Coding Structures for Seated Row Simulation of an Active Controlled Vibration Isolation and Stabilization System for Astronaut's Exercise Platform

Ziraguen O. Williams, Shield B. Lin, Fouad N. Matari, Leslie J. Quiocho

***Abstract*—**Simulation for seated aerobic row exercise was a continued task to assist NASA in analyzing a one-dimensional vibration isolation and stabilization system for astronaut's exercise platform. Feedback delay and signal noise were added to the simulation model. Simulation runs for this study were conducted in two software simulation tools, Trick and MBDyn, software simulation environments developed at the NASA Johnson Space Center. The exciter force in the simulation was calculated from motion capture of an exerciser during a seated aerobic row exercise. The simulation runs include passive control, active control using a Proportional, Integral, Derivative (PID) controller, and active control using a Piecewise Linear Integral Derivative (PWLID) controller. Output parameters include displacements of the exercise platform, the exerciser, and the counterweight; transmitted force to the wall of spacecraft; and actuator force to the platform. The simulation results showed excellent force reduction in the active controlled system compared to the passive controlled system, which resulted in less force reduction.

***Keywords*—**Simulation, counterweight, exercise, vibration.

I. Introduction

The NASA Johnson Space Center (JSC) in Houston, Texas, USA, conducts space flight training for astronauts. The center sponsors many astronaut health related studies, such as muscle mass loss in space. Due to lack of gravity, crew members have experienced significant muscle mass loss in long spaceflights [1]. Astronauts must spend a significant amount of time in exercise in order to slow down the muscle mass loss. As might be expected, astronaut exercise in a spacecraft generates forces and moments which can be transmitted to the spacecraft. These forces and moments are undesirable because they may affect the experiments performed in the spacecraft and the operation of the spacecraft itself. A Vibration Isolation and Stabilization (VIS) system has been designed to minimize the transmitted forces to counter these effects [2-5].

The two software simulation environments used for this simulation work are a stand-alone Trick simulation environment using a lumped-sum model and a Trick-MBDyn simulation environment using a multibody system model. Trick Simulation Toolkit is a simulation program written in C, C++, and Java [6]. MBDyn is a multibody dynamics software engine [7-9]. It calculates and integrates the dynamics states of kinematic, rigid, or flexible articulated multibody systems. MBDyn interfaces are compatible with Trick simulation environment.

A single degree of freedom VIS system was developed in MATLAB/Simulink by Lin et al. [10]. This study used the single degree of freedom VIS system as the base, modified by adding feedback delay and signal noise to the model to simulate with more realistic conditions. The parameters used in this study were full-scale system parameters. The excited force used in this study was calculated from motion capture of an exerciser during a seated row exercise. A rowing machine is one of the common devices that astronauts use for exercises in space travel as shown in a photo in Fig. 1 [11].

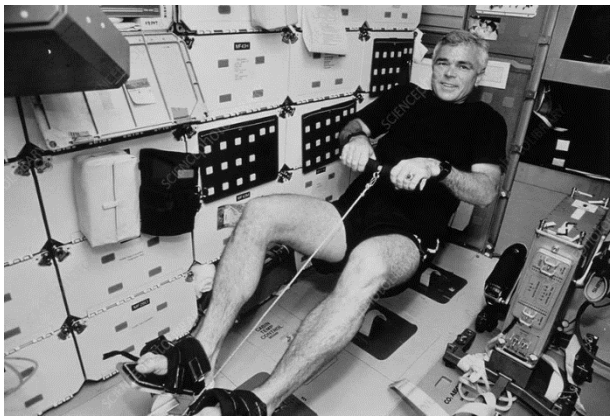

Fig. 1 Astronaut Using Rowing Machine for Exercise [11]

II. Simulation Code

We have previously described the VIS system in terms of schematic diagram, free body diagram, and dynamic equations [12]. The focus of this paper is to show the coding structures and simulation results from seated row exercise. C++ was used as the primary programming language for coding in the Trick simulation environment while Python programming language was used for the input files. Table I shows the functions that were coded to run this simulation. There was a total of 10

Ziraguen O. Williams is with CACI International Inc., Houston, Texas, USA.

Shield B. Lin is with the Department of Mechanical Engineering at Prairie View A&M University, Prairie View, Texas, USA (e-mail: shlin@pvamu.edu).

Fouad N. Matari is with CACI International Inc., Houston, Texas, USA.

Leslie J. Quiocho is with NASA Johnson Space Center, Houston, Texas, USA.

functions that needed to be written. For the feedback delay function, it was found that the longest calculation cycle took less than 0.000022 seconds. The sensor's base measurement rate is 0.0003 seconds. These two combined are less than 0.0004 seconds. Since this is supposed to be worst-case the feedback delay was rounded up to 0.001 seconds.

TABLE I
TRICK FUNCTION TABLE

Name of Function	Description
default_data()	Sets default values for simulation
init()	Initializes simulation using values from input file
PID_control()	Computes output from controller
signal_noise()	Compute signal noise
feedback_delay()	Delays signal feedback
calc_motor()	Computes some linear actuator values
calc_omega()	Computes some linear actuator values
calc_force()	Computes forces from controls
derivative()	Does necessary calculations before integration
integ()	Integrates necessary values

The stand-alone Trick simulation uses a Python-based input file to set variables and environment parameters. Some variables might not be needed depending on the type of input being used. The variables needed as input are shown in Table II.

TABLE II
TRICK INPUTS

Property	Variables
Lead screw length	L_IN
Inductance	La_IN
Resistance	Ra_IN
Motor Torque Constant	Km_IN
Motor Inertia	Jl_IN
Back EMF Constant	Kb_IN
Saturation Upper Limit, V	sat_high_IN
Saturation Lower Limit, V	sat_low_IN
Upper Dead Zone, Amps	dz_high_IN
Lower Dead Zone, Amps	dz_low_IN
Spring Coefficient	K_IN
Damping Coefficient	C_IN
P1	P1_IN
P2	P2_IN
Integral Gain	I_IN
Derivative Gain	D_IN
Time filter coefficient	N_IN
Controller frequency	dt_IN
E0	err0_in
Force Amplitude	f_amp_IN
Force Frequency	freq_IN
Time step	sim_step
Duration	sim_duration
Force file	execfile("Modified_data/forces_XX.py")
Noise sigma	sigma_IN
Total mass	Mt_IN

The MBDyn simulation is fundamentally still a Trick simulation but with added functionality. The code is very similar to the stand-alone Trick simulation with some slight changes made. The main difference between the two is that MBDyn provides multibody functionality.

