Table of Contents


```
%Chan
function Final()
```

Initial

```
l1 = 0; %mm
l2 = 0.05; %mm
l3 = 0.1; %mm
beta = 4/6;
stride = 0.03;
v = 0.01;
robotHeight = 0.1;
robotLength = .220;%mm
robotWidth = .25;
```

Question 1

```
disp('The average velocity with respect to body is')
uBody = (beta/(1-beta))*v
```

```
The average velocity with respect to body is

uBody =

         0.02
```

Question 2

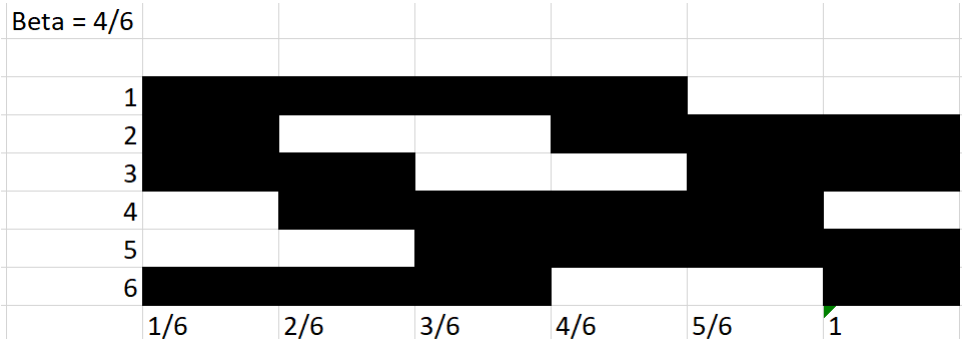
```
disp('The average velocity with respect to ground is')
uGround = v/(1-beta)
```

```
The average velocity with respect to ground is
```

```
uGround =

    0.03
```

Question 3

There are always at least 4 legs in contact with the ground at all points in the walking period.



Question 4

```
p(1)=0; % Kinematic phase of leg 1
p(2)=p(1)+1/2; % Kinematic phase of leg 2
p(3)=p(1)+beta; % Kinematic phase of leg 3
p(4)=p(2)+beta; % Kinematic phase of leg 4
p(5)=p(3)+beta; % Kinematic phase of leg 5
p(6)=p(4)+beta; % Kinematic phase of leg 6

j=1;
for j=1:6
    for i=1:6
        if p(i)>=1
            p(i)=p(i)-1;
        end
    end
    j=j+1;
end
disp('The relative kinematic phase for each leg is as follows')
p
```

```
The relative kinematic phase for each leg is as follows

p =

            0          0.5      0.66667      0.16667      0.33333
 0.83333
```

Question 5

```
disp('The length of one cycle time period in seconds is')
```

```
period = stride/v
```

```
The length of one cycle time period in seconds is

period =

     3
```

Question 6

```
disp('Transfer time is just time in transfer phase * period')
transferPhase = 1-beta;
transferTime = transferPhase * period
```

```
Transfer time is just time in transfer phase * period

transferTime =

     1
```

Foot Planning

```
time = linspace(0,transferTime,6);
% time = linspace(0,period,6);
% t1 is lift off, t5 is touch down
dt = time(2)-time(1);
xdotmaxf_g=2*(time(6)-time(1))*uGround/((time(5)-time(2))+(time(4)-
time(3))); %maximum leg transferring speed.
% this is the average velocity divided by the surface are under the
 velocity graph.

zdotmaxf_g=xdotmaxf_g; % it is arbitrary.

L = stride;
%hard coded foot to ground
trajectoryX = [-L/2,-L/2,-L/4,L/4,L/2,L/2];
trajectoryY = [0,0,0,0,0,0];
trajectoryZ =
 [0,robotHeight/2,robotHeight,robotHeight,robotHeight/2,0];

% For all 6 legs
for i = 1:6
    legXtraj(i,:) = trajectoryX;
end

for i=1:6
    legZtraj(i,1)=0;
    legZtraj(i,2)=legZtraj(i,1)+dt*zdotmaxf_g/2;
    legZtraj(i,3)=legZtraj(i,2)+dt*zdotmaxf_g/2;
    legZtraj(i,4)=legZtraj(i,3)+0;
    legZtraj(i,5)=legZtraj(i,4)-dt*zdotmaxf_g/2;
```

