CMake

Gustav Häger

CMake

- Can generate most kinds of project files
    - Visual studio projects, eclipse or unix makefiles are most common
    - Can be included in version control, to create a more platform independent build system than otherwise
    - Allows for simple compile-time configuration of projects.

Compile a simple example

```
$ g++ main.cpp functions.cpp special_functions.cpp -o awsome_program
```

Linking with external library (like OpenCV)

```
$ g++ main.cpp functions.cpp special_functions.cpp -o awsome_program -I/usr/lib/
opencv_core /usr/lib/opencv_imgproc
```

Kind of a long thing to type every time you need to test something

Makefile example

```
all: awsome_program

awsome_program: useful_functions.o
	g++ main.cpp

useful_functions.o:
	g++ -c utilities.cpp stuff_that_should_be_in_opencv.cpp -L/usr/lib/opencv_core

clean:
	rm *.o awsome_program
```

Compile make:

```
$ make
```

or make [target]:

```
$ make all
```

Makefiles quickly become large and complicated

For example the makefile for python is around 1200 lines, and requires an additional script to be run first.

Better to let CMake generate the makefile

```
CMAKE_MINIMUM_REQUIRED(VERSION 2.8)

PROJECT(example_of_a_cmake_project)

SET(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11 -Wall")


FIND_PACKAGE(OpenCV REQUIRED)

ADD_SUBDIRECTORY(vision_utils)

ADD_EXECUTABLE(track_object tracker_main.cpp)
ADD_EXECUTABLE(test_tracker test_tracker_main.cpp)

TARGET_LINK_LIBRARIES(track_object vision_utils ${OpenCV_LIBS})
TARGET_LINK_LIBRARIES(test_tracker ${OpenCV_LIBS})
```

Using CMake from the terminal

```
$ mkdir build
$ cd build
$ cmake ..
$ make
```

There is also a gui,useful when compiling projects with many options like OpenCV or when using a terrible operating system

CMake can build objects in subfolders:

Bad way (but ok for single files)

```
ADD_EXECUTABLE(track_object tracker_main.cpp vision_utils/cool_functions.cpp)
```

Better way:

```
FIND_PACKAGE(OpenCV REQUIRED)

ADD_SUBDIRECTORY(vision_utils)

ADD_EXECUTABLE(track_object tracker_main.cpp)

TARGET_LINK_LIBRARIES(track_object vision_utils ${OpenCV_LIBS})
```

CMakeLists.txt in vision_utils/

```
ADD_LIBRARY(vision_utils tracker_utils.cpp)
TARGET_INCLUDE_DIRECTORIES(vision_utils PUBLIC ${CMAKE_CURRENT_SOURCE_DIR})

#dlib is special snowflake and needs a million libraries to do the actual work for it
TARGET_LINK_LIBRARIES(vision_utils ${CMAKE_THREAD_LIBS_INIT} X11 jpeg png blas)
```

Variables defined in a file higher in the hierarchy can be used later