

Sistemas de Visão e Percepção Industrial

Aula Prática nº 2

Introdução às transformações geométricas.

Transformações com coordenadas homogéneas.

Sumário

- 1 Exercícios básicos com Transformações Geométricas
- 2 Transformações sobre objetos mais complexos
- 3 Criação de efeito de animação com operador set
- 4 Coordenadas homogêneas
- 5 Animações mais complexas

Tópicos principais

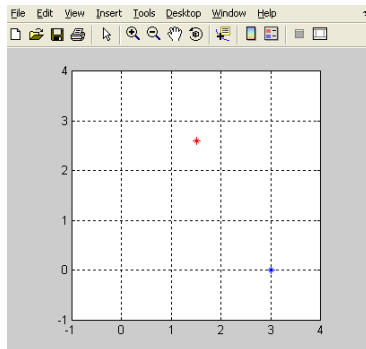
- Rotação no plano
- Rotação de pontos isolados
- Rotação de polígonos
- Animação de movimento em Matlab
- Translação no plano
- Combinações de translação e rotação
- Coordenadas homogêneas
- Matriz de transformação geométrica
- Combinações de matrizes de transformação

NB. A partir da aula 2 alguns exemplos de código estarão incompletos cabendo ao aluno completá-los. Adicionalmente, poderão ser propostos exercícios ditos "opcionais" assinalados com '**' e que representam desafios mais elaborados.

Exercício 1a)

Fazer a representação ('plot') do ponto P no plano $P=[3\ 0]'$, bem como a sua rotação de 60° .

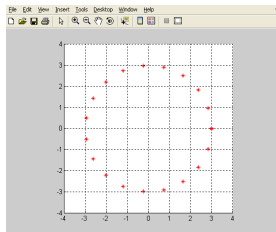
```
%% Ex1a
P=[3 0]';
plot(P(1),P(2),'*');
a=pi/3;
Rot=[cos(a) -sin(a)
      sin(a)  cos(a)];
axis([-1 4 -1 4]);
hold on; grid on; axis square
Pc=Rot*P;
plot(Pc(1),Pc(2),'*r');
```



Exercício 1b)

- Criar uma função para devolver a matriz de rotação, dado o ângulo como parâmetro: $M = \text{rota}(a)$
- Fazer a representação de N cópias de P em torno de uma circunferência.

```
P=[3 0]';  
axis([-4 4 -4 4]);  
hold on; grid on; axis square  
N=20;  
angs=linspace(0, 2*pi,N);  
for a=angs  
    Q=rota( )*P;  
    plot( , , '*r');  
    pause(0.1);  
end
```

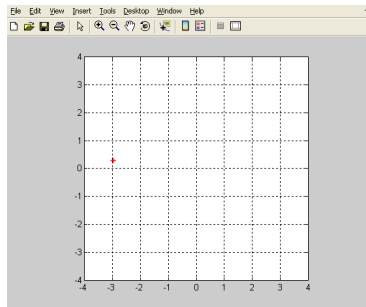


```
function M=rota(a)  
%rota - function that returns a rotation matrix in 2D  
%       accepts angle in radians and returns a 2x2 matrix  
M=[cos(a) -sin(a)  
   sin(a)  cos(a) ];
```

Exercício 2a)

- Repetir o exercício anterior mas com animação a simular movimento.
- Os “handles” gráficos – as suas propriedades

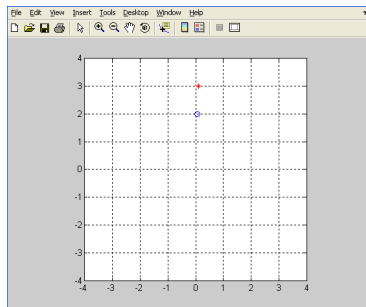
```
P=[3 0]';  
h=plot(P(1),P(2),'dr');  
axis([-4 4 -4 4]);  
hold on; grid on; axis square  
N=100;  
angs=linspace(0,2*pi,N);  
for a=angs  
    Q=rota( )*P;  
    set(h, 'Xdata', Q(1), 'YData' , Q( ));  
    pause(0.01);  
end
```



Exercício 2b)

- Ao exemplo anterior, acrescentar um segundo ponto $[2 \ 0]'$ que se "move" à mesma velocidade