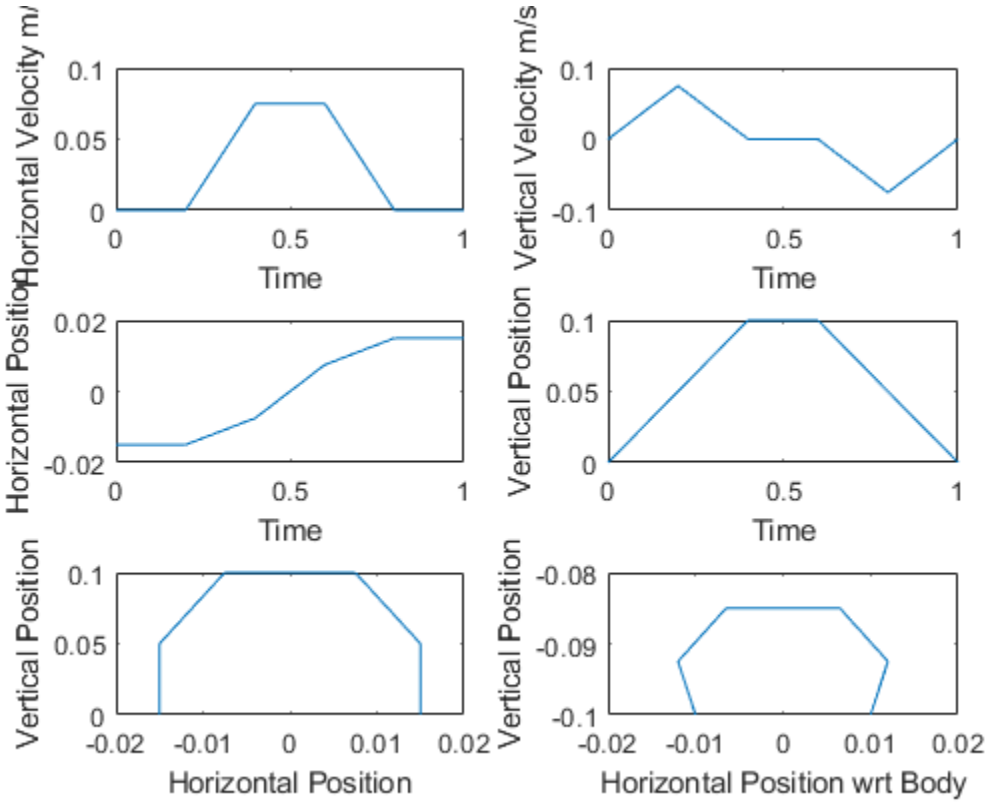
```
    legZtraj(i,6)=legZtraj(i,5)-dt*zdotmaxf_g/2;
end

% velX = [0,0,uGround,uGround,0,0];
% velY = [0,0,0,0,0,0];
% velZ = [0,uGround,0,0,-uGround,0];
velX = [0,0,xdotmaxf_g,xdotmaxf_g,0,0];
velY = [0,0,0,0,0,0];
velZ = [0,xdotmaxf_g,0,0,-xdotmaxf_g,0];



D=l1+l2; %(m) Distance between the foot tip and hip joint from top
 view

% This is the home position for the leg angles wrt body
alphaH=[-30*pi/180, 30*pi/180, -90*pi/180, 90*pi/180, -150*pi/180,
 150*pi/180];

% This sets up body to ground coordinates at t = 0
xb_g(1,1)=-((1-beta)/2)*L-D*cos(alphaH(1)); % frame b is a frame
 attached to the hip joint whose
xb_g(2,1)=-((1-beta)/2)*L-D*cos(alphaH(2));
xb_g(3,1)=-((1-beta)/2)*L-D*cos(alphaH(3)); %This should be zero, but
 for completion sake we keep it in.
xb_g(4,1)=-((1-beta)/2)*L-D*cos(alphaH(4)); %Same for this
xb_g(5,1)=-((1-beta)/2)*L-D*cos(alphaH(5));
xb_g(6,1)=-((1-beta)/2)*L-D*cos(alphaH(6));

% xb_g(1,1)=-(1-beta)*L-D*cos(alphaH(1)); % frame b is a frame
 attached to the hip joint whose
% xb_g(2,1)=-(1-beta)*L-D*cos(alphaH(2));
% xb_g(3,1)=-(1-beta)*L-D*cos(alphaH(3)); %This should be zero, but
 for completion sake we keep it in.
% xb_g(4,1)=-(1-beta)*L-D*cos(alphaH(4)); %Same for this
% xb_g(5,1)=-(1-beta)*L-D*cos(alphaH(5));
% xb_g(6,1)=-(1-beta)*L-D*cos(alphaH(6));


%Since robot height should not change, height is always h.
h = robotHeight;
zb_g(1,1)=h;
zb_g(2,1)=h;
zb_g(3,1)=h;
zb_g(4,1)=h;
zb_g(5,1)=h;
zb_g(6,1)=h;


