

Instruktioner och lösningsförslag till uppgift 2, del i
momentet
T2, Matlabprogrammering, modellering av tvådimensionella
objekt,
ingående i kursen
ET1522 SOM17 Beräknings- och simuleringsteknik (distans)
påbörjat 2016, senaste uppdateringen 12 juni 2017

Huvudförfattare: Anders Hultgren
övriga författare: Frida Gleisner och Matz Lenells

12 juni 2017

Innehåll

1 Inledning	1
1.1 Att tänka på vid inlämning av filer	1
2 Allmänt om uppgift 2	2
3 Rita hus och koordinatsystem	3
4 Matrimultiplikation	4
5 Linjär avbildning	6
5.1 Spegling	6
5.2 Rotation	7
5.3 Translation	8
5.4 Spegling i linjen $x=4$	10
5.5 Rotation runt punkten $(4,0)$	12
A Beskrivning av transformationer i \mathbb{R}	13
A.1 Hur en translation i \mathbb{R} kan beskrivas med hjälp av linjär avbildning i \mathbb{R}^2 . .	14
A.1.1 Kommentarer om beteckningar	16
A.2 Hur en spegling i \mathbb{R} kan beskrivas med hjälp av linjära avbildning i \mathbb{R}^2 . . .	16

1 Inledning

1.1 Att tänka på vid inlämning av filer

Anmärkning 1. När ni lämnar in filer så tillfoga era namn i filnamnet. Nedan finns exempel på hur man kan skriva.

raypolygonAsaÖrn.m
om man heter Åsa Örn och
raypolygonEHakansson.m

om man heter Esmaralda Håkansson.

Matlab är känslig för valet av filnamn, därför bör man undvika tecken såsom å, ä och ö. Vidare ska man inte ha blanktecken i filnamn.

□

Anmärkning 2. Vi har fått in någon fil som inte går att köra. Detta kan bero på att ni har ändrat filnamnet och exempelvis använt ett ”ö” i filnamnet. För att minska risken att vi inte kan köra de filer som ni lämnar in så ber vi er att provköra just de filer som ni lämnar in. Vidare, använd en ny version av Matlab, gärna från 2017 och inte senare än från 2015.

□

2 Allmänt om uppgift 2

Uppgift 2 är ägnad åt modellering och animering av tvådimensionella objekt, en enkel form av datorgrafik. Vi kommer att modellera och animera med användning av samma metoder som används i dataspel som då ofta avbildar en tredimensionell virtuell värld. Då vi nu nöjer oss med två dimensioner blir dimensionen av de matriser vi använder lägre och objekten kan visas direkt på skärmen utan användning av perspektivprojektion.

Modelleringen av objekt i dataspel sker på samma sätt som vi modellerade polygonen i uppgift 1, dvs objektet består av en matris som innehåller ett antal Ortsvektorer till punkter i objektet. Se matrisen `rectangle` i filen `rayrectangle.m`.

Animeringen, dvs objektets rörelser, genereras genom att objektet (Låt oss kalla objektmatrisen O) multipliceras med en animeringsmatris, T . Till exempel kan ett objekt roteras en viss vinkel genom att multipliceras med en matris. Beräkningen för den animeringen blir då $O_{ny} = T \cdot O_{gammal}$. Genom att använda upprepade sådana beräkningar skapas en animering.

Beräkningen $O_{ny} = T \cdot O_{gammal}$ kommer vi att kalla en avbildning, man kan se det som att O_{ny} är en avbildning av O_{gammal} .

Då avbildningen sker genom en matrismultiplikation blir det nya objektet en linjärkombination av det gamla. Vi återkommer till varför detta är en linjär operation.

En anledning till att arbeta med linjära avbildningar är att dynamiska system ofta kan modelleras med hjälp av sådana. Vi kommer att använda denna metod i uppgift 3 och i senare kurser för att modellera dynamiska objekt. Exempelvis kan uttrycket

$$\dot{x}(t) = A \cdot x(t)$$

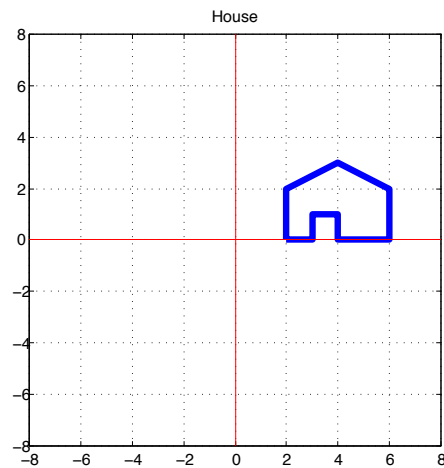
modellera ett kontinuerligt system och

$$x(k+1) = F \cdot x(k).$$

kan modellera ett diskret dynamiskt system. Vi återkommer till detta i uppgift 3.

