EEE 178 Homework 8

Curtis Muntz

```
% Pre processing
clear all, close all, clc;
frameK=[1,2,5;5,4,6];
frameK1=[5,9,9;5,7,1];

j1=((1/4)*(frameK(1,2)+frameK(2,2)+frameK1(1,2)
    +frameK1(2,2)));
j=((1/4)*(frameK(1,1)+frameK(2,1)+frameK1(1,1)+
    frameK1(2,1)));
Ix(1)=j1-j;

i1=((1/4)*(frameK(2,1)+frameK(2,2)+frameK1(2,1)
    +frameK1(2,2)));
i=((1/4)*(frameK(1,1)+frameK(1,2)+frameK1(1,1)+
    frameK1(1,2)));
Iy(1)=i1-i;

k1=((1/4)*(frameK1(1,1)+frameK1(1,2)+frameK1
    (2,1)+frameK1(2,2)));
k=((1/4)*(frameK(1,1)+frameK(1,2)+frameK(2,1)+
    frameK(2,2)));
It(1)=k1-k;

j1=((1/4)*(frameK(1,3)+frameK(2,3)+frameK1(1,3)
    +frameK1(2,3)));
j=((1/4)*(frameK(1,2)+frameK(2,2)+frameK1(1,2)+
    frameK1(2,2)));
Ix(2)=j1-j

i1=((1/4)*(frameK(2,2)+frameK(2,3)+frameK1(2,2)
    +frameK1(2,3)));
i=((1/4)*(frameK(1,2)+frameK(1,3)+frameK1(1,2)+
    frameK1(1,3)));
Iy(2)=i1-i

k1=((1/4)*(frameK1(1,2)+frameK1(1,3)+frameK1
    (2,2)+frameK1(2,3)));
k=((1/4)*(frameK(1,2)+frameK(1,3)+frameK(2,2)+
    frameK(2,3)));
It(2)=k1-k

clear i1 i k1 k j1 j
```

```
Ix =

    1.5000   -0.2500


Iy =

    1.0000   -1.7500


It =

    3.5000    2.2500
```

A. Part 1

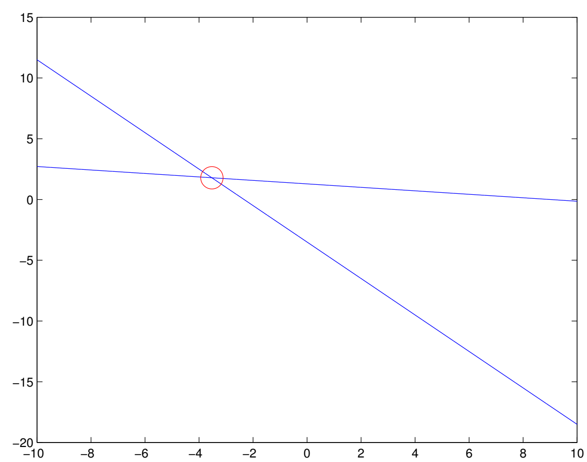
The constraint equations were calculated in the ”pre-processing” section and will be used in the remaining sections. Solving graphically means plotting the two functions and finding their intersection.

```
x=-10:.001:10;
constraint1=((-Ix(1)*x-It(1))/Iy(1));
constraint2=((-Ix(2)*x-It(2))/Iy(2));
plot(x,constraint1)
hold on
plot(x,constraint2)


idx = find(constraint1 - constraint2 < eps, 1);
    %// Index of coordinate in array
Vx = x(idx);
Vy = constraint1(idx);
plot(Vx, Vy, 'ro', 'MarkerSize', 18)

disp('Graphical Answers:');
disp(['Vx = ' num2str(Vx)])
disp(['Vy = ' num2str(Vy)])
disp('');
```

```
Graphical Answers:
Vx = -3.526
Vy = 1.789
```



B. Part 2: Horn and Schunk

The Horn and Shunk method is the recursive function that iteratively attempts to solve for the answer

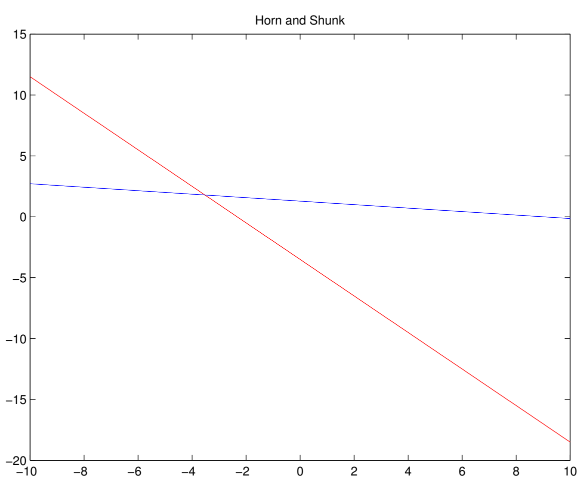
```
Lambda=1;
vx=[0;0];
vy=[0;0];
for k = 1:99
    X1=mean(vx);
```

```
    Y1=mean(vy);
    for i=1:2
        P(i)=Ix(i)*X1+Iy(i)*Y1+It(i);
        D(i)=Lambda^2+Ix(i).^2+Iy(i).^2;
        vx(i)=X1-Ix(i)*P(i)/D(i);
        vy(i)=Y1-Iy(i)*P(i)/D(i);
    end
end

disp('Horn and Shunk method answer:');
disp(['Vx = ' num2str(vx(1))])
disp(['Vy = ' num2str(vy(1))])
disp('')

%plotlines
figure('name', 'Horn and Shunk')
vx=-10:0.1:10;
vy=(-It(1)-Ix(1)*vx)/Iy(1);
plot(vx,vy,'r');
hold on
vy=(-It(2)-Ix(2)*vx)/Iy(2);
plot(vx,vy,'b');
title('Horn and Shunk')
```

```
Horn and Shunk method answer:
Vx = -3.5263
Vy = 1.7895
```



C. *Part 3: Kanade and Lucas*

The Lucas and Kanade method was a very fast and efficient matrix solution for the system of constraint equations.

```
M=[Ix(1) Iy(1);Ix(2) Iy(2)];
b=-[It(1);It(2)];
disp('Lucas and Kanade method answer:');
v=inv(M'*M)*M'*b;
disp(['Vx = ' num2str(v(1))])
disp(['Vy = ' num2str(v(2))])
```

```
Lucas and Kanade method answer:
Vx = -3.5263
Vy = 1.7895
```

Comparison of results

All three methods produced the identical solution, $Vx = -3.526$ and $Vy = 1.7895$. Comparing their implementations, the Lucas and Kanade method is the most useful, considering it runs faster than the Horn and Schunk, and doesn't require plots in order to solve.