Universiteit van Amsterdam

Rockin Camp 2015

Technical Report

Auteurs:
Tessa Bouzidi
Celeste Kettler
Valerie Scholten
Victor Milewski

Supervisor:
Arnoud Visser

August 24, 2015

Contents

1 Introduction

The RoCKIn Camp is organized with the aim of stimulating progress and innovation of robotics. [1] The camp contains lectures about robotics and a competition. The competition is split into two parts: @Home, which is focussed on the use of robots at home, and @Work, which focusses on industrial robots. As we had worked with the KUKA youBot in our project of "Zoeken Sturen en Bewegen", we were interested to learn more about the opportunities of robotics. Last year (2014) a team from the UvA participated in the RoCKIn Camp in Rome. Because they considered it an informative and useful activity, we decided to go this year for the same reasons.

Figure 1: Setting at the @Work room in Peccioli

RoCKIn Camp is a place where teams can learn about and improve the systems they work with. We participated in the RoCKIn@Work Camp, in which the system we, and all the other @Work teams, used, was the KUKA youBot. The @Work competition aimes to encourage the development of robots that can help in businesses. [2] The @Work competition makes use of the developed technologies of the @Home competition, to make those technologies useful for the industry. This part of the competition can help creating the factory of the future.

The principal question we aimed to answer throughout the whole RoCKIn camp is: How can we make the KUKA youBot able to accomplish some of the functional- and task benchmarks (described in the following section)?

1.1 Functional Benchmarks

The goal of the camp was to achieve as many Functional Benchmarks(FB) as possible. In the @Work competition there are three functional benchmarks which are simple tasks to test how well a participant is in making the system do different skills.

[1]http://rockinrobotchallenge.eu/
[2]http://rockinrobotchallenge.eu/

The first FB is an object recognition task. In an area of 1 by 1 meter an object is placed. The system is required to tell which object it is and it's exact location (Fig. 2). In the second FB the system needs to pick up different items. The robot should be able to control his arm in the correct way, so the items can be picked up decently. The last FB is about control. The system needs to copy a drawing of a simple figure, like the figure eight (or an infinity sign). It needs to move the arm while holding a pencil to draw the figure (Fig. 2).

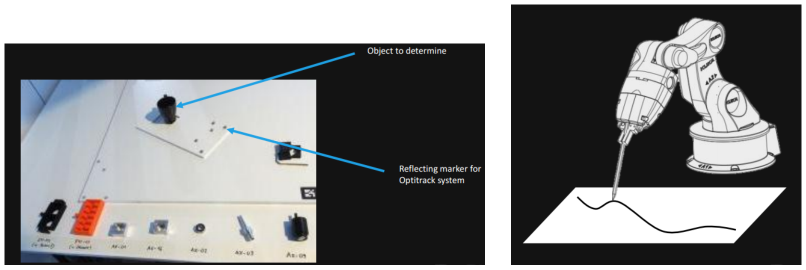


Figure 2: Recognizing objects, functional benchmark 1 (Left) and drawing, functionality benchmark 3 (Right)

1.2 Task Benchmarks

Besides the Functional Benchmarks, was one of the goals of the RoCKIn camp for each team to try to achieve performing in as many Task Benchmarks as possible. There where three tasks each consisting of three Task benchmarks. All of these task could be performed in the @Work arena (Fig. 3). We decided to participate in two of the three tasks, which were the "Assemble Aid Tray for Force Fitting" task and the "Prepare Box for manual Assemble Step" task[4].

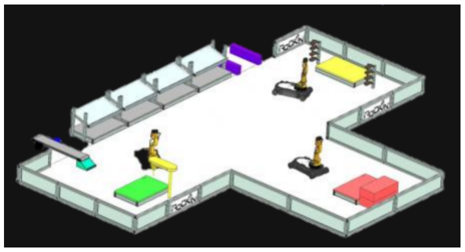


Figure 3: The @Work arena

In the "Assemble Aid Tray for Force Fitting" task, the robot should be able to recognize the correct aid tray using Aruco Code recognition (Task Benchmark 1). An Aruco code is a simplified QR code and is also shown in figure 4. Afterwards the robot should take the bearing boxes and an assembly aidtray out of this box and place it on its own platform to take it with him to the assembly table. There it should be able to put the bearing boxes into the aidtray (Fig. 4) (Task Benchmark 2). The third Task Benchmark is communication with the RefBox. The RefBox is the computer that controls all the machines in the RoCKIn@work arena and also sends a list of objects the KUKA youbot should collect and bring to the assembly bench, in this case it is also called the Central Factory Hub. The connection with this computer is essential for the KUKA youbot to interact with the arena and more will be explained in the section Central Factory Hub.

