Use case 5:

Additive TiG welding

1. Introduction:

Welcome to this comprehensive training module, designed to guide you through the process of understanding, installing, and implementing a specific module within your system.

In this tutorial, you will find the following sections:

2. System Requirements: Before diving into the training, it's crucial to ensure that your system meets all technical requirements. This section will provide you with information on the required software, hardware tools, operating systems, libraries, and programming languages. Additionally, you will find useful links to documentation and tutorials, as well as sequence and architecture diagrams to help you better understand the module.
3. System Architecture: This section delves deeper into the structure and functionality of the module. You will learn more about sequence and architecture diagrams, and any other information deemed valuable for your specific module. We will also provide code snippets and links to facilitate hands-on learning.
4. Conclusion: In closing, we will express our gratitude for your participation and encourage you to explore the use case lectures to further enhance your understanding of the module and its applications.

By the end of this training, you will be well-equipped to tackle any challenges related to the module and integrate it into your projects.

2. System Requirements:

The robotic TIG welding system is designed for Wire Arc Additive Manufacturing (WAAM) and incorporates a KUKA robot and rotary table. This section will outline the module's technical requirements and the necessary software and hardware tools for successful operation. It should be noted that use case 5 module 2 (Additive TiG welding), has the same hardware and software requirements.

Hardware Infrastructure:
The hardware infrastructure comprises a six-axis KUKA 30-3 robot arm and a DPK400 rotary table, both connected to a KUKA controller version 2 (KR C2). The system is also compatible with KUKA controller version 4 (KR C4). A Fronius MagicWave 5000 welding equipment connects to the PLC, which in turn links to the KUKA controller for laser fence and welding equipment control. The system is managed by an external computer for KUKA system control and another for OPC UA server hosting. These computers can run any operating system (Windows, Linux, Mac) supporting the OPC UA standard. For enhanced security, it is recommended to connect the system to a separate Wi-Fi router, though this is not mandatory. The hardware setup is illustrated in Figure 1.

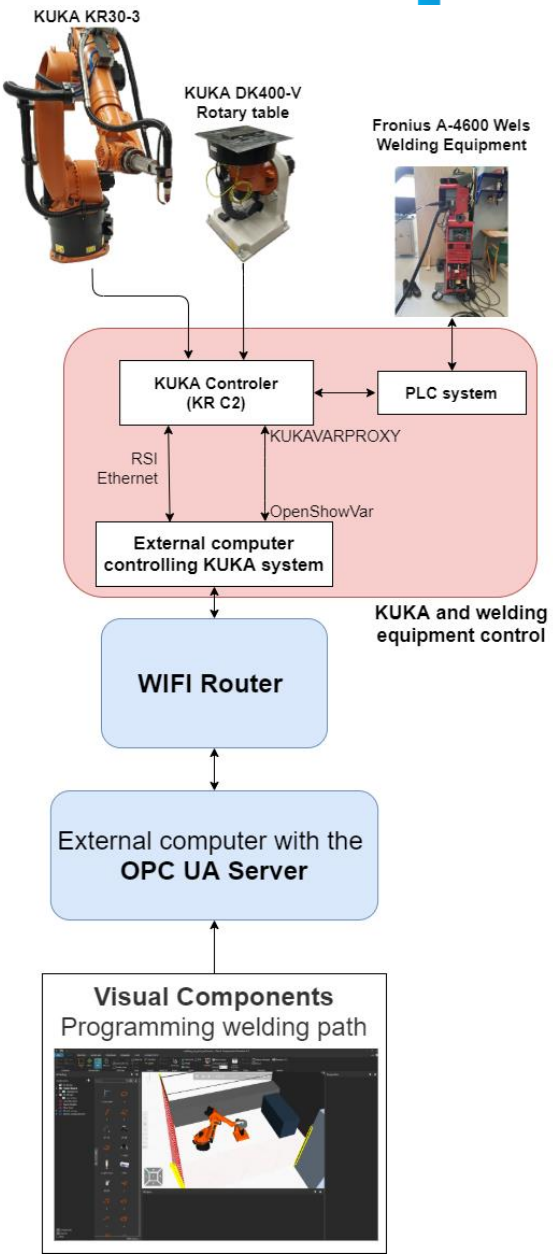

Figure 1. How the welding system is setup and connected.

