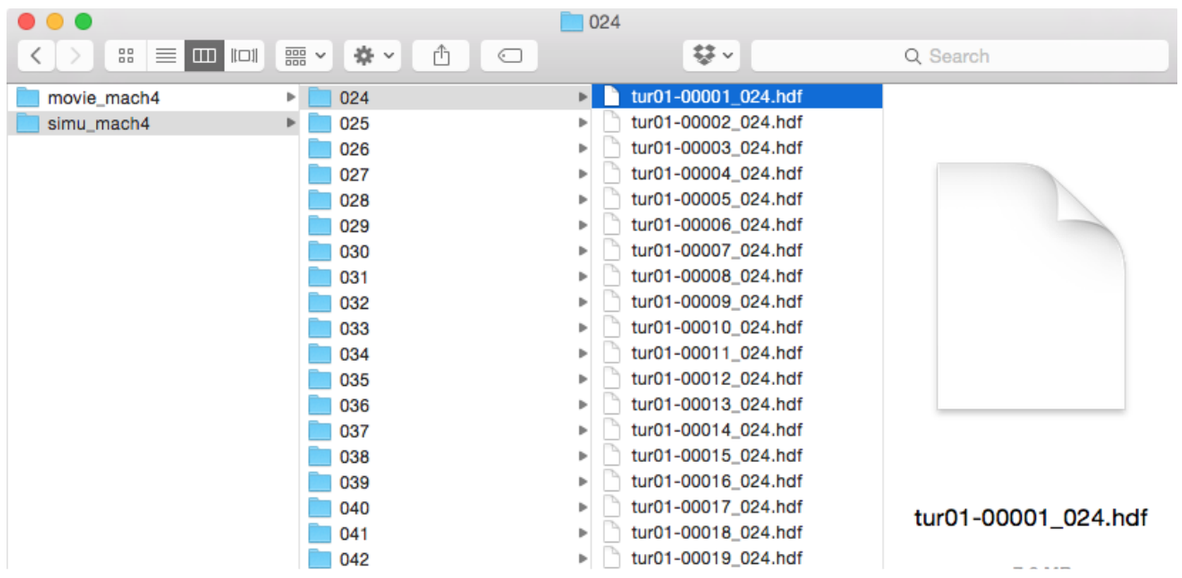
unfold

production example
visual material::shockwave simulation data

1. original simlation data

shockwave simulation
voxel size : 400 x 400 x 400
total simulation frame : 94 frame (024-094)
file format : hdf5
parameter : density, velocity, energy, ionization
file per voxel : 50 x 50 x 50
number of files per frame : 512 files

1 simulation frame is splited into 512 files.
In total 94 frame x 512 = 48,128 file.



