ASSIGNMENT 3

Upper Limb Model Matrix and Knee Extension

Name	: Jeremia Christ Immanuel Manalu
NRP	: 5023231017
Course	: Biomodelling (A)
Class	: A
Lecturer	: Dr. Achmad Arifin S.T., M.Eng.
Department	: Biomedical Engineering

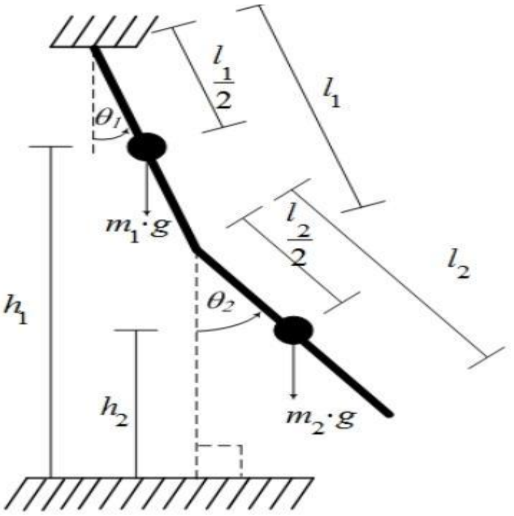
FACULTY OF INTELLIGENT ELECTRICAL AND INFORMATICS TECHNOLOGY

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

2025

CHAPTER I. FUNDAMENTAL THEORY

1.1. Kinetic Energy



$$h_1 = l_1 + l_2 - \frac{l_1}{2}\cos\theta_1 \qquad h_2 = l_1 + l_2 - \left(l_1\cos\theta_1 + \frac{l_2}{2}\cos\theta_2\right)$$

Figure 1.1. *An Example of Double Pendulum Properties*

The total kinetic energy of the system contains the translational contribution of the center of mass (COM) of each segment and the rotation of each segment. For a double-pendulum consisting of two segments (index 1 and 2) with masses m_1, m_2, COM distances r_1, r_2 from the joint axis, moments of inertia I_1, I_2, bone/rod lengths l_1, l_2, and general angles θ_1, θ_2 (for example θ_1= shoulder angle, θ_2 = relative angle at the elbow). The COM positions and velocities can be written down and then inserted into the kinetic energy.

Position (writing example) for COM:

$$x_1 = r_1 \sin\,\theta_1, \qquad y_1 = -r_1 \cos\,\theta_1,$$
$$x_2 = l_1 \sin\,\theta_1 + r_2 \sin\,(\theta_1 + \theta_2), \quad y_2 = -l_1 \cos\,\theta_1 - r_2 \cos\,(\theta_1 + \theta_2).$$

The squared velocity of each COM (with time derivative $\dot{\theta}_i$) yields the expression v_1^2, v_2^2.

The full kinetic energy:

$$T \;= \frac{1}{2} m_1 v_1^2 + \frac{1}{2} I_1 \dot{\theta}_1^{\,2} + \frac{1}{2} m_2 v_2^2 + \frac{1}{2} I_2 (\dot{\theta}_1 + \dot{\theta}_2)^2.$$

By plugging in v_1^2 and v_2^2 (their derivatives) we get a form that is usually written as the square of $\dot{\theta}_1$, $\dot{\theta}_2$ plus the cross term $\dot{\theta}_1\,\dot{\theta}_2$. One frequently used form, emphasizing the structure of the inertia matrix, is:

$$T = \frac{1}{2}[\dot{\theta}_1 \quad \dot{\theta}_2] M(\theta_1, \theta_2) \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix},$$

where $M(\theta)$is a symmetric configuration-dependent mass/inertia matrix containing $I_i, m_i, l_i, r_i$and trigonometric functions like cos (θ_2). The full equation for Kinetic Energy is as follows:

$$E_K = \tfrac{1}{2} m_1 v_1^2 + \tfrac{1}{2} I_1 \dot{\theta}_1^2 + \tfrac{1}{2} m_2 v_2^2 + \tfrac{1}{2} I_2 \dot{\theta}_2^2$$

$$E_K = \tfrac{1}{2} m_1 \left(\tfrac{l_1}{2}\right)^2 \dot{\theta}_1^2 + \tfrac{1}{2} m_2 \left[l_1^2 \dot{\theta}_1^2 + \left(\tfrac{l_2}{2}\right)^2 \dot{\theta}_2^2 + 2 l_1 \left(\tfrac{l_2}{2}\right) \cos(\theta_1 - \theta_2) \dot{\theta}_1 \dot{\theta}_2 \right] + \tfrac{1}{2} I_1 \dot{\theta}_1^2 + \tfrac{1}{2} I_2 \dot{\theta}_2^2$$

$$E_K = \tfrac{1}{2}\left(m_1\frac{l_1^2}{4} + m_2 l_1^2 + I_1\right)\dot{\theta}_1^2 + \tfrac{1}{2}\left(m_2\frac{l_2^2}{4} + I_2\right)\dot{\theta}_2^2 + m_2\frac{l_1 l_2}{2}\cos(\theta_1 - \theta_2)\,\dot{\theta}_1\dot{\theta}_2$$

Where:

- $m1, m2$: mass of the upper arm and lower arm
- $l1, l2$: length of the upper arm and lower arm
- $I1, I2$: moment of inertia of each segment about its center of mass
- $\theta1, \theta2$: angle of rotation of the upper arm and lower arm about the reference axis
- $\dot{\theta}1, \dot{\theta}2$: angular velocity of the upper arm and lower arm

In simulations, kinetic energy can be used to determine the speed and impact of collisions between objects.

1.2. Potential Energy

Gravitational potential energy is computed from the vertical heights of each segment's center of mass relative to a datum (e.g. $y = 0$). For the example configuration:

$$V = m_1 g y_1 + m_2 g y_2.$$

Substituting y_1, y_2yields a form such as:

$$V = -m_1 g r_1 \cos\ \theta_1 - m_2 g(l_1 \cos\ \theta_1 + r_2 \cos\ (\theta_1 + \theta_2)),$$

Also noted that the sign convention depends on the chosen positive axis, the important point is that $\partial V/\ \partial\theta_i$produces the gravitational torques. This expression supplies the conservative-force terms that appear later in the Lagrangian equations. From **Figure 1.1**, the height of the center of mass of the upper arm and lower arm (m1 and m2) is given in the following equation:

$$h_1 = l_1 + l_2 - \tfrac{l_1}{2}\cos\theta_1,$$

$$h_2 = l_1 + l_2 - \left(l_1\cos\theta_1 + \tfrac{l_2}{2}\cos\theta_2\right).$$

The potential energy when substituted will be as in the given equation:

$$E_P = m_1 g h_1 + m_2 g h_2$$

$$E_P = m_1 g\left(l_1 + l_2 - \tfrac{l_1}{2}\cos\theta_1\right) + m_2 g\left[l_1 + l_2 - \left(l_1\cos\theta_1 + \tfrac{l_2}{2}\cos\theta_2\right)\right].$$

1.3. Torque

Torque (generalized torque τ_i) on generalized coordinate $\theta_i$is defined as virtual work per unit angular displacement. In the Lagrangian framework the relation between kinetic energy T, potential energy V, and external torques $\tau_i$is given by the Euler–Lagrange equations (see the Lagrangian section). Briefly:

For generalized coordinates q_i,

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i,$$

with $L = T - V$and Q_i the generalized (non-conservative) forces (e.g. applied torques). Thus, applied torques τchange the rate of generalized momentum $\partial L/\ \partial\dot{q}$, while gradients of the potential $\partial L/\ \partial q$produce gravitational torques (components of $G(\theta)$).

Common matrix form for multibody systems:

$$M(\theta)\,\ddot{\theta} + C(\theta,\dot{\theta})\,\dot{\theta} + G(\theta) = \tau,$$

where:

- $M(\theta)$ derives from T(mass/inertia matrix),
- $C(\theta,\dot{\theta})\dot{\theta}$ are Coriolis/centripetal terms coming from cross derivatives in T,

- $G(\theta) = \partial V/\,\partial\theta$ is the gravity torque vector,
- τ is the external/actuator torque vector.

1.4. Lagrangian Mechanics

- Lagrangian: $L(q,\dot{q},t) = T(q,\dot{q}) - V(q)$.
- Euler–Lagrange equations for each generalized coordinate q_i:

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_i}) - \frac{\partial L}{\partial q_i} = Q_i,$$

where $Q_i$are generalized non-conservative forces/torques. If only gravity acts and there is no non-conservative forcing, $Q_i = \tau_i$represents input torques; for passive systems $Q_i = 0$.

- Matrix form commonly used:

$$M(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + G(q) = \tau.$$

Terms used:

- $M(q)$: symmetric positive-definite mass/inertia matrix,
- $C(q,\dot{q})\dot{q}$: Coriolis and centripetal contributions (contains products $\dot{q}_i\dot{q}_j$),
- $G(q)$: conservative gravity vector,
- τ: input torques/damping/friction if present.
- Energy relation: $\partial L/\,\partial\dot{q}$is generalized momentum; its time derivative equals applied generalized forces minus conservative gradients. Lecture materials show the step-by-step derivation from positions → velocities → T, V→ L→ equations of motion.

1.5. Fourth Order Runge-Kutta Method

RK4 is an explicit numerical integrator for first-order ODE systems below:

$$\dot{y} = f(t,y), y(t_0) = y_0.$$

With timestep $h$and $y_n \approx y(t_n)$:

$$\begin{aligned}
k_1 &= f(t_n, y_n),\\
k_2 &= f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right),\\
k_3 &= f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right),\\
k_4 &= f(t_n + h, y_n + h\,k_3),\\
y_{n+1} &= y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4).
\end{aligned}$$

For second-order systems (like $\ddot{q} = g(q,\dot{q},t)$) convert to first-order form with the state vector:

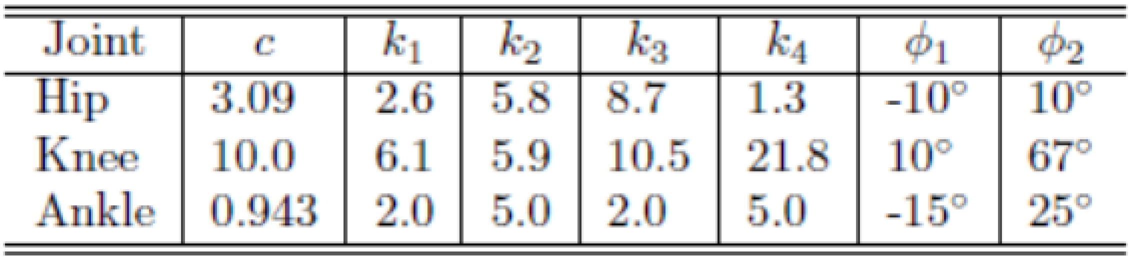
$$x = [\begin{matrix} q \\ \dot{q} \end{matrix}], \dot{x} = \begin{bmatrix} \dot{q} \\ g(q,\dot{q},t) \end{bmatrix},$$

and apply RK4 to $\dot{x} = F(t,x)$. There are also Runge–Kutta–Nyström variants that directly integrate second-order equations; the RK4 principle. combining four slope estimates for fourth-order accuracy, remains the same.

1.6. Anthropometric Data Standards

Table 1.1 (Passive Parameters) are lists the empirical coefficients (c, k, ϕ) used in the passive torque equation for lower limb joints. In this upper limb model, these functional forms are adapted to represent the damping and limits of the arm, elbow, and wrist.

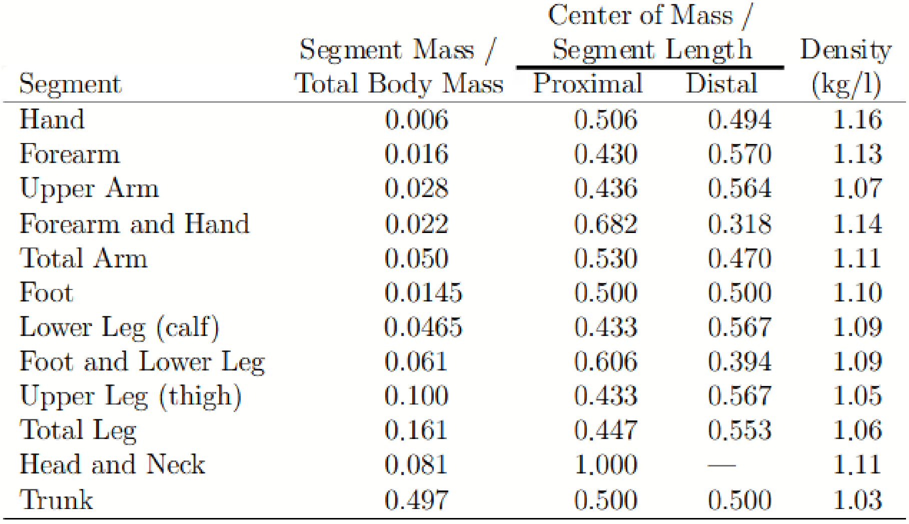
Table 1.1. *Parameters values of passive joint torque model*



Joint	c	k_1	k_2	k_3	k_4	ϕ_1	ϕ_2
Hip	3.09	2.6	5.8	8.7	1.3	-10°	10°
Knee	10.0	6.1	5.9	10.5	21.8	10°	67°
Ankle	0.943	2.0	5.0	2.0	5.0	-15°	25°

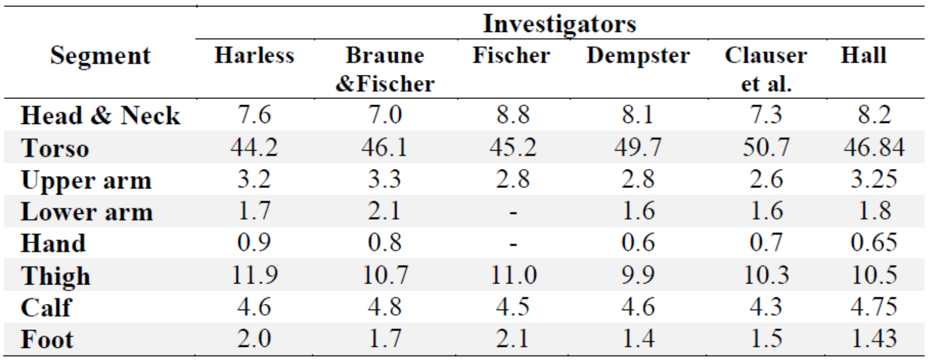
Table 1.2 and 1.3 (Segment Weights), these tables, derived from studies by investigators like Dempster, Braune, and Fischer, provide the regression coefficients to calculate the mass of individual segments (Upper Arm, Forearm, Hand) as a percentage of the total Body Weight.

Table 1.2. *Normalized Mass and Length of Body Segments (Standard Human)*



		Center of Mass / Segment Length		
Segment	Segment Mass / Total Body Mass	Proximal	Distal	Density (kg/l)
Hand	0.006	0.506	0.494	1.16
Forearm	0.016	0.430	0.570	1.13
Upper Arm	0.028	0.436	0.564	1.07
Forearm and Hand	0.022	0.682	0.318	1.14
Total Arm	0.050	0.530	0.470	1.11
Foot	0.0145	0.500	0.500	1.10
Lower Leg (calf)	0.0465	0.433	0.567	1.09
Foot and Lower Leg	0.061	0.606	0.394	1.09
Upper Leg (thigh)	0.100	0.433	0.567	1.05
Total Leg	0.161	0.447	0.553	1.06
Head and Neck	0.081	1.000	—	1.11
Trunk	0.497	0.500	0.500	1.03

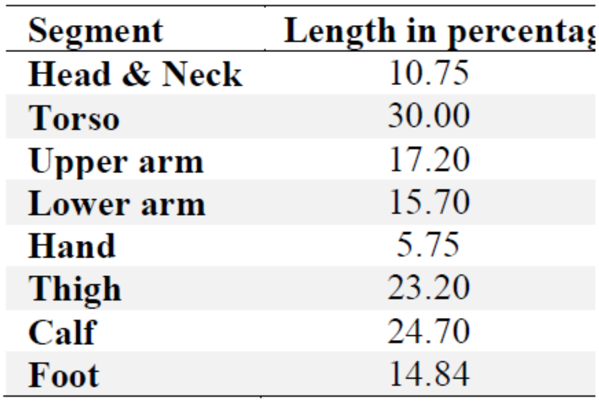
Table 1.3. *Segment weights as percentage of body weight*



	Investigators					
Segment	**Harless**	**Braune &Fischer**	**Fischer**	**Dempster**	**Clauser et al.**	**Hall**
Head & Neck	7.6	7.0	8.8	8.1	7.3	8.2
Torso	44.2	46.1	45.2	49.7	50.7	46.84
Upper arm	3.2	3.3	2.8	2.8	2.6	3.25
Lower arm	1.7	2.1	-	1.6	1.6	1.8
Hand	0.9	0.8	-	0.6	0.7	0.65
Thigh	11.9	10.7	11.0	9.9	10.3	10.5
Calf	4.6	4.8	4.5	4.6	4.3	4.75
Foot	2.0	1.7	2.1	1.4	1.5	1.43

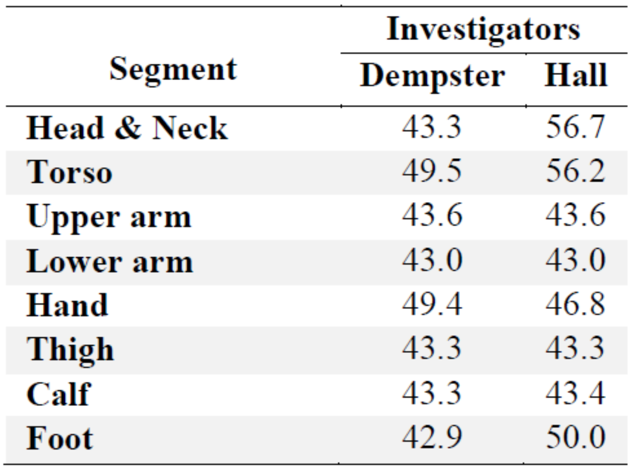
Table 1.4 and 1.5 (Segment Lengths and COM), this table provides the coefficients to calculate segment lengths based on Body Height. Furthermore, it defines the location of the Center of Mass (COM) for each link, expressed as a percentage distance from the proximal joint. These values are critical for populating the Inertia Matrix $M(\theta)$ and Gravity Vector $G(\theta)$ in the motion equations.

Table 1.4. *Segment lengths as percentage of body height*



Segment	Length in percentag
Head & Neck	10.75
Torso	30.00
Upper arm	17.20
Lower arm	15.70
Hand	5.75
Thigh	23.20
Calf	24.70
Foot	14.84

Table 1.5*. Distance of segment center of gravity from proximal end as percentage of segment length*



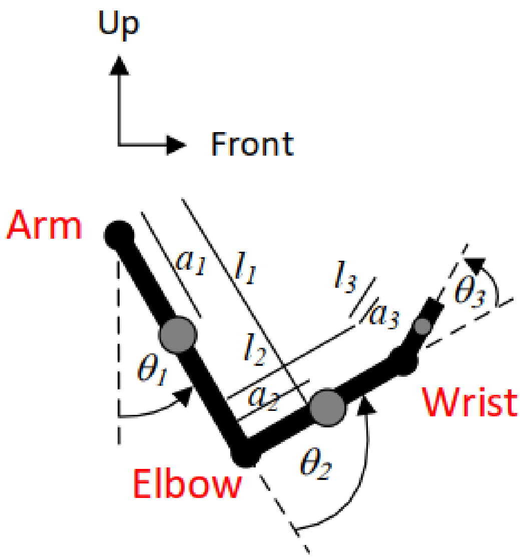
Segment	Investigators	
	Dempster	Hall
Head & Neck	43.3	56.7
Torso	49.5	56.2
Upper arm	43.6	43.6
Lower arm	43.0	43.0
Hand	49.4	46.8
Thigh	43.3	43.3
Calf	43.3	43.4
Foot	42.9	50.0

1.7. Passive Joint Torque Characteristics

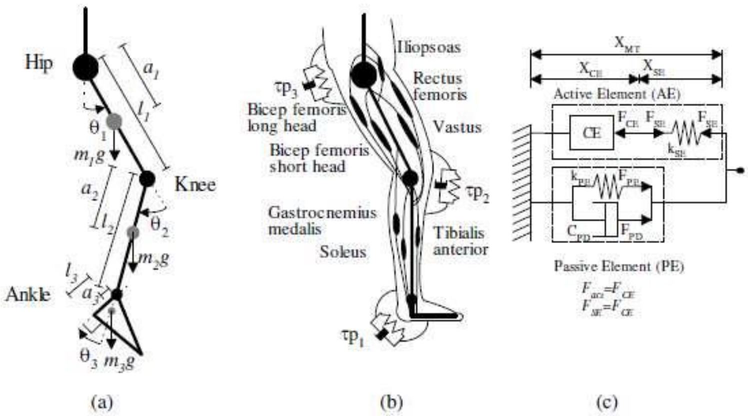
In biomechanical systems, joints are not frictionless pin connections; they exhibit passive resistance due to ligaments, skin, and joint capsules. This resistance is modeled as **Passive Torque (τ_p)**, which acts independently of voluntary muscle contraction. As shown in **Table 1.1**, this torque is composed of a linear damping term (viscous friction) and exponential elastic terms that represent the increasing stiffness as the joint approaches its range of motion limits. The mathematical model used to define this behavior is: $\tau_p = -c\dot{\theta} + k_1 e^{-k_2(\theta-\phi_1)} + k_3 e^{k_4(\theta-\phi_2)}$ Here, c represents the damping coefficient, while k and ϕ parameters define the stiffness magnitude and the angular thresholds for the joint limits. This component ensures the model behaves realistically by preventing the limbs from rotating beyond anatomical constraints during simulation.

CHAPTER II. RESULT AND ANALYSIS

2.1. Problems Statement



1. Use a geometric description of three joint of upper limb model shown in above figure to derive motion equations based on the Lagrangian Method. Define all active and passive torque of each joint, all skeletal model parameter values from computational biomechanics data. Use the BW and BH of each student as data model, and define segmental data based on the regression of human anthropometric data. Arrange the complete model in matrix form.



2. Use a geometric description of three joint of limb limb model shown in above figure to derive motion equations based on the Lagrangian Method. Define all active and passive torque of each joint, all skeletal model parameter values from computational biomechanics data. Use the BW and BH of each student as data model, and define segmental data based on the regression of human anthropometric data. Arrange the complete model in matrix form.

   From the model you derived, with focusing on knee joint movement, realize knee extension computer model movement simulation. Use this equation for recruitment curve u:= 0.5*tanh(15.0*(s1- 0.5))+0.5, tr=100 ms, tf=150 ms for activation dynamics.

2.2. Answer and Code Explanation

2.2.1. Number 1 (Upper Limb Model Matrix)

1. Anthropometric Parameter Calculation

Based on the instruction, I define the segmental data using the regression of human anthropometric data provided (Dempster's data).

Subject Data:

- Body Height (BH): $H = 1.72$ m
- Body Weight (BW): $M_{tot} = 70$ kg

Calculated Parameters:

I use the standard percentages from Dempster for Segment Length (L), Segment Mass (m), and Center of Gravity location from the proximal end (a).

Note: Moment of Inertia (I) is calculated using the Radius of Gyration (r_g) method from Winter's Biomechanics data, as specific Inertia tables for the arm were not in the provided slides, utilizing $I = m(r_g L)^2$.