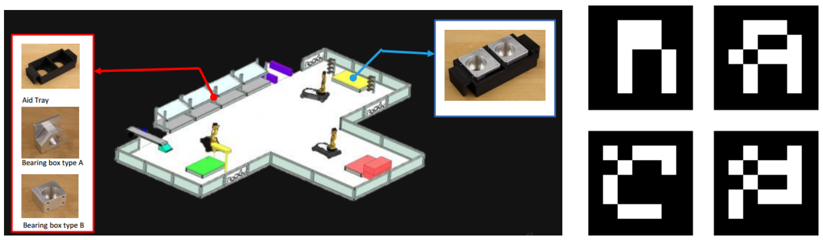


Figure 4: Task benchmark 2 (Left) and four Aruco codes (Right)

In the ”Prepare Box for manual Assemble Step” task, the robot should bring several requested objects to the assembly table and put the parts in the right boxes by using Aruco Code recognition (Fig. 5) (Task Benchmark 1) (Fig. 9). In Figure 5 the box is shown with an QR code instead, because of the former use of QR codes. Task Benchmark 2 is collecting parts for sets of objects. And the third benchmark is the same as the third benchmark from the ”Assemble Aid Tray for Force Fitting” task, which means a connection with the RefBox should be established.

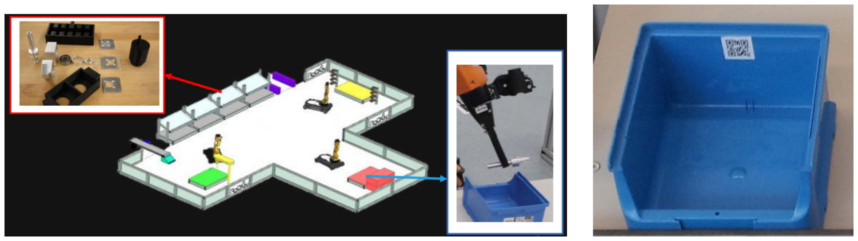


Figure 5: Task benchmark 1 from the second task (Left) and the box with Aruco code (Right)

1.3 Hypothesis

The expectation of the prject is that The youBot will start by receiving orders from the Central Factory Hub. In the begining of the project the youBot will be located at the right workstation, but it still has to find the right shelf or box. With the use of the kinect, the youBot can scan the Aruco codes located on the shelfs and boxes. The youBot will drive along the station until it found its goal. Now it can send a message to the Factor Hub that it has found the object. In the beginning the object will be placed manually on the youBot base to make the start easier. If there are still more orders to handle, the youBot will do this next order until there are no more orders left.

2 Method

2.1 The KUKA youBot

The KUKA youBot is the robot that is used by all competers in de @work competition. This robot has an arm with 5 joints with a gripper to pick up al kind of objects. It also has a platform on omni-directional wheels on which the arm is fixed (Fig. 6). In this platform the KUKA youBot has its own PC running on a 12.04 Linux operating system. Because of its mobility and the plurality of positions the arm can accomplish, this robot can be used in a wide range of tasks. This year a new gripper was used, that creates a better grip than the original gripper (Fig. 6). It is larger and more flexible, but still has a powerful grip. In the next section this year's RoCKIn task benchmarks are described and explained.

