USER DOCUMENTATION

Contents

1 Introduction

BLACKBOX is an automated trigger-based reporting, data recording and playback unit, collecting data from robot and manufacturing systems. It takes error reporting and recovery of industrial robot systems to a new level, by developing and utilizing the innovative ROS based framework. The framework is built upon components from the project partner's previous research and existing ROS modules.

Additionally, the graphical multi-platform user interface FlexGui 4.0 with the Technology Readiness Level (TRL) 9 (official Horizon 2020 TRL scale), is already a standard ROS tool to create easy-to-use interfaces to ROS-based robot and software applications.

A part of BLACKBOX will be programming and planning software on TRL6 and will be adapted and exported to general ROS components. In addition, BLACKBOX will use the following ROS components:

- *roscore* for basic messaging
- *rosbridge* for platform-independent communication
- *rosbag* for logging and debugging
- *FlexGui 4.0*

Therefore, knowledge is required:

- ROS
- *FlexGui 4.0*

2 Components

The BLACKBOX project consists of 4 basic elements: (a) BLACKBOX Control, (b) BLACKBOX Main, (c) FlexGui 4.0, (d) BLACKBOX GUI.

(a) Since rosbag is a console application without a UI or offered services to control, this element is responsible for controlling the recording and replaying process.
(b) The main package is responsible for the cloud control – uploading and downloading bag packages – and for the notification and triggering system.
(c) Framework hosting the GUI
(d) Easy-to-use intuitive UI for controlling BLACKBOX.

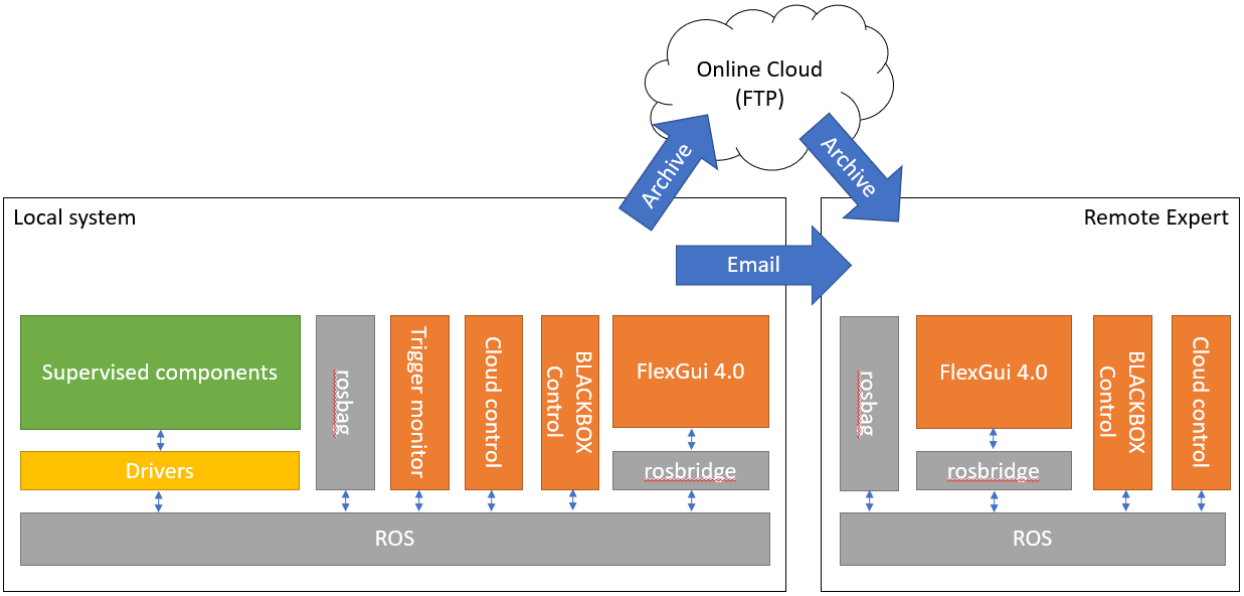

Figure 1 System architecture

3 Installation

In this section you will find information about the system requirements and how to install all of the necesary components to your machine.

3.1 Installation manuals:

- Ubuntu 16.04: https://www.tecmint.com/ubuntu-16-04-installation-guide/
- ROS: http://wiki.ros.org/ROS/Installation
- ROSbridge: http://wiki.ros.org/rosbridge_suite
- FlexGui 4.0: https://www.ppm.no/FlexGui4-Home/Index/downloads
- Apache2: http://httpd.apache.org/docs/2.4/install.html
- Python 2.7: https://tecadmin.net/install-python-2-7-on-ubuntu-and-linuxmint/
- PIP: https://pip.pypa.io/en/stable/installing/
- PIP/Python 2.7 packages: https://packaging.python.org/tutorials/installing-packages/
- VSCode: https://code.visualstudio.com/docs/setup/linux

You can install the required ROS packages with the following commands:

```
sudo apt-get install ros-kinetic-rosbridge-suite
```

Please note, that you will only need VSCode, if you want to modify the code or finetune the settings.

