**ME 4773/5493 Fundamental of Robotics**

**Fall 2016**

Simulation of a Jumping Two Link System With End Mass

Christian Wahrmund

*Department of Mechanical Engineering, The University of Texas at San Antonio, UTSA*

This paper details the construction and results of a simulated jumping two link model. After deriving relevant equations of motion using an inverse dynamics model, a MATLAB simulation of the jumping two link system consisting of a thigh and shank with a point mass was constructed and then solved for over a set time span. Joint torque, reaction force, and angular position, velocity, and acceleration are mapped versus time across the stance phase of the jump cycle.

# INTRODUCTION

In order to build devices that provide more efficient locomotion, it is necessary to construct simplified models of gait motion to predict how the forces involved in motion will change. While there are several ways to derive relevant equations of motion for a gait cycle the inverse dynamics method was chosen for its relative simplicity and accuracy [1] [4] [5]. After solving for all unknowns, the only necessary input to the systems is the change in position with respect to time, . Once position data for the first joint is input and initial conditions are set, the simulation will find relevant data for the jumping gait cycle across a time span. For the purposes of testing the model, approximated joint position data was used for the jump stance phase. Future work to the model includes capturing actual joint position data with a Vicon motion tracker system to input to the model.

# NOMENCLATURE

1. = The reaction of the point mass against the thigh in the y direction

2. = The reaction of the point mass against the thigh in the x direction

3. = The reaction of the shank to thigh in the y direction

4. = The reaction of the shank to thigh in the x direction

5. = The reaction of ground to shank in the y direction

6. = The reaction of ground to shank in the x direction

7. = Moment about the beginning of the thigh

8. = Moment about connecting joint of thigh and shank

10. = Angular position of thigh

11. = Angular velocity of thigh

12. = Angular acceleration of thigh

13. = Angular position of shank

14. = Angular velocity of shank

15. = Angular acceleration of shank

16. = Mass moment of inertia of thigh

17. = Mass moment of inertia of shank

18. = y component of linear acceleration of thigh

19. = x component of linear acceleration of thigh

20. = y component of linear acceleration of shank

21. = x component of linear acceleration of shank

22. = Mass of thigh

23. = Mass of shank

24. **=** Length of thigh

25. = Length of shank

# METHODS

## Inverse Dynamics

Building the model started by performing an inverse dynamics analysis on a simple two link system consisting of a thigh, shank, and point mass. While considered an accurate model there are, however, several assumptions and generalizations made that should be noted.

1. All joints are assumed to be zero friction hinge joints with one degree of rotational freedom
2. All links are perfectly stiff with no damping
3. The density of all links is perfectly homogenous
4. All forces associated with linear acceleration and gravity act about the midpoint of each link
5. The center of rotation about the joints does not change throughout a gait cycle

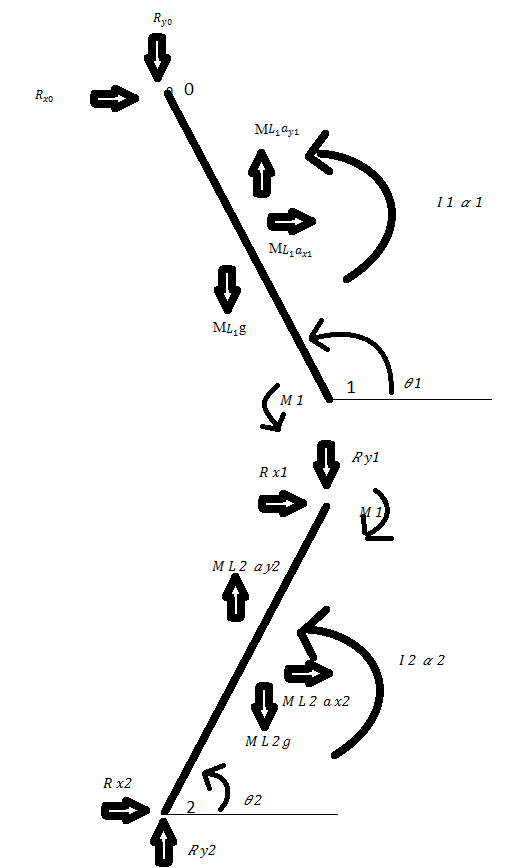
9. = Moment of shank about ground

The final developed model with all reactions and moments is shown in Figure **1**. This model splits the system into two separate links and solves for reaction and torque about each individual link. The equations for the moment of joints one and two are shown in Equation 1 and Equation 2

Equation 5: This equation is the output of Equation 1after it has been solved symbolically in MATLAB for the reaction force in the y direction of joint one.

