**ENED 1091: Homework #2**

**Due: Week of February 8th at beginning of Recitation**

**Problem 1:** The height, y, of a projectile launched at an angle, , and an initial velocity equal to Vo is given by:



|  |  |
| --- | --- |
| y | Projectile height (m) |
| g | Gravitational Constant |
| Vo | Initial Velocity (m/s) |
| Ɵ | Launch Angle (degrees or radians) |
| yo | Initial Height (m) |

1. Assuming an initial height of zero, use the symbolic toolbox to solve for the time of impact (when projectile hits the ground) in terms of the variables g, Vo, and .

*Notes:*

* *Obviously you could do this by hand algebraically or simply Google the answer, but to receive credit for this part, you must enter the equation as a symbolic expression in MATLAB (setting y = 0) and solve for t.*
* *Remember sind and cosd don’t work in symbolic toolbox*
* *Never start a variable name with the small letters: th when using symbolic toolbox functions. So theta is unacceptable but Theta or even Th is OK.*

**MATLAB Commands**

height\_fcn = sym('0 = -0.5\*g\*t^2 + V\_0\*sin(Th)\*t')

time\_fcn = solve(height\_fcn, 't')

**Impact Time =**

0

(2.0\*V\_0\*sin(Th))/g

1. Repeat part (a) leaving the initial height, yo, as a variable as well. Again, this must be done in MATLAB.

**MATLAB Commands**

height\_fcn = sym('0 = -0.5\*g\*t^2 + V\_0\*sin(Th)\*t + y\_0')

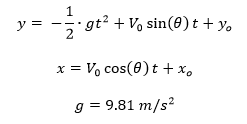
time\_fcn = solve(height\_fcn, 't')

**Impact Time =**

(0.5\*(2.0\*V\_0\*sin(Th) + (4.0\*V\_0^2\*sin(Th)^2 + 8.0\*g\*y\_0)^(1/2)))/g

(0.5\*(2.0\*V\_0\*sin(Th) - (4.0\*V\_0^2\*sin(Th)^2 + 8.0\*g\*y\_0)^(1/2)))/g

**Problem 2:** The equations of motion for the x and y position of a projectile fired at an initial velocity of Vo (m/s) and at an angle of θ (degrees) assuming an initial position of (xo,yo) are:



Assume the initial position of the projectile is (0m, 0m). Suppose there is a target located at (1200m, 1600m) and we wish to hit the target in 10 seconds.

1. Using the ***solve*** function in MATLAB, find the initial velocity, V0, and the launch angle required to hit the target in 10 seconds**.** *Hint: solve the two motion equations simultaneously by plugging in all known values – the only unknowns in your symbolic expressions should be the two variables that you are solving for: Vo and* *. Then use a single solve command.*

**MATLAB Commands:**

y\_fcn = sym('y = -0.5\*g\*t^2 + V\_0\*sin(Th)\*t + y\_0');

x\_fcn = sym('x = V\_0\*cos(Th)\*t + x\_0');

result = solve(x\_fcn, y\_fcn, 'V\_0', 'Th');

g = 9.81; y\_0 = 0; x\_0 = 0; x = 1200; y = 1600; t = 10;

ansV\_0 = double(subs(result.V\_0)); ansTh = double(subs(result.Th));

angle = ansTh\*180/pi;

disp(ansV\_0);

disp(angle);

**Valid Initial Velocity, Vo =** 241.0434 m/s

**Valid Launch Angle,  =** 60.1431⁰

1. Verify your solution by plotting the target and the projectile path. Don’t use symbolic expressions for this part. Simply create a vector of time values and plug these values into the given equations to create vectors for x and y. Then plot y vs. x and the target on the same graph. Turn in your plot and the associated MATLAB commands. Remember title, labels, and units.

