20.8.2021

Build process - The CHARMing software suite

The core features of the build process are:

- The build process is based on cmake
- All code and dependent library code is available as source code in the `/CHARMing` directory or one of its subdirectories.
- Windows and Linux builds are currently supported.
- For Linux systems an .rpm or .deb file ready for installation is produced by the build system.

Quick way to get the build process working:

1. Use a linux x86_64 machine with docker installed.

2. Copy https://github.com/zweistein-frm2/CHARMing/blob/master/distribution/get_charm.sh to your local Linux machine.

Then: `sudo chmod +x ./get_charm.sh`

3. run `./get_charm.sh`
This will download the latest code from the repository and start the build process (takes a few hours). Don't run the command with sudo , you will be asked for sudo rights at the right moment.

The build process will process all Dockerfile.* files in the `__get_charming/CHARMing/ci` subdirectory, the final output will
be placed in the `__get_charming/CHARMing/distributio`n folder.

FAQ:

- ***How to compile for a different linux distribution?***

  Please add a new Dockerfile.yourlinux to the `CHARMing/ci` subdirectory. Use existing Dockerfile.* and adapt it. Fine-tuning of the build process can be done by placing files in `CHARMing/distribution/yourlinux`, see existing files.

- How to compile witout docker?

  Linux: Please install gcc-9 or higher and cmake 3.14 or higher on your target machine. gtk3 must also be installed (it is by default) for visualization. also check: https://github.com/zweistein-frm2/CHARMing/blob/master/README.BUILD

  Linux, option 2: Open `CHARMing` folder with Visual Studio code

Dipl. Phys. Andreas Langhoff , 089-289-12132 andreas.langhoff@frm2.tum.de

(https://code.visualstudio.com/download)

Windows: Open `CHARMing` folder with Visual Studio 2019
(https://visualstudio.microsoft.com/de/vs/)
Windows, option 2: build from the command line:
start in the `CHARMing` subdirectory:

```
mkdir out // create directory for build
cd out
cmake -DCMAKE_BUILD_TYPE:STRING="Release" ..
cmake --build . --config Release -- /p:CMAKE_INTDIR=""
```

- ***Why are environment variables INSTALL_DEPS and LINUX_FLAVOUR needed on Linux?***

  These variables are needed for adding the proper dependencies in the .rpm or .deb installation file.

- ***How to compile for Raspberry Pi (arm linux 32 bit)?***

  The build process is the same , only difference is that the the address model is 32 bit. There you need to change the file https://github.com/zweistein-frm2/CHARMing/blob/master/buildboost.sh and change around line 16 to address-model=32. You also have to modify the file CHARMing/charm/asio-extensions/include/asioext/detail/impl/posix_file_ops.cpp : Change #36: int64_t -> int32_t .
  Then rebuild the boost libraries.

- ***How to rebuild everything?***

  The boost and opencv libraries (we use static builds here) are build only once. If you need to rebuild them then you have to delete the file REPOSITORY_INIT_OK and run cmake again.