Contents

Chapter 1

Source code

Contents

R.E.A.R. was developed and released by Daniele Ferro and Loris Fichera, after a foreign study period spent at the University of Hertfordshire (United Kingdom). Back in our country, an improvement version of *R.E.A.R.* has been written. New code, and this document itself, are based on the primary work done in U.K., presenting the following features in addition:

- working with log data saved from previous *3m.o.r.d.u.c.* session.
  See section *DataLogicLogMorduc*, **??**.
- connecting to real *3m.o.r.d.u.c.* robot to implement an online teleguiding session. See section *DataLogicMorduc*, **??**.
- implementing a new version of *Sweep Metric Algorithm*, to overcame deficiencies detected in performed test.
  See sections *Another sweep metric algorithm* an *AnotherSweepMetricCalc class*, **??** and **??**.

Other minor changes to the original C++ code are occurred, such as source reorganization in new modules; some class has been renamed to better identify its role. Project has been developed and tested with *Debian Lenny (5.0)* and *Debian Squeeze (6.0)*. In the following sections major details about packages and freee libraries utilized are given.
R.E.A.R. is distributed under under *GNU GPL v3* license, documented in [1]. General Public Licenses are designed to give these rights: to run the program, for any desired purpose; to study how the program works, and modify it; to redistribute copies and to improve the program, and release the improvements to the public.

1.1 Download, compile and run *R.E.A.R.*

This section will explain how to download, compile and run *R.E.A.R.*. At last, a brief user guide to using the framework is provided.

1.1.1 Download code

The *R.E.A.R.* project is hosted on *Google Code*:

`http://code.google.com/p/3morduc/`

New *R.E.A.R.* version, covered by this document, can be downloaded at the following URL, together with log files created with *3m.o.r.d.u.c.*'s simulator and from real *3m.o.r.d.u.c.* online session:

`http://3morduc.googlecode.com/files/rear.thesis.tar.gz`

This report itself can be found at:

`http://3morduc.googlecode.com/files/thesis_daniele_ferro.pdf`

The first release of the source code, together with some relative log files, can be obtained by downloading the tar archive named *rear.tar.gz*, from the following URL:

`http://3morduc.googlecode.com/files/rear.tar.gz`

while the related document can be found at:

`http://3morduc.googlecode.com/files/report.14.07.pdf`

1.1.2 Compile code

The instructions exposed in this section are also valid for the first version of *R.E.A.R.*, since it utilizes a restricted set of libraries.
R.E.A.R. is written in C++ and makes use of OpenGL commands. Hence, in order to use it, you need to install a C++ compiler and the C++ standard library, together with an implementation of the OpenGL API and some free libraries.
In particular, *R.E.A.R.* has been developed using GNU *gcc* as C++ compiler (versions 4.3.2, 4.4.1 and 4.4.5) and *Mesa* as the implementation of OpenGL (versions 7.0.3, 7.6.0, and 7.7.1).
A list of other required libraries are:

- `freeGLUT`
  Description: GLUT allows the user to create and manage windows containing

OpenGL contexts and also read the mouse, keyboard and joystick inputs.
Site: `http://freeglut.sourceforge.net/`
Tested with version: 2.4.0, 2.6.0

- `libPNG`
  Description: Official PNG reference library, supporting almost all PNG features.
  Site: `http://www.libpng.org/pub/png/libpng.html`
  Tested with version: 1.2.27, 1.2.44

- `libJPEG`
  Description: Widely used free library for JPEG image compression.
  Site: `http://www.ijg.org/`
  Tested with version: 62
  Note: Since version 62 does not contain `jinclude.h`, which defines few but essential macros, the latter has been included in *R.E.A.R.* source code. This assures compatibility with the stable libJPEG version (62) as well as the most recent ones.

- `cURLpp`
  Description: A C++ wrapper for `libcURL`, a free and easy-to-use client-side URL transfer library, supporting FTP, FTPS, HTTP, HTTPS and other protocols.
  Site: `http://curlpp.org/`
  Tested with version: 0.7.3
  Note: After compiling and installing the library, still not present in Debian repository, you need to add `/usr/local/lib` directory to your library path. For this purpose you can add the directory in `/etc/ld.so.conf`, and the run `ldconfing` as superuser to active new path.

After installing all the previous libraries, *R.E.A.R.* source code can be compiled. In order to make this task easier, a `Makefile` for Unix system is available, assuming source code is stored in `source` folder, whereas object files in - temporanely - `obj` folder.
Accepted `make` commands are:

- `$ make`
  Compile source code and produce `REAR` executable file.

- `$ make clean`
  Delete every object files and the executable one.