**MATLAB Commands and Plot:**

t = 0:.1:20;

x = zeros(1,201);

y = zeros(1,201);

figure(1)

plot(1200, 1600, 'rx'); hold on;

for k = 1:201

x(k) = 241.0434\*cosd(60.1431)\*(t(k));

y(k) = 241.0434\*sind(60.1431)\*(t(k)) - 4.9\*(t(k))^2;

plot(x(k), y(k), 'bo'); hold on;

end

xlabel('Distance(m)','FontSize',14);

ylabel('Height(m)','FontSize',14);

title('Height vs. Distance','FontSize',20);

legend('Target','Trajectory');



1. Add the following code after your code for part (b). The code assumes that you used the variable names x and y for the x and y position of the projectile – if not modify the first two lines of this code to use your variable names for x and y.

figure % Opens new figure window

for k = 1:length(x)

plot(x(k),y(k),'ko',1200,1600,'r\*');

axis([0 1250 0 2000])

pause(0.01); % 0.01 was the increment in my time vector

end

title('Projectile Path')

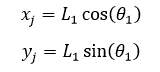
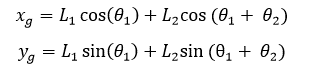
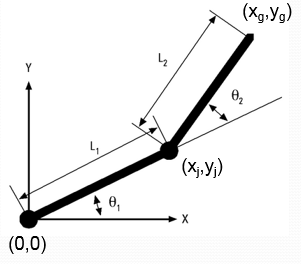
ylabel('Projectile Height (m)');

xlabel('Projectile Range (m)');

**What happens?**

It shows the projectile’s path in a motion clip instead of a static line. The projectile does hit the point.

**Problem 3:** The diagram below shows a schematic for a 2-link robot arm along with the key equations describing the joint and gripper coordinates. Assume that L1 = 5 ft., L2 = 4 ft.,  = 30o and  = 20o.



1. Calculate the joint coordinates (xj,yj) and the gripper coordinates (xg,yg).

**(xj, yj) =** (4.3301, 2.5) **(xg, yg) =** (6.9013, 5.5642)

1. What is the maximum reach for the robot arm (i.e., greatest distance between the base of the robot and the gripper)?

**Maximum Reach =** 9 ft

1. How close can the gripper get to the base?

**Minimum Distance From Base =** 1 ft

1. Suppose we want the robot arm to move so that the final gripper coordinates are (2, 4) ft. Use the symbolic toolbox to calculate the final angle values 1 and 2 assuming the following constraints: **.** Don’t forget to use the ***double*** function to covert the result from a symbolic to a numerical value!

**MATLAB Commands:**

xg\_fcn = sym('xg = L1\*cos(Th1) + L2\*cos(Th1 + Th2)');

yg\_fcn = sym('yg = L1\*sin(Th1) + L2\*sin(Th1 + Th2)');

angles = solve(xg\_fcn, yg\_fcn, 'Th1', 'Th2');

L1 = 5; L2 = 4; xg = 2; yg = 4;

angle1 = double(subs(angles.Th1));

angle2 = double(subs(angles.Th2));

angle1 = angle1\*180/pi;

angle2 = angle2\*180/pi;

disp(angle1);

disp(angle2);

**Final Base Angle,  =** 13.8605

**Final Joint Angle,  =** 121.6682

1. Plot the 2-link robot arm at the final position (Hint: refer to last Lecture 4 slide)

**MATLAB Commands & Plot**

xj\_fcn = sym('xj = L1\*cos(Th1)');

yj\_fcn = sym('yj = L1\*sin(Th1)');

xg\_fcn = sym('xg = L1\*cos(Th1) + L2\*cos(Th1 + Th2)');

yg\_fcn = sym('yg = L1\*sin(Th1) + L2\*sin(Th1 + Th2)');

angles = solve(xg\_fcn, yg\_fcn, 'Th1', 'Th2');

L1 = 5; L2 = 4; xg = 2; yg = 4;

angle1 = double(subs(angles.Th1));

angle2 = double(subs(angles.Th2));

angle1 = angle1\*180/pi;

angle2 = angle2\*180/pi;

disp(angle1);

disp(angle2);

xj = L1\*cosd(angle1(2));

yj = L1\*sind(angle1(2));

plot([0 xj],[0 yj],'k-o',[xj xg],[yj yg],'k-o');

xlabel('Horizontal Distance (m)','FontSize',14);

ylabel('Vetical Distance (m)','FontSize',14);

title('Map of a Articulated Arm','FontSize',16);

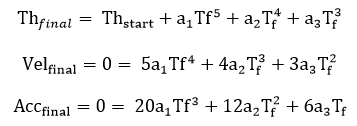
****

**Problem 4:** In problem 3, you determined the base and joint angles required to position the gripper of the robot arm at (2, 4) ft. One of the main problems in robotics is path planning which involves finding a collision-free path along which to move a robot from the starting position to the final destination. We are going to make the assumption that nothing is in the way of the 2-link robot arm and each angle is going to follow a smooth polynomial path:



*Note: I’ve replaced the variable with the variable Th.*

To calculate the constants, a1, a2, and a3, we need additional information: the desired amount of time to travel from the initial position to the final position, Tf. We can then use boundary conditions (final angle, velocity, and acceleration) to write three equation involving the three unknowns:



If we had numerical values for the initial and final angles and the final time, Tf, we could simply solve for the unknowns using a matrix equation.