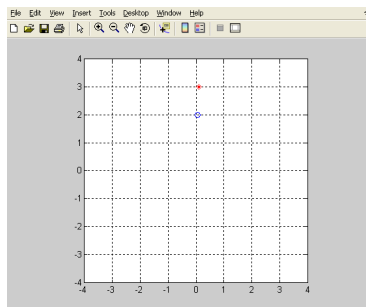
```
close all
P=[3 0]';
P2=[2 0]';
h=plot(P(1),P(2),'*r');
axis([-4 4 -4 4]);
hold on; grid on; axis square
N=500;
h2=plot( , , 'ob');
angs=linspace(0,10*pi,N);
for a=angs
    Q=rota(a)*P;
    Q2=      *P2;
    set(h, 'XData', Q(1), , Q(2));
    set(h2, , Q2(1), 'YData' , Q2(2));
    pause(0.01);
end
```



Exercício 2c)

- Alterar o programa anterior de modo a que o ponto interior se mova ao dobro da velocidade do exterior.

```
close all
P=[3 0]';
P2=[2 0]';
h=plot(P(1),P(2),'*r');
axis([-4 4 -4 4]);
hold on; grid on; axis square
N=500;
h2=plot(      , 'ob');
angs=linspace(0,10*pi,N);
for a=angs
    Q=rota(a)*P;
    Q2=      *P2;
    set(h,      ,      ,      ,      );
    set(h2,      ,      ,      , Q2(2));
    pause(0.01);
end
```



Considerações sobre objetos mais complexos e transformações

- Se um conjunto de pontos P for agrupado numa matriz, podem ser todos afetados da mesma forma e em simultâneo pela transformação:

$$Q = \text{rot}(\alpha) \times P$$

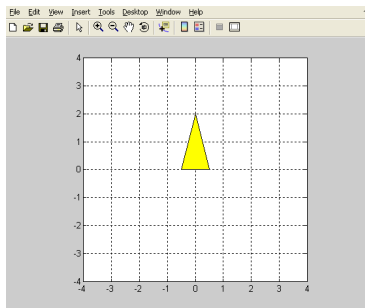
$$\begin{bmatrix} q_{1x} & q_{2x} & q_{3x} \\ q_{1y} & q_{2y} & q_{3y} \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} p_{1x} & p_{2x} & p_{3x} \\ p_{1y} & p_{2y} & p_{3y} \end{bmatrix}$$

Q1 Q2 Q3 P1 P2 P3

Exercício 3a)

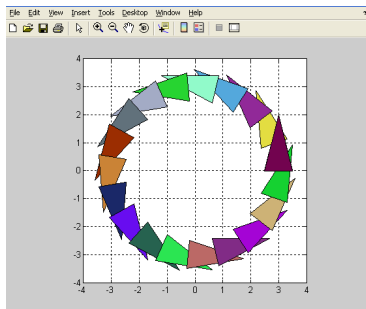
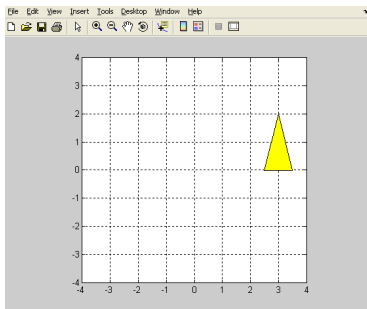
- Representar um triângulo e animá-lo em torno de uma circunferência:
 - Definir o triângulo num referencial base e desenhá-lo ('fill').
 - Fazer a animação do movimento.
- Aproveitar o código do exercício anterior e adaptar os 'plot' a 'fill' para resultar num efeito de animação.
 - Sugestão de código:

```
close all
P=[-0.5 0.5 0
    0 0 2];
h=fill(P(1,:),P(2,:), 'y');
axis([-4 4 -4 4]);
hold on; grid on; axis square
N=200;
angs=linspace(0,20*pi,N);
for a=angs
    Q=      *P;
    set(h, 'Xdata', Q(1,:),      ,      );
    pause(0.05);
end
```



Exercício 3b)

- Translacionar o triângulo de 3 unidades na horizontal e animá-lo em rotação.
- Sugestão para ornamentar o efeito:
 - Mudar a cor aleatoriamente durante o movimento!



- NB. Não se pretende representar múltiplos triângulos. A segunda figura simplesmente ilustra o triângulo a mudar de cor ao longo do movimento!

