## 1)

```
r0_pos = [4 0]
r0_grad = [-10 2]
r1_pos = [-.625 .125]
r1_r0_dis = [3.375 .125]
r1_grad = [-8.625 1.125]
r2_pos = [2.7281 .2094]
r2_r1_dis = [-.6469 .0844]
```

### 2)

```
%Find angle of the Gradient and subtract initial angle of the robot (90deg)
theta = atan2(r0_grad(2),r0_grad(1))-pi/2
```

### 3)

```
distance = 6.7453
mps = .1
fps = 3.281*mps;
time = distance/fps;
```

### 4)

```
% Check
```

### 5)

### 8)

```
g = [0; 0; 1];
Rx = @(fi) [1 0 0; 0 cos(fi) sin(fi); 0 sin(fi) cos(fi)];
Ry = @(th) [cos(th) 0 -sin(th); 0 1 0; sin(th) 0 cos(th)];
Rz = @(ps) [cos(ps) sin(ps) 0; -sin(ps) cos(ps) 0; 0 0 1];
%a)
a8 = Rx(pi/6)*g
%b)
b8 = Ry(pi/4)*g
%c)
c8 = Rz(pi/12)*g
```

### 9)

```
%The yaw is rotating the orientation of the phone while keeping the
%bottom aligned with the gravity vector.  Since we are rotating around the
%gravity vector, it doesnt change the direction of gravity relative to the
```

```
%phone.
```

**10)**

```
%Only 2 angles are needed to represent the orientation of the gravity
%vector.
```

**11)**

```
%You need 3 angles to represent the orientation of the NEATO
```

Neato Discrete Code

```
clf;
unitConv = 3.281; %meters to feet
pub = rospublisher('/raw_vel');
sub = rossubscriber('/encoders');

d=.25*unitConv;
Gradient = @(x) [(x(2)-2*x(1)-2); (x(1)-2*x(2)-2)];
% set initial vals
% Define the initial step-size
lambda = 1/16;
% Define the step-size multiplier
delta = 1.2;
time = 0;
stoptime = 80;
stop = 0;
movmult = 1;
turndiv = 5;
maxturnspd = .05;
maxlinspd = .2;
%desiredDistance = 0;

strtmsg = rosmessage(pub);
stopmsg = rosmessage(pub);

stopmsg.Data = [0 , 0];

tic

%send(pub, strtmsg)
%wheeldata = receive(sub);
wheeldata = receive(sub);
data_old = wheeldata.Data * unitConv;
hold on;
x = [4; 1];
ang = pi/2;
while norm(Gradient(x)) > 0.3
    desTrav = lambda.*Gradient(x);
    desAng = mod(atan2(desTrav(2), desTrav(1)),2*pi);
    desDist = norm(desTrav);
    lambda = lambda.*delta;
```

```
    angDiff = ang - desAng;
    while abs(angDiff) > .03
%         if abs(angDiff) > pi
%             turndir = -1;
%         else
%             turndir = 1;
%         end
        wheeldata = receive(sub);
        data_new = wheeldata.Data * unitConv;
        dpl = data_new(1) - data_old(1);
        dpr = data_new(2) - data_old(2);
        dp = (dpl + dpr)/2;
        dang=(dpr-dpl)/d;
        dx=dp*cos(ang);
        dy=dp*sin(ang);
        ang = mod(ang+dang, 2*pi);
        x = x + [dx;dy];
        turnspd = angDiff/turndiv;
        if turnspd > maxturnspd
            strtmsg.Data = [1, -1] * maxturnspd ;
        elseif turnspd < -maxturnspd
            strtmsg.Data = [1, -1] * -maxturnspd;
        else
            strtmsg.Data = [1, -1] * turnspd;
        end
        send(pub, strtmsg)
        if (toc>stoptime || norm(Gradient(x)) < 0.01);
            send(pub, stopmsg)
            stop = 1
            break
        end
        plot(x(1), x(2), 'bo')
        data_old = data_new;
        angDiff = ang - desAng;
        if angDiff > pi
            angDiff = angDiff - 2*pi;
        elseif angDiff < -pi
            angDiff = angDiff + 2*pi;
        end
        angDiff
    end
    if stop
        break
    end
    cur_step_move = 0;
    while cur_step_move < desDist
        wheeldata = receive(sub);
        data_new = wheeldata.Data * unitConv;
        dpl = data_new(1) - data_old(1);
        dpr = data_new(2) - data_old(2);
        dp = (dpl + dpr)/2;
        cur_step_move = cur_step_move + dp;
        dang=(dpr-dpl)/d;
        dx=dp*cos(ang);
        dy=dp*sin(ang);
        ang = mod(ang+dang, 2*pi);
        x = x + [dx;dy];
        linspd = desDist*movmult;
        if linspd > maxlinspd
            strtmsg.Data = [1, 1] * maxlinspd;
            disp('wow')
```

```
        else
            strtmsg.Data = [1, 1] * linspd;
            disp('MEME')
        end
        send(pub, strtmsg)
        if (toc>stoptime || norm(Gradient(x)) < 0.01)
            send(pub, stopmsg)
            stop = 1
            break
        end
        plot(x(1), x(2), 'bo')
        data_old = data_new;
    end
    if stop
        break
    end
end
send(pub, stopmsg)
```