MBDyn is used here to perform rigid multibody simulation. MBDyn uses bodies, joints, and nodes to describe the system its simulating. Bodies are used to describe each mass. Joints are either linear or rotational and connect bodies. Each body has an input node, a center of mass node, and can have multiple output nodes if there are additional bodies being linked. In order to use the multibody functionality of MBDyn, there were additional files created. One file sets up the topology and another file collects the forces in the simulation and places them at the appropriate nodes. Fig. 2 shows a diagram that represents the topology of the MBDyn-based simulation. Fig. 3 shows a schematic diagram of the VIS system as simulated.

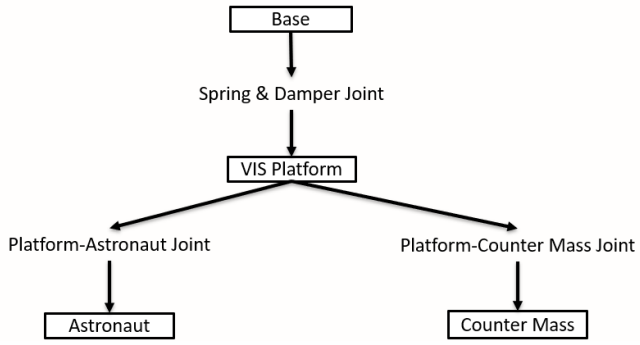

Fig. 2 MBDyn Topology Used in Simulation

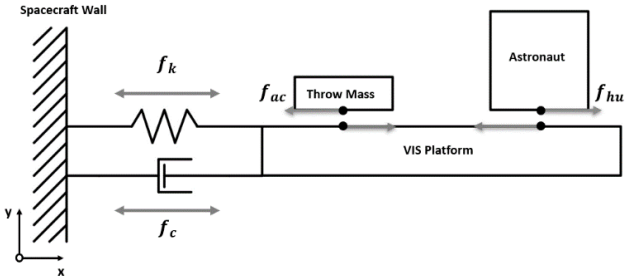

Fig. 3 VIS System Schematic Diagram

Table III shows a list of functions that were coded for the MBDyn simulation. It highlights a reduction in necessary hardcoded functions when compared to the stand-alone Trick simulation. This simulation only needed 8 hardcoded functions.

TABLE III
MBDYN FUNCTION TABLE

Name of Function	Description
initialize()	Initializes simulation using values from input file
control()	Computes output from controller
dynamics()	Does necessary calculations before integration
calc_motor()	Computes some linear actuator values
calc_omega()	Computes some linear actuator values
integ()	Integrates necessary values
signal_noise()	Compute signal noise
feedback_delay()	Delays signal feedback

The MBDyn simulation uses an input file similar to the Trick simulation with the addition of two files that set up the topology of the model and the collection of forces. Inside the input file

we set environment parameters, extra files that need to be executed, control values, system parameters, and duration of the simulation. Table IV lists the MBDyn inputs.

TABLE IV
MBDYN INPUTS

Property	Variable
Lead screw length	L_IN
Inductance	La_IN
Resistance	Ra_IN
Motor Torque Constant	Km_IN
Motor Inertia	Jl_IN
Back EMF Constant	Kb_IN
Saturation Upper Limit, V	sat_high_IN
Saturation Lower Limit, V	sat_low_IN
Upper Dead Zone, Amps	dz_high_IN
Lower Dead Zone, Amps	dz_low_IN
Spring Coefficient	spring_stiffness
Damping Coefficient	spring_damping
Small Proportional Gain	P1_IN
Large Proportional Gain	P2_IN
Integral Gain	I_IN
Derivative Gain	D_IN
Time filter coefficient	N_IN
Controller frequency	dt_IN
E0	err0_in
Force Amplitude	amplitude[0][0]
Force Frequency	frequency[0][0]
Time step	sim_step
Duration	sim_duration
Force file	execfile("Modified_data/forces_XX.py")
Noise sigma	sigma_IN
Topology file	execfile("Modified_data/control_X.d")
Force collection	execfile("Modified_data/springX.py")

III. SIMULATION DATA

In order to simulate the VIS and astronaut exercising, assumptions about masses and geometries had to be made. The assumption was made that there would be three masses: the VIS platform, the astronaut, and the throw mass or inertial mass. Other components would be included in one of those three masses. For example, the mass of the linear actuator was considered to be part of the VIS platform. The throw mass/ inertial mass and the astronaut were modeled as cubes. The VIS platform was modeled as a rectangular prism. In addition to these components there was a spacecraft wall modeled. However, since this wall is fixed within the simulations its properties have no effects on the results. For a real scale system, the VIS platform properties are listed in Table V, astronaut properties are listed in Table VI, and throw-mass properties are listed in Table VII.

TABLE V
VIS PLATFORM PROPERTIES

Property	Value
Mass	120 kg
Length (x)	2 m
Height (y)	0.1 m
Width (z)	0.25 m
Inertia (I_{xx}, I_{yy}, I_{zz})	[0.725, 40.625, 40.1] kg m^2

TABLE VI
ASTRONAUT PROPERTIES

Property	Value
Mass	75 kg
Length (x)	0.25 m
Height (y)	0.25 m
Width (z)	0.25 m
Inertia (I_{xx}, I_{yy}, I_{zz})	[0.78125, 0.78125, 0.78125] kg m^2

TABLE VII
THROW-MASS/INERTIAL-MASS PROPERTIES

Property	Value
Mass	200 kg
Length (x)	0.2 m
Height (y)	0.2 m
Width (z)	0.2 m
Inertia (I_{xx}, I_{yy}, I_{zz})	[1.333, 1.333, 1.333] kg m^2

The force calculated from seated row exercise was used as the excited force to the VIS system. It is labeled as RUN 3 in the simulation runs. The force data was provided through a NASA JSC data collection and was calculated based on motion capture data from seated aerobic row exercise [13]. The force data can be fed into the simulations from a comma-separated values file (.CSV). The force data from the motion capture are referred to as Forcing Functions or FFns. Since this simulation and study is a single degree of freedom, the axis with the highest magnitude of force was chosen as the input. It is assumed that the astronaut exercising would be placed on the platform in such a way that the data's axis of highest magnitude would line up with the x-axis of the simulation. The force data used in simulation runs in this paper are from a seated row exercise.