1. Use the solve command to solve for the three polynomial coefficients (a1, a2, and a3) in terms of the variables Thfinal, Thstart, and Tf. *Hint: enter in all three equations as symbolic expressions and solve them simultaneously with a single solve command.*

**a1 =** (6\*(Th\_f - Th\_i))/Tf^5

**a2 =** -(15\*(Th\_f - Th\_i))/Tf^4

**a3 =** (10\*(Th\_f - Th\_i))/Tf^3

**MATLAB Commands:**

f1 = sym('Th\_f = Th\_i + a1\*Tf^5 + a2\*Tf^4 + a3\*Tf^3');

f2 = sym('0 = 5\*a1\*Tf^4 + 4\*a2\*Tf^3 + 3\*a3\*Tf^2');

f3 = sym('0 = 20\*a1\*Tf^3 + 12\*a2\*Tf^2 + 6\*a3\*Tf');

aEqns = solve(f1, f2, f3, 'a1', 'a2', 'a3');

disp(aEqns.a1);

disp(aEqns.a2);

disp(aEqns.a3);

1. Using your equations for a1, a2, and a3 in part (a), calculate a polynomial path for the base angle assuming Tf = 10 seconds, Th1start = 30o, and Th1final = 13.86o.

**Th1(t) = 30 -** .0009684\*Tf^5 **+** .0242\*Tf^4 - .1614\*Tf^3

1. Using your equations for a1, a2, and a3 in part (a), calculate a polynomial path for the joint angle assuming Tf = 10 seconds, Th2start = 20o, and Th2final = 121.67o.

**Th2(t) = 20 +** .0061\*Tf^5 - .1525\*Tf^4 + 1.0167\*Tf^3

1. Create a vector of time values, t, starting at 0, incrementing by 0.1, and ending at 10. Plug these values into your equations in (b) and (c) to calculate a vector of values for the base angle and the joint angle. ***Don’t use symbolic expressions for this – just enter in the polynomial equations.*** Plot the base angle and the joint angle with time on the x-axis. Don’t forget about titles, labels, and units. Verify graphically that each angle starts and ends where it is supposed to then paste your plot and MATLAB commands below.

**PLOT and MATLAB Commands:**

t = 0:.1:10;

base = 30 - .0009684\*t.^5 + .0242\*t.^4 - .1614\*t.^3;

joint = 20 + .0061\*t.^5 - .1525\*t.^4 + 1.0167\*t.^3;

figure(1)

plot(t,base,'r-',t,joint,'b-');

xlabel('Time(s)','FontSize',14);

ylabel('Angle(degrees)','FontSize',14);

title('Angle vs. Time','FontSize',16);

legend('Base Angle','Joint Angle');

****

1. Calculate the joint (xj, yj) and gripper (xg, yg) positions over time by plugging in your vectors of angle values. Again, just enter the equations, you don’t need symbolic toolbox. Also, remember your angles are in degrees. Now write MATLAB code to create a plot that shows the 2-link robot arm moving from the starting position to the final position over a time period of 10 seconds. Hint: modify the code given to you in Problem 2(c) making use of the plot statement you wrote in Problem 3(e).

**MATLAB Commands:**

L1 = 5; L2 = 4;

t = 0:.1:10;

base = 30 - .0009684\*t.^5 + .0242\*t.^4 - .1614\*t.^3;

joint = 20 + .0061\*t.^5 - .1525\*t.^4 + 1.0167\*t.^3;

xj = L1\*cosd(base);

yj = L1\*sind(base);

xg = L1\*cosd(base) + L2\*cosd(base + joint);

yg = L1\*sind(base) + L2\*sind(base + joint);

figure(1)

for k = 1:length(t)

plot([0 xj(k)],[0 yj(k)],'k-o',[xj(k) xg(k)],[yj(k) yg(k)],'k-o');

axis([0 10 0 10])

pause(0.1); % 0.1 was the increment in my time vector

end

title('Articulated Arm Path')

ylabel('Horizontal Distance(m)');

xlabel('Vetical Distance(m)');