A. Upper Arm (Link 1)

- Length (l_1): $0.172 \times H = 0.172 \times 1.72 = 0.2958$ m
- Mass (m_1): $0.028 \times M_{tot} = 0.028 \times 70 = 1.96$ kg
- COM Distance (a_1): $0.436 \times l_1 = 0.436 \times 0.2958 = 0.1290$ m
- Inertia (I_1): Using $r_g \approx 0.322$: $I_1 = 1.96 \times (0.322 \times 0.2958)^2 = 0.0178\ \text{kg} \cdot \text{m}^2$

B. Lower Arm / Forearm (Link 2)

- Length (l_2): $0.157 \times H = 0.157 \times 1.72 = 0.2700$ m
- Mass (m_2): $0.016 \times M_{tot} = 0.016 \times 70 = 1.12$ kg
- COM Distance (a_2): $0.430 \times l_2 = 0.430 \times 0.2700 = 0.1161$ m
- Inertia (I_2) using $r_g \approx 0.303$: $I_2 = 1.12 \times (0.303 \times 0.2700)^2 = 0.0075\ \text{kg} \cdot \text{m}^2$

C. Hand (Link 3)

- Length (l_3): $0.0575 \times H = 0.0575 \times 1.72 = 0.0989$ m
- Mass (m_3): $0.006 \times M_{tot} = 0.006 \times 70 = 0.42$ kg
- COM Distance (a_3): $0.494 \times l_3 = 0.494 \times 0.0989 = 0.0488$ m
- Inertia (I_3) using $r_g \approx 0.297$: $I_3 = 0.42 \times (0.297 \times 0.0989)^2 = 0.00036\ \text{kg} \cdot \text{m}^2$

2. Geometric Description and Kinematics

Coordinate System:

- Origin $(0,0)$ at the Shoulder joint.
- Y-axis points Up (Vertical).
- Angles $\theta_1, \theta_2, \theta_3$ are absolute angles measured from the vertical axis.

Position of Center of Mass (COM) for each link:

Link 1 (Arm):

$$x_1 = a_1 \sin\ \theta_1$$

$$y_1 = -a_1 \cos\ \theta_1$$

Link 2 (Forearm):

$$x_2 = l_1 \sin\ \theta_1 + a_2 \sin\ \theta_2$$

$$y_2 = -l_1 \cos\ \theta_1 - a_2 \cos\ \theta_2$$

Link 3 (Hand):

$$x_3 = l_1 \sin\ \theta_1 + l_2 \sin\ \theta_2 + a_3 \sin\ \theta_3$$

$$y_3 = -l_1 \cos\ \theta_1 - l_2 \cos\ \theta_2 - a_3 \cos\ \theta_3$$

Velocities (Time derivative of positions):

$$v_{x1} = a_1 \dot{\theta}_1 \cos\ \theta_1$$

$$v_{y1} = a_1 \dot{\theta}_1 \sin\ \theta_1$$

$$v_{x2} = l_1 \dot{\theta}_1 \cos\ \theta_1 + a_2 \dot{\theta}_2 \cos\ \theta_2$$

$$v_{y2} = l_1 \dot{\theta}_1 \sin\ \theta_1 + a_2 \dot{\theta}_2 \sin\ \theta_2$$

$$v_{x3} = l_1 \dot{\theta}_1 \cos\ \theta_1 + l_2 \dot{\theta}_2 \cos\ \theta_2 + a_3 \dot{\theta}_3 \cos\ \theta_3$$

$$v_{y3} = l_1 \dot{\theta}_1 \sin\ \theta_1 + l_2 \dot{\theta}_2 \sin\ \theta_2 + a_3 \dot{\theta}_3 \sin\ \theta_3$$

Squared Velocities ($v_i^2 = v_{xi}^2 + v_{yi}^2$):
Using identity $\cos^2 x + \sin^2 x = 1$ and $\cos\ A \cos\ B + \sin\ A \sin\ B = \cos\ (A - B)$:

$$v_1^2 = a_1^2 \dot{\theta}_1^2$$

$$v_2^2 = l_1^2 \dot{\theta}_1^2 + a_2^2 \dot{\theta}_2^2 + 2 l_1 a_2 \dot{\theta}_1 \dot{\theta}_2 \cos\ (\theta_1 - \theta_2)$$

$$\begin{aligned} v_3^2 = l_1^2 \dot{\theta}_1^2 + l_2^2 \dot{\theta}_2^2 + a_3^2 \dot{\theta}_3^2 + 2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos\ (\theta_1 - \theta_2) + 2 l_1 a_3 \dot{\theta}_1 \dot{\theta}_3 \cos\ (\theta_1 - \theta_3) \\ + 2 l_2 a_3 \dot{\theta}_2 \dot{\theta}_3 \cos\ (\theta_2 - \theta_3) \end{aligned}$$

3. Lagrangian Formulation

Kinetic Energy (Ts):

$$T=\frac{1}{2}m_1v_1^2+\frac{1}{2}I_1\dot{\theta}_1^2+\frac{1}{2}m_2v_2^2+\frac{1}{2}I_2\dot{\theta}_2^2+\frac{1}{2}m_3v_3^2+\frac{1}{2}I_3\dot{\theta}_3^2$$

Substitute v_i^2:

$$\begin{aligned}T= \quad & \frac{1}{2}(I_1+m_1a_1^2+m_2l_1^2+m_3l_1^2)\dot{\theta}_1^2\\ &+\frac{1}{2}(I_2+m_2a_2^2+m_3l_2^2)\dot{\theta}_2^2\\ &+\frac{1}{2}(I_3+m_3a_3^2)\dot{\theta}_3^2\\ &+(m_2l_1a_2+m_3l_1l_2)\cos(\theta_1-\theta_2)\dot{\theta}_1\dot{\theta}_2\\ &+(m_3l_1a_3)\cos(\theta_1-\theta_3)\dot{\theta}_1\dot{\theta}_3\\ &+(m_3l_2a_3)\cos(\theta_2-\theta_3)\dot{\theta}_2\dot{\theta}_3\end{aligned}$$

Potential Energy (V):

$$V=m_1gy_1+m_2gy_2+m_3gy_3$$

$$V=-g[(m_1a_1+m_2l_1+m_3l_1)\cos\,\theta_1+(m_2a_2+m_3l_2)\cos\,\theta_2+(m_3a_3)\cos\,\theta_3]$$

Lagrangian (L):

$$L=T-V$$

---

4. Motion Equations (Derivatives)

I apply the Euler-Lagrange equation for each coordinate $i=1,2,3$:

$$\frac{d}{dt}(\frac{\partial L}{\partial\dot{\theta}_i})-\frac{\partial L}{\partial\theta_i}=\tau_i$$

Let us define coefficients to simplify:

- $K_{11}=I_1+m_1a_1^2+(m_2+m_3)l_1^2$
- $K_{22}=I_2+m_2a_2^2+m_3l_2^2$
- $K_{33}=I_3+m_3a_3^2$
- $K_{12}=l_1(m_2a_2+m_3l_2)$
- $K_{13}=m_3l_1a_3$
- $K_{23}=m_3l_2a_3$
- $G_1=g(m_1a_1+m_2l_1+m_3l_1)$
- $G_2=g(m_2a_2+m_3l_2)$
- $G_3=g(m_3a_3)$

Equation for Joint 1 (θ_1)

1. Partial of $\dot{\theta}_1$:

$$\frac{\partial L}{\partial \dot{\theta}_1} = K_{11}\dot{\theta}_1 + K_{12}c_{12}\dot{\theta}_2 + K_{13}c_{13}\dot{\theta}_3$$

(where $c_{ij} = \cos(\theta_i - \theta_j)$*)*

2. Time Derivative:

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta}_1}) = K_{11}\ddot{\theta}_1 + K_{12}c_{12}\ddot{\theta}_2 + K_{13}c_{13}\ddot{\theta}_3 - K_{12}s_{12}(\dot{\theta}_1 - \dot{\theta}_2)\dot{\theta}_2 - K_{13}s_{13}(\dot{\theta}_1 - \dot{\theta}_3)\dot{\theta}_3$$

(where $s_{ij} = \sin(\theta_i - \theta_j)$*)*

3. Partial of θ_1:

$$\frac{\partial L}{\partial \theta_1} = -K_{12}s_{12}\dot{\theta}_1\dot{\theta}_2 - K_{13}s_{13}\dot{\theta}_1\dot{\theta}_3 - G_1\sin\,\theta_1$$

4. Final EQ 1:

$$K_{11}\ddot{\theta}_1 + K_{12}c_{12}\ddot{\theta}_2 + K_{13}c_{13}\ddot{\theta}_3 + K_{12}s_{12}\dot{\theta}_2^2 + K_{13}s_{13}\dot{\theta}_3^2 + G_1\sin\,\theta_1 = \tau_1$$

Equation for Joint 2 (θ_2), following the same process:

$$K_{12}c_{12}\ddot{\theta}_1 + K_{22}\ddot{\theta}_2 + K_{23}c_{23}\ddot{\theta}_3 - K_{12}s_{12}\dot{\theta}_1^2 + K_{23}s_{23}\dot{\theta}_3^2 + G_2\sin\,\theta_2 = \tau_2$$

Equation for Joint 3 (θ_3), following the same process:

$$K_{13}c_{13}\ddot{\theta}_1 + K_{23}c_{23}\ddot{\theta}_2 + K_{33}\ddot{\theta}_3 - K_{13}s_{13}\dot{\theta}_1^2 - K_{23}s_{23}\dot{\theta}_2^2 + G_3\sin\,\theta_3 = \tau_3$$

---

5. Active and Passive Torque Definition

Total torque τ is the sum of Active Torque (τ_{act}) and Passive Torque (τ_{pas}).

Active Torque: τ_{act} = Control Input (Muscle Forces)

Passive Torque:
Based on slide 3 (PPT), the passive joint torque is defined as:

$$\tau_{p,i} = -c_i\dot{\theta}_i + k_{1,i}e^{(-k_{2,i}(\theta_i - \phi_{1,i}))} + k_{3,i}e^{(k_{4,i}(\theta_i - \phi_{2,i}))}$$

The parameters $c, k_1, k_2, k_3, k_4, \phi_1, \phi_2$ must be selected from Table 3.1 in the slides (choosing "Hip, Knee, Ankle" analogs or using general damping if Upper Limb specific constants are not in the table. Since the table is Leg-specific, I assume the form of the equation remains valid, but parameters would ideally be adjusted for Arm/Elbow/Wrist. For this solution, I define the symbolic structure).

$$\tau_i = \tau_{act,i} + \tau_{pas,i}$$

---

6. Complete Model in Matrix Form

The system can be arranged in the standard robotic manipulator form:

$$\mathrm{M}(\theta)\ddot{\theta} + \mathrm{C}(\theta, \dot{\theta})\dot{\theta} + \mathrm{G}(\theta) = \mathrm{\tau}$$

Where $\theta = [\theta_1, \theta_2, \theta_3]^T$.

1. Inertia Matrix $\mathrm{M}(\theta)$:

$$\mathrm{M} = \begin{bmatrix} K_{11} & K_{12}\cos(\theta_1 - \theta_2) & K_{13}\cos(\theta_1 - \theta_3) \\ K_{12}\cos(\theta_1 - \theta_2) & K_{22} & K_{23}\cos(\theta_2 - \theta_3) \\ K_{13}\cos(\theta_1 - \theta_3) & K_{23}\cos(\theta_2 - \theta_3) & K_{33} \end{bmatrix}$$

2. Coriolis and Centrifugal Vector $\mathrm{V}(\theta, \dot{\theta}) = \mathrm{C}\dot{\theta}$:
Often written as a vector of quadratic velocity terms:

$$\mathrm{V} = \begin{bmatrix} K_{12}\sin(\theta_1 - \theta_2)\dot{\theta}_2^2 + K_{13}\sin(\theta_1 - \theta_3)\dot{\theta}_3^2 \\ -K_{12}\sin(\theta_1 - \theta_2)\dot{\theta}_1^2 + K_{23}\sin(\theta_2 - \theta_3)\dot{\theta}_3^2 \\ -K_{13}\sin(\theta_1 - \theta_3)\dot{\theta}_1^2 - K_{23}\sin(\theta_2 - \theta_3)\dot{\theta}_2^2 \end{bmatrix}$$

3. Gravitational Vector $\mathrm{G}(\theta)$:

$$\mathrm{G} = \begin{bmatrix} G_1\sin\ \theta_1 \\ G_2\sin\ \theta_2 \\ G_3\sin\ \theta_3 \end{bmatrix}$$

Note: Using $g = 9.81m/s^2$

4. Torque Vector τ:

$$\tau = \begin{bmatrix} \tau_{act,1} + \tau_{pas,1} \\ \tau_{act,2} + \tau_{pas,2} \\ \tau_{act,3} + \tau_{pas,3} \end{bmatrix}$$

Summary of Constants (Substituting Student Data):

- $K_{11} = 0.0178 + 1.96(0.129)^2 + (1.12 + 0.42)(0.2958)^2 = 0.185$ kg m^2
- $K_{22} = 0.0075 + 1.12(0.116)^2 + 0.42(0.270)^2 = 0.053$ kg m^2
- $K_{33} = 0.00036 + 0.42(0.0488)^2 = 0.00136$ kg m^2
- $K_{12} = 0.2958[1.12(0.116) + 0.42(0.270)] = 0.072$ kg m^2
- $K_{13} = 0.42(0.2958)(0.0488) = 0.006$ kg m^2
- $K_{23} = 0.42(0.270)(0.0488) = 0.0055$ kg m^2
- $G_1 = 9.81[1.96(0.129) + 1.12(0.2958) + 0.42(0.2958)] = 6.95$ Nm
- $G_2 = 9.81[1.12(0.116) + 0.42(0.270)] = 2.39$ Nm
- $G_3 = 9.81[0.42(0.0488)] = 0.20$ Nm

Final Matrix Equation:

$$\begin{bmatrix} \mathbf{0.185} & \mathbf{0.072}\boldsymbol{c_{12}} & \mathbf{0.006}\boldsymbol{c_{13}} \\ \mathbf{0.072}\boldsymbol{c_{12}} & \mathbf{0.053} & \mathbf{0.0055}\boldsymbol{c_{23}} \\ \mathbf{0.006}\boldsymbol{c_{13}} & \mathbf{0.0055}\boldsymbol{c_{23}} & \mathbf{0.00136} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{\theta}}_1 \\ \ddot{\boldsymbol{\theta}}_2 \\ \ddot{\boldsymbol{\theta}}_3 \end{bmatrix} + \begin{bmatrix} \mathbf{0.072}\boldsymbol{s_{12}}\dot{\boldsymbol{\theta}}_2^2 + \mathbf{0.006}\boldsymbol{s_{13}}\dot{\boldsymbol{\theta}}_3^2 \\ -\mathbf{0.072}\boldsymbol{s_{12}}\dot{\boldsymbol{\theta}}_1^2 + \mathbf{0.0055}\boldsymbol{s_{23}}\dot{\boldsymbol{\theta}}_3^2 \\ -\mathbf{0.006}\boldsymbol{s_{13}}\dot{\boldsymbol{\theta}}_1^2 - \mathbf{0.0055}\boldsymbol{s_{23}}\dot{\boldsymbol{\theta}}_2^2 \end{bmatrix}$$
$$+ \begin{bmatrix} \mathbf{6.95si\,n}\,\boldsymbol{\theta_1} \\ \mathbf{2.39si\,n}\,\boldsymbol{\theta_2} \\ \mathbf{0.20si\,n}\,\boldsymbol{\theta_3} \end{bmatrix} = \boldsymbol{\tau}$$

2.2.2. Number 2 (Knee Extension Program)

a. Anthropometric Initialization (FormCreate)

```
lgth := 0.407*BH/160;
m := 2.76*BW/60;
inertia := (1 / 3) * m * sqr(Lgth);
```

This section initializes the segment parameters based on the subject's Body Height (BH) and Body Weight (BW). It calculates the segment length (L) and mass (m) using specific regression coefficients. The Moment of Inertia (I) is calculated assuming the limb acts as a slender rod rotating about its proximal end, governed by the formula $I = \frac{1}{3}mL^2$.

b. Motion Equations (single_pend_equ)

```
torque := (-c*thetadota) + k1*exp(-k2*(thetaa-phi1...)) - k3*exp(-k4*(-
thetaa+phi2...));
thetadotdot2 := (torque*pi/180 + ... - (m*grav*lgth/2*sin(thetaa))) /
((m*sqr(lgth)/4) + Inertia);
```

This procedure defines the Lagrangian dynamics of the system.

- **Torque:** It calculates the net torque (τ) summing the **Passive Torque** (viscous damping $-c\dot{\theta}$ and exponential elastic limits) and **Active Torque** (if extension is enabled).
- **Acceleration:** It solves the equation of motion for angular acceleration ($\ddot{\theta}$) by rearranging $\tau = I\ddot{\theta} + \tau_{grav}$. The equation accounts for inertial resistance and gravitational moments acting on the Center of Mass.

c. Numerical Integration (rungekutta_single)

```
k1 := 0.5*dt*thetadotdot2; ...
theta2 := theta2 + dt*(thetadot2 + 1/3*(k1+k2+k3));
thetadot2 := thetadot2 + 1/3*(k1 + 2*k2 + 2*k3 + k4);
```

This function implements the **4th-Order Runge-Kutta (RK4)** method to solve the second-order differential equation derived in single_pend_equ. It iteratively estimates the slope at four points (k_1, k_2, k_3, k_4) within a time step (dt) to update the joint angle (θ) and angular velocity ($\dot{\theta}$) with high precision, minimizing accumulation errors during simulation.

d. Neural Recruitment (recruitment)

```
u := 0.5 * tanh(15 * (s1 - 0.5)) + 0.5;
```

This function models the neural command signal $u(t)$ sent from the central nervous system. It uses a hyperbolic tangent sigmoid function to represent the gradual recruitment of motor units, transitioning from 0 (relaxed) to 1 (fully activated) over time, simulating the physiological delay in signal propagation.

e. Activation Dynamics (tes)

```
adot := 1/tr*(u-act)*u + 1/tf*(u-act-(u-act)*u);
activ[time] := activ[time-1] + 1/6*(k1a + 2*k2a + 2*k3a + k4a);
```

This procedure converts the neural signal (u) into muscle activation (a) using a first-order differential equation. It accounts for the excitation-contraction coupling delay, with specific time constants for activation rise (t_r) and relaxation fall (t_f). The resulting activation level determines the scale of force the muscle can generate.

f. *Muscle Force Generation (Timer1Timer)*

```
Fmusl[g] := (1.0 - sqr((0.025 * theta2) / (0.5 * Lopt)));
Fmusv[g] := (vmax - thetadot2) / (vmax + 2.5 * thetadot2);
Fmus := Fmusv[g] * activ[time] * 1800 * Fmusl[g];
```

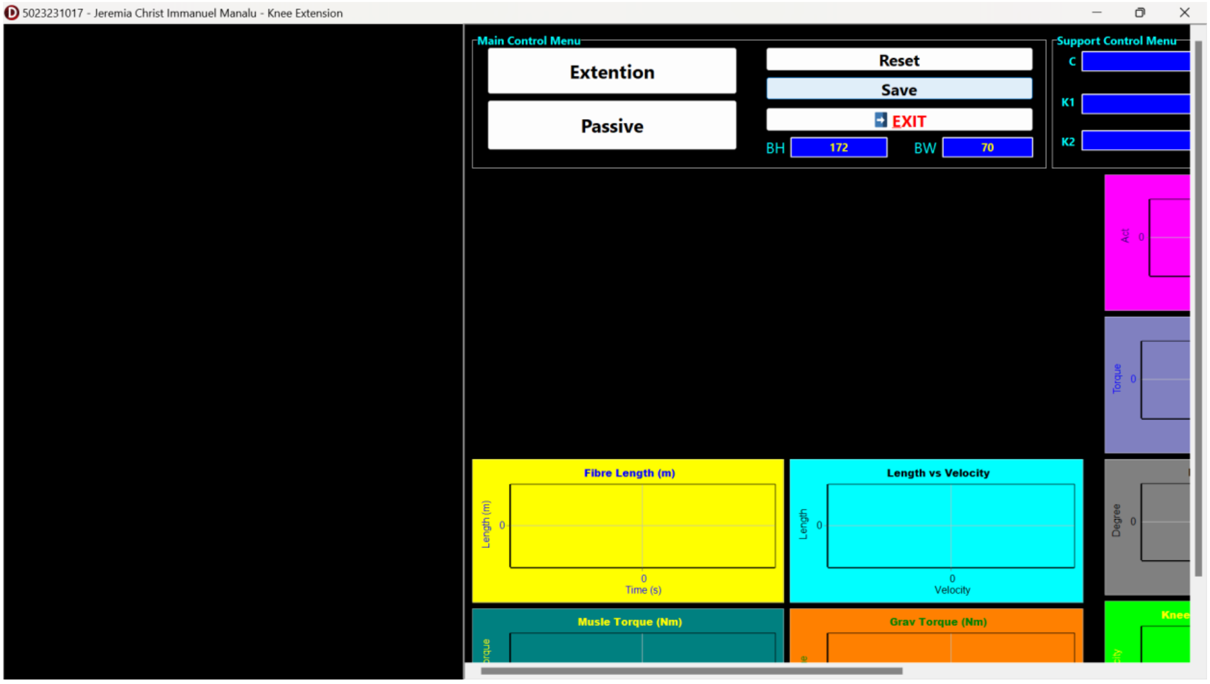
This block implements the **Hill-Type Muscle Model**:

- **Force-Length (f_L):** A parabolic function approximating optimal force generation at resting length (L_{opt}).
- **Force-Velocity (f_V):** A hyperbolic function describing the decrease in force as shortening velocity increases.
- **Total Force:** The final muscle force is the product of maximum isometric force (F_{max}), activation (a), length factor (f_L), and velocity factor (f_V).