Fig. 4 shows the simplified process followed by NASA engineers to calculate the reaction forces and moments generated by astronauts exercising. Motion capture was used to collect position data and then inverse kinematics was applied. Using the results from inverse kinematics a math plug-in was used in OpenSim [14, 15] to generate the reaction forces and moments.

Fig. 4 Astronauts Exercising Force Calculation

IV. SIMULATION RESULTS AND DISCUSSIONS

Fig. 5 shows the input forces (N) generated by the astronaut performing a seated aerobic row exercise on the VIS platform. The figure shows the peak magnitude going above 400 N.

Fig. 6 shows two MBDyn simulation plots of the performance of the passive control system when simulating the seated aerobic row exercise, using Koviz, a NASA JSC data visualization tool [16]. The variables being plotted are the position of the VIS displacement in Part (a) of the plot and the astronaut displacement in Part (b) of the plot. The plots show decent control of the VIS platform and reasonable or expected. displacement of the astronaut.

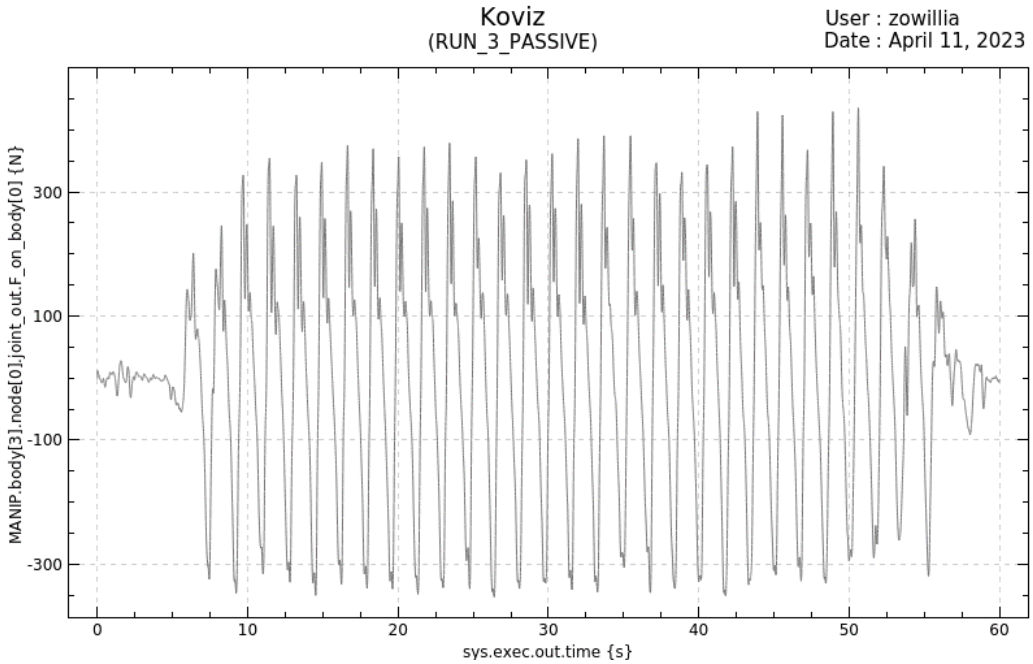

Fig. 5 RUN 3: Astronaut Input Force (N)

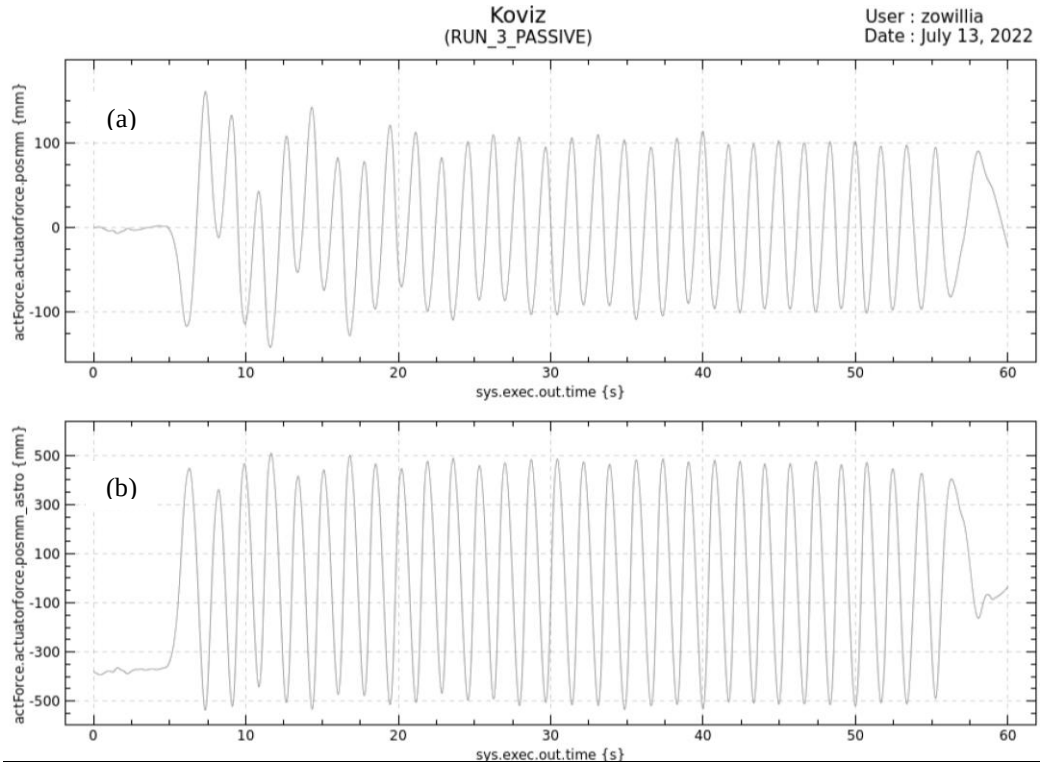

Fig. 6 MBDyn RUN 3: Passive – VIS and Astronaut Displacement (mm)

Fig. 7 shows a MBDyn simulation plot of the performance of the passive control system when simulating the seated row exercise. The variable being plotted is the force being transmitted to the wall. The plot shows about a 75% or more reduction in forces being transmitted. Input forces have a peak amplitude of about 240 N. This shows decent performance from the passive control system.