for i=1:6
    %     This fills in the rest of the body to ground positons for
 the rest of
    %     the time steps.
    for t=1:5 % 5 is because of time dividing to 5 equal extents.
```

```
        xb_g(i,t+1)=xb_g(i,t)+v*dt;
        % Since robot is moving forward at rate of 0.1 m/s, it is just
 v times the timestep difference
        zb_g(i,t+1)=zb_g(i,t);
        % Z does not change, so for all legs it is the same.
    end
    %     Now we calculate the foot to body, which is foot to ground
    %     coordinate minus the body to ground coordinate
    for t=1:6
        xf_b(i,t)=legXtraj(i,t)-xb_g(i,t);
        zf_b(i,t)=legZtraj(i,t)-zb_g(i,t);
%         zf_b(i,t)=legZtraj(i,t)+zb_g(i,t);
    end
end
% While x and z are changing, y doesn't since there is no rotaion,
 thus, y
% is constant across all
yf_b=[D*sin(30*pi/180), -D*sin(30*pi/180), D,  -D,  D*sin(30*pi/180),
 -D*sin(30*pi/180)];

for i=1:6 % this is because of 6 legs
    for j=1:6 % this is because we have divided out T to 6 dt.
        %     Now we calculate the foot to hip positons for each leg
 as it goes
        %     through the time steps.
        %     Since it is a rotation about the z axis, we use the z
 rotation matrix
        %     and multiply it against the xyz coordinates of the
 robot.
        xf_H(i,j)=[cos(alphaH(i)),-
sin(alphaH(i)),0]*[xf_b(i,j);yf_b(i);zf_b(i,j)];

 yf_H(i,j)=[sin(alphaH(i)),cos(alphaH(i)),0]*[xf_b(i,j);yf_b(i);zf_b(i,j)];
        zf_H(i,j)=zf_b(i,j);
    end
end
```

