

Grafik: 2D-geometri. Något om dubbelbuffring. Java: 2D, awt & Swing.
Föreläsning 3

Innehåll

- 2D-transformationer
- Mer om 2D-grafik i Java
- Något om Javas AWT och Swing

◀ previous next ▶

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Grundläggande transformationer

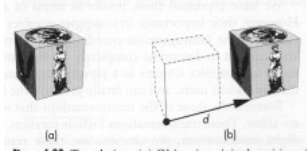
- Translation
 - $P' = P + d$
- Rotation
 - $P' = R * P$
- Skalning
 - $P' = S * P$

◀ previous next ▶ 2

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

...

- Translation
 - $P' = P + d$

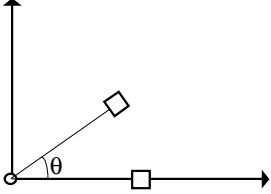


◀ previous next ▶ 3

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

...

- Rotation kring origo

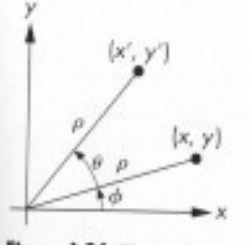


$P' = R \cdot P$

◀ previous next ▶ 4

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

...rotation på matrisform...

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$


Visas enklast med tex Eulers formel:

$$\rho e^{i\theta} = \rho(\cos \theta + i \sin \theta)$$

◀ previous next ▶ 5

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

...

$$x = \rho \cos \phi, x' = \rho \cos(\phi + \theta)$$

$$y = \rho \sin \phi, y' = \rho \sin(\phi + \theta)$$

$$\rho e^{i(\phi+\theta)} = \rho e^{i\phi} e^{i\theta}$$

$$\Leftrightarrow \rho(\cos(\phi + \theta) + i \sin(\phi + \theta)) =$$

$$\rho(\cos \phi \cdot \cos \theta - \sin \phi \cdot \sin \theta + i(\cos \phi \cdot \sin \theta + \sin \phi \cdot \cos \theta))$$

$$\Rightarrow$$

$$x' = x \cos \theta - y \sin \theta$$

$$y' = x \sin \theta + y \cos \theta$$

◀ previous next ▶ 6

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

...

- Skalning




Figure 4.27 Non-rigid-body transform


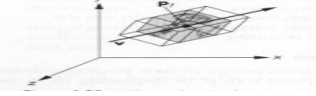


Figure 4.28 Uniform and nonuniform

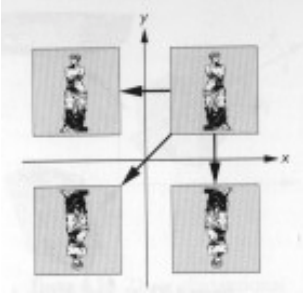


◀ previous next ▶ 7

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

...

- Reflektion



◀ previous next ▶ 8

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Transformationer i homogena koordinater

- För att behandla transformationer på ett uniformt sätt, dvs translation, skalning och rotation använder vi homogena koordinater
- En punkt representeras av en trippel

$$P = (x_h, y_h, h)$$

• brukar man välja $h = 1$ i sammanhang

- Alla transformationer kan representeras av matrismultiplikationer

◀ previous next ▶ 9

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Rotation i 2D med homogena koordinater

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

◀ previous next ▶ 10

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Translation i homogena koordinater

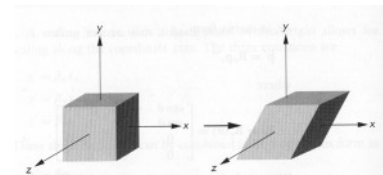
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha_x \\ 0 & 1 & \alpha_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

◀ previous next ▶ 11

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Fler grundläggande operationer

- Skevning



◀ previous next ▶ 12

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Sammanslagning av transformationer

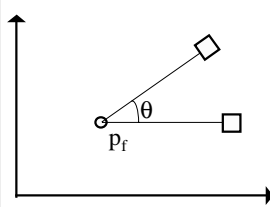
- Om homogena koordinater används kan alla transformationer utföras som matrismultiplikationer
- Detta gör att vi slå samman flera transformationer till en ny matris innan vi applicerar den på alla punkter i objektet (/objekten)
 - detta kan ge stora prestandavinstor speciellt om vi skall utföra samma transformationer på tusentals punkter