Fig. 8 shows a Trick and MBDyn performance co-plot of the PID active control system with a seated row exercise. The variable being plotted is the displacement of the VIS platform. The total displacement of the VIS platform is less than 30 mm. This shows extremely good performance considering the magnitude of the input forces. The plot shows good agreement between the two simulation models.

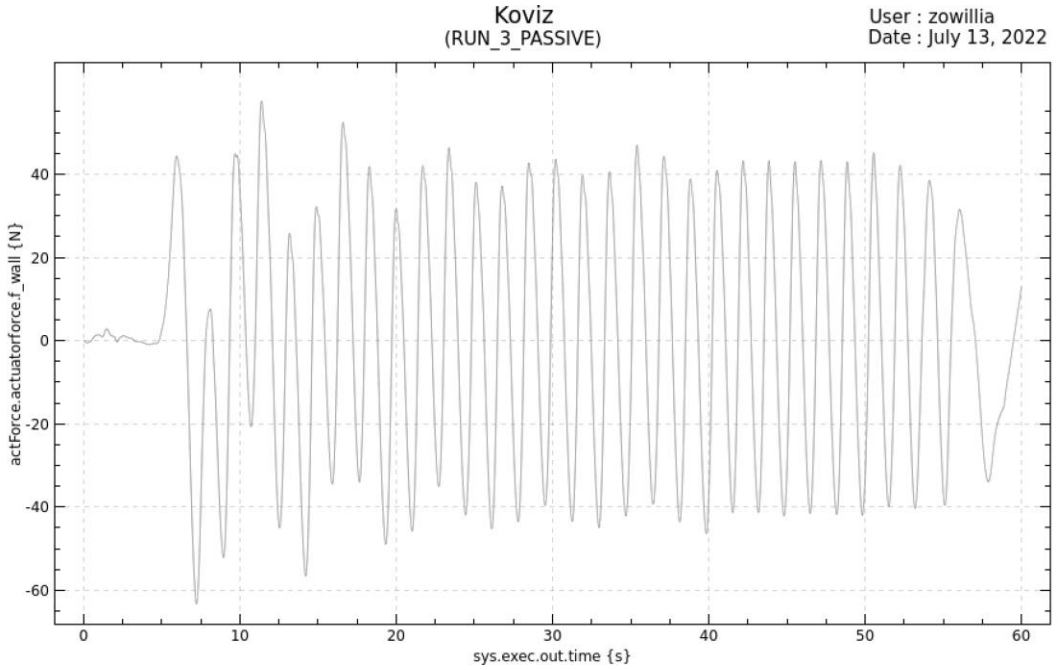

Fig. 7 MBDyn RUN 3: Passive – Transmitted Force to Wall (N)

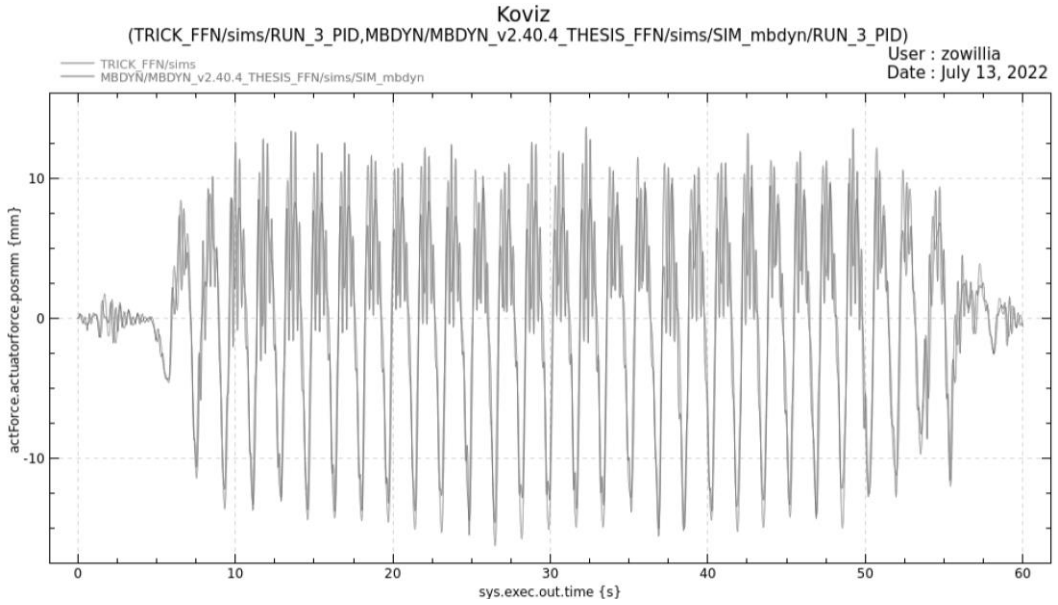

Fig. 8 Trick vs. MBDyn RUN 3: PID – VIS Displacement (mm)

Fig. 9 shows a Trick and MBDyn co-plot of the performance of the PID active control system with a seated aerobic row exercise. The variable being plotted is the force being transmitted to the spacecraft wall. The max peak-to-peak force is under 4 N. The plot shows around a 99% reduction in transmitted force. This shows extremely good performance. The plot also shows agreement between the two simulation models.

Fig. 10 shows a Trick and MBDyn co-plot of the performance of the PID active control system with a seated row exercise. The variable being plotted is the force being exerted by the linear actuator. This plot shows us the force output performance needed by the linear actuator in order to compensate for the astronaut's exercise. The plot also shows good agreement between the two simulation models.