I kursens andra uppgift ska du arbeta med linjära avbildningar. I dagligt tal syftar ordet avbilda på processen att avbilda, exempelvis måla av något, och resultatet skulle vi kunna kalla en för *avbildning*. I matematiken används begreppet avbildning på ett annorlunda sätt och avser, förutom resultatet, den regel enligt vilken avbildningen sker. När vi med en avbildning menar en regel får vi skillnader mot det vi i dagligt tal kallar för avbildning. Till exempel kan avbildningen ”spegling i y -axeln” genererar olika resultat beroende på hur bilden (originalet) ser ut. Speglar vi en båt blir resultatet en båt, speglar vi ett hus blir resultatet ett hus, men avbildningen, att spegla i y -axeln, är den samma.

Formellt är en avbildning en funktion. En funktion parar ihop ett x -värde med ett y -värde, till exempel funktionen $y = \sin x$, en funktion som låter sig avbildas i ett tvådimensionellt koordinatsystem. I den här uppgiften ska vi till punkter i xy -planet para punkter i xy -planet. För att åskådliggöra detta kan bilden flyttas i planet.



Figur 1: Ett hus placerat i ett koordinatsystem. Hör till avsnitt 3.

Öppna matlabfilen house.p som ligger på itslearning. Filformatet gör att du kan se figuren som programmet genererar men inte programkoden. Din uppgift är att skriva ett program som genererar samma figur. Figuren är en animation och dess startbild ser ut som den i figur 1.

Instruktionerna innehåller följande delar

- 1 Rita hus och koordinatsystem
- 2 Matrimultiplikation
- 3 Linjär avbildning
 - 3.1 Spegling i y -axeln
 - 3.2 Rotation runt origo
 - 3.3 Translation
 - 3.4 Spegling i linjen $x=4$
 - 3.5 Rotation runt punkten $(4,0)$

3 Rita hus och koordinatsystem

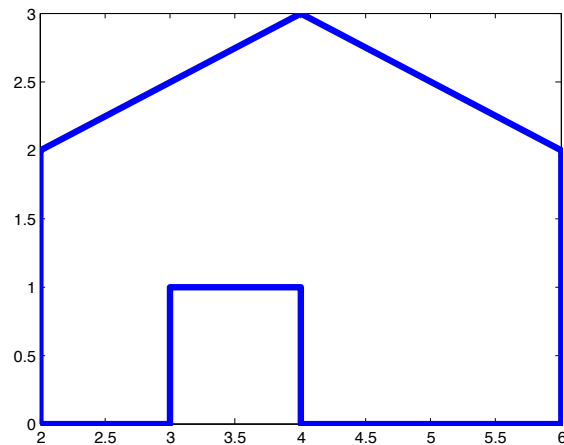
Precis som polygonen är huset en figur som består av linjer som dras mellan olika vektorer. Figur 1 visar ett hus som är placerat i ett koordinatsystem. Vi börjar med att mata in koordinaterna för vektorerna (det går bra att skriva flera vektorer på samma rad):

```
h1=[2;0];h2=[3;0];h3=[3;1];
h4=[4;1];h5=[4;0];h6=[6;0];
h7=[6;2];h8=[4;3];h9=[2;2];
```

På samma sätt som när polygonen ritades skapar vi först en matris av vektorerna där första vektorn även står med sist i matrisen för att binda samman linjen.

```
h=[h1 h2 h3 h4 h5 h6 h7 h8 h9 h1];
```

Sedan ritar vi ut huset och gör linjerna lite bredare. Anges ingen färg för linjen kommer den att bli blå.



Figur 2: Första huset som ritas. Hör till avsnitt 3.

```
figure(gcf)
plot(h(1,:),h(2,:), 'LineWidth', 4)
```

Kompileras programmet bör diagrammet se ut som det i figur 2.

För att anpassa koordinatsystemet, rita axlarna röda, förse diagrammet med en titel samt lägga till ett "grid"(ord på svenska) skriver vi längs ner i koden

```
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
```

Observera att `hold on` och `hold off` bara används runt kommandot `plot`, detta eftersom `plot` annars rensar diagrammet och ritar en ny bild. De andra kommandona modifierar bara det diagram som redan finns. Kommandot `'square'` gör diagrammet kvadratisk och eftersom axlarna har getts samma längd så blir skalstegen för de båda axlarna lika.

Programmet bör nu generera en bild som den i figur 1 och är tänkt att se ut så här:

```
h1=[2;0];h2=[3;0];h3=[3;1];
h4=[4;1];h5=[4;0];h6=[6;0];
h7=[6;2];h8=[4;3];h9=[2;2];

h=[h1 h2 h3 h4 h5 h6 h7 h8 h9 h1];
plot(h(1,:),h(2,:), 'LineWidth', 4)
figure(gcf)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
```

4 Matrismultiplikation

För att förflytta huset i koordinatsystemet multiplicerar vi matrisen som beskriver huset med andra matriser, en matris för varje typ av förflyttning. En kort beskrivning av hur matrismultiplikation går till finns i filen "Inför uppgift 2". Här infogas två sidor från den presentationen, se figurer 3 och 4.