Software Infrastructure:
Visual Components Premium 4.2+ is used for programming and testing (https://www.visualcomponents.com/products/visual-components/premium/). This manufacturing simulation software features an open Python API for customization and built-in OPC UA connectivity for communication between Visual Components and the KUKA system. A Python version of the OPC UA standard is utilized (https://github.com/FreeOpcUa/python-opcua).

Data from Visual Components is sent to the OPC UA server, which is then translated by a Python program into a format executable on the KUKA controller. The translation program is written in Python 3. Additional software is needed for KUKA robot and rotary table control from an external computer:

Robot Sensor Interface (RSI) for joint and TCP control with a 12ms update rate.
OpenShowVar-KUKAVARPROXY for communication between an external computer and the robot controller (https://github.com/ImtsSrl/KUKAVARPROXY and https://github.com/linuxsand/py_openshowvar).

Cybersecurity:Similar cybersecurity measures apply to this system and the Production flow simulation/supervision system:

Connect robots to a separate Wi-Fi router.
Implement endpoint protection on computers and shield ports.
Restrict computer access via Jump server or Team Viewer, with two-factor authentication.
Use SecurityOnion to monitor computers, including the OPC UA server host.
Employ encryption and signature for OPC UA server communication.
Refer to Figure 2 for an overview of the different cybersecurity levels.