It is strongly advised, but not required to install the addons for python and cpp development for VSCode. You can install these extensions from the Extension Marketplace:

https://code.visualstudio.com/docs/editor/extension-gallery

3.2 Installing BLACKBOX packages

The BLACKBOX has two main packages: control and main. The control controls the rosbag package by providing a ROS interface, the main contains such functionalities, like Cloud Upload, Trigger Monitor, etc.

3.2.1 Prequesites

- Ubuntu 16
- ROS Kinetic
- Git
- FlexGui 4.0

3.2.2 Building BLACKBOX

```
cd ~
mkdir -p blackbox_ws/src
cd blackbox_ws/src
git clone http://rosin.git.ppm.no/control.git
git clone http://rosin.git.ppm.no/main.git
cd ..
```

```
catkin build
```

3.2.3 Running BLACKBOX

Note: your should fill data like usernames, password, server addresses before running this command in the launch file. For the missing data, you will get detailed error messages. If there is mandatory data missing, BLACKBOX will not start.

```
roslaunch blackbox_main startup_min.launch
```

3.2.4 Building the unit tests

Note: only required for the control package.

```
cd ~/blackbox_ws
catkin run_tests
```

3.2.5 Running the unit tests

```
#control package:
cd ~/blackbox_ws/src/control
roscore
rosrun blackbox_control test
#main package:
cd ~/blackbox_ws/src/main
pytest
```

3.2.6 Building the code documentation

```
sudo apt install ros-kinetic-rosdoc-lite -y
sudo apt-get install doxygen -y
pip install kitchen
rosdoc_lite -o ~/blackbox_ws/docs/control ~/blackbox_ws/src/control
rosdoc_lite -o ~/blackbox_ws/docs/main ~/blackbox_ws/src/main
```

After the documentation is built, you can open it with a browser:

```
#control package:
firefox ~/blackbox_ws/docs/control/docs/html/annotated.html
#main package:
firefox ~/blackbox_ws/docs/main/docs/html/annotated.html
```

3.2.7 BLACKBOX GUI

To be able to easily use BLACKBOX, you must include the project file to FlexGui 4.0. Follow FlexGui 4.0 documentation and include the following file from the Main package: **flexgui/project.fgproj**.

4 Configuration

Before starting the system, it is advised and strongly recommended to configure your system, including the Cloud File Storage, Triggers and bag file settings.

The system gets its settings parameters as rosparams, so you can customize each in e.g. a launch file. The sample launch file is included in the main package.

For a detailed description with examples, how to configure each component, please see the component description below.

5 Usage

In this section you will find the necessary info to be able to use the BLACKBOX system. The components are described on a component-level, so you can use them one-by-one. An end-user doesn't need to know the component-level usage, just the GUI frontend.

5.1 BLACKBOX Control

5.1.1 Importing a bag archive

Place the zip archive containing the bag fragments onto the path:

```
~/fglicense/files/black_box_import.zip
```

Currently this path can't be changed without recompiling the source.

Call the service: `/black_box/import` to initiate the uncompress process, that removes all existing files from the bag folder (`~/black_box/bags`) and uncompresses the archive.

The service has no parameters.

5.1.2 Exporting a bag archive

Call the service: `/black_box/export` to start the export process.

The service has 2 input and 1 output parameters:

int32 begin beginning timestamp, beginning from which the bag fragments will be included in the archive.

int32 end ending timestamp, ending with which the bag fragments will be included in the archive.

string file_name file name of the generated archive file.

The file is generated into the folder "`~/fglicense/files`", using the filename "`black_box_export_123456.zip`", where 123456 is replaced by the actual UNIX timestamp.

The bag files in the bags folder are not changed during an export operation.

5.1.3 Starting the playback of the recorded data

Use the service `/black_box/playback` to start the playback. The service has 2 input parameters and 0 output parameters:

bool play determines, if the playback should be started or stopped.

int32 pos sets the start position – the shift in seconds from the beginning of the available range.

By calling the playback service, all other rosbag processes will be terminated, and a new one will be started playing through all available data beginning at the `pos` position.

5.1.4 Get the available range of bag fragments

Use the service /black_box/get_available_range to get the first and last bag fragment recorded available in the bags folder, in unix timestamp format. Please note, that there might be missing parts of the recording, if the computer or the system was not running during a period of time. These parts will be skipped by default. You can change this behavior by modifying the rosbag.sh in the scripts folder.