Matrismultiplikation

Med hjälp av matrismultiplikation kan en Ortsvektor flyttas.

I exemplet nedan flyttas Ortsvektorn $\begin{pmatrix} 3 \\ 4 \end{pmatrix}$ till $\begin{pmatrix} 11 \\ 20 \end{pmatrix}$.

$$\begin{pmatrix} 1 & 2 \\ 0 & 5 \end{pmatrix} \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 1 \cdot 3 + 2 \cdot 4 \\ 0 \cdot 3 + 5 \cdot 4 \end{pmatrix} = \begin{pmatrix} 11 \\ 20 \end{pmatrix}$$

Har vi två Ortsvektorer och vill göra samma förflyttning med dem båda kan vektorerna ordnas i en matris. I exemplet nedan görs förflyttningen även av vektorn $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$.

$$\begin{pmatrix} 1 & 2 \\ 0 & 5 \end{pmatrix} \begin{pmatrix} 3 & 1 \\ 4 & 2 \end{pmatrix} = \begin{pmatrix} 11 & 5 \\ 20 & 10 \end{pmatrix}$$

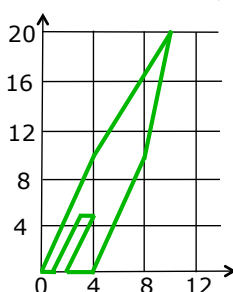
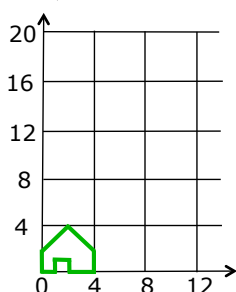
Vi kommer att kalla beräkningen för en avbildning. De nya Ortsvektorerna är avbildningar av de gamla.

Figur 3: Illustration av matrismultiplikation när en vektor avbildas på en annan vektor. Hör till avsnitt 4.

Matrismultiplikation

Låt oss applicera avbildningen $\begin{pmatrix} 1 & 2 \\ 0 & 5 \end{pmatrix}$ på huset, tidigare benämnd h .

$$\begin{pmatrix} 1 & 2 \\ 0 & 5 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 2 & 2 & 4 & 4 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 2 & 4 & 2 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 3 & 4 & 2 & 4 & 8 & 10 & 4 & 0 \\ 0 & 0 & 5 & 5 & 0 & 0 & 10 & 20 & 10 & 0 \end{pmatrix}$$



Huset förändras, det behåller inte sin form.

Figur 4: Matrismultiplikation som används för att avbilda en geometrisk figur på en annan geometrisk figur. Hör till avsnitt 4.

5 Linjär avbildning

Vid beräkningen av avbildningar skiljer man bl a på linjära avbildningar och icke-linjära avbildningar. Linjära avbildningar kan exempelvis vara spegling och rotation. Linjära avbildningar utmärks av att de kan göras med hjälp av matrismultiplikation. På papper kan matrismultiplikation vara en tidskrävande uppgift men i Matlab och andra programvaror görs beräkningarna effektivt. Vill vi simulera en rörelse blir uppgiften enklare om rörelsen kan beskrivas som en linjär avbildning eller sammansättningen av flera linjära avbildningar. I uppgift två ska vi använda de linjära avbildningarna spegling och rotation samt den affina avbildningen translation. Även om translation inte är en linjär avbildning så kan man ändå beskriva avbildningen som en sammansättning av två enkla avbildningar och en linjär avbildning. Mer om detta i avsnitt 5.3.

5.1 Spegling

En spegling som sker i en linje beräknas på olika sätt beroende på var linjen går. Det enklaste fallet, som vi ska göra här, är spegling i någon av axlarna.

I Matlab kan vi samla definitionerna för variabler och matriser överst i programmet, under dessa kommer de sekvenser som utför och ritar förflyttningarna av huset. För ett förslag till ordning av kommandon, se slutet av avsnittet. Vi kallar vi vår speglingsmatris för T1 och skriver

```
T1=[-1 0
    0 1];
```

Definitionen för T1 sker lämpligen högst upp i programmet.

Efter att huset har visats i några sekunder ska det nu flyttas med hjälp av speglingen, vi inför en paus och tilldelar matrisen h ett nytt värde.

