BIOROB
EPFL Biorobotics Laboratory

Project Guidelines

Version 0.6

In this document we propose some guidelines that should be applied when doing a project at *BioRob*. We also recommend that you refer to the Student FAQ page on the *BioRob* website [1].

1 Final documentation

Your final documentation will include the following items:

- A well written report. Please provide at least one printed copy for the lab and one for your supervisors. You may want to send one or two test versions - not more - to your supervisor and ask your supervisor for comments and corrections. State clearly your achievements and distinguish them from existing resources that you have used. Reference correctly. In the report you should clearly:
  - state the goals of the project
  - make a brief literature review giving the context of research with appropriate references
  - describe the work you have done with a careful analysis of the results
- One CD per printed copy of your final report that you hand over together with the report that includes a digital copy of your written report as PDF together with the source files (most likely LaTeX), possibly movies, source code, design files, a copy of your latest SVN, all weekly reports, all meeting protocols etc.
- A well prepared oral presentation. Please send your slides a couple of days before your presentation to your supervisor. You might want as well to ask her/him for an appointment to give a test presentation. The presentation lasts 20 minutes and is followed by 5-10 minutes of questions. Your presentation should fit within this timing! This is especially important as other presentations will follow yours. Please also check the day before that everything works (beamer, connection of the laptop if you use yours, etc.). The presentation can be done either in English (preferred) or in French. During the presentation you should clearly state the goals of the project and your main results.
- A small website at the student section of the *BioRob* web-page summarizing the work done (small description of the work, videos, link to the report...). Please ask your supervisor to create the page on the website.
- Your SVN directory.

Your final grade will be compiled based on the achievements during your project, the challenges that you have taken on your own, the quality of your documentation and the style of your work (your motivation, your creativity, your enthusiasm).

2 Progress reports

In order to allow your supervisors to follow your work and to facilitate the redaction of the final report, we recommend to write weekly reports during your project. In these reports, you should try to answer three main questions:

- What have I done this week?
- What are the problems that I am currently facing?
- What I plan to do for next week?

These weekly reports do not have to be long and extensive. Instead try to be precise and write in a compact way. You really need to show a critical thinking when presenting your work. For example, do not hesitate to explain the approach you've tried even if they were unsuccessful, and propose some possible explanations.

We recommend also to define a timetable at the beginning of your work and to clearly state the goals of your project. There will be an intermediate presentation and an intermediate report handing around week 7/8 of the project.

You should also pay attention to the external results and material you use during the project: it is perfectly fine to use them as long as you precisely cite the references each time.

Regarding the tools to write your report, we strongly encourage you to use LaTeX instead of word processors (*Word* for example). In any cases, we expect to have the report in *pdf* format (the use of the *doc*, *ppt* and *docx* format is prohibited). For figures, you can use software like *Inkscape* [2] or macro packages for LaTeX like *PGF/TikZ* [3].

3 Version control system

A version control system is a tool to ensure an efficient and reliable tracking of the changes done in a project. For a student project it also allows us to have all the relevant information about the project (weekly reports, code, references,...) in one place. Thus we encourage you to create a repository at the beginning of the project. There exist various control version systems (*CVS*, *Subversion*, *Git*, *Mercurial*,...). As the *EPFL* proposes some nice graphical management tool for *Subversion* we will use this system for the student projects. You should update regularly the svn repository and be careful to keep it clean and well organized. Make extensive use of it so that you have backup files in case of hardware failures or if you have to go back to a previous version in case you find out that your latest developments (code, documents etc.) have not been successful.

4 Coding

4.1 Style

If your project involves implementation you should keep in mind that you're not the only one who will have to read and understand your code. In fact only well documented, stable code that is reusable by your colleagues is a valuable development. If nobody can reuse your developments after you are leaving the group your project was not a success. You are strongly encouraged to look into the document about coding standards [4]. As a general advice, you should try to keep your code as simple as possible.

4.2 Documentation

Even if a well written code can be understood without documentation or comments, we strongly encourage you to write an external documentation for your software and to comment it whenever it's needed. You should use tools such as *Doxygen* [5] to generate the external documentation both in *html* and LaTeX;

4.3 Multi-platform

When implementing something, do not forget that your code should be as portable as possible to the different existing platforms (*Windows*, *Mac* and *Linux*). We recommend the use of a cross-platform build-system (for example *CMake* [6] or *SCons* [7]) to ensure an easy and efficient building of your software.

4.4 Version

When you use some external libraries or tools, don't forget to mention the version you are using (or the version you have tested). Mention also the characteristics of the system on which your code was tested. In theory, the final version of your code should work out of the box (which means all the tools needed should be included or at least explicitly and precisely defined).

5 Licensing

In order to allow us to reuse what you have done during your project, we encourage you to release your work under the terms of the *GNU General Public License* version 2 or later. By doing so, you make possible for others to use, modify and distribute your work. You will find relevant information about the way of applying this licence to your source code in the last section of the coding standards document [4]. More general information about the *GNU General Public License* can be found at [8].

6 Embedded systems

To help you designing and implementing software for embedded devices we provide you with a guide available at [9].

7 Style of work

An important part of the project is that you learn how to solve problems on your own. It will have a high impact on the final evaluation of your work. That is why you should not try to get every problem directly solved by your supervisor but why you should first try to solve them on your own. Try to e.g. have a look into the literature and make a web search. Often your problem was experienced before by somebody else. If you feel that you start spending too much time on solving the problem or if you still feel unsure after your own approach contact your supervisor. Maybe you can even propose several solutions to your supervisor that you can discuss together.

8 Meetings

You should meet your supervisor regularly to discuss current problems and your ideas. Prepare these meetings so that you can actually show your progress. Ideally you send your weekly report to your supervisor a few days before the meeting so that she/he can prepare as well. Take notes during the meeting and send a summary of the things that have been discussed and the decisions that have been made to your supervisor by email. This highly avoids misunderstandings.

9 Citing information sources

You should be very vigilant when referring to the work of other people. All the work that is not yours must be correctly cited (including the source code for software). Every form of plagiarism will be sanctioned by a failure. If you are not sure about how to cite a work, ask your supervisor. Examples of citations (in French) can be found on [10]. We recommend you to check the EPFL Code of Ethics concerning the citing of Information Sources [11].

References

[1] http://biorob.epfl.ch/site/biorob/page-36417.html.

[2] Inkscape. http://www.inkscape.org/?lang=en&css=css/base.css.

[3] Tikz examples. http://www.texample.net/tikz/examples/.

[4] Biorob coding standards, 2009. http://biorob2.epfl.ch/download/pdf/biorob_coding_std.pdf.

[5] Dimitri van Heesch. Doxygen, 1997-2010. http://www.stack.nl/~dimitri/doxygen/.

[6] Kitware. Cmake, 2008-2010. http://www.cmake.org/.

[7] The SCons Foundation. Scons, 2004-2010. http://www.scons.org/.

[8] http://www.gnu.org/licenses/gpl.html.

[9] Rico Möckel. How to design and implement firmware for embedded systems, 2010. http://biorob.epfl.ch/cms/page-36419.html.

[10] http://www.bibliotheques.uqam.ca/InfoSphere/fichiers_communs/module7/regles.html.

[11] https://documents.epfl.ch/groups/p/po/polylex/www/formation_etudes/bachelor-master/Plagiat-ENG.pdf.