by formulating a user performance-based penalty rule, and taking the task score obtained based on the rule as the user proficiency of the skill corresponding to the task. The user's score for each task will be updated in the user profile. The tutoring system learns the user's proficiency in each skill by checking the latest scores for each task. A full score means complete mastery. In addition, the tutoring system should also consider the forgetting curve of knowledge when generating tasks, that is, consider the time factor. Periodically repeating tasks with perfect proficiency and re-evaluating the user's skill performance can effectively combat the forgetting curve. Therefore, the learning intensity of each skill expressed as the learning interval should also be recorded in the user profile and updated regularly.

It should be noted that all the tasks that the tutoring system can provide, and their corresponding scenes, should be defined in advance in the underlying data model of the task generation module. A task selection strategy that takes into account the user's skill level, proficiency, and timely spaced repetition needs to be designed and integrated. Before each task is generated, the module will read the relevant information of each skill from the skill ontology for use by the task selection algorithm. This is exactly why we created the skill ontology in the semantic layer of the tutoring system of the flow rig, as described in Section 4.2. A preliminary and simple skill ontology for the flow rig training can be constructed as shown in Figure 6. Each skill is endowed with proficiency and learning intensity properties. During system initialization, the proficiency and learning intensity information of the skills from the user profile will populate the skill ontology. In fact, in addition to the properties directly related to task selection, there are some properties in the ontology that are defined due to the need to build rule reasoning, which are flexibly added during the rule design process. The new tasks are inferred based on the designed ontology rules. The information of the newly generated tasks will also be updated in time to the tutoring task ontology. A preliminary tutoring task ontology for the flow rig is presented in Figure 7. All tutoring tasks for the flow rig are performed in the same scene.

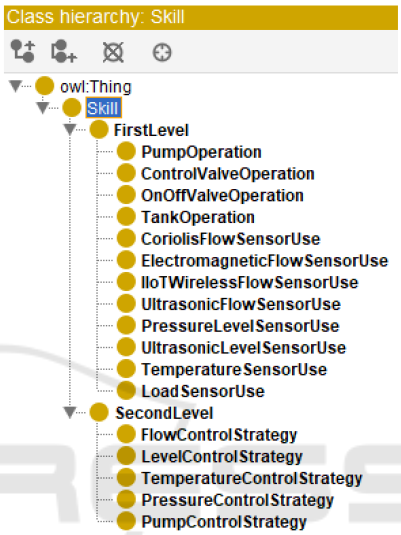

Figure 6: Skill ontology for flow rig training.

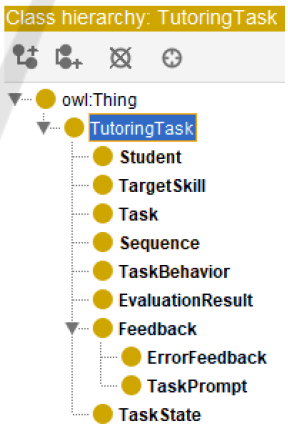

Figure 7: Tutoring task ontology.

Secondly, for the task real-time evaluation module, its main goal is to segment the user's interaction with the immersive environment in real time, especially for continuous interactions, identify the user's misbehavior, and ultimately evaluate the proficiency of the skill. For example, in a driving simulation, the computer should segment the user's driving behavior

in real time into a sequence of discrete atomic behaviors or events (e.g. acceleration, shifting, etc.) based on the time series of the user's hardware input data (e.g. the simulator's pedal signals, steering wheel signals, etc.) and the driving simulation output data. For the flow rig example above, the user's interaction with the flow rig virtual twin is inherently discrete, so it does not need to be segmented. And since each user's behavior will be recorded in the behavior ontology, the user's behavior can be easily traced. Since the task is also typically represented as a sequence of behaviors or events and is pre-stored in the semantic layer, the reasoner can detect whether the user is misbehaving based on predefined ontology rules. Detections can be event-triggered or periodically triggered. Here, based on the example of the control valve adjustment behavior in the previous section, the following simple example is given to show how the semantic layer implements the evaluation of user behavior, which is implemented based on SWRL and SPARQL.

