Transformações Geométricas 3D

Rosane Minghim

Maria Cristina F. de Oliveira

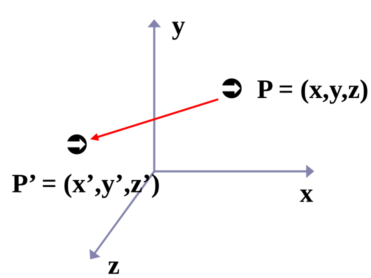
ICMC

Universidade de São Paulo

2006

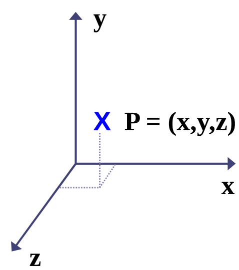
Translação

- Um ponto (objeto) é deslocado de uma posição para outra posição no mesmo espaço 3D

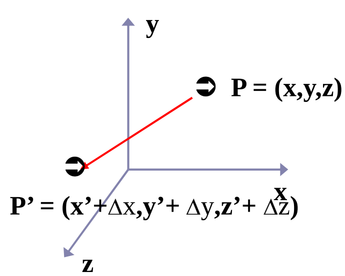


Sistemas de Coordenadas

- Representam uma forma de indexar e localizar elementos no espaço (que é 3D).
- Eixos com orientação formam o Sistema de Coordenadas Cartesianas
- Um ponto P é definido por uma tripla de coordenadas (x,y,z)



Translação



- Vetor Translação: $(\Delta_x\ \Delta_y\ \Delta_z)$
  $x' = x + \Delta_x$
  $y' = y + \Delta_y$
  $z' = z + \Delta_z$
- Representação vetorial do ponto: $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$
- Em coordenadas homogêneas:

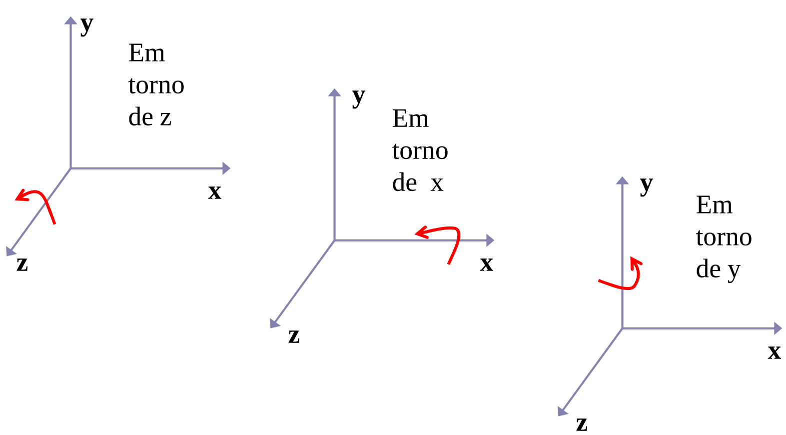
$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- Ou $P'=T(\Delta_x,\Delta_y,\Delta_z)*P$

Rotação

- Em 2D, a rotação se dá em torno de um ponto (1D). Em 3D é necessário especificar uma reta (2D), em torno da qual a rotação ocorrerá
- Um objeto é rotacionado de um ângulo específico em torno de um eixo
- Rotação em torno do eixo x
- Rotação em torno do eixo y
- Rotação em torno do eixo z
- Rotação em torno de um eixo generalizado

Orientação
Sentido Positivo da Rotação

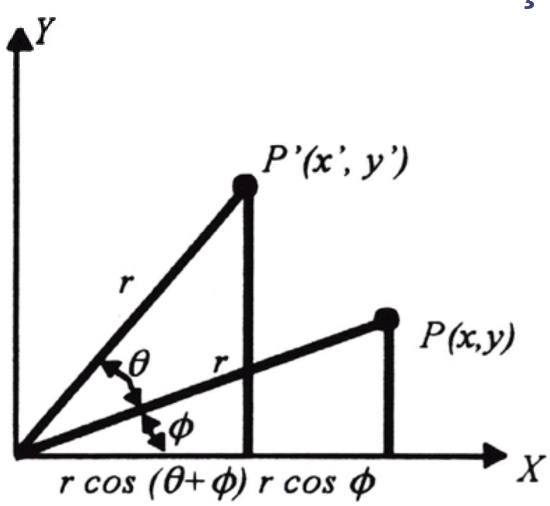


Regras para o sentido positivo de rotação

- Regra da mão direita
- Sentido oposto ao do relógio, quando observado do 'topo' do eixo, olhando para o centro
- Regras Específicas:

