Karl-Friedrich Penning

# Einfache Schwarzweiß-Grafik unter C

Die Programmiersprache C eignet sich sehr gut für Berechnungen im technisch-wissenschaftlichen Bereich. Leider fehlt aber bei den meisten Compilern die Möglichkeit, mathematische Ergebnisse grafisch darzustellen, was in der Technik unerläßlich ist. Auf dem Markt werden zwar „Graphic Toolboxes" angeboten, sie sind jedoch teuer und bieten meist mehr, als man in der Regel zur Darstellung einfacher Kurvenzüge benötigt.

```
/*            Einfache s/w-Grafik unter 'C'              */
/*********************************************************/
/*
**    Vers. 2.1       11.2.87
**    Autor: Karl-Fr. Penning
*/

#include <dos.h>
#include <stdlib.h>

#define SCHIRM  16          /*  Bildschirminterrupt       */
#define HRG      6          /*  High Resolution Grafics   */
#define TEXT     2          /*  Text-Bildschirm           */


static int xalt,yalt;       /*  Merker fuer letzten Punkt */


void grafset()              /*  Bildschirm auf hochauflösende
                            Grafik 640 * 200 umstellen    */
{
    union REGS cpureg;

    cpureg.h.ah  = 0;       /*  Video-Modus               */
    cpureg.h.al  = HRG;     /*  Grafik                    */
    int86( SCHIRM , &cpureg , &cpureg );
}


void textset()              /*  Bidschirm auf Textdarstellung
                            umschalten                    */
{
    union REGS cpureg;

    cpureg.h.ah  = 0;       /*  Video-Modus               */
    cpureg.h.al  = TEXT;    /*  Textdarstellung           */
    int86( SCHIRM , &cpureg , &cpureg );
}


void punkt(x,y)             /*  Punkt in Position x,y setzen */
    unsigned x,y;           /*  0,0 ist links oben           */
{
    union REGS cpureg;

    cpureg.h.ah  = 12;      /*  Punkt schreiben           */
    cpureg.h.al  = 1;       /*  Farbcode                  */
    cpureg.x.dx  = (yalt = y);
    cpureg.x.cx  = (xalt = x);
    if(x<0 || y<0 || x>639 || y>199)
        return;
    int86( SCHIRM , &cpureg , &cpureg );
}


void line(x1,y1,x2,y2)      /*  Linie zeichnen von x1,y1 */
    int x1,y1,x2,y2;        /*  nach x2,y2               */
{
    long dx,dy;
    int i;

    dx = x2 - x1;           /* laengere Achse ermitteln  */
    dy = y2 - y1;
    if(abs(dx) > abs(dy))   /* und relativ dazu zeichnen */
        if(dx>0) for( i=0 ; i<=dx ; i++ )
            punkt(x1+i,(int) (y1+((dy*i*2)/dx+1)/2));
            else for( i=0 ; i>=dx ; i-- )
                punkt(x1+i,(int) (y1+((dy*i*2)/dx+1)/2));
        else
            if(dy>0) for( i=0 ; i<=dy ; i++ )
                punkt( (int) (x1+((dx*i*2)/dy+1)/2),y1+i);
            else for( i=0 ; i>=dy ; i-- )
                punkt( (int) (x1+((dx*i*2)/dy+1)/2),y1+i);
    }


    void lineto(x,y)            /* Linie vom letzten         */
        unsigned x,y;           /* gezeichneten Punkt nach   */
    {                           /* x,y zeichnen              */
        line(xalt,yalt,x,y);
    }
```

**Bild 1. Quelltext der Grafik-Prozeduren**

```
/*            Einfache s/w-Grafik unter 'C'              */
/*********************************************************/
/*
**    Vers. 2.1       11.2.87
**    Autor: Karl-Fr. Penning
**
**       INCLUDE - File
**  Deklaration der Grafik-Routinen aus GRAFIK.LIB
*/

void grafset(void);         /*  Bildschirm auf hochauflösende
                            Grafik 640 * 200 umstellen    */

void textset(void);         /*  Bidschirm auf Textdarstellung
                            umschalten                    */

void punkt(int,int);        /*  Punkt in Position x,y setzen */
                            /*  0,0 ist links oben           */

void line(int,int,int,int); /*  Linie zeichnen von x1,y1  */
                            /*  nach x2,y2                */

void lineto(int,int);       /*  Linie vom letzten         */
                            /*  gezeichneten Punkt nach   */
                            /*  x,y zeichnen              */

/*    E n d e    */
```

**Bild 2. Mit dieser Include-Datei kann man die Prozeduren deklarieren**

Die hier vorgestellte kleine Grafik-Bibliothek wurde erstellt, um z. B. Meßwertkurven und Diagramme in hochauflösender Grafik auf IBM-kompatiblen PCs darstellen zu können. Das wird möglich durch fünf Funktionen:

