19CSE433 Computer Graphics and Visualization Lab1

S.PADMAVATHI

CSE, AMRITA SCHOOL OF ENGINEERING, COIMBATORE

Graphics in TC

graphics.h header file is must in all the programs, if you are using graphics functions.

initgraph is used to initialize the system graphics by loading a graphics driver from disk and thereby putting the system into graphics mode. initgraph takes 3 parameters: graphic detect, graphic mode, and bgi file path.

bgi stands for **Borland Graphics Interface**, it is a graphic library. This library loads graphic drivers and vector fonts (*.CHR).

closegraph deallocates the memory allocated by the system graphics and then restores the screen to the mode it was before calling initgraph.

```
#include<graphics.h>
#include<stdio.h>
#include<conio.h>
void main()
{   int gd=DETECT, gm;
   int x1=100, y1=100, x2=200, y2=200;
   initgraph(&gd, &gm, "c:\\turboc3\\bgi");
   cleardevice();
   line(x1,y1,x2,y2);
   getch();
   closegraph(); }
```

Ubuntu GCC graphics

If you want to use graphics.h on Ubuntu platform you need to compile and install libgraph. It is the implementation of turbo c graphics API on Linux using SDL. You can download it

Step by Step Instructions:

STEP 1: First install build-essential by typing

sudo apt-get install build-essential

STEP 2: Install some additional packages by typing

sudo apt-get install libsdl-image1.2 libsdl-image1.2-dev guile-2.0 \

guile-2.0-dev libsdl1.2debian libart-2.0-dev libaudiofile-dev \

libesd0-dev libdirectfb-dev libdirectfb-extra libfreetype6-dev \

libxext-dev x11proto-xext-dev libfreetype6 libaa1 libaa1-dev \

libslang2-dev libasound2 libasound2-dev

STEP 3: Now extract the downloaded libgraph-1.0.2.tar.gz file.

Ubuntu graphics configuration

STEP 4: Goto extracted folder and run following command

```
./configure
make
sudo make install
sudo cp /usr/local/lib/libgraph.* /usr/lib
```

Now you can use graphics.h lib using following lines:

```
int gd = DETECT,gm;
initgraph (& gd,& gm,NULL);
```

Graphics example in Linux

```
// C code to illustrate using
// graphics in linux environment
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
```

```
int main()
{    int gd = DETECT, gm;
    initgraph(&gd, &gm, NULL);
      circle(50, 50, 30);
     delay(500000);
    closegraph();
    return 0;
}
```

GCC compiler

For terminal you need to add the graphics.h libraray to you GCC compiler. For this you will have type in the following commands.

```
>sudo apt-get install build-essential

>sudo apt-get install libsdl-image1.2 libsdl-image1.2-dev guile-2.0 \

 guile-2.0-dev libsdl1.2debian libart-2.0-dev libaudiofile-dev \

 libesd0-dev libdirectfb-dev libdirectfb-extra libfreetype6-dev \

 libxext-dev x11proto-xext-dev libfreetype6 libaa1 libaa1-dev \

 libslang2-dev libasound2 libasound2-dev \

>sudo make install

sudo cp /usr/local/lib/libgraph.* /usr/lib
```

On sequentially typing all the above commands you can successfully install the graphics.h library in your GCC compiler of terminal.

Configuring DevC++

Go to the location where DevC++ is installed. For me its D drive. Go inside the MinGW64 folder. Copy the graphics.h and winbgim.h in the include folder and D:\Dev-Cpp\MinGW64\x86_64-w64-mingw32\include folder.

Copy the libbgi.a file into lib folder and in D:\Dev-Cpp\MinGW64\x86_64-w64-mingw32\lib folder.

Copy the ConsoleAppGraphics.template, ConsoleApp_cpp_graph.txt files and paste them inside the template folder of the devc++ installer location.

Graphics project options

Go to “Project” menu and choose “Project Options”.

Go to the “Parameters” tab.

