# Linjära avbildningar

### 1 Inledning

Vi skall se på några geometriska transformationer; *rotation, skalning* och *translation*. Rotation och skalning är linjära avbildningar och kan beskrivas med standardmatriser, däremot är translation *inte* en linjär avbildning. Avslutningsvis skall vi se lite på en molekylmodell.

## 2 Rotation, skalning och translation i $\mathbb{R}^2$

Betrakta en punkt  $\mathbf{x} = (x_1, x_2)$  i  $\mathbb{R}^2$ . Rotation moturs med vinkeln  $\phi$  runt origo ges av  $\mathbf{T} : \mathbb{R}^2 \to \mathbb{R}^2$ , där  $\mathbf{T}(\mathbf{x}) = \mathbf{A}\mathbf{x}$  med standardmatrisen

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

Motsvarande för skalning och translation blir

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

respektive

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

För translation finns ingen standardmatris eftersom det inte är en linjär avbildning.

Vi illustrerar 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 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'), 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'), pause(1)
end
plot(0,0,'ko','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'), 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'), pause(1)
end
hold off
```

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

Innan vi går vidare är det viktigt att tänka igenom det vi gjorde: Vi antar att vi redan har två radvektorer med koordinater för polygonområde och placerar dem i en matris med P=[X;Y] så att koordinaterna för punkterna ligger som kolonner.

$$\mathbf{P} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_2 & \cdots & \mathbf{P}_n \end{bmatrix}, \text{ med } \mathbf{P}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

I MATLAB roterar vi med P=A\*P så att varje punkt roteras  $\widehat{\mathbf{P}}_i = \mathbf{A}\mathbf{P}_i$ , fast alla samtidigt.

$$\widehat{\mathbf{P}} = \left[\widehat{\mathbf{P}}_1 \ \widehat{\mathbf{P}}_2 \ \cdots \ \widehat{\mathbf{P}}_n\right] = \left[\mathbf{A}\mathbf{P}_1 \ \mathbf{A}\mathbf{P}_2 \ \cdots \ \mathbf{A}\mathbf{P}_n\right] = \mathbf{A}\left[\mathbf{P}_1 \ \mathbf{P}_2 \ \cdots \ \mathbf{P}_n\right] = \mathbf{A}\mathbf{P}$$

Vi translaterar i MATLAB med P=P+t\*ones(size(X)) så att varje punkt translateras  $\widehat{\mathbf{P}}_i = \mathbf{P}_i + \mathbf{t}$ , fast alla samtidigt.

$$\widehat{\mathbf{P}} = \begin{bmatrix} \widehat{\mathbf{P}}_1 \ \widehat{\mathbf{P}}_2 \ \cdots \ \widehat{\mathbf{P}}_n \end{bmatrix} = \begin{bmatrix} \mathbf{P}_1 + \mathbf{t} \ \mathbf{P}_2 + \mathbf{t} \ \cdots \ \mathbf{P}_n + \mathbf{t} \end{bmatrix} =$$
$$= \mathbf{P} + \begin{bmatrix} \mathbf{t} \ \mathbf{t} \ \cdots \ \mathbf{t} \end{bmatrix} = \mathbf{P} + \begin{bmatrix} t_1 \ t_1 \ \cdots \ t_1 \\ t_2 \ t_2 \ \cdots \ t_2 \end{bmatrix} = \mathbf{P} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \begin{bmatrix} 1 \ 1 \ \cdots \ 1 \end{bmatrix}$$

### **3** Rotation, skalning och translation i $\mathbb{R}^3$

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.

Motsvarande för skalning och translation blir

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

respektive

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

Nu skall ni rita en tetraeder som ni sedan skall transformera på lite olika sätt. För att underlätta ritandet av tetraedern visar vi hur man kan rita en kub.



Vi ritar en enhetskub enligt

```
H = [0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1
                     % H(:,j), j:te kolonnen i H, ger koordinater för hörnpunkt j
   0 0 1 1 0 0 1 1
                     % size(H,2) ger antal hörnpunkter
   0 0 0 0 1 1 1 1]:
                     % S(i,:), i:te raden i S, ger nr på hörnpunkter på sida i
S=[1 2 4 3
   1 2 6 5
   1 3 7 5
   3487
   2486
   5687];
                     % 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),'b','facealpha',0.7)
end
axis equal, axis tight, axis off, hold 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. Rita en liksidig tetraeder.