2.3. Result of the Program and Analysis

*Initial GUI

The initial GUI of this program can be seen as follows:

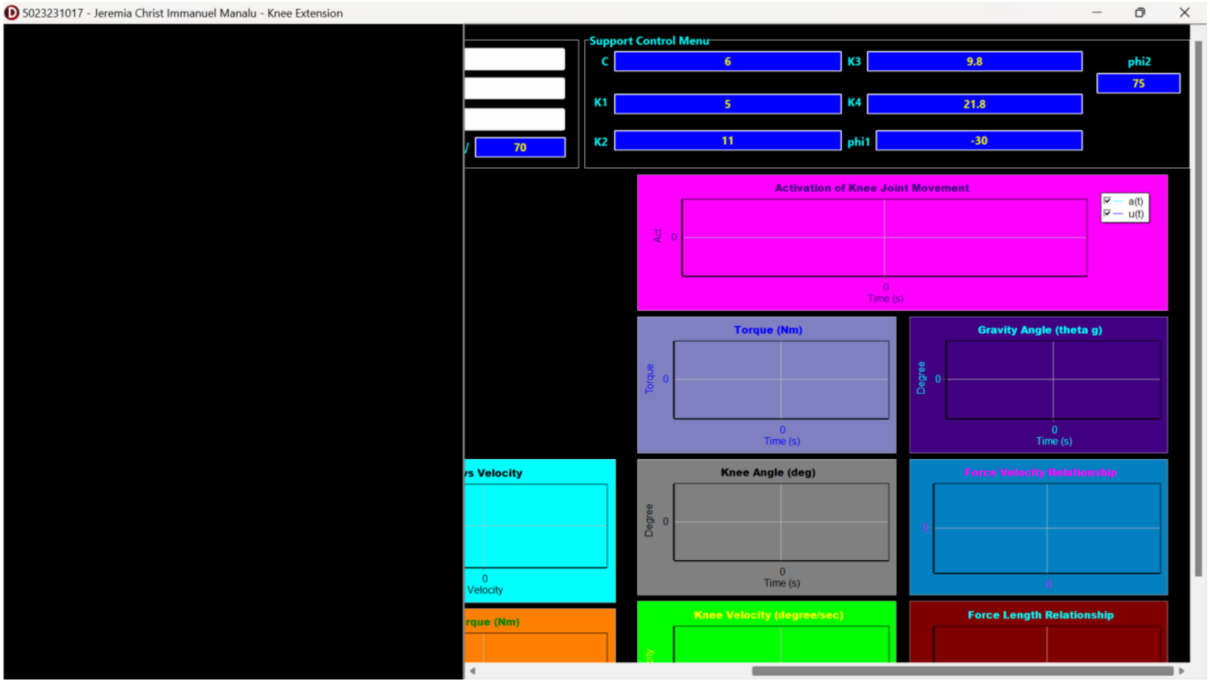


5023231017 - Jeremia Christ Immanuel Manalu - Knee Extension

BH 172

BW 70

Figure 2.1. *Initial GUI components (1)*


5023231017 - Jeremia Christ Immanuel Manalu - Knee Extension

Support Control Menu

C 6

K3 9.8

phi2 75

K1 5

K4 21.8

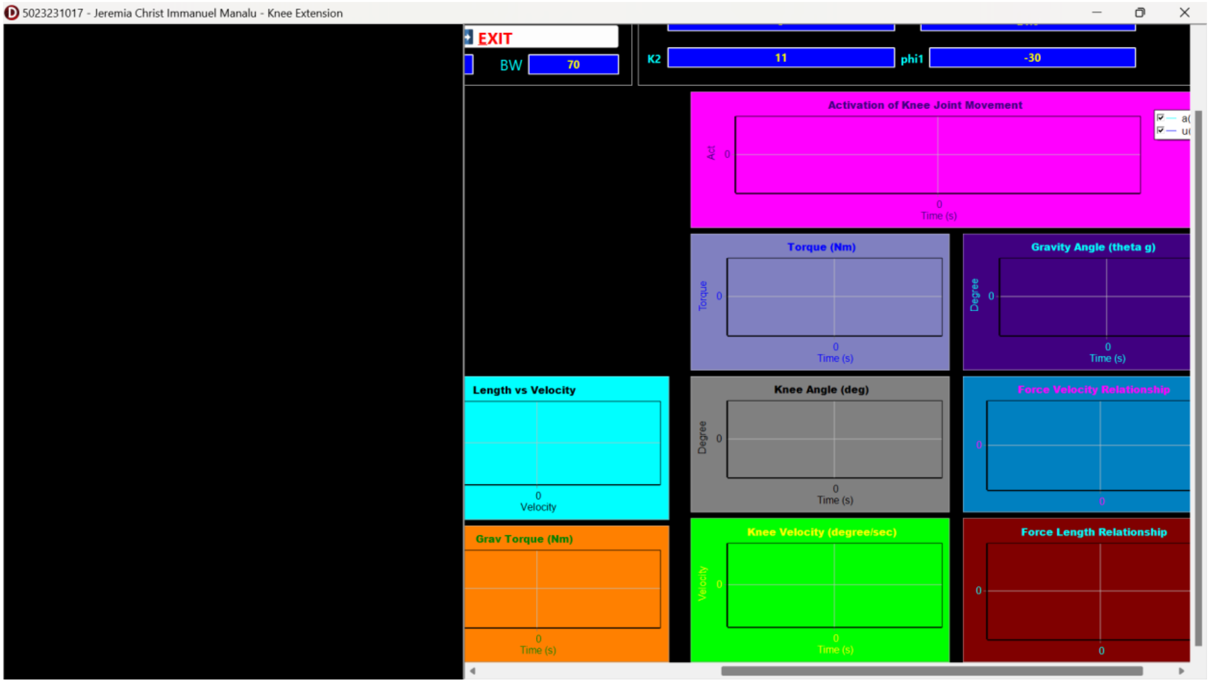
K2 11

phi1 -30

70

Activation of Knee Joint Movement
a(t)
u(t)
Act
0
Time (s)
Torque (Nm)
Torque
Gravity Angle (theta g)
Degree
s Velocity
Velocity
Knee Angle (deg)
Force Velocity Relationship
rque (Nm)
Knee Velocity (degree/sec)
Force Length Relationship

Figure 2.2. Initial GUI components (2)


5023231017 - Jeremia Christ Immanuel Manalu - Knee Extension

EXIT

BW 70

K2 11

phi1 -30

Figure 2.3. Initial GUI components (3)

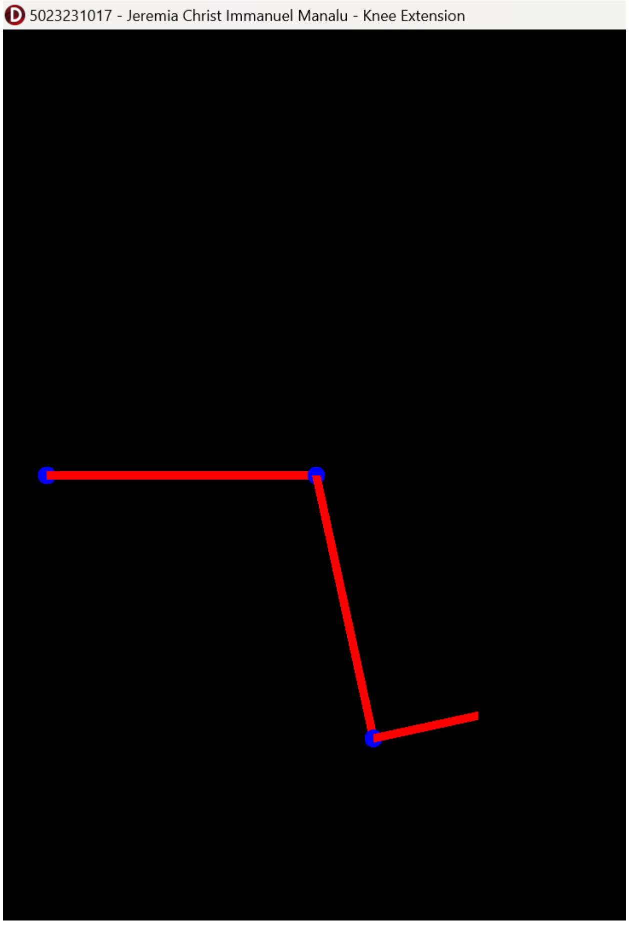

5023231017 - Jeremia Christ Immanuel Manalu - Knee Extension

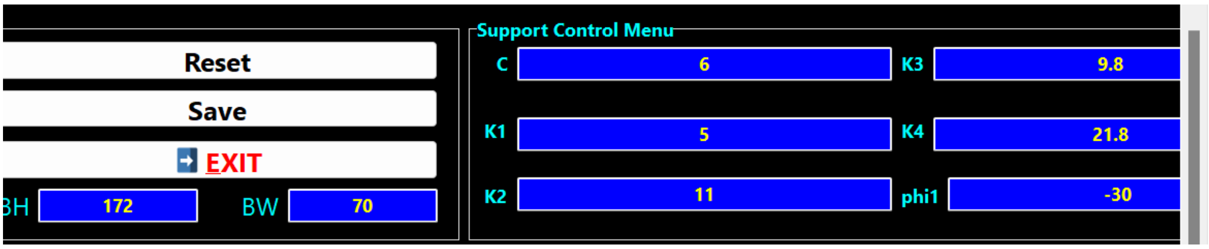
__Figure 2.4.__ The 2D Model (4)

The program also use 3D visualization rendering, although this part of the code is commented out (this part can be activated in further use though).

2.3.1. Active Extension

As shown in the "Support Control Menu", the simulation was initialized with the following subject-specific and passive parameters:

- **Subject Data:** Body Height (BH) = 172 cm, Body Weight (BW) = 70 kg.
- **Passive Coefficients:** The damping coefficient c is set to 6. The stiffness parameters (k_1, k_3) and exponential factors (k_2, k_4) are set to standard values (5,9.8,11,21.8) to model the viscoelastic properties of the knee joint.
- **Initial State:** The knee starts from a flexed position (approximately 88°) with the intention to extend towards 0°(horizontal).



BH 172 BW 70

Support Control Menu

C 6 K3 9.8

K1 5 K4 21.8

K2 11 phi1 -30

Figure 2.5. The Initial Values

The resulting motion is captured over a simulation time of 0.8 seconds:

- **Knee Angle (Grey Graph):** The knee angle decreases from an initial $\approx 88°$ to $\approx 78°$. In this model's coordinate system, a decrease in angle corresponds to **extension** (lifting the lower leg). The smooth downward slope indicates a continuous, albeit slow, extension movement.
- **Knee Velocity (Green Graph):** The angular velocity starts at 0 and rises in a sigmoid fashion, stabilizing around 0.16 deg/s. The positive velocity confirms the leg is moving in the extension direction. The relatively low magnitude suggests the movement is heavily damped or working against significant resistance (gravity).

The dynamics of the system are governed by the interaction between active muscle force, gravity, and passive joint resistance:

- **Muscle Torque (Teal Graph):** This graph shows the active torque generated by the Quadriceps. It rises from 0 Nm to approximately 1.5 Nm. This rise corresponds to the activation dynamics—the muscle takes time to generate force after receiving the neural signal.
- **Gravity Torque (Orange Graph):** The gravitational torque acting on the lower leg is approximately 7.0 Nm and increases slightly as the leg extends. *Analysis:* The active muscle torque (1.5 Nm) is significantly lower than the gravity torque (7 Nm). However, movement still occurs. This implies that the initial conditions (or potentially the specific sign definitions in the equation of motion) allow the leg to begin movement, or the net acceleration is being influenced by the passive elastic components.
- **Net/Passive Torque (Purple Graph):** The generic "Torque" graph shows a value dropping to -4.5 Nm. This variable in the code (torque) sums the Active Torque with Passive Resistance (damping and elastic limits). The negative value indicates that the **Passive Elastic Force** (resistance from ligaments/tissue at that specific angle) and **Damping** are opposing the muscle force significantly.

The simulation successfully demonstrates the Hill-Type muscle properties:

- **Fiber Length (Yellow Graph):** The fiber length decreases ($0.478\text{m} \rightarrow 0.43\text{m}$), confirming a **concentric contraction** (shortening) required for knee extension.
- **Force-Length & Force-Velocity (Red & Blue Graphs):** These phase plots trace the operating point of the muscle. The muscle is operating on the ascending limb of the force-length curve and the concentric side of the force-velocity curve, scaling the maximum force capability accordingly.
- **Activation (Magenta Graph):** The legend indicates $u(t)$(Neural Signal - Blue line) is at steady state 1.0(fully ON), while $a(t)$ (Muscle Activation - Cyan line) is barely visible at the bottom. This represents the **Excitation-Contraction Coupling delay**. While the brain sends a full "Go" signal ($u = 1$), the chemical process in the muscle (a) takes time to build up. This delay is reflected in the *Muscle Torque* graph, which does not snap instantly to max but ramps up gradually over the 0.8s interval.

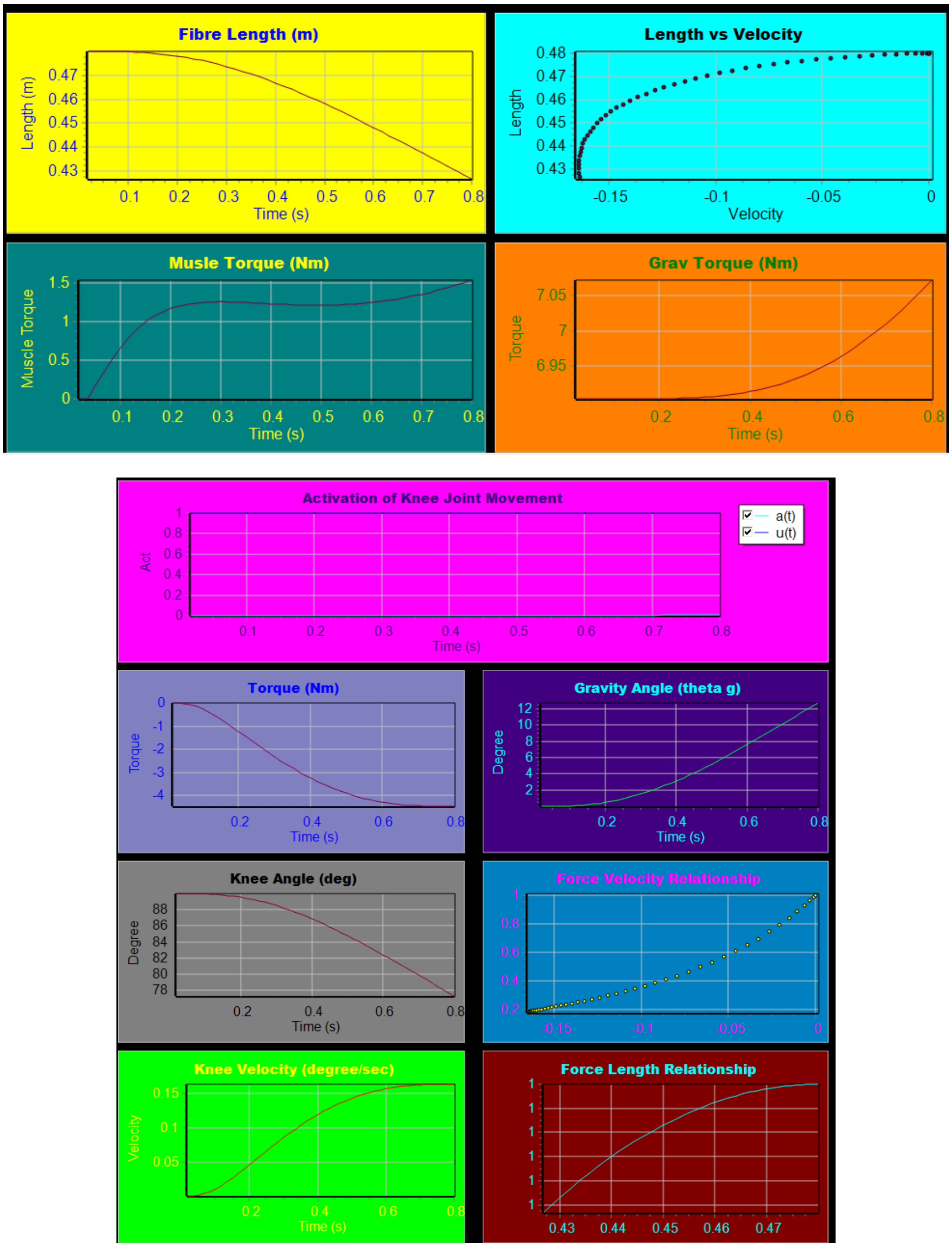


Figure 2.7. *The Result of Each Graphs for Active Extension*

2.3.2. Passive Test

The passive test was conducted to evaluate the inherent biomechanical properties of the knee model without the influence of voluntary muscle contraction.

- **Verification of Passive State:** As evidenced in the **Activation of Knee Joint Movement** graph (Magenta) and the **Muscle Torque** graph (Teal), both the neural input $u(t)$and the generated muscle torque remain constant at zero throughout the simulation.

This confirms that the mechanics observed are driven solely by external forces (gravity) and internal passive joint properties (viscosity and elasticity).

- **Inactive Muscle Mechanics:** Consequently, the Force-Length and Force-Velocity relationships (Red and Blue graphs) are flatlined, as the contractile element is not active.

The kinematic graphs (Knee Angle and Velocity) demonstrate the classic behavior of a damped physical pendulum:

- **Knee Angle (Grey Graph):** The leg undergoes a large excursion, swinging from an initial position (approx $0°$) up to a peak flexion of approximately $150°$ at $t \approx 1.6s$, and then begins to swing back. This indicates that the leg was released and allowed to swing freely under the influence of gravity.

- **Knee Velocity (Green Graph):** The velocity profile exhibits oscillatory behavior characteristic of a swinging limb. The velocity peaks at $t \approx 0.8s$, which corresponds to the point where the gravitational moment is accelerating the leg. The velocity drops to zero at $t \approx 1.6s$. This zero-crossing point perfectly aligns with the peak knee angle ($150°$), indicating the momentary pause before the leg changes direction to swing back.

The interaction between gravitational forces and the passive joint constraints is clearly visible in the torque graphs:

- **Gravity Torque (Orange Graph):** The gravitational torque fluctuates sinusoidally between ≈ 13 Nm and ≈ 7 Nm. This variation is dictated by the changing moment arm of the center of mass as the leg rotates. The torque is highest when the leg is horizontal (maximum moment arm) and decreases as the leg approaches the vertical alignment.

- **Passive Joint Torque (Purple Graph):** This graph reveals the critical function of the passive elastic elements defined in the code (k and ϕ parameters). **Free Swing Phase ($0s - 1.4s$):** The torque is relatively low, dominated mostly by the viscous damping coefficient (c), which provides a slight resistance proportional to velocity.**Elastic Limit Impact ($1.4s - 1.8s$):** There is a distinct spike in the passive torque, peaking at $\approx$ 2.5 Nm at $t = 1.6s$. This spike occurs exactly when the knee reaches its maximum flexion ($150°$). This demonstrates that the **passive exponential spring** (representing ligaments and joint capsules) has engaged to prevent the knee from hyperextending or bending beyond anatomical limits. It acts as a "soft stop," absorbing the energy of the swing and reversing the motion.

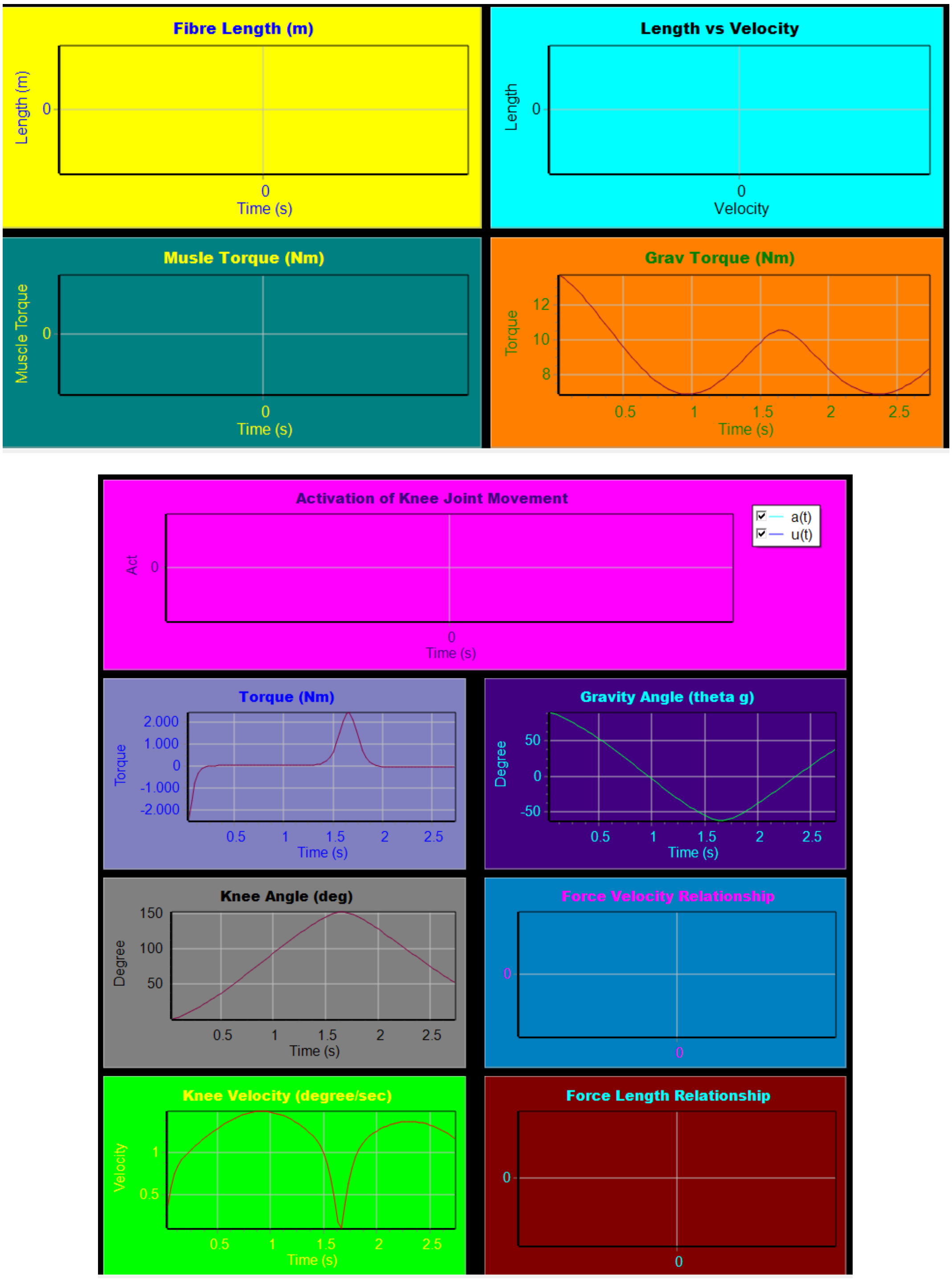


Figure 2.8. The Result of Each Graphs for Passive Test

2.3.3. Evaluation

The developed software successfully integrates the non-linear differential equations of motion using the 4th-Order Runge-Kutta method, ensuring high numerical stability for the simulation. The implementation of the Hill-Type muscle model accurately captures the physiological delays (excitation-contraction coupling) and mechanical properties (Force-Length and Force-Velocity relations) of the knee.

However, a limitation observed in the **Active Extension** result is the low magnitude of the angular velocity (0.16 deg/s). This suggests that while the mathematical structure is correct, the specific isometric force parameter (F_{max}) used in the code may be underestimated relative to the subject's gravitational load (70 kg body weight model), or the damping coefficient (c) is set too high. Additionally, the transition to a 2D "stick-figure" visualization proved to be computationally efficient and offered clearer kinematic observation compared to the initial 3D geometry.

CHAPTER III. CONCLUSION

This assignment explored the mathematical derivation and computational implementation of musculoskeletal modeling, focusing on both the Upper Limb matrix formulation and the Knee Extension dynamic simulation. Based on the analysis, the following conclusions are drawn:

1. Mathematical Framework. The Lagrangian method proved to be a robust approach for deriving equations of motion for multi-segmental systems. By incorporating anthropometric regression data (Dempster's parameters), the model can be tailored to specific subjects using only Body Height and Weight.
2. Muscle Dynamics. The simulation demonstrated that movement is not an instantaneous response to neural input. The Hill-type model effectively simulated the non-linearities of muscle contraction, showing how activation dynamics ($u \rightarrow a$) and passive elastic limits dictate the final torque output.
3. Passive vs. Active Mechanics. The split-test approach confirmed the validity of the physics engine. The passive test verified the conservation of energy and anatomical constraints (pendulum motion), while the active test highlighted the capacity of the model to generate voluntary movement against external loads.