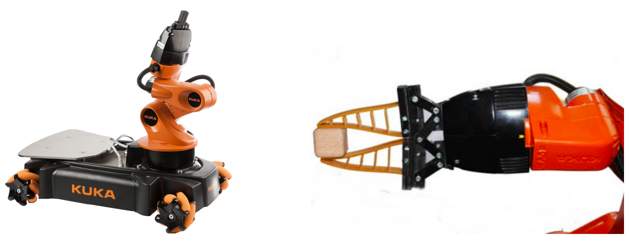


Figure 6: The KUKA YouBot (Left) and the new gripper (Right)

2.2 Connecting the youBot

To make a wireless connection from your own computer to the youBot, it is necessary to have a wifi usb-stick. Make sure that the wifi works on Linux on your computer. It is also possible to make a wired connection, but cables are not provided on the RoCKIn camp and are difficult to use when the youBot is moving around. When the users computer is connected to the youBot, an SSH connetion can be established in a terminal. For each node that is needed to run on the youBot, the user needs to make a new SSH connection in each new terminal.

For the recognition task, a wired connection between (Kinect) camera and the youBot is required. The camera needs to be attached to the youBot on a strategic position, so it is able to see all the objects and the Aruco-codes.

2.3 Logging

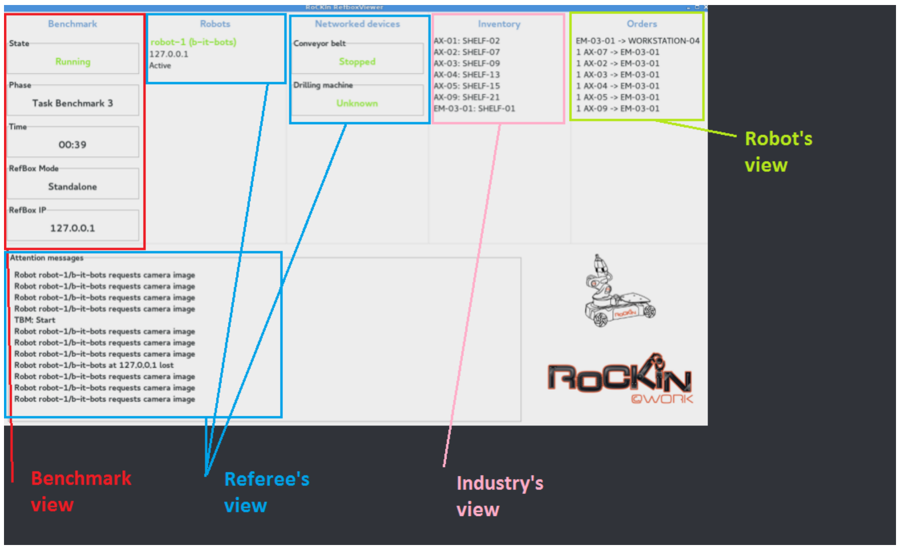
Logging all the data is an important part of the competition. With a right logfile the test can be repeated, which is essential for algorithm development and reproduction of experiments.[3] This helps when comparing multiple runs, to check where the run went wrong, but also for the judge and other teams to replay the run for comparison.

The team of RobOTTO made a rosnode for easy logging, for which the team from UvA gained permission to use. To set the logging up for the robot, only adjusting a couple of files is required. First of all is a config.yaml file in the config directory necessary. In this file the topic can be given names easily, which match the right type of data, that needs to be logged. If a topic is left empty, it will not be logged. Next, the team has to enter a name and the name of the benchmark that will be logged in the recbag.sh file in the bags directory. Finally the rockin_logger.cpp file can be changed in the src directory, to add more topics to config file. After adding them here, they can also be added in the config file.

When the logger is set, the rockin_logger node can simply be launched before the run on the benchmark. All the chosen data will be logged into the bag file.