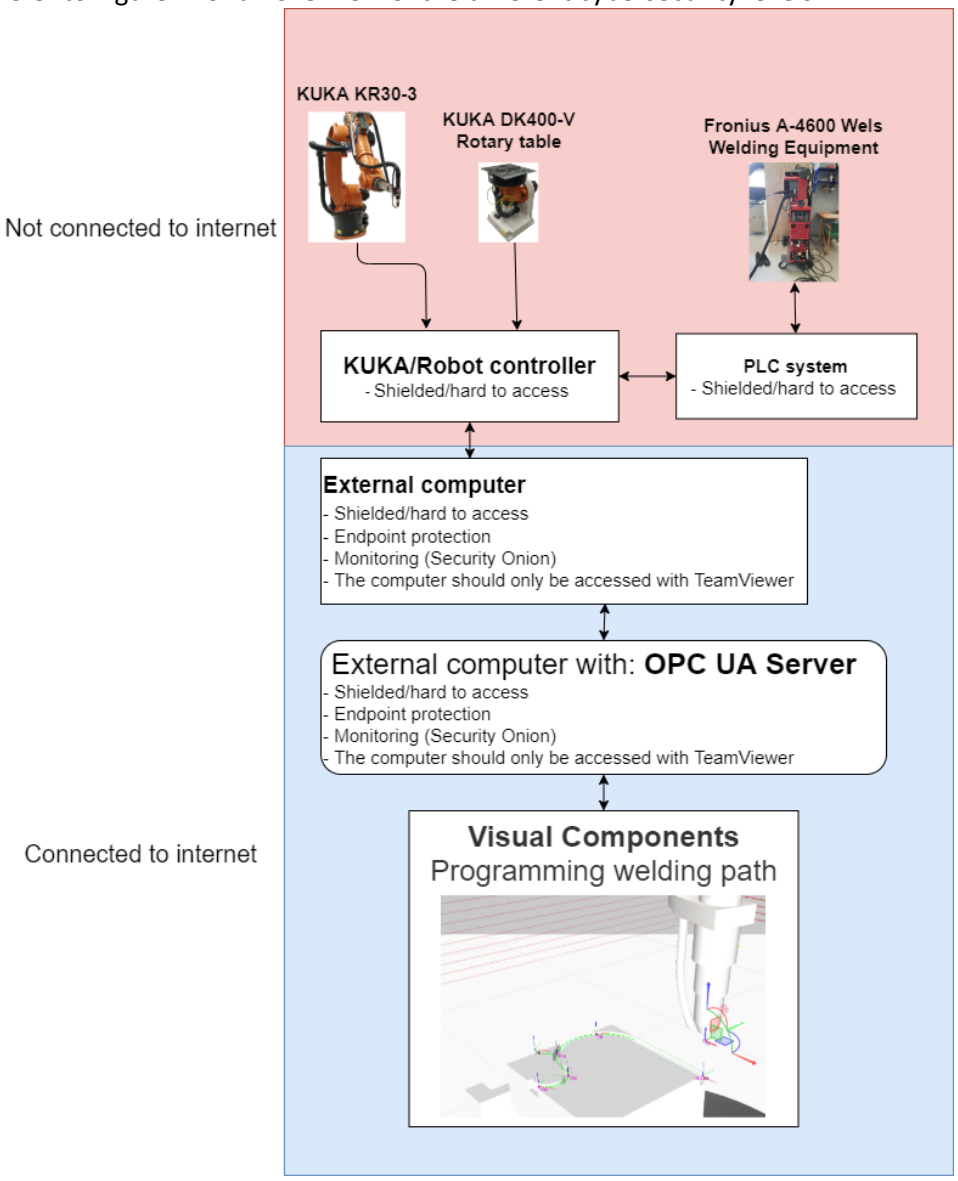

Figure 2. Cybersecurity setup of the welding system

2. System Architecture:

This module builds upon the "simulation welding" module by using the same programming interface for robot additive TiG welding. To implement these functions, it is essential to first complete the "simulation welding" module.

As in the previous module (simulation welding), Visual Components is utilized for simulation due to its user-friendly programming environment and the capability to automatically generate paths.

Start by navigating to the "PROGRAM" tab in Visual Components and creating a subprogram. Within the subprogram, program the path or movement for the weld's bottom layer. Next, specify the number of layers, the distance between each layer, and a time delay to prevent excessive stress on the welding plate. The system automatically generates multiple layers with a specific height increment and layer count, as illustrated in Figure 3.

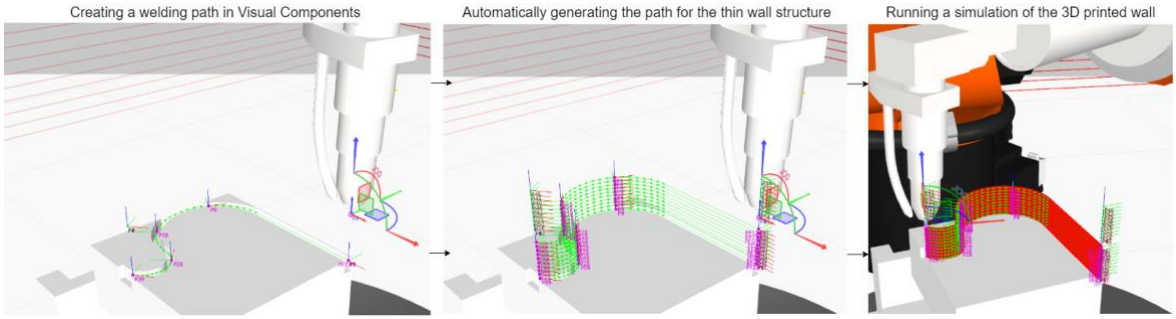

Figure 3: Visualization of automatic generation of multiple layers from a simple path movement

To implement this new function, a second Python script is added to the robot arm, as demonstrated in Figure 4. Note that Python 2 is used within the scripts.