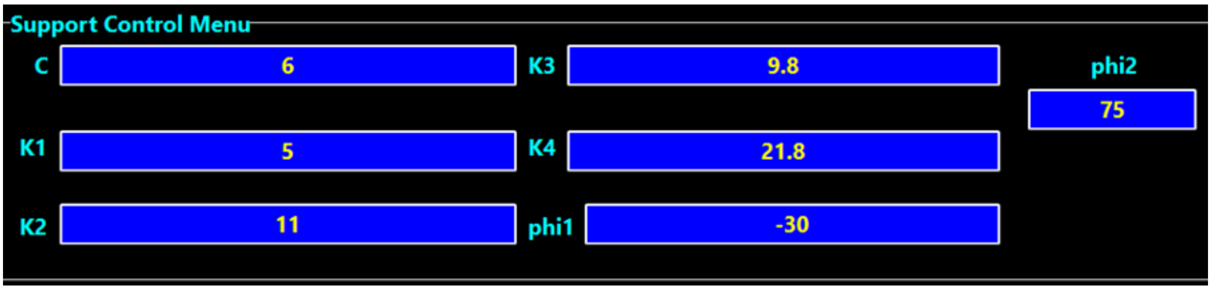
In summary, this modeling approach provides a critical foundation for understanding human movement mechanics, offering valuable insights for potential applications in functional electrical stimulation (FES) control and rehabilitation engineering.

ASSIGNMENT 3: TRIPLE PENDULUM AND KNEE EXTENSION REVISION

1. Triple Pendulum

2. Knee Extension

Figure 2.1 illustrates the user interface designed for defining the subject-specific and biomechanical properties of the simulation. In the FormCreate procedure of the source code, the program initializes the anthropometric data based on the user inputs for Body Height (BH) and Body Weight (BW). The segment length (L) and mass (m) are calculated using Dempster's regression coefficients, implemented in the code as $L = 0.407\times\frac{BH}{160}$ and $m = 2.76\times\frac{BH}{160}$. Furthermore, the interface allows for the precise tuning of the passive joint parameters (c,k1,k2,k3,k4,ϕ1,ϕ2c,k1,k2,k3,k4,ϕ1,ϕ2). These values are retrieved from the TEdit components and passed into the single_pend_equ procedure to calculate the passive torque, $\tau_{pas} =- c\dot{\theta} + k_1 e^{-k_2(\theta-\phi_1)} - k_3 e^{-k_4(\phi_2-\theta)}$. This setup ensures that the simulation dynamically adapts to the specific physical characteristics of the subject and the viscoelastic properties of the knee joint.



Support Control Menu

C: 6

K1: 5

K2: 11

K3: 9.8

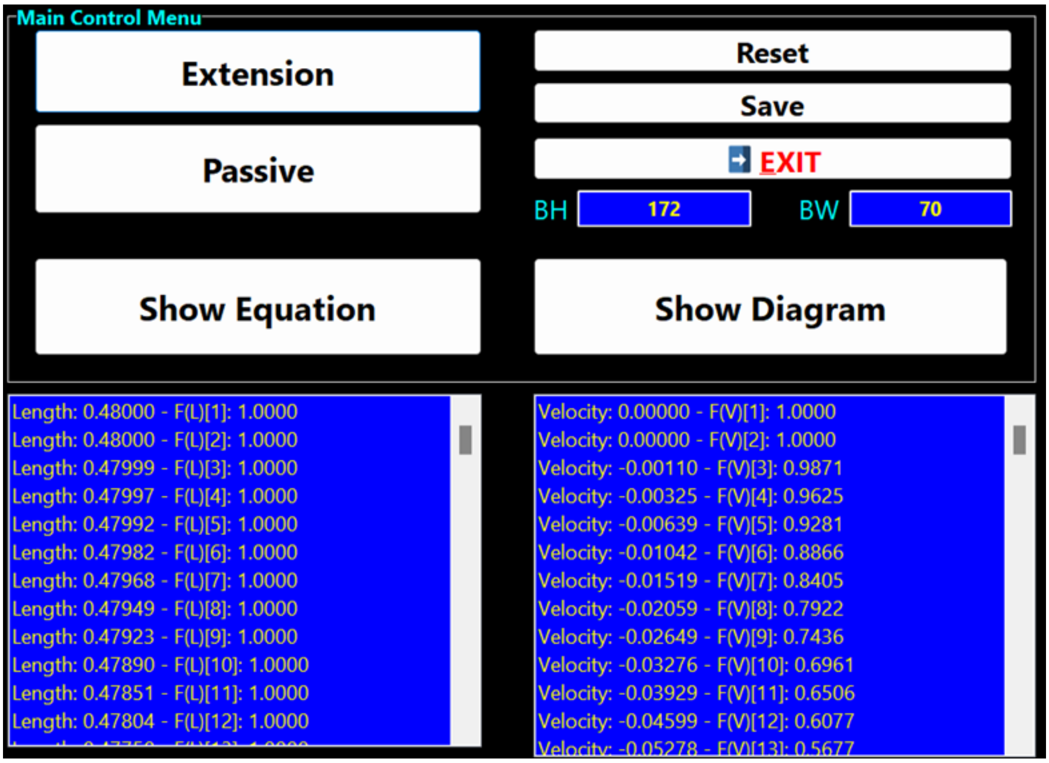
K4: 21.8

phi1: -30

phi2: 75

Figure 2.1. Initial Parameter Tuning

Figure 2.2 demonstrates the real-time monitoring capability added to the software, specifically for observing the internal states of the Hill-type muscle model. Within the Timer1Timer loop, the program calculates the Force-Length factor (f_L) and Force-Velocity factor (f_V) at every time step to modulate the total muscle force. The Force-Length relationship is modeled as a parabolic function, calculated in the code as $f_L = 1.0 - (\frac{0.025\cdot\theta}{0.5\cdot L_{opt}})^2$, representing how force capacity diminishes as the muscle fiber length deviates from its optimal length (L_{opt}). Simultaneously, the Force-Velocity relationship is calculated using a hyperbolic equation, $f_V = \frac{V_{max}-\dot{\theta}}{V_{max}-\dot{\theta}}$, describing the decrease in force generation during rapid concentric shortening. These values are extracted and displayed in the interface components (ListBox or Edit), providing immediate feedback on how the muscle's mechanical operating point evolves during the knee extension movement.

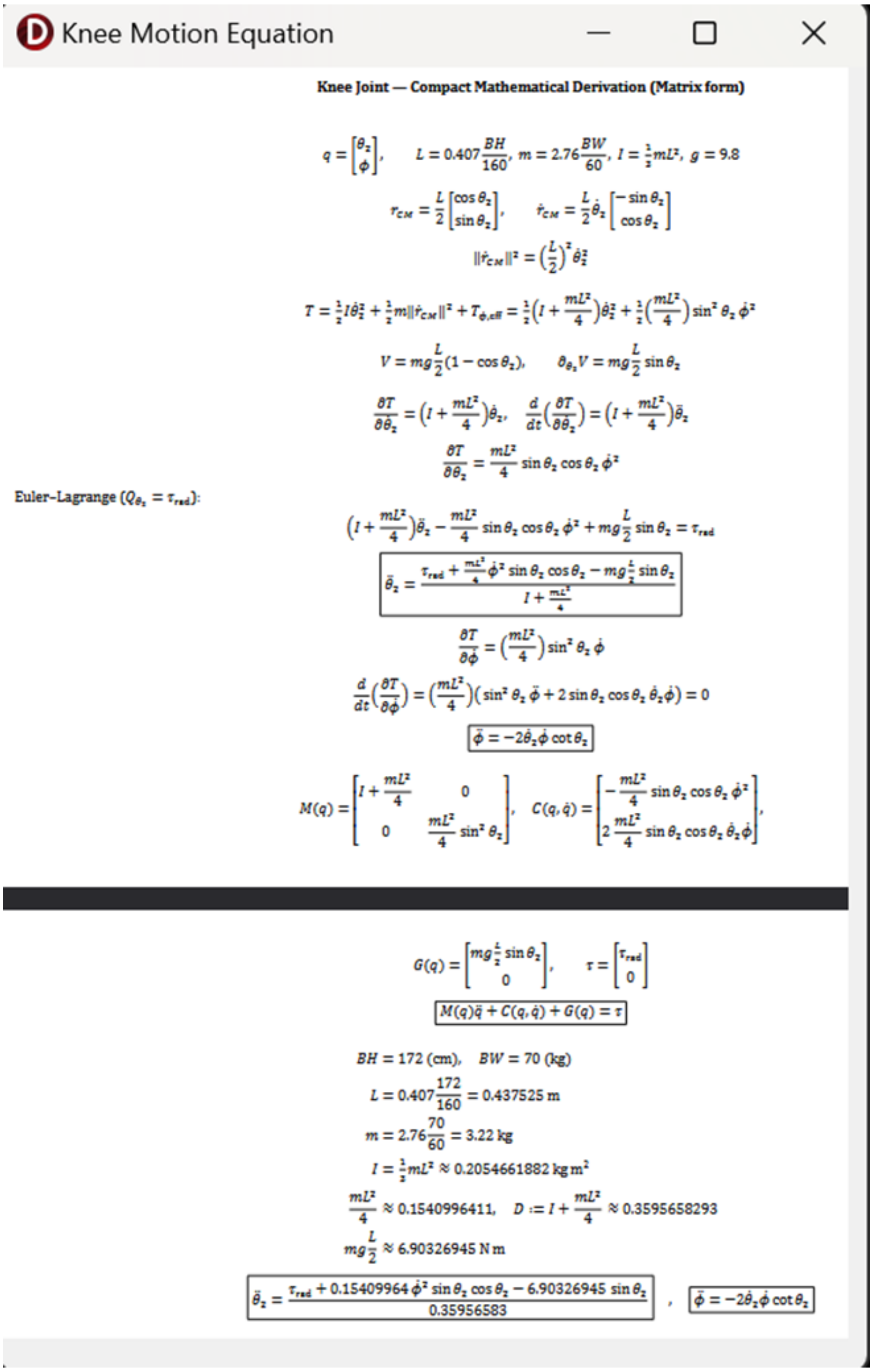


```
Length: 0.48000 - F(L)[1]: 1.0000
Length: 0.48000 - F(L)[2]: 1.0000
Length: 0.47999 - F(L)[3]: 1.0000
Length: 0.47997 - F(L)[4]: 1.0000
Length: 0.47992 - F(L)[5]: 1.0000
Length: 0.47982 - F(L)[6]: 1.0000
Length: 0.47968 - F(L)[7]: 1.0000
Length: 0.47949 - F(L)[8]: 1.0000
Length: 0.47923 - F(L)[9]: 1.0000
Length: 0.47890 - F(L)[10]: 1.0000
Length: 0.47851 - F(L)[11]: 1.0000
Length: 0.47804 - F(L)[12]: 1.0000
```

```
Velocity: 0.00000 - F(V)[1]: 1.0000
Velocity: 0.00000 - F(V)[2]: 1.0000
Velocity: -0.00110 - F(V)[3]: 0.9871
Velocity: -0.00325 - F(V)[4]: 0.9625
Velocity: -0.00639 - F(V)[5]: 0.9281
Velocity: -0.01042 - F(V)[6]: 0.8866
Velocity: -0.01519 - F(V)[7]: 0.8405
Velocity: -0.02059 - F(V)[8]: 0.7922
Velocity: -0.02649 - F(V)[9]: 0.7436
Velocity: -0.03276 - F(V)[10]: 0.6961
Velocity: -0.03929 - F(V)[11]: 0.6506
Velocity: -0.04599 - F(V)[12]: 0.6077
Velocity: -0.05278 - F(V)[13]: 0.5677
```

Figure 2.2. *Two Additional TEdit Components for Visualizing Both F(L) and F(V) Relationship Output*

The mathematical derivation shown in **Figure 2.3** corresponds directly to the single_pend_equ procedure in the Delphi code, which solves the equation of motion using the Lagrangian method. The total moment of inertia (J) about the knee joint is calculated using the parallel axis theorem, $J = I_{cm} + m(\frac{L}{2})^2$, appearing in the code as $m \times (\frac{lgth}{4})^2 + Inertia$. The angular acceleration ($\ddot{\theta}$) is derived by summing all moments acting on the system: the Active Muscle Torque $\tau_{act} = F_{mus} \cdot d$, the Passive Viscoelastic Torque (τ_{pas}), and the Gravitational Torque $\tau_{grav} = -mg\frac{L}{2}sin\theta$. The final equation implemented in the software for the angular acceleration thetadotdot2 is $\ddot{\theta} = \frac{\tau_{act} + \tau_{pas} - \tau_{grav}}{J}$. This differential equation is then solved numerically using the 4th-Order Runge-Kutta method to update the joint angle and velocity over time.



Knee Motion Equation

Knee Joint — Compact Mathematical Derivation (Matrix form)

$$q = \begin{bmatrix} \theta_2 \\ \phi \end{bmatrix}, \quad L = 0.407\frac{BH}{160}, \; m = 2.76\frac{BW}{60}, \; I = \tfrac{1}{3}mL^2, \; g = 9.8$$

$$r_{CM} = \frac{L}{2}\begin{bmatrix} \cos\theta_2 \\ \sin\theta_2 \end{bmatrix}, \quad \dot{r}_{CM} = \frac{L}{2}\dot{\theta}_2\begin{bmatrix} -\sin\theta_2 \\ \cos\theta_2 \end{bmatrix}$$

$$\|\dot{r}_{CM}\|^2 = \left(\frac{L}{2}\right)^2\dot{\theta}_2^2$$

$$T = \tfrac{1}{2}I\dot{\theta}_2^2 + \tfrac{1}{2}m\|\dot{r}_{CM}\|^2 + T_{\phi,\mathrm{eff}} = \tfrac{1}{2}\left(I + \frac{mL^2}{4}\right)\dot{\theta}_2^2 + \tfrac{1}{2}\left(\frac{mL^2}{4}\right)\sin^2\theta_2\,\dot{\phi}^2$$

$$V = mg\frac{L}{2}(1 - \cos\theta_2), \quad \partial_{\theta_2}V = mg\frac{L}{2}\sin\theta_2$$

$$\frac{\partial T}{\partial\dot{\theta}_2} = \left(I + \frac{mL^2}{4}\right)\dot{\theta}_2, \quad \frac{d}{dt}\left(\frac{\partial T}{\partial\dot{\theta}_2}\right) = \left(I + \frac{mL^2}{4}\right)\ddot{\theta}_2$$

$$\frac{\partial T}{\partial\theta_2} = \frac{mL^2}{4}\sin\theta_2\cos\theta_2\,\dot{\phi}^2$$

Euler–Lagrange ($Q_{\theta_2} = \tau_{\mathrm{rad}}$):

$$\left(I + \frac{mL^2}{4}\right)\ddot{\theta}_2 - \frac{mL^2}{4}\sin\theta_2\cos\theta_2\,\dot{\phi}^2 + mg\frac{L}{2}\sin\theta_2 = \tau_{\mathrm{rad}}$$

$$\boxed{\ddot{\theta}_2 = \frac{\tau_{\mathrm{rad}} + \frac{mL^2}{4}\dot{\phi}^2\sin\theta_2\cos\theta_2 - mg\frac{L}{2}\sin\theta_2}{I + \frac{mL^2}{4}}}$$

$$\frac{\partial T}{\partial\dot{\phi}} = \left(\frac{mL^2}{4}\right)\sin^2\theta_2\,\dot{\phi}$$

$$\frac{d}{dt}\left(\frac{\partial T}{\partial\dot{\phi}}\right) = \left(\frac{mL^2}{4}\right)\left(\sin^2\theta_2\,\ddot{\phi} + 2\sin\theta_2\cos\theta_2\,\dot{\theta}_2\dot{\phi}\right) = 0$$

$$\boxed{\ddot{\phi} = -2\dot{\theta}_2\dot{\phi}\cot\theta_2}$$

$$M(q) = \begin{bmatrix} I + \frac{mL^2}{4} & 0 \\ 0 & \frac{mL^2}{4}\sin^2\theta_2 \end{bmatrix}, \quad C(q,\dot{q}) = \begin{bmatrix} -\frac{mL^2}{4}\sin\theta_2\cos\theta_2\,\dot{\phi}^2 \\ 2\frac{mL^2}{4}\sin\theta_2\cos\theta_2\,\dot{\theta}_2\dot{\phi} \end{bmatrix},$$

$$G(q) = \begin{bmatrix} mg\frac{L}{2}\sin\theta_2 \\ 0 \end{bmatrix}, \quad \tau = \begin{bmatrix} \tau_{\mathrm{rad}} \\ 0 \end{bmatrix}$$

$$\boxed{M(q)\ddot{q} + C(q,\dot{q}) + G(q) = \tau}$$

$$BH = 172\ (\mathrm{cm}), \quad BW = 70\ (\mathrm{kg})$$

$$L = 0.407\frac{172}{160} = 0.437525\ \mathrm{m}$$

$$m = 2.76\frac{70}{60} = 3.22\ \mathrm{kg}$$

$$I = \tfrac{1}{3}mL^2 \approx 0.2054661882\ \mathrm{kg\,m^2}$$

$$\frac{mL^2}{4} \approx 0.1540996411, \quad D := I + \frac{mL^2}{4} \approx 0.3595658293$$

$$mg\frac{L}{2} \approx 6.90326945\ \mathrm{N\,m}$$

$$\boxed{\ddot{\theta}_2 = \frac{\tau_{\mathrm{rad}} + 0.15409964\,\dot{\phi}^2\sin\theta_2\cos\theta_2 - 6.90326945\,\sin\theta_2}{0.35956583}}, \quad \boxed{\ddot{\phi} = -2\dot{\theta}_2\dot{\phi}\cot\theta_2}$$

Figure 2.3. *Motion Equation Derivation for Knee Extension Program*

Figure 2.4 presents the visual output of the simulation, rendered using OpenGL in the render procedure. The software constructs a stick-figure representation of the lower limb by using GL_LINES to draw the thigh, shank, and foot segments, and GL_POINTS to visualize the hip, knee, and ankle joints. The movement is driven by the kinematic variable rotangle[g], which is the result of the numerical integration. The code applies geometric transformations, specifically glTranslate to position the segments and glRotate(rotangle[g] - 90, ...) to orient the shank relative to the thigh based on the calculated knee angle. This visualization confirms that the mathematical model translates correctly into physical motion, showing the leg extending from a flexed position towards the horizontal plane as the active muscle torque overcomes gravity.

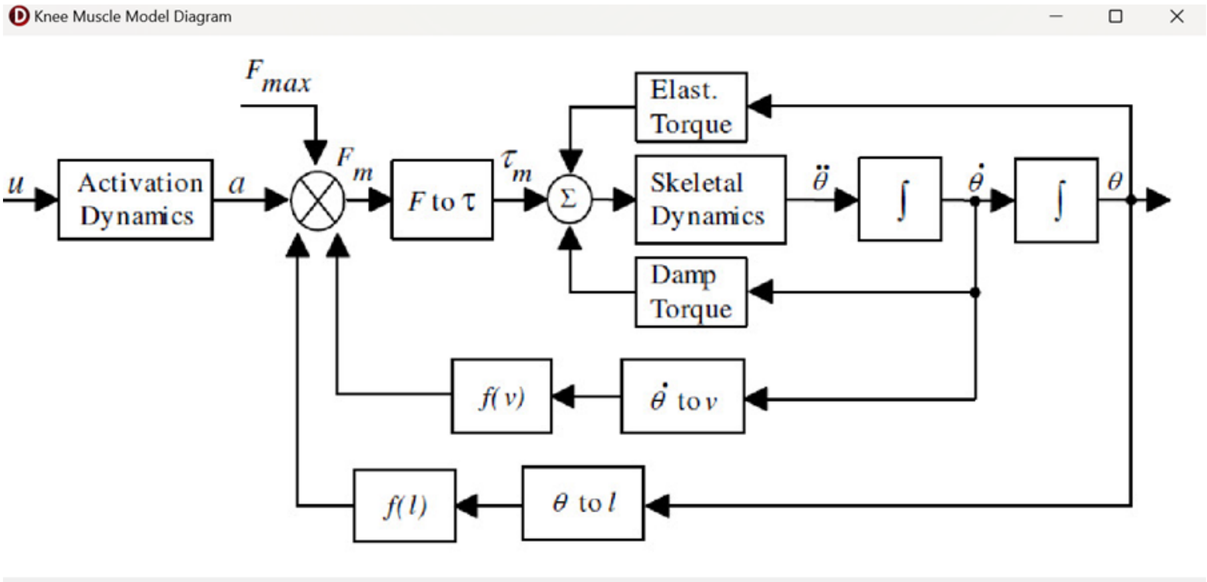


Figure 2.4. *Knee Extension Model Diagram*

ASSIGNMENT 3: TRIPLE PENDULUM AND KNEE EXTENSION

REVISION 2

1. Triple Pendulum

a. Motion Equation Derivation (New)

- System Definition and Anthropometry

The system is modelled as a planar triple pendulum consisting of three rigid links connected by revolute joints. These links correspond to the human upper limb segments:

- Upper Arm (Shoulder to Elbow) - Index 1
- Forearm (Elbow to Wrist) - Index 2
- Hand (Wrist to Distal) - Index 3

Anthropometric Parameter

Using the regression coefficients provided in the system logic and the input data:

- Body Weight (BW): $70\ \text{kg}$
- Body Height (BH): $1.72\ \text{m}$

The segment properties (m = mass, l = length, a = distance from proximal joint to Center of Mass (COM), r = radius of gyration, I = Moment of Inertia) are calculated as follows:

- Segment 1 (Upper Arm):

$m_1 = 0.028 \times BW = 1.96\ \text{kg}$

$l_1 = 0.172 \times BH = 0.2958$

$a_1 = 0.436 \times l_1 = 0.1290\ \text{m}$

$I_1 = m_1 \times (0.322 \times l_1)^2 = 0.0178\ \text{kg} \cdot \text{m}^2$

- Segment 2 (Forearm):

$m_2 = 0.016 \times BW = 1.12\ \text{kg}$

$l_2 = 0.157 \times BH = 0.2700\ \text{m}$

$a_2 = 0.430 \times l_2 = 0.1161\ \text{m}$

$I_2 = m_2 \times (0.303 \times l_2)^2 = 0.0075\ \text{kg} \cdot \text{m}^2$

- Segment 3 (Hand):

$m_3 = 0.006 \times BW = 0.42\ \text{kg}$

$l_3 = 0.0575 \times BH = 0.0989\ \text{m}$

$a_3 = 0.494 \times l_3 = 0.0489\ \text{m}$

$I_3 = m_3 \times (0.297 \times l_3)^2 = 0.00036\ \text{kg} \cdot \text{m}^2$

- Kinematics

Let $\theta_1, \theta_2, \theta_3$ be the absolute angles of the links with respect to the vertical axis (where implies the arm is hanging straight down). Let (x_i, y_i) be the coordinates of the Center of Mass (COM) for each segment i.

