Homogena koordinater och datorgrafik

Inledning

Vi såg tidigare på några geometriska transformationer; rotation, skalning, translation och projektion. Rotation och skalning är linjära avbildningar och vi såg på standardmatriser för dessa, däremot är ju translation inte en linjär avbildning.

Vi såg också på ortogonal projektion i $\mathbb{R}^3$ på ett plan och detta är en linjär avbildning om och endast om planet går genom origo.

Det är väldigt praktiskt att kunna beskriva en transformation med en matrismultiplikation. Med s.k. homogena koordinater kan vi beskriva även translation och projektion på ett plan som inte går genom origo med hjälp av matrismultiplikation.

Affina avbildningar och homogena koordinater

En avbildning $\mathbf{F}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}$ kallas affin. Vi har redan sett exempel på sådana bl.a. i samband med matrisiterationer.

En affin avbildning $\mathbf{F}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}$ kan beskrivas med matrismultiplikation om vi inför en extra koordinat w enligt

$$\begin{bmatrix} \hat{\mathbf{x}} \\ \hat{w} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ w \end{bmatrix}$$

där $\mathbf{0}^T$ är en rad med nollor.

Vi kommer då ha

$$\begin{aligned} \hat{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{b}w \\ \hat{w} &= w \end{aligned}$$

Så om låter vi $w = 1$ så får vi $\hat{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}$ och $\hat{w} = 1$.

Detta kallas *homogena koordinater* och vi kan läsa om detta i Lay kap. 2.7.

Translation med en vektor $\mathbf{t}$ ges av den affina avbildningen

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

Vi har $\mathbf{A} = \mathbf{I}$ enhetsmatrisen och $\mathbf{b} = \mathbf{t}$, dvs. translationsvektorn, och vi kan beskriva translationen med matrismultiplikation

$$\begin{bmatrix} \hat{\mathbf{x}} \\ \hat{w} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ w \end{bmatrix} \tag{1}$$

Uppgift 1. Bestäm inversen till matrisen i (1). Tolka vad multiplikation med inversen motsvarar geometriskt.

Tidigare bestämde vi ortogonala projektionen på ett plan

$$ax + by + cz = d$$

där a, b, c och d är konstanter.

Om $\mathbf{x}$ är den punkt vi projicerar och $\hat{\mathbf{x}}$ är projektionen så gäller

$$\hat{\mathbf{x}} = \mathbf{x} + \alpha\,\mathbf{n}, \qquad \alpha = \frac{d - \mathbf{n}\cdot\mathbf{x}}{\mathbf{n}\cdot\mathbf{n}}$$

där $\mathbf{n} = (a, b, c)$ är en normalvektor till planet.

Om $d = 0$, dvs. planet går genom origo, har vi

$$\hat{\mathbf{x}} = \mathbf{x} + \alpha\,\mathbf{n} = \mathbf{x} - \frac{\mathbf{n}\cdot\mathbf{x}}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n}$$

Eftersom $\mathbf{n}\cdot\mathbf{x}$ är en skalär så gäller

$$\hat{\mathbf{x}} = \mathbf{x} - \frac{(\mathbf{n}\cdot\mathbf{x})}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n} = \mathbf{x} - \frac{1}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n}\,(\mathbf{n}\cdot\mathbf{x})$$

Vidare gäller att $\mathbf{n}\cdot\mathbf{x} = \mathbf{n}^T\mathbf{x}$, dvs. skalärprodukten kan beräknas genom att $\mathbf{n}^T$ som är en radvektor matrismultipliceras med kolonnvektorn $\mathbf{x}$ och vi får

$$\hat{\mathbf{x}} = \mathbf{x} - \frac{1}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n}\,(\mathbf{n}^T\mathbf{x}) = \mathbf{x} - \frac{1}{\mathbf{n}\cdot\mathbf{n}}\,(\mathbf{n}\,\mathbf{n}^T)\,\mathbf{x} = \left(\mathbf{I} - \frac{1}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n}\,\mathbf{n}^T\right)\mathbf{x}$$

där vi utnyttjade att $\mathbf{n}\,(\mathbf{n}^T\mathbf{x}) = (\mathbf{n}\,\mathbf{n}^T)\,\mathbf{x}$. Observera att $\mathbf{n}\,\mathbf{n}^T$ är en matris.

Vi har kommit fram till

$$\hat{\mathbf{x}} = \left(\mathbf{I} - \frac{1}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n}\,\mathbf{n}^T\right)\mathbf{x} = \mathbf{P}\mathbf{x}$$

Alltså en linjär avbildning med standardmatrisen $\mathbf{P}$.

