# Transformationer i $\mathbb{R}^2$ och $\mathbb{R}^3$

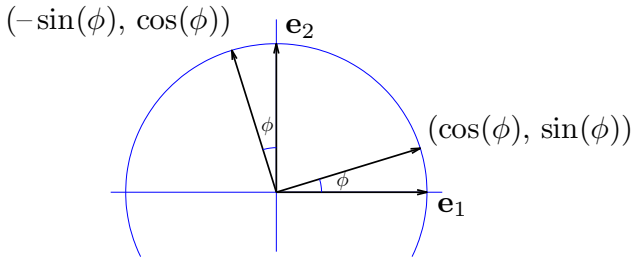
## Inledning

Vi skall se på några olika geometriska transformationer; rotation, skalning, translation och projektion.

Rotation och skalning är linjära avbildningar och kan beskrivas med standardmatriser, däremot är translation *inte* en linjär avbildning. När det gäller projektion får vi skilja på olika fall, t.ex. en ortogonal projektion i $\mathbb{R}^3$ på ett plan är en linjär avbildning om och endast om planet går genom origo.

## Rotation, skalning och translation

Som exempel på rotation tar vi: Låt $\mathbf{T} : \mathbb{R}^2 \to \mathbb{R}^2$ vara en rotation motsols med vinkeln $\phi$ runt origo.




Vi får

$$\mathbf{T}(\mathbf{e}_1) = \begin{bmatrix} \cos(\phi) \\ \sin(\phi) \end{bmatrix}, \quad \mathbf{T}(\mathbf{e}_2) = \begin{bmatrix} -\sin(\phi) \\ \cos(\phi) \end{bmatrix}$$

med standardmatrisen

$$\mathbf{A} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix}$$

Betraktar vi en punkt $\mathbf{x} = (x_1, x_2)$ i $\mathbb{R}^2$ som vi vill rotera runt origo så ges dess nya position av $\mathbf{T}(\mathbf{x}) = \mathbf{A}\mathbf{x}$.

Låt $\mathbf{S} : \mathbb{R}^2 \to \mathbb{R}^2$ vara en skalning med faktorerna $s_1$ och $s_2$ i respektive axelriktning. Vi får

$$\mathbf{S}(\mathbf{e}_1) = \begin{bmatrix} s_1 \\ 0 \end{bmatrix}, \quad \mathbf{S}(\mathbf{e}_2) = \begin{bmatrix} 0 \\ s_2 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix}$$

En translation med vektorn $\mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$ ges av avbildningen $\mathbf{F} : \mathbb{R}^2 \to \mathbb{R}^2$,

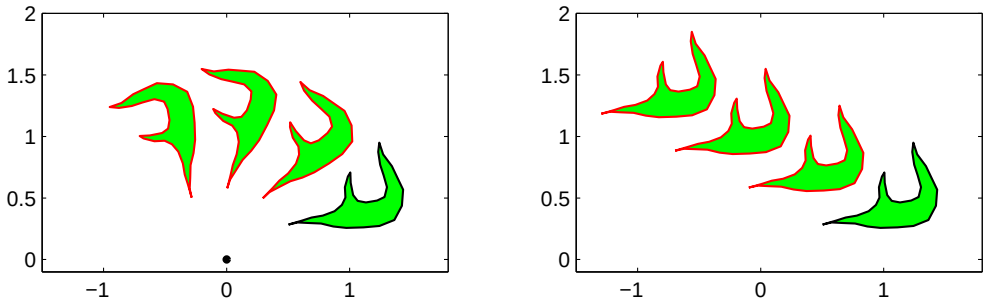
$$\mathbf{F}(\mathbf{x}) = \mathbf{x} + \mathbf{t} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

men här finns ingen standardmatris.

Vi illustrerar rotation och translation i $\mathbb{R}^2$ med MATLAB. I figuren nedan till vänster ser vi ett polygonområde med svart rand som vi roterar några gånger med vinkeln $\frac{\pi}{6}$ och ritar de roterade områdena med röd rand.