Equation 1: This equation finds the moment about joint 1. In this system, this is the only joint that will have a moment.

Equation 2: As the moment about the end of link two is always zero, this equation is used to solve for the reaction forces of the ground on the two link system.

The point mass magnitude, mass of each length, length of each length, and initial values of angular position, velocity, and acceleration are required along with the rate of change of position over time to solve Equation 1 and Equation 2. The mass moment of inertia is found for each link by Equation 3 and Equation 4.

Equation 3: The mass moment of inertia of the first length is found by integrating for the moments of inertia for a solid rod and a point mass. They are then combined using the parallel axis theorem

Equation 4: The mass moment of inertia of the second length is found by modelling the second link as a solid rod

After solving for the moment of inertia of each link the only unknowns left in the system are the angular velocity and angular acceleration of each joint. As the position of the joint is given with respect to time the MATLAB function “diff” is used to obtain the first and second derivatives of angular position (angular velocity and angular acceleration) with respect to time.

Figure **1**: Inverse dynamics model of two link system showing moments, reactions about the joints, and torque required for angular acceleration of the link.

Equation 4: The mass moment of inertia about link two is found by using the moment of inertia equation for a simple solid rod.

As the moment about the end of link two is zero, Equation 2 is used to solve for the reaction forces of the shank. Equation 1 is input into MATLAB and symbolically solved to find the reaction of link one in the y direction about link two.



After solving for the reactions of link one, a simple static summation of forces can be used to find the reaction of link two about the ground as shown in Equation 6.

( - g) -

Equation 6: After summing the forces in the y direction of link two the reaction force of link two is found

# RESULTS

## MATLAB Model

After solving for all equations of motion and inputting them along into MATLAB, initial conditions where set and all equations where solved over a time span with a step size set by Equation 7

Equation 7: This sets the number of times all equations are solved for. The step size is determined by the h parameter, The linspace function is used to create an even number of time points between and .

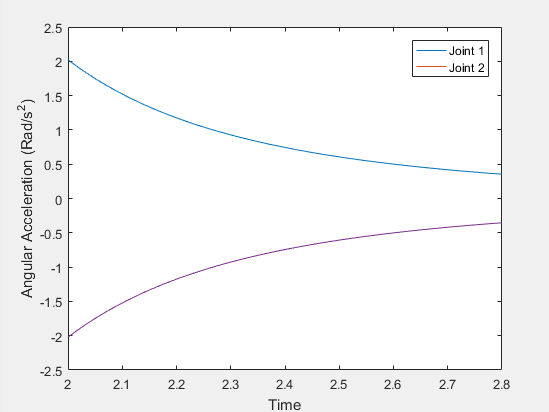
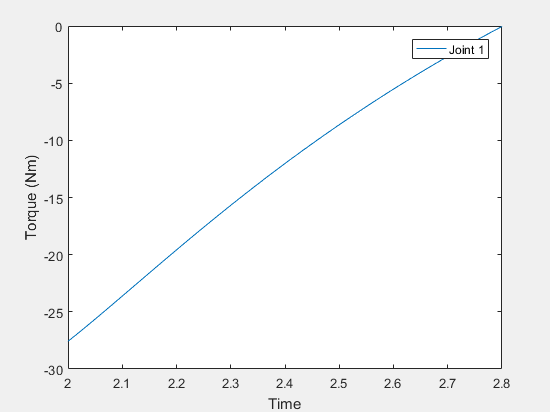
This creates a set of [1 x N] matrices of each output that can then be plotted to time. 5 variables were chosen to be plotted against time, the angular position, velocity, acceleration, and torque of each joint as well as the reaction forces in the y direction of the end of each joint. An initial length of .8 meters was given for each joint with a link mass of 2 kg. The beginning of link one was loaded with a point mass of 10 kg. finally initial positions of and were set to and respectively.

Figure 4: Similar to Figure 3, this graph shows the angular acceleration of each joint across the stance phase of the two link system for a jump cycle.

Figure 3: This graph shows the angular velocity of each joint about the stance phase of the jump cycle.

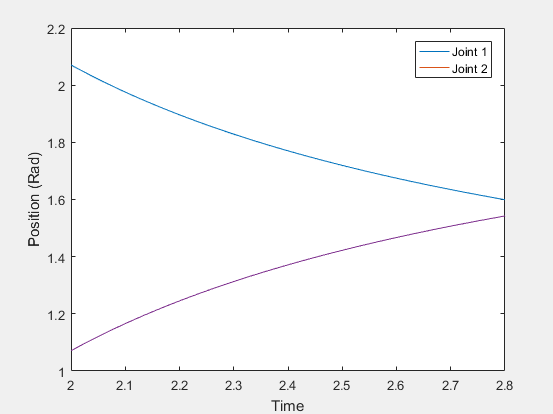


Figure 2: Plot detailing the position in radians of each joint over time. As both joints are the same length they also change at the same rate.