Om $d \neq 0$, dvs. planet går *inte* genom origo, har vi

$$\hat{\mathbf{x}} = \mathbf{x} + \alpha\,\mathbf{n} = \mathbf{x} + \frac{d - \mathbf{n}\cdot\mathbf{x}}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n} = \left(\mathbf{I} - \frac{1}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n}\,\mathbf{n}^T\right)\mathbf{x} + \frac{d}{\mathbf{n}\cdot\mathbf{n}}\,\mathbf{n} = \mathbf{P}\mathbf{x} + \beta\mathbf{n}$$

Detta är en affin avbildning som vi beskriver med

$$\begin{bmatrix} \hat{\mathbf{x}} \\ \hat{w} \end{bmatrix} = \begin{bmatrix} \mathbf{P} & \beta\mathbf{n} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ w \end{bmatrix}$$

dvs. med hjälp av matrismultiplikation.

Uppgift 2. En linjär avbildning $\mathbf{T}(\mathbf{x}) = \mathbf{A}\mathbf{x}$ kan också bäddas in med homogena koordinater. Hur ser blockmatrisen ut?

Uppgift 3. Läs nu Lay kap. 2.7 ordentligt, speciellt om perspektiv projektion. Där beskrivs hur man får fram perspektivprojektion av en 3-dimensionell figur på xy-planet när betraktaren är i punkten $(0, 0, d)$.

Generalisera nu detta till en godtycklig betraktelsepunkt (b, c, d). Ni får förutsätta att figuren är mellan betraktaren och planet. Observera hur boken lägger koordinatsystemet.

Datorgrafik

Man kan i MATLAB åstadkomma relativt avancerad datorgrafik. Vi har ingen möjlighet att hinna med att se på allt utan vi får nöja oss med att rita några figurer och samtidigt se på några olika kommandon för grafik.

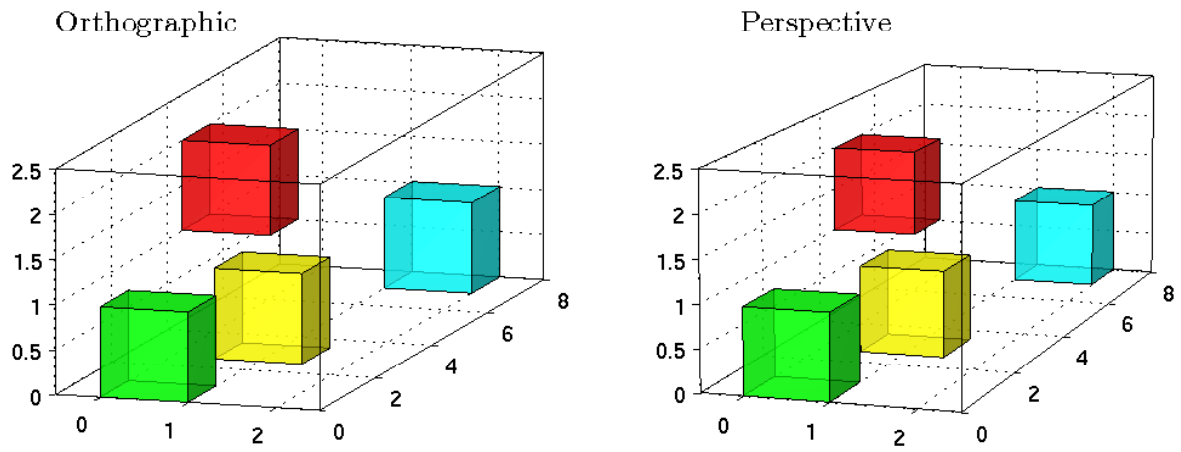
Läroböcker om MATLAB ger dåligt stöd för mer avancerad grafik och vi får istället gå till hjälptexterna i MATLAB och läsa. Det finns däremot en omfattande litteratur om datorgrafik som eget ämne.

Vi skall ge två lite större exempel. Det första exemplet, som kommer i denna text, är en tub runt en kurva i rummet och det andra exemplet, som kommer i texten för nästa vecka, är en ikosaeder som vi ritar med stänger och noder.

Men först ser vi hur man kan välja projektionstyp och betraktelsepunkt i en bild vi redan har ritat (några lika stora kuber utströdda i rummet).

```
camproj orthographic
campos([15 -40 10])
```

Lika stora objekt kommer se lika stora ut oavsett hur långt borta de är från betraktaren. Detta är standard projektionstypen i MATLAB.



Vi kan välja perspektivprojektion med

```
camproj perspective
campos([15 -40 10])
```

Lika stora objekt kommer se mindre ut ju längre bort från betraktaren de ligger. Skillnaden är inte så stor, men det är ändå det som behövs för att göra en realistisk bild av t.ex. ett hus.