- `order(adjustInstance, ?ord1)^ type(adjustInstance, ?t1)^ adjustedObject(adjustInstance, ?obj1)^ value(adjustInstance, ?val)^ TaskBehavior(?tb)^order(?tb, ?ord1)^ type(?tb, ?t1)^adjustedObject(?tb, ?obj1)^ value(?tb, ?val)-> isCorrect(adjustInstance, true)`
- `SELECT (COUNT(?x) AS ?c) WHERE {adjustInstance isCorrect ?x}`

The first one is the behavior evaluation rule based on SWRL, which is triggered immediately after the user performs the behavior. As shown in Figure 7, the *TaskBehavior* class is defined in the tutoring task ontology, which is a subclass of the behavior class. Each expected behavior in the task behavior sequence is an instance of this class, and each behavior instance has an *order* property value that represents the position of the behavior instance in the behavior sequence. The *type* property is used to indicate the type of the behavior. The *isCorrect* property indicates whether the user is performing the correct behavior, which is not asserted by default. The evaluation logic of this rule is to compare the behavior performed by the user with the expected behavior at the same position in the task behavior sequence. If the two are completely consistent, the reasoner asserts that the user's current behavior is correct.

The second one is the SPARQL query statement, which is used to output the reasoning results of the reasoner. It can also be implemented using Semantic Query-Enhanced Web Rule Language (SQWRL) (O'Connor and Das, 2009). The *WHERE* part defines the pattern to be matched in the query, which is based on the RDF triple structure. The wildcard *?x* dynamically binds the value obtained through pattern matching. For brevity, the prefixes of the behavior instance *adjustInstance* and property *isCorrect*, which represent the namespace, are omitted here. The *SELECT* part determines what data to output. Here, we use the *COUNT* function to calculate the number of *?x* that satisfies the pattern, and assign it to the wildcard *?c* for output. Since the *isCorrect* property of a behavior instance is not asserted by default, when the output value is *0*, it indicates that the current behavior is wrong; when the output value is *1*, it indicates that the behavior is correct. SPARQL queries can also be used to implement prompts for task information.

To be sure, different developers define the rules differently. The evaluation results will also populate the tutoring task ontology for use by the user feedback generation module. It should be noted that tasks should be defined with consideration that there may be more than one correct sequence of user behaviors, which can support students in exploratory learning and thus promote their understanding and mastery of skills (Aleven et al., 2009). Moreover, in order to simplify the task authoring process, it is promising to arrange the task behavior sequence by directly letting tutors to demonstrate in the virtual environment, called authoring-by-doing (Richard et al., 2021a). Based on the user's performance, this module will ultimately evaluate the user's proficiency in the currently trained skill and update it with the skill learning intensity to the skill ontology. Once the tutoring concludes, all skill-related information will be saved in the user's profile for future reference.

Finally, the main goal of the user feedback generation module is to present the prompts of task information and the real-time evaluation results of the evaluation module to the user. The presentation can be visual or audio output. During the first training of the task, the system should show the user the recommended and detailed task solution, and provide the most detailed error feedback in real time, including the wrong behavior and the corrective solution. In order to promote students' mastery of skills, the system should adaptively increase the difficulty of the task with the improvement of users' skill proficiency, which can be achieved by reducing the level of detail of prompts and feedback. To achieve this, a variety of feedback forms with different levels of detail can be pre-designed, and difficulty levels defined for them. When a new task is generated, its difficulty level is set, which will be recorded in the tutoring task ontology. Based on the difficulty property value of the task and real-time evaluation results, the feedback generation module generates feedback content with the corresponding level of detail with the help of

the reasoner, which will also be populated in the tutoring task ontology for visual or audio output module to read. There should also be the possibility for users to manage their training themselves, including task selection and difficulty settings.

5 CONCLUSION AND FUTURE WORK

This paper proposed a framework of Semantic Web-Enabled Intelligent VR Tutoring System, and expounded the implementation methods of the three core modules of the framework in combination with the flow rig use case. With regard to the development of an immersive VR learning environment, the problems and expected solutions in implementing the virtual twin for training were discussed, and it was emphasized that automating the virtualization process is helpful to accelerate the development of training applications in the field of mechanical engineering.

An intelligent VR tutoring system based on this framework can quickly deploy personalized training content according to the user's context and provide real-time evaluation and feedback on user behavior, which will help improve the effectiveness of VR training. The application of the ontology-driven methodology for updating training content is expected to lower the cost of setting up virtual environments for all training scenarios, which will be evaluated experimentally in the future. Indeed, the proposed framework can be extended to the development of skills training applications in other fields beyond engineering education, such as driving, construction, emergency scenarios, and more.