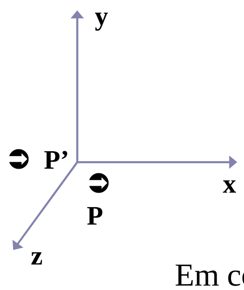
Eixo de Rotação	Direção da Rotação Positiva
x	y para z
y	z para x
z	x para y

Recordando algumas relações envolvidas na Rotação



Rotação em Torno do Eixo z

É dada por:



x’= x*cos(θ) - y*sen(θ)

y’= x*sen(θ) + y*cos(θ)

z’= z

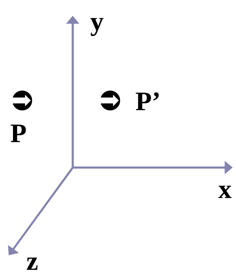
Em coordenadas homogêneas:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\mathrm{sen}(\theta) & 0 & 0 \\ \mathrm{sen}(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

ou:

P’= R_z(θ)*P

Rotação em torno do Eixo y



É dada por:

x’= z*sen(θ) + x*cos(θ)

y’= y

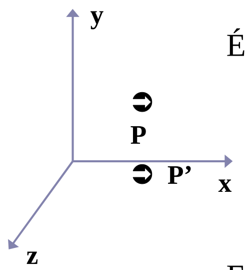
z’= z*cos(θ) - x*sen(θ)

Em coordenadas homogêneas:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & 0 & \mathrm{sen}(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\mathrm{sen}(\theta) & & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

ou: P’= R_y(θ)*P

Rotação em Torno do Eixo x



É dada por:

x’= x

y’= y*cos(θ) - z*sen(θ)

z’= y*sen(θ) + z*cos(θ)

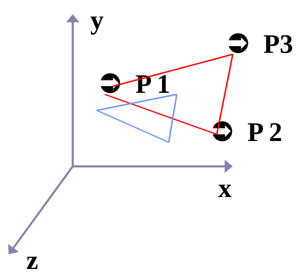
Em coordenadas homogêneas:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\mathrm{sen}(\theta) & 0 \\ 0 & \mathrm{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

ou:

P’= R_x(θ)*P

Escala



- **Vetor de fator de escala: (S_x S_y S_z)**

  Para cada ponto P = (x,y,z), o correspondente transformado é:

  x’ = x*S_x

  y’ = y*S_y

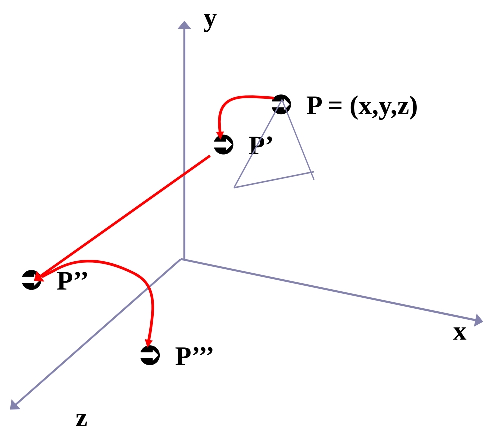
  z’ = z*S_z

- **Em coordenadas homogêneas:**

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & & S_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- **Ou P’=S(S_x,S_y,S_z)*P**

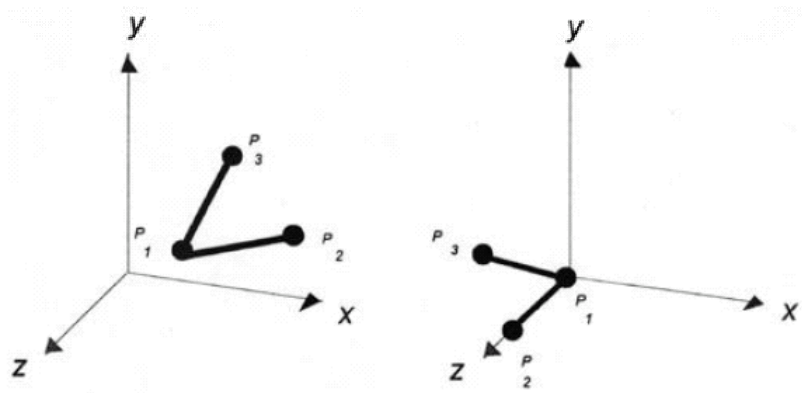
Composição de Transformações



Composição de Transformações

- É dada, como no caso 2D, pela sequência de transformações individuais
- Matematicamente, isto significa multiplicar as matrizes das transformação individuais.
- **Ex:** Deseja-se transformar um ponto P pelas operações de rotação em torno de x de α, seguida de uma translação de $(\Delta x1, \Delta y1, \Delta z1)$, e então de uma rotação em torno de z de β.
- Temos: $P' = \underline{R_x(\alpha)*P1}$; (1)
  $P'' = T(\Delta x1, \Delta y1, \Delta z1)*\underline{P1'}$ (2)
  $P''' = R_z(\beta)*P1''$; (3)
- Ou: $P''' = R_z(\beta)*T(\Delta x1, \Delta y1, \Delta z1)*R_x(\alpha)*P1$
- Isto é, a ordem da multiplicação das matrizes é inversa à ordem das transformações consecutivas!!!