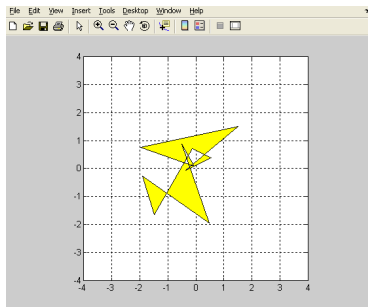
Sugestões para uma solução

```
%% Ex. 3b)

close all
P=[-0.5 0.5 0
    0    0    2];
P=P+repmat([3 0]',1,size(P,2));
h=fill(P(1,:), 'y');
axis([-4 4 -4 4]);
hold on; grid on; axis square
N=500;
angs=linspace(0,20*pi,N);
for a=angs
    Q=      )*P;
    set(h, 'Xdata', Q(1,:), 'YData' ,      );
    set(h, 'FaceColor',rand(1,3));
    %set(h, 'LineWidth',0.00001+a/2);
    pause(0.05);
end
```

Exercício 3c) – Opcional

- Gerar um polígono aleatório com 10 vértices e animá-lo com rotação sobre si próprio.
- Os vértices devem pertencer ao intervalo: $[-2; +2]^2$.
 - Usar a função `rand()`:
 - para gerar o polígono
 - para gerar uma cor aleatória



A transformação geométrica homogênea

- Matriz geral

$$\begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & v_x \\ \sin(\alpha) & \cos(\alpha) & v_y \\ 0 & 0 & 1 \end{bmatrix}$$

- Rotação pura

$$\begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Translação pura

$$\begin{bmatrix} 1 & 0 & v_x \\ 0 & 1 & v_y \\ 0 & 0 & 1 \end{bmatrix}$$

Exercício 4a) – Translação e Rotação

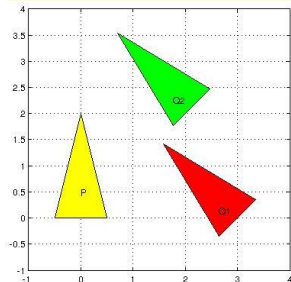
- Criar as funções para as transformações homogêneas no plano:
 - Rotação (função `rot()`): ficheiro `rot.m`
 - Translação (função `trans()`): ficheiro `trans.m`
- Usando o objeto $P = \begin{bmatrix} 0.5 & 0 & -0.5 \\ 0 & 2 & 0 \end{bmatrix}$, e as funções anteriores, fazer o seguinte:
 - Homogeneizar P ($P = [P; 1 \ 1 \ 1]$), e definindo $T1 = \text{trans}(3,0)$ e $T2 = \text{rot}(\pi/4)$, representar:
 - $Q1 = T1 * T2 * P$; %1ª rotação, 2ª translação
 - $Q2 = T2 * T1 * P$; %1ª translação, 2ª rotação
 - Comparar as diferenças.
- N.B. As funções `rot` e `trans` deverão ser guardadas numa pasta `lib` separada porque serão úteis noutras aulas. O "path" do matlab deve ser ajustado para incluir a pasta `lib`.

```
%% Ex 4a)
close all
P=[-0.5 0.5 0
    0 0 2
    1 1 1];
h=fill(P(1,:),P(2,:), 'y');
axis([-1 4 -1 4]);
hold on; grid on; axis square

T1=trans2(3,0);
T2=rot(pi/4);

Q1=T1*T2*P;
Q2=T1*T2*P;
h1=fill(Q1(1,:),Q1(2,:), 'r');
h2=fill(Q2(1,:),Q2(2,:), 'g');

text(mean(P(1,:)), mean(P(2,:)), 'P')
text(mean(Q1(1,:)), mean(Q1(2,:)), 'Q_1')
text(mean(Q2(1,:)), mean(Q2(2,:)), 'Q_2')
```



Exercício 4b) – Animação simples de objeto

- Implementar uma sequência de transformações de forma a simular o movimento animado do objeto P correspondente a uma translação de (0,3): `trans(0,3)`
- Fragmento de código sugerido:

```
%...  
h1=fill(P(1,:),P(2,:), 'r'); %initial object drawing  
%...  
% simulate object motion in translation.  
for t=linspace(0,3,20)  
    Q=trans(0,t)*P;  
    set(h1, 'XData', Q(1,:), 'YData', Q(2,:));  
    pause(0.05)  
end
```

