Module name: Object detection

- **Main functionalities**:

The object detection module is used to perceive the changing environment and modify systems actions accordingly. The module receives color frames and depth information from a camera sensor and returns information about objects to the robot control. The camera sensor could be placed above the pile of objects as well as at the end-effector of the robot manipulator. The object detection module is mainly responsible for object detection from the bin that can be picked by an industrial robot.

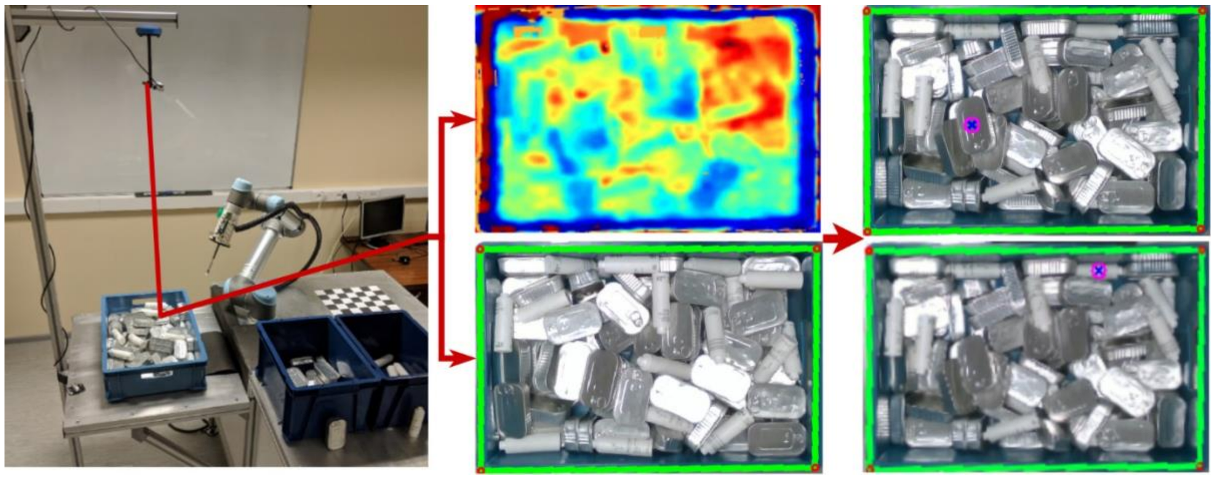

Fig. 1: Module main functionalities

- **Technical specifications:**

The overall system description is shown in Fig. 3. Object detection can be done by using two different detection methods. The first type uses standard computer vision algorithms where functionality is achieved by the OpenCV library. The second method uses CNN-based YOLO computer vision algorithms. YOLO is trained on synthetically generated data sets which are generated by EDI. Steps of generating training data are shown in Fig. 2. Currently, YOLO training is done on the HPC server in approximately 2 days, the trained model can be used on standard desktop PC. YOLO in one frame detects all the objects that have been trained to the model and then chooses the pickable object by the highest confidence rating, then in the same frame object pick position and name are given. The

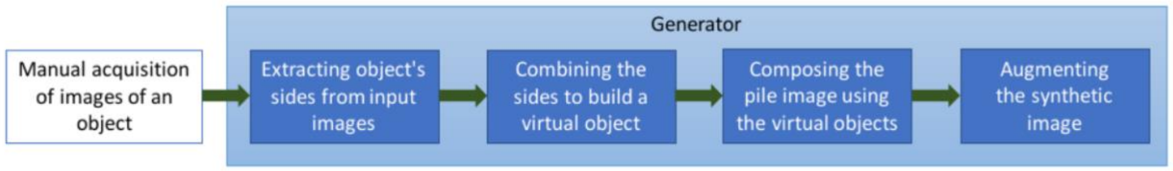
Generator
Manual acquisition of images of an object
Extracting object's sides from input images
Combining the sides to build a virtual object
Composing the pile image using the virtual objects
Augmenting the synthetic image

Fig. 2: Synthetic data generation

YOLO has been tested with simple shape objects (bottle, can), but it can be trained and adjusted for more complex shape objects. The precision of YOLO is about 90%.

The depth sensor is connected to the PC that runs the ROS Kinetic on Ubuntu 16.04. Currently, Intel RealSense d415, d435, Kinect v2 depth cameras are supported, but any camera with ROS driver can be used, if the data can be published as PointCloud2. All the software for this module is implemented using Python 2.7 programming language.

Robot and 3D camera must be extrinsically calibrated.

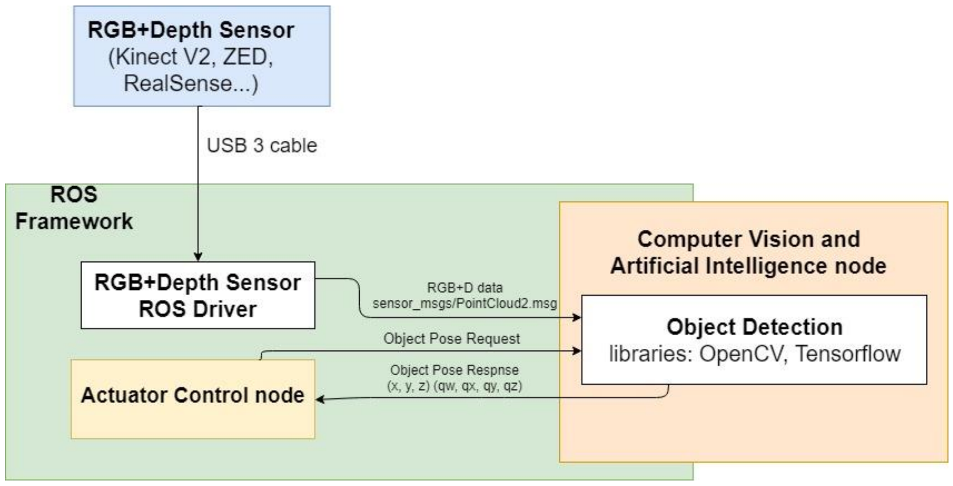
RGB+Depth Sensor
(Kinect V2, ZED, RealSense...)
USB 3 cable
ROS Framework
RGB+Depth Sensor ROS Driver
RGB+D data
sensor_msgs/PointCloud2.msg
Computer Vision and Artificial Intelligence node
Object Detection
libraries: OpenCV, Tensorflow
Object Pose Request
Object Pose Respnse
(x, y, z) (qw, qx, qy, qz)
Actuator Control node

Fig. 3: Module components

- **Inputs and outputs:**

All the data is transferred via a standard ROS transport system with publish/subscribe and request/response semantics. This module subscribes to RGB+Depth sensor data and produces pose of the object: position (x, y, z) and orientation in quaternion format (qw, qx, qy, qz) as a response to ROS service request.

- **Interface specification:**

User can specify the region of interest (bin), where objects should be detected. The detected object location, which is sent to the robot, is shown in the RGB image which is displayed in Fig. 1.

- **Formats and standards used**:

ROS service communication to request the pose of the object.

The sensor data is received from the sensor driver in sensor_msgs.PointCloud2.msg format.

ROS, OpenCV, Tensorflow, PCL, Python standard libraries

- **Availability:**

The module is available with standard detection algorithms. YOLO detection method is available, but there is work on additional functionalities and precision improvement. Repository link can be provided by request.

- **Application scenarios:**

Bin-Picking

- **Offered for internal / external use**

Internal and external