1 Introduction

This document provides instructions for using the referee box and rosbag recorder for coordinating and recording data during benchmark executions in the HEART-MET competitions in Metrics.

Four ROS packages are provided, namely:

- `rosbag_recorder`: This package consists of a node with an interface to start and stop recording ROS bagfiles. The topics to be recorded are specified in a `.yaml` file. This node is expected to run on the robot, and the bagfiles will be stored on the robot.

- `metrics_refbox_msgs`: This packages contains definitions of ROS messages which will be used to communicate commands, results of benchmarks etc. This package is required on both the robot and the computer running the refbox.

- `metrics_refbox`: This package consists of a node which provides a UI for setting up benchmark trial configurations, sending commands to the robot to start or stop a benchmark trial and to log the results of benchmark trials. This node must be run on a computer which is on the same network as the robot, and will use the ros master of the robot for communication with the robot.

- `metrics_refbox_client`: This package contains a node which is the 'client' counterpart to the `metrics_refbox`, and runs on the robot. It is responsible for relaying information between the robot and the refbox, and for starting and stopping recording of ROS bagfiles via the `rosbag_recorder`. In addition, this package also contains a node which returns mockup results for each of the benchmarks. This is meant for testing; teams can also look here to see how the result messages should be filled.

2 Rosbag recorder

1. Clone the `rosbag_recorder` package in the robot's catkin workspace and build it.

2. Specify the topics to be recorded in `ros/config/topics.yaml`

3. In `ros/launch/rosbag_recorder.launch` specify the `file_path`. This is the path on the robot's computer where the bagfiles will be stored. Make sure that the path already exists.

4. In `ros/launch/rosbag_recorder.launch` specify the `file_prefix`. The names of all bagfiles will be prefixed with this string. This can be the team name or another string that identifies the robot.

If you are recording a large number of topics, the `timeout` parameter in the launch file may need to be increased.

3 metrics_refbox_msgs

1. Clone the `metrics_refbox_msgs` package both on the robot and the refbox and build it.

4 metrics_refbox_client

1. Clone the `metrics_refbox_client` on the robot and build it

5 metrics_refbox

5.1 Setup

1. Clone the `metrics_refbox` on the refbox computer and build it

2. Set the `ROS_MASTER_URI` environment variable on the computer to point to the ROS master of the robot being tested (for example: `export ROS_MASTER_URI=http://192.168.1.100:11311` where `192.168.1.100` is the IP address of the robot).

3. If necessary, set the `ROS_IP` environment variable on the computer to its own IP address (for example: `export ROS_IP=192.168.1.200`, where `192.168.1.200` is the IP address of the refbox computer). It is only necessary to set the `ROS_IP` if the router you are connected to does not provide name resolution. You can test if this is required by using the 'Test communication' button in the GUI.

4. In the previous step, if you have set `ROS_IP` on the refbox, you must also set the `ROS_IP` on the robot. It should be set to the IP address of the robot.

5. Make sure that the `ROS_HOSTNAME` is set neither on the robot nor the refbox.

5.2 Benchmark configuration

The configurations for each benchmark are defined in `config/`. The `refbox.json` file contains the list of benchmarks and properties for each benchmark, such as the topic and type of the result message.

The configuration for individual benchmarks is in their associated `.json` file. This includes the possible variations for the benchmark, file where trial configurations are stored, and the name of the module and class where benchmark specific actions are performed. All benchmark-specific elements in the refbox GUI are generated based on these configuration files.

In `config/trials`, the configurations for trials for each of the benchmarks is stored. A trial configuration consists of an instantiation of the different variations in a given benchmark. These files are generated by the GUI, and their contents can be edited through the GUI as well.

5.3 GUI

The GUI consists of the following elements (see Figure 1)

1. Team selection: select the team currently running the benchmark

2. Communication test: test whether the refbox and refbox client on the robot can communicate with each other. If a reply is not received from the robot (seen in the Status box), there is a problem with the setup on either the refbox, the robot or both.

3. Benchmark selection: select the benchmark which is being currently executed. The configuration in 5., and the results in 7. will change according to this selection.

4. Trial configuration: select and edit configurations for trials for the current benchmark. A trial configuration consists of an instantiation of

the configurations in 5. You can `Generate` a new random configuration and optionally edit it by first `Unlock`ing the trial configuration. If you edit it, you must click `Save` to save it. Trial configurations can be `Delete`d as well.

5. Current trial configuration: shows the configuration of the current trial. If this is edited, you must Save it to make the changes permanent.

6. Benchmark controls: The `Start` and `Stop` buttons send the corresponding command to the robot (via the refbox client). `Previous` and `Next` can be used to go to the previous or next trial.

7. Results: Once the robot sends back a result for a trial, some fields of the result are shown here. You can add additional notes to the result if necessary (in this case, you must click the `Save` button to save the notes)

8. Status box: various info and error messages are shown here

6 Usage

After completing the setup on both the robot and refbox, follow these steps to start and record benchmark trials.