[3]http://thewiki.rockinrobotchallenge.eu/index.php?title=Datasets

2.4 Central Factory Hub



Benchmark

State: Running

Phase: Task Benchmark 3

Time: 00:39

RefBox Mode: Standalone

RefBox IP: 127.0.0.1

Robots

robot-1 (b-it-bots)
127.0.0.1
Active

Networked devices

Conveyor belt: Stopped

Drilling machine: Unknown

Inventory

```
AX-01: SHELF-02
AX-02: SHELF-07
AX-03: SHELF-09
AX-04: SHELF-13
AX-05: SHELF-15
AX-09: SHELF-21
EM-03-01: SHELF-01
```

Orders

```
EM-03-01 -> WORKSTATION-04
1 AX-07 -> EM-03-01
1 AX-02 -> EM-03-01
1 AX-03 -> EM-03-01
1 AX-04 -> EM-03-01
1 AX-05 -> EM-03-01
1 AX-09 -> EM-03-01
```

Attention messages

```
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots requests camera image
TBM: Start
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots at 127.0.0.1 lost
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots requests camera image
Robot robot-1/b-it-bots requests camera image
```

Figure 7: The different views of the Central Factory Hub

The Central Factory Hub serves as central for production planning, task execution and benchmarking. It has multiple views (Fig 7), the industry's view, the referee's view, the public visitor's view, the benchmarking view and the robot's view. The industry's view is for the warehouse and inventory organization and the tracking of production process. The referee's view has more of a overseeing function, like command the robot to start or stop or to see which robots are available and connected. The public visitor's view is mainly for visitors to see what the task of the robot is and how far the robot has progressed in the task. Benchmarking view is used for scoring, it looks at the efficiency of the robot and keeps track of when the robot performs an action and which knowledge the robot has about his action. The last view is one of the most important views for the ends of our study, and that is the robot's view. The robot gets information about the task, where to find objects and which machines are available.

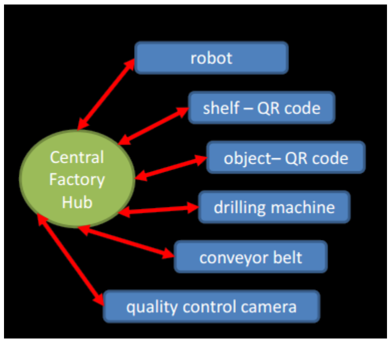


Figure 8: The incoming and outgoing information of the Central Factory Hub

To connect to the Central Factory Hub a wireless connection is needed. This is created by making a peer connection in the node ros_cfh_example. The node is created with an example from the RobOTTO team [2] and the RoCKIn wiki[3]. When this node is runned, a connection is made and with a publisher and subscriber the robot is able to send and receive messages. To let the central factory hub know that the youbot is connected, every second a beacon signal, which contains information like the teamname, is send over the network. Through this connection all kinds of signals can be send for controlling various systems in the factory and handling orders.

The signal handler receives all kinds of messages, but when there is not specified to do anything with them, they will not be reported. It is possible to make the youBot react on specific messages. Most of the received messages are for feedback on the status of systems. This can be used to test if the right messages are received and handled. For example to turn on the conveyor belt, the message has to be sent in a loop, until the conveyor belt status message reports that it is running. Another type of machinery that could be turned on and off was the drilling machine, used for the drilling of faulty plates. This, managing faulty plates and drill them, belongs to the task 'Plate drilling', one we did not attempt to do. But since a connection to the Central Factory Hub should be established, directing the machinery was part of every task.

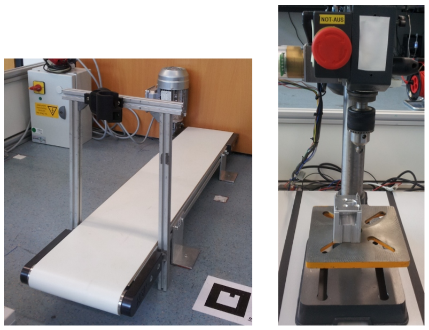


Figure 9: The conveyor belt (Left) and the drilling machine (Right), both part of the factory (the @Work arena)

Right now the order messages are being received and handled. The shelf number of the order is read from the message and stored to be handled.

2.5 Aruco-code Recognition

To recognize the right product, specific Aruco-codes have been placed in front of the products. These Aruco codes are connected to a specific code, which can be matched with the codes sent by the central factory hub.

The Aruco-codes should first be recognized with the camera. In order to achieve this recognition, the camera has to be calibrated (Fig. 10). We used files of Aruco_ros from Github (by pal_robotics)[1], to make a connection from our computer with the Kinect camera and to calibrate the camera. This calibration was done by holding a chessboard in different angles, which gives the KUKA the ability to recognize the Aruco codes from different points of view (Fig. . It is important that the camera can recognize the Aruco-codes from a useful distance.

