Malta-Imager (Nicola) α-version

Installation, User and Reference Manual

Daniel Muscat

March 13, 2014

Chapter 1

Installation

1.1 Pre-requisites

The mt-imager source code compiles using cmake, make, g++ and nvcc (CUDA compiler). They must be installed on the system prior to installing the mt-imager. Nvcc is part of the CUDA toolkit available form NVIDIA website (https:://developer.nvidia.com/cuda-downloads).

The mt-imager is dependent on the following libraries: Install them prior installing the mt-imager.

- CUDA Toolkit version 5.0: Available from https://developer.nvidia.com/cuda-downloads
- casacore version 1.5. Available from http://code.google.com/p/casacore
- boost version 1.35. Please note that Ubuntu 12.04 LTS does not provide packages for this boost version. It is recommended to install from the source (website: http://www.boost.org)
- log4cxx
- CFITSIO

1.2 Installation

To install the imager untar the mt-imager package. The source untars in a directory called mt-imager-nicola. Change directory to mt-imager-nicola and proceed as follows:

```
cd mt-imager-nicola
mkdir build
cd build
cmake ..
make install
```

The above commands compile and install the mt-imager in the /usr/local. To install in a different location (recommended) set the CMAKE_INSTALL_PREFIX to the desired location. For example if the desired location is /opt/software/mtimager then the cmake command should be as follows:

cmake -DCMAKE_INSTALL_PREFIX=/opt/software/mtimager ..

It is likely that other parameters will need to be set, in particular the parameters BOOST_ROOT and CASACORE_ROOT_DIR

To test the installation run the mt-imager command, found in the bin directory with the parameter -test. If the imager is set up correctly, the string "OK" is printed to standard output and the program exits with a value of 0.

```
cd /opt/software/mtimager/bin
./mt-imager -test
OK
echo $?
0
```

Chapter 2

Using mt-imager

Runtime configuration of the *mt-imager* is done via command line arguments and configuration files. The configuration is a set of parameters defined by keyword and value. This is similar to the *Common Astronomy Software Applications(CASA)* and the standalone *lwimager* tool. The keyword and its respective value are separated by the equals (=) character. For example, *nx=100* defines a parameter named *nx* with its value set to 100. A type such as a string, boolean or integer, is attributed to a parameter value. Boolean parameters can be set to true by putting, a minus (-) sign just before the parameter name. For example, *-test* and *test=true* are equivalent. Refer to section 3 for a list of parameters.

Parameters are set by means of command line arguments or a configuration file. In principle, all parameters can be specified as command line arguments, but this makes the command quite long, and subject to errors. The configuration file helps in reducing long command lines. It is a simple text file parameters are listed on separate lines. Empty lines are allowed, and lines beginning with the number (#) sign are assumed as comments and ignored. Parameters set through the command line have precedence over those defined in the configuration file. This enables the user to have a default configuration stored in a file and partially overridden by command line arguments. The configuration file location is either set by the *conf* parameter as a command line argument or taken as default to *INSTALLATION DIRECTORY*/conf/mtimager.properties. A sample configuration file is available on the default location on install.

Some examples on running the mt-imager are shown below:

Example 1:

```
mt-imager ms=/tmp/msFile.ms outputFITS=/tmp/test.fits
```

Command images the data found in MeasurmentSet /tmp/msFile.ms and outputs the resultant image cube in /tmp/test.fits. All other required parameters are taken from the default configuration file

Example 2

```
mt-imager conf=/tmp/myconf.properties ms=/tmp/msFile.ms
```

Images the data found in MeasurmentSet /tmp/msFile.ms and consult the configuartion file /tmp/myconf.properties for all the other parameters (including outputFITS).

The *mt-imager* uses a logging system based on *Apache log4cxx* API. It requires an extra configuration with a format defined by the API. Its location is configurable through the *logconf* parameter. A sample configuration file is available in the conf directory.

Chapter 3

Reference Manual

conf (string) :

A string pointing to a valid configuration file.

Example: conf=/tmp/myConf.properties

logconf (string) :

A string pointing to a valid properties file to be read by log4cxx

Example: logconf=/opt/software/mt-imager/conf/log.properties

test (boolean) :

When set to true, the imager outputs "OK" and exits. The parameter is to be used to ensure that the imager was compiled correctly and can load all shared libraries.

Example: test=false

info (boolean) :

When parameter is set to true, the mt-imager prints release, copyright and license.

Example: info=false

help (boolean) :

When parameter is set to true mt-imager outputs a help page and exists.

Example: help=false

dataType (string) :

The input data type. Only ms (standing for MeasurementSet) is supported which stands for MeasurementSet.

Example: dataType=ms

ms (string) :

The input MeasurementSet.

Example: ms=/opt/fast/data/measurement.ms

channel (integer) :

Channel to image. If set to -1, all channels are imaged.

Example: channel=0

parallel_channels (Integer) :

Parameter controls channels to image in parallel. Set this parameter to the number of GPUs being used.

Example : parallel_channels=1

psf (boolean) :

When set to true PSFs are generated instead of sky images. This is done by setting all visibilities to unity.

field (Integer) :

For multi-field MeasurementSet, this parameter sets which field to image. Only one field per run is supported. For a single-field MeasurementSet, the parameter is ignored.

Example: field=0

outputFITS (string) :

FITS File where image cube will be saved. Use an exclamation mark (!) at the beginning to allow overwriting.

Example: outputFITS=!/opt/testtoday.fits

image.nx and image.ny (Integers) :

Parameter Specifies the image length and height in pixels.

Example: image.nx=1000 and image.ny=1000

image.lintervalsize and image.mintervalsize (Integers) :

Specify the pixel length/width in arcsec

Example: image.lintervalsize=1 image.mintervalsize=1

mode (string) :

Specifies the algorithm to be used for gridding. Supported values are "normal" and "wproj". wproj enables w-projection

Example: mode=wproj

records (Integer) :

Number of records to load for each run of the gridder. 1000000 is a good value.

Example: records=1000000

taper.operator (string) :

Specifies the tapering operator to use for gridding. Only casagridsf is implemented.

Example: taper.operator=casagridsf

taper.support (string) :

Full width support of the tapering function. Parameter is only queried when mode is set to normal; otherwise it is ignored.

Casagridsf has support of 7. If a larger value is requested the function is zero padded. If less then 7 is requested, the function is truncated.

Example: taper.support=7

taper.sampling (Integer) :

Parameter controls sampling of the tapering function. Parameter is only queried when mode is set to normal, otherwise it is ignored.

Example: taper.sampling=4

wproj.convFunctions.sampling (Integer) :

Sets the sampling factor of the w-dependent convolution functions. Parameter is only relevant when w-projection is activated. When the imager is in normal mode, the parameter is ignored.

Example: wproj.convFunction.sampling=4

wproj.wplanes (Integers) :

The number of wplanes to consider. Parameter is only relevant when w-projection is activated. When in normal mode parameter is ignored.

Example: wproj.wplanes=16

statistics_file.engine (string) :

Engine statistics file in CSV format.

Example: statistics_file.engine=/tmp/engine.csv

statistics_file.gridding (string) :

Gridding statistics file in CSV format.

Example: statistics_file.gridding=/tmp/gridding.csv

statistics_file.main (string) :

Main statistics file in CSV format.

Example: statistics_file.main=/tmp/performance.csv