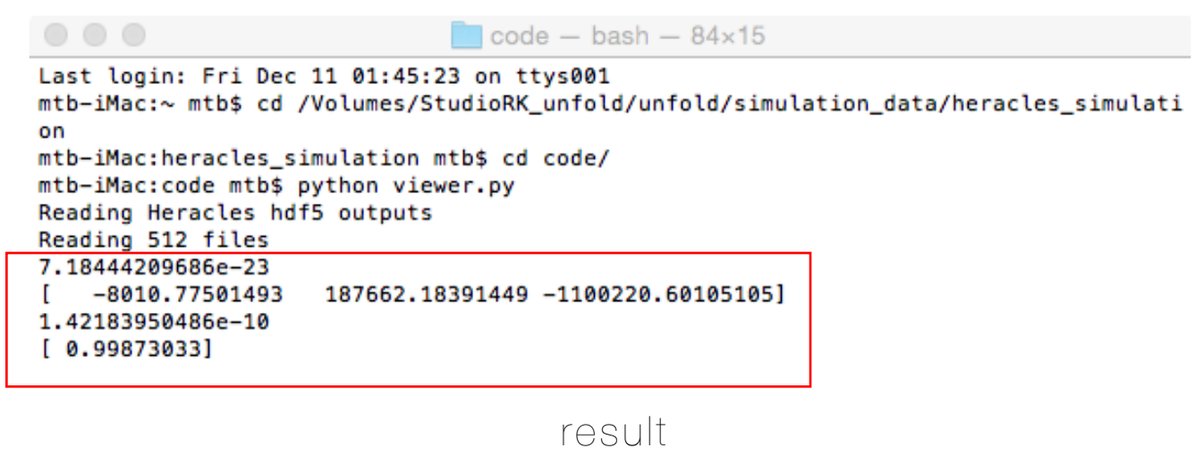
example HDF5 files

2. simulation data example

rho : density
u : velocity(x,y,z)
E : energy
fx : ionization

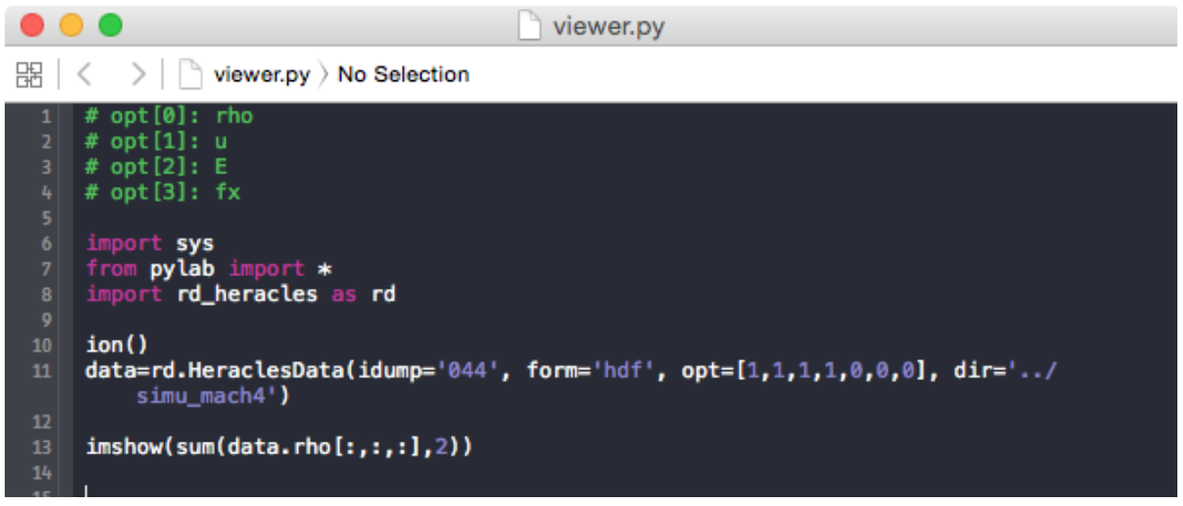
python code showing parameter at [0,0,0]

