**Module name**: Projection-based interaction interface for HRC.

- **Main functionalities**:

*Increase the human operator awareness, safety and capabilities during a human-robot collaboration in a shared workspace. Software module creates and visualize a dynamic safety hull around the robot using a standard projector installed to the ceiling. In addition, user-defined interface components can be created and projected to the workspace.*

- **Technical specifications:**

The overall description of the hardware requirements and the different software nodes in the module are shown in Fig. 1. The workspace is monitored by the Kinect v2 sensor which can be installed at the ceiling overseeing the whole working area. Frame rate of the sensor is 30 Hz. A standard 3LCD projector is used to project the safety hull and the user interface components on the workspace. The projector outputs a 1920×1080 color projection image with 50 Hz frame rate. Due to the short distance from the ceiling to the workspace the physical projection size can be increased by installing a mirror in 45˚ angle to re-project the image to the workspace. The robot is UR5 from the Universal Robot family.

A modified version of *ur_modern_driver* and *univeral_robot* ROS packages are used to establish communication channel between the robot low-level controller and the projector node. Iai-kinect2 ROS package is used to receive data from the Kinect-2 sensor and further transmit it to the projector node. The sensor monitors the usage of the interface components. The projector node is responsible of creating an RGB images of the current status of the workspace for the projector and sending start and stop commands for the robot controller.

Projector, robot and depth sensor are all connected to a single laptop computer that runs the ROS Melodic distribution on Ubuntu 18.04 and performs all computing. Right now only Kinect v2 and Universal Robot 5 are supported.

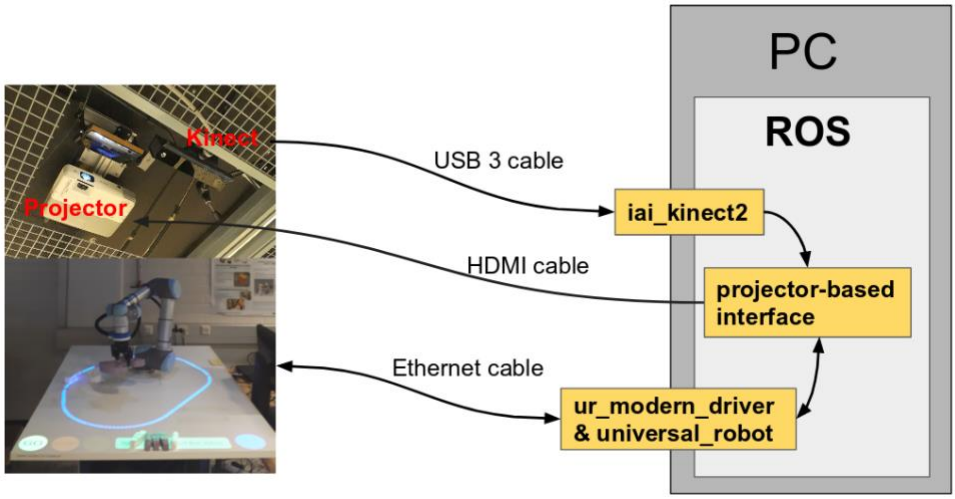


Fig. 1. Module software nodes and the hardware components.

- **Preliminary software configurations:**

Before the module can be used, 1) projector, robot and sensor must be extrinsically calibrated, 2) placement of the user interface components and size of the virtual safety hull has to be defined, 3) the locations of the interface buttons on the workspace must be defined and 4) wired communication link between PC and robot must be established

- **Inputs and outputs:**

All the data is transferred via a standard ROS transport system with publish / subscribe semantics.

In figures 2 and 3 are described the topic names and data formats used by the module. The projector node subscribes to topics */joint_states* and */trinity/system_data* where the robot joint values and the task related data are published respectively. The joint values are used to calculate the shape and position of the safety hull. The text in the information bars and the appearance of other interface components are based on the system data.