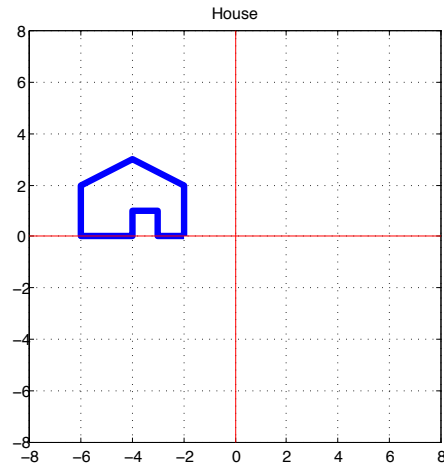
```
s=3;
pause(s)
h=T1*h;
plot(h(1,:),h(2:,:), 'LineWidth', 4)
```

Den första bilden kommer nu att visas i 3 sekunder . Kommandot `plot` som följer efter matrismultiplikationen ritar ett nytt diagram och för att det ska se ut som att huset flyttas får vi ange igen hur diagrammet ska se ut. Programkoden kan nu se ut så här

```
h1=[2;0];h2=[3;0];h3=[3;1];
h4=[4;1];h5=[4;0];h6=[6;0];
h7=[6;2];h8=[4;3];h9=[2;2];

h=[h1 h2 h3 h4 h5 h6 h7 h8 h9 h1];
s=1;
T1=[-1 0
    0 1];

plot(h(1,:),h(2:,:), 'LineWidth', 4)
figure(gcf)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
pause(s)
```



Figur 5: Huset efter spegling i y -axeln. Hör till avsnitt 5.1.

```
%1 Reflection
h=T1*h;
plot(h(1,:),h(2:,:), 'LineWidth', 4)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
```

och ge en slutbild som den i figur 5.

5.2 Rotation

Vi ska nu rotera huset 90° moturs, eller $\frac{\pi}{2}$, runt origo. För rotationen ska vi använda en matris där vinkeln skrivs in som en variabel, på detta sätt kan vi enkelt ändra vinkeln för rotationen.

Den matris som ger avbildningen *rotation 90° moturs runt origo* vrider alla eventuella linjer i xy -planet på samma sätt. Om vi använder vektorerna $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ och $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ får vi följande rotationsmatris som vi kallar för T2:

$$T2 = \begin{pmatrix} \cos \frac{\pi}{2} & -\sin \frac{\pi}{2} \\ \sin \frac{\pi}{2} & \cos \frac{\pi}{2} \end{pmatrix}$$

Vi använder variabeln v2 och definierar matrisen T2

```
v2=pi/2;
T2=[cos(v2) -sin(v2)
    sin(v2)  cos(v2)];
```

Efter speglingen ska vi nu längst ner i programmet lägga till rotationen

```
h=T2*h
```

Programmet kan nu se ut så här

```
h1=[2;0];h2=[3;0];h3=[3;1];
h4=[4;1];h5=[4;0];h6=[6;0];
h7=[6;2];h8=[4;3];h9=[2;2];
```

```

h=[h1 h2 h3 h4 h5 h6 h7 h8 h9 h1];
s=1;
T1=[-1 0
     0 1];

v2=pi/2;
T2=[cos(v2) -sin(v2)
     sin(v2)  cos(v2)];

plot(h(1,:),h(2:,:), 'LineWidth', 4)
figure(gcf)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
pause(s)

%1 Reflection
h=T1*h;
plot(h(1,:),h(2:,:), 'LineWidth', 4)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
pause(s)

%2 Rotation
h=T2*h;
plot(h(1,:),h(2:,:), 'LineWidth', 4)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on

```

Se house.p-filen för att kontrollera att din figur ser ut som den ska.

5.3 Translation

En translation är en förflyttning av urbilden och är en avbildning som inte är linjär. Vi har ovan studerat spegling i en linje som går genom origo samt rotation runt origo. Dessa båda typer av transformationer är linjära. Det visar sig mycket praktiskt att kunna beskriva avbildningar med hjälp av matrismultiplikation och för att använda en sådan beskrivning för en translation tar vi hjälp av en tredje dimension¹. Den tredje dimensionen gör det möjligt att placera huset i planet $z=1$. Vi börjar med att lägga till z -koordinaten 1 i matrisen h .

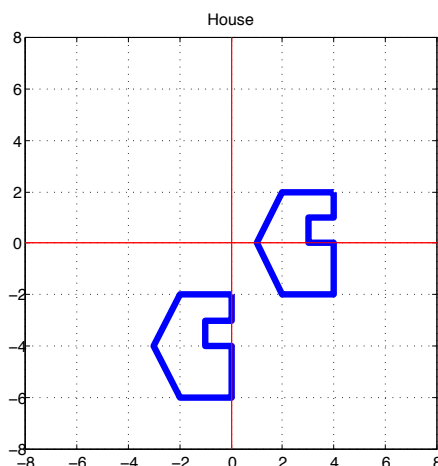
```

h=[h
   ones(1,length(h))];

```

Kommandot tilldelar matrisen h den ursprungliga matrisen h med en extra rad av ettor längst ner, `length(h)` betyder att etterna är lika många som antalet kolumner i matrisen

¹I appendix A studerar vi transformationer i \mathbb{R} . Där studeras speciellt två metoder som motsvarar metoder som beskrivs i detta avsnitt. Beskrivningarna i appendix A är enklare än beskrivningarna här.



Figur 6: Huset nederst i figuren har erhållits efter rotation 90 grader av huset som visas i figur 5. Denna transformation behandlas i avsnitt 5.2. En translation av huset med vektorn $(4, 4)$ ger det övre högra huset. Translationen behandlas i avsnitt 5.2.

h. När detta är gjort behöver även de andra matriserna ändras, för att se hur detta görs se förslaget till programkod sist i detta avsnitt.

