# Installing the VTF/AMROC Software

July 28, 2011

The VTF/AMROC software is parallel high-performance computing software. Installation is possible on any UNIX system, however, for a first exercise, we assume a typical Linux installation on a single-user workstation or laptop. Recently tested distributions include Ubuntu, Fedora, and RedHat.

It is assumed that the user has the privileges to install system software, meaning, the command `sudo -s` is functional for the user's ID or the password for the `root` user is known. Note that the VTF/AMROC software is a software framework based on source code and as such should *always* be installed under a user ID and never in a system wide directory or under the ID `root`.

## 1 Setting up the system for development

### 1.1 Installing standard UNIX programs

Your system must be set up for comprehensive source code development. Experience has shown that many default Linux distributions nowadays are not. Open the default software installation tool for your distribution (requires the root password). Fedora, for instance, in the graphical `Add/Remove Software` tool, shows a section `Programming`. On my Fedora system, the following packages are activated:

- `autoconf` and `automake`
- binutils: Includes the linker `ld` and the library tools `ar` and `ranlib`
- `gawk`: GNU version of the `awk` utility
- `gcc` (all versions should work). Various GNU compilers
- `g++`. C++ support for `gcc`
- `gfortran`. Fortran 95 support for `gcc`
- `glibc-devel`: Package for development using standard C libraries
- `make`: Tool for built processes
- `python`: Interpreted programming language
- `python-devel`: Libraries and headers for Python development
- Berkeley yacc: This provides the `yacc` parser.
- `flex`: This provides the `lex` tool.

For working with the VTF/AMROC software:

- `gnuplot`: For plotting mathematical data

- `xemacs`: Graphical frontend to the `emacs` editor

After verifying your installation, the following UNIX commands should be available:

- `gcc`, `g++`, `ld`, `ar`, `ranlib`
- `gfortran` or alternatively `f77`/`g77`
- `make`
- `python`, `awk`
- `yacc`, `lex`
- `autoconf`, `automake`
- `gnuplot`

## 1.2 Installing parallel support

VTF/AMROC has full parallel support through the MPI-library. A first exercise can be done without parallel processing/emulation. However, if your distribution provides pre-packaged MPI support, it is recommended to install it also at this point. Fedora already includes the `OpenMPI` libaries:

- `openmpi`: Open Message Passing Interface
- `openmpi-devel`: Development with `openmpi`
- `openmpi-libs`: Libraries used by OpenMPI programs

The following UNIX commands should be available after installing these packages:

- `mpicc`
- `mpiCC` or `mpicxx` or `mpic++`
- `mpirun`

## 1.3 Installing a visualization package

Among others file formats, VTF/AMROC supports binary VTK files. To visualize such files both Paraview and VisIt are available. VisIt is significantly easier to use, while Paraview might be available as a package for your distribution. Fedora allows me to install `paraview` directly as a system-wide application. If your distribution does not contain any of these tools:

VisIT:

1. Download a binary Visit executable from https://wci.llnl.gov/codes/visit
   The version `Linux - x86 32 bit` will work on most 32 bit distributions. Four distribution-dependent binary packages are available for Linux 64 bit systems.
2. Install it system-wide in `/usr/local` or in your home directory, e.g., by executing
   `tar -xvzf visit2_1_0.linux-rhel3.tar.gz`
3. Test the installation by executing `visit2_1_0.linux-intel/bin/visit`

Paraview:

1. Download a binary Paraview executable from http://www.paraview.org
   The version `Linux x86, 32 bit` will work on most 32 bit distributions. `Linux x86, 64 bit` should work on most 64 bit distributions.
2. Install it system-wide in `/usr/local` or in your home directory, e.g., by executing
   `tar -xvzf ParaView-3.8.1-RC1-Linux-i686.tar.gz`
3. Test the installation by executing `ParaView-3.8.1-RC1-Linux-i686/bin/paraview`

# 2 Quickstart with AMROC V2.0

Prepare your system according to Section 2.1. The demonstration of the software during the lecture will start with Section 2.2.

## 2.1 Preparation

1. Set up your system for source code development and scientific visualization. See Section 1 for the tools to install.
2. Download the installation files for the HDF4 libraries used by AMROC from
   http://www.csm.ornl.gov/~r2v/pub/AMR/hdf4_src.tgz
3. Download the source codes for AMROC/VTF from
   http://www.cacr.caltech.edu/asc/wiki/bin/view/Main/SoftwareDownload
   For this exercise, the file AMROC-Clawpack-1.0.tgz is sufficient.

