Journal of information and communication convergence engineering
http://jicce.org

J. Inf. Commun. Converg. Eng. 17(4): 255-260, Dec. 2019 Regular paper

3D Modeling and Balancing Control of Two-link Underactuated Robots using Matlab/Simulink

Dong Sang Yoo*

Department of Electrical, Electronic and Control Engineering, Hankyong National University, Anseong 17579, Korea

Abstract

A pendubot is a representative example of an underactuated system that has fewer actuators than the degree of freedom of the system. In this study, the characteristics of the pendubot are first reviewed; each part is then designed using Solidworks by dividing the pendubot into three parts: the base frame, first link frame, and second link frame. These three parts are then imported into the Simulink environment via a STEP file format, which is the standard protocol used in data exchange between CAD applications. A 3D model of the pendubot is then constructed using Simscape, and the usefulness of the 3D model is validated by a comparison with a dynamic equation derived using the Lagrangian formulation. A linearized model around an upright equilibrium position is finally obtained, and a sliding mode controller is designed based on the linear quadratic regulator. Simulation results showed that the designed controller effectively maintained upright balance of the pendubot in the presence of disturbance.

Index Terms: 3D Modeling, Balancing Control, Sliding Mode Control, Underactuated Robot

I. INTRODUCTION

Underactuated systems have fewer actuators than the degree of freedom of the system, and they are found in many robotic applications, such as walking robots, robotic hands with many joints, or robots used in hazardous environments [1]. They belong to the class of non-minimum systems because of severe nonlinearity and input/state limitations which make it hard to control. Therefore, the control of the inverted pendulum, which is one of underactuated systems, is the most attractive problem in controls, and the inverted pendulum is often used to demonstrate a variety of concepts relating to nonlinear control, such as stabilization or the postural balance of unstable systems. Predictive control is a good method among systematic methodologies for rectifying this challenging control problem [2].

The pendubot is a type of inverted pendulum, and the main problems associated with its use relate to swing up, posture balance, and trajectory tracking. The swing-up and balancing control problems can be addressed as a control system design problem using linearization with partial feedback and the linear quadratic regulator (LQR) or pole placement method for stabilization around the desired equilibrium position [3].

Based on the difference between the actual energy and the pendulum energy required to reach the desired position in the case of swing-up, Gulan et al. proposed that a model predictive control to guarantee stability [2]. In addition, Xin et al. improved previous energy-based control solutions by further analyzing the motion of the pendubot [4].

Lee and Coverstone-Carrol proposed three control algorithms for stabilizing underactuated robots at unstable equilibrium points. The well-known LQR technique was firstly described, and a stabilization control method using partial

Received 29 October 2019, **Revised** 14 November 2019, **Accepted** 15 November 2019
***Corresponding Author** Dong Sang Yoo (E-mail: dsyoo@hknu.ac.kr, Tel:+82-31-670-5322)
Department of Electrical, Electronic and Control Engineering, Hankyong National University, Anseong 17579, Korea.

Open Access https://doi.org/10.6109/jicce.2019.17.4.255 print ISSN: 2234-8255 online ISSN: 2234-8883

feedback linearization technique was then presented. However, both of these methods assumed that the dynamic model was accurate, whereas the mathematical modeling was inaccurate in practice; therefore, a robust controller using sliding mode theory was adopted [5].

A typical modeling method used in the simulation of dynamic mechanical systems is the modification of Lagrangian equations or Newton-Euler dynamics into state equations; these are then solved numerically according to given input and state conditions. Another approach is the use of 3D modeling and simulation tools, such as Mathworks' Simulink and Simscape.

Simscape enables designers to rapidly create models of physical systems within the Simulink environment, such as humanoid robots, mobile robots, and electrical machines. Simscape supports 3D modeling primitives, such as body elements, joints, frames and transforms, and forces and torques. It can also solve the equations of motion for the system in assembled 3D models without the derivation of dynamics, and it can import complete CAD data through several protocols that can be exchanged between 3D CAD applications. A 3D animation can also automatically visualize the behavior of mechanical systems [6].