Composição de Transformações
Exemplo



Composição de Transformações: Exemplo

1. Translação de P1 = (x_1,y_1,z_1) para a origem.

$$T_1(-x_1,-y_1,-z_1) = \begin{pmatrix} 1 & 0 & 0 & -x_1 \\ 0 & 1 & 0 & -y_1 \\ 0 & 0 & 1 & -z_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$P1' = T_1 * P1$

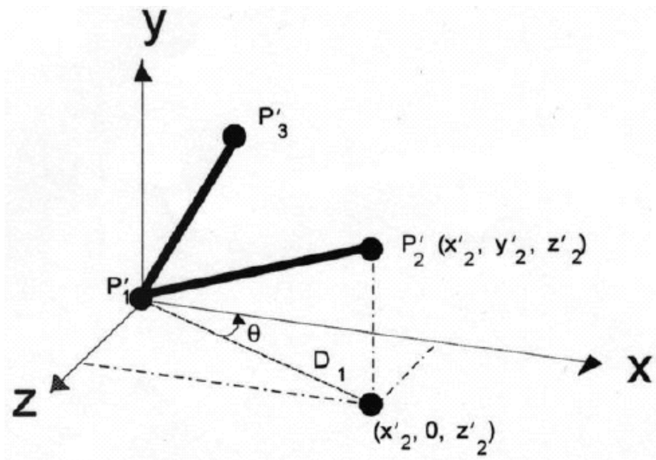
$P2' = T_1 * P2$

$P3' = T_1 * P3$

Exemplo (cont.)

2. Rotação de P1P2 em torno do eixo y, colocando P1P2 no plano yz.

(lembrando que cos(θ-90) = sin(θ), e cos(θ-90) = -cos(θ))



Exemplo (cont.)

2. Rotação de P1P2 em torno do eixo y, colocando P1P2 no plano yz.

$R_y(-(90-\theta)) = R_y(\theta-90)$

$\cos(\theta-90) = \mathrm{sen}(\theta) = z_2'/D_1 = (z_2-z_1)/D_1$

$\mathrm{sen}(\theta-90) = -\cos(\theta) = -x_2'/D_1 = -(x_2-x_1)/D_1$

$$D_1 = \sqrt{z'^2_2 + x'^2_2} = \sqrt{(z_2 - z_1)^2 + (x_2 - x_1)^2}$$

$P2'' = R_y(\theta-90) * P2' = (0\ y_2-y_1\ D_1\ 1)^T$

$P1'' = R_y(\theta-90) * P1' = (0\ 0\ 0\ 1)^T = P1'$

$P3'' = R_y(\theta-90) * P3'$ = (?? Faça a mão??)

Exemplo (cont.)

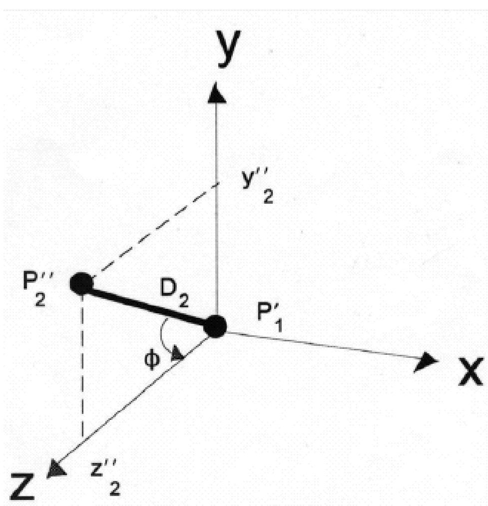
2. Rotação de P1P2 em torno do eixo y, colocando P1P2 no plano yz (Matriz).

$$R_y(\theta\text{-}90) = \begin{pmatrix} \cos(\theta-90) & 0 & \mathrm{sen}(\theta-90) & 0 \\ 0 & 1 & 0 & 0 \\ -\mathrm{sen}(\theta-90) & & \cos(\theta-90) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} (z2-z1)/D1 & 0 & -(x2-x1)/D1 & 0 \\ 0 & 1 & 0 & 0 \\ (x2-x1)/D1 & & (z2-z1)/D1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Exemplo(cont.)