- `$ make online`, `$ make log`, `$ make logsim`
  Run *R.E.A.R.* with common parameters, to execute, respectively, an online run with *3m.o.r.d.u.c.*, a run based on previous *3m.o.r.d.u.c.*'s log files, a run based on log created with *3m.o.r.d.u.c.*'s simulator.

`Makefile` can be easily edited to obtain a corresponding Windows based version.

1.1.3 Run *R.E.A.R.*

R.E.A.R. takes several parameters in input, to let user specify with which features the main program must be called.

A complete description of possible options to state is shown with command:

```
./REAR -h
```

while the following general usage is displayed when a right options sequence is not submitted:

```
Usage:  REAR -dl data_logic dl_opt1 [dl_opt2] [-is image_select
is_dist] [-r radius]
```

Let's examine what value each options can assume.
The unique mandatory option is the one defining which type of *DataLogic* instance has to be instantiated. The `-dl` token must always be expressed after the executable file name, followed by *DataLogic* identifier and at least one option.
Based on the value assumed by `data_logic`, one or two options can be specified. All possible combinations are showed in table 1.1.

`data_logic`	`dl_opt1`	`dl_opt2`	`example`
morduc	<IP_or_URL>	<data_path>	-dl morduc 192.168.1.2 ../myfolder
logmorduc	<log_number>	-	-dl logmorduc 5
logsimul	<log_number>	-	-dl logsimul 2

Table 1.1: Possible values for `-dl` options.

By indicating `morduc` as `data_logic`, a instance of `DataLogicMorduc` is created (section **??**). Needed information are the server's IP or URL to connect to and directory path where images and odometric data will be saved as they are received from *3m.o.r.d.u.c.*. If the latter is not specified they are saved in

```
../log_morduc/log_online/
```

starting from the executable file path.
Instead, if `logmorduc` or `logsimul` options are specified, data are retrieved from previously stored image and text files (further details can be found in chapter **??**). In first case logs were created from previous real *3m.o.r.d.u.c.*'s teleguiding session and an instance of `DataLogicLogMorduc` is used (section **??**); in second one logs were created with *3m.o.r.d.u.c.*'s simulator and an object of type `DataLogicLogSimulator` is instantiated (section **??**).
Since both type of log files must be saved in a specific directories and are identified by an integer number, their unique option allows to indicate which set of log files must be used.
After the `-dl` option, user can state other two optional aspects, in any order. One regards what kind of *Image Selector Algorithm* must be used, along with one single options which defines its *optimal distance*, i.e. the distance between robot and

exocentric point of view that ensures the better performance in teleguiding the robot.
As covered in previous sections, three algorithms have been developed, so the total number of possible values for the `-is` option is three, as table 1.2 summarizes.

`image_select`	`is_dist`	`example`
spacial	<opt_distance>	-is spacial 15
sweep	<opt_distance>	-is sweep 20
asweep	<opt_distance>	-is asweep 10

Table 1.2: Possible values for `-is` options.

In order, a `SpacialMetricCalc` instance (section **??**), a `SweepMetricCalc` instance (section **??**) or a `AnotherSweepMetricCalc` instance (section **??**) is created to exploit the image selection algorithm encapsulated in each one. If `-is` is not specifed, the latter class is used as defualt, with optimal distance value set to twentyfive.
Last possible customization can be obtained by stating the *3m.o.r.d.u.c.*'s radius value, used to draw in OpenGL space the three vertical disks composing the robot. After several tests, the most performance value when using data from real *3m.o.r.d.u.c.* robot (retrieved online or from log files) has been proved to be five; instead, if images and data come from simulator, a better values for *3m.o.r.d.u.c.* radius is four.
If not specified with the `-r <radius>` option, robot will be drawn with a radius equal to five.
At last, table 1.3 shows some complete example.

`command`	`description`
REAR -dl morduc 192.168.1.2 -is asweep 25	Connect to real *3m.o.r.d.u.c.* with address 192.168.1.2, draw robot with radius equal to 5 (default) and use *Another Sweep Metric Alogorithm* with optimal distance set to 25.
REAR -dl logmorduc 4 -r 4 -is sweep 10	Use *3m.o.r.d.u.c.*'s log number four, draw robot with radius equal to four and use *Sweep Metric Alogorithm* with optimal distance set to 10.
REAR -dl logsimul 5	Use simulator's log number five, draw robot with radius equal to five (default) and use *Another Sweep Metric Alogorithm* with optimal distance set to 25 (default values).

Table 1.3: Some valid command inputs.

1.1.4 User guide

Bibliography

[1] F. S. Foundation, "GNU general public license," 2011, [Online; accessed 1-March-2011]. [Online]. Available: http://www.gnu.org/licenses/gpl.html (Cited on page 1.)