5.1.5 Receiving the current position of the playback

Subscribe to the topic `/blackbox_time` to receive updates of the current position in the playback. The position is published until the rosbag play command is running. If the process ends or gets terminated, the publish will be stopped. The published information is a unix timestamp recorded during the recording phase.

5.2 BLACKBOX Cloud Uploader

5.2.1 Configuration

The available configuration parameters are the following:

Mandatory:

ftp_host	the address of the FTP server
ftp_username	username of the FTP host
archive_path	the path where system will store the bag files

Optional:

ftp_password	password of the FTP host
smtp_host	to be able to use the email notification service, you must define the SMTP host. The system also can be used with e.g. Gmail.
smtp_username	the system will send its notification in the name of this user
smtp_password	the password for the selected user
email_toaddress	the recipient of the notification
email_body	body of the emails to be sent, the parameters which will be passed to this string are the following: topic, archive name and value in this order. You can use {0}, {1} and {2} in your string to refer to the data.
email_subject	the email will be sent with this subject

Example of the launch file node parameters:

```
<node name="cloud_uploader" pkg="blackbox_main" type="cloud_uploader.py">
   <param name="ftp_host" value="192.168.1.1"/>
   <param name="ftp_username" value="user"/>
   <param name="ftp_password" value="pass"/>
   <param name="smtp_host" value="smtp.gmail.com"/>
```

```
    <param name="smtp_username" value="user"/>
    <param name="smtp_password" value="pass"/>
    <param name="email_toaddress" value="user@gmail.com"/>
    <param name="archive_path" value="$(arg ARCHIVE_DIR)"/>
</node>
```

5.2.2 Uploading and downloading archives

The cloud uploader provides a service, with which we can trigger an upload. The upload can be optionally an email, that contains basic information about the event, and an upload to an FTP server, that contains all information available in a set timeframe.

The node offers two services:

/cloud_uploader/load uploads the archive to the cloud

/cloud_uploader/upload downloads an archive from the cloud to the local machine

Both services have the same `/black_box/Upload` type, but the download is using only the filename parameter.

The parameters of the Upload service type are the following:

string filename filename with extension to upload

string value current value of the topic

string topic topic name which triggered the upload

The upload service will trigger the upload of the archive and an email notification to the selected recipient. If the upload fails, an email is still sent with the error message. If there is no email sending set up, only a warning message is shown. The file upload will be repeated until it succeeds, keeping all upload tasks in a queue. Please note, that stopping the Cloud Uploader will erase this queue, causing the upload tasks to be canceled. The exported files however remain on the hard drive, so you can manually upload them.

5.2.3 Status monitoring

The Cloud Uploader provides information about it's queue through the /cloud_uploader/status topic. Every second a list of UploadStatus objects are published, containing the following information:

bool uploading true, if the file is being (or tried to be) uploaded currently

string error shows any errors (not used for now)

string file_name the name of the uploaded file

5.3 BLACKBOX Trigger Monitor

The trigger monitor is responsible for subscribing to the selected topics defined in the given rosparam.

During each change of the topic the trigger monitor will run the defined expressions and checks if it applies or not. If one of the expressions is true, the trigger monitor will call the Control's export, then the Cloud Uploader's upload service.

5.3.1 Configuration

Mandatory:

string conditions definition of the conditions to be checked. See an example below.

string bag_path path of the bag storage

Optional:

bool on_change_only If true, the condition will only be checked, if the topic value has changed. The default value is false.

int trigger_delay Defines the minimum milliseconds to past between two triggers. The default value is 1000.

int bag_from start the export with the last N seconds counted backwards from the time of the trigger

int bag_to keep recording for M seconds after the trigger. The length of the exported archive will be N+M seconds. This will be uploaded to the cloud.

Example of the launch file node parameters:

```
<node name="trigger_monitor" pkg="blackbox_main" type="trigger_monitor.py">
   <param name="bag_path" value="$(arg BAG_DIR)"/>
   <param name="conditions" value='{"/test":"data.data != 0", "/test2":"data.data > 5 and data.data < 10"}'/>
   <param name="bag_from" value="3"/>
   <param name="bag_to" value="3"/>
</node>
```

5.3.2 Conditions Example

The following value is valid for the condition parameter.

```
{"/ABCD/Variables/Output34":"data.value == \"true\"","/ABCD/Variables/EStop":"data.value == \"false\"", "/welding_driver/current_params":"45 > data.voltage and data.voltage > 20"}
```

Explanation of the example above:

/ABCD/Variables/Output34 *data.value == true*

The trigger applies if the Output34 on a NACHI robot is true (=enabled).

/ABCD/Variables/Estop *data.value == false*

The trigger applies if the ESTOP on a NACHI robot is pressed.

/welding_driver/current_params *45 > data.voltage > 20*

The trigger applies if the voltage is greater than 20V and smaller than 45V.