3. Rotação de P1''P2'' em relação ao eixo x, colocando P1'P2'' sobre o eixo z.



Exemplo(cont.)

3. Rotação de P1''P2'' em relação ao eixo x, colocando P1'P2'' sobre o eixo z.

$\cos(\Phi) = z_2''/D_2$

$\text{sen}(\Phi) = y_2''/D_2$

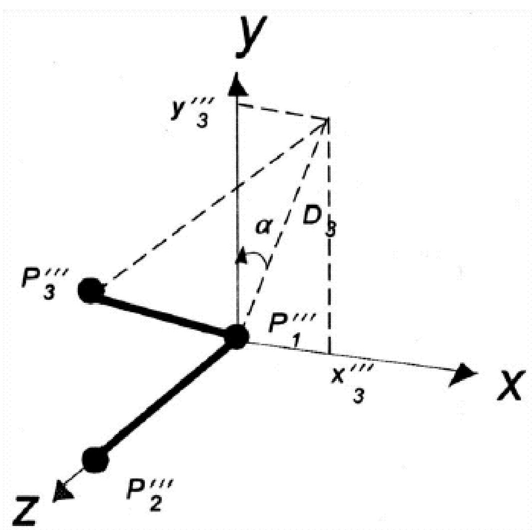
$D_2 = |P1''P2''| = |P1P2| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$

$P2''' = R_x(\Phi) * P2'' = R_x(\Phi) * R_y(\theta\text{-}90) * P2' =$

$= R_x(\Phi) * R_y(\theta\text{-}90) * T_1 * P2 = (0\ 0\ |P1P2|\ 1)^T$

$P3''' = R_x(\Phi) * R_y(\theta\text{-}90) * T_1 * P3$

4. Rotação de P1'''P3''' em relação ao eixo z, colocando P1P3 no plano yz.



4. Rotação de P1'''P3''' em relação ao eixo z, colocando P1P3 no plano yz.

Neste ponto, tem-se P3''' = (x_3''', y_3''',z_3''')

cos (α) = y_3'''/D_3

sen(α) = x_3'''/D_3

$D_3 = \sqrt{x_3'''^2 + y_3'''^2}$

P3''''= R_z (α) *P3'''

Assim, a matriz de Composição M, capaz de transformar a figura inicial na figura final, é:

M = R_z (α) * R_x (Φ) * R_y (θ-90) * $T_1(-x_1,-y_1,-z_1)$

Para todos os pontos da figura:

$P_{final} = M*P_{inicial}$

Rotação em Torno de Eixos generalizados

- **Quando paralelo a um dos eixos de coordenadas:**
  - **translade para o eixo de coordenada**
  - **rotacione**
  - **faça a translação inversa**
- **Quando não é paralelo e nenhum dos eixos:**
  - **Faça uma translação de forma que o eixo passe pela origem**
  - **Faça quantas rotações forem necessárias até que o eixo coincida com um dos eixos de coordenadas**
  - **Faça a rotação desejada**
  - **Realize a transformação inversa às rotações de ajuste do eixo**
  - **Faça a translação inversa à primeira translação**

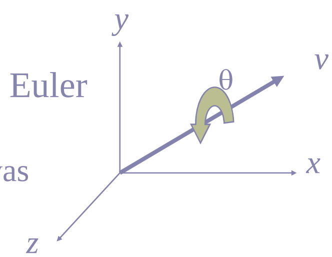
Rotação em Torno de Eixos generalizados

Matriz de rotação q em torno de um eixo arbitrário pode ser obtida pela concatenação de 3 rotações em torno dos eixos principais *x*, *y*, e *z*

$$\mathbf{R}(\theta) = \mathbf{R}_z(\theta_z)\ \mathbf{R}_y(\theta_y)\ \mathbf{R}_x(\theta_x)$$

$\theta_x\ \theta_y\ \theta_z$ são chamados ângulos de Euler

Note que as rotações não são comutativas



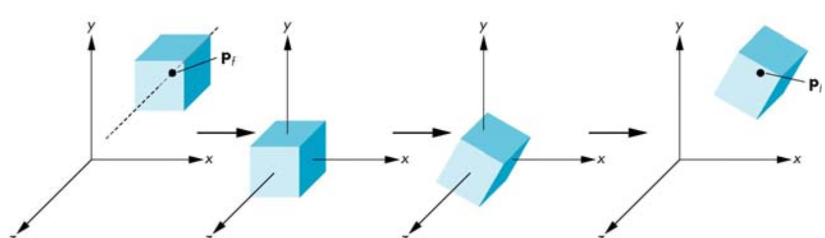
Rotação em torno de um ponto fixo arbitrário