Huset ska nu flyttas så att vektorn $\begin{pmatrix} 0 \\ -4 \\ 1 \end{pmatrix}$ blir $\begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix}$, se figur 6. Vi ser att följande matrisoperation ger förflyttningen.

$$\begin{pmatrix} 1 & 0 & 4 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ -4 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 + 0 + 4 \\ 0 - 4 + 4 \\ 0 + 0 + 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix}$$

Oavsett vilken vi väljer av vektorerna som beskriver huset så gör de samma förflyttning, fyra steg i x -led och fyra steg i y -led. Vi definierar matrisen T3

```
T3=[1 0 4
    0 1 4
    0 0 1];
```

och beräknar och ritar ut förflyttningen

```
h=T3*h;
plot(h(1,:),h(2,:), 'LineWidth', 4)
```

Precis som vid rotationen behöver dessa kommandon följas av instruktioner för hur diagrammet ska se ut samt en paus. Programkoden kan nu se ut så här:

```
h1=[2;0];h2=[3;0];h3=[3;1];
h4=[4;1];h5=[4;0];h6=[6;0];
h7=[6;2];h8=[4;3];h9=[2;2];
h=[h1 h2 h3 h4 h5 h6 h7 h8 h9 h1];
h=[h
ones(1,length(h))];
s=3;
T1=[-1 0 0
    0 1 0
```

```

        0 0 1];
v2=pi/2;
T2=[cos(v2) -sin(v2) 0
    sin(v2) cos(v2) 0
    0 0 1];
T3=[1 0 4
    0 1 4
    0 0 1];

plot(h(1,:),h(2,:), 'LineWidth', 4)
figure(gcf)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
pause(s)

%1 Reflection
h=T1*h;
plot(h(1,:),h(2,:), 'LineWidth', 4)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
pause(s)

%2 Rotation
h=T2*h;
plot(h(1,:),h(2,:), 'LineWidth', 4)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
pause(s)

%3 Translation
h=T3*h;
plot(h(1,:),h(2,:), 'LineWidth', 4)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on

```

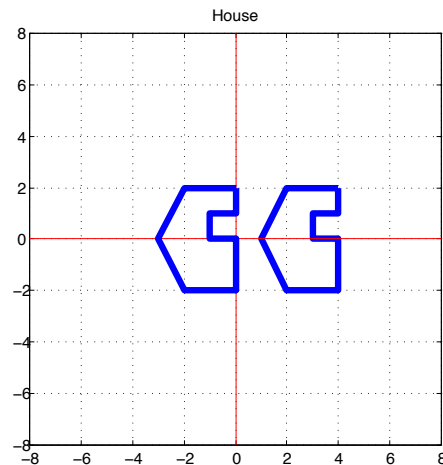
5.4 Spegling i linjen $x=4$

Om vi tänker oss att huset ligger i xy -planet ska det nu reflekteras i *linjen* $x=4^2$.

För att spegla huset i en linje som inte går genom origo flyttar vi huset till origo, reflekterar det och flyttar sedan tillbaka huset. Alltså translation, spegling, translation.

Som huset är placerat i Matlab vill vi att vektorn $\begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix}$ efter translationen ska bli $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$,

²Beroende på hur vi betraktar matriserna i Matlab kan vi påstå att huset inte ligger kvar i xy -planet, vi har flyttat det till planet $z=1$. Speglingen sker då inte i en linje utan i planet $x=4$.



Figur 7: Det högra huset är ett av husen som visas i figur 6. Translation av detta hus till origo ger huset till vänster. Denna translation, T4a, beskrivs i avsnitt 5.4.

se figur 7. Det görs med följande matrismultiplikation

$$\begin{pmatrix} 1 & 0 & -4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 + 0 - 4 \\ 0 + 0 + 0 \\ 0 + 0 + 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Vi definierar matrisen T4a

```
T4a=[1 0 -4
      0 1 0
      0 0 1];
```

För att huset ska göra förflyttningen skriver vi

```
h=T4a*h;
plot(h(1,:),h(2,:), 'LineWidth', 4)
```

Testkör gärna programmet för att se att huset hamnar vid origo.

Nu ska huset reflekteras i y -axeln, en avbildning som vi tidigare gjorde med matrisen T1. Vi för in matrisen T1 före T4a.

```
h=T1*T4a*h;
```

Slutligen ska huset tillbaka igen, istället för att x -värdet ska minska med 4 ska det nu öka med 4, vi kan definiera matrisen T4b

```
T4b=[1 0 4
      0 1 0
      0 0 1];
```

och för in matrisen T4b så att beräkningen för speglingen ser ut så här

```
h=T4b*T1*T4a*h;
```

Jämför ditt resultat med filen house.p. För ett förslag till programkod se slutet på nästa avsnitt.