Figure 2 shows the angular position of each link with respect to the local x axis of each joint. As the two link system jumps the top link rotates clockwise while the bottom link rotates counterclockwise to straighten out the system causing an upward jumping motion.

The velocity (Figure 3) linearly decays throughout the stance phase of the jump cycle simulating an exponential decay of rotational speed of each joint. This is an approximation of what the joint velocity would be, towards the end of the cycle the velocity and angular acceleration (Figure 4) should trend to zero. This issue is addressed further in section VI.

Figure 5: Graph of torque (moment) about each joint. The moments about joint 0 and 2 are zero.

Figure 6: This graph shows the reaction force in the y direction about the end of link 1 and more importantly about the end of link 2.

Figure 5 shows the decreasing torque about joint 1. As the two link straightens out and begins to transition to the swing phase less torque is required to hold and accelerate the system until finally the torque input of the joint is 0. This coincides with the moment the two link system leaves the ground. The beginning of link one and end of link two have no torque input or output.

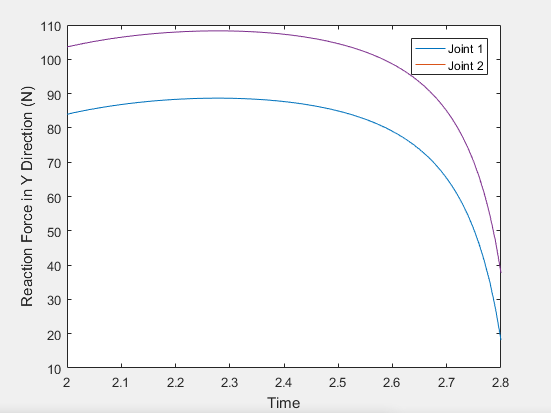
For the first half of the stance phase the reaction force (Figure 6) on each joint increases beyond the initial reaction force, this is due to the link not only supporting the mass of the system but also supporting the force of the links as it is accelerated forward. After roughly one third of the way through the stance phase the rotating mass begins to pull up on the link and reduces the reaction force felt by the ground.

# CONCLUSION

The purpose of this project was to build a simulation within MATLAB that would output the reaction forces at the ground and the moment about the middle joint of a jumping two link system. After using an inverse dynamics model to derive all relevant equations of motion, initial conditions were set, and an approximated joint position function with respect to time was input. After running the simulation with a step size of h = .01 5 outputs (angular position, velocity, acceleration, joint reaction forces, joint torque) were mapped against time.

# FUTURE WORK

This project was built to help the ARM lab with its ongoing research in the field of exoskeletons. Once the Vicon motion capture system is up and running actual joint position data as opposed to approximated joint position data will be input into the system and the simulated values of reaction force against the ground will be compared to data captured from a force plate system. Further work will be done to modify this system for a walking gait cycle.



# ACKNOWLEDMENTS

The author would like to thank Professor Pranav Bhounsule for his support in constructing the MATLAB model and to Professor Amir Jafari.

# REFERENCES

|  |  |
| --- | --- |
| [1] | "Clinical Gait Analysis," [Online]. Available: http://www.clinicalgaitanalysis.com/teach-in/inverse-dynamics.html. [Accessed 12 2016]. |
| [2] | R. Miller, B. Umberger and G. Caldwell, "OPTIMAL CONTROL SOLUTIONS FOR A SIMPLE MODEL OF HUMAN JUMPING," in *ASME 2008 Summer Bioengineering Conference*, Marco Island, Florida, USA, 2008. |
| [3] | M. M. van der Krogt, W. W. de Graaf, C. Farley, C. Moritz, R. Casius and M. Bobbert, "Robust passive dynamics of the musculoskeletal system compensate for," *Applied Physioligy,* pp. 801-808, 2009. |
| [4] | C. Vaughan, B. Davis and O. Jeremy, Dynamics of Human Gait, vol. 2nd , Kiboho Publishers, 1999, pp. 15-43. |
| [5] | S. Whittlesey and G. Robertson, "Two-Dimensional Inverse Dynamics," in *Research Methods in Biomechanics*, Human Kinetics, 2013, pp. 103-122. |

# APPENDIX

## Main Code

%Constants

pi = 3.14159265359;

g = 9.81;