In this study, a 3D model of the pendubot is constructed using Simscape and the usefulness of the 3D model is verified by comparing its behavior with dynamic motion induced by Lagrangian-formulation. Using this 3D model, the author then explores the use of an effective control method to ensure the posture balance of the pendubot.

II. MODELING OF TWO-LINK ROBOT

As shown in Fig. 1, the pendubot is an interesting example of an inverted pendulum and is an underactuated system with fewer control inputs than degrees of freedom. The pendubot is thus a planar two-link robot working against gravity, with an actuator at the first joint but no actuator at the second joint. The most common control tasks for the pendubot that have been previously studied are those of swing-up, posture balance, and trajectory tracking [2-3].

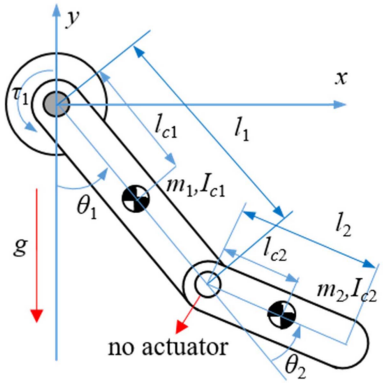


Fig. 1. Schematic of two-link pendubot.

A. 3D Pendubot Model

1) 3D Modeling

Simscape contains 3D modeling capabilities and can thus be used to construct a mechanical model; however, it lacks efficiency when constructing complex structures. This study used Solidworks, which is specialized for 3D CAD, to design each part of the pendubot. The pendubot was divided into three parts: the base frame, first link frame, and second link frame (as shown in Fig. 2). These parts were then imported into Simscape through the STEP format, which is the standard protocol used in data exchange between CAD applications. All parts were designed based on the z-axis of the Cartesian coordinate system, and the z-axis became the rotation axis during assembly of each part.

By designing complex parts in Solidworks and importing them into Simscape, it is possible to automatically calculate the center of mass and the corresponding moment of inertia for each part, which is advantageous when establishing the equation of motion for the pendubot. It is also useful when designing controllers that utilize dynamics in whole or in part; for example, when employing the computed torque method.

2) Assembly of 3D Models

Each part was assembled using Revolute Joint blocks and Rigid Transform blocks, as shown in Fig. 3. In the joint block, input ports can be defined to receive external force/torque inputs and output ports for the sensing position, velocity, and acceleration; these ports can then be used in the design of a controller.

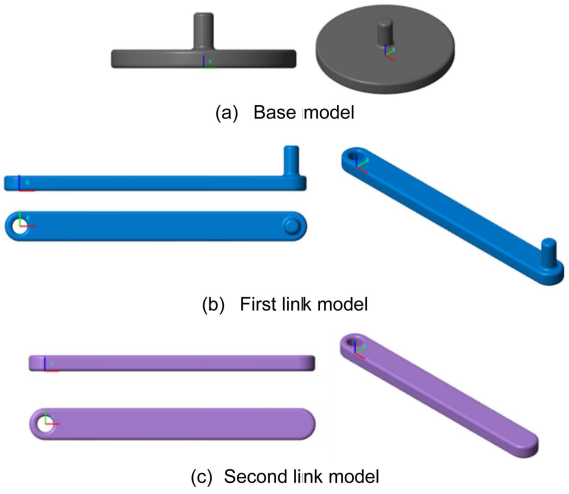


Fig. 2. Individual model of each pendubot part.

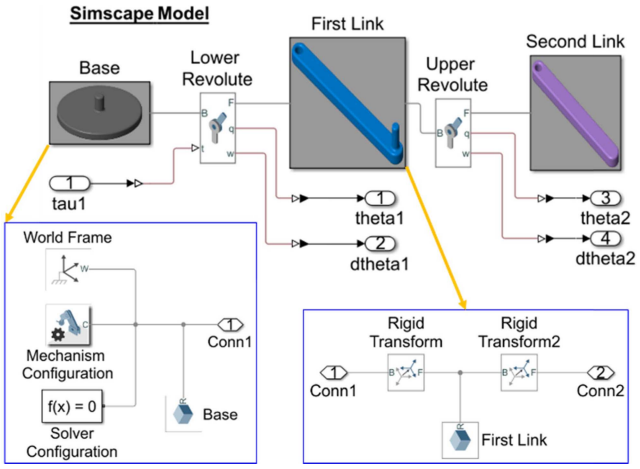


Fig. 3. Assembly of two-link pendubot.