5.5 Rotation runt punkten (4,0)

Om vi tänker oss att huset ligger i xy -planet ska det nu roteras runt punkten (4,0). På samma sätt som vid speglingen i 5.4 kan detta göras med den rotationsmatris vi redan använt, T2. På samma sätt som innan flyttar vi huset till origo med T4a, gör rotationen och flyttar sedan tillbaka det med T4b. Vi behöver inte definiera någon ny matris utan kan direkt göra beräkningen

```
h=T4b*T2*T4a*h;
```

Rita ut huset, det bör nu sluta där det startade.

Ett förslag på hur koden för programmet kan se ut

```
%% House
%
%% Definition of matrixes and variabels
h1=[2;0];h2=[3;0];h3=[3;1];
h4=[4;1];h5=[4;0];h6=[6;0];
h7=[6;2];h8=[4;3];h9=[2;2];
h=[h1 h2 h3 h4 h5 h6 h7 h8 h9 h1];
h=[h
ones(1,length(h))];
s=3;
T1=[-1 0 0
     0 1 0
     0 0 1];
v2=pi/2;
T2=[cos(v2) -sin(v2) 0
     sin(v2) cos(v2) 0
     0 0 1];
T3=[1 0 4
     0 1 4
     0 0 1];
Ta=[1 0 -4
     0 1 0
     0 0 1];
Tb=[1 0 4
     0 1 0
     0 0 1];

%% Plot of house
plot(h(1,:),h(2:,:), 'LineWidth', 4)
figure(gcf)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
title('House')
grid on
pause(s)

%1 Reflection
h=T1*h;
plot(h(1,:),h(2:,:), 'LineWidth', 4)
axis([-8 8 -8 8], 'square')
hold on, plot([0 0], [-8 8], 'r', [-8 8], [0 0], 'r'), hold off
```

```

title('House')
grid on
pause(s)

%2 Rotation
h=T2*h;
plot(h(1,:),h(2:),'LineWidth', 4)
axis([-8 8 -8 8],'square')
hold on, plot([0 0],[-8 8],'r',[-8 8],[0 0],'r'), hold off
title('House')
grid on
pause(s)

%3 Translation
h=T3*h;
plot(h(1,:),h(2:),'LineWidth', 4)
axis([-8 8 -8 8],'square')
hold on, plot([0 0],[-8 8],'r',[-8 8],[0 0],'r'), hold off
title('House')
grid on
pause(s)

%5 Reflection
h=Tb*T1*Ta*h;
plot(h(1,:),h(2:),'LineWidth', 4)
axis([-8 8 -8 8],'square')
hold on, plot([0 0],[-8 8],'r',[-8 8],[0 0],'r'), hold off
title('House')
grid on
pause(s)

%6 Rotation
h=Tb*T2*Ta*h;
plot(h(1,:),h(2:),'LineWidth', 4)
axis([-8 8 -8 8],'square')
hold on, plot([0 0],[-8 8],'r',[-8 8],[0 0],'r'), hold off
title('House')
grid on

```

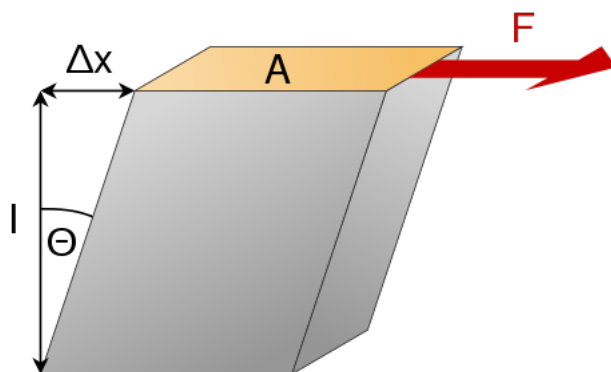
A Beskrivning av transformationer i \mathbb{R}

I detta avsnitt vill vi förklara två metoder. Med den första metoden beskrivs en translation i \mathbb{R} med hjälp av en linjär avbildning i en högre dimension, i detta fall i \mathbb{R}^2 . Den andra metoden används för att beskriva en godtyckligt vald spegling som en sammansättning av en translation, en spegling i origo, dvs $x = 0$, och en ytterligare translation.

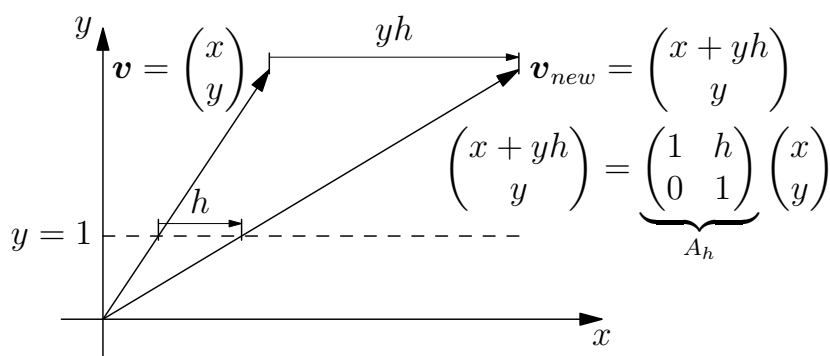
Dessa metoder har sina motsvarigheter i fallet med transformationer i \mathbb{R}^2 där vi tar hjälp av \mathbb{R}^3 för att få en metod som är praktisk att använda. Dock, det är enklare att förklara och beskriva metoderna när transformationerna sker i \mathbb{R} än när de sker i \mathbb{R}^2 .