◀ previous next ▶

13

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Rotation kring fix punkt



Metod:

1. flytta rotationspunkten till origo
2. rotera
3. flytta tillbaks rotationspunkten

$$M = T(p_f)R(\theta)T(-p_f)$$


◀ previous next ▶

14

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Dubbelbuffring

- Problem
 - I många fall ger omritande av en applikations fönster ett flimrigt intryck
- Lösning
 - Använd dubbelbuffring
 - Dvs en teknik att innan man ritar ut på skärmen först rita i en buffert i minnet
 - när omritningen är klar presenteras hela bufferten (bitkartan) direkt på skärmen



◀ previous next ▶

15

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

- ...
- Användbara kompletterande tekniker
 - Bitkartsmanipulerande funktioner som RasterOp/BitBlt
 - Som ger oss möjlighet att kombinera olika bitkartor och samtidigt ange hur med masker och/eller logiska villkor som or, and, osv
- Användningsområden
 - Där vi inte vill presentera delresultat
 - Animering
 - Gummibandsfigurer
- Resultat
 - Vi får en icke flimrande uppdatering
 - I och för sig så går inte utritandet snabbare än vi ritar direkt på skärmen men upplevelsen blir ofta att den gör det

◀ previous next ▶

16

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Gummibandsfigurer

- Ett sätt att rita linjesegment och andra figurer på ett direktmanipulativt sätt som förändras i interaktion med användaren
- Metafor (dvs bildligt uttryckt, liknelse, efterapning av)
 - Gummiband
- Lösningar
 - Rita linje över bakgrund, återställ, rita ny linje
 - Någon xor-liknande teknik där gummiband och bakgrund "blandas"
 - Dubbelbuffring
 - spara "gammal" bakgrund
 - rita gummiband
 - återställ "gammal" bakgrund

◀ previous | next ▶ 17

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java 1.1: definiera utritning

- Använd "subklasstrategin"
- Skriv över metoden `paint(Graphics g)`
- Dvs skriv

```
public void paint(Graphics g) {
    // använd g för att rita
}
```

- metoden anropas automatiskt då fönstret behöver ritas om

◀ previous | next ▶ 18

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java 1.1: rita text på fönstret

- Använd metoden
 - `drawString(String str, int x, int y)` i klassen Graphics

```
public void paint(Graphics g) {
    // vi skriver ut fönstrets höjd
    System.out.println(getHeight());
    g.drawString("Här kommer en sträng halvvägs
    ner i y-led",
        20, //x-koordinat
        getHeight() / 2); //y-koordinat
}
```

◀ previous | next ▶ 19

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java

- I Java 1.1 är ritrutinerna ganska primitiva
 - Man kan endast
 - rita objekt med tjocklek 1 (ett)
 - rita enklare typer av primitiva objekt
 - hantera pixelkartor på ett begränsat sätt
 - Man kan också använda tex färg och sätta klipprektangel
- I Java2D har vi fler möjligheter
 - som tex
 - rita splines och andra kurvor
 - ange tjocklek på kanter
 - ange typ av linje (tex rita streckad linjer mm)
 - ange transformationer som skalning, translation och rotation
 - rita genomskinligt eller ange att antialiasing skall användas

◀ previous | next ▶ 20

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java 1.1: rita ickefyllda grafiska objekt

Några exempel

- linje
`gc.drawLine(int x1, int y1, int x2, int y2);`
- oval
`gc.drawOval(int x, int y, int width, int height);`
- rektangel
`gc.drawRect(int x, int y, int width, int height);`
- polygon
`gc.drawPolyline(int xPoints[], int yPoints[], int nPoints);`

alt
`gc.drawPolygon(Polygon p);`

◀ previous | next ▶ 21

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java 1.1: rita fyllda grafiska objekt