rho : density
u : velocity(x,y,z)
E : energy
fx : ionization

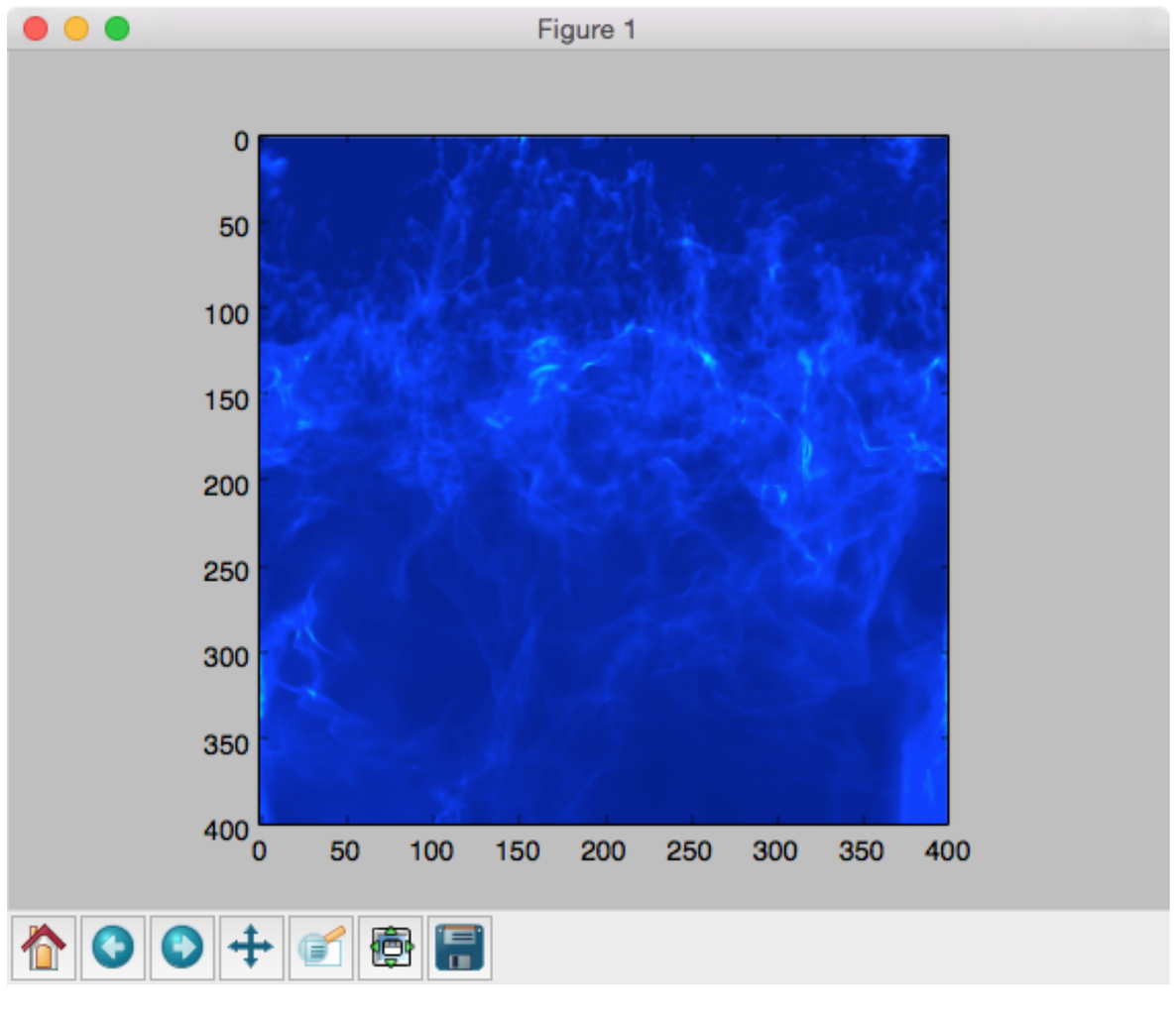


result

3. visualization for scientific usage



python code for visualization



2D visualization of density

4. split/ceonvert to binary file

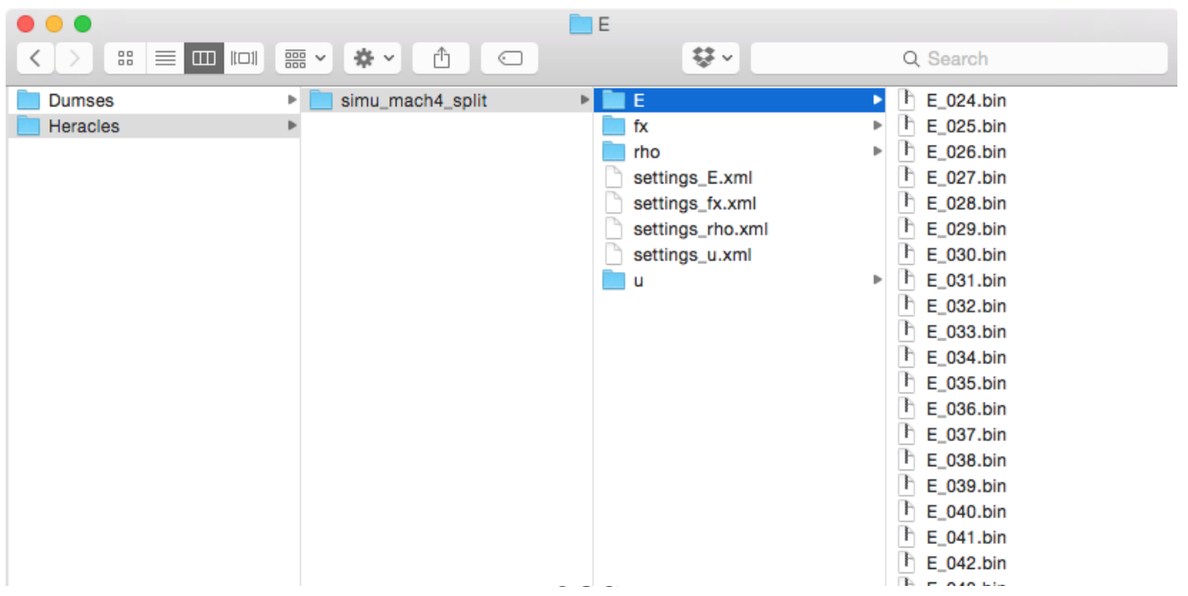
splite parameter and convert hdf5 to binary for C++ app.

