Libreria grafica libplot

Uso

- Serve a visualizzare interattivamente i risultati dei programmi;
- per compilare usare
  g++ -o programma programma.c -lplotter;
- includere sempre ⟨*plotter.h*⟩;

Struttura del programma

- definire almeno un'istanza della classe Plotter e una della classe PlotterParams;
- chiamare nel giusto ordine le funzioni
  per inizializzare ;
- specificare sempre l'istanza della classe nelle chiamate alle funzioni grafiche ;
- chiamare le funzioni per terminare;

Classi

Due classi vengono definite dalle plotutils per fare le animazioni: la classe Plotter che indica il disegno in sé, e la classe PlotterParams che permette di modificare i parametri e le caratteristiche del plot.

Funzioni per inizializzare e terminare

Se grafico è un'istanza della classe Xplotter e parametri è un'istanza della classe Plotterparams

- *PlotterParams parametri()*; definisce e inizializza i parametri.

- *XPlotter = grafico(cin, cout, cerr,parametri)*
  "X" può essere sostituito da "PS";

- *grafico.openpl()*
  per aprire il plot;

- *grafico.closepl()*
  per chiudere il plot;

Funzioni per disegnare

- *grafico.erase()*
  per cancellare il contenuto del plot;

- *grafico.fspace(xini, yini, xfin, yfin)*
  per definire le coordinate utente;

- *grafico.fmove(x, y)*
  per spostare il cursore grafico in x,y;

- *grafico.pencolorname("red")*
  per stabilire con che colore si disegna;

- *grafico.flinewidth(0.25)*
  stabilisce la largezza della linea;

- *grafico.fcircle(x, y, raggio)*
  disegna una circonferenza;

- *grafico.bgcolorname("blue")*
  stabilisce il colore dello sfondo;

- *grafico.fline(x1,y1,x2,y2)*
  disegna una linea;

Copia e incolla: un metodo semplice per fare cose complicate.

Per usare un tipo nuovo di programma non è di solito necessario capirlo profondamente: si può cominciare facendo taglia e incolla sui programmi scritti da altri, come questo preso dalle pagine info, e cercare di utilizzarlo con piccole modifiche.

```
#include <plotter.h>
const int maxorder = 12;

void draw_c_curve (Plotter& plotter, double dx, double dy, int order)
{
  if (order >= maxorder)
    plotter.fcontrel (dx, dy); // continue path along (dx, dy)
  else
    {
      draw_c_curve (plotter, 0.5 * (dx - dy), 0.5 * (dx + dy), order + 1);
      draw_c_curve (plotter, 0.5 * (dx + dy), 0.5 * (dy - dx), order + 1);
    }
}
int main ()
{
  // set a Plotter parameter
  PlotterParams params;
  params.setplparam ("PAGESIZE", (char *)"letter");

  PSPlotter plotter(cin, cout, cerr, params); // declare Plotter
  if (plotter.openpl () < 0)                  // open Plotter
    {
      cerr << "Couldn't open Plotter\n";
      return 1;
    }

  plotter.fspace (0.0, 0.0, 1000.0, 1000.0); // specify user coor system
  plotter.flinewidth (0.25);       // line thickness in user coordinates
  plotter.pencolorname ("red");    // path will be drawn in red
  plotter.erase ();                // erase Plotter's graphics display
  plotter.fmove (600.0, 300.0);    // position the graphics cursor
  draw_c_curve (plotter, 0.0, 400.0, 0);
  if (plotter.closepl () < 0)      // close Plotter
    {
      cerr << "Couldn't close Plotter\n";
      return 1;
    }
  return 0;
```

Analisi del programma

A una prima occhiata si vede subito che le istruzioni si possono raggruppare in tre tipi: all'inizio (e alla fine) quelle per inizializzare le capacità grafiche del sistema, subito dopo quelle che definiscono lo spazio in cui si lavora (dimensione della finestra, sistema di coordinate) e infine vengono le funzioni che disegnano quello che ci interessa. Possiamo quindi mantenere quasi inalterate quelle appartenenti ai primi due tipi, cambiando a nostro vantaggio quellle del terzo tipo Prima di far questo si compila ed esegue il programma, e questo ci chiarirà le idee su come procedere per il caso che ci interessa.

Problema del biliardo con due sfere

Nel biliardo mostrato nell'esercizio, due sfere rimbalzano sulle pareti. Come si fa a tener conto dei loro urti? Prima di tutto consideriamo che le sfere si urtano quando la loro distanza è $< 2R$. La componente della velocità tangente alla congiungente $\vec{\Delta} = \vec{x_2} - \vec{x_1}$ i due centri resta inalterata, mentre le altre due componenti si scambiano.

$$\vec{v}_1' = \vec{v}_1 + \vec{v}_{2\parallel} - \vec{v}_{1\parallel}$$

$$\vec{v}_2' = \vec{v}_2 + \vec{v}_{1\parallel} - \vec{v}_{2\parallel}$$

Se

$$\vec{u} = \frac{(\vec{v_2} - \vec{v_1}) \cdot \vec{\Delta}}{\Delta^2} \vec{\Delta}$$

abbiamo

$$\vec{v}_1' = \vec{v}_1 + \vec{u} \qquad \vec{v}_2' = \vec{v}_2 - \vec{u}$$

Attenzione alle condizioni iniziali!