# Quick start manual for Building Region Templates (RT) and Running a Simple example

The code has been tested on CentOS, and Ubuntu.

Requirements:

1. Cmake 2.6 or later
2. make – GNU make utility to maintain and compile groups of programs
3. OpenCV 2.4.9 (compile w/ CUDA support it GPU will be used)

<http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.4.9/opencv-2.4.9.zip/download>

1. gcc/g++compiler
2. libeigen2-dev, libpng-dev, libpng++-dev, libtiff 3.9.4-5ubuntu6 (through

ubuntu), OpenMP

1. OpenMPI 1.5
2. CUDA 5.0 (need drivers, toolkit, and SDK from NVIDIA) (Optional – for examples

with CUDA support)

1. NScale library compiled and installed. (Optional – required only if you are building the example Segmentation and Feature computation pipeline)

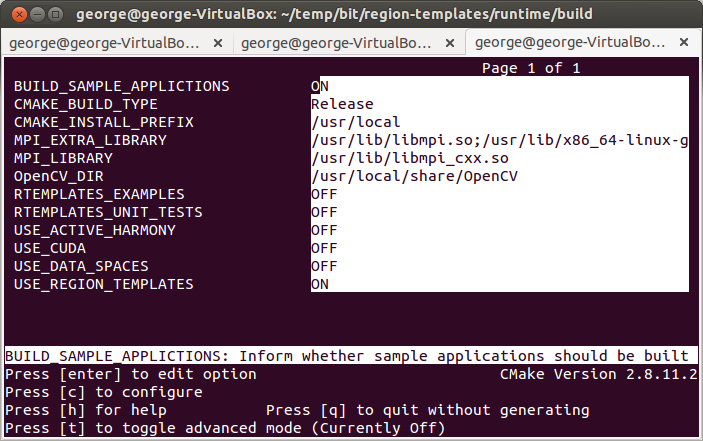
Compilation Steps:

Assuming that the directory where the source code package of RT was extracted is named ROOT, and BUILD\_DIR is the directory where the code will be built:

1. cd BUILD\_DIR

2. ccmake ROOT

2.1 Press 'c' to configure. The resulting screen is shown bellow:



2.2 In this example compilation, cmake found the OpenCV default installation. If it is not the case, you will have to provide it in OpenCV\_DIR.

2.3 Press ‘c’ and ‘g’ in this sequence, to configure and generate the make files. Press ‘q’ to quit ccmake.

3. make

After step 3, the libraries for the RT system will be in folders inside BUILD\_DIR/. In special, check for libruntimesystem.so in BUILD\_DIR/ and libregiontemplates.so in BUILD\_DIR/regiontemplates/

Building the example pipeline

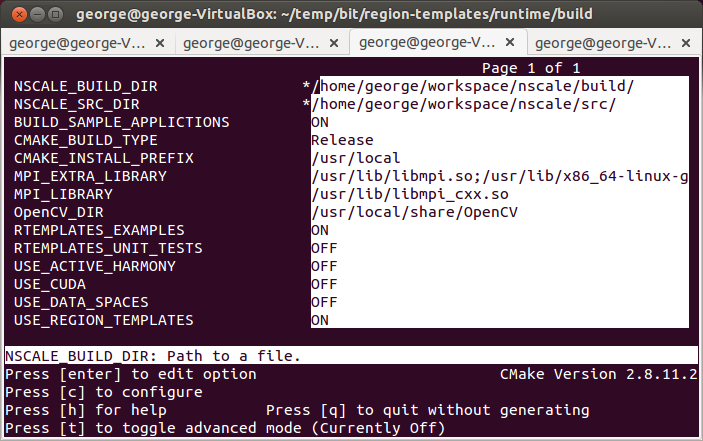
4. Inside BUILD\_DIR/, enter in the ccmake again: cd BUILD\_DIR/; ccmake .

5. Enable RTEMPLATES\_EXAMPLES option: “ON”;

6. Press 'c' to configure;

7. You will be requested to set NSCALE\_BUILD\_DIR and NSCALE\_SRC\_DIR paths to the NScale library (requirement #8).

7.1 NSCALE\_BUILD\_DIR is the directory in which it was built . NSCALE\_SRC\_DIR refers to the directory in which the package was extract + “/src, e.g., home/george/workspace/nscale/src. The example screen bellow shows an example configuration.



7.2 Press ‘c’ and ‘g’ in this sequence, to configure and generate the make files. Press ‘q’ to quit ccmake.

8. make

The example application pipeline will be created in the following folder: BUILD\_DIR/regiontemplates/examples/PipelineRTFS-SF/

Running the example pipeline:

1. cd BUILD\_DIR/regiontemplates/examples/PipelineRTFS-SF/

2. Configure the caching infrastructure used by Region Templates by editing file “rtconf.xml” in the current directory.

This example file is configures RT with a single storage layer using the filesystem. Set the path to a location in your system. If a cluster is used, all nodes must access that location.

3. Run the application

mpirun -n <#procs> PipelineRTFS-SF -i <input folder with images>

Parameters:

1. <#procs>: number of MPI processes running the application. It must be higher than 1, because one process is used as the system Manager and the rest as Workers.
2. <input folder with images>: folder containing a list of input images ("tiff");

OBS: in the current pipeline the Segmentation outputs are being store into the folder configured in the “rtconf.xml” path. It will be modified soon to write all the results to an user selected folder.

# Bibliography

1. Teodoro, G., et al. *Accelerating large scale image analyses on parallel, CPU-GPU equipped systems*. in *Parallel & Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International*. 2012. IEEE.

2. Teodoro, G., et al., *A Fast Parallel Implementation of Queue-based Morphological Reconstruction using GPUs.* Technical Report, CCI-TR-2012-2, Center for Comprehensive Informatics, Emory University., 2012.

3. Teodoro, G., et al. *High-throughput analysis of large microscopy image datasets on CPU-GPU cluster platforms*. in *Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*. 2013. IEEE.

4. Saltz, J.H., et al., *Feature-based analysis of large-scale spatio-temporal sensor data on hybrid architectures.* International Journal of High Performance Computing Applications, 2013. **27**(3): p. 263-272.

5. Teodoro, G., et al., *Comparative Performance Analysis of Intel Xeon Phi, GPU, and CPU.* arXiv preprint arXiv:1311.0378, 2013.

6. Teodoro, G., et al., *Efficient irregular wavefront propagation algorithms on hybrid CPU–GPU machines.* Parallel computing, 2013. **39**(4): p. 189-211.