Move ponto de referência p/ origem

Rotaciona

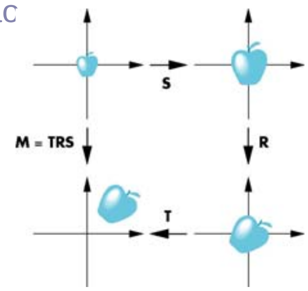
Move ponto arbitrário de volta

$\mathbf{M} = \mathbf{T}(p_f)\ \mathbf{R}(\theta)\ \mathbf{T}(-p_f)$



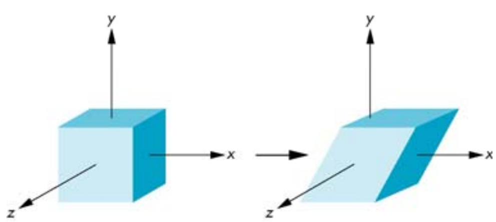
Instanciamento de objetos

- Ao modelar objetos: tipicamente parte-se de um objeto simples, centrado na origem, orientado segundo os eixos principais, de tamanho padrão
- Transformações de instanciamento aplicadas para transformar os vértices conforme o tamanho/posição/orientação desejadas para o objeto
  - escala
  - orienta
  - posiciona



Shear

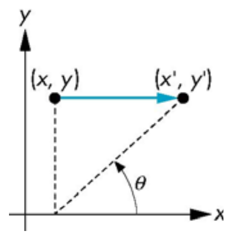
- Cisalhamento
- Equivalente a puxar as faces do objeto em direções opostas



Shear Matrix

Shear simples ao longo do eixo *x*

$x' = x + y \cot\theta$
$y' = y$
$z' = z$



$$\mathbf{H}(\theta) = \begin{bmatrix} 1 & \cot\theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Resumo

- Transformações exercem o papel, em CG, de apoiar o movimento de objetos ou câmeras:
  - para mudar o sistema de coordenadas
  - para apoiar interação
  - para criar animações
- Transformações:
  - Rotação
  - Translação
  - Escala
- Matrizes e Composição de Transformações em 3D

Resumo dos Parâmetros envolvidos nas Transformações

Translação	Δx	>0	movimento positivo no eixo
	Δy	<0	movimento negativo no eixo
	Δz		
Rotação	eixo,		
	ângulo	>0	movimento anti-horário ou pela regra da mão direita
Escala	Sx		
	Sy		Sx=Sy=Sz escala uniforme
	Sz		caso contrário, ocorre deformação
Composição	Sequência de Transformações		A ordem das transformações deve ser bem especificada

- Matrizes em OpenGL
- Exemplos
  - Cubo colorido
  - Cubo colorido rotacionado
  - Cubo colorido rodando
  - Cubo colorido rodando sobre diferentes eixos

Idle and Mouse callbacks

```
void spinCube()
{
  theta[axis] += 2.0;
  if( theta[axis] > 360.0 ) theta[axis] -=
  360.0;
  glutPostRedisplay();
}
```

```
void mouse(int btn, int state, int x, int y)
{
   if(btn==GLUT_LEFT_BUTTON && state == GLUT_DOWN)
           axis = 0;
   if(btn==GLUT_MIDDLE_BUTTON && state == GLUT_DOWN)
           axis = 1;
   if(btn==GLUT_RIGHT_BUTTON && state == GLUT_DOWN)
           axis = 2;
}
```

Display callback

```
void display()
{
   glClear(GL_COLOR_BUFFER_BIT |
  GL_DEPTH_BUFFER_BIT);
   glLoadIdentity();
   glRotatef(theta[0], 1.0, 0.0, 0.0);
   glRotatef(theta[1], 0.0, 1.0, 0.0);
   glRotatef(theta[2], 0.0, 0.0, 1.0);
   colorcube();
   glutSwapBuffers();
}
```

Note that because of fixed form of callbacks, variables such as **`theta`** and **`axis`** must be defined as globals

Camera information is in standard reshape callback

Bibliografia

Hearn, D. Baker, M. P. Computer Graphics, Prentice Hall, 1994

Foley, J et. al - Introduction to Computer Graphics, Addison-Wesley, 1993.

Watt, A. - Fundamentals of Three-Dimensional Computer Graphics, Addison-Wesley, 1989.

Angel, E. Interactive Computer Graphics: A Top-Down Approach with OpenGL, 3rd. Edition, Addison-Wesley, 2003