Model-Based Controller Development for UAVs and Walking Robots Using OpenModelica

Sunil Shah, ModeliCon InfoTech LLP,
Pavan P, ModeliCon InfoTech LLP,
Sourabh TK, ModeliCon InfoTech LLP,
Anurag Giri, ModeliCon InfoTech LLP,
Peter Fritzson, Linköping University

Agenda

Applications of Modelica/OpenModelica for:

1. Software-In-The-Loop (SITL) and Hardware-In-The-Loop (HITL) framework for **UAV operation** and **control**

1. Digital Twin for **self-balancing robot**

1. Software-In-The-Loop (SITL) and Hardware-In-The-Loop (HITL) Framework for UAV Operation and Control

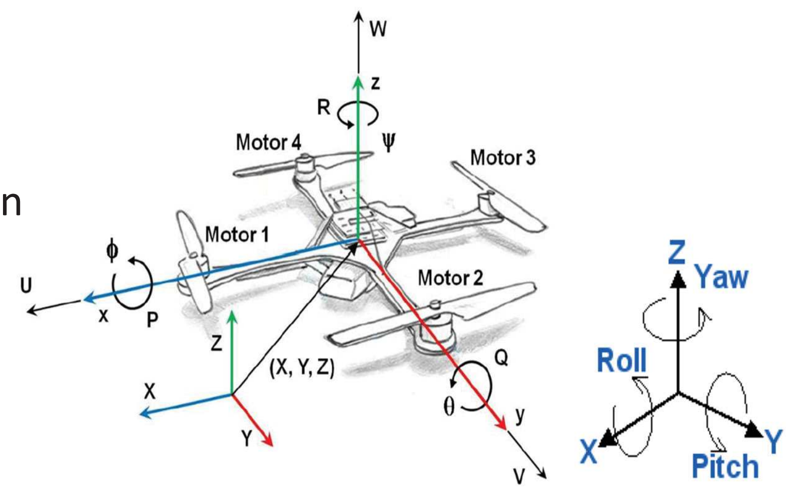
Objective

1. To allow for rapid deployment of UAVs in industrial environments by observing simulated flight control behaviour with varying payloads.
2. Facilitated by creating dynamic UAV models for Hardware-In-Loop simulations for different applications allowing:
   a. Flight controller evaluation
   b. Controller tuning
3. A Modelica Based Simulator will allow for:
   a. Enabling “Flight Controller - Ground Control Station - MAVLink Server - Modelica” communication
   b. Building custom UAV configurations (fixed wing/n-frame multicopters)
   c. Simulating faults, disturbance and noise in the controller and environment
   d. Include additional operational requirements (e.g. gimbal)
   e. Tune flight controllers
   f. Developing custom flight controllers

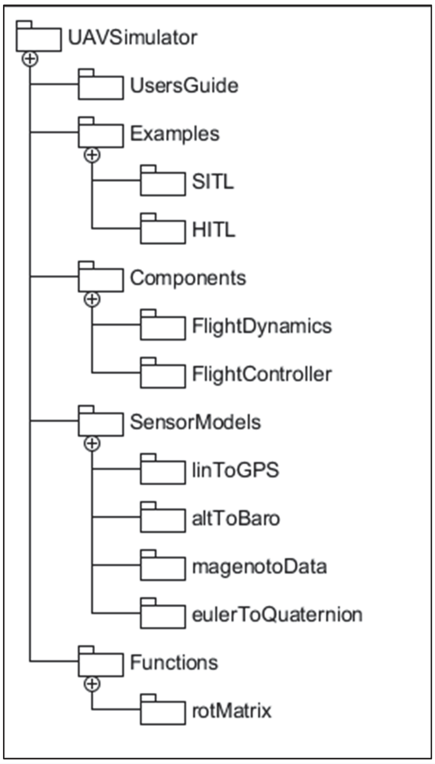
UAV Model - OpenModelica

Key components considered to develop UAV model:

1. Mass Moment of Inertia
2. Configuration
3. Thrust coefficient
4. Torque coefficient
5. Throttle - Motor relation
6. Gyroscopic forces
7. External forces
8. State equations
    a. Angular velocity
    b. Euler angles
    c. Position
    d. Velocity

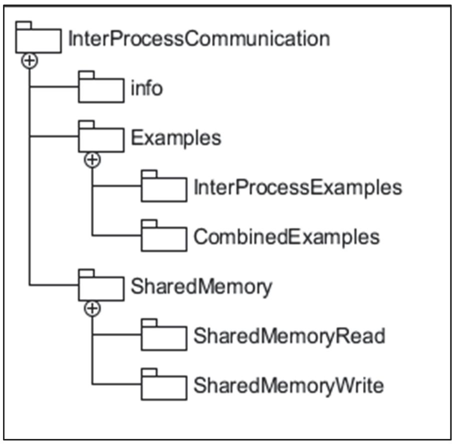


UAV Model - OpenModelica Libraries (Built in-house)



UAVSimulator : Modelica/OpenModelica Library Containing UAV **Flight Dynamics Models**, Sensor Models and Flight **Controller Models** built using **PID** blocks from Modelica Library

InterProcess-Communication Library: SharedMemory and Serial **communication** between OpenModelica and other software/hardware