Plotting Code

```
figure(1)
hold on
subplot(3,2,1)
plot(time,velX)
ylabel('Horizontal Velocity m/s')
xlabel('Time')
subplot(3,2,2)
plot(time,velZ)
ylabel('Vertical Velocity m/s')
xlabel('Time')
subplot(3,2,3)
plot(time,trajectoryX)
ylabel('Horizontal Position')
xlabel('Time')
```

```
subplot(3,2,4)
plot(time,trajectoryZ)
ylabel('Vertical Position')
xlabel('Time')
subplot(3,2,5)
plot(trajectoryX,trajectoryZ)
ylabel('Vertical Position')
xlabel('Horizontal Position')
subplot(3,2,6)
plot(xf_b(4,:),zf_b(4,:))
ylabel('Vertical Position')
xlabel('Horizontal Position wrt Body')
hold off;
```



Question 7

```
%To calculate the positions of the legs throughout time, we just need
 to
%compute the inverse kinematics for each leg. Since we know the
 positons of
%the feet with respect to the hip, this is pretty trivial as now we
 can
%just assume the leg as a serial manipulator.

% Taken directly from lecture 7, slide 125
% This code was also shown in example code
```

```
for i=1:6 % for all 6 legs
    for j=1:6 % time discrete
        Theta1(i,j)=(atan(yf_H(i,j)/xf_H(i,j)));
        %                 Theta1(i,j) = atan2(yf_H(i,j),xf_H(i,j));
        %         THIS CHANGES IT TO DEGREES
        l(i,j)=sqrt(yf_H(i,j)^2+xf_H(i,j)^2);
        d(i,j)=sqrt(zf_H(i,j)^2+(l(i,j)-l1)^2);
        Theta2(i,j)=acos((l2^2+d(i,j)^2-l3^2)/(2*l2*d(i,j)))-
atan(abs(zf_H(i,j))/(l(i,j)-l1));
        %                 Theta2(i,j)=acos((l2^2+d(i,j)^2-l3^2)/
(2*l2*d(i,j)))-atan2(abs(zf_H(i,j)),(l(i,j)-l1));
        Theta3(i,j)=pi-(acos((l2^2+l3^2-d(i,j)^2)/(2*l2*l3)));
    end
end
disp('Joint Positions are as follows:')
T1=Theta1*180/pi
T2=-Theta2*180/pi
T3=Theta3*180/pi
```

```
Joint Positions are as follows:

T1 =

       6.8964        8.614       4.1892      -3.3436      -5.6737
 -4.8719
      -6.8964       -8.614      -4.1892       3.3436       5.6737
  4.8719
       11.31       13.496       7.4069      -7.4069      -13.496
  -11.31
       -11.31      -13.496      -7.4069       7.4069       13.496
   11.31
       4.8719       5.6737       3.3436      -4.1892       -8.614
 -6.8964
      -4.8719      -5.6737      -3.3436       4.1892        8.614
  6.8964


T2 =

      0.40093      -8.1096      -17.395       -17.08       -7.903
 0.45214
      0.40093      -8.1096      -17.395       -17.08       -7.903
 0.45214
    0.0056179      -8.6041      -17.413      -17.413      -8.6041
 0.0056179
    0.0056179      -8.6041      -17.413      -17.413      -8.6041
 0.0056179
      0.45214       -7.903       -17.08      -17.395      -8.1096
 0.40093
      0.45214       -7.903       -17.08      -17.395      -8.1096
 0.40093
```

```
T3 =

      94.393       103.53       109.24       102.53       91.493
  84.456
      94.393       103.53       109.24       102.53       91.493
  84.456
      89.427       97.468       105.86       105.86       97.468
  89.427
      89.427       97.468       105.86       105.86       97.468
  89.427
      84.456       91.493       102.53       109.24       103.53
  94.393
      84.456       91.493       102.53       109.24       103.53
  94.393
```