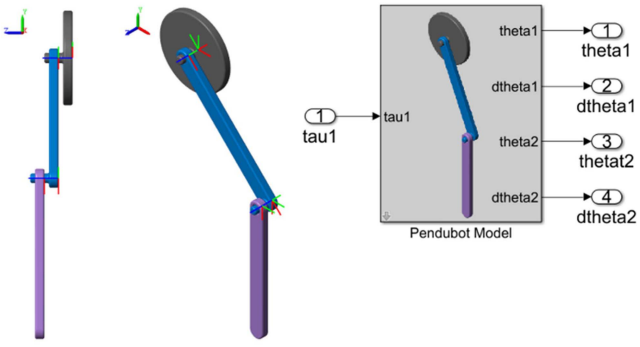


Fig. 4. 3D model of two-link pendubot.

The Revolute Joint block uses the common z axis of the attached frames as the joint rotation axis. For the pendubot to be influenced by gravity according to the coordinate system and gravity direction shown in Fig. 1, the gravity vector was set to [0 -9.81 0] using the Mechanism Configuration block. The y-axis was subject to gravity, and the rotation through the z-axis could thus be observed.

The final assembled 3D model of the pendubot is shown in Fig. 4, and this 3D model simulation enabled physics-based animation.

B. Mathematical Modeling

To verify the 3D pendubot model constructed in the previous section, we derived a mathematical model using the following Lagrangian formulation based on the geometric parameters shown in Fig. 1 [1-2],

$$\frac{d}{dt}\frac{\partial L(\theta,\dot{\theta})}{\partial \dot{\theta}_i}-\frac{\partial L(\theta,\dot{\theta})}{\partial \theta_i}+\frac{\partial R(\dot{\theta})}{\partial \dot{\theta}_i}=\tau,\quad i=1\&2, \tag{1}$$

where the Lagrangian, L, is defined as the difference between

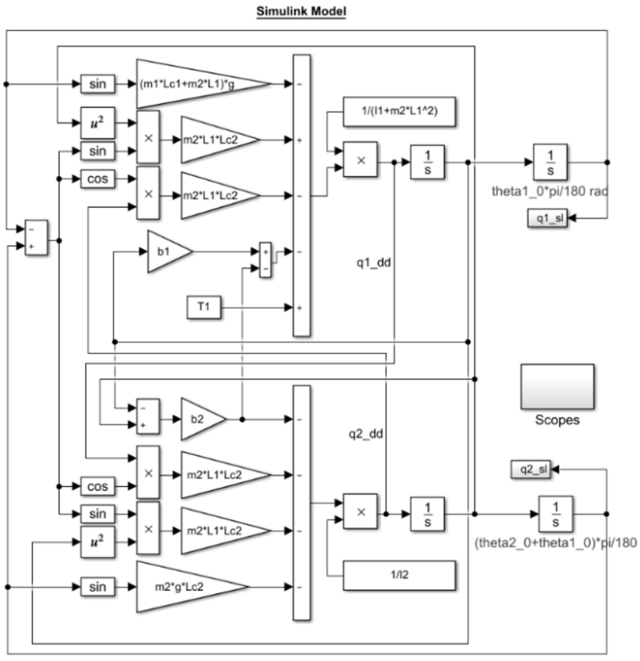


Fig. 5. Mathematical model created using Simulink.two-link pendubot.

the kinetic energy, T, and the potential energy, G, of the system, and the Rayleigh dissipation function, R, is denoted for friction. $\theta = [\theta_1\ \theta_2]^T$ is the vector of the angles of all joints in generalized coordinates, and $\tau = [\tau_1\ \ \tau_2]^T$ denotes the external control force vector.