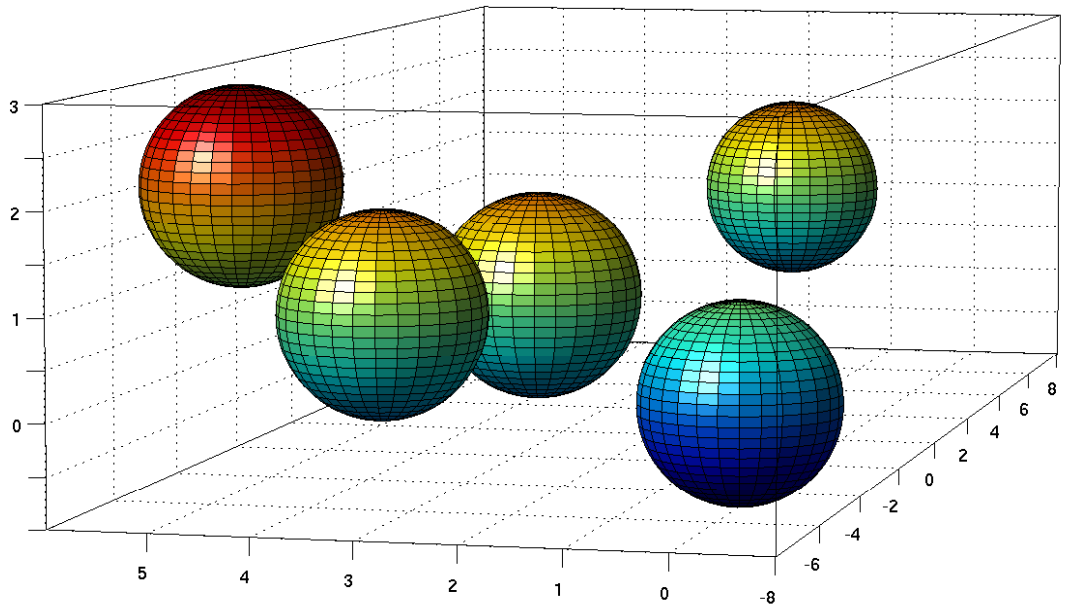
I bilden ovan har vi lagt på ljus med `camlight` och valt material med `material` enligt

```
camlight headlight  % left, right
material shiny      % metal, dull
```

Läs gärna i hjälptexterna om kommandona ovan.

Uppgift 4. Rita nu några bollar i rummet, lägg på lite belysning, välj projektionstyp och rotera det hela några varv. Använd `sphere`.

Något liknande det här kommer ni nog fram till



Vi kan rotera bollsamlingen genom att använda `camorbit` på olika sätt

```
camlight left
for i=1:200
    camorbit(5,0)
    drawnow; pause(0.05)
end
```

```
h=camlight('left');
for i=1:200
    camorbit(5,0)
    camlight(h,'left')
    drawnow; pause(0.05)
end
```

Pröva och skriv in koden. Alternativet till vänster lägger på belysning och sedan följer vi med kameran runt. I alternativet till höger följer även belysningskällan med kameran runt. Pausen väljer man så att det går lagom fort på datorn man använder.

Tub runt kurva i $\mathbb{R}^3$

Nu till vårt exempel. Vi gör en funktion som lägger en tub runt en kurva $\mathbf{x}(t)$ i rummet.

Funktionen behöver som indata förutom kurvan $\mathbf{x}(t)$, beräknad i ett antal punkter, även första derivatan $\mathbf{x}'(t)$, beräknad i samma punkter. Radien $r(t)$ på tuben kan få variera längs kurvan och även radiens värde för de olika punkterna skall vara indata. Vidare skall antal segment i tuben m anges.

```
function [U,V,W]=tuben(x,y,z,r,m,dx,dy,dz)
l=length(x);s=linspace(0,2*pi/1,m);
U=zeros(l,m);V=U;W=U;
p=[x(1);y(1);z(1)];
t=[dx(1);dy(1);dz(1)];
if sum(t'==[0 0 0])==3
    error(['kurvan saknar tangent i punkten [' num2str(p) ']^T'])
end
```

```
N=null(t');n=N(:,1);b=N(:,2);t=t/norm(t)
                        % Det första lokala koordinatsystemet

XI=p*ones(1,m)+n*r(1)*cos(s)+b*r(1)*sin(s);
U(1,:)=XI(1,:);V(1,:)=XI(2,:);W(1,:)=XI(3,:);
for k=2:l
    t1=t;n1=n;b1=b;
    p=[x(k);y(k);z(k)];
    t=[dx(k);dy(k);dz(k)];t=t/norm(t);
    if sum(t'==[0 0 0])==3
      error(['kurvan saknar tangent i punkten [' num2str(p) ']^T'])
    end
    d=norm(t-t1);
    if d~=0             % Om d=0 så behåller vi n och b
        a=2*asin(d/2); % Den nya tangenten t har vridit vinkeln a i
                       % förhållande till gamla tangenten
        v=cross(t1,t);v=v/norm(v); % Rotationsaxeln
        u=cross(v,t1);
        M=[cos(a) -sin(a) 0;sin(a) cos(a) 0;0 0 1];
                       % Rotationsmatrisen i ON-basen {t1, u, v} för rotation
                       % vinkeln a kring axeln v (svarar mot z-axeln)

        P=[t1 u v];    % Basbytesmatrisen
        R=P*M*P';      % Rotationsmatrisen i standardbasen
        n=R*n1;b=R*b1; % "Nya basvektorer (roterade vinkeln a),
                       % notera att t=R*t1
    end

    XI=p*ones(1,m)+n*r(k)*cos(s)+b*r(k)*sin(s);
    U(k,:)=XI(1,:);V(k,:)=XI(2,:);W(k,:)=XI(3,:);
end
```