Question 8

Now that we have the joint positions, we now need to find the joint velocities. These were covered in lecture as well as shown in example code.

```
xdotb_g=v;
zdotb_g=0;
ydotf_b=0;
for t=1:6
    xdotf_b(t)=velX(t)-xdotb_g;
    zdotf_b(t)=velZ(t)-zdotb_g;
end
for i=1:6
    for j=1:6
        xdotf_H(i,j)=[cos(alphaH(i)),-
sin(alphaH(i)),0]*[xdotf_b(j);ydotf_b;zdotf_b(j)];

 ydotf_H(i,j)=[sin(alphaH(i)),cos(alphaH(i)),0]*[xdotf_b(j);ydotf_b;zdotf_b(j)];
        zdotf_H(i,j)=zdotf_b(j);
    end
end
for j=1:6 % time
    for i=1:6 % leg number
        theta1=Theta1(i,j);
        theta2=Theta2(i,j);
        theta3=Theta3(i,j);

        %         This builds the jacobian to take the inverse of
        %         This is directly from slides
        J(1,1)=-(-sin(theta1)*sin(theta2)*cos(theta3)-
sin(theta1)*cos(theta2)*sin(theta3))*l3-sin(theta1)*l2*cos(theta2)-
l1*sin(theta1);
        J(1,2)=-(-
cos(theta1)*sin(theta2)*sin(theta3)+cos(theta1)*cos(theta2)*cos(theta3))*l3-
cos(theta1)*l2*sin(theta2);
        J(1,3)=(cos(theta1)*sin(theta2)*sin(theta3)-
cos(theta1)*cos(theta2)*cos(theta3))*l3;
```

```
        J(2,1)=-
(cos(theta1)*cos(theta2)*sin(theta3)+cos(theta1)*sin(theta2)*cos(theta3))*l3+cos(t
        J(2,2)=-(-
sin(theta1)*sin(theta2)*sin(theta3)+sin(theta1)*cos(theta2)*cos(theta3))*l3-
sin(theta1)*l2*sin(theta2);
        J(2,3)=-(-
sin(theta1)*sin(theta2)*sin(theta3)+sin(theta1)*cos(theta2)*cos(theta3))*l3;
        J(3,1)=0;
        J(3,2)=-(-cos(theta2)*sin(theta3)-sin(theta2)*cos(theta3))*l3-
l2*cos(theta2);
        J(3,3)=-(-cos(theta2)*sin(theta3)-sin(theta2)*cos(theta3))*l3;

 Thetadot(i,j,:)=inv(J)*[xdotf_H(i,j);ydotf_H(i,j);zdotf_H(i,j)];
    end
end
disp('Joint Velocities are as follows:')
Thetadot
```

```
Joint Velocities are as follows:

Thetadot(:,:,1) =

     -0.12066     -0.14362       1.1228      0.74497     -0.08384
 -0.085865
      0.12066      0.14362      -1.1228     -0.74497      0.08384
  0.085865
     -0.19614     -0.20843       1.7956       1.7956     -0.20843
 -0.19614
      0.19614      0.20843      -1.7956      -1.7956      0.20843
  0.19614
    -0.085865     -0.08384      0.74497       1.1228     -0.14362
 -0.12066
     0.085865      0.08384     -0.74497      -1.1228      0.14362
  0.12066


Thetadot(:,:,2) =

      -2.0829      -2.9856       2.6182       4.6551       2.4982
   1.8642
      -2.0829      -2.9856       2.6182       4.6551       2.4982
   1.8642
      -3.9221        -2.85     -0.51273      0.51273         2.85
   3.9221
      -3.9221        -2.85     -0.51273      0.51273         2.85
   3.9221
      -1.8642      -2.4982      -4.6551      -2.6182       2.9856
   2.0829
      -1.8642      -2.4982      -4.6551      -2.6182       2.9856
   2.0829


Thetadot(:,:,3) =
```

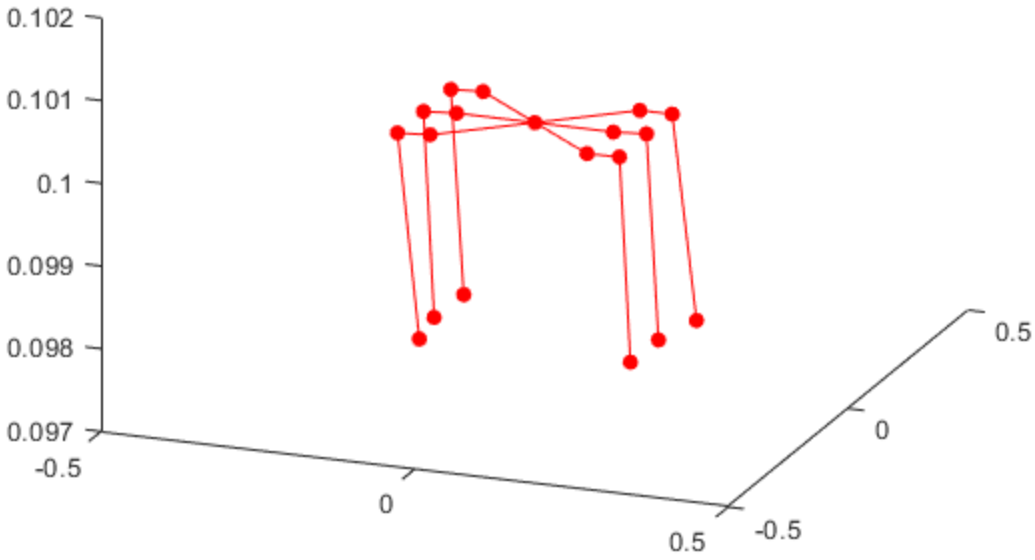
```
      1.0389       2.2026      -1.0614       -2.096      -2.0043
 -0.92698
      1.0389       2.2026      -1.0614       -2.096      -2.0043
 -0.92698
       1.961       2.1642      0.22015      -0.22015      -2.1642
  -1.961
       1.961       2.1642      0.22015      -0.22015      -2.1642
  -1.961
     0.92698       2.0043        2.096        1.0614      -2.2026
 -1.0389
     0.92698       2.0043        2.096        1.0614      -2.2026
 -1.0389
```