- Position of COM 1 (Upper Arm): $x_1 = a_1 \sin\theta_1\,,\ y_1 = -a_1 \cos\theta_1$

- Position of COM 2 (Forearm): The origin of the second link is the end of the first link. $x_2 = l_1 \sin\theta_1 + a_2 \sin\theta_2,\ y_2 = -l_1 \cos\theta_1 - a_2 \cos\theta_2$
- Position of COM 3 (Hand): The origin of the third link is the end of the second link. $x_3 = l_1 \sin\theta_1 + l_2 \sin\theta_2 + a_3 \sin\theta_3,\ \ y_3 = -l_1 \cos\theta_1 - l_2 \cos\theta_2 - a_3 \cos\theta_3$
- Velocities (Time Derivatives): By differentiating positions with respect to time (t), I obtain the velocities ($\dot{x}, \dot{y}$). Using the chain rule ($\frac{d}{dt}\sin\theta = \dot{\theta}\cos\theta$), the squared magnitudes of velocity ($v_i^2 = \dot{x}_i^2 + \dot{y}_i^2$) are derived as:

$$v_1^2 = a_1^2\dot{\theta}_1^2$$

$$v_2^2 = l_1^2\dot{\theta}_1^2 + a_2^2\dot{\theta}_2^2 + 2l_1a_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2)$$

$$v_3^2 = l_1^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + a_3^2\dot{\theta}_3^2 + 2l_1l_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2) + 2l_1a_3\dot{\theta}_1\dot{\theta}_3\cos(\theta_1 - \theta_3) \\ + 2l_2a_3\dot{\theta}_2\dot{\theta}_3\cos(\theta_2 - \theta_3)$$

- **Lagrangian Dynamics**

The Lagrangian L is defined as the difference between total Kinetic Energy (E_k) and total Potential Energy (E_p): $L = E_k - E_p$

- Kinetic Energy (E_k)

The total kinetic energy is the sum of translational and rotational energy for all three segments:

$$E_k = \sum_{i=1}^{3}\left(\frac{1}{2}m_iv_i^2 + \frac{1}{2}I_i\dot{\theta}_i^2\right)$$

Substituting the velocity equations derived above and grouping terms by angular velocity pairs ($\dot{\theta}_i\dot{\theta}_j$), I define the Inertia Coefficients (M_{ij}):

$$E_k = \frac{1}{2}M_{11}\dot{\theta}_1^2 + \frac{1}{2}M_{22}\dot{\theta}_2^2 + \frac{1}{2}M_{33}\dot{\theta}_3^2 + M_{12}\dot{\theta}_1\dot{\theta}_2 + M_{13}\dot{\theta}_1\dot{\theta}_3 + M_{23}\dot{\theta}_2\dot{\theta}_3$$

Where the structural constants (K) are:

$K_{11} = I_1 + m_1a_1^2 + (m_2 + m_3)l_1^2$

$K_{22} = I_2 + m_2a_2^2 + m_3l_2^2$

$K_{33} = I_3 + m_3a_3^2$

$K_{12} = l_1(m_2a_2 + m_3l_2)$

$K_{13} = m_3l_1a_3$

$K_{23} = m_3l_2a_3$

And the Inertia Matrix elements depend on the configuration:

$M_{11} = K_{11}$

$M_{22} = K_{22}$

$M_{33} = K_{33}$

$M_{12} = M_{21} = K_{12}\cos(\theta_1 - \theta_2)$

$M_{13} = M_{31} = K_{13}\cos(\theta_1 - \theta_3)$

$M_{23} = M_{32} = K_{23}\cos(\theta_2 - \theta_3)$

- Potential Energy (E_p)

Assuming gravity (g) acts in the negative y-direction:

$$E_p = m_1 g y_1 + m_2 g y_2 + m_3 g y_3$$

Substituting y_i coordinates:

$$E_p = -\mathrm{g}\cos\theta_1(m_1 a_1 + m_2 l_1 + m_3 l_1) - \mathrm{g}\cos\theta_2(m_2 a_2 + m_3 l_2) - \mathrm{g}\cos\theta_3(m_3 a_3)$$

Let us define the Gravity Coefficients:

$G_{coef1} = g(m_1 a_1 + m_2 l_1 + m_3 l_1)$

$G_{coef2} = g(m_2 a_2 + m_3 l_2)$

$G_{coef3} = g(m_3 a_3)$

Thus:

$$E_p = -(G_{coef1}\cos\,\theta_1 + G_{coef2}\cos\,\theta_2 + G_{coef3}\cos\,\theta_3)$$

- Lagrange's Equations

The equations of motion are derived using:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_i}\right) - \frac{\partial L}{\partial \theta_i} = \tau_i$$

Applying this to $\theta_1, \theta_2, \theta_3$ results in a system of second-order differential equations in the form:

$$M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) = \tau$$

Inertia Matrix (M), using the coefficients defined in the Kinetic Energy section:

$$M = \begin{bmatrix} K_{11} & K_{12}c_{12} & K_{13}c_{13} \\ K_{12}c_{12} & K_{22} & K_{23}c_{23} \\ K_{13}c_{13} & K_{23}c_{23} & K_{33} \end{bmatrix}$$

Notation: $c_{ij} = \cos(\theta_i - \theta_j)$

Coriolis and Centrifugal Vector (V), this vector arises from differentiating the cosine coupling terms with respect to time and angles.

$$V = \begin{bmatrix} K_{12}s_{12}\dot{\theta}_2^2 + K_{13}s_{13}\dot{\theta}_3^2 \\ -K_{12}s_{12}\dot{\theta}_1^2 + K_{23}s_{23}\dot{\theta}_3^2 \\ -K_{13}s_{13}\dot{\theta}_1^2 - K_{23}s_{23}\dot{\theta}_2^2 \end{bmatrix}$$

Notation: $s_{ij} = \sin(\theta_i - \theta_j)$

Gravity Vector (G), derived from $\frac{\partial E_p}{\partial \theta_i}$:

$$G = \begin{bmatrix} G_{coef1}\sin\theta_1 \\ G_{coef2}\sin\theta_2 \\ G_{coef3}\sin\theta_3 \end{bmatrix}$$

- **Torque Model (τ)**

The generalized torque τ applied to each joint is the sum of Passive Torques (internal biomechanics) and Active Torques (control/muscles).

$$\tau_i = \tau_{passive,i} + \tau_{active,i}$$

- Passive Torque Model

  The passive torque represents joint friction (viscous damping) and the elasticity of ligaments/tendons at the limits of motion. It is modeled using a linear damping term and a double-exponential spring term:

  $$\tau_{passive} = -c\dot{\theta} + k_1 e^{-k_2(\theta-\phi_1)} + k_3 e^{k_4(\theta-\phi_2)}$$

  c: Damping coefficient.

  k_1, k_2 : Coefficients for the extension limit spring.

  k_3, k_4 : Coefficients for the flexion limit spring.

  ϕ_1, ϕ_2 : Angular limits (engagement angles).

- Active Torque Model (PD Controller)

  The active torque drives the arm toward a target trajectory. It is modeled using a Proportional-Derivative (PD) controller, representing muscle stiffness (K_p) and damping (K_d).

  $$\tau_{active} = K_p\left(\theta_{target}(t) - \theta\right) + K_d\left(\dot{\theta}_{target}(t) - \dot{\theta}\right)$$

  The target trajectory is defined by a sinusoidal function:

  $$\theta_{target}(t) = \text{Offset} + A\sin\left(2\pi f t + \text{Phase}\right)$$

  Computing Acceleration ($\ddot{\theta}$), the complete Motion Equation is:

  $$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} - \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} - \begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix}$$

  To solve for the angular accelerations $\ddot{\theta}_1, \ddot{\theta}_2, \ddot{\theta}_3$, we define the Right Hand Side (RHS) vector R:

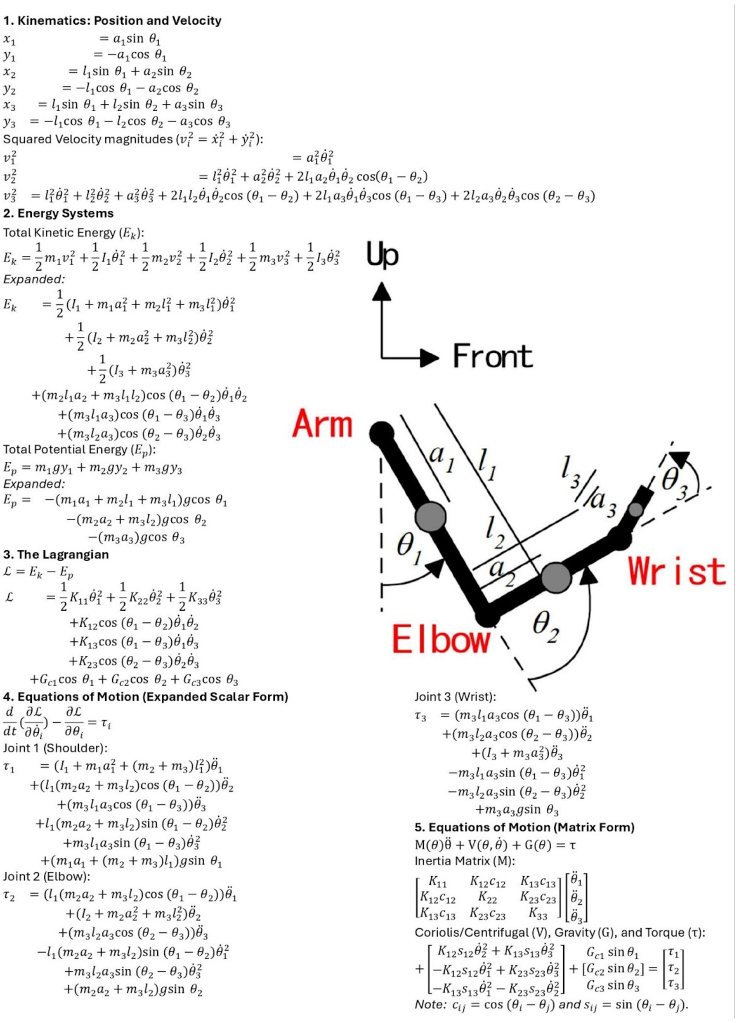
  $$R = \tau - V - G$$

  The system becomes a linear equation $M\ddot{\theta} = R$. This is solved using Cramer's Rule to ensure numerical stability for the 3x3 system:

  $$\ddot{\theta}_1 = \frac{\det(M_1)}{\det(M)}, \ddot{\theta}_2 = \frac{\det(M_2)}{\det(M)}, \ddot{\theta}_3 = \frac{\det(M_3)}{\det(M)}$$

  Where M_i is the matrix M with the i-th column replaced by vector R. These accelerations are then integrated over time step dt to update velocity and position.

Figure 1 below showed the general mechanism of the triple pendulum motion equation that the previous method derived.



1. Kinematics: Position and Velocity

$x_1 = a_1\sin\,\theta_1$

$y_1 = -a_1\cos\,\theta_1$

$x_2 = l_1\sin\,\theta_1 + a_2\sin\,\theta_2$

$y_2 = -l_1\cos\,\theta_1 - a_2\cos\,\theta_2$

$x_3 = l_1\sin\,\theta_1 + l_2\sin\,\theta_2 + a_3\sin\,\theta_3$

$y_3 = -l_1\cos\,\theta_1 - l_2\cos\,\theta_2 - a_3\cos\,\theta_3$

Squared Velocity magnitudes ($v_i^2 = \dot{x}_i^2 + \dot{y}_i^2$):

$v_1^2 = a_1^2\dot{\theta}_1^2$

$v_2^2 = l_1^2\dot{\theta}_1^2 + a_2^2\dot{\theta}_2^2 + 2l_1a_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2)$

$v_3^2 = l_1^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + a_3^2\dot{\theta}_3^2 + 2l_1l_2\dot{\theta}_1\dot{\theta}_2\cos\,(\theta_1 - \theta_2) + 2l_1a_3\dot{\theta}_1\dot{\theta}_3\cos\,(\theta_1 - \theta_3) + 2l_2a_3\dot{\theta}_2\dot{\theta}_3\cos\,(\theta_2 - \theta_3)$

2. Energy Systems

Total Kinetic Energy (E_k):

$E_k = \frac{1}{2}m_1v_1^2 + \frac{1}{2}I_1\dot{\theta}_1^2 + \frac{1}{2}m_2v_2^2 + \frac{1}{2}I_2\dot{\theta}_2^2 + \frac{1}{2}m_3v_3^2 + \frac{1}{2}I_3\dot{\theta}_3^2$

Expanded:

$$\begin{aligned} E_k &= \frac{1}{2}(I_1 + m_1a_1^2 + m_2l_1^2 + m_3l_1^2)\dot{\theta}_1^2 \\ &+ \frac{1}{2}(I_2 + m_2a_2^2 + m_3l_2^2)\dot{\theta}_2^2 \\ &+ \frac{1}{2}(I_3 + m_3a_3^2)\dot{\theta}_3^2 \\ &+(m_2l_1a_2 + m_3l_1l_2)\cos\,(\theta_1 - \theta_2)\dot{\theta}_1\dot{\theta}_2 \\ &+(m_3l_1a_3)\cos\,(\theta_1 - \theta_3)\dot{\theta}_1\dot{\theta}_3 \\ &+(m_3l_2a_3)\cos\,(\theta_2 - \theta_3)\dot{\theta}_2\dot{\theta}_3 \end{aligned}$$

Total Potential Energy (E_p):

$E_p = m_1gy_1 + m_2gy_2 + m_3gy_3$

Expanded:

$$\begin{aligned} E_p = &-(m_1a_1 + m_2l_1 + m_3l_1)g\cos\,\theta_1 \\ &-(m_2a_2 + m_3l_2)g\cos\,\theta_2 \\ &-(m_3a_3)g\cos\,\theta_3 \end{aligned}$$

3. The Lagrangian

$\mathcal{L} = E_k - E_p$

$$\begin{aligned} \mathcal{L} &= \frac{1}{2}K_{11}\dot{\theta}_1^2 + \frac{1}{2}K_{22}\dot{\theta}_2^2 + \frac{1}{2}K_{33}\dot{\theta}_3^2 \\ &+K_{12}\cos\,(\theta_1 - \theta_2)\dot{\theta}_1\dot{\theta}_2 \\ &+K_{13}\cos\,(\theta_1 - \theta_3)\dot{\theta}_1\dot{\theta}_3 \\ &+K_{23}\cos\,(\theta_2 - \theta_3)\dot{\theta}_2\dot{\theta}_3 \\ &+G_{c1}\cos\,\theta_1 + G_{c2}\cos\,\theta_2 + G_{c3}\cos\,\theta_3 \end{aligned}$$

4. Equations of Motion (Expanded Scalar Form)

$\frac{d}{dt}(\frac{\partial\mathcal{L}}{\partial\dot{\theta}_i}) - \frac{\partial\mathcal{L}}{\partial\theta_i} = \tau_i$

Joint 1 (Shoulder):

$$\begin{aligned} \tau_1 &= (I_1 + m_1a_1^2 + (m_2 + m_3)l_1^2)\ddot{\theta}_1 \\ &+(l_1(m_2a_2 + m_3l_2)\cos\,(\theta_1 - \theta_2))\ddot{\theta}_2 \\ &+(m_3l_1a_3\cos\,(\theta_1 - \theta_3))\ddot{\theta}_3 \\ &+l_1(m_2a_2 + m_3l_2)\sin\,(\theta_1 - \theta_2)\dot{\theta}_2^2 \\ &+m_3l_1a_3\sin\,(\theta_1 - \theta_3)\dot{\theta}_3^2 \\ &+(m_1a_1 + (m_2 + m_3)l_1)g\sin\,\theta_1 \end{aligned}$$

Joint 2 (Elbow):

$$\begin{aligned} \tau_2 &= (l_1(m_2a_2 + m_3l_2)\cos\,(\theta_1 - \theta_2))\ddot{\theta}_1 \\ &+(I_2 + m_2a_2^2 + m_3l_2^2)\ddot{\theta}_2 \\ &+(m_3l_2a_3\cos\,(\theta_2 - \theta_3))\ddot{\theta}_3 \\ &-l_1(m_2a_2 + m_3l_2)\sin\,(\theta_1 - \theta_2)\dot{\theta}_1^2 \\ &+m_3l_2a_3\sin\,(\theta_2 - \theta_3)\dot{\theta}_3^2 \\ &+(m_2a_2 + m_3l_2)g\sin\,\theta_2 \end{aligned}$$

Joint 3 (Wrist):

$$\begin{aligned} \tau_3 &= (m_3l_1a_3\cos\,(\theta_1 - \theta_3))\ddot{\theta}_1 \\ &+(m_3l_2a_3\cos\,(\theta_2 - \theta_3))\ddot{\theta}_2 \\ &+(I_3 + m_3a_3^2)\ddot{\theta}_3 \\ &-m_3l_1a_3\sin\,(\theta_1 - \theta_3)\dot{\theta}_1^2 \\ &-m_3l_2a_3\sin\,(\theta_2 - \theta_3)\dot{\theta}_2^2 \\ &+m_3a_3g\sin\,\theta_3 \end{aligned}$$

5. Equations of Motion (Matrix Form)

$\mathrm{M}(\theta)\ddot{\theta} + \mathrm{V}(\theta,\dot{\theta}) + \mathrm{G}(\theta) = \tau$

Inertia Matrix (M):

$$\begin{bmatrix} K_{11} & K_{12}c_{12} & K_{13}c_{13} \\ K_{12}c_{12} & K_{22} & K_{23}c_{23} \\ K_{13}c_{13} & K_{23}c_{23} & K_{33} \end{bmatrix}\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{bmatrix}$$

Coriolis/Centrifugal (V), Gravity (G), and Torque (τ):

$$+\begin{bmatrix} K_{12}s_{12}\dot{\theta}_2^2 + K_{13}s_{13}\dot{\theta}_3^2 \\ -K_{12}s_{12}\dot{\theta}_1^2 + K_{23}s_{23}\dot{\theta}_3^2 \\ -K_{13}s_{13}\dot{\theta}_1^2 - K_{23}s_{23}\dot{\theta}_2^2 \end{bmatrix} + \begin{matrix} G_{c1}\sin\theta_1 \\ [G_{c2}\sin\theta_2] \\ G_{c3}\sin\theta_3 \end{matrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix}$$

Note: $c_{ij} = \cos\,(\theta_i - \theta_j)$ *and* $s_{ij} = \sin\,(\theta_i - \theta_j)$.

Figure 1. *Triple Pendulum Motion Equation Derivation*

b. Code Explanation

(i) CalculateAnthropometry

```
procedure TForm1.CalculateAnthropometry;
var
  BW, BH: Double;
  r1, r2, r3: Double;
begin
  BW := ReadVal(edBW, 70.0);
  BH := ReadVal(edBH, 1.72); // Meter

  // Upper Arm
  SegArm.m := 0.028 * BW;
  SegArm.l := 0.172 * BH;
  SegArm.a := 0.436 * SegArm.l;
  r1 := 0.322 * SegArm.l;
  SegArm.I_inert := SegArm.m * Sqr(r1);

  // Forearm
  SegForearm.m := 0.016 * BW;
  SegForearm.l := 0.157 * BH;
  SegForearm.a := 0.430 * SegForearm.l;
  r2 := 0.303 * SegForearm.l;
  SegForearm.I_inert := SegForearm.m * Sqr(r2);

  // Hand
  SegHand.m := 0.006 * BW;
  SegHand.l := 0.0575 * BH;
  SegHand.a := 0.494 * SegHand.l;
  r3 := 0.297 * SegHand.l;
  SegHand.I_inert := SegHand.m * Sqr(r3);
end;
```

This procedure is responsible for initializing the physical properties of the arm segments based on the user's input for Body Weight (BW) and Body Height (BH). It utilizes standard anthropometric regression coefficients (likely from Dempster or Winter) to estimate the mass (m), length (l), distance to the Center of Mass (a), and Moment of Inertia (I_inert) for the Upper Arm, Forearm, and Hand. For each segment, the radius of gyration (r) is first calculated as a percentage of the segment length, which is then squared and multiplied by the segment mass to determine the moment of inertia about the center of mass ($I = m\cdot r^2$). These values form the constants ($K_{11}, K_{22}, etc.$) used later in the Lagrangian motion equations.

(ii) ComputePassiveTorque

```
function TForm1.ComputePassiveTorque(theta, dtheta: Double; p: TJointParams): Double;
var
  thDeg: Double;
  PasTorque: Double;
  extTerm, flexTerm: Double;
begin
  thDeg := RadToDeg(theta);

  // Soft-limit di bawah (theta < phi1) -> dorong balik ke atas (+)
  extTerm := p.k1 * SafeExp(-p.k2 * (thDeg - p.phi1));

  // Soft-limit di atas (theta > phi2) -> dorong balik ke bawah (-)
  flexTerm := -p.k3 * SafeExp( p.k4 * (thDeg - p.phi2));

  PasTorque := (-p.c * dtheta) + extTerm + flexTerm;

  // Clamp torsi pasif agar integrasi stabil (hindari NaN)
  PasTorque := ClampD(PasTorque, -300.0, 300.0);

  if BadNumber(PasTorque) then PasTorque := 0.0;
  Result := PasTorque;
end;
```

This function models the internal biomechanical forces acting on a joint when no active muscle force is applied. It calculates the passive torque ($\tau_{passive}$) based on the current angle (theta) and angular velocity (dtheta). The model consists of three components: a viscous damping term ($-p.c \cdot \dot{\theta}$) that represents joint friction and energy dissipation, and two exponential spring terms (extTerm and flexTerm) that simulate the elasticity of ligaments and tissues at the joint limits. extTerm provides a restoring force when the joint extends beyond its lower limit (phi1), while flexTerm provides a restoring force when it flexes beyond its upper limit (phi2). The function includes safety clamps (ClampD and BadNumber) to prevent numerical instability or overflow (NaN values) during the simulation.

(iii) ComputeActiveTorque

```
function TForm1.ComputeActiveTorque(SimTime, th, dth: Double; p: TJointParams): Double;
var
  TargetPos, TargetVel: Double;
  ErrorPos, ErrorVel: Double;
  Omega: Double;
  CalculatedTorque: Double;
begin
  Omega := 2 * Pi * p.TargetFreq;

  TargetPos := p.TargetOffset + p.TargetAmp * Sin(Omega * SimTime + p.TargetPhase);
  TargetVel := p.TargetAmp * Omega * Cos(Omega * SimTime + p.TargetPhase);

  ErrorPos := TargetPos - th;
  ErrorVel := TargetVel - dth;

  CalculatedTorque := (p.Kp * ErrorPos) + (p.Kd * ErrorVel);

  CalculatedTorque := ClampD(CalculatedTorque, -50.0, 50.0);
  if BadNumber(CalculatedTorque) then CalculatedTorque := 0.0;

  Result := CalculatedTorque;
end;
```

This function calculates the active muscle torque required to drive the arm along a desired trajectory. It employs a Proportional-Derivative (PD) controller logic. First, it determines the instantaneous target position (TargetPos) and target velocity (TargetVel) using a sinusoidal function defined by the amplitude, frequency, and phase parameters. It then calculates the positional error (ErrorPos) and velocity error (ErrorVel). The final torque is the sum of the proportional term ($K_p \cdot ErrorPos$, representing stiffness) and the derivative term ($K_p \cdot ErrorVel$, representing damping). This torque mimics the central nervous system's command to correct deviations from the intended movement path. Similar to the passive torque, the output is clamped to realistic limits to ensure simulation stability.