In the future work, the details of functional implementation of each module in the framework will be studied, such as the design method of each domain ontology in the semantic layer, the method of generating task behavior sequence through authoring-by-doing (Richard et al., 2021a), the task selection algorithm based on user context, and the adaptive user feedback, etc. The proposed framework and methodology will be utilized for the flow rig immersive training application, along with more complex use cases. Finally, in-depth user tests will be conducted to fully evaluate our research results.

ACKNOWLEDGEMENTS

This work is funded by the China Scholarship Council (CSC) under grant No. 202206890036, and the Ministry of Science, Research and Arts of the Federal State of Baden-Württemberg, Germany under KolabBW project within the InnovationCampus Future Mobility (ICM).

REFERENCES

Aleven, V., Mclaren, B. M., Sewall, J., and Koedinger, K. R. (2009). A new paradigm for intelligent tutoring systems: Example-tracing tutors. *International Journal of Artificial Intelligence in Education*, 19(2):105–154.

Benferdia, Y., Ahmad, M. N., Mustafa, M., and Ali, M. A. M. (2021). The role of ontologies through the lifecycle of virtual reality based training (VRT) development process: A review study. *International Journal of Advanced Computer Science and Applications*, 12(9).

Bhagat, K. K., Liou, W.-K., and Chang, C.-Y. (2016). A cost-effective interactive 3D virtual reality system applied to military live firing training. *Virtual Reality*, 20:127–140.

Chang, M., D'Aniello, G., Gaeta, M., Orciuoli, F., Sampson, D., and Simonelli, C. (2020). Building ontology-driven tutoring models for intelligent tutoring systems using data mining. *IEEE Access*, 8:48151–48162.

Dale, E. (1969). *Audiovisual Methods in Teaching*. Dryden Press, New York, 3nd edition.

Elenius, D., Denker, G., and Kim, M. (2016). Semantically enhanced virtual learning environments using sunflower. In *Proc. MTSR*, pages 81–93. Springer, Cham.

Filho, F. T., Aguiar, Y. P. C., and Vieira, M. F. Q. (2015). Ontology based modelling of operator training simulator scenarios from human error reports. In *Proc. SIMULTECH*, pages 261–270, Setúbal, Portugal. SciTePress.

Filho, F. T. and Vieira, M. F. Q. (2014). An ontology-driven framework to support scenario representation in a 3D operator training simulator. In *Proc. SIMULTECH*, pages 298–303, Setúbal, Portugal. SciTePress.

Gorecky, D., Khamis, M., and Mura, K. (2017). Introduction and establishment of virtual training in the factory of the future. *International Journal of Computer Integrated Manufacturing*, 30(1):182–190.

Gorecky, D., Loskyll, M., and Stahl, C. (2014). Semantic digital factory–using engineering knowledge to create ontologies for virtual training. *IFAC Proceedings Volumes*, 47(3):7825–7830.

Häfner, P. (2020). Categorisation of the benefits and limitations of immersive technologies for education. In *Proc. MAS*, pages 154–159, Rende, Italy. Cal-Tek.

Häfner, P. (2021). *Holistic Approach for Authoring Immersive and Smart Environments for the Integration in Engineering Education*. PhD thesis, Karlsruhe Institute of Technology, Germany.

Häfner, V. (2019). *PolyVR - A Virtual Reality Authoring Framework for Engineering Applications*. PhD thesis, Karlsruhe Institute of Technology, Germany.

Havard, V., Jeanne, B., Savatier, X., and Baudry, D. (2017). Inoovas-Industrial ontology for operation in virtual and augmented scene: The architecture. In *Proc. CoDIT*, pages 0300–0305, New York. IEEE.

Herbert, B., Ens, B., Weerasinghe, A., Billinghurst, M., and Wigley, G. (2018). Design considerations for combining augmented reality with intelligent tutors. *Computers & Graphics*, 77:166–182.

Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., and Dean, M. (2004). SWRL: A semantic web rule language combining OWL and RuleML. https://www.w3.org/submissions/SWRL/. Accessed: 2023-09-01.

Häfner, P., Häfner, V., Wicaksono, H., and Ovtcharova, J. (2013). Semi-automated ontology population from building construction drawings. In *Proc. KEOD*, pages 379–386, Setúbal, Portugal. SciTePress.

Häfner, V. (2017). Modelling smart virtual environments. In *4. Fachkonferenz zu VR/AR-Technologien in Anwendung und Forschung an der Professur Werkzeugmaschinen und Umformtechnik*, pages 151–162, Chemnitz, Deutschland. Technische Universität Chemnitz. https://var2.org/downloads/Tagungsband-VAR2-2017_digital.pdf.

