A UNITY BASED INDUSTRIAL PROCESS SIMULATION CONTROLLED VIA VIRTUAL PLC

David Michalík
Doctoral Degree Programme (2nd year), FEEC BUT
E-mail: david.michalik@vut.cz

Supervised by: Petr Fiedler
E-mail: fiedlerp@vutbr.cz

Abstract: This paper describes the development of an industrial process simulation application that is used for educational purposes. It is based on the Unity game engine and Game4Automation plugin, which offers a base framework and assets to establish the industrial process in a virtual environment. The application is then connected via Modbus TCP to a project created in the CODESYS programming environment, which is also responsible for the simulation of a control system - PLC.

Keywords: unity, game engine, industrial process simulation, virtual plc, education methods

1 INTRODUCTION

Game engines have increased their palette of functions since their creation in the 1990s. They offer a framework for game development, cinematic industry, education and even science. Nowadays, the two most widely used game engines, Unity and Unreal Engine 4, have found their way into the development of scientific applications, such as driving simulators. They offer a practically unlimited development potential of delivering a custom experience to the user. With the pandemic situation in the world as it is, online learning and software simulations have become one of the most used methods for education in universities. In this paper, a method to handle the situation of restricted access to school laboratories is proposed. In the Materials and Methods section, the Unity game engine is briefly introduced with its capabilities, together with its **Game4Automation** plugin by in2Sight GmbH that enables the simulation of industrial process and control. For the control IDE, the **CODESYS** is utilized, for it offers a free alternative of programming IDEs for industrial applications and enables the use of a simulated controller. These two vital components are financially available for students and therefore come with no license policy problems. In the results section, the overview of the developed application is presented.

2 MATERIALS AND METHODS

There are several key components that are used to complement the whole project. The Unity game engine, Game4Automation plugin and the CODESYS development environment.

2.1 UNITY GAME ENGINE

Unity is a game engine by Unity Technologies developed in 2005 and has since become one of the most widely used frameworks for developing 2D or 3D applications, be it by small indie developers or large companies. The applications in Unity3D are based on C# or JavaScript. The Unity3D editor offers a graphical user interface and enables the developers to easily access specific components, objects or assets and program/modify them according to the project's needs. This game engine comes with a proprietary licensing policy. As long as the projects developed are not used for commercial

purposes, Unity Technologies does not claim any revenue produced by the application. Also, the framework is available for educational and personal purposes without any fees [1].

One of the key advantages of the Unity game engine is the community of developers and available assets/plugins. They are easily accessible through the Asset Store and a developer can include these assets or plugins in a project. The pricing of these assets is individual. The main object of a developed application is the Scene. An opened scene can be observed in Figure 1. In this scene, GameObjects can be inserted and are handled through the hierarchy system that specifies parent-child relations to the objects present. GameObjects can be physical objects represented by a 3D model (these can be imported from many widely used 3D modelling programs, such as Autodesk, 3dsMax or Blender), or virtual objects that handle the behavior of some components or even objects (e.g. sky simulation, animations). Each of these GameObjects has several attributes and components, that can be modified or added to change the behavior of the said objects in the scene.

The editor comes with several advanced components and tools, such as the PhysX physics engine, which is a vital component for this paper, for it enables the simulation of collision events and therefore gives a user a simulated representation of real-world object behavior.

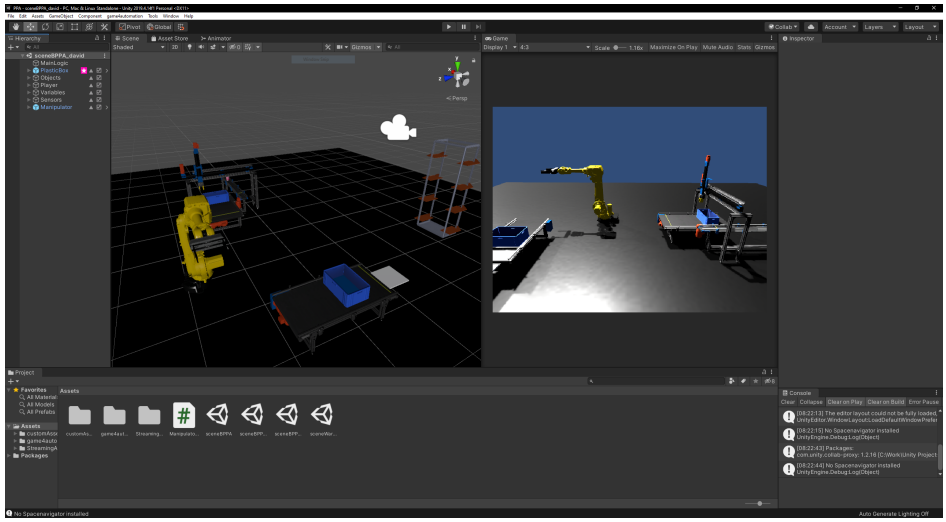

Figure 1: Unity3D Game Engine.