From (1), the equation of motion was obtained as follows,

$$\begin{aligned}&(I_1+I_2+m_2l_1^2+2m_2l_1l_{c2}c_2)\ddot{\theta}_1+(I_2+m_2l_1l_{c2}c_2)\ddot{\theta}_2\\&\quad-2m_2l_1l_{c2}s_2\dot{\theta}_1\dot{\theta}_2-m_2l_1l_{c2}s_2\dot{\theta}_2^2+(m_1l_{c1}+m_2l_1)gs_1\\&\quad+m_2gl_2s_{1+2}+b_1\dot{\theta}_1=\tau_1\\&(I_2+m_2l_1l_{c2}c_2)\ddot{\theta}_1+I_2\ddot{\theta}_2+m_2l_1l_{c2}s_2\dot{\theta}_1^2\\&\quad+m_2gl_2s_{1+2}+b_2\dot{\theta}_2=\tau_2,\end{aligned} \tag{2}$$

where b_i is the coefficient of friction, $s_i = \sin\theta_i$, $c_i = \cos\theta_i$, and $s_{i+j} = \sin(\theta_i + \theta_j)$. In addition, I_i is the moment of inertia at the joint, which was obtained from the moment of inertia at the centroid using the parallel axis theorem as follows.

$$I_1 = I_{c1}+m_1l_{c1}^2,\ \ I_2 = I_{c2}+m_2l_{c2}^2. \tag{3}$$

I_{ci} was calculated automatically according to the shape in Solidworks or Simscape.

To enable simple construction of the Simulink model, (2) was rewritten with $[q_1\ \ q_2]^T = [\theta_1\ \ \theta_1+\theta_2]^T$, as follows,

$$\begin{aligned}&(I_1+m_2l_1^2)\ddot{q}_1+m_2l_1l_{c2}c_2\ddot{q}_2-m_2l_1l_{c2}s_2\dot{q}_2^2\\&\quad+(m_1l_{c1}+m_2l_1)gs_1+b_1\dot{q}_1-b_2(\dot{q}_2-\dot{q}_1)=\tau_1-\tau_2\\&(m_2l_1l_{c2}c_2)\ddot{q}_1+I_2\ddot{q}_2+m_2l_1l_{c2}s_2\dot{q}_1^2\\&\quad+m_2gl_2s_{1+2}+b_2(\dot{q}_2-\dot{q}_1)=\tau_2.\end{aligned} \tag{4}$$

The pendubot is a robot with no actuator in the second joint; therefore, the applied torque, τ_2, was set as 0.

The mathematical model designed in (4) was then constructed using Simulink, as shown in Fig. 5.

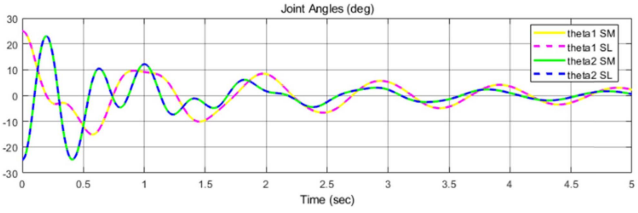
C. Comparison between 3D Model and Mathematical Model

To verify the conformity between the 3D model and the mathematical model of the pendubot shown in Figs. 4 and 5, a simulation of free pendulum movement with no control input was conducted, and the mechanical parameters of the pendubot used in the simulation are shown in Table 1.

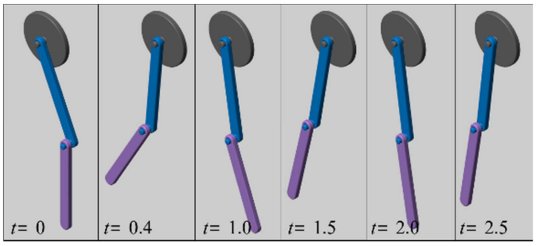
The results of the simulation with the initial position set to $[\theta_1\ \theta_2] = [25°\ -25°]$ are shown in Fig. 6, and it is evident that the simulation results of the two models agree with each other. It was thus expected that the same result would be obtained using only a 3D model of Simscape when simulating the balancing control of the pendubot after configuring the controller.