Häfner, V., Benedix, A.-C., and Häfner, P. (2020). Automatisierung des virtualisierungsprozesses im anlagenbau. *Zeitschrift für wirtschaftlichen Fabrikbetrieb*, 115(3):148–152.

Jacinto, A. S. and Oliveira, J. M. P. (2008). An ontology-based architecture for intelligent tutoring systems. *Interdisciplinary Studies in Computer Science*, 19(1):25–35.

Laine, J., Lindqvist, T., Korhonen, T., and Hakkarainen, K. (2022). Systematic review of intelligent tutoring systems for hard skills training in virtual reality environments. *International Journal of Technology in Education and Science*, 6(2):178–203.

Michels, F. L. and Häfner, V. (2022). Automating virtualization of machinery for enabling efficient virtual engineering methods. *Frontiers in Virtual Reality*, 3:1034431.

Muñoz Merino, P. J. and Kloos, C. D. (2008). An architecture for combining semantic web techniques with intelligent tutoring systems. In *Proc. ITS*, pages 540–550. Springer, Berlin, Heidelberg.

Musen, M. A. (2015). The protégé project: A look back and a look forward. *AI Matters*, 1(4):4–12.

OASIS Standard (2015). MQTT Version 3.1.1. https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html. Accessed: 2023-07-03.

O'Connor, M. and Das, A. (2009). SQWRL: a query language for OWL. In *Proc. OWLED*, page 208–215, Aachen, Germany. CEUR-WS.org.

Pan, J. Z. (2009). *Resource Description Framework*, pages 71–90. Springer, Berlin, Heidelberg.

Richard, K., Havard, V., and Baudry, D. (2021a). Authoring-by-doing: An event-based interaction module for virtual reality scenario authoring framework. In *Proc. AVR*, pages 519–527, Cham. Springer.

Richard, K., Havard, V., His, J., and Baudry, D. (2021b). INTERVALES: INTERactive Virtual and Augmented framework for industriaL Environment and Scenarios. *Advanced Engineering Informatics*, 50:101425.

Ruthenbeck, G. S. and Reynolds, K. J. (2015). Virtual reality for medical training: the state-of-the-art. *Journal of Simulation*, 9(1):16–26.

Schiavi, B., Havard, V., Beddiar, K., and Baudry, D. (2021). Semi-automatic generation of virtual reality procedural scenarios for operation in construction based on 4D building information models. In *Proc. CONVR*, pages 104–111, Middlesbrough, UK. Teesside University Press.

U.S. National Academy of Engineering (2017). NAE grand challenges for engineering. http://www.engineeringchallenges.org/File.aspx?id=11574&v=34765dff. Accessed: 2023-09-14.

Vaughan, N., Dubey, V. N., Wainwright, T. W., and Middleton, R. G. (2016). A review of virtual reality based training simulators for orthopaedic surgery. *Medical engineering & physics*, 38(2):59–71.

Vesin, B., Ivanović, M., Klašnja-Milićević, A., and Budimac, Z. (2012). Protus 2.0: Ontology-based semantic recommendation in programming tutoring system. *Expert Systems with Applications*, 39(15):12229–12246.

Walczak, K., Flotyński, J., Strugała, D., Strykowski, S., Sobociński, P., Gałązkiewicz, A., Górski, F., Buń, P., Zawadzki, P., Wielgus, M., and Wojciechowski, R. (2020). Semantic modeling of virtual reality training scenarios. In *Proc. EuroVR*, pages 128–148. Springer, Cham.

Wen, Y., Zhu, X., and Zhang, L. (2022). CQACD: A concept question-answering system for intelligent tutoring using a domain ontology with rich semantics. *IEEE Access*, 10:67247–67261.

Wicaksono, H., Dobreva, P., Häfner, P., and Rogalski, S. (2013). Ontology development towards expressive and reasoning-enabled building information model for an intelligent energy management system. In *Proc. KEOD*, pages 38–47, Setúbal, Portugal. SciTePress.

World Wide Web Consortium (2013). SPARQL 1.1 Query Language. https://www.w3.org/TR/sparql11-query/. Accessed: 2023-08-25.

Zahabi, M. and Abdul Razak, A. M. (2020). Adaptive virtual reality-based training: a systematic literature review and framework. *Virtual Reality*, 24:725–752.