Simulation

```
figure(2)

% Sets up legs in home position
% hold on

for i=1:1 % time step
    clf
    axis([-.5 .5 -.5 .5 0.097 .102])
    hold on
    for j=1:6 % leg number
        view([21 27]);
%         view([1 81]);
        % view([-106 17]);
        %         H =
 plotarm(Theta1(i,j),Theta2(i,j),Theta3(i,j),l1,l2,l3,j,robotHeight,robotLength,ro
        [H,staticLegPlot{j}] =
 plotarm(T1(j,i),T2(j,i),T3(j,i),l1,l2,l3,j,robotHeight,robotLength,robotWidth);
    end
    drawnow;
end

% We want all possible time instances between 0 and 3
cycleTime = linspace(0,period,31);
cycleTime = single(cycleTime);
% We want to have the legs only move it is time for it to enter
 transfer
% phase at that point in the period.

% We have fractions of when transfer phase ends/support phase starts.
% Therefore all we need to do to find where transfer phase starts, is
 add
% beta to the support phase.
transferStart = p+beta;

j=1;
for j=1:6
    for i=1:6
```

```
        if transferStart(i)>=1
            transferStart(i)=transferStart(i)-1;
        end
    end
    j=j+1;
end

% To find out what time in period to start transfer phase, just
 multiply
% the fractions by the period.
startPeriod = transferStart*period;
startPeriod = single(startPeriod);
% If the cycle time equals the start time, have that leg go through
 the
% motion
format shortg
for k=1:numel(cycleTime)
    if cycleTime(k) == startPeriod(1)
        delete(staticLegPlot{1})
        transferSim(1,T1, T2,
 T3,robotHeight,robotLength,robotWidth,l1,l2,l3);
    elseif cycleTime(k) == startPeriod(2)
        delete(staticLegPlot{2})
        transferSim(2,T1, T2,
 T3,robotHeight,robotLength,robotWidth,l1,l2,l3);
    elseif cycleTime(k) == startPeriod(3)
        delete(staticLegPlot{3})
        transferSim(3,T1, T2,
 T3,robotHeight,robotLength,robotWidth,l1,l2,l3);
    elseif cycleTime(k) == startPeriod(4)
        delete(staticLegPlot{4})
        transferSim(4,T1, T2,
 T3,robotHeight,robotLength,robotWidth,l1,l2,l3);
    elseif cycleTime(k) == startPeriod(5)
        delete(staticLegPlot{5})
        transferSim(5,T1, T2,
 T3,robotHeight,robotLength,robotWidth,l1,l2,l3);
    elseif cycleTime(k) == startPeriod(6)
        delete(staticLegPlot{6})
        transferSim(6,T1, T2,
 T3,robotHeight,robotLength,robotWidth,l1,l2,l3);
    end
    %     M(k) = getframe(gcf);
end
hold off
disp('done')

% video = VideoWriter('hexapodWalker.avi','Uncompressed AVI');
% open(video)
% writeVideo(video,M)
% close(video)
```

done



Question 10

If we only knew that the legs 1,2, and 5 were on the ground, the robot would be dynamically stable as due to the definition of the Foot Force Stability Margin, there are at least 3 non-collinear legs pushing off of the ground at that exact moment. I believe that the robot would also be statically stable due to the fact that the CoG of the robot should lie within the support polygon created by legs 1,2,and 5.

```
end

function transferSim(legNumber, Theta1, Theta2,
 Theta3,robotHeight,robotLength,robotWidth,l1,l2,l3)
% This is the simulation for a transfer phase
for i=1:6 % time step
    %     clf;
    %     hold on
    j=legNumber; % leg number
    view([21 27]);
%     view([1 81]);
%     view([43 15])
    %         H =
 plotarm(Theta1(i,j),Theta2(i,j),Theta3(i,j),l1,l2,l3,j,robotHeight,robotLength,ro
    [H,legPlot] =
 plotarm(Theta1(j,i),Theta2(j,i),Theta3(j,i),l1,l2,l3,j,robotHeight,robotLength,ro
    drawnow;
    if i<6
        delete(legPlot)
```

```
        end

    end
end
```

Published with MATLAB® R2017b