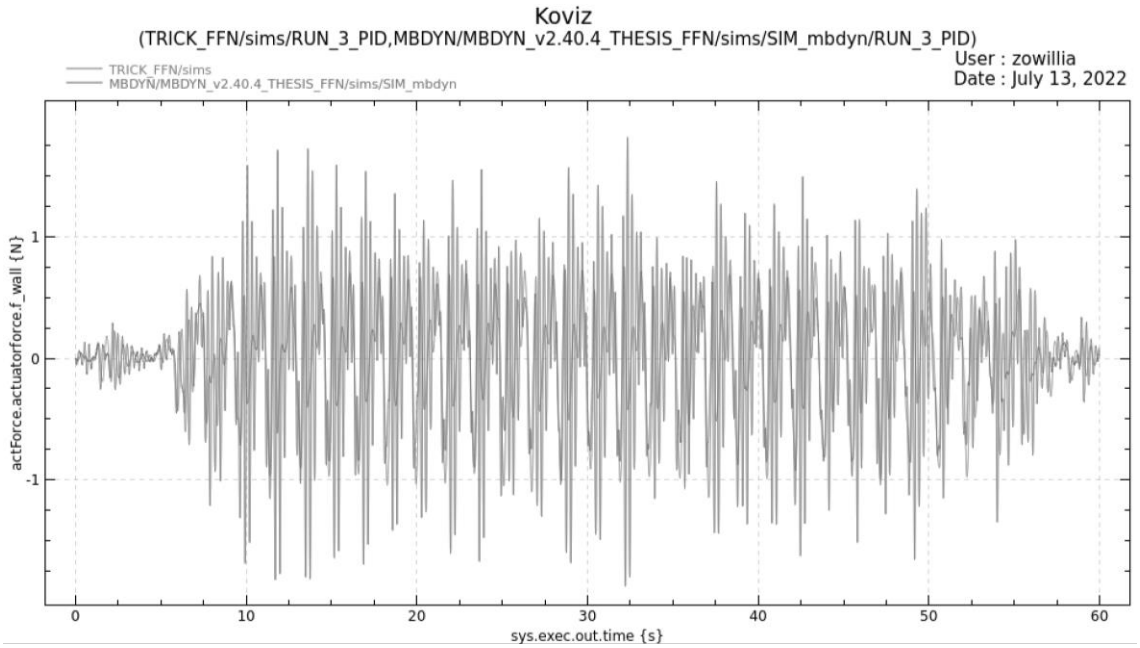

Fig. 9 Trick vs. MBDyn RUN 3: PID – Transmitted Force to Wall (N)

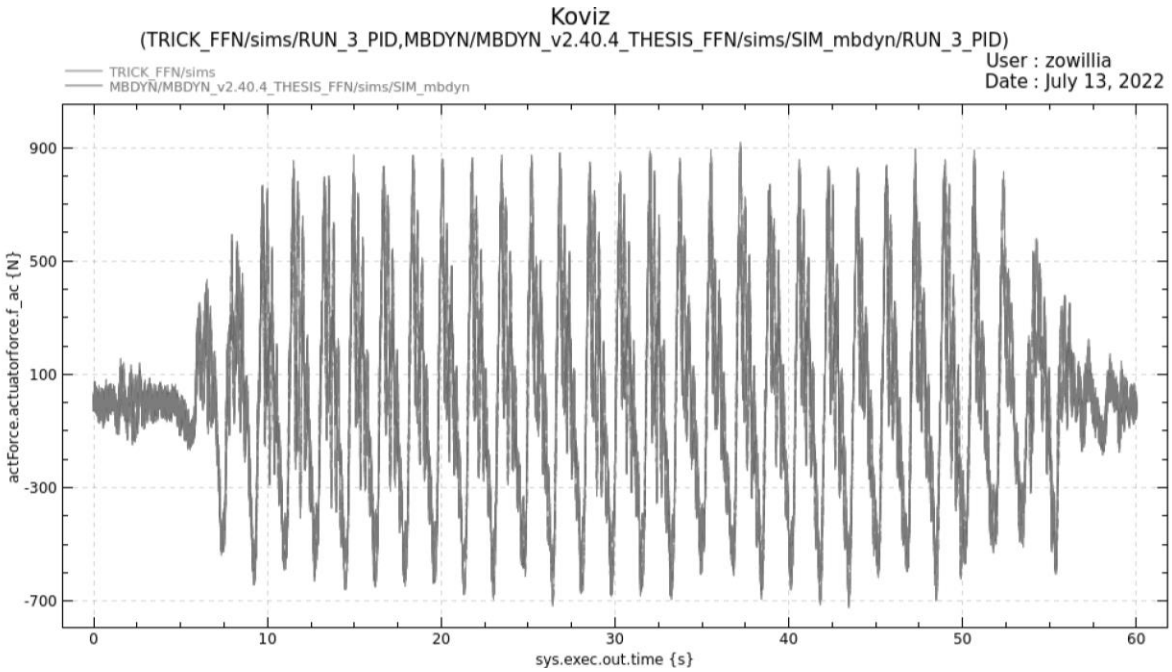

Fig. 10 Trick vs. MBDyn RUN 3: PID – Actuator Force (N)

Fig. 11 shows MBDyn plots of the astronaut and throw mass displacement for the seated row exercise and the PID controlled linear actuator. The plot shows reasonable displacement for both masses. This magnitude of displacement would be possible within the volume of a spacecraft like the International Space Station (ISS).

Fig. 12 shows a Trick and MBDyn co-plot of the performance of the PWLID active control system with a seated row exercise. The variable being plotted is the displacement of the VIS platform. Similar to the PID controller, the total displacement of the VIS platform is less than 33 mm. This shows extremely good performance considering the magnitude of the input forces. The plot also shows agreement between the two simulation models.

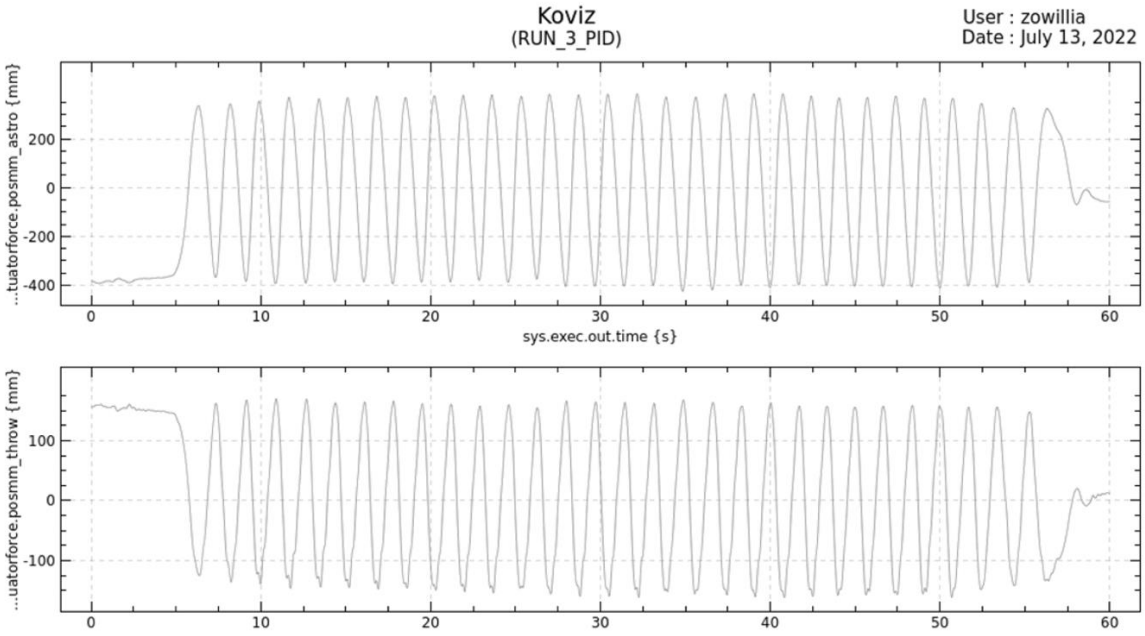

Fig. 11 MBDyn RUN 3: PID – Astronaut and Throw Mass Displacement (mm)