In the “Linker” field, enter the following text:

```
-lbgi
-lgdi32
-lcomdlg32
-luuid
-loleaut32
-lole32
```

Click “Ok” to save settings.

Test your first Graphics program

```
#include <graphics.h>
#include <iostream>
using namespace std;
int main()
{
initwindow(800,600);
circle(200,300,100);
while(!kbhit());
closegraph();
return 0;
}
```

arc() – creates arc of a given angle and given radius.

bar() – creates a bar with given coordinates.

circle() – creates a circle of given radius.

ellipse() – creates an ellipse with given major and minor axis.

floodfill() – flood fill is used to fill a specific color to a specific point whose coordinates are given.

line() – creates a line of given starting and ending points.

rectangle() – creates a rectangle with given coordinates.

Graphics shape

Circle function is used to draw a circle which required 3 parameters i.e x, y and radius.

Rectangle function is used to draw a rectangle which required 4 parameters i.e x1(left), y1(top), x2(right) and y2(bottom)

Arc function is used to draw a arc. Syntax: arc(int x, int y, int stangle, int endangle, int radius);

Bar function draws a rectangle and fill it with current fill pattern and color.
Syntax: bar(int left, int top, int right, int bottom);

outtextxy(100, 200, "My first C graphics program");

drawpoly(int n_points, int* points)

sector(int x, int y, int stangle, int endangle, int xradius, int yradius)

fillellipse(int x, int y, int xradius, int yradius);

fillpoly(int n_points, int* points);

Color functions in C Graphics

In C Graphics, there are 16 colors declared. We use colors to set background color, change color of shapes, fill color of shapes, change text color.

textcolor is use to set the text color.

textcolor(GREEN);

cprintf("Text in Green");

setbkcolor is use to set the background color by specifying the color name or the number

setbkcolor(GREEN);

rectangle(100,100,200,200);

setcolor is use to set the text color or set the outline color of the various shapes.

setcolor(GREEN); or setcolor(3);

rectangle(100,100,200,200);

outtextxy(110,110,"Text in Green")

Line and Fill styles

```
setfillstyle(SOLID_FILL, BROWN);
   setfillstyle(SLASH_FILL, BLUE);
   setfillstyle(HATCH_FILL, GREEN);
   floodfill(200, 105, WHITE);
   floodfill(210, 105, WHITE);
   setlinestyle(SOLID_LINE,0,2);
```

setlinestyle (style, pattern, thickness);

It has three parameters all of int types. These may be int type values or variables.

Where

style:

specifies the line style. Its value may be from 0 to 4.

pattern:

specifies the line pattern. Its value may be from 0 to 12.

thickness:

specifies the thickness of line. Its value may be from 0 to 3.

Text style

```
settextstyle(BOLD_FONT,HORIZ_DIR,2);
    outtextxy(275,0,"3D BAR GRAPH");
```

settextstyle (style, dir, size);

All the three parameters are of int type. These may be int type values or variables.

Where:

Style:

specifies the font style. Its value range is from 0 to 10.

Dir:

specifies the direction of the text in which it is to be displayed. Its value is from 0 to 1. It may be a numerical constant identifier. It is HORIZ DIR (for horizontal direction) or VERT_DIR (for vertical direction).

Size:

specifies the font size of the text. Its value is from 0 to 72.

Image and pixel

Getimage

getimage (left, top, right, bottom, void far *bitmap);

Putimage

putimage(left, top, void far *bitmap);

Getpixel

putpixel

Exercise

```
for(i=0;i<100;i++)
   {
      setcolor(i);
      //coordinates of center from x axis, y axis, radius
      circle(200,200,20+i*2);
      delay(30);
      //cleardevice(); //try with and
without cleardevice();
   }
```

```
while(!kbhit())
   {
      for(i=0; i<=500; i++)
      {
        x=rand()%getmaxx();
        y=rand()%getmaxy();
        putpixel(x,y,15); /* putpixel(x,y,i); */
      }
      delay(500);
      cleardevice();
   }
```