```
%COMPLETE CODE

clear


mylego = legoev3('usb'); % Set up MATLAB and EV3 communication
mymotorA = motor(mylego,'A'); % Set up motor A
mymotorB = motor(mylego,'B'); % Set up motor B
mymotorA.Speed = 0;
mymotorB.Speed = 0;
resetRotation(mymotorA)
resetRotation(mymotorB);
start(mymotorA);
start(mymotorB)


motor_on = true;

L1 = 105;
L2 = 40;

mytouchsensor1 = touchSensor(mylego,1); % set up sensor 1

%%
%main
coordinates = [0, 145]; % x and y coordinates of desired point

coordinates_1 = [45, 135];
coordinates_2 = [0, 145];

%MeasureDistance(mylego, mymotorA, mymotorB, L1, L2, mytouchsensor1);
%MeasureAngle(mylego, mymotorA, mymotorB, L1, L2, mytouchsensor1);


%testCoord(mylego, mymotorA, mymotorB, L1, L2, mytouchsensor1)

%angles = getAngle(coordinates(1), coordinates(2), L1, L2)
%SetAngles(mymotorA, mymotorB, angles);

drawLine(mymotorA, mymotorB, L1, L2, coordinates_1, coordinates_2);

stop(mymotorA);
stop(mymotorB);
%%
% %% Part 1 digital controller implementation
%


function SetAngles(mymotorA, mymotorB, angles)

    t = 0:Ts:(N-1)*Ts;
    Tsim = 4;     % simulate over Tsim seconds
    Ts = 0.01;    % sampling time
    N= Tsim/Ts;   % number of steps to simulate
    u1 = zeros(1,N); % plant A input
    u2 = zeros(1,N); % plant B input
    e1 = zeros(1,N); % error 1
    e2 = zeros(1,N); % error 2
    current_angle1 = zeros(1,N); % c1
    current_angle2 = zeros(1,N); % c2
    set_angle1 = angles(1);  % reference for A -- r1
    set_angle2 = angles(2);  % reference for B -- r2
    k=2;

    e1(k) = 0;
    e2(k) = 0;

    while (k<=N)
        % --- Motor A controller
        current_angle1(k) = 180 + readRotation(mymotorA);
        e1(k) = set_angle1 - current_angle1(k);
        u1(k) = u1(k-1) + 0.412*e1(k)- 0.4114*e1(k-1);
        mymotorA.Speed = u1(k);

        %--- Motor B controller
        current_angle2(k) = 180 + readRotation(mymotorB);
        e2(k) = set_angle2 - current_angle2(k);
        u2(k) = u2(k-1) + 0.412*e2(k)- 0.4114*e2(k-1);
        mymotorB.Speed = u2(k);

        k = k + 1;
        pause(0.01)

    end

    plot(t, current_angle1);
    hold on
    plot(t, current_angle2);
    xlabel('Time (s)')
    ylabel('Angle')
    title('Motor Response')


end

%%

function MeasureDistance(mylego, mymotorA, mymotorB, L1, L2, mytouchsensor1)
    clearLCD(mylego);
    count = 0;
    while true
        while(~readTouch(mytouchsensor1))
        end
        pause(0.3);
        count = count + 1;
        if (count == 1)
            coordinates_1 = calculate_coordinates(mymotorA, mymotorB, L1, L2);

        elseif (count == 2)
            coordinates_2 = calculate_coordinates(mymotorA, mymotorB, L1, L2);

            distance = sqrt((coordinates_2(1) - coordinates_1(1))^2 +
             (coordinates_2(2) - coordinates_1(2))^2);

            break;
        end

    end

    clearLCD(mylego);
    writeLCD(mylego,strcat('measureDistance'),1,1);

    writeLCD(mylego,strcat('Distance: ', num2str(distance), 'mm'),2,1);

end

%%

%%
function coordinates = calculate_coordinates(mymotorA, mymotorB, L1, L2)

    angle1 = cast(readRotation(mymotorA), 'single');
    angle2 = -cast(readRotation(mymotorB), 'single');

    x_coordinate = L1*cosd(angle1) + L2*cosd(angle1+angle2);
    y_coordinate = L1*sind(angle1) + L2*sind(angle1+angle2);

    coordinates = [x_coordinate; y_coordinate]

end

function angles = getAngle(xw_coordinate, yw_coordinate, L1, L2)

    theta_1 = (atand(yw_coordinate/xw_coordinate)) - (acosd((xw_coordinate^2 +
     yw_coordinate^2+L1^2-L2^2)/(2*L1*(sqrt(xw_coordinate^2 +
     yw_coordinate^2)))));
    theta_2 = -(180 - (acosd((L1^2+L2^2-xw_coordinate^2 -
     yw_coordinate^2)/(2*L1*L2))));


    angles = [theta_1, theta_2];
```

```
end

function DrawLine(mymotorA, mymotorB,L1, L2, coordinates_1, coordinates_2)

    slope = (coordinates_2(2) - coordinates_1(2))/(coordinates_2(1) -
     coordinates_1(1));
    y_intercept = coordinates_2(2) - (slope*coordinates_2(1));

    points_x = linspace(coordinates_1(1), coordinates_2(1), 25);
    n = length(points_x);
    points_y = zeros(1, n);
    i = 0;
    while i < n
        points_y(i+1) = slope*points_x(i+1) + y_intercept;
        i=i+1;
    end

    i = 0;
    while (i < n)
        angles  = getAngle(points_x(i+1), points_y(i+1), L1, L2);
        SetAngles(mymotorA, mymotorB, angles);
        i=i+1;
    end

end

%%
function MeasureAngle(mylego, mymotorA, mymotorB, L1, L2, mytouchsensor1)
    clearLCD(mylego);
    count = 0;
    angle = 0;

    while true
        while(~readTouch(mytouchsensor1))
        end
        pause(0.5);
        count = count + 1;
        if (count == 1)
            coordinates_int = calculate_coordinates(mymotorA, mymotorB, L1,
             L2);

        elseif (count == 2)
            coordinates_l1 = calculate_coordinates(mymotorA, mymotorB, L1, L2);
            m1 = ((coordinates_l1(2) - coordinates_int(2)) /
             (coordinates_l1(1) - coordinates_int(1)));

        elseif (count == 3)
            coordinates_l2 = calculate_coordinates(mymotorA, mymotorB, L1, L2);
            m2 = ((coordinates_l2(2) - coordinates_int(2)) /
             (coordinates_l2(1) - coordinates_int(1)));
            angle = atand(abs((m2 - m1) / (1 + m2 * m1)));
            break;
        end

    end

    clearLCD(mylego);

    writeLCD(mylego,strcat('measureAngle'),1,1);

    writeLCD(mylego,strcat('Angle: ', num2str(angle), 'Degs'),2,1);

end

%%

function testCoord(mylego, mymotorA, mymotorB, L1, L2, mytouchsensor1)
    clearLCD(mylego);
    count = 0;


    while true
        while(~readTouch(mytouchsensor1))
        end
        pause(0.3);
        count = count + 1;
        if (count ~= 7)
            calculate_coordinates(mymotorA, mymotorB, L1, L2);
        else
            calculate_coordinates(mymotorA, mymotorB, L1, L2);

            break;
        end

    end


end
```