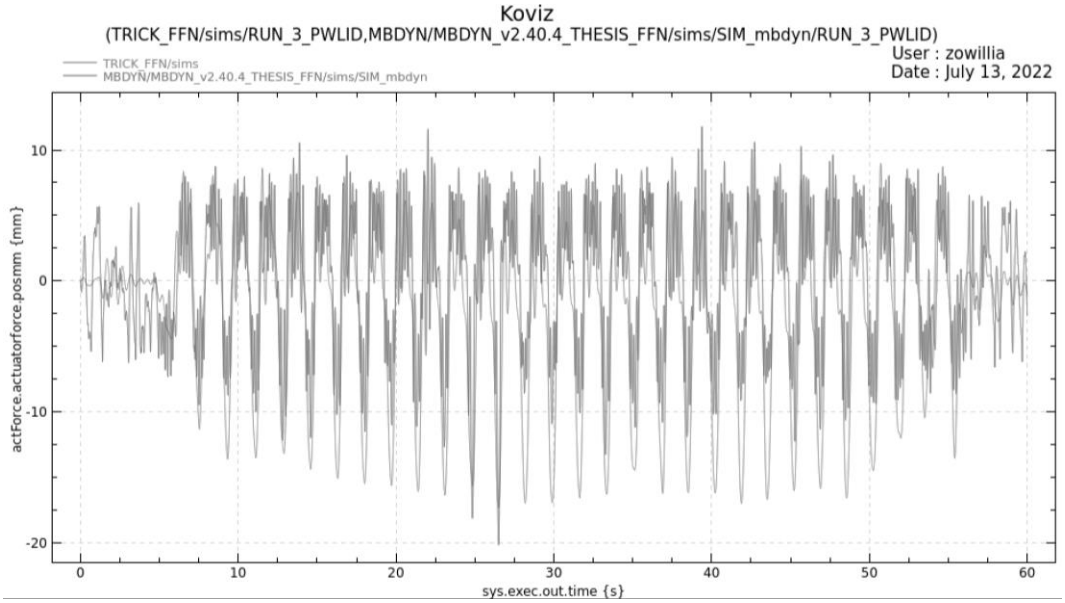

Fig. 12 Trick vs. MBDyn RUN 3: PWLID – VIS Displacement (mm)

Fig. 13 shows a Trick and MBDyn co-plot of the performance of the PID active control system with a seated row exercise. The variable being plotted is the force being transmitted to the spacecraft wall. The max peak-to-peak force is under 5.75 N. The plot shows around a 99% reduction in transmitted force. This shows good performance. The plot shows some agreement between the two simulation models in terms of the overall reduction of transmitted force.

Fig. 14 shows a Trick and MBDyn co-plot of the performance of the PWLID active control system with a seated row exercise. The variable being plotted is the force being exerted by the linear actuator. The forces being generated are larger than when controlled by the PID controller. The plot suggests some agreement between the models, but the Trick simulation line is not completely visible. In the Trick simulation, the actuator force ranges from about 1015 N to -775 N but follows the same pattern as the MBDyn simulation.

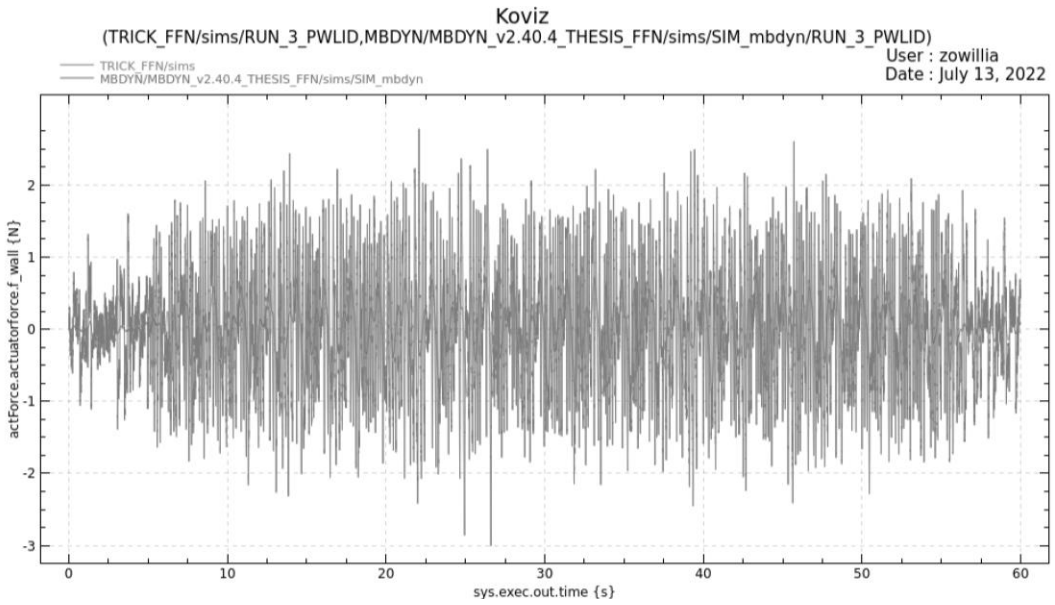

Fig. 13 Trick vs. MBDyn RUN 3: PWLID – Transmitted Force to Wall (N)