Table 1. Mechanical parameters of pendubot model

	Parameters	Value
Link 1	Mass, m_1	0.78304 kg
	Length, l_1	0.18 m
	Distance from centroid to joint 1, l_{c1}	0.09556 m
	Moment of inertia at centroid, I_{c1}	0.002569 kg m^2
	Moment of inertia at joint 1, I_1	0.0097197 kg m^2
	Friction coefficient, b_1	0.05 kg m^2/s
Link 2	Mass, m_2	0.67290 kg
	Length, l_2	0.14 m
	Distance from centroid to joint 2, l_{c2}	0.08201 m
	Moment of inertia at centroid, I_{c2}	0.001684 kg m^2
	Moment of inertia at joint 2, I_2	0.006209 kg m^2
	Friction coefficient, b_2	0.001 kg m^2/s



(a) Time histories



(b) Animation

Fig. 6. Simulation results of 3D model and mathematical model of pendulum movement.

III. BALANCING CONTROL

In this study, a sliding mode controller for balancing control at equilibrium positions was also designed, as shown in Fig. 7.

Although it was possible to obtain a linearization model around the operating point by applying the linearization technique using Taylor expansion [7], it was easier to obtain an linearization model (such as in (5)) for the upright equilibrium position, $[\theta_1\ \theta_2] = [180°\ 0°]$ using the linearization function provided by Matlab (as shown in Fig. 8),

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 67.57 & -3.20 & -55.34 & 0.17 \\ 0 & 0 & 0 & 1 \\ -88.48 & 8.32 & 231.21 & -0.59 \end{bmatrix} \quad (5)$$
$$B = \begin{bmatrix} 0 \\ 63.97 \\ 0 \\ -166.31 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where $x(t)$ and $u(t)$ represent the perturbation state vector and perturbation input, respectively, which indicate the degree of deviation from the upright equilibrium position.

In addition, for the sliding mode controller design, the sliding surface function was defined as follows:

$$\{\sigma \in R \mid \sigma(t) = Sx(t)\}. \quad (6)$$

Furthermore, to obtain reduced-order sliding mode dynamics, the following orthogonal transformation matrix was introduced as follows:

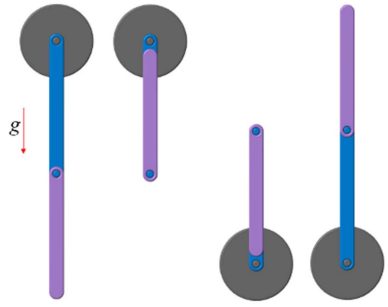


Fig. 7. Equilibrium positions of pendubot.

```
Pendubot_linearization.m
function Pendubot_linearization
mdl = 'Pendubot';
[~,X,~] = sim(mdl,0); % Get full state at operating point
[A,B,C,D] = linmod(mdl,X,0);% Linearizing about at operating position
Pendubot_param.A = A(1:4,1:4); Pendubot_param.B = B(1:4);
Pendubot_param.C = C(:,1:4); Pendubot_param.D = D;
assignin('base','Pendubot_param',Pendubot_param);
end
```

Fig. 8. Linearization of pendubot using Matlab.

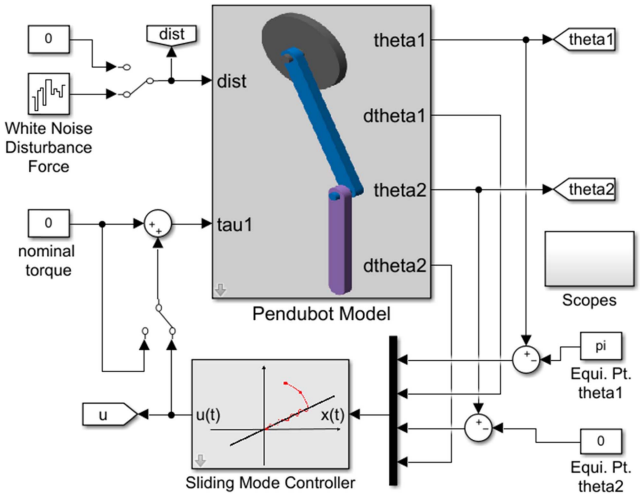


Fig. 9. Overall control system of pendubot.