- oval
`gc.fillOval(int x, int y, int width, int height);`
- rektangel
`gc.fillRect(int x, int y, int width, int height);`
- polygon
`gc.fillPolygon(int[] xPoints, int[] yPoints, int nPoints)`

eller
`gc.fillPolygon(Polygon p)`

◀ previous | next ▶ 22

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java 1.1: rita med olika färg

- ändra färg
`gc.setColor(Color c)`
- rita med XOR-mod, dvs alternera mellan den grafiska kontextens färg och den givna
`gc.setXORMode(Color c1)`

◀ previous | next ▶ 23

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java 1.1: rita bilder

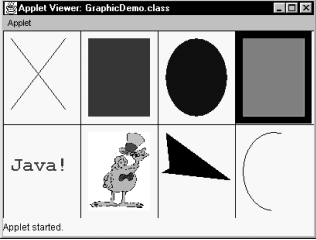
- Hämta bild från fil (i ett Frame-program)
`Image image = getToolkit().getImage(imageFile);`
- Rita bilden på grafiska kontexten (skalad att passa in i given rektangel)
`gr.drawImage(image, posX, posY, width, height, bgcol, this);`

◀ previous | next ▶ 24

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Exempel

- Exemplet innehåller bland annat
 - Med metoder för att sätt klipprektangel, färg och rita olika typer av figurer
- En liten test av olika grafikrutiner



previous next 25

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

...kod...

```
import java.awt.*;

public class GraphicsDemoFrame extends Frame
{
    static int offsetX = 10, offsetY = 10;
    String imageFile = "c:/katalog/frog.gif";
    Image image = null;

    public GraphicsDemoFrame(String s){
        super(s);
    }
}
```

previous next 26

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```
...
private void drawFunc1(Graphics gr, int x, int y,
    int w, int h)
{
    // Rita ett X
    gr.drawLine(x+offsetX,y+offsetY,w-20,h-20);
    gr.drawLine(x+offsetX,h-20,w-20,y+offsetY);
}

private void drawFunc2(Graphics gr, int x, int y,
    int w, int h)
{
    // Rita en röd rektangel
    Color prevColor = gr.getColor();
    gr.setColor(new Color(255,0,0));
    gr.fillRect(x+offsetX,y+offsetY,w-20,h-20);
    gr.setColor(prevColor);
}

```

previous next 27

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```
...
private void drawFunc3(Graphics gr, int x, int y,
    int w, int h)
{
    // Rita en blå cirkel
    Color prevColor = gr.getColor();
    gr.setColor(new Color(0,0,255));
    gr.fillOval(x+offsetX,y+offsetY,w-20,h-20);
    gr.setColor(prevColor);
}

```

previous next 28

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```

...
private void drawFunc4(Graphics gr, int x, int y,
                      int w, int h)
{
    // Rita en svart rektangel med en grön, förhöjd,
    // rektangel ovanpå
    Color prevColor = gr.getColor();
    gr.setColor(new Color(0,0,0));
    gr.fillRect(x+1,y+1,w-2,h-2);
    gr.setColor(prevColor);
    prevColor = gr.getColor();
    gr.setColor(new Color(0,255,0));
    gr.fill3DRect(x+offsetX,y+offsetY,w-20,h-
                20,false);
    gr.setColor(prevColor);
}

```

◀ previous next ▶ 29

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```

...
private void drawFunc5(Graphics gr, int x, int y,
                      int w, int h)
{
    // Skriv "Java" i röd fet Courier font
    Color prevColor = gr.getColor();
    gr.setColor(new Color(255,0,0)); //alt.
    Color.red
    Font f = new Font("Courier",Font.BOLD,25);
    gr.setFont(f);
    gr.drawString("Java!",x+offsetX,y+h/2);
    gr.setColor(prevColor);
}

```

◀ previous next ▶ 30

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```

...
private void drawFunc6(Graphics gr, int x, int y,
                      int w, int h)
{
    // Rita en bild (grodan)
    Color bgcol = new Color(127,127,127);
    gr.drawImage(image,x+offsetX,y+offsetY,w-20,h-
                20,bgcol,this);
}

```

◀ previous next ▶ 31

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```

...
private void drawFunc7(Graphics gr, int x, int y,
                      int w, int h)
{
    // Rita en polygon (femhörning)
    Polygon fillPol = new Polygon();
    fillPol.addPoint(x+offsetX,y+offsetY);
    fillPol.addPoint(x+15,y+56);
    fillPol.addPoint(x+5,y+62);
    fillPol.addPoint(x+95,y+72);
    gr.fillPolygon(fillPol);
}