2.2 GAME4AUTOMATION PLUGIN

Game4Automation is a Unity3D plugin (compatible with Unity 2019.4) made by in2Sight GmbH and offers a solution for the simulation of industrial processes. It is a framework for developing a more simplified digital twin of a real project. It includes assets widely used in the industry, such as conveyor belts, manipulators, robots and handles their behavior via the control of several components. The main advantage of this plugin lies in the possibility to connect the simulation in Unity to a controller via communication standard, TCP-IP, Modbus or OPC UA for example.

The features of the plugin include [2]:

- CAD Link - CAD Interface to Import CAD data based on STEP and 3MF format
- Scripts for Interfaces, Signals, Drives, Drive behavior models, Sensors, Picking, Sources, Sink
- TCP-IP Interfaces to Siemens Hardware like Sinumerik, Simatic S7-300, S7-400, S7-1200, S7-1500 and Simotion

- Modbus Master Interface
- OPC UA client interface
- Interface to RoboDK, a virtual industrial robot controller
- Game4Automation base component for easy Scene navigation with the mouse and saving of camera positions

And many more. Full documentation can be found on the developer's website [2]. The plugin comes with multiple licensing options, Starter and Professional, which differ in available features and are paid for, and the Education that is not available for commercial use.

2.3 CODESYS - VIRTUAL PLC SIM

For controlling the simulation, the CODESYS Development System IEC 61131-3 is used. It serves as a software platform for tasks in industrial automation technology. In our case, the main advantage is the option to simulate a control device. Therefore, students can operate the simulation at their homes without the requirement of having a physical device present. The CODESYS platform offers multiple programming options determined by IEC 61131-3, like FBD, LD, IL, ST or SFC. It also supports multiple industrial communication protocols, such as OPC-UA, PROFINET, EtherNet/IP or Modbus TCP, which is used in our case [3].

3 RESULTS

Application overview

An application based on Unity was developed to simulate an industrial process. As seen in Figure 2, there are two main parts to this application. The CODESYS project, which simulates a PLC with a user's control programme, and the Unity application, which serves as a physical simulation of the manufacturing process. The application gives a user visual information about the environment and also specific information about several sensors or devices through virtual variables, which are connected to the PLC sim through Modbus TCP.

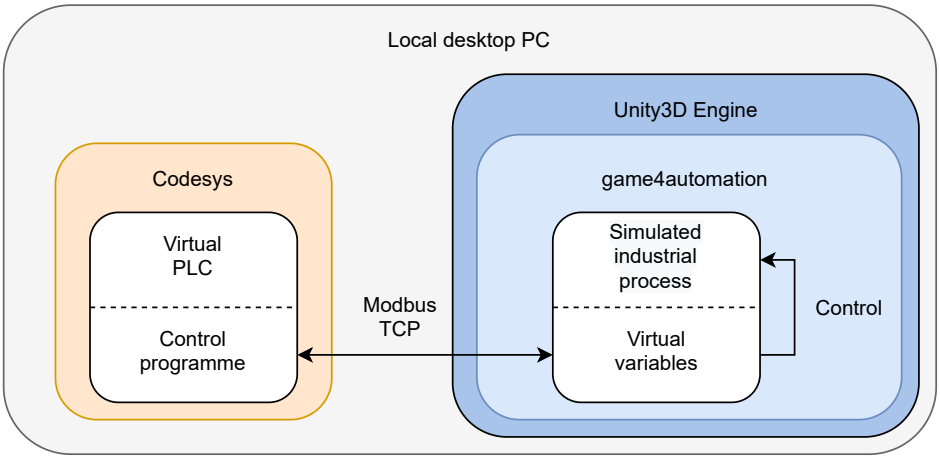

Figure 2: Application diagram

The main goal of this simulation is to replicate an industrial process from the point of view of a system designer and a programmer. It serves an educational purpose to allow students to access physical

devices, such as PLCs, I/O devices or robotic arms. In this way, a proper control requirements of said process can be simulated on any desktop computer with the support of Unity and CODESYS.