spliter_mp.py

spliter_mp.py 〉 No Selection

```python
import sys
from pylab import *
import rd_heracles as rd
from multiprocessing import Pool

def split_param( idump ) :
    dirName = '0' + str(idump)
    print( "start : " + dirName )
    data=rd.HeraclesData(idump=dirName, form='hdf', opt=[1,1,1,1,0,0,0], dir='../
        simu_mach4')

    file = open( '../split/rho/rho_' + dirName + '.bin', 'w')
    file.write(data.rho)
    file.close()

    file = open( '../split/u/u_' + dirName + '.bin', 'w')
    file.write(data.u)
    file.close()

    file = open( '../split/E/E_' + dirName + '.bin', 'w')
    file.write(data.E)
    file.close()

    file = open( '../split/fx/fx_' + dirName + '.bin', 'w')
    file.write(data.fx)
    file.close()
    print( "finish : " + dirName )
    print("\n\n")

if __name__ == "__main__":

    range_st = int(sys.argv[1])
    range_end = int(sys.argv[2])
    rng = range( range_st, range_end )

    processor = 4
    p = Pool( processor )
    p.map( split_param, rng)
    print( "\n\nFINISH ALL :" + str(range_st) + "-" + str(range_end) + "\n")

```

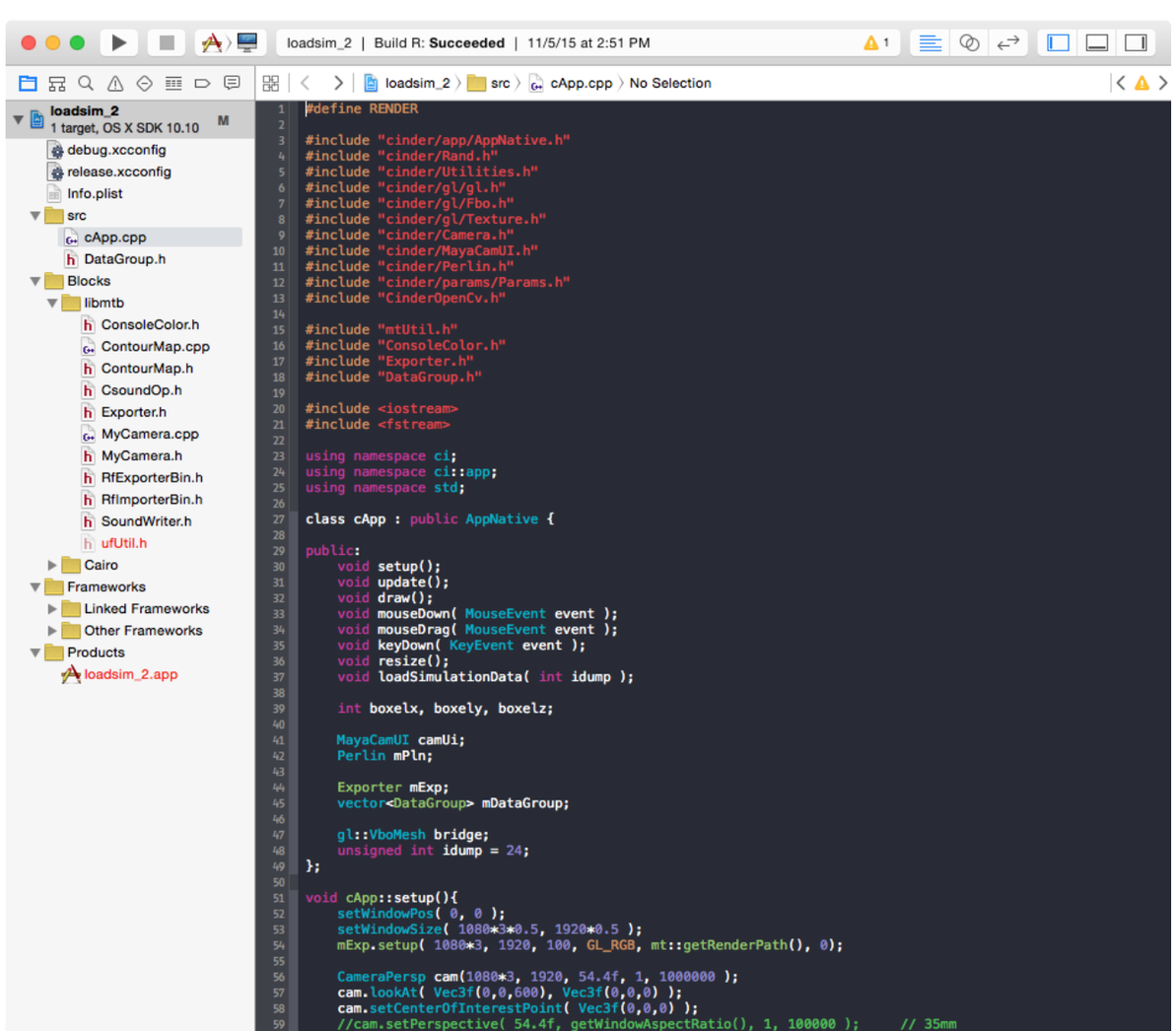
python code for split and convert



5. load to C++ framework

Example visualization process

1. Load density data
2. apply log calulation
3. separate to 7 groups based on parameter range
4. add pseudo color for each groups
5. add line between groups




```cpp
#define RENDER

#include "cinder/app/AppNative.h"
#include "cinder/Rand.h"
#include "cinder/Utilities.h"
#include "cinder/gl/gl.h"
#include "cinder/gl/Fbo.h"
#include "cinder/gl/Texture.h"
#include "cinder/Camera.h"
#include "cinder/MayaCamUI.h"
#include "cinder/Perlin.h"
#include "cinder/params/Params.h"
#include "CinderOpenCv.h"

#include "mtUtil.h"
#include "ConsoleColor.h"
#include "Exporter.h"
#include "DataGroup.h"

#include <iostream>
#include <fstream>

using namespace ci;
