Five easy steps to install ***Torch*** on a *Linux* or *Unix* machine

Ronan Collobert

February 4, 2003

1 What do I need ?

You must have:

- A *Linux* or *Unix* machine (a motherboard, an hard disk, a screen,a keyboard and a processor could be interesting).
- A `C++` compiler. You can take for example the *GNU compiler*[1], but other compilers should work.
- *GNU make*[2]. (a version posterior to 1997) (If you don't have it, it's fast and easy to install) (Note that other make program will probably not work)

2 Download the library an unpack archives

The library is available in one big archive. Just go in the download[3] section of the ***Torch***[4] website, and take the *Unix/Linux* archive. Unpack it: a `Torch3` directory should appear, with some directories inside (the sources of the core will be in the `Torch3/core` directory...). The library is divided into several parts: the *core*, which is the foundation of the library (it should be stable...) and some packages developped by any user.

3 Set your compilation options

The library needs a file named `Makefile_options_<os>` to compile, where `<os>` is the name of you operating system, given by the command `uname -s`. This file must be in the `Torch3` directory. There are some examples of this file in the `Torch3/config` directory. Therefore, check the name of your OS with `uname -s` and copy an example of `Makefile_options_*` from the directory `Torch3/config` to `Torch3/`. For example, if you have a *SunOs* system, and you're using the CC workshop compiler, copy the file `Torch3/config/Makefile_options_CC` in the file `Torch3/Makefile_options_SunOs`. After you've copied this file, edit it. It should look like the following (if you are using the *GNU compiler* file example).

[1] `http://www.gnu.org/software/gcc`
[2] `http://www.gnu.org/software/make`
[3] `http://www.torch.ch/downloads.php`
[4] `http://www.torch.ch`

```
# Packages you want to use
# For example if you want to use gradient machines and distributions, put
# "PACKAGES = gradients distributions"
# Don't include packages which contain main programs (such as "examples")
# Don't include "core"
PACKAGES =

# Magik key if you have several makefile
# for the same platform
# (It's useful if you're using two different compilers: a different file
# for dependencies will be generated for each MAGIC_KEY)
MAGIK_KEY =

# Compiler, linker and archiver
CC = g++
LD = g++
AR = ar -rus [new!]

# Command for creating dependencies
# Check the documentation of your compiler if you don't know that...
DEP = g++ -MM [new!]

# Your librairies
# (for example "-lm", but not needed on most systems...)
MYLIBS =

# optimize mode
# Comment one of these lines... OPT is for optimized code,
# and DBG for debug code (if you plan to use a debugger)
DEBUG = OPT
# debug mode
#DEBUG = DBG

# Comment one of these lines... if you take DOUBLE (and comment FLOAT)
# the "real" variables will be "double". Otherwise "float".
# double version
#FLOATING = DOUBLE
# floating version
FLOATING = FLOAT

# Check here the flags for your compiler.
# -DUSEDOUBLE is used to define USE_DOUBLE in the code (for CFLAGS_*_DOUBLE flags).
# -DDEBUG is used to define DEBUG in the code (for CFLAGS_DBG_* flags)
# Debug double mode
CFLAGS_DBG_DOUBLE = -g -Wall -pedantic-errors -DUSEDOUBLE -DDEBUG

# Debug float mode
CFLAGS_DBG_FLOAT = -g -Wall -pedantic-errors -DDEBUG

# Optimized double mode
CFLAGS_OPT_DOUBLE = -Wall -O2 -ffast-math -mcpu=i686 -march=i686 -malign-double -DUSEDOUBLE
```

```
# Optimized float mode
CFLAGS_OPT_FLOAT = -Wall -O2 -ffast-math -mcpu=i686 -march=i686 -malign-double

# Don't care about the end of the file
[...]
```

4 Compile the library

That's easy. Just do `make depend` and then `make`. If you want you can use the option `-j` of the *GNU make* compiler (for example `make -j 4`): it could improve the speed of the compilation, especially on multiprocessor systems. If everything went ok, several directories should have been created: `objs` in which you'll find all the object files, and `lib` in which you'll find the library.
You should'nt have any warning during the compilation... otherwise, send me an email, if it's related to the code of the library.
In fact, the following commands are available for make:

- `make` : compile the library.
- `make clean` : remove the dependency files, the objects and the library for the current system.
- `make distclean` : remove the dependency files, the objects and the library for all the systems.
- `make depend` : create the dependency files.

5 Compile your program

Create a directory in `Torch3/`. Go in this directory, edit your program. Then copy in the directory the `Makefile` that you'll found in `Torch3/config`. After that, if the name of you program is `foo.cc`, just do `make foo` (*without* the `.cc` extension!). A subdirectory will appear (with a name corresponding to your compilation flags and your OS) with the program `foo`...