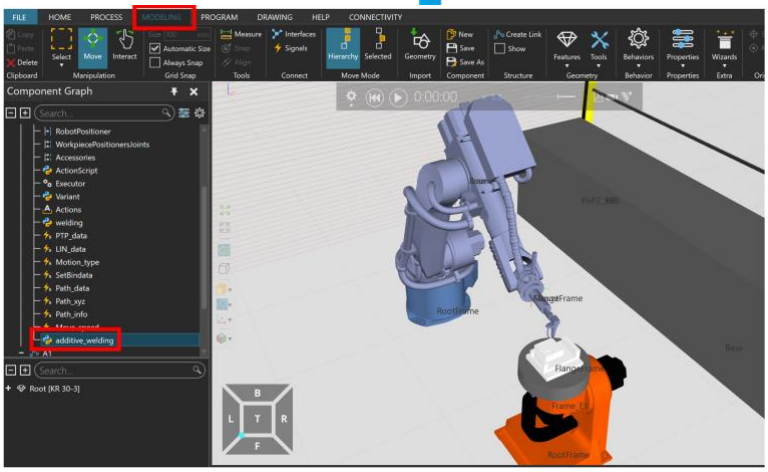

Figure 4: Location of the new Python script within the robot arm

In the new Python script, functions are included that take the robot program from the "PROGRAM" window and generate additional layers at different heights. The new function used for creating more layers is shown in Figure 5.

```
def replicate_printing_statement(input_statement, z_increase):
    app = getApplication()
    kuka_comp = app.findComponent("KR 30-3")
    robotExecutor = kuka_comp.findBehavioursByType("rRobotExecutor")[0]
    robotProgram = robotExecutor.Program

    PrintingRoutine = robotProgram.findRoutine("printing_routine")

    if (input_statement.Type == "SetBin"):
        new_statement = PrintingRoutine.addStatement(VC_STATEMENT_SETBIN)
        new_statement.OutputPort = input_statement.OutputPort
        new_statement.OutputValue = input_statement.OutputValue

    elif (input_statement.Type == "Delay"):
        new_statement = PrintingRoutine.addStatement(VC_STATEMENT_DELAY)
        new_statement.Delay = input_statement.Delay

    elif (input_statement.Type == "PtpMotion"):
        new_statement = PrintingRoutine.addStatement(VC_STATEMENT_PTPMOTION)
        new_statement.Base = input_statement.Base
        new_statement.Tool = input_statement.Tool
        new_statement.JointSpeed = input_statement.JointSpeed

        pos_matrix = input_statement.Positions[0].PositionInWorld

        pos_matrix.translateAbs(0,0,z_increase)

        new_statement.Positions[0].PositionInWorld = pos_matrix

    elif (input_statement.Type == "LinMotion"):
        new_statement = PrintingRoutine.addStatement(VC_STATEMENT_LINMOTION)
        new_statement.Base = input_statement.Base
        new_statement.Tool = input_statement.Tool
        new_statement.MaxSpeed = input_statement.MaxSpeed

        pos_matrix = input_statement.Positions[0].PositionInWorld

        pos_matrix.translateAbs(0,0,z_increase)
```

Figure 5: A snapshot of the code used to generate additional layers

The new program with multiple layers can also be executed in the simulation to ensure everything is functioning correctly, and the estimated shape or result can be observed once the robot finishes running. Furthermore, the simulation can be used to validate robot movement and visualize the weld's appearance. The system utilizes the built-in connectivity function for the OPC UA server to connect Visual Components to the server. As the simulation runs, it automatically sends robot data to the OPC UA server.

The remainder of the system is set up in the same way as in the "simulation welding" module, where the same program reads the "PROGRAM", translates it into a string, and sends it to the OPC UA server. The OPC UA server is also connected to the KUKA controller, where RSI communication manages the KUKA robot and rotary table, and KUKAVARPROXY oversees the welding equipment.

2. Conclusion:

Thank you for completing this comprehensive training module on additive TiG welding using a robotic TIG welding system for Wire Arc Additive Manufacturing (WAAM). In this tutorial, you have acquired a understanding of the necessary hardware and software tools, system requirements, and cybersecurity measures essential for successfully implementing this module. By leveraging Visual Components for programming and simulation, you have also learned how to create multi-layer welding paths for additive manufacturing applications.

We appreciate your participation and encourage you to explore other use case lectures to further enhance your understanding of the module and its applications. As you continue to build your expertise in robotic welding systems, you will discover that the knowledge and skills gained from this tutorial are invaluable for optimizing the performance and integration of these systems into your projects.