Vi tänker oss att vi redan skapat koordinater i radvektorerna `X` och `Y` som beskriver det ursprungliga området.

```
fill(X,Y,'g','edgecolor','k','linewidth',1), hold on
axis equal, axis([-1.5 2 -0.1 2]), pause(1)
v=pi/6; A=[cos(v) -sin(v); sin(v) cos(v)];
P=[X;Y];
for i=1:3
    P=A*P;   % Varje koordinatpar roteras med vinkeln pi/6
    fill(P(1,:),P(2,:),'g','edgecolor','r','linewidth',1), pause(1)
end
plot(0,0,'ko','linewidth',2,'markersize',2)  % origo
hold off
```




Till höger ser vi samma område i ursprungsläget (svart kant) samt några upprepade translationer (röd kant) med vektorn $t$.

```
fill(X,Y,'g','edgecolor','k','linewidth',1), hold on
axis equal, axis([-1.5 2 -0.1 2]), pause(1)
t=[-0.6;0.3];
P=[X;Y];
for i=1:3
    P=P+t*ones(size(X));
    fill(P(1,:),P(2,:),'g','edgecolor','r','linewidth',1), pause(1)
end
hold off
```

**Uppgift 1.** Rotera och translatera ett polygonområde ni genererar själva, t.ex. en triangel.

Betrakta en punkt $\mathbf{x} = (x_1, x_2, x_3)$ i $\mathbb{R}^3$. Rotation med vinkeln $\phi$ kring $x_1$-, $x_2$- och $x_3$-axlarna ges av $\mathbf{T} : \mathbb{R}^3 \to \mathbb{R}^3$, där $\mathbf{T}(\mathbf{x}) = \mathbf{A}\mathbf{x}$ med följande respektive standardmatriser

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotationen sker medurs sett i axelriktningarna.

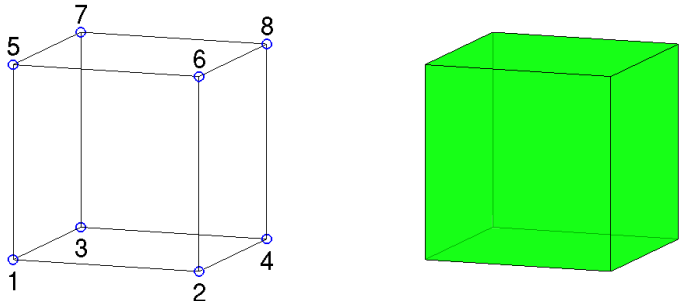
Skalning i $\mathbb{R}^3$ ges av

$$\mathbf{S}(\mathbf{x}) = \mathbf{B}\mathbf{x} = \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

och translation ges av

$$\mathbf{F}(\mathbf{x}) = \mathbf{x} + \mathbf{t} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

Nu ritar vi en kub som vi sedan skall transformera på lite olika sätt.




Vi ritar kuben enligt

```
H=[0 1 0 1 0 1 0 1      % H(:,j), j:te kolonnen i H, ger koordinater för punkt j
   0 0 1 1 0 0 1 1
   0 0 0 0 1 1 1 1];
S=[1 2 4 3              % S(i,:), i:te raden i S, ger nr på hörnpunkter på sida i
   1 2 6 5
   1 3 7 5
   3 4 8 7
   2 4 8 6
   5 6 8 7];            % size(S,1) ger antal sidor
figure(1), clf
hold on
for i=1:size(S,1)
    Si=S(i,:); fill3(H(1,Si),H(2,Si),H(3,Si),'g','facealpha',0.7)
end
hold off
axis equal, axis tight, axis off, view(20,10)
```

Lägg lite tid på att tänka igenom det vi gjort. Hörnpunkternas koordinater ligger som kolonner i matrisen `H`. På raderna i matrisen `S` har vi numren på hörnpunkterna på sidorna, t.ex. första raden (`1 2 4 3`) ger numren på hörnpunkterna som ger botten på kuben.

Antal kolonner i `H`, som ges av `size(H,2)`, är antalet hörnpunkter. Antalet rader i `S`, som ges av `size(S,1)`, är antalet sidor.

Med `for`-satsen ritar vi upp alla sidor. Vi plockar ut en sidas hörnpunkter med `Si=S(i,:)` och motsvarande kolonner i `H`, dvs. `H(:,Si)` ger koordinaterna för hörnpunkterna.

Vi ritar sidan med `fill3`, som fungerar som `fill` fast i rummet. Här måste vi separera $x_1$-, $x_2$- och $x_3$-koordinaterna. Med `H(1,Si)` får vi $x_1$-koordinaterna för hörnpunkterna på sidan, osv.

Vi får kuben lite lätt genomskinlig med `'facealpha',0.7`. För solid kub sätt `'facealpha',1` eller utelämna. Med `axis equal` får vi skalor på axlarna så att en kub ser ut som en kub och inte blir tillplattad. Vidare ger `axis tight` ett koordinatsystem *utan* luft runt kuben som vi ritat och `axis off` gör att axlarna och skalmarkeringar inte syns. Med `view(20,10)` ges betraktelsevinklar, se gärna hjälptexterna.