Neato Continuous Code

```
clf;
unitConv = 3.281; %meters to feet
pub = rospublisher('/raw_vel');
sub = rossubscriber('/encoders');

d=.25*unitConv;
Gradient = @(x) [(x(2)-2*x(1)-2); (x(1)-2*x(2)-2)];
Tangent = @(x) [cos(pi/2) sin(pi/2); -sin(pi/2) cos(pi/2)]*[(x(2)-2*x(1)-2); (x(1)-2*x(2)-2)]
% set initial vals
% Define the initial step-size
lambda = 1/16;
% Define the step-size multiplier
%delta = 1.2;
time = 0;
stoptime = 50;
stop = 0;
movmult = 1;
maxspd = .22;
sharpness = .8; %How much the robot cares about turning to the right angle before moving.  If
%^ 1 tends to do the job fine, 2 is too much with maxspd of .1
%desiredDistance = 0;

strtmsg = rosmessage(pub);
stopmsg = rosmessage(pub);

stopmsg.Data = [0 , 0];

tic

%send(pub, strtmsg)
%wheeldata = receive(sub);
wheeldata = receive(sub);
data_old = wheeldata.Data * unitConv;
hold on;
```

```
points = [4 -8 4 -6; 1 -5 1 -6];
ang = pi/2;
j = 1;
while j <= size(points,2)
    tic
    x = points(:,j)
    while norm(Gradient(x)) > 0.3

        %v = norm(dr)
        desTrav = lambda.*Gradient(x);

        desAng = mod(atan2(desTrav(2), desTrav(1)),2*pi);
        %lambda = lambda.*delta;
        wheeldata = receive(sub);
        data_new = wheeldata.Data * unitConv;
        dpl = data_new(1) - data_old(1);
        dpr = data_new(2) - data_old(2);
        dp = (dpl + dpr)/2;
        dang=(dpr-dpl)/d;
        dx=dp*cos(ang);
        dy=dp*sin(ang);
        ang = ang+dang;
        x = x+[dx;dy];
        v = norm(desTrav);

        angDiff = ang - desAng;
        if angDiff > pi
            angDiff = angDiff - 2*pi;
        elseif angDiff < -pi
            angDiff = angDiff + 2*pi;
        end

        w = -angDiff*sharpness;
        Vs= magclip2(v-((w*d)/2), v+((w*d)/2), maxspd);
        VL = Vs(1);
        VR = Vs(2);
        strtmsg.Data = [VL, VR];
        send(pub, strtmsg)
        if toc>stoptime
            send(pub, stopmsg)
            break
        end
        data_old = data_new;
        plot(x(1), x(2), 'bo')
    end
    send(pub, stopmsg)
    pause(2)
    j = j + 1;
end
send(pub, stopmsg)
```