Outils pour le C++:

Cmake et Gnuplot

Sommaire

- CMake
- Gnuplot et gnuplot-iostream
- Exemple : roblib.cpp

CMake : présentation

- Famille d'outil cross plateforme pour compilé, testé des logiciels.
- Plus haut niveau que la commande make
- Fonctionne avec des Cmakelists
- Produit un Makefile

Utilisation:

```
mkdir build
cd build
cmake ..
make
```

CMake : exemple

Nom du projet ->
Version de Cmake ->

```
project(Gnuplot_cours)
cmake_minimum_required(VERSION 3.0)
```

Pour gdb ->
Standard c++ ->

```
set(CMAKE_BUILD_TYPE Debug)
set (CMAKE_CXX_STANDARD 11)
```

Package supplémentaire

```
find_package(Boost 1.58.0 COMPONENTS filesystem system iostreams REQUIRED)
include_directories(${Boost_INCLUDE_DIR})
link_directories(${Boost_LIBRARY_DIRS})
```

Pour ajouter des .h et les .cpp liés

```
file(GLOB SOURCES
    src/*)

file(GLOB HEADERS
    include/*)
```

Pour ajouter des exécutables

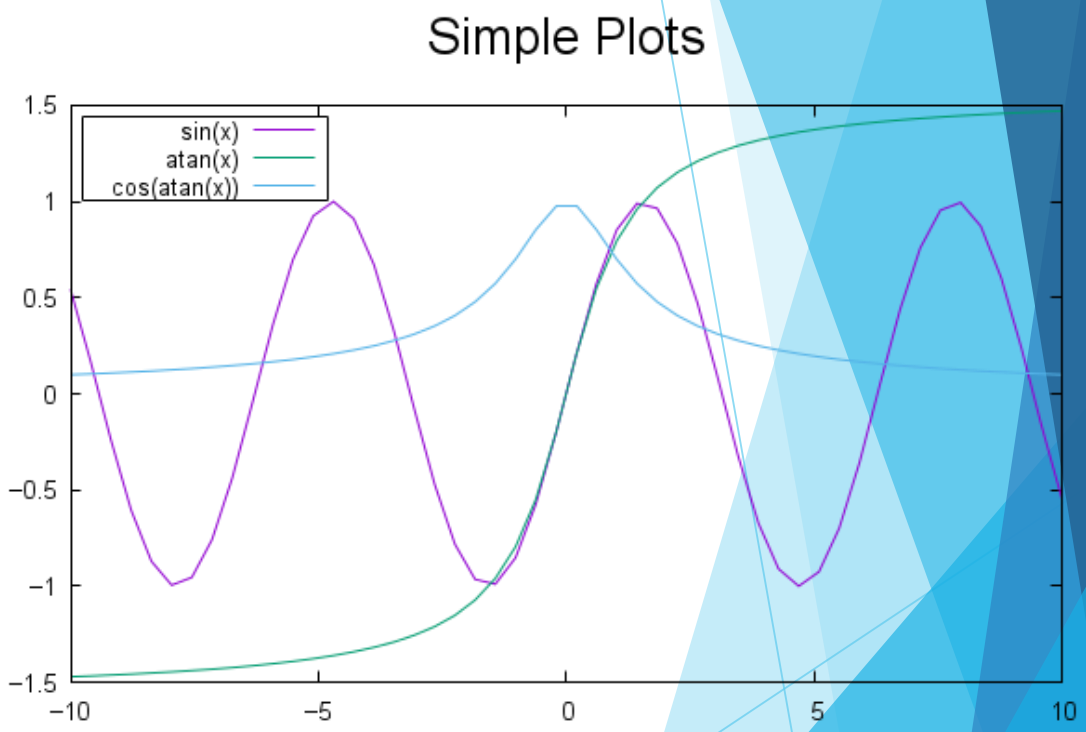
```
add_executable(exemple exemple.cpp)

target_link_libraries(exemple ${SOURCES} ${HEADERS} ${Boost_LIBRARIES})
```

Gnuplot

- Affichage en 2D/3D pour Linux/Windows/OSX/…
- Ligne de commande

```
# set terminal pngcairo transparent enhanced font "arial,10" fontscale 1.0 size 600, 400
# set output 'simple.1.png'
set key fixed left top vertical Right noreverse enhanced autotitle box lt black linewidth 1.000 dashtype solid
set style increment default
set samples 50, 50 set title "Simple Plots"
set title font ",20" norotate
set xrange [ * : * ] noreverse writeback
set x2range [ * : * ] noreverse writeback
set yrange [ * : * ] noreverse writeback
set y2range [ * : * ] noreverse writeback
 set zrange [ * : * ] noreverse writeback
set cbrange [ * : * ] noreverse writeback
 set rrange [ * : * ] noreverse writeback
plot [-10:10] sin(x),atan(x),cos(atan(x))
```