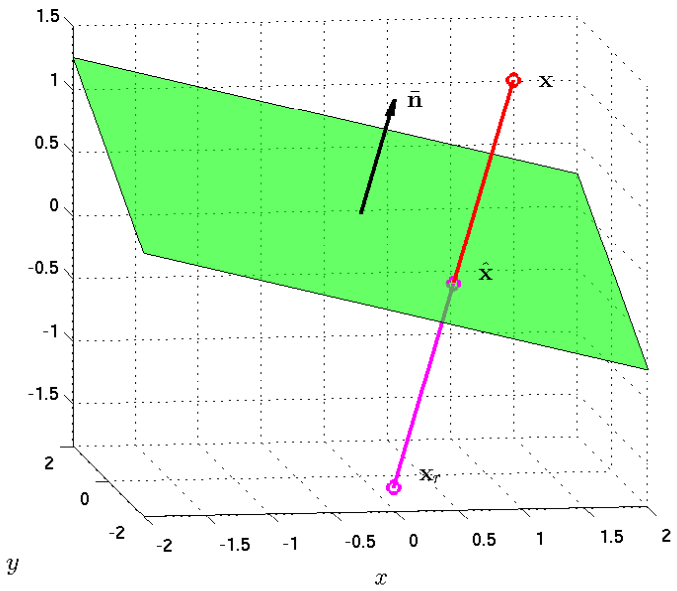
**Uppgift 2.** Rotera kuben runt någon axel. Gör en translation av kuben bort från origo. Rotera den åter runt någon axel.

# Ortogonal projektion

Vi skall bestämma ortogonala projektionen på planet

$$ax + by + cz = d$$

Planet har normalvektorn $\mathbf{n} = (a, b, c)$. I bilden har vi ritat enhetsnormal $\bar{\mathbf{n}}$ och ortogonala projektionen $\hat{\mathbf{x}}$ längs enhetsnormalen av en punkt $\mathbf{x}$.




Vi gör ansatsen $\hat{\mathbf{x}} = \mathbf{x} + \alpha\,\mathbf{n}$, där $\alpha$ skall bestämmas så att $\hat{\mathbf{x}}$ ligger på planet. Ekvationen för planet kan skrivas

$$\mathbf{n} \cdot \hat{\mathbf{x}} = d$$

och sätter vi in ansatsen får vi

$$\mathbf{n} \cdot (\mathbf{x} + \alpha\mathbf{n}) = \mathbf{n} \cdot \mathbf{x} + \alpha\,\mathbf{n} \cdot \mathbf{n} = d$$

och därmed

$$\alpha = \frac{d - \mathbf{n} \cdot \mathbf{x}}{\mathbf{n} \cdot \mathbf{n}}$$

För speglingen av $\mathbf{x}_r$ av punkten $\mathbf{x}$ i planet gäller

$$\mathbf{x}_r = \mathbf{x} + 2\alpha\,\mathbf{n}$$

med samma val av $\alpha$.

Nu skall vi i MATLAB rita planet $ax + by + cz = d$, för $a = 1, b = -1, c = 4$ och $d = 1$. Eftersom $c \neq 0$ så kan vi lösa ut $z$, i annat fall får vi modifiera koden

```
xmin=-2; xmax=2; ymin=-2; ymax=2;
a=1; b=-1; c=4; d=1;
X=[xmin xmax xmax xmin]; Y=[ymin ymin ymax ymax];
Z=(d-a*X-b*Y)/c;
figure(1), clf
fill3(X,Y,Z,'g','facealpha',0.7)
xlabel('x'), ylabel('y')
grid on
```

Resultatet blir planet i figuren ovan.

**Uppgift 3.** Rita planet vi just tittade på. Bestäm normalvektorn och rita ut den som ett streck från en punkt på planet. Välj en punkt $\mathbf{x}$, rita ut den, bestäm dess ortogonala projektion $\hat{\mathbf{x}}$ på planet och rita ut även denna punkt. Rita även ut speglingen $\mathbf{x}_r$ av $\mathbf{x}$.

**Uppgift 4.** Om $d = 0$ i ekvationen för planet så går planet genom origo. Då blir ortogonala projektionen en linjär avbildning. Vad blir standardmatrisen?

**Uppgift 5.** Rita planet från uppgift 3. I samma bild skall ni rita en translaterad kub. Translationen skall vara sådan att kuben inte skär planet. Rita därefter speglingen av kuben i planet.