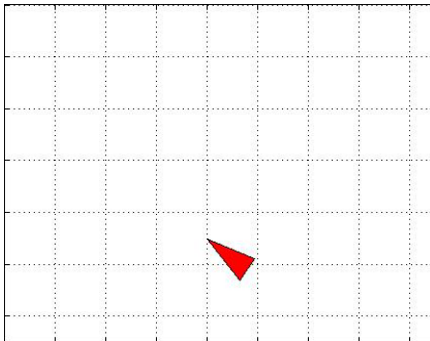
- **Nota:** A operação `set(h1,'XData',Q(1,:), 'YData',Q(2,:))`; anterior, poderia ser substituída pelo seguinte com o mesmo efeito:
 - `h1.XData=Q(1,:);`
 - `h1.YData=Q(2,:);`

Exercício 4c) – Animação composta de um objeto

- Implementar sequências de transformações de forma a simular o movimento animado do objeto P partindo da posição anterior do seguinte modo:
 - Parte1 → Trans(0,3) [é a alínea anterior: 4b)]
 - Parte2 → Rotação (+90°) em torno da origem (global)
 - Parte3 → Trans (-6,0) visto da origem (global)
 - Parte4 → Rotação (-90°) em torno de si próprio (local)
- O resultado esperado está ilustrado no filme animtri.avi
- NB. Para concretizar as transformações compostas, a ordem da multiplicação é determinante: **pré-multiplicar** significa operar visto do referencial **global**; **pós-multiplicar**, significa operar no referencial **local**.
- Seguem fragmentos de código para apoio ao exercício.

Visualização do movimento no Exercício 4c)

- 1 Trans(0,3)
- 2 Rotação ($+90^\circ$) em torno da origem (global)
- 3 Trans (-6,0) visto da origem (global)
- 4 Rotação (-90°) em torno de si próprio (local)



External player



Fragmentos para uma solução do exercício 4c)

```
%...
% simulate object motion in rotation starting from previous.
for a=linspace(0,pi/2,20)
    Q=rot(a)*trans(0,3)*P;    %why?
    set( h1, 'XData', Q(1,:), 'YData', Q(2,:));
    pause(0.05)
end
% (Part 3)
for t=linspace(0,-6,20)
    Q=trans2(t,0)*rot(pi/2)*trans(0,3)*P; %why?
    set( h1, 'XData', Q(1,:), 'YData', Q(2,:));
    pause(0.05)
end
% (Part 4)
%...
```

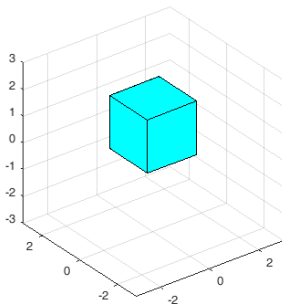
Exercício 5) Extensão a 3D – Opcional**

- Ilustrar a rotação de um cubo em 3D em torno de zz
- Sugestão: Descrição e visualização de um cubo em matlab:

```
V=[ 1  1  0  %P1  
   -1  1  0  %P2  
   -1 -1  0  %P3  
    1 -1  0  %P4  
    1  1  2  %P5  
   -1  1  2  %P6  
   -1 -1  2  %P7  
    1 -1  2 ];%P8
```

```
F=[ 1 2 3 4  
    5 6 7 8  
    1 2 6 5  
    1 5 8 4  
    3 7 8 4  
    2 6 7 3 ];
```

```
h=patch('Vertices',V,'Faces',F,'Facecolor','c');  
grid on
```



Exercício 6) Polígono manual – Opcional**

- Fazer um programa que permita ao utilizador definir manualmente um polígono arbitrário com o comando 'ginput' do Matlab.
- Depois de criado, o polígono deve realizar os mesmos movimentos definidos no exercício 4c), mas com transformações adicionais simultâneas para cada passo:
 - Parte1 - Trans(0,3) em simultâneo com o aumento de escala do objeto de modo a que este fique com o dobro da dimensão inicial (usar fator de escala global em matrizes de transformação homogêneas).
 - Parte2 - Rotação (+90°) em torno da origem (global) em simultâneo com diminuição da escala do objeto de modo a que passe do dobro da dimensão inicial (no início da parte 2), para a escala inicial (final da parte 2).
 - Parte3 - Trans (-6,0) visto da origem (global), em simultâneo com uma rotação do objeto em torno de si próprio no total de uma volta completa, 0° no início da parte 3, 360° no final da parte 3.
 - Parte4 - Rotação (-90°) em torno de si próprio (local), em simultâneo com uma transição suave de cor do amarelo inicial para o roxo, tal como definido pelo colormap "parula" (ver help do matlab).

Exercício 6) Filme demonstrativo