I fallet med transformationer i \mathbb{R}^2 , fallet som beskrivs i avsnitt 5, behandlas rotationer. Några sådana finns dock inte i \mathbb{R} . Det betyder att vi bara har två typer av operationer som vi vill göra, translationer och speglingar. Detta gör att den praktiska nyttan med beskriva translationer i \mathbb{R} med hjälp av en linjär avbildning i \mathbb{R}^2 (i det närmaste) uteblir.

A.1 Hur en translation i \mathbb{R} kan beskrivas med hjälp av linjär avbildning i \mathbb{R}^2



Figur 8: Från webbsidan: *Skjuvning av ett rätblock; en lika stor skjuvkraft i motsatt riktning på blockets undersida håller kroppen på sin plats.* Author: C.lingg Licensing: *I grant anyone the right to use this work for any purpose, without any conditions, unless such conditions are required by law.*



Begränsa avbildningen till linjen $y = 1$.

Om vi enbart studerar förstakomponenten fås en translation.

$$\underbrace{\begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix}}_{A_h} \begin{pmatrix} x \\ 1 \end{pmatrix} = \begin{pmatrix} x + h \\ 1 \end{pmatrix}$$

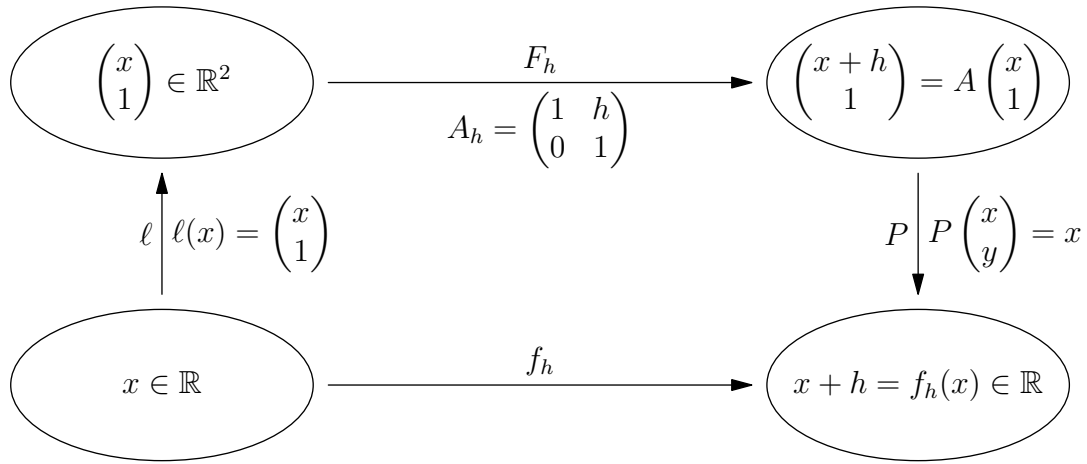
Figur 9: Koordinatsystemet i figurer är valt så att beskrivningen i ekvation (A.1) gäller. Det innebär att x -axeln ligger i skjuvningsens huvudriktning och z -axeln, som inte syns i figuren, ligger så att den inte påverkas av skjuvningsen. y -axeln är som vanligt vinkelrätt mot de två andra axlarna.

Skjuvning är ett bekant begrepp från hållfasthetsläran. På webbsidan

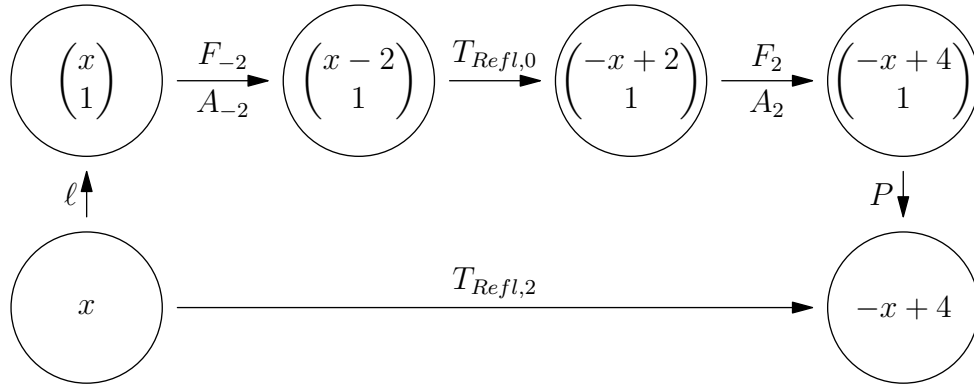
<https://sv.wikipedia.org/wiki/Skjuvning>

hämtar vi följande beskrivning: *Skjuvning, eller skjuvtöjning, är en deformation utan volymändring. Den definieras som vinkeländringen skapad av deformationen.* På webbsidan finns figur 8.