### grafset()

Diese Funktion schaltet auf Grafikdarstellung um und löscht den Bildschirm.

```
/*
**          G R A F T S T
**
**  Vers.: 1.0      11.2.87
**  Autor: Karl-Fr. Penning
**
**  Test für die Grafik-Bibliothek GRAFIK.LIB
*/

#include <stdio.h>
#include <conio.h>
#include <grafik.h>
#include <math.h>

#define PI2  6.2831852      /*  2 Pi  */

/* Vorwärtsdeklaration */
float funktion(int);

main()
{
        int i,x,y;
        float wert;

        printf("\n\n\
                        G R A F T E S T\n\n\
Dieses Grafik-Testprogramm zeichnet zunächst einen Oszilloskop-ähnlichen\n\
Rahmen und trägt eine Sinuskurve ein.\n\n\
    - Taste drücken - ");
        getch();

        grafset();              /* grafische Darstellung einschalten */

        /*  Oszilloskop-ähnliche Darstellung zeichnen  */

        for( y=0 ; y<181 ; y+=18 )        /* horizontale punktierte */
            for( x=135 ; x<636 ; x+=3 )   /* Linien zeichnen        */
                punkt(x,y);

        for( x=135 ; x<636 ; x+=50 )      /* vertikale punktierte   */
            for( y=0 ; y<181 ; y+=3 )     /* Linien zeichnen        */
                punkt(x,y);

        line(135,0,135,180);              /* durchgezogene Umrandung */
        lineto(635,180);                  /* zeichnen                */
        lineto(635,0);
        lineto(135,0);

        line(125,90,639,90);              /* Null-Linie              */

        /*  Funktionswerte eintragen  */

        wert = funktion(0);
        punkt( 135 , 90 - 80.0 * wert );
        for( i=1 ; i<500 ; i++ )
        {
            wert = funktion(i);
            lineto( 135+i , 90 - 80.0 * wert );
        }

        /* auf Tastendruck warten  */
        printf("\n\n\n\n\n TASTE\ndruecken\n");
        getch();
        textset();
}

/*********************************************************/

float funktion(phi)             /* Testfunktion */
        int phi;
{
        return( sin( PI2 * phi / 500.0 ) );
}

/*************** E N D E  ******************************/
```

**Bild 3. Dieses Beispiel erzeugt die Grafik von Bild 4**

### textset()

Diese Funktion schaltet auf Textdarstellung (80 Zeichen) zurück und löscht den Bildschirm.

### punkt(x,y)

Über die beiden Integer-Variablen x und y wird dieser Funktion die Position eines zu setzenden Punktes mitgeteilt. Bei der Standard-IBM-Grafik werden x-Werte im Bereich von 0 bis 639 und y-Werte im Bereich von 0 bis 199 akzeptiert. x und y dürfen auch außerhalb dieses Bereiches liegen; ein Punkt wird dann jedoch nicht gezeichnet. Der Punkt mit den Koordinaten 0,0 ist oben links.

### line(x1,y1,x2,y2)

Diese Funktion zeichnet eine Gerade zwischen den Punkten x1,y1 und x2,y2. Die Punkte dürfen auch außerhalb des Darstellbereiches liegen. Es wird dann nur der im Darstellungsbereich liegende Geradenteil gezeichnet.

### lineto(x,y)

Von dem zuletzt angesprochenen Punkt, der auch außerhalb des Darstellbereiches liegen kann, wird eine Gerade zum Punkt mit den Koordinaten x,y gezeichnet.

Den Quelltext zeigt *Bild 1*. Will man die Funktionen anwenden, muß man sie zunächst deklarieren. Das kann über ein Include-File geschehen, wie in *Bild 2* gezeigt. *Bild 3* zeigt ein Anwendungsbeispiel und *Bild 4* das Resultat.

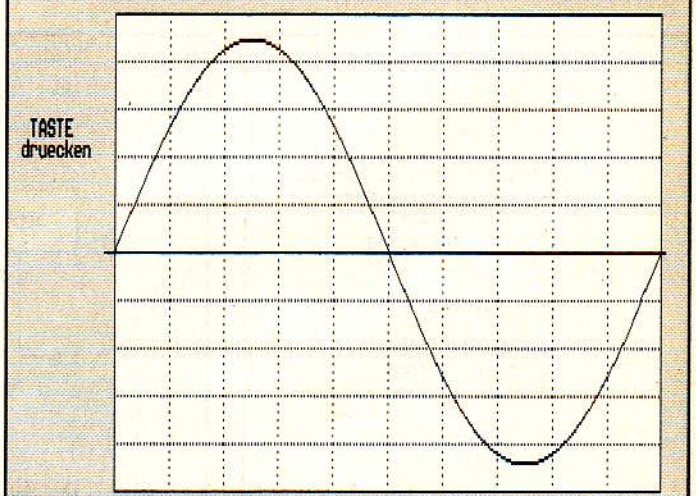


**Bild 4. Das Ergebnis des Beispielprogramms von Bild 3**