$$z(t) = [z_1(t),\ z_2(t)]^T = Wx(t)$$
$$WAW^T = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},\ WB = \begin{bmatrix} 0 \\ B_2 \end{bmatrix}. \tag{7}$$

The sliding matrix, S, was determined by the matrix, M, and designed to minimize the cost function, $J=\frac{1}{2}\int x^T Q x dt$ [8],

$$S = \begin{bmatrix} M & 1 \end{bmatrix} W, \tag{8}$$

where

$$M = (A_{12}^T P + Q_{21})Q_{22}^{-1},\ WQW^T = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}, \tag{9}$$

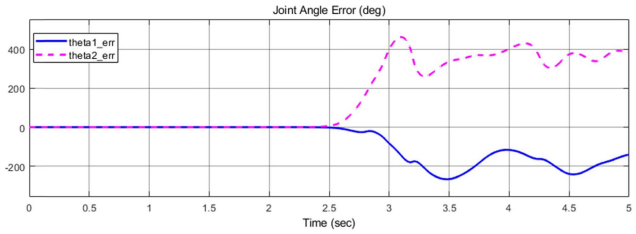
and the positive matrix, P, is the unique solution of the following Riccati equation:

$$P\hat{A} + \hat{A}P - PA_{12}A_{12}^T PQ_{22}^{-1} + \hat{Q} = 0$$
$$\hat{A} = A_{11} - A_{12}Q_{21}Q_{22}^{-1},\ \hat{Q} = Q_{11} - Q_{12}Q_{21}Q_{22}^{-1}. \tag{10}$$

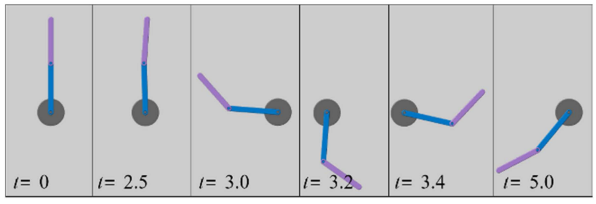
Furthermore, to eliminate chattering, the sliding controller was designed as follows:

$$u(t) = -(SB)^{-1}SAx(t) - \rho\frac{\sigma(t)}{|\sigma(t)| + \delta}. \tag{11}$$

To satisfy the stability of Lyapunov, ρ must be determined to satisfy $\rho \cdot (SB) > 0$.

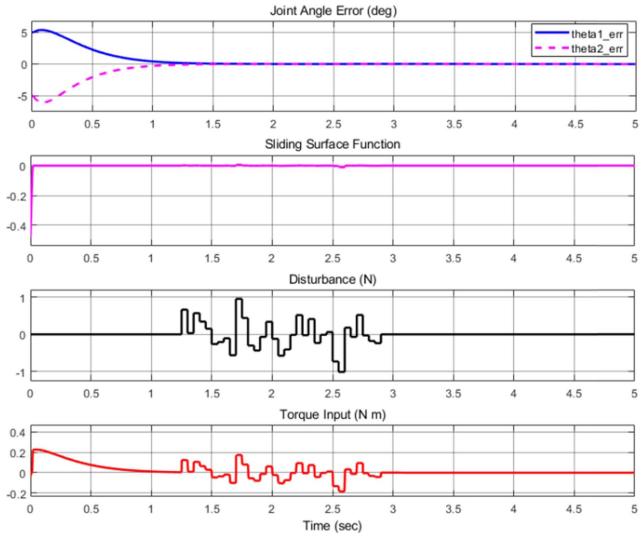


(a) Time histories

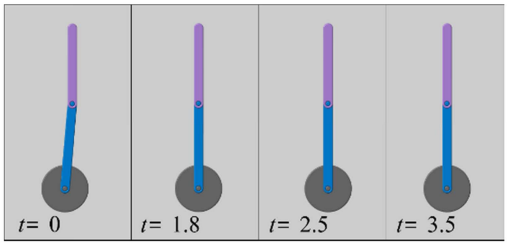


(b) Animation

Fig. 10. Simulation results of balancing without control.



(a) Time histories



(b) Animation

Fig. 11. Simulation results of balancing with control.