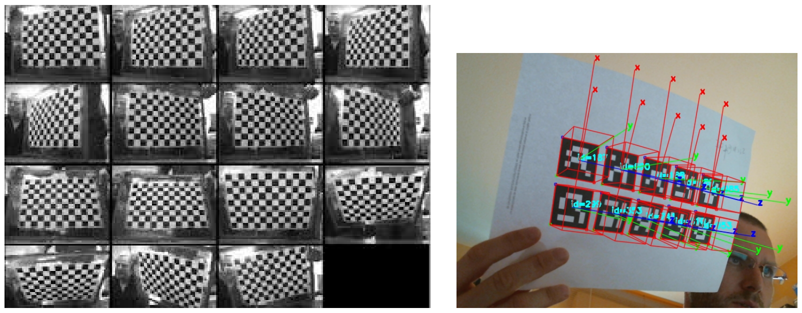


Figure 10: The different positions of the chessboard for calibrating (Left) and the recognition of the Aruco codes when calibrated (Right)

3 Results

As described above separtly, the youBot is able to scan aruco-codes, communicate with the Centrafl Factory Hub and log all data. In the final testrun the youBot was placed on a stationary spot in the arena. It opened an connection with the Central Factory Hub and started receiving the orders. It readed out the first orders object number. To this number was an aruco-code connected. Manually a sequence of codes was held in front of the camera (see figure 11). If the right code was scanned, it printed a message of succes. After this the next order from the Factory Hub was translated to an aruco-code. This kept happening until all the orders where handled.

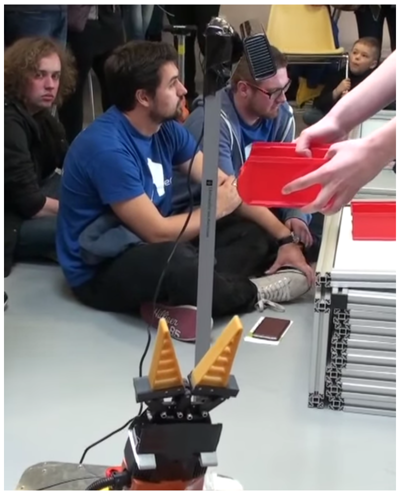


Figure 11: The demo of the program. The camera scans an aruco-code and tries to match it to a received order form the Central Factory Hub

Part of the code to receive messages and compare the order with the scanned shelf

```
void handle_message(boost::asio::ip::udp::endpoint &sender,
                    uint16_t component_id, uint16_t msg_type,
                    std::shared_ptr<google::protobuf::Message> msg){

  std::shared_ptr<AttentionMessage> am; /* shared_ptr on the message type */
  if((am = std::dynamic_pointer_cast<AttentionMessage>(msg))){
  /* test it is the right message type */
    cfh_msgs_example::AttentionMessage attention_msg; /* ros message handling */
    attention_msg.message      = am->message();
    attention_msg.time_to_show = am->time_to_show();
    attention_msg.team         = am->team();
    AttentionMessage_pub.publish(attention_msg);
  }

  std::shared_ptr<Inventory> in;
  if ((in = std::dynamic_pointer_cast<Inventory>(msg))) {
    std::cout << "Inventory received:" << std::endl;

    for (int i = 0; i < in->items_size(); i++) {
      const Inventory_Item &item = in->items(i);
      if(i == k) {
      if (item.has_location()) std::cout << "    In location: "
      << item.location().description() << std::endl;
      location = item.location().description();
      }
    }
  }

}

void markerArrayCallBack(const aruco_msgs::MarkerArray::ConstPtr& msg)
{
std::string shelf = "";
//ROS_INFO("The found markers are:");
for (int i = 0; i < msg->markers.size(); i++) {
ROS_INFO("%i  +  %s", msg->markers[i].id, location.c_str());
switch (msg->markers[i].id) {
          case 81: shelf = "SHELF-01"; break;
          case 82: shelf = "SHELF-02"; break;
          case 83: shelf = "SHELF-03"; break;
          case 84: shelf = "SHELF-04"; break;
          case 85: shelf = "SHELF-05"; break;
          case 86: shelf = "SHELF-06"; break;
          case 87: shelf = "SHELF-07"; break;
          case 88: shelf = "SHELF-08"; break;
          case 89: shelf = "SHELF-09"; break;
          case 90: shelf = "SHELF-10"; break;
          case 91: shelf = "SHELF-11"; break;
          case 92: shelf = "SHELF-12"; break;
          case 93: shelf = "SHELF-13"; break;
          case 94: shelf = "SHELF-14"; break;
          case 95: shelf = "SHELF-15"; break;
          case 96: shelf = "SHELF-16"; break;
          case 97: shelf = "SHELF-17"; break;
          case 98: shelf = "SHELF-18"; break;
          case 99: shelf = "SHELF-19"; break;
          case 100: shelf = "SHELF-20"; break;
```

```
            case 101: shelf = "SHELF-21"; break;
            case 102: shelf = "SHELF-22"; break;
            case 103: shelf = "SHELF-23"; break;
            case 104: shelf = "SHELF-24"; break;
          }
}
if(shelf.compare(location) == 0) {
ROS_INFO("RIGHT SHELF FOUND!");
                ++k;
}
}
```