```

◀ previous next ▶ 32


```

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing
...
private void drawFunc8(Graphics gr, int x, int y,
                      int w, int h)
{
    // Rita en båge
    gr.drawArc(x+offsetX,y+offsetY,w-20
              h-20,75,190);
}

```

◀ previous | next ▶ 33

```

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing
...
public void paint(Graphics gr)
{Rectangle r = gr.getClipBounds();
  int nBoxWidth = r.width / 4;
  int nBoxHeight = r.height / 2;

  gr.clearRect(r.x,r.y,r.width,r.height);
  // Rama in rutor
  for (int nTemp1 = 0; nTemp1 < 2; nTemp1++)
    for (int nTemp2 = 0; nTemp2 < 4; nTemp2++)
      { int x = nBoxWidth * nTemp2;
        int y = nBoxHeight * nTemp1;
        gr.drawRect(x, y, x + nBoxWidth, y +
                    nBoxHeight);
      }
}

```

◀ previous | next ▶ 34

```

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing
...
// Anropa respektive ritfunktion
drawFunc1(gr,nBoxWidth*0,nBoxHeight*0,nBoxWidth,nBoxHeight);
drawFunc2(gr,nBoxWidth*1,nBoxHeight*0,nBoxWidth,nBoxHeight);
drawFunc3(gr,nBoxWidth*2,nBoxHeight*0,nBoxWidth,nBoxHeight);
drawFunc4(gr,nBoxWidth*3,nBoxHeight*0,nBoxWidth,nBoxHeight);
drawFunc5(gr,nBoxWidth*0,nBoxHeight*1,nBoxWidth,nBoxHeight);
drawFunc6(gr,nBoxWidth*1,nBoxHeight*1,nBoxWidth,nBoxHeight);
drawFunc7(gr,nBoxWidth*2,nBoxHeight*1,nBoxWidth,nBoxHeight);
drawFunc8(gr,nBoxWidth*3,nBoxHeight*1,nBoxWidth,nBoxHeight);
}

```

◀ previous | next ▶ 35

```

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing
...
public void init() {
  image = getToolkit().getImage(imageFile);
  setSize(430,300);
}

public static void main(String args[]){
  //Skapa en instans med givet namn på fönstret
  GraphicsDemoFrame f = new GraphicsDemoFrame("Olika
grafik");

  //Initiera
  f.init();

  // Visa fönstret
  f.setVisible(true);
}
}

```

◀ previous | next ▶ 36

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java2D: hur

- Kör vi JDK1.2 eller senare används egentligen Graphics2D istället för Graphics
- Graphics2D är definierad som en direkt subclass till Graphics
 - därför kan vi om vi vill använda den som en Graphics
- För att komma åt Graphics2D:s rutiner måste vi dock göra en cast (i tex paint-metoden)


```
public void paint(Graphics g) {
    Graphics2D g2 = (Graphics2D) g;
    // Nu kan vi använda rutiner i Graphics2D utan att
    // komplikatorn klagar
    ...
}
```

◀ previous | next ▶ 37

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java2D: några primitiver och enkla ritrutiner

- linjetjocklek


```
float width = 10;
BasicStroke bs = new BasicStroke(width);
gc2.setStroke(bs);
gc2.drawLine(10, 10, 100, 200);
```
- streckad linje


```
float [] dash = {10};
BasicStroke b = new BasicStroke(3.0f,
                               BasicStroke.CAP_BUTT,
                               BasicStroke.JOIN_MITER,
                               10.0f, dash, 0.0f);

g2.setStroke(b);
```

◀ previous | next ▶ 38

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

- polygon


```
GeneralPath p = new GeneralPath();
p.moveTo(0, 0);
p.lineTo(w/12, h/10);
p.lineTo(0, h/5);
p.closePath();
g2.draw(p);
```