In this study, the sliding plane was obtained as S = [27.19 4.80 25.77 2.92] by setting to $Q = diag$[100 100 100 100] for the sliding mode controller design in an upright equilibrium position. The overall control system implemented in the inte-

grated environment in Simulink for balancing control of the pendubot is shown in Fig 9 (as *SB* is negative, ρ was also set to be negative).

The simulation results with no control or disturbance applied are shown in Fig. 10, where it is evident that the pendubot was initially balanced; however, as the upright equilibrium position was unstable, it eventually moved to a stable equilibrium position that pointed downward.

The behavior of the pendubot with the sliding mode controller in operation is shown in Fig. 11. The initial position was set to $[\theta_1\ \theta_2] = [175°, 5°]$, and a white noise disturbance force was applied from 1.3 sec to 2.9 sec to verify the robustness of the designed sliding mode controller. As shown in the figure, the designed controller enabled the pendubot's trajectory to quickly reach the sliding plane and to move to the equilibrium position. The upright balance was also well-maintained, even though disturbance was applied.

IV. CONCLUSIONS

The pendubot is one of the representative systems inherent in underactuated robots. In this study, a pendubot was created using Simscape and Solidworks, and the usefulness of the 3D model was verified using a comparison with a dynamic equation derived using the Lagrangian formulation. A linearization model was extracted to investigate balancing control around the upright position, and a sliding mode controller based on LQR was designed and implemented through Matlab and Simulink. Using the integrated simulation environment in Simulink, the designed sliding mode controller was shown to effectively maintain the upright balance of the pendubot, even in the presence of disturbance.

ACKNOWLEDGEMENTS

This work was supported by a research grant from Hankyong National University in the year of 2018.

REFERENCES

[1] R. Tedrake, “Underactuated robotics: Algorithms for walking, running, swimming, flying, and manipulation” [Online], Available: http://underactuated.mit.edu/underactuated.html.

[2] M. Gulan, M. Salaj, and B. Rohal’sIlkiv, “Achieving an equilibrium position of pendubot via swing-up and stabilizing model predictive control,” *Journal of Electrical Engineering*, vol. 65, no. 6, pp. 356-363, 2014. DOI: 10.2478/jee-2014-0058.

[3] D. J. Block, “Mechanical design and control of the pendubot,” Master’s thesis, University of Illinois, Champaign: IL, 1996.

[4] X. Xin, S. Tanaka, J. She, and T. Yamasaki, “New analytical results of energy-based swing-up control for the Pendubot,” *International Journal of Non-Linear Mechanics*, vol. 52, pp. 110-119, 2013. DOI: 10.1016/j.ijnonlinmec.2013.02.003.

[5] K. Lee and V. Coverstone-Carrol, “Control algorithms for stabilizing underactuated robots,” *Journal of Robotic Systems*, vol. 15, no. 12, pp. 681-697, 1998. DOI: 10.1002/(SICI)1097-4563(199812)15:12<681::AID-ROB2>3.0.CO;2-P.

[6] Mathworks, Getting Started with Simscape Multibody [Internet], Available: https://www.mathworks.com/help/physmod/sm/getting-started-with-simmechanics.html.

[7] M. W. Spong, “Partial feedback linearization of underactuated mechanical systems,” in *Proceeding of IEEE/RSJ International Conference on Intelligent Robots and Systems*, September 1994. DOI: 10.1109/IROS.1994.407375.

[8] C. Edwards and S. K. Spurgeon, *Sliding mode control Theory and Applications*, New York, NW:CRC Press, 1998.

Dong Sang Yoo

Dong Sang Yoo was born in Gwangju, Korea, in 1962. He received his B.S. degree in Electrical Engineering from Seoul National University, Seoul, Korea in 1985, and Masters and Ph.D. degrees in Electrical and Electronic Engineering from KAIST, Taejon, Korea in 1987, and 1992, respectively. From 1992 to 2000, he was a senior research engineer with LG Electronics Company, where he was engaged in research and development of semiconductor equipment and factory automation. He has been a Professor at Hankyong National University since September 2000. His current research interests include the areas of robust control, automation, robotics, and underactuated mechanical systems.