3.1 Lessons Learned

We have learned a lot at the RoCKIn@Work camp, starting with some colleges we had from several professors. The first real college we got was an instruction from Mattheo Matteucci on how the benchmarking and scoring worked. He also explained the utility of both, and stressed that it was one of the main reasons the RoCKIn camp was so unique. His message was that keeping track of everything the robots achieve in the RoCKIn camp was of use of the whole robotics society, and could help science develop.

The second and at the same time last college was given by Oskar von Stryk, a professor at the TU Darmstadt in Germany. He told the story of a young engineer who designed a robotic arm called the BioRob. This robot was designed with springs, which increased the safety of the robot because it was less static. When someone would be in the way of a KUKA robot arm, it would not give in and therefore it could cause a lot of damage and pain. The BioRob is able to decrease the impact, because it is more bendy and light. The concept of this robot was one in which he and the developper saw a future, but because the lack of salesman, management and business skills it was hard to manage the further development of the robot. One of the main messages of this presentation therefore was: There is a difference between being a good robot developper and being able to distribute it and make it a good and selling product. We think many robot developpers may face this problem, and by addressing it professor von Stryk made the people at the RoCKIn camp more aware of it.

4 Conclusion

A part of the set goal was acheived. The youBot can do all the necessary parts that does not involve movement. With the connection to the Central Factory Hub a very importand part is done. Without this connection the youBut would not be able to handle any assignments and would not start any program. With the detection of the aruco-codes the youBot is able to detect if a destination is reached or an ordered object is found. Also the youBot is able to log all the data it has during doing the benchmarks, which was very important. However not all the goals where reached. The youBot was not yet able to drive or move so the camera would get a good vision on the aruco-codes. There is still a lot of work that needs to be done.

4.1 Future Work

Moving the youBot around is the main goal for the future. With the map of the arena and use of laser scanners, the youBot has to establish its location. With the knowledge of its location it can move around the arena to different stations. Here for it needs some sort of path planning algorithm.

The next step would be picking up the different items. The youbot needs to find the best gripping position on the object. To move to this position with the gripper, two diffent methods can be used. The first, and in the beginning probably the best method, is with the use of a serie of static positions for the joint, so it can pick up the item. The second method is with the use of inverse kinematics.

References

[1] pal-robotics. aruco_ros. `https://github.com/pal-robotics/aruco_ros`, 2015.

[2] RobOTTO. ros-cfh-example. `https://github.com/robotto0vGU/ros-cfh-example`, 2015.

[3] Rockin. Central factory hub @work 2015 camp. `http://rm.isr.ist.utl.pt/projects/rockin-competitions-wiki/wiki/Centralfactoryhub2015Camp`, 2015.

[4] Rockin. Rockin wiki. `http://rm.isr.ist.utl.pt/projects/rockin-competitions-wiki/wiki`, 2015.