## 2.2 Installation

1. Unpack the file `hdf4_src.tgz` into your home directory, i.e. `cd; tar -xvzf hdf4_src.tgz`
2. Execute the script `build_hdf4.sh` in the new directory `asc`, i.e `cd asc; ./build_hdf4.sh`
3. If the last step was successful you will find the libraries `libdf.a`, `libjpeg.a`, `libmfhdf.a`, `libsz.a`, `libz.a` in `$HOME/asc/hdf4/lib`.
4. Unpack the file `AMROC-Clawpack-1.0.tgz` into `$HOME/asc`:
   `cd $HOME/asc; tar -xvzf AMROC-Clawpack-1.0.tgz`

## 2.3 Compilation and testing

### 2.3.1 Parallel code

If the commands `mpicc`, `mpicxx` are available:

1. `cd vtf`
2. Configure the VTF software for MPI usage providing the location of the libraries in `$HOME/asc/hdf4`:
   `./configure -C --enable-opt=yes --enable-mpi=yes HDF4_DIR=$HOME/asc/hdf4`
   If the commands `autoconf`, `automake` are also available add `--enable-maintainer-mode` to the previous line.
3. Change into the newly created compilation directory. The name depends on the used compiler, e.g., `cd gnu-opt-mpi`
4. Compile the VTF/AMROC libraries: `make`

5. Add the compilation directory to your shell paths: `source ../ac/paths.sh`
6. Optional unit test. Requires the `gnuplot` command.
   (a) Run a sequence of AMROC test simulations on four processors and display results: `../amroc/testrun.sh -m make -r 4 -s`
   (b) Compare the test simulation with stored reference results: `../amroc/testrun.sh -c`

#### 2.3.2 Serial code

Without MPI commands:

1. `cd vtf`
2. Configure the VTF software without MPI usage providing the location of the libraries in `$HOME/asc/hdf4`:
   `./configure -C --enable-opt=yes --enable-mpi=no HDF4_DIR=$HOME/asc/hdf4`
   If the commands `autoconf`, `automake` are also available add `--enable-maintainer-mode` to the previous line.
3. Change into the newly created compilation directory. The name depends on the used compiler, e.g., `cd gnu-opt`
4. Compile the VTF/AMROC libraries: `make`
5. Add the compilation directory to your shell paths: `source ../ac/paths.sh`
6. Optional unit test. Requires the `gnuplot` command.
   (a) Run a sequence of AMROC test simulations in serial and display results: `../amroc/testrun.sh -m make -r 0 -s`
   (b) Compare the test simulation with stored reference results: `../amroc/testrun.sh -c`

### 2.4 Realistic example

Initially only the AMROC libraries and some auxiliary tools are compiled. Specific applications need to be compiled separately.

1. Change into the main compilation directory, e.g., `cd gnu-opt` or `cd gnu-opt-mpi`
2. Change into the compilation sub-directory of the 2D application `SphereLiftOff`:
   `cd amroc/clawpack/applications/euler/2d/SphereLiftOff`
3. Compile the application code: `make`
4. Change into the into corresponding directory with solver.in:
   `cd $HOME/asc/vtf/amroc/clawpack/applications/euler/2d/SphereLiftOff`
5. Execute the application by typing `./run.py` or `./run.py 2` (if you have compiled with MPI on a dual-core system)
6. After the execution, the command `gnuplot Density.gnu` shows the density evolution along the lower boundary
7. Create binary VTK files for VisIt or Paraview for two-dimensional graphical visualization by running: `hdf2tab.sh "-f display_file_visit.in"`

8. Execute VisIt or Paraview as sketched in Section 1.3 and load the newly produced VTK files for display.

To compile all applications for AMROC's Clawpack solver:

1. `cd $HOME/asc`
2. `cd gnu-opt` or `cd gnu-opt-mpi`
3. `cd amroc/clawpack/applications`
4. `make`

A fluid-structure interaction application using a beam element solver is available in `vtf/fsi/beam-amroc`. Note that this code can only be compiled and executed when MPI is available.

1. `cd $HOME/asc/gnu-opt-mpi/vtf/fsi/beam-amroc/VibratingBeam`
2. `make`
3. `cd $HOME/asc/vtf/fsi/beam-amroc/VibratingBeam`
4. `./run.py 4`
   Note that the `LastNode` entry in `solver.in` needs to be changed for executing on a different processor number. The last processor is always dedicated to the solid solver.
5. `hdf2tab.sh`
6. Execute VisIt or Paraview and load the VTK files for visualization.

For further documentation see http://www.cacr.caltech.edu/asc