(iv) ComputeMotionEquation

```
procedure TForm1.ComputeMotionEquation;
var
  K11, K22, K33, K12, K13, K23: Double;
  G_coeff1, G_coeff2, G_coeff3: Double;
  g_grav: Double;

  M11, M12, M13, M21, M22, M23, M31, M32, M33: Double;
  V1, V2, V3: Double;
  G1, G2, G3: Double;

  Tau1, Tau2, Tau3: Double;
  TauPassive1, TauPassive2, TauPassive3: Double;
  TauActive1, TauActive2, TauActive3: Double;

  c12, c13, c23: Double;
  s12, s13, s23: Double;
  s1, s2, s3: Double;

  RHS1, RHS2, RHS3: Double;
begin
  g_grav := 9.81;

  c12 := Cos(th1 - th2);
  c13 := Cos(th1 - th3);
  c23 := Cos(th2 - th3);

  s12 := Sin(th1 - th2);
  s13 := Sin(th1 - th3);
  s23 := Sin(th2 - th3);

  s1 := Sin(th1);
  s2 := Sin(th2);
  s3 := Sin(th3);

  K11 := SegArm.I_inert + SegArm.m*Sqr(SegArm.a) + (SegForearm.m + SegHand.m)*Sqr(SegArm.l);
  K22 := SegForearm.I_inert + SegForearm.m*Sqr(SegForearm.a) + SegHand.m*Sqr(SegForearm.l);
  K33 := SegHand.I_inert + SegHand.m*Sqr(SegHand.a);

  K12 := SegArm.l * (SegForearm.m*SegForearm.a + SegHand.m*SegForearm.l);
  K13 := SegHand.m * SegArm.l * SegHand.a;
  K23 := SegHand.m * SegForearm.l * SegHand.a;

  G_coeff1 := g_grav * (SegArm.m*SegArm.a + SegForearm.m*SegArm.l + SegHand.m*SegArm.l);
  G_coeff2 := g_grav * (SegForearm.m*SegForearm.a + SegHand.m*SegForearm.l);
  G_coeff3 := g_grav * (SegHand.m*SegHand.a);

  M11 := K11;
  M12 := K12 * c12;  M21 := M12;
  M13 := K13 * c13;  M31 := M13;
  M22 := K22;
  M23 := K23 * c23;  M32 := M23;
  M33 := K33;

  UpdateMatrixGrid([M11, M12, M13, M21, M22, M23, M31, M32, M33]);

  V1 := K12*s12*Sqr(dth2) + K13*s13*Sqr(dth3);
  V2 := -K12*s12*Sqr(dth1) + K23*s23*Sqr(dth3);
  V3 := -K13*s13*Sqr(dth1) - K23*s23*Sqr(dth2);

  G1 := G_coeff1 * s1;
  G2 := G_coeff2 * s2;
  G3 := G_coeff3 * s3;

  TauPassive1 := ComputePassiveTorque(th1, dth1, ParamSh);
  TauPassive2 := ComputePassiveTorque(th2, dth2, ParamEl);
  TauPassive3 := ComputePassiveTorque(th3, dth3, ParamWr);

  TauActive1 := ComputeActiveTorque(SimTime, th1, dth1, ParamSh);
  TauActive2 := ComputeActiveTorque(SimTime, th2, dth2, ParamEl);
  TauActive3 := ComputeActiveTorque(SimTime, th3, dth3, ParamWr);

  Tau1 := TauPassive1 + TauActive1;
  Tau2 := TauPassive2 + TauActive2;
  Tau3 := TauPassive3 + TauActive3;

  RHS1 := Tau1 - V1 - G1;
  RHS2 := Tau2 - V2 - G2;
  RHS3 := Tau3 - V3 - G3;

  SolveCramer(
    M11, M12, M13,
    M21, M22, M23,
    M31, M32, M33,
    RHS1, RHS2, RHS3,
    ddth1, ddth2, ddth3
  );

  // Clamp percepatan untuk stabilitas numerik
  ddth1 := ClampD(ddth1, -800.0, 800.0);
  ddth2 := ClampD(ddth2, -800.0, 800.0);
  ddth3 := ClampD(ddth3, -800.0, 800.0);
end;
```

This procedure is the core physics engine of the simulation, implementing the Lagrangian dynamics for a triple pendulum. It first calculates the necessary trigonometric values and structural constants (K_{11}, K_{12}, etc.) derived from the anthropometric data. It then constructs the Inertia Matrix (M), the Coriolis and Centrifugal Vector (V), and the Gravitational Vector (G) according to the derived equations of motion. The total torque (τ) is computed by summing the active and passive torques for each joint. The system of equations is then formulated as $M \cdot \ddot{\theta} = \tau - V - G$. Finally, it solves this linear system for the angular accelerations (ddth1, ddth2, ddth3) using Cramer's Rule via the SolveCramer procedure. These accelerations are the output required to update the system's state in the next time step.

(v) SolveCramer

```
procedure TForm1.SolveCramer(
  M11, M12, M13,
  M21, M22, M23,
  M31, M32, M33,
  R1, R2, R3 : Double;
  out res1, res2, res3 : Double);
var
  DetM, Det1, Det2, Det3: Double;
  eps: Double;
```

```
begin
  DetM := M11*(M22*M33 - M23*M32) -
          M12*(M21*M33 - M23*M31) +
          M13*(M21*M32 - M22*M31);

  eps := 1e-9;
  if Abs(DetM) < eps then
  begin
    if DetM >= 0 then DetM := eps else DetM := -eps; // preserve sign
  end;

  Det1 := R1 *(M22*M33 - M23*M32) -
          M12*(R2 *M33 - M23*R3 ) +
          M13*(R2 *M32 - M22*R3 );

  Det2 := M11*(R2 *M33 - M23*R3 ) -
          R1 *(M21*M33 - M23*M31) +
          M13*(M21*R3  - R2 *M31);

  Det3 := M11*(M22*R3  - R2 *M32) -
          M12*(M21*R3  - R2 *M31) +
          R1 *(M21*M32 - M22*M31);

  res1 := Det1 / DetM;
  res2 := Det2 / DetM;
  res3 := Det3 / DetM;

  if BadNumber(res1) then res1 := 0.0;
  if BadNumber(res2) then res2 := 0.0;
  if BadNumber(res3) then res3 := 0.0;
end;
```

This helper procedure solves a system of linear equations in the form $Ax = B$ for a 3x3 matrix using Cramer's Rule. In the context of this simulation, it is used to solve for the angular accelerations ($\ddot{\theta}_1, \ddot{\theta}_2, \ddot{\theta}_3$). It calculates the determinant of the main inertia matrix (DetM). Then, it calculates three auxiliary determinants (Det1, Det2, Det3) by sequentially replacing the columns of the inertia matrix with the Right Hand Side vector (R, which represents the net forces). The solutions are obtained by dividing the auxiliary determinants by the main determinant ($res_i = \frac{Det_i}{DetM}$). The code includes a check for a near-zero determinant to prevent division by zero errors, ensuring the robustness of the solver.

(vi) Timer1Timer

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  ComputeMotionEquation;

  // Jika ada NaN/Inf, stop agar tidak “meledak”
  if BadNumber(ddth1) or BadNumber(ddth2) or BadNumber(ddth3) or
     BadNumber(dth1)  or BadNumber(dth2)  or BadNumber(dth3)  or
     BadNumber(th1)   or BadNumber(th2)   or BadNumber(th3) then
  begin
    Timer1.Enabled := False;
    Exit;
  end;

  // Semi-implicit Euler
  dth1 := dth1 + ddth1 * dt;
  dth2 := dth2 + ddth2 * dt;
  dth3 := dth3 + ddth3 * dt;

  th1 := th1 + dth1 * dt;
  th2 := th2 + dth2 * dt;
  th3 := th3 + dth3 * dt;

  SimTime := SimTime + dt;

  UpdateCharts;
  RenderScene;

  if FPassiveMode then
    StopIfSettled;
end;
```

This procedure is the main simulation loop, triggered at every timer interval. It orchestrates the entire simulation process. First, it calls ComputeMotionEquation to calculate the current angular accelerations based on the current state (angles and velocities). It then performs numerical integration using the Semi-Implicit Euler method: updating the velocities (dth) using the new accelerations (ddth), and subsequently updating the positions (th) using the new velocities. The simulation time (SimTime) is incremented by the time step (dt). Finally, it updates the visual output by calling UpdateCharts to plot the data and RenderScene to draw the 3D animation. If the simulation is in Passive Mode, it also checks if the arm has come to a rest using StopIfSettled.

(vii) RenderScene

```
procedure TForm1.RenderScene;
var
  vL1, vL2, vL3: Double;
  angleElbow, angleWrist: Double;
  radShoulder, radElbow, radWrist: Double;
  i: Integer;
  fingerLen, fingerWidth: Double;
  thumbLen: Double;
  HandScale: Double;
  rotangle1, rotangle2, rotangle3: Double;
begin
  if (myDC = 0) or (myRC = 0) then Exit;
  wglMakeCurrent(myDC, myRC);
  glClear(GL_COLOR_BUFFER_BIT or GL_DEPTH_BUFFER_BIT);
  glLoadIdentity;

  glTranslate(xpos, ypos, zpos);
  glRotate(pitch, 1, 0, 0);
  glRotate(yaw, 0, 1, 0);
  glRotate(roll, 0, 0, 1);

  DrawAxis(4.0);

  vL1 := SegArm.l * 10;
  vL2 := SegForearm.l * 10;
  vL3 := SegHand.l * 10;

  radShoulder := 0.75;
  radElbow := 0.55;
  radWrist := 0.40;

  rotangle1 := 20.0;
  rotangle2 := 30.0;
  rotangle3 := 10.0;

  glColor3f(0.75, 0.75, 0.75);

  glPushMatrix();
    gluSphere(Sphere, radShoulder * 1.1, 32, 32);

    glRotate(RadToDeg(th1) + 90, 1, 0, 0);

    gluCylinder(Cylinder, radShoulder, radElbow, vL1, 32, 32);

    glTranslate(0, 0, vL1);
    gluSphere(Sphere, radElbow * 1.0, 32, 32);

    angleElbow := RadToDeg(th2 - th1);
    if angleElbow < 0 then angleElbow := 0;
    glRotate(angleElbow, 1, 0, 0);

    gluCylinder(Cylinder, radElbow, radWrist, vL2, 32, 32);

    glTranslate(0, 0, vL2);
    gluSphere(Sphere, radWrist, 24, 24);

    angleWrist := RadToDeg(th3 - th2);
    glRotate(angleWrist, 1, 0, 0);

    glPushMatrix();
      glRotate(-90, 0, 0, 1);

      if vL3 > 0.1 then
        HandScale := (vL3 * 1.9) / 2.4
      else
        HandScale := 1.0;

      glScalef(HandScale, HandScale, HandScale);

      glPushMatrix();
        glTranslate(0, 0, 1.2);
        DrawBox(1.2, 0.5, 2.4);
      glPopMatrix();

      fingerWidth := 0.24;
      for i := 0 to 3 do
      begin
       glPushMatrix();
        glTranslate((0.33 - (i * fingerWidth)), 0, 2.4);

        if i = 1 then fingerLen := 1.6
        else if i = 3 then fingerLen := 1.1
        else fingerLen := 1.4;

        glRotate(rotangle1, 1, 0, 0);
        gluCylinder(Cylinder, 0.11, 0.10, fingerLen * 0.5, 12, 4);
        glTranslate(0, 0, fingerLen * 0.5);
        gluSphere(Sphere, 0.10, 8, 8);

        glRotate(rotangle2 * 0.8, 1, 0, 0);
        gluCylinder(Cylinder, 0.10, 0.09, fingerLen * 0.3, 12, 4);
        glTranslate(0, 0, fingerLen * 0.3);
        gluSphere(Sphere, 0.09, 8, 8);

        glRotate(rotangle3 * 0.5, 1, 0, 0);
        gluCylinder(Cylinder, 0.09, 0.08, fingerLen * 0.2, 12, 4);
        glTranslate(0, 0, fingerLen * 0.2);
        gluSphere(Sphere, 0.08, 8, 8);
       glPopMatrix();
         end;

         glPushMatrix();
          glTranslate(0.7, 0, 0.8);
          glRotate(50, 0, 1, 0);
          glRotate(20, 0, 0, 1);
          glRotate(rotangle1 * 0.5, 1, 0, 0);

          thumbLen := 1.2;

          gluCylinder(Cylinder, 0.14, 0.12, thumbLen * 0.4, 12, 4);
          glTranslate(0, 0, thumbLen * 0.4);
          gluSphere(Sphere, 0.12, 8, 8);

          glRotate(rotangle2 * 0.5, 1, 0, 0);
          gluCylinder(Cylinder, 0.12, 0.10, thumbLen * 0.35, 12, 4);
          glTranslate(0, 0, thumbLen * 0.35);
          gluSphere(Sphere, 0.10, 8, 8);

          glRotate(rotangle3 * 0.5, 1, 0, 0);
          gluCylinder(Cylinder, 0.10, 0.09, thumbLen * 0.25, 12, 4);
          glTranslate(0, 0, thumbLen * 0.25);
          gluSphere(Sphere, 0.09, 8, 8);
         glPopMatrix();

        glPopMatrix();

       glPopMatrix();

      SwapBuffers(myDC);
    end;
```

This procedure handles the 3D visualization of the arm using OpenGL. It sets up the viewing perspective based on camera variables (yaw, pitch, roll, position). The arm is constructed using a hierarchical transformation approach. The global coordinate system is pushed onto the stack. First, the shoulder joint sphere is drawn, and the coordinate system is rotated by the shoulder angle (th1) to draw the upper arm cylinder. The system is then translated to the end of the upper arm to draw the elbow. A relative rotation (th2 - th1) is applied for the forearm, ensuring the forearm moves relative to the upper arm. This process repeats for the wrist and hand. This hierarchy ensures that the movement of proximal segments (like the shoulder) correctly propagates to distal segments (elbow and hand), creating a realistic articulated animation.

(vi) BitBtnPassiveClick

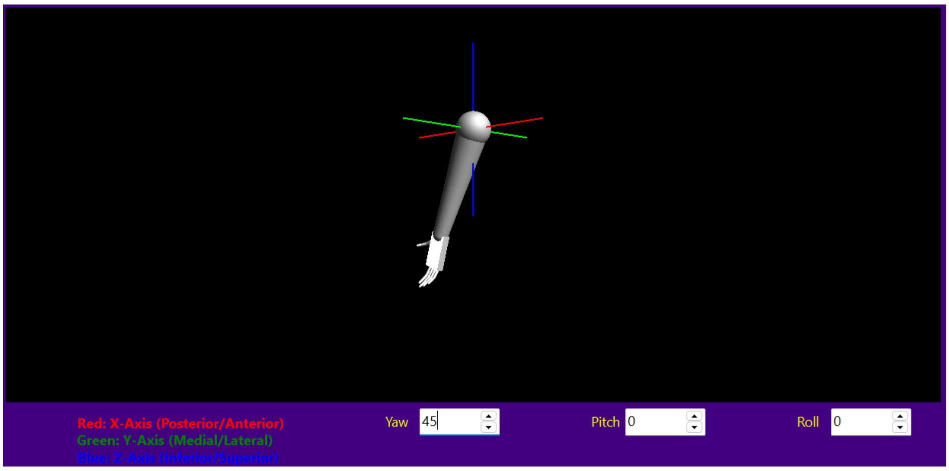
```
procedure TForm1.BitBtnPassiveClick(Sender: TObject);
var
  sh0, el0, wr0: Double;
begin
  Timer1.Enabled := False;
  FPassiveMode := True;
  FStillCount := 0;

  CalculateAnthropometry;

  // === Passive parameters (lebih realistis, tidak "runaway") ===
  // Damping cukup untuk menghabiskan energi perlahan
  ParamSh.c := 0.2;
  ParamEl.c := 0.3;
  ParamWr.c := 0.10;

  // Shoulder soft limits
  ParamSh.phi1 := -20.0;  // ext
  ParamSh.phi2 := 150.0;  // flex
  ParamSh.k1 := 2.5;
  ParamSh.k2 := 0.25;
  ParamSh.k3 := 2.0;
  ParamSh.k4 := 0.20;

  // Elbow soft limits (hindari hyperextension)
  ParamEl.phi1 := -5.0;
  ParamEl.phi2 := 150.0;
  ParamEl.k1 := 3.0;
  ParamEl.k2 := 0.35;
  ParamEl.k3 := 7.0;
  ParamEl.k4 := 0.45;

  // Wrist soft limits
  ParamWr.phi1 := 0;
  ParamWr.phi2 := 80.0;
  ParamWr.k1 := 1.8;
  ParamWr.k2 := 0.30;
  ParamWr.k3 := 1.8;
  ParamWr.k4 := 0.30;

  // Matikan active torque
  ParamSh.TargetFreq := 0; ParamSh.TargetAmp := 0; ParamSh.TargetOffset := 0; ParamSh.TargetPhase := 0;
  ParamEl.TargetFreq := 0; ParamEl.TargetAmp := 0; ParamEl.TargetOffset := 0; ParamEl.TargetPhase := 0;
  ParamWr.TargetFreq := 0; ParamWr.TargetAmp := 0; ParamWr.TargetOffset := 0; ParamWr.TargetPhase := 0;
  ParamSh.Kp := 0; ParamSh.Kd := 0;
  ParamEl.Kp := 0; ParamEl.Kd := 0;
  ParamWr.Kp := 0; ParamWr.Kd := 0;

  // (Opsional) tampilkan parameter pasif ke edit box supaya user lihat nilainya (tidak ubah GUI)
  edS_c.Text  := FloatToStr(ParamSh.c);
  edS_k1.Text := FloatToStr(ParamSh.k1);
  edS_k2.Text := FloatToStr(ParamSh.k2);
  edS_p1.Text := FloatToStr(ParamSh.phi1);
  edS_k3.Text := FloatToStr(ParamSh.k3);
  edS_k4.Text := FloatToStr(ParamSh.k4);
  edS_p2.Text := FloatToStr(ParamSh.phi2);

  edE_c.Text  := FloatToStr(ParamEl.c);
  edE_k1.Text := FloatToStr(ParamEl.k1);
  edE_k2.Text := FloatToStr(ParamEl.k2);
  edE_p1.Text := FloatToStr(ParamEl.phi1);
  edE_k3.Text := FloatToStr(ParamEl.k3);
  edE_k4.Text := FloatToStr(ParamEl.k4);
  edE_p2.Text := FloatToStr(ParamEl.phi2);

  edW_c.Text  := FloatToStr(ParamWr.c);
  edW_k1.Text := FloatToStr(ParamWr.k1);
  edW_k2.Text := FloatToStr(ParamWr.k2);
  edW_p1.Text := FloatToStr(ParamWr.phi1);
  edW_k3.Text := FloatToStr(ParamWr.k3);
  edW_k4.Text := FloatToStr(ParamWr.k4);
  edW_p2.Text := FloatToStr(ParamWr.phi2);

  // Initial angles (derajat). Kalau user masih 0 semua, beri default yang "jatuh" terlihat
  sh0 := ReadVal(edInitSh, 0);
  el0 := ReadVal(edInitEl, 0);
  wr0 := ReadVal(edInitWr, 0);

  if (Abs(sh0) < 1e-6) and (Abs(el0) < 1e-6) and (Abs(wr0) < 1e-6) then
  begin
    sh0 := 35;  // bahu agak terangkat
    el0 := 20;  // siku sedikit menekuk
    wr0 := 10;  // wrist sedikit fleksi
    edInitSh.Text := FloatToStr(sh0);
    edInitEl.Text := FloatToStr(el0);
    edInitWr.Text := FloatToStr(wr0);
  end;
```

```

th1 := DegToRad(sh0);
th2 := DegToRad(el0);
th3 := DegToRad(wr0);

dth1 := 0; dth2 := 0; dth3 := 0;
ddth1 := 0; ddth2 := 0; ddth3 := 0;

SimTime := 0;

// Step & timer (stabil)
Timer1.Interval := 2;
    dt := 0.002;

    SeriesShoulder.Clear;
    SeriesElbow.Clear;
    SeriesWrist.Clear;