◀ previous | next ▶ 39

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java2D: ange typ av linje och hörn

- I många grafikpaket kan man ange om ändor av linjer skall vara räta, rundade eller projicerade
 - CAP_BUTT, CAP_ROUND, CAP_SQUARE
- Motsvarande gäller för hörn i polygoner
 - BasicStroke.JOIN_MITER, BasicStroke.JOIN_ROUND, BasicStroke.JOIN_BEVEL

```
BasicStroke bs = new BasicStroke(20.0f,
                               BasicStroke.CAP_BUTT, BasicStroke.JOIN_ROUND);
```
- Antialiasing


```
g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
                    RenderingHints.VALUE_ANTIALIAS_ON);
```

 - Rita sen som vanligt

◀ previous | next ▶ 40

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

Java2D: rita figurer (urartat) exempel

- Ett exempel på olika typer av finesser i Java2D, kommentarerna muntligt. (Med Swing (JFrame) blir det hela helt analogt)

```
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;

public class Test extends Frame {
    private static int join[] = {BasicStroke.JOIN_MITER,
        BasicStroke.JOIN_ROUND, BasicStroke.JOIN_BEVEL };
    private static String desc[] = {"Mitered", "Rounded",
        "Beveled"};

    public Test(String s){
        super(s);
    }
}
```

◀ previous next ▶ 41

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```
...
public void drawDemoLines(int w, int h, Graphics2D g2) {
    AffineTransform at = g2.getTransform();
    g2.translate(w/3, h/9);
    g2.setColor(Color.black);
    for (int i=0; i < 3; i++) {
        BasicStroke bs = new BasicStroke(20.0f,
            BasicStroke.CAP_BUTT, join[i]);
        GeneralPath p = new GeneralPath();
        p.moveTo(0, 0);
        p.lineTo(w/12, h/10);
        p.lineTo(0,h/5);
        p.closePath();
        g2.setStroke(bs);
        g2.setRenderingHint(
            RenderingHints.KEY_ANTIALIASING,
            RenderingHints.VALUE_ANTIALIAS_OFF);
        g2.draw(p);
    }
}
```

◀ previous next ▶ 42

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```
...
Color oldColor = g2.getColor();
g2.setColor(Color.red);
g2.drawString(desc[i], w/12+20, h/10);
g2.setColor(Color.blue);
AffineTransform atTemp = g2.getTransform();
g2.translate(200, 0);
g2.rotate(Math.PI / 3);
g2.setRenderingHint(
    RenderingHints.KEY_ANTIALIASING,
    RenderingHints.VALUE_ANTIALIAS_ON);
g2.draw(p);
g2.setTransform(atTemp);
g2.translate(0, h/4);
g2.setColor(oldColor);
}
g2.setTransform(at);
}
```

◀ previous next ▶ 43

GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing

```
...
public void paint(Graphics g) {
    int x = 30;
    int y = 50;
    g.drawString("Streckade linjer", x, y);
    Graphics2D g2 = (Graphics2D) g;
    y += 10;
    float j = 1.1f;
    for (int i = 0; i < 10; i++, j += 1.0f) {
        float dash[] = { j };
        BasicStroke b = new BasicStroke(3.0f,
            BasicStroke.CAP_BUTT,
            BasicStroke.JOIN_MITER,
            10.0f, dash, 0.0f);

        g2.setStroke(b);
        g2.drawLine(5, y, 300, y);
        y += 5;
    }
}
```

◀ previous next ▶ 44

```
GRIP-F3 2D-transformationer. Java: AWT, 2D, Swing  
  
...  
    drawDemoLines(this.getWidth(),this.getHeight(),  
                  g2);  
}  
  
public static void main(String s[]) {  
    Frame f = new Test("A Window");  
    f.addWindowListener(new WindowAdapter() {  
        public void windowClosing(WindowEvent e){  
            System.exit(0);}  
    });  
    f.setSize(new Dimension(500,400));  
    f.setVisible(true);  
}  
}
```

◀ previous | next ▶ 45