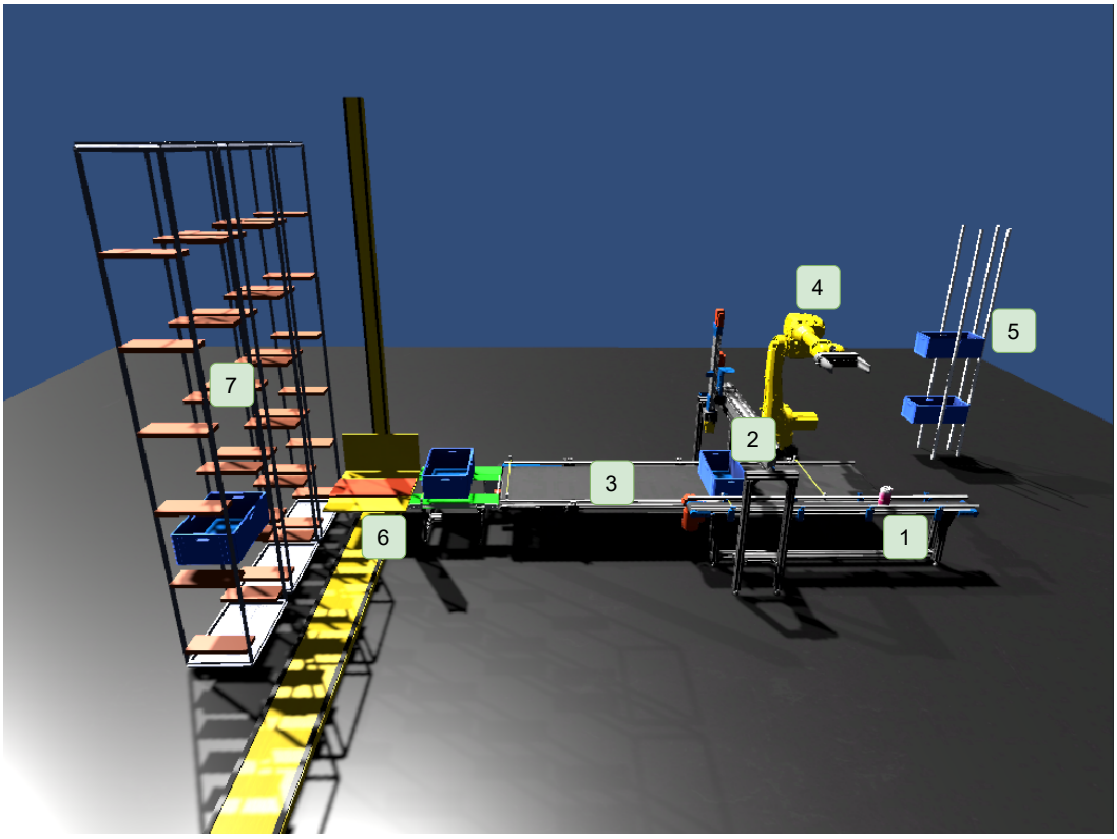

Figure 3: Game4Automation Simulation Scene

Simulated industrial process
As seen in Figure 3, the factory scene consists of several components:

- 1 - Small conveyor belt
- 2 - Manipulator
- 3 - Big conveyor belt
- 4 - Robot manipulator
- 5 - Box storage
- 6 - Storage handler
- 7 - Storage units

These components put together a process in which the product is stacked, handled and stored properly. It is up to the programmer to design a system to control all of these components appropriately. Unity's physical engine renders the physical reality of the world, so if the user does not handle these factors, the process shall fail. At the small conveyor belt, the Source object is present. This object is responsible for generating an item of the selected product. By default, the product is represented by the can model. User can change the model according to their needs. The important component of the

conveyor belt is the Drive script, which handles the movement of the conveyor. A user can control the direction, target position, target speed and the initial start/stop commands through exposed variables. At the end of the small conveyor belt is a Sensory object. The range and cone of this sensor can be manipulated and it returns a boolean value based on the presence of the product.

The robot picks empty boxes from the box storage and puts them on the big conveyor belt. It has multiple positions set via the animation state machine that handles the conditions of state transitions. Each state represents a specific position of the robot's axes. A gripper is attached to the end of the robot. The gripper has a sensor attached with a Grip component that handles picking up the item detected via the sensor. In the next step, the user has to program the behavior of the Manipulator to stack the items inside the box. The Manipulator operates in two dimensions and has a grip attached to it (which works like the robot gripper). Similar to the conveyor belt, the Manipulator is operated via the Drive script. When the items are stacked properly, the big conveyor belt moves the box to the storage handler. This handler has a custom script that controls its movements in three dimensions, enabling a user to program it to move the box with stacked items to a previously specified storage unit position. For variety purposes in students' projects, this position is generated based on their IDs. If the control system operating all of the aforementioned components is set up properly, it should follow the process as explained.

CODESYS control

A list of variables that operate components implemented in the scene is presented to the students. Game4Automation plugin enables the creation of various data types, including bool, float or integer. These can be set to be either input or output variables. Through Modbus TCP, it is possible to map these variables in Unity to variables created in CODESYS project and control the objects via this protocol.

4 CONCLUSION

An application simulating an industrial process has been developed. It serves educational purposes for students to simulate the programmable environment of handling a PLC control and gives them feedback through the application developed in Unity. For that, the Game4Automation plugin is used, which offers a set of assets and functions to make it possible. In the future, the application will be extended with more means of controlling the objects and more scenes that represent different types of manufacturing processes.

5 ACKNOWLEDGEMENT

The completion of this paper was made possible by the grant No. FEKT-S-20-6205 - "Research in Automation, Cybernetics and Artificial Intelligence within Industry 4.0" financially supported by the Internal science fund of Brno University of Technology.

REFERENCES

[1] P. Megha, L. Nachammai, and T. M. S. Ganesan, "3D game development using unity game engine", *International Journal of Scientific & Engineering Research*, vol. 9, no. 3, pp. 1353–1356, 2018.

[2] *Game4automation documentation*, 2019. [Online]. Available: `https://game4automation.com/documentation/current/index.html` (visited on 10/03/2021).

[3] *Codesys online help*. [Online]. Available: `https://help.codesys.com/` (visited on 10/03/2021).