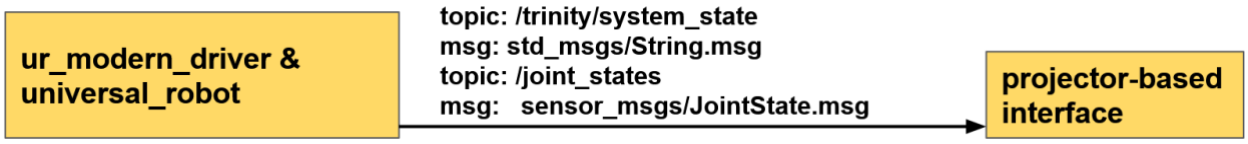


Fig. 2. Data transport layer.

In addition, the projector node subscribes to */trinity/sensor/points* topic where the depth measurements from the sensor are published. The measurements are used to calculate the usage of the interface components (e.g. is a human hand on the stop button)

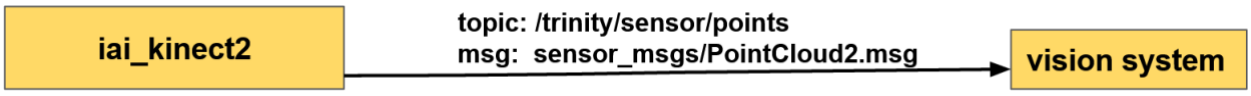


Fig. 3. Data transport layer.

The module output is shown in Fig. 4. An 1920x1080 RBG image is sent to the project and start/stop command are published to the robot controller over */ur_driver/dashbord_command* topic.

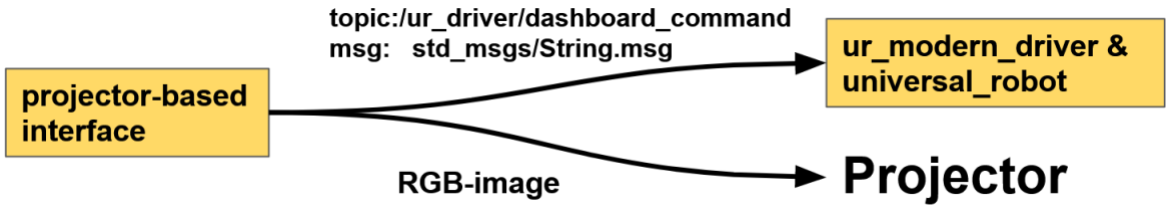


Fig. 4. Data transport layer.

- **Interface specification:**

In Fig. 2 is shown the main interface components of the system. The GO (green) and the STOP (red) buttons can be used to start and stop the robot respectively. The CONFIRM OBJECT (yellow) is used to update the workspace model manually. The CONFIRM (blue) is used simultaneously with the start button to prevent accidental start of the robot. In addition, information bar (big vertical bar) and the safety hull are projected to the workspace to increase the human-operator knowledge about the robot state and the current task.

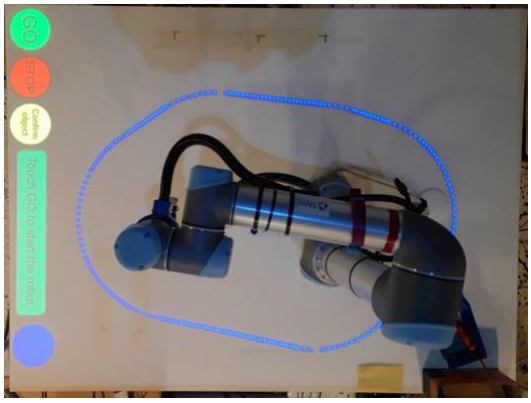


Fig. 5. Projection-based user interface.

The end user interacts with the button by placing her hand on the location where the projected button is located.

- **Formats and standards used**:

*ROS communication layer, ROS-industrial, OpenCV, PCL and C++ and Python standard libraries.*

- **Availability:**

Module library: https://github.com/Herrandy/HRC-TUNI.git

- **Application scenarios:**

*Industrial assembly.*

- **Offered for internal / external use**

*Available for internal/external use.*