L1 = .8; %Lengths of link 1 and link 2

L2 = .8;

h = .01; %step size, lower values improve resolution

M1 = 10; % magnitude of point mass located at begining of link 1

ML1 = 2;

ML2 = 2; %magnitude of mass of links 1 and 2

%Inputs

Th1st = .75\*pi; %starting angle of Theta 1

Th1end = pi/2;

Tstart = 2;

Tend = 2.8; % starting and ending time of jump cycle

%Equations

N = round (((Tend - Tstart)/h),0);

t = linspace(Tstart, Tend, N);

syms T;

R0y = M1\*g;

%equations of position, angular velocity and angular acceleration of joint 1

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Th1 = .2\*3.14+1/log(T); %Th1st - .42\*T.^2 - .1\*T.^3;

dth1 = diff(Th1,T);

ddth1 = diff(dth1,T);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% solving for equations of position, angular velocity and angular acceleration of joint 1

% solve position of joint 2

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Th1 = subs(Th1, T, t);

dth1 = subs(dth1, T, t);

ddth1 = subs(ddth1, T, t);

Th2 = pi - Th1;

dth2 = -dth1;

ddth2 = -ddth1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

I1 =(1/3)\*ML1\*L1.^2 + M1\*L1.^2;

I2 = (1/3)\*ML2\*L2.^2;

T1 = I1\*ddth1 + L1\*cos(Th1)\*R0y + (ML1\*g\*L1\*cos(Th1))/2;

R1y = -(T1 + I2\*ddth2 - (L2\*ML2\*g\*cos(Th2))/2)./(L2\*cos(Th2));

R2y = R1y + ML2\*g;

%%%%%%%%%%%%%%%%%%%%%%% Plotting information %%%%%%%%%%%%%%%%%%%%%%%

Time = transpose(eye(N,1)\*t);

Torque\_1 = transpose(eye(N,1)\*T1);

Pos\_J1 = transpose(eye(N,1)\*Th1);

Pos\_J2 = transpose(eye(N,1)\*Th2);

Ang\_Vel\_J1 = transpose(eye(N,1)\*dth1);

Ang\_Vel\_J2 = transpose(eye(N,1)\*dth2);

Ang\_Accel\_J1 = transpose(eye(N,1)\*ddth1);

Ang\_Accel\_J2 = transpose(eye(N,1)\*ddth2);

Reaction\_J1y = transpose(eye(N,1)\*R1y);

Reaction\_J2y = transpose(eye(N,1)\*R2y);

if Torque\_Plot == 1

figure('Name','Torque at Joints over Jump Cycle','NumberTitle','off')

plot (Time, Torque\_1)

xlabel('Time');

ylabel('Torque (Nm)');

legend ('Joint 1');

xlim([Tstart,Tend]);

end

if Position\_Plot == 1

figure('Name','Position at Joints over Jump Cycle','NumberTitle','off')

plot (Time, Pos\_J1, Time, Pos\_J2)

xlabel('Time');

ylabel('Position (Rad)');

legend ('Joint 1', 'Joint 2');

xlim([Tstart,Tend]);

end

if Ang\_Vel\_Plot == 1

figure('Name','Angular Velocity of Joints over Jump Cycle','NumberTitle','off')

plot (Time, Ang\_Vel\_J1, Time, Ang\_Vel\_J2)

xlabel('Time');

ylabel('Angular Velocity (Rad/s');

legend ('Joint 1', 'Joint 2');

xlim([Tstart,Tend]);

end

if Ang\_Accel\_Plot == 1

figure('Name','Angular Acceleration at Joints over Jump Cycle','NumberTitle','off')

plot (Time, Ang\_Accel\_J1, Time, Ang\_Accel\_J2)

xlabel('Time');

ylabel('Angular Acceleration (Rad/s^2)');

legend ('Joint 1', 'Joint 2');

xlim([Tstart,Tend]);

end

if Reaction\_Force\_Plot == 1

figure('Name','Reaction Forces at Joints','NumberTitle','off')

plot (Time, Reaction\_J1y, Time, Reaction\_J2y)

xlabel('Time');

ylabel('Reaction Force in Y Direction (N)');

legend ('Joint 1', 'Joint 2');

xlim([Tstart,Tend]);

end

## Symbolic Solver

clear all

close all

clc

%Solves Ry1 in terms of torque at joint one, run then copy and paste into main

%file

syms I2 ddth2 L2 Th2 R1y T1 ML2 g

T2 = I2\*ddth2 + L2\*cos(Th2)\*R1y + T1 - .5\*L2\*cos(Th2)\*ML2\*g;

Ry1 = solve (T2==0, R1y)