Inför ett koordinatsystem sådant att x -axeln ligger i blockets underkant, se figur 8, och y -axeln sammanfaller med den vertikala linjen i figurens vänstra del. z -axeln är som vanligt vinkelrätt mot de två andra axlarna. Nedan ska vi förklara varför en skjuvning kan



Figur 10: Illustration av hur en translation kan betraktas som en sammansättning $f_h(x) = (P \circ F_h \circ \ell)(x) = P(F_h(\ell(x)))$ av tre funktioner varvid de två funktionerna ℓ och P är mycket enkla, och avbildningen F_h är linjär och beskrivs av 2x2-matrisen A_h .



Figur 11: En spegling i punkten 2 kan ses som en sammansättning av 5 funktioner, varvid funktionerna ℓ och P är mycket enkla och de tre övriga alla är linjära funktioner som kan beskrivas med 2x2-matriser.

beskrivas av den linjära avbildningen

$$F \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x + yh \\ y \\ z \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x + yh \\ y \\ z \end{pmatrix} \quad (\text{A.1})$$

där F betecknar en funktion. I figur 8 betecknar F en kraft.

Figur 9 visar xy -planet och hur punkterna $(x, y) \in \mathbb{R}^2$ förflyttas vid skjuvning. När vi begränsar oss till xy -planet så kan avbildningen som skjuvningen ger beskrivas enligt

$$\begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x + yh \\ y \end{pmatrix} \quad (\text{A.2})$$

Dvs denna avbildning är linjär, den kan ju beskrivas mha en matris på vanligt vis. Låt oss betrakta punkterna $(x, 1)$, dvs de punkter som ligger på den streckade linjen i figur 9. Då ger ekvation (A.2)

$$\begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix} = \begin{pmatrix} x + h \\ 1 \end{pmatrix} \quad (\text{A.3})$$

Om vi bara tittar på förstakomponenterna i denna avbildning så fås att x avbildas på $x + h$, dvs vi får en beskrivning av translation. Genom att göra som figur 10 illustrerar

kan vi genom att använda två enkla operationer, ℓ och P , lyfta vår translation till \mathbb{R}^2 där den blir en skjuvning som är en linjär avbildning och som kan beskrivas med hjälp av en matris.

Funktionen ℓ är ingen linjär avbildning och kan inte beskrivas som en matrismultiplikation. P är däremot en linjär avbildning. Den matris som representerar avbildningen P ges av

$$P \begin{pmatrix} x \\ y \end{pmatrix} = x \quad (1 \ 0) \begin{pmatrix} x \\ y \end{pmatrix} = x \quad [P] = \begin{pmatrix} 1 & 0 \end{pmatrix} \quad (\text{A.4})$$

A.1.1 Kommentar om beteckningar

Beteckningarna för en linjär avbildning och den matris som representerar avbildningen kan vara svåra att hålla isär. Matrisen som representerar avbildningen beror på val av vilka koordinatsystem som används. Ibland används flera olika koordinatsystem samtidigt i en framställning. När då matrisrepresentation A anges för en linjär avbildning F så måste man också ange vilka koordinatsystem som används. I detta appendix har vi använt en beteckning för en linjär avbildning och en annan för den matris som representerar den linjära avbildningen. Exempelvis används i figur 10 beteckningen F_h för en linjär avbildning och beteckningen A_h för den matris som representerar F_h och beteckningen $[P]$ för den matris som representerar avbildningen P .

Om man använder olika beteckningar för avbildningar och de matriser som representerar avbildningarna ökar antalet beteckningar och en text kan bli svår att läsa. Om sammanhanget gör det klart vilka koordinatsystem som används så kan man överväga om man ska använda samma beteckning för en avbildning som för den matris som representerar avbildningen.

A.2 Hur en spegling i \mathbb{R} kan beskrivas med hjälp av linjära avbildning i \mathbb{R}^2

En spegling i en punkt utanför origo är inte en linjär avbildning. Figur 11 visar hur en spegling i punkten $x = 2$ kan ses som en sammansättning av 5 avbildningar där två av avbildningarna, ℓ och P är mycket enkla, och de övriga tre är linjära och därför kan beskrivas med hjälp av matriser. Nedan finns matrisbeskrivningarna för de tre linjära avbildningarna.

$$A_{-2} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 & -2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x - 2 \\ y \end{pmatrix} \quad (\text{A.5})$$

$$T_{Ref\ell,0} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -x \\ y \end{pmatrix} \quad (\text{A.6})$$

$$A_2 \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x + 2 \\ y \end{pmatrix} \quad (\text{A.7})$$