Plus d’info : http://www.gnuplot.info

gnuplot-iostream.h

- Permet à gnuplot d'être utiliser en c++

```
#include <vector>
#include <utility>
#include <gnuplot-iostream/gnuplot-iostream.h>

typedef std::pair<double, double> Point;

int main() {
  std::vector<Point> data;

  double x = 0.0;
  double y = 0.0;
  double c = 0.0;

  Gnuplot gp;
  gp << "set terminal wxt size 800, 400\n";

  while (x < 10000) {
    x += 0.01;
    y = sin(x);
    c += 0.01;
    data.push_back(Point(x,y));
    //std::cout <<  x << std::endl;
    if (c > 0.1) {
      gp << "plot '-' with lines title 'sin(x)'\n";
      gp.send1d(data);
      c = 0.0;
    }
  }

  return 0;
}
```

http://stahlke.org/dan/gnuplot-iostream/

Exemple : roblib.cpp

Objectif : refaire roblib.py en c++ et si possible facile d'utilisation

- numpy -> Eigen
- matplotlib -> gnuplot-iostream.h
- Création d'une class Plot

Code source :
https://github.com/phoenix1249/Gnuplot_Cmake

Exemple : Plot.h, roblib.h

```
class Plot
{
private:
    Gnuplot gp;
    std::vector<std::vector<point>> points;
    std::vector<std::string> colors;
    double x_min;
    double x_max;
    double y_min;
    double y_max;



public:
    Plot();
    ~Plot();
    void set_x_limit(double xmin, double xmax);
    void set_y_limit(double ymin, double ymax);
    void clear();
    void draw_circle(point Center,double radius,std::string color);
    void draw_circle(double x,double y,double radius,std::string color);
    void draw_ellipse(point Center,double a, double b,std::string color);
    void draw_ellipse(double x,double y,double a, double b,std::string color);
    void plot(std::vector<point> & p,std::string color);
    void plot(Eigen::MatrixXd & M,std::string color);
    void draw_tank(double x,double y,double theta,std::string color,double r);
    void draw_tank(double x,double y,double theta,std::string color);
    void draw_disk(double x, double y,double radius,std::string color);//TODO
    void draw_sailboat(double x,double y,double theta,double deltas,double deltar,double psi,double awind);
    void draw_car(double x,double y,double theta,std::string color);//TODO
    void show();
};
```

```
Eigen::MatrixXd eulermat(double phi,double theta,double psi);
Eigen::MatrixXd eulerderivate(double phi,double theta,double psi);
double angle(double x,double y);
double angle(point p);
Eigen::MatrixXd adjoint(Eigen::VectorXd w);
Eigen::MatrixXd move_motif(Eigen::MatrixXd M,double x,double y,double theta);
Eigen::MatrixXd translate_motif(Eigen::MatrixXd M,double x);
Eigen::MatrixXd ones(int rows,int cols);
void kalman_predict(Eigen::MatrixXd & xup,Eigen::MatrixXd & Gup,Eigen::MatrixXd &u,Eigen::MatrixXd & Ga,Eigen::MatrixXd & A,Eigen::MatrixXd & x_out,Eigen::
void kalman_correct(Eigen::MatrixXd & x0,Eigen::MatrixXd & G0,Eigen::MatrixXd & y,Eigen::MatrixXd & Gb,Eigen::MatrixXd & C,Eigen::MatrixXd & xup,Eigen::Mat
std::vector<Eigen::MatrixXd> kalman(Eigen::MatrixXd & x0,Eigen::MatrixXd & G0,Eigen::MatrixXd & u,Eigen::MatrixXd & y,Eigen::MatrixXd & Ga,Eigen::MatrixXd
double sawtooth(double x);
```

Exemple

```cpp
#include "include/roblib.h"
#include <string>
#include <iostream>


int main(int argc, char **argv){
    Plot plt;
    //plt.draw_ellipse(point(0,0),2,1,"'red'");
    //plt.draw_circle(1,0.5,1,"'blue'");
    plt.set_x_limit(-10,10);
    plt.set_y_limit(-10,10);
    for (int i = 0;i<360;i++){
        plt.clear();
        plt.draw_tank(1,0.5,i,"'blue'");
        plt.draw_circle(-2,1,5,"'red'");
        plt.draw_sailboat(0,0,i,0.1,0.2,40,1);
        usleep(100000);
        plt.show();
    }
    return 0;
}
```