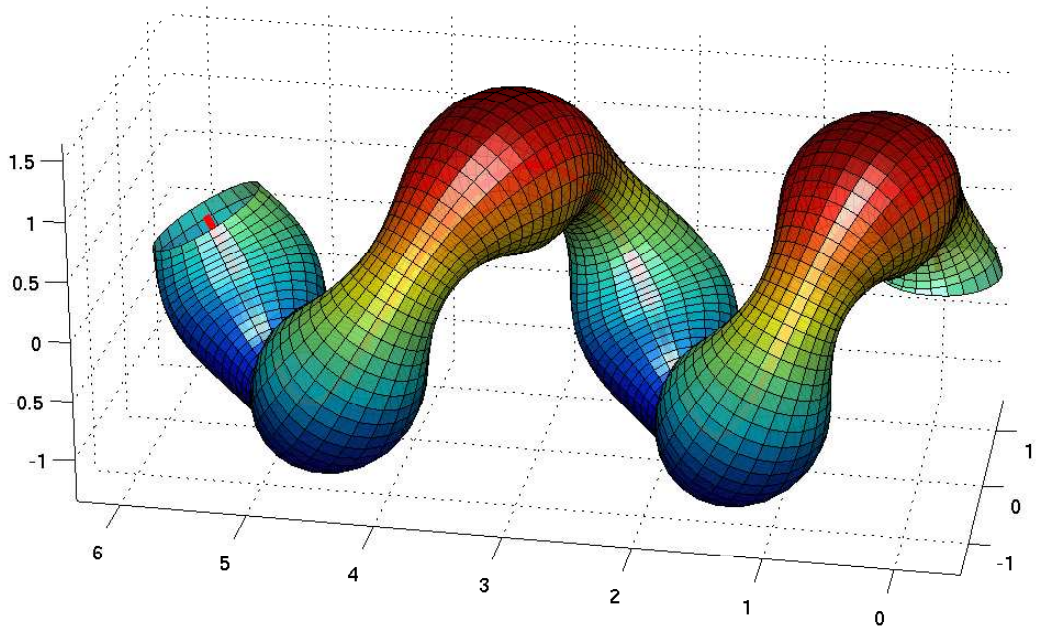
Vi väljer kurvan $\mathbf{x}(t) = (\cos(t), \sin(t), at^{5/4})$, $0 \leq t \leq 4\pi$, med $a = \frac{1}{4}$ och en radie på tuben $r(t)$ som varierar längs kurvan. Derivan blir $\mathbf{x}'(t) = (-\sin(t), \cos(t), \frac{5}{4}at^{1/4})$. Vi väljer $n = 40$ punkter längs kurvan och tar $m = 25$ tubsegment.

Vi ritar upp, lägger på lite belysning med `camlight` och väljer projektionstyp med `camproj` enligt

```
n=150;m=25;a=0.25;t=linspace(0,4*pi,n);
x=cos(t);z=sin(t);y=a*t.^(5/4);
dx=-sin(t);dz=cos(t);dy=a*(5/4)*t.^(1/4);
r=0.5*(1-0.3*sin(5*t));
[X,Y,Z]=tuben(x,y,z,r,m,dx,dy,dz);
figure(1)
hold off
surf(X,Y,Z,'FaceAlpha',0.85)
hold on
plot3(x,y,z,'r','LineWidth',2)
axis equal vis3d
```

```
camlight left
camproj perspective
shg
```

och får bilden



Pröva gärna och skriv in koden för att rotera kameran runt objektet.

Uppgift 5(a). Gör nu två torusar som kedjar i varandra, dvs. gör två olika kurvor som ni lägger tuber runt. Lägg på lite belysning, välj projektionstyp och rotera det hela några varv.

(b). Hitta på en egen kurva i rummet och lägg en tub runt den. Låt gärna radien variera längs kurvan. Lek!