using namespace ci::app;
using namespace std;

class cApp : public AppNative {

public:
    void setup();
    void update();
    void draw();
    void mouseDown( MouseEvent event );
    void mouseDrag( MouseEvent event );
    void keyDown( KeyEvent event );
    void resize();
    void loadSimulationData( int idump );

    int boxelx, boxely, boxelz;

    MayaCamUI camUi;
    Perlin mPln;

    Exporter mExp;
    vector<DataGroup> mDataGroup;

    gl::VboMesh bridge;
    unsigned int idump = 24;
};

void cApp::setup(){
    setWindowPos( 0, 0 );
    setWindowSize( 1080*3*0.5, 1920*0.5 );
    mExp.setup( 1080*3, 1920, 100, GL_RGB, mt::getRenderPath(), 0);

    CameraPersp cam(1080*3, 1920, 54.4f, 1, 1000000 );
    cam.lookAt( Vec3f(0,0,600), Vec3f(0,0,0) );
    cam.setCenterOfInterestPoint( Vec3f(0,0,0) );
    //cam.setPerspective( 54.4f, getWindowAspectRatio(), 1, 100000 );     // 35mm
    camUi.setCurrentCam( cam );

    mPln.setSeed(123);
    mPln.setOctaves(4);

    boxelx = boxely = boxelz = 400;

#ifdef RENDER
    mExp.startRender();
#endif
}

void cApp::update(){

    for( auto dg : mDataGroup )
    dg.clear();
    mDataGroup.clear();

    idump++;
    loadSimulationData( idump );

}


void cApp::loadSimulationData( int idump){

    string fileName = "sim/Heracles/simu_mach4_split/rho/rho_0" + to_string(idump) + ".bin";

    fs::path assetPath = mt::getAssetPath();
    string path = ( assetPath/fileName ).string();
    cout << "loading binary file : " << path << endl;
    std::ifstream is( path, std::ios::binary );
    if(is){
        ccout::b("load OK binary file");
    }else{
        ccout::r("load ERROR bin file");
        quit();
    }

    // get length of file:
    is.seekg (0, is.end);
    int fileSize = is.tellg();
    cout << "length : " << fileSize << " byte" << endl;
    is.seekg (0, is.beg);
```

6. rendering as sequence of PNG file