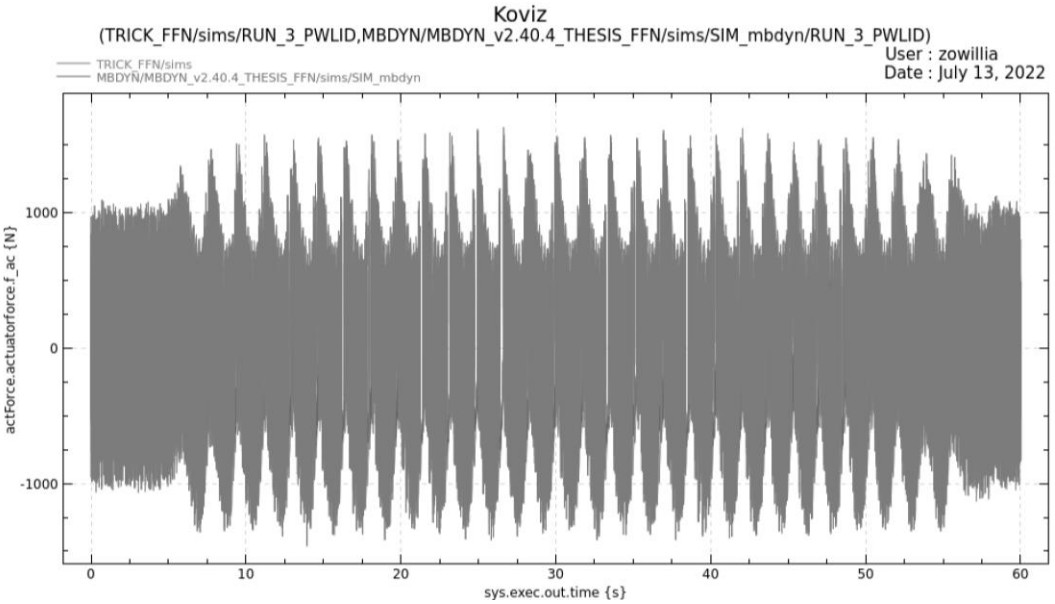

Fig. 14 Trick vs. MBDyn RUN 3: PWLID – Actuator Force (N)

Fig. 15 shows a MBDyn co-plot of the VIS displacement of all three control methods: passive, PID, and PWLID when simulating a seated row exercise. The plot shows the active control systems have better performance than the passive control system with a displacement reduction of at least 85%. The PID and PWLID controllers have similar performance.

Fig. 16 shows a MBDyn co-plot of the force being transmitted to the spacecraft wall for all three control methods: passive, PID, and PWLID when simulating a seated row exercise. The plot shows the active control systems have better performance in reducing the force being transmitted than the passive control system by about 95%. The PID and PWLID controllers have similar performance.

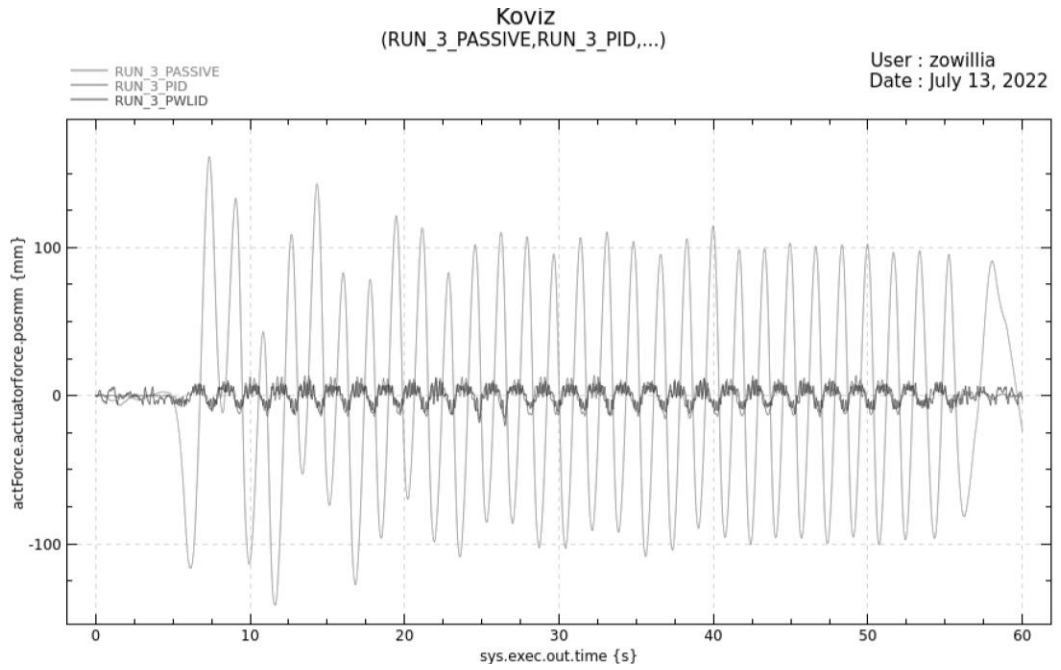

Fig. 15 MBDyn RUN 3: Controls Comparison – VIS Displacement (mm)

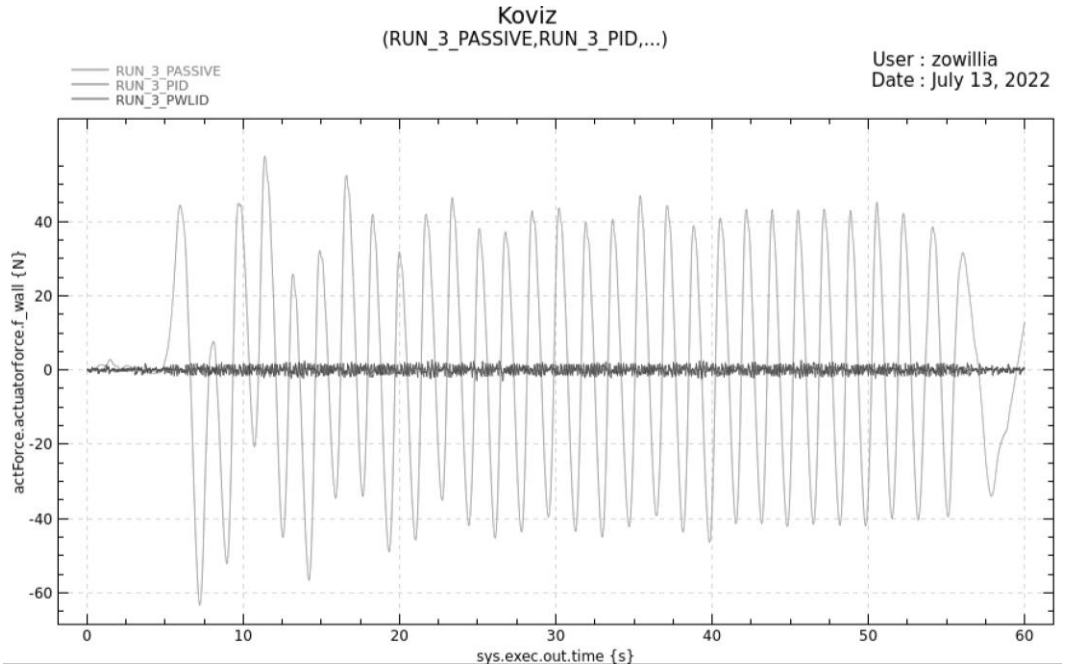

Fig. 16 MBDyn RUN 3: Controls Comparison – Transmitted Force to Wall (N)