You can add multiple expressions to a single topic, all you need to do is to group the expressions and use the or operator between them.

5.3.3 Topics

The trigger monitor offers a `trigger` named topic, with `/black_box/Trigger` type. Each time an expression turns true, the Trigger monitor will publish the following parameters:

string filename	output of the Control's export topic, the new filename of the archive
string value	current value of the topic
string topic	topic name which triggered the upload
float32 start	unix timestamp of the beginning of the archive
float32 end	unix timestamp of the end of the archive

5.4 Frontend

After including the BLACKBOX UI to FlexGui 4.0, you will see the following screen. You can start a manual data export – manual trigger – by selecting a time frame in the Export panel and press the Download button.

Pressing the play button will start a rosbag play command and play all of the imported bag files. This means, your system will show the same status as it was recorded on the remote device.

The FlexGui 4.0 manual can be access through the *https://www.ppm.no/FlexGui4-Home/Index/downloads* website.

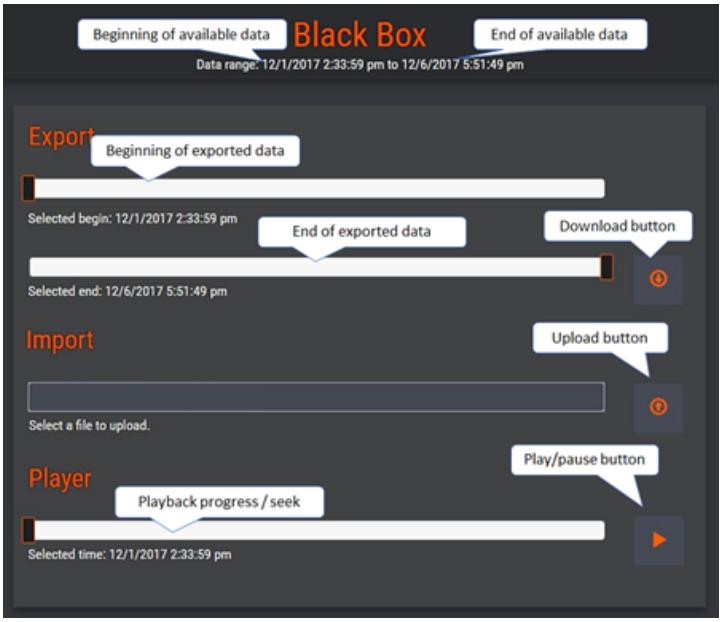

Figure 2 BLACKBOX Control GUI

Exporting is done by first setting the exported timespan: beginning and end. After the timespan is selected, press the download button to receive a zip file, containing all data of the selected timespan. You can save this zip file to a USB drive, send in email, it is a standard zip file.

Importing is done by adding a filename and pressing the upload button. This file is downloaded to the player from the cloud. WARNING: all previous imported or recorded data will be deleted, because only one set of data can be loaded at a time. IMPORTANT: one zip file can be imported any times, so as long as the zip file is not deleted, all data will be available, you just need to import again.

Playback can be started after importing data. The playback button will start playing back all recorded data, and the progress slider will show how time passes. It is possible to pause the playback, and the slider can be used to seek and find a specific point in time.

NOTE: if the personnel having access to this panel are not allowed to stop the recording or delete the recorded data, it might be a good idea to remove the playback and import functions of the panel, as they will stop recording and erase all recorded data.

6 Compatibility

The following drivers and components are verified to be able to work together with the BLACKBOX System. ***Please note***, this is not a complete list of drivers supported by the system, it is possible to use BLACKBOX with other drivers as well as-is.

- NACHI FlexGui 2.4 Driver (*)
- OTC FlexGui 2.4 Driver (*)
- ROSWELD Drivers
    - NACHI Driver
    - Hyundai Driver
    - Fronius WPS Driver
    - OTC WPS Driver
    - IP Camera Driver
    - MEL iLAN Driver
    - uEpsilon Laser scanner driver
- uvc_camera Driver
- usb_cam Driver
- modbus Driver

Please note, the (*) marked drivers are not free software.

7 Acknowledgements

The Focused Technical Project “BLACKBOX – ROS based monitoring system for industrial robot systems with data recording, playback and reporting functionality” is co-financed by the EU project ROSIN (www.rosin-project.eu) and ROS-Industrial initiative (www.rosindustrial.eu).

The FTP’s coordinator and performer is PPM Robotics AS, Norway.

Contact

Prof PhD Trygve Thomessen

Managing Director and BLACKBOX Coordinator

PPM Robotics AS

Leirfossveien 27

7038 Trondheim

Norway

+47 92 23 74 35

info@ppm.no

Supported by ROSIN - ROS-Industrial Quality-Assured Robot Software Components.
More information: rosin-project.eu

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement no. 732287.