Vi kan ta hörnpunkter på enhetssfären med hörnpunkternas koordinater som

$$\left(\frac{2\sqrt{2}}{3}, 0, -\frac{1}{3}\right), \quad \left(-\frac{\sqrt{2}}{3}, \pm\sqrt{\frac{2}{3}}, -\frac{1}{3}\right), \quad (0, 0, 1)$$

**Uppgift 3.** Rotera tetraedern runt någon axel. Gör en translation av tetraedern bort från origo. Rotera den åter runt någon axel. Välj ett koordinatsystem så att tetraedern syns i alla lägen den hamnar i.

**Uppgift 4.** Nu skall vi göra om uppgift 3, fast som en *animation*. När ni roterade och translaterade tetraedern, så behöll ni i figuren resultatet efter varje rotation och translation. På så sätt fick ni en figur med många tetraedrar och det var bra för att förstå vad som hände. Nu vill vi bara se en enda tetraeder som roterar, blir translaterad för att sedan rotera igen. Rotationerna skall göras genom att upprepade gånger rotera med en liten vinkel så att vi får en jämn rörelse. För varje liten rotation skall vi sudda figuren (clf) och rita upp tetraedern i den senaste positionen samt lägga in en kort paus (pause) så att vi hinner se resultatet. På motsvarande sätt med translationen. Som hjälp ger vi följande början till ett script där vi ser på den första rotationen

```
ax=[...]; % Gränser för koordinataxlarna, ax=[xmin xmax ymin ymax zmin zmax].
          % Dessa sätts så att hela scenen får plats
v=...;
          % Liten vinkel för upprepad rotation, t.ex. v=pi/8
          % Rotationsmatrisen
A = [...];
          % H är matrisen med tetraederns koordinater
P=H;
clf
hold on
for i=1:size(S,1) % Rita tetraedern
    Si=S(i,:); fill3(P(1,Si),P(2,Si),P(3,Si),'b')
end
hold off
axis equal, axis(ax) % Undvik deformation och använd samma skalor på axlarna
                      % Förstärker tredimensionella känslan något
box on, grid on
pause(0.5) % Paus så vi hinner se tetraedern innan den roteras
```

```
kmax=...; % Antal små rotationer, t.ex. kmax=16; (kmax*v=2*pi, ett varv)
for k=1:kmax % Rotera kmax gånger
    P=A*P;
    clf
    hold on
    for i=1:size(S,1) % Rita tetraedern i nya positionen
        Si=S(i,:); fill3(P(1,Si),P(2,Si),P(3,Si),'b')
    end
    hold off
    axis equal, axis(ax), box on, grid on
    pause(0.1)
end
```

Använd koden ovan och fyll i det som saknas.

#### 4 Molekylmodeller

Det finns flera olika sätt att göra molekylmodeller, t.ex. kul-och-pinnmodell (ball-and-stick), se Atkins och Jones kapitel C.3, sid F24.

Här har vi ritat två exempel på symmetriska molekylstrukturer, den ena tetraedrisk och den andra oktaedrisk. Se figur 3.1, Atkins och Jones kapitel 3.1, sid 95.



Ni har ju redan ritat en tetraeder och här ovan har vi ritat även en oktaeder (på föreläsningen såg ni en ikosaeder.)

Uppgift 5. Rita den tetraedriska strukturen. Till hjälp finns två funktioner klot och stav på studiohemsidan, för att rita ett klot (eller kula) respektive en stav (eller pinne). Det matematiska

innehållet i dessa funktioner kommer vi se på i nästa läsperiod, vi nöjer oss med att använda dem just nu.

Stavarna eller pinnarna kan ni låta vara enfärgade. t.ex. vita (lite enklare än att använda två färger). Kloten eller kulorna ger ni färger efter eget val.

För att titta på strukturen från olika håll kan vi givetvis använda våra rotationer. Men vi gör det lite enklare för oss, det finns ju färdiga redskap i MATLAB. Vi kan ta tag i figuren med musen och snurra runt och vi kan rotera runt en scen genom att använda camorbit på olika sätt

```
>> axis equal vis3d
>> camlight left
>> for i=1:200
      camorbit(5,0)
      drawnow; pause(0.05)
end>> axis equal vis3d
>> h=camlight('left');
>> for i=1:200
      camorbit(5,0)
      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.