1. On the refbox, launch:
   `roslaunch metrics_refbox metrics_refbox.launch`

2. On the robot, launch:
   `roslaunch metrics_refbox_client metrics_refbox_client.launch`
   (Note: this also launches the `rosbag_recorder`)

3. Test communication with the `Test communication` button on the refbox. A status message should be printed that you received a reply from the refbox client.

4. Select the team and benchmark to be executed.

5. Generate (and optionally edit) a set of trials to be executed. These trials should remain the same for all teams.

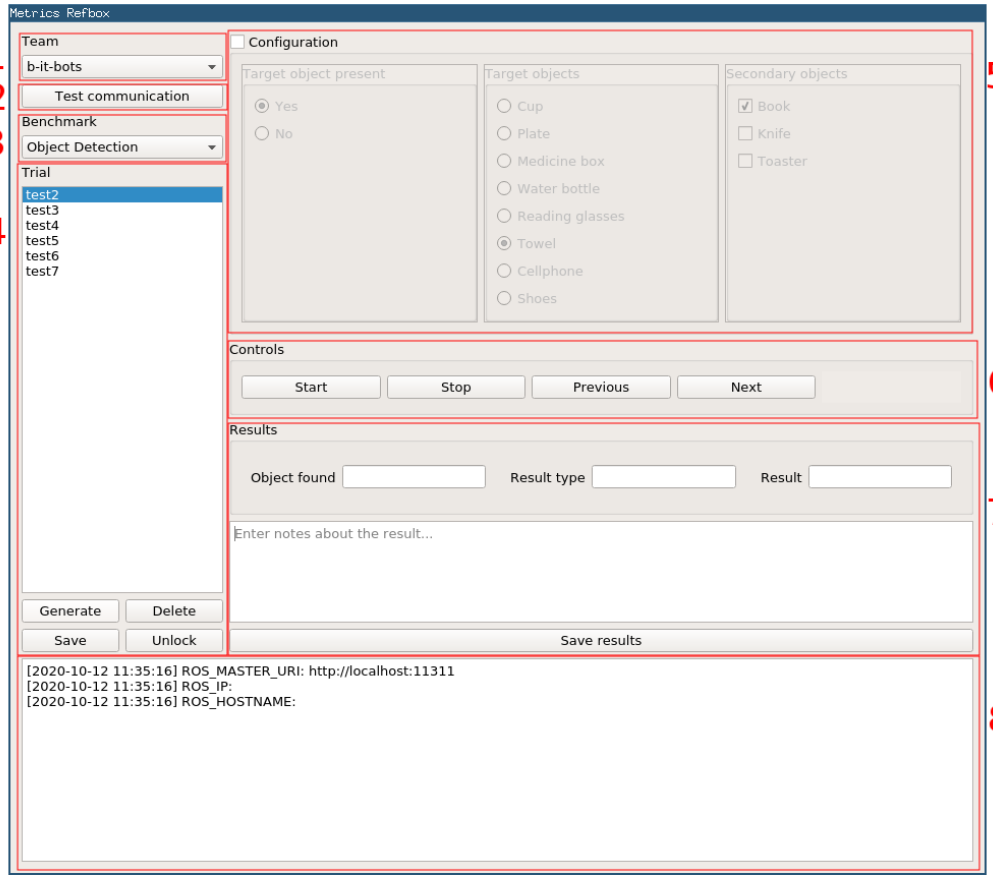
Metrics Refbox
1
Team
b-it-bots
2
Test communication
3
Benchmark
Object Detection
4
Trial
test2
test3
test4
test5
test6
test7
Generate
Delete
Save
Unlock
Configuration
Target object present
Yes
No
Target objects
Cup
Plate
Medicine box
Water bottle
Reading glasses
Towel
Cellphone
Shoes
Secondary objects
Book
Knife
Toaster
5
Controls
Start
Stop
Previous
Next
6
Results
Object found
Result type
Result
Enter notes about the result...
7
Save results
[2020-10-12 11:35:16] ROS_MASTER_URI: http://localhost:11311
[2020-10-12 11:35:16] ROS_IP:
[2020-10-12 11:35:16] ROS_HOSTNAME:
8

Figure 1: GUI

6. Select a trial and click `Start`.

7. Once the start message is confirmed by the client, a timer will begin.

8. In case the start message is not confirmed within a timeout (around 10 seconds), recheck whether communication with the robot and whether the rosbag recorder is running on the robot.

9. When the robot completes the trial, the results should be displayed. Add notes to the result if necessary (for example, if there were penalizing behaviours by the robot), and save.

10. Click `Next` to proceed to the next trial and continue again with step 6.

11. When all the trials have been executed, collect the bagfiles from the robot, and collate them with the results stored in `results/`