    Timer1.Enabled := True;
  end;
```

This procedure sets up and starts the "Passive Test" simulation mode. It initializes the anthropometric data and then specifically configures the joint parameters for a passive scenario. Crucially, it sets the active control gains (K_p, K_d) to zero, ensuring no muscle force is generated. It assigns values for passive damping (c) and limit spring constants (k1-k4, phi1-phi2) to simulate joint friction and physical constraints (ligaments). It reads the initial joint angles from the user interface (edInitSh, etc.), converts them to radians, and sets the initial velocities to zero. Finally, it enables the simulation timer, allowing the arm to move solely under the influence of gravity and passive viscoelastic forces, typically demonstrating a decaying oscillation towards an equilibrium position.

c. 3D Coordinate System Plotting

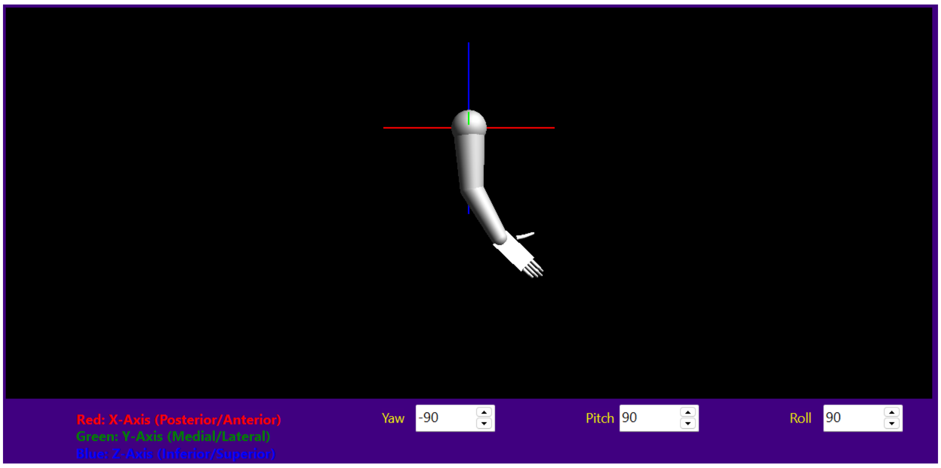
The visualization module utilizing OpenGL renders the triple pendulum arm within a defined three-dimensional Cartesian space to provide immediate visual feedback on the simulation's physical behavior. **Figure 2** ("Initial POV of the Arm Model") illustrates the default state of the system upon initialization. To facilitate spatial orientation, a global reference frame is drawn using color-coded axes: the Red line represents the X-axis (Posterior-Anterior direction), the Green line represents the Y-axis (Medial-Lateral direction), and the Blue line represents the Z-axis (Inferior-Superior direction). In this initial configuration, the arm segments, Upper Arm, Forearm, and Hand, are aligned vertically downward, corresponding to the equilibrium position where all joint angles (θ_1, θ_2, θ_3) are effectively zero. This visual confirmation is essential to verify that the hierarchical geometric transformations defined in the RenderScene procedure are correctly determining the origin and orientation of each segment before any dynamic forces (gravity or muscle torque) are applied.


Red: X-Axis (Posterior/Anterior)
Green: Y-Axis (Medial/Lateral)
Blue: Z-Axis (Inferior/Superior)

Yaw 45
Pitch 0
Roll 0

Figure 2. *Initial POV of the Arm Model*

Figure 3 ("The POV of the Arm Model from Different Axes Values") demonstrates the interactive inspection capabilities of the simulation environment. By manipulating the Yaw, Pitch, and Roll parameters via the user interface, the observer can rotate the camera perspective around the model. The software implements these viewing transformations using standard OpenGL rotation commands (glRotate) applied to the model-view matrix. This feature is critical for validating the biomechanical constraints of the model; for instance, by rotating the view to a lateral perspective (e.g., Yaw = 90°), the user can visually confirm that the arm's movement is strictly confined to the sagittal plane as dictated by the 2D planar motion equations, ensuring that no erroneous lateral movements or mathematical divergences are occurring during the simulation.


Red: X-Axis (Posterior/Anterior)
Green: Y-Axis (Medial/Lateral)
Blue: Z-Axis (Inferior/Superior)

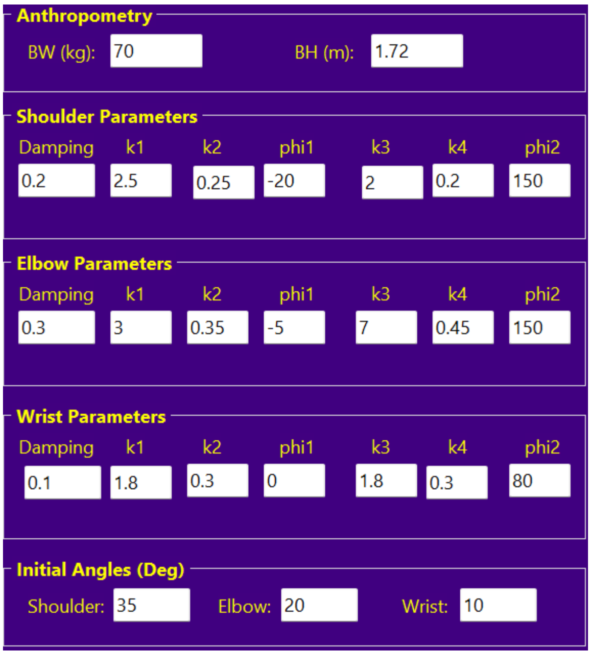
Yaw -90
Pitch 90
Roll 90

Figure 3. *The POV of the Arm Model from Different Axes Values*

d. Experimentation

- ***Passive Test***

- *Normal*



Anthropometry

BW (kg): 70   BH (m): 1.72

Shoulder Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.2	2.5	0.25	-20	2	0.2	150

Elbow Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.3	3	0.35	-5	7	0.45	150

Wrist Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.1	1.8	0.3	0	1.8	0.3	80

Initial Angles (Deg)

Shoulder: 35   Elbow: 20   Wrist: 10

Figure 4. *Passive Test Joint Angle Charts on Normal Condition*

This figure shows the passive-mode setup with moderate damping (Shoulder 0.2, Elbow 0.3, Wrist 0.1) and meaningful passive stiffness parameters ($k_1 – k_4$) plus realistic joint boundaries (Φ_1, Φ_2). The initial angles (Shoulder 35°, Elbow 20°, Wrist 10°) provide enough gravitational potential to start a natural “drop-and-swing.”

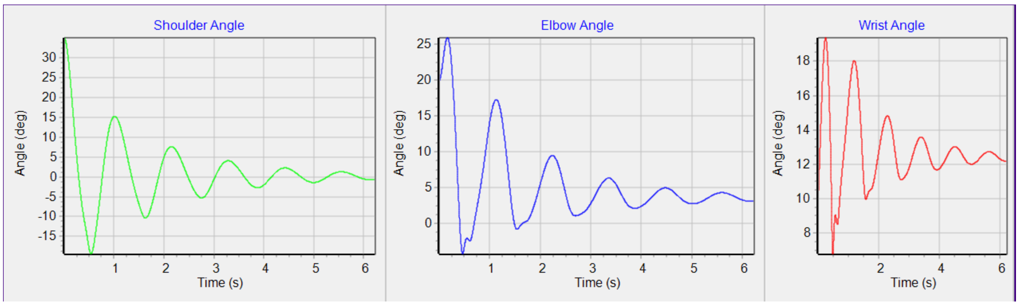
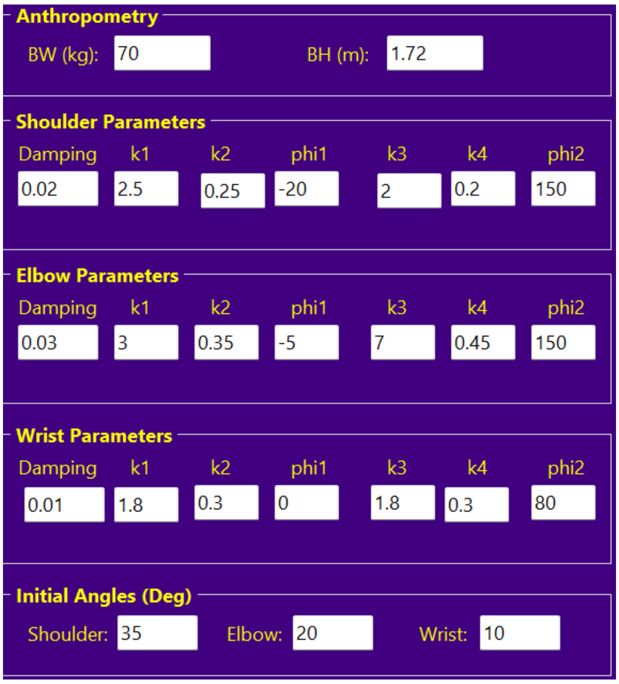


Figure 5. *Passive Test Joint Angle Charts on Normal Condition*

The charts show a realistic decaying oscillation: the arm falls under gravity, overshoots, then continues to oscillate with decreasing amplitude until settling. This is the expected behavior for a physically plausible passive system, energy is gradually dissipated by damping, but motion does not stop instantly.

- *Underdamped*



Anthropometry

BW (kg): 70 BH (m): 1.72

Shoulder Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.02	2.5	0.25	-20	2	0.2	150

Elbow Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.03	3	0.35	-5	7	0.45	150

Wrist Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.01	1.8	0.3	0	1.8	0.3	80

Initial Angles (Deg)

Shoulder: 35 Elbow: 20 Wrist: 10

Figure 6. *Active Test Parameters Input on Underdamped Condition*

Here the damping coefficients are reduced drastically (Shoulder 0.02, Elbow 0.03, Wrist 0.01). With damping this small, the system removes very little energy per cycle, so oscillations persist and can become very sensitive to coupling effects between links.

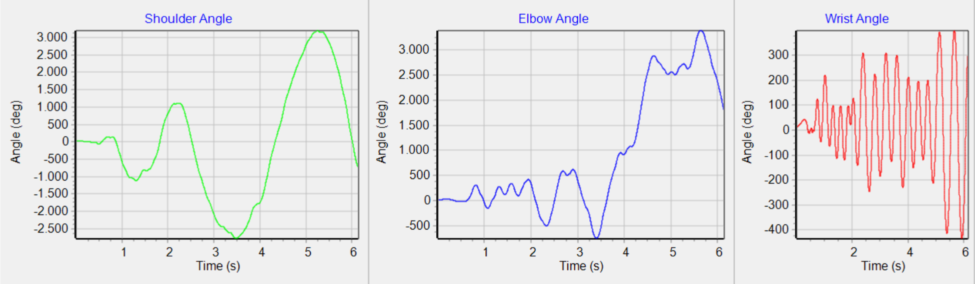
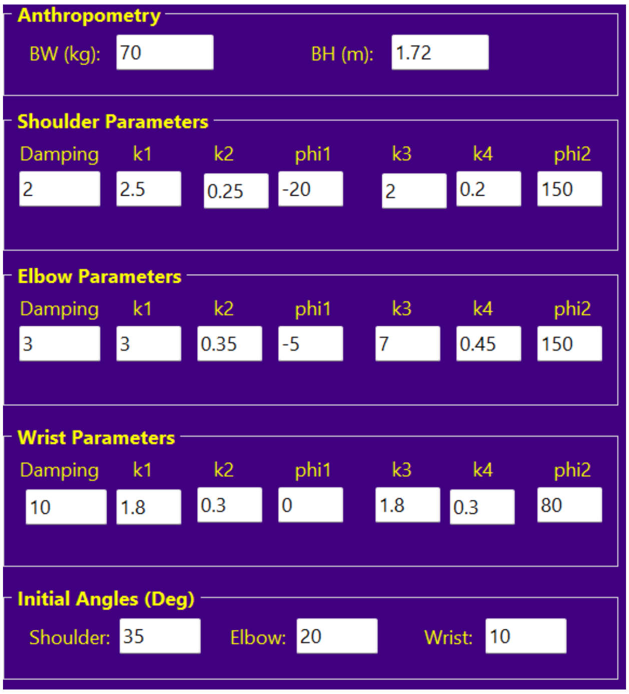


Figure 7. *Passive Test Joint Angle Charts on Underdamped Condition*

The plots show very large oscillations and even runaway growth (angles exploding to extremely high values). This is an underdamped "worst-case": the joints keep exchanging energy, and with near-zero dissipation the motion can become unstable (and may also indicate the passive exponential torques are injecting energy near the limits or the integration step is being pushed too hard). Practically, this condition looks not realistic for human passive motion because the arm would not spin uncontrollably like this.

- *Overdamped*



Anthropometry

BW (kg): 70   BH (m): 1.72

Shoulder Parameters

Damping	k1	k2	phi1	k3	k4	phi2
2	2.5	0.25	-20	2	0.2	150

Elbow Parameters

Damping	k1	k2	phi1	k3	k4	phi2
3	3	0.35	-5	7	0.45	150

Wrist Parameters

Damping	k1	k2	phi1	k3	k4	phi2
10	1.8	0.3	0	1.8	0.3	80

Initial Angles (Deg)

Shoulder: 35   Elbow: 20   Wrist: 10

Figure 8. *Passive Test Parameters Input on Overdamped Condition*

In this condition, damping is increased heavily (Shoulder 2, Elbow 3, Wrist 10). This makes the joints behave as if moving through a thick viscous medium—momentum is suppressed quickly and oscillation should be strongly reduced.

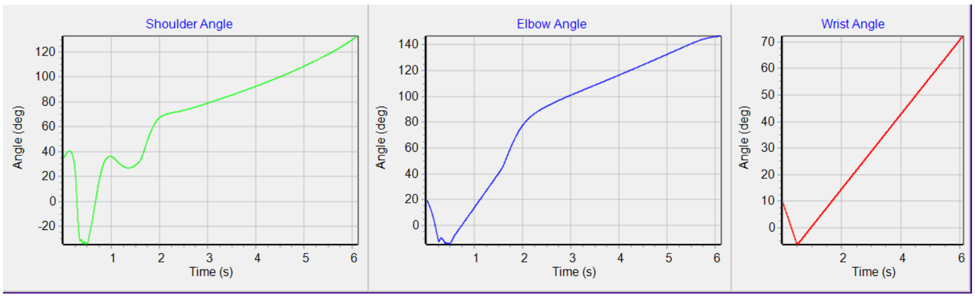
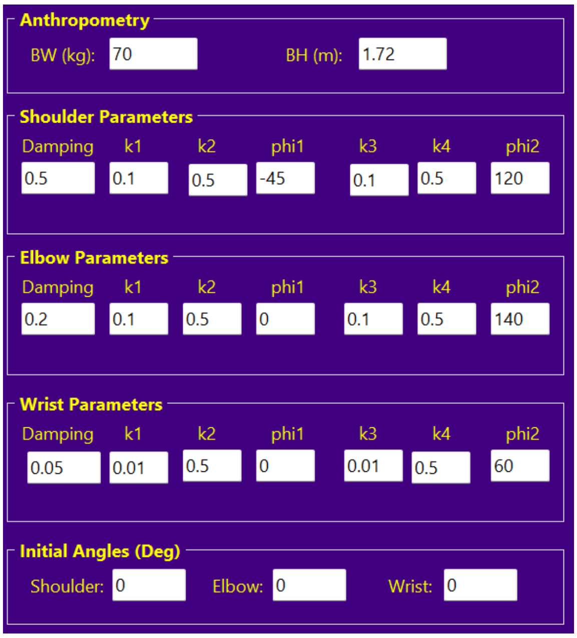


Figure 9. *Passive Test Joint Angle Charts on Overdamped Condition*

The charts show motion that becomes highly non-oscillatory (little to no swinging). Instead of oscillating, the angles trend more monotonically toward a new posture, because damping dominates the dynamics and prevents inertia-driven overshoot. This looks stable, but it can feel too stiff / too quickly damped to resemble a natural passive arm swing.

- ***Active Test***

- *Condition 1*



Anthropometry

BW (kg): 70 BH (m): 1.72

Shoulder Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.5	0.1	0.5	-45	0.1	0.5	120

Elbow Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.2	0.1	0.5	0	0.1	0.5	140

Wrist Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.05	0.01	0.5	0	0.01	0.5	60

Initial Angles (Deg)

Shoulder: 0 Elbow: 0 Wrist: 0

Figure 10. *Active Test Parameters Input on Condition 1*

In **Figures 10 and 11**, the simulation is run under "Condition 1" using the baseline anthropometry values (BW = 70 kg, BH = 1.72 m) and relatively light passive joint parameters (small k-values and low damping). The initial angles are set to 0° for shoulder, elbow, and wrist, meaning the arm starts from a neutral pose. The displayed inertia matrix $[M]$ is shown at the current posture; the off-diagonal terms (coupling) can change over time because they depend on the relative joint angles through cosine terms.

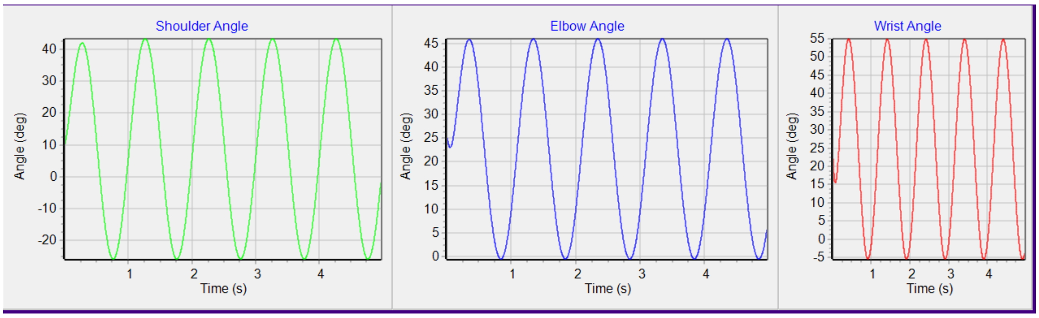
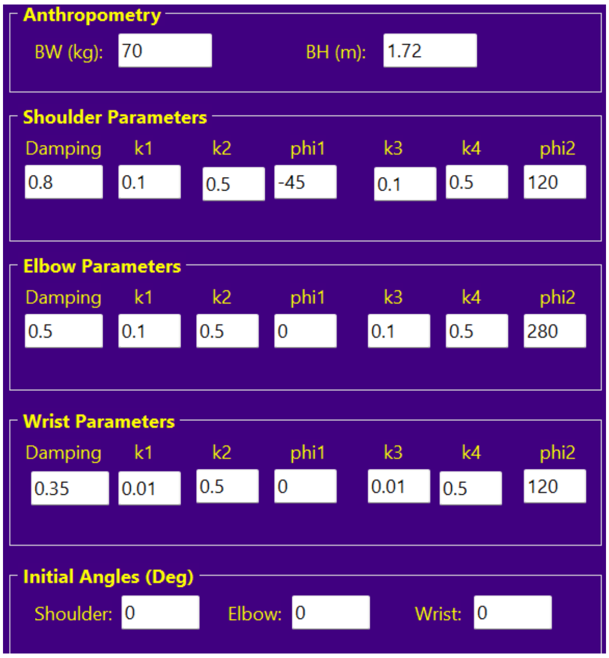


Figure 11. *Active Test Joint Angle Charts on Condition 1*

The resulting charts in **Figure 11** show smooth, periodic oscillations at a consistent frequency, indicating that the active controller is successfully driving the joints to follow the intended trajectory. The shoulder oscillates around a mid-position, followed by the elbow and wrist with visible phase delays (the distal joints lag behind the proximal joint). This phase lag illustrates the "whip-like" behavior of a coupled multi-link pendulum: motion initiated at the shoulder propagates to the elbow and wrist through inertial coupling, producing coordinated oscillations without numerical blow-up.

- *Condition 2*



Anthropometry

BW (kg):	70	BH (m):	1.72

Shoulder Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.8	0.1	0.5	-45	0.1	0.5	120

Elbow Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.5	0.1	0.5	0	0.1	0.5	280

Wrist Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.35	0.01	0.5	0	0.01	0.5	120

Initial Angles (Deg)

Shoulder:	0	Elbow:	0	Wrist:	0

Figure 12. *Active Test Parameters Input on Condition 2*

In **Figures 12 and 13**, the simulation is pushed to “Condition 2” by increasing several passive settings, most noticeably the damping values (e.g., shoulder and wrist damping are higher) and expanding the upper joint limit parameters (larger ϕ_2). Conceptually, this combination makes the model dissipate energy faster (via damping) while also reducing the chance of the passive exponential “end-stop” torque interfering within the normal motion range (because the flexion limit is moved farther away).

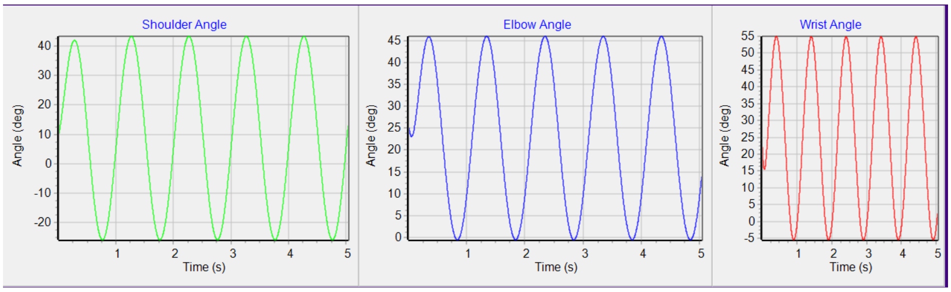
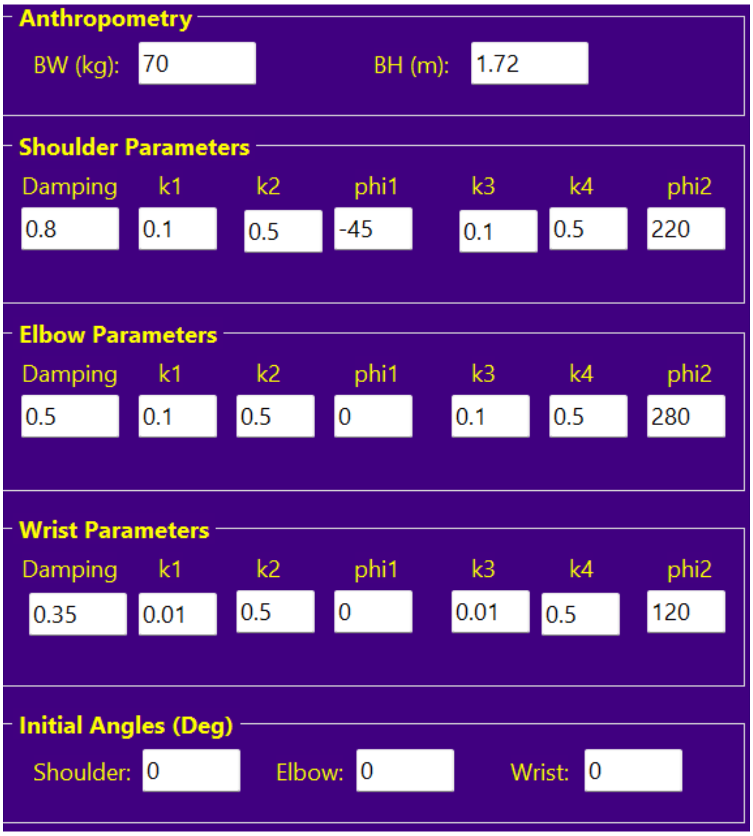


Figure 13. *Active Test Joint Angle Charts on Condition*

The charts in **Figure 13** still show stable oscillatory motion with clear periodicity, demonstrating that the system remains well-behaved even with stronger passive effects. Compared to Condition 1, the increased damping tends to suppress overly sharp transients and makes the motion look slightly “heavier” (less abrupt). The shoulder–elbow–wrist phase differences remain apparent, again reflecting the coupled dynamics where the forearm and hand respond after the upper arm due to inertial coupling and the active driving terms.

- *Condition 3*



Anthropometry

BW (kg): 70

BH (m): 1.72

Shoulder Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.8	0.1	0.5	-45	0.1	0.5	220

Elbow Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.5	0.1	0.5	0	0.1	0.5	280

Wrist Parameters

Damping	k1	k2	phi1	k3	k4	phi2
0.35	0.01	0.5	0	0.01	0.5	120

Initial Angles (Deg)

Shoulder: 0

Elbow: 0

Wrist: 0

Figure 12. *Active Test Parameters Input on Condition 2*

In **Figures 12 and 13**, the simulation is pushed to “Condition 2” by increasing several passive settings, most noticeably the damping values (e.g., shoulder and wrist damping are higher) and expanding the upper joint limit parameters (larger Φ_2). Conceptually, this combination makes the model dissipate energy faster (via damping) while also reducing the chance of the passive exponential “end-stop” torque interfering within the normal motion range (because the flexion limit is moved farther away).

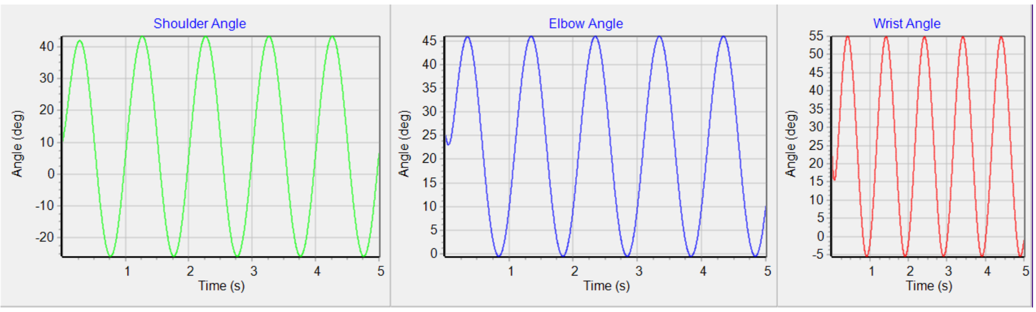


Figure 13. *Active Test Joint Angle Charts on Condition*

The charts in **Figure 13** still show stable oscillatory motion with clear periodicity, demonstrating that the system remains well-behaved even with stronger passive effects. Compared to Condition 1, the increased damping tends to suppress overly sharp transients and makes the motion look slightly “heavier” (less abrupt). The shoulder–elbow–wrist phase differences remain apparent, again reflecting the coupled dynamics where the forearm and hand respond after the upper arm due to inertial coupling and the active driving terms.

2. Knee Extension

a. Motion Equation Derivation (New)

- **Anthropometric Parameters**

First, I establish the physical properties of the segment (shank + foot) based on the provided subject data.

Subject Data:

- Body Weight (BW) = 70 kg
- Body Height (BH) = 172 cm

Calculated Parameters:

- Mass (m): $m = \frac{2.76 \times BW}{60} = \frac{2.76 \times 70}{60} \approx 3.22 \text{ kg}$
- Segment Length (L): $L = \frac{0.407 \times BH}{160} = \frac{0.407 \times 172}{160} \approx 0.4375 \text{ m}$
- Inertia Parameter (I), defined in the system as the moment of inertia for a rod about its end:

$$I = \frac{1}{3}mL^2 \approx 0.205 \text{ kg} \cdot \text{m}^2$$

- **Kinematics and Coordinate System**

I model the knee extension as a pendulum moving in 3D spherical coordinates, though the primary movement is in the sagittal plane (θ). I define the Center of Mass (COM) of the shank to be located at the midpoint of the segment length ($L/2$).

Coordinate Definition:

- r: Radial distance to COM ($r = L/2$).
- θ: Polar angle (flexion/extension angle from the vertical).
- ϕ: Azimuthal angle (abduction/adduction or external rotation).

Position Vector ($\vec{P}_{cm}$):

Using rotation matrices to transform from the origin (knee joint) to the Center of Mass. The position vector in Cartesian coordinates (x, y, z) is derived as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{L}{2}\sin\theta\cos\phi \\ \frac{L}{2}\sin\theta\sin\phi \\ -\frac{L}{2}\cos\theta \end{bmatrix}$$

Velocity Vector:

Differentiating the position with respect to time (t) to find the velocity of the COM:

$$\begin{aligned}\dot{x} &= \frac{L}{2}(\dot{\theta}\cos\,\theta\cos\,\phi - \dot{\phi}\sin\,\theta\sin\,\phi)\\ \dot{y} &= \frac{L}{2}(\dot{\theta}\cos\,\theta\sin\,\phi + \dot{\phi}\sin\,\theta\cos\,\phi)\\ \dot{z} &= \frac{L}{2}\dot{\theta}\sin\,\theta\end{aligned}$$

Velocity Squared (v^2), the square of the linear velocity magnitude is required for Kinetic Energy. After simplifying trigonometric identities ($\sin^2 + \cos^2 = 1$):

$$v^2 = \dot{x}^2 + \dot{y}^2 + \dot{z}^2 = \frac{L^2}{4}(\dot{\theta}^2 + \dot{\phi}^2\sin^2\theta)$$

- **Lagrangian Mechanics**

I use the Lagrangian approach $\mathcal{L} = E_k - E_p$ to derive the equation of motion.

- Kinetic Energy (E_k)

The total kinetic energy is the sum of translational energy of the COM and rotational energy about the COM.

$$E_k = \frac{1}{2}mv^2 + \frac{1}{2}I\dot{\theta}^2$$

Substituting v^2:

$$E_k = \frac{1}{2}m\left[\frac{L^2}{4}(\dot{\theta}^2 + \dot{\phi}^2\sin\theta^2)\right] + \frac{1}{2}I\dot{\theta}^2$$

$$E_k = \frac{1}{2}(\frac{mL^2}{4} + I)\dot{\theta}^2 + \frac{1}{2}(\frac{mL^2}{4})\dot{\phi}^2\sin^2\theta$$

- Potential Energy (E_p)

Assuming the datum is at the pivot point, the potential energy depends on the vertical height z of the COM.

$$E_p = mgz = mg\left(-\frac{L}{2}\cos\theta\right)$$

(Note: Depending on the reference frame, this can be written as $mg\frac{L}{2}(1 - \cos\,\theta)$*).*

- The Lagrangian ($\mathcal{L}$)

$$\mathcal{L} = E_k - E_p$$

$$\mathcal{L} = [\frac{1}{2}(\frac{mL^2}{4} + I)\dot{\theta}^2 + \frac{1}{8}mL^2\dot{\phi}^2\sin^2\theta] - [-mg\frac{L}{2}\cos\,\theta]$$

- **Derivation of the Motion Equation**

I apply the Euler-Lagrange equation for the coordinate θ:

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{\theta}}\right) - \frac{\partial\mathcal{L}}{\partial\theta} = \tau_{net}$$

Step 1: Partial derivative with respect to velocity ($\dot{\theta}$)

$$\frac{\partial \mathcal{L}}{\partial \dot{\theta}} = 2 \cdot \frac{1}{2}(\frac{mL^2}{4} + I)\dot{\theta} = (\frac{mL^2}{4} + I)\dot{\theta}$$

Step 2: Time derivative

$$\frac{d}{dt}(\frac{\partial \mathcal{L}}{\partial \dot{\theta}}) = (\frac{mL^2}{4} + I)\ddot{\theta}$$

Step 3: Partial derivative with respect to position (θ). I must differentiate the $\sin^2\theta$ term in E_k and the $\cos\theta$ term in E_p.

Derivative of $\sin^2\theta$ is $2\sin\theta\cos\theta$. Derivative of $-\cos\theta$ is $\sin\theta$.

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{8}mL^2\dot{\phi}^2(2\sin\theta\cos\theta) - mg\frac{L}{2}\sin\theta$$

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{4}mL^2\dot{\phi}^2\sin\theta\cos\theta - \frac{1}{2}mgL\sin\theta$$

Step 4: Assembling the Equation

$$\left[\left(\frac{mL^2}{4} + I\right)\ddot{\theta}\right] - \left[\frac{1}{4}mL^2\dot{\phi}^2\sin\theta\cos\theta - \frac{1}{2}mgL\sin\theta\right] = \tau_{net}$$

Rearranging to isolate the inertial term:

$$(\frac{mL^2}{4} + I)\ddot{\theta} = \tau_{net} + \frac{1}{4}mL^2\dot{\phi}^2\sin\theta\cos\theta - \frac{1}{2}mgL\sin\theta$$

- **Generalized Forces (Torque Model)**

The net torque (τ_{net}) is defined in the system as a combination of Active Torque (Muscle) and Passive Torque (Elastic/Viscous components of the joint).

Passive Torque ($\tau_{passive}$). Modeled using a viscosity coefficient c and exponential springs to represent joint limits (extension and flexion):

$$\tau_{passive} = -c\dot{\theta} + k_1e^{-k_2(\theta-\phi_1)} - k_3e^{-k_4(\phi_2-\theta)}$$

$-c\dot{\theta}$: Damping term.

$k_1e^{\cdots}$: Exponential spring force preventing hyperextension.

$k_3e^{\cdots}$: Exponential spring force preventing hyperflexion.

Active Torque (τ_{active}). $\tau_{active} = F_{mus} \times \text{Moment Arm}$, where F_{mus} is derived from muscle activation dynamics, force-length relationship $f(l)$, and force-velocity relationship $f(v)$.

Total Torque:

$$\tau_{net} = \tau_{active} + \tau_{passive}$$

- **Final Acceleration Equation ($\ddot{\theta}$)**

Finally, I isolate the angular acceleration $\ddot{\theta}$ (theta-double-dot) to be used in the Runge-Kutta integration solver.

$$\ddot{\theta} = \frac{\tau_{net} + \left(\frac{mL^2}{8}\dot{\phi}^2 \cdot 2\sin\theta\cos\theta\right) - \left(\frac{mgL}{2}\sin\theta\right)}{\left(\frac{mL^2}{4} + I\right)}$$

This matches the implementation in the simulation loop, where:

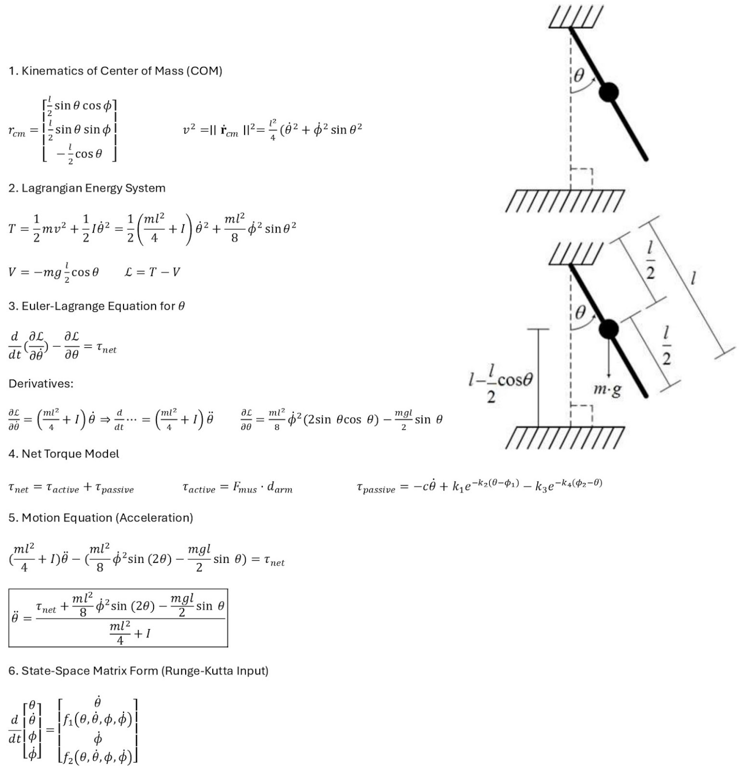
- Numerator:

τ_{net} (Torque input), Centrifugal term due to 3D rotation: $\frac{mL^2}{8}\dot{\phi}^2\sin(2\theta)$, Gravity moment: $-\frac{mgL}{2}\sin\theta$

- Denominator:

Total effective inertia: $\frac{mL^2}{4} + I$. And the differential equation matrix form for the state vector $X = [\theta, \dot{\theta}, \phi, \dot{\phi}]^T$ is solved numerically.

Figure 1 below showed the general mechanism of the knee joint extension motion equation that the previous method derived.



1. Kinematics of Center of Mass (COM)

$$r_{cm} = \begin{bmatrix} \frac{l}{2}\sin\theta\cos\phi \\ \frac{l}{2}\sin\theta\sin\phi \\ -\frac{l}{2}\cos\theta \end{bmatrix} \qquad v^2 = \|\dot{\mathbf{r}}_{cm}\|^2 = \frac{l^2}{4}(\dot{\theta}^2 + \dot{\phi}^2\sin\theta^2$$

2. Lagrangian Energy System

$$T = \frac{1}{2}mv^2 + \frac{1}{2}I\dot{\theta}^2 = \frac{1}{2}\left(\frac{ml^2}{4} + I\right)\dot{\theta}^2 + \frac{ml^2}{8}\dot{\phi}^2\sin\theta^2$$

$$V = -mg\frac{l}{2}\cos\theta \qquad \mathcal{L} = T - V$$

3. Euler-Lagrange Equation for θ

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{\theta}}\right) - \frac{\partial\mathcal{L}}{\partial\theta} = \tau_{net}$$

Derivatives:

$$\frac{\partial\mathcal{L}}{\partial\dot{\theta}} = \left(\frac{ml^2}{4} + I\right)\dot{\theta} \Rightarrow \frac{d}{dt}\cdots = \left(\frac{ml^2}{4} + I\right)\ddot{\theta} \qquad \frac{\partial\mathcal{L}}{\partial\theta} = \frac{ml^2}{8}\dot{\phi}^2(2\sin\theta\cos\theta) - \frac{mgl}{2}\sin\theta$$

4. Net Torque Model

$$\tau_{net} = \tau_{active} + \tau_{passive} \qquad \tau_{active} = F_{mus} \cdot d_{arm} \qquad \tau_{passive} = -c\dot{\theta} + k_1 e^{-k_2(\theta-\phi_1)} - k_3 e^{-k_4(\phi_2-\theta)}$$

5. Motion Equation (Acceleration)

$$\left(\frac{ml^2}{4} + I\right)\ddot{\theta} - \left(\frac{ml^2}{8}\dot{\phi}^2\sin(2\theta) - \frac{mgl}{2}\sin\theta\right) = \tau_{net}$$

$$\boxed{\ddot{\theta} = \frac{\tau_{net} + \frac{ml^2}{8}\dot{\phi}^2\sin(2\theta) - \frac{mgl}{2}\sin\theta}{\frac{ml^2}{4} + I}}$$

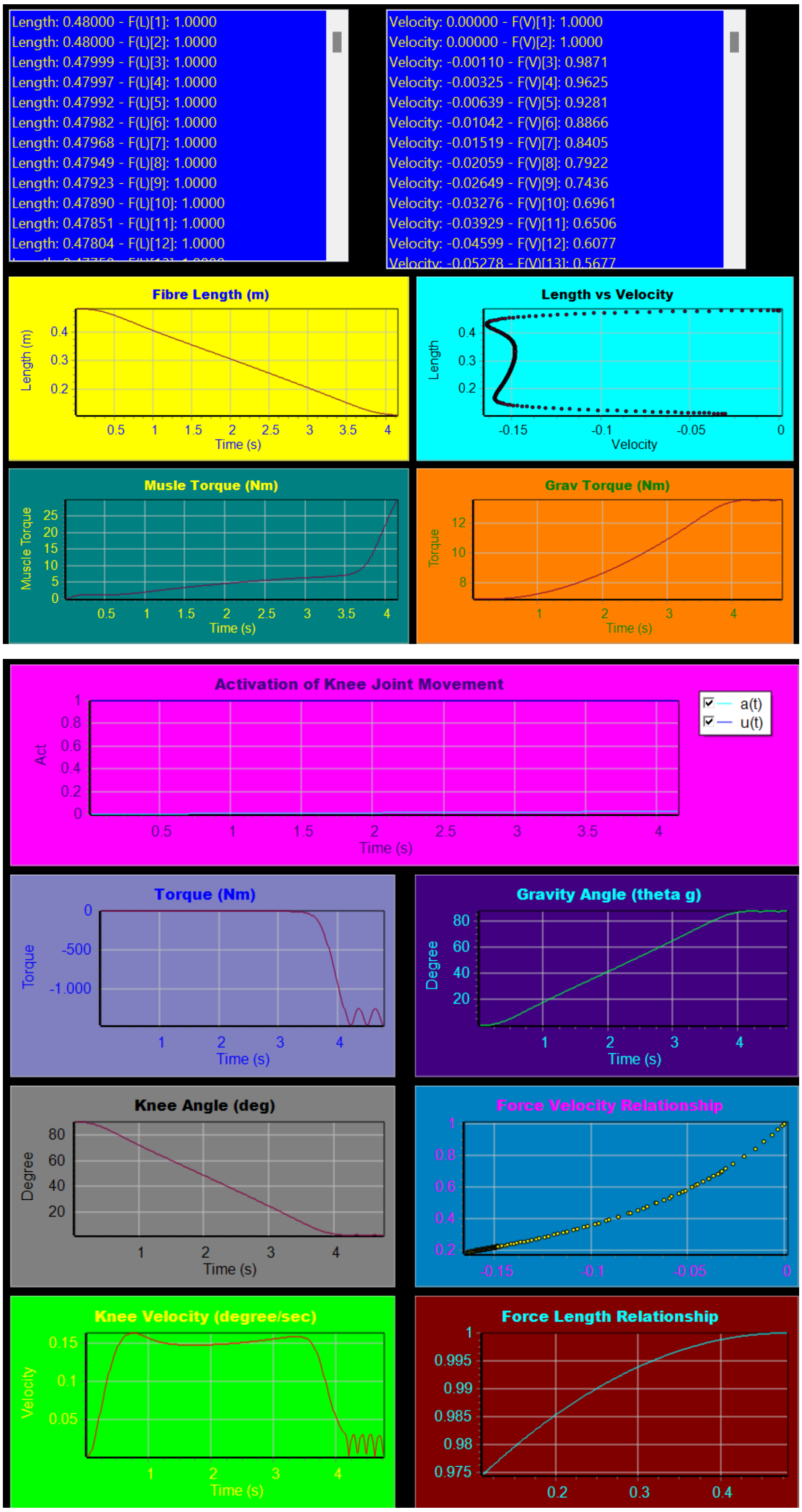
6. State-Space Matrix Form (Runge-Kutta Input)

$$\frac{d}{dt}\begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ f_1(\theta, \dot{\theta}, \phi, \dot{\phi}) \\ \dot{\phi} \\ f_2(\theta, \dot{\theta}, \phi, \dot{\phi}) \end{bmatrix}$$

Figure 1. *Knee Joint Extension Motion Equation Derivation*

b. Experimentation

- Active Knee Joint Extension