V. Conclusions and Recommendations

Coding structures for simulating the seated aerobic row exercise of an astronaut on the VIS platform in Trick and MBDyn simulation environments are described in the paper. The calculated force from a seated row exercise was used as the exciter force to the simulation model of a one-dimensional VIS system for an astronaut's exercise platform. Simulation runs were performed for an actively controlled VIS system and a passively controlled VIS system. Even with the addition of feedback delay and signal noise to the simulation model, all simulation results indicated that the active controlled systems outperformed the passive control system.

The results from the Trick and MBDyn simulations demonstrated that there was an agreement between the two simulation environments. The results also show that the lumped-sum model and multibody model agreed in the general behavior of the system with some minor differences between the results.

Control algorithms used in this study include PID control and PWLID control. Both control algorithms obtained excellent simulation results which transmitted a minimal amount of force

to the spacecraft wall. The results suggest that active control systems can be very useful to limit vibrations during astronaut exercise.

One of the important future research tasks is to add degrees of freedom, namely, increase the dimension from the current one-dimensional VIS system to multiple dimensions including rotation. Other potential improvements to the model include adding friction equations to the system, considering power consumption of the actuator, heat dissipation of all moving parts, and converting the human exerciser's model from rigid body motion to flexible body motion.

ACKNOWLEDGMENTS

The authors would like to thank Robert Zehentner and Daniel Erdberg of CACI, Mike Red at NASA JSC, and many other colleagues working at JSC for the research opportunities, sponsorship, and support.

REFERENCES

[1] J.R. Bagley, K.A. Murach, and S.W. Trappe, "Microgravity-Induced Fiber Type Shift in Human Skeletal Muscle." Gravitational and Space Biology, Volume 26(1), pp. 34-40, May 2012.

[2] Niebuhr, J.H. and Hagen, R.A., "Development of the vibration isolation system for the advanced resistive exercise device," in 41st Aerospace Mechanisms Symposium, January 2011.

[3] GRODSINSKY, C., and BROWN, G., "Nonintrusive inertial vibration isolation technology for microgravity space experiments," in 28th Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, 1-8.

[4] Yang, B. J., Calise, A., Craig, J., & Whorton, M., "Adaptive control for a microgravity vibration isolation system," in AIAA guidance, navigation, and control conference and exhibit, American Institute of Aeronautics and Astronautics, pp. 1-19, August 2005.

[5] L. J. Quiocho, K. Lostroscio, S. Joshi, E. Kovel, K. Vetter, D. Frenkel, C. Bell, L. Nilsson, A. Reeves, "Modeling and Simulation for Exercise Vibration Isolation and Stabilization System Design," in IEEE Aerospace Conference, pp.1-19, 2023.

[6] J.M. Penn and A.S. Lin, "The Trick Simulation Toolkit: A NASA/Open source Framework for Running Time Based Physics Models," in AIAA Modeling and Simulation Technologies Conference, American Institute of Aeronautics and Astronautics, pp.1-13, 2016.

[7] L. J. Quiocho, A. Huynh and E. Z. Crues, "Application of Multibody Dynamics to On-Orbit Manipulator Simulations," in 5th International Conference on Multibody Systems, Nonlinear Dynamics, and Control (MSNDC), ASME DETC 2005-85545, Long Beach, CA, 2005.

[8] J. MacLean, T. Brain, L. Quiocho, A. Huynh and T. Ghosh, "Linked-List-Based Multibody Dynamics (MBDyn) Engine," MSC-24925-1, NASA Tech Briefs, September 2012.

[9] A. Huynh, T. Brain, J. R. MacLean and L. J. Quiocho, "Evolution of Flexible Multibody Dynamics for Simulation Applications Supporting Human Spaceflight," in 12th International Conference on Multibody Systems, Nonlinear Dynamics, and Control (MSNDC), ASME DETC 2016-60108, Charlotte, NC, 2016.

[10] S.B. Lin and S. Abdali, "Simulation of Active Controlled Vibration Isolation System for Astronaut's Exercise Platform," International Journal of Mechanical and Mechatronics Engineering, Vol. 15, No.2, pp.107-112, 2021.

[11] Photo: Science Photo Library, London W9 3RB. United Kingdom, https://www.sciencephoto.com/media/336689/view

[12] Z.O. Williams, S.B. Lin, F.N. Matari, L.J. Quiocho, "Simulation for Squat Exercise of an Active Controlled Vibration Isolation and Stabilization System for Astronaut's Exercise Platform," International Conference on Dynamic Systems, Control and Robotics, Ottawa, Canada, July 12-13, 2022.

[13] NASA JSC, Motion Capture Based Forcing Functions from E4D Data Collection, 2019 Available: https://www.danishaerospace.com/en/news/new-danish-space-exercise-machine-completes-testing-at-nasa

[14] Delp, S.L., et al., "OpenSim: Open-source Software to Create and Analyze Dynamic Simulations of Movement", IEEE Transactions on Biomedical Engineering, 2007.

[15] Seth, A., Hicks J.L., Uchida, T.K., Habib, A., Dembia, C.L., Dunne, J.J., Ong, C.F., DeMers, M.S., Rajagopal, A., Millard, M., Hamner, S.R., Arnold, E.M., Yong, J.R., Lakshmikanth, S.K., Sherman, M.A., Delp, S.L. OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. Plos Computational Biology, 14(7), 2018.

[16] Nasa. "NASA/Koviz: Koviz Is a Trick Simulation Data Plotting, Visualization and Analysis Tool." GitHub, https://github.com/nasa/koviz.