```
Length: 0.48000 - F(L)[1]: 1.0000
Length: 0.48000 - F(L)[2]: 1.0000
Length: 0.47999 - F(L)[3]: 1.0000
Length: 0.47997 - F(L)[4]: 1.0000
Length: 0.47992 - F(L)[5]: 1.0000
Length: 0.47982 - F(L)[6]: 1.0000
Length: 0.47968 - F(L)[7]: 1.0000
Length: 0.47949 - F(L)[8]: 1.0000
Length: 0.47923 - F(L)[9]: 1.0000
Length: 0.47890 - F(L)[10]: 1.0000
Length: 0.47851 - F(L)[11]: 1.0000
Length: 0.47804 - F(L)[12]: 1.0000
```

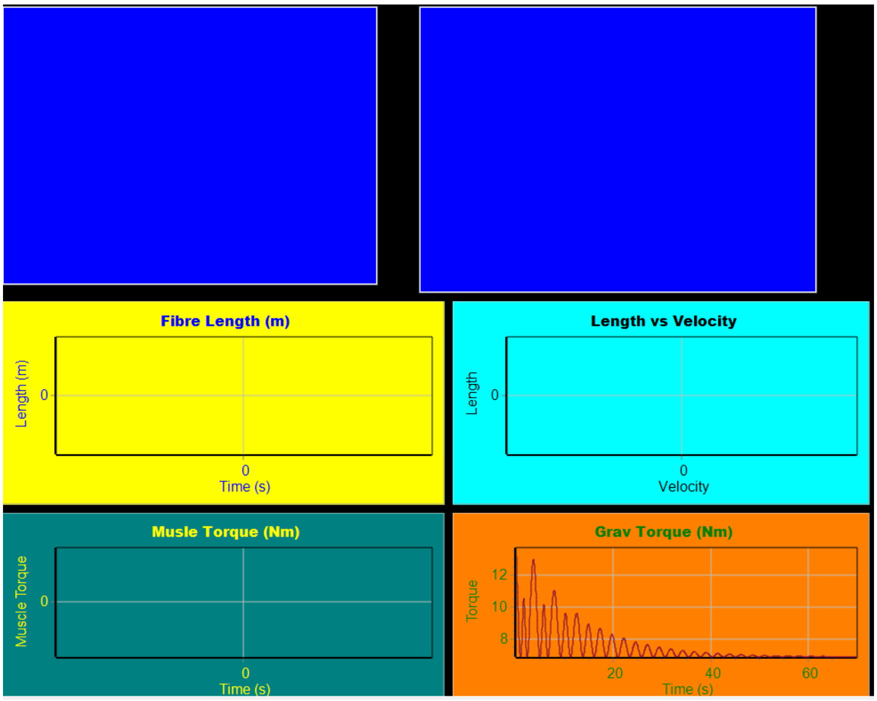
```
Velocity: 0.00000 - F(V)[1]: 1.0000
Velocity: 0.00000 - F(V)[2]: 1.0000
Velocity: -0.00110 - F(V)[3]: 0.9871
Velocity: -0.00325 - F(V)[4]: 0.9625
Velocity: -0.00639 - F(V)[5]: 0.9281
Velocity: -0.01042 - F(V)[6]: 0.8866
Velocity: -0.01519 - F(V)[7]: 0.8405
Velocity: -0.02059 - F(V)[8]: 0.7922
Velocity: -0.02649 - F(V)[9]: 0.7436
Velocity: -0.03276 - F(V)[10]: 0.6961
Velocity: -0.03929 - F(V)[11]: 0.6506
Velocity: -0.04599 - F(V)[12]: 0.6077
Velocity: -0.05278 - F(V)[13]: 0.5677
```

Figure 2. *Active Knee Joint Extension Charts*

Figure 2 presents the comprehensive dataset for the Active Knee Extension, where the motion is driven by the Hill-type muscle model integrated into the Lagrangian dynamics derived earlier. The driving force of this simulation is visible in the Activation chart, which shows a step function where the muscle activation signal (a(t)) rises from zero to a positive value. This input triggers the internal muscle dynamics, resulting in the generation of active force. Consequently, the Muscle Torque chart displays a corresponding rise in torque, which acts to overcome the inertia of the shank-foot segment and the gravitational moment. This is directly correlated with the Knee Angle chart, where we observe the limb extending, starting from a flexed position (approximately 90 degrees or vertical) and moving towards a more extended angle (plateauing as it reaches equilibrium between muscle torque and gravity). The Knee Velocity chart shows a positive spike during the concentric contraction phase as the leg accelerates upward, followed by a return to zero as the leg holds its position.

Furthermore, the internal verification of the Hill-type model is explicitly demonstrated in the bottom row charts. The Force-Velocity Relationship chart accurately plots the hyperbolic curve characteristic of concentric muscle action, where force generation capacity decreases as shortening velocity increases. Simultaneously, the Force-Length Relationship chart displays the parabolic nature of the active force capability, peaking at the optimal fiber length (L_opt). The Length vs Velocity chart serves as a phase plane plot, visualizing the state-space trajectory of the muscle fibers during the movement. The Total Torque chart represents the net algebraic sum of the active muscle torque, gravitational torque, and passive joint torques; its fluctuations stabilize as the limb reaches a steady state. The Grav Torque chart confirms the physics of the rotating reference frame, showing how the gravitational moment increases as the leg extends and the moment arm of the center of mass increases relative to the pivot point.

- Passive Movement



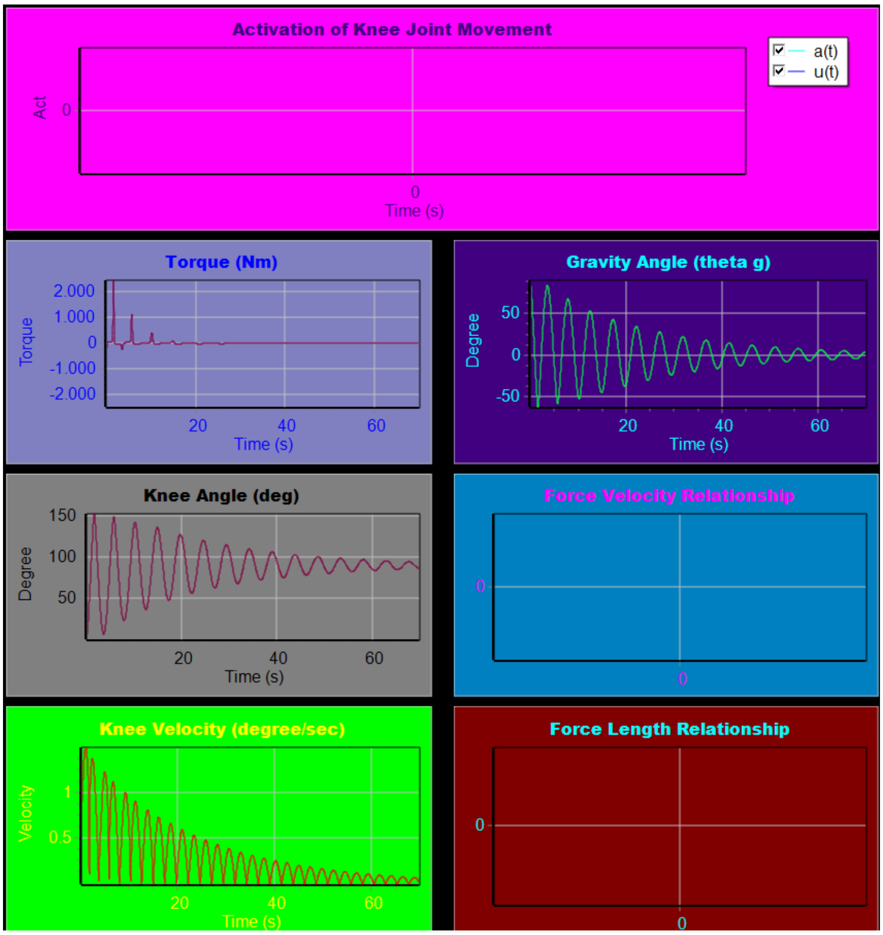


Figure 3. Passive Knee Joint Charts

Figure 3 illustrates the system's behavior in Passive Mode, effectively modeling the shank as a physical pendulum subject to gravity and viscoelastic damping without any active contractile contribution. The most critical indicator of this mode is the Activation chart and the Muscle Torque chart, both of which remain flat at zero throughout the simulation time. This confirms that the Hill-type muscle model is disengaged, and no active force is driving the system. Consequently, the Force-Velocity and Force-Length charts are empty or flat, as these factors are only relevant when calculating active muscle force (F_mus), which is non-existent in this scenario.

Instead of a controlled extension, the Knee Angle chart reveals a classic underdamped harmonic oscillation. The leg is released from an initial displacement, swings past its equilibrium point, and gradually settles down. This decaying sinusoidal pattern is mirrored in the Knee Velocity chart, where the amplitude of the angular velocity decreases over time due to the energy dissipation caused by the damping coefficient (B or c in the motion equation τ_passive = -c(thetadot)...). The Total Torque chart in this figure is dominated entirely by the interaction between the gravitational restoring force and the passive viscous damping torque. The Grav Torque chart shows an oscillating waveform that corresponds to the changing position of the limb's center of mass as it swings back and forth. This figure serves as a validation of the underlying physics engine, confirming that the Lagrangian formulation correctly simulates the conservation of energy and its gradual dissipation through viscosity when no biological control signals are applied.