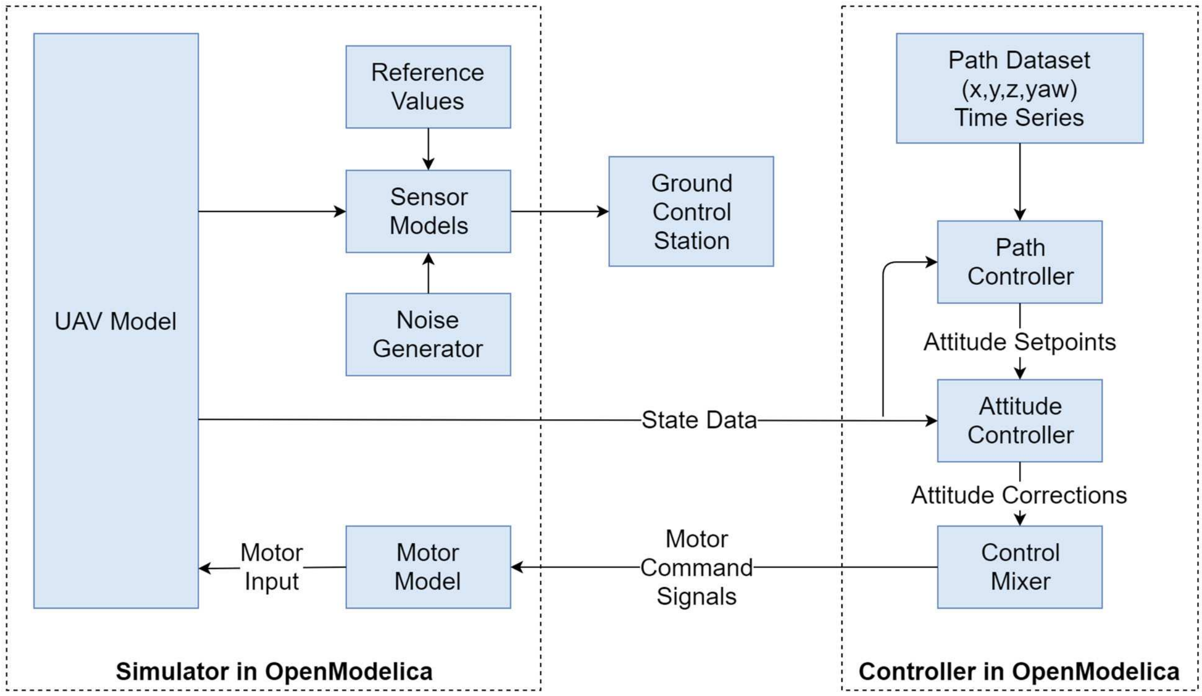


Sensor Model Development

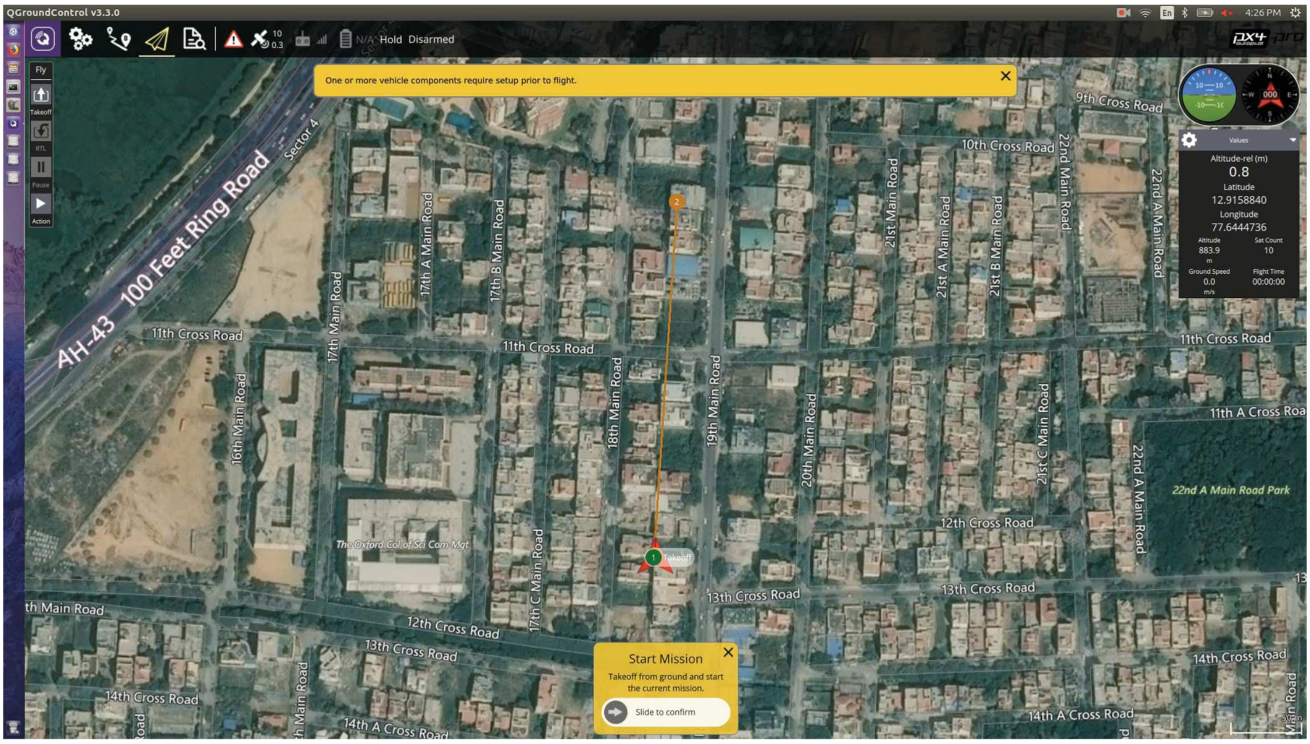
State variables calculated using mathematical models is converted to sensor data to mimic their behaviour. The **sensors** are:

1. **GPS** - Latitude and Longitude conversion from Cartesian coordinates
2. **Gyro** sensor - Angular rates in BodyFrame
3. **Accelerometer** - Linear acceleration
4. **Magnetometer** - Magnetic field vector calculation based on the orientation
5. **Barometer** - Pressure at given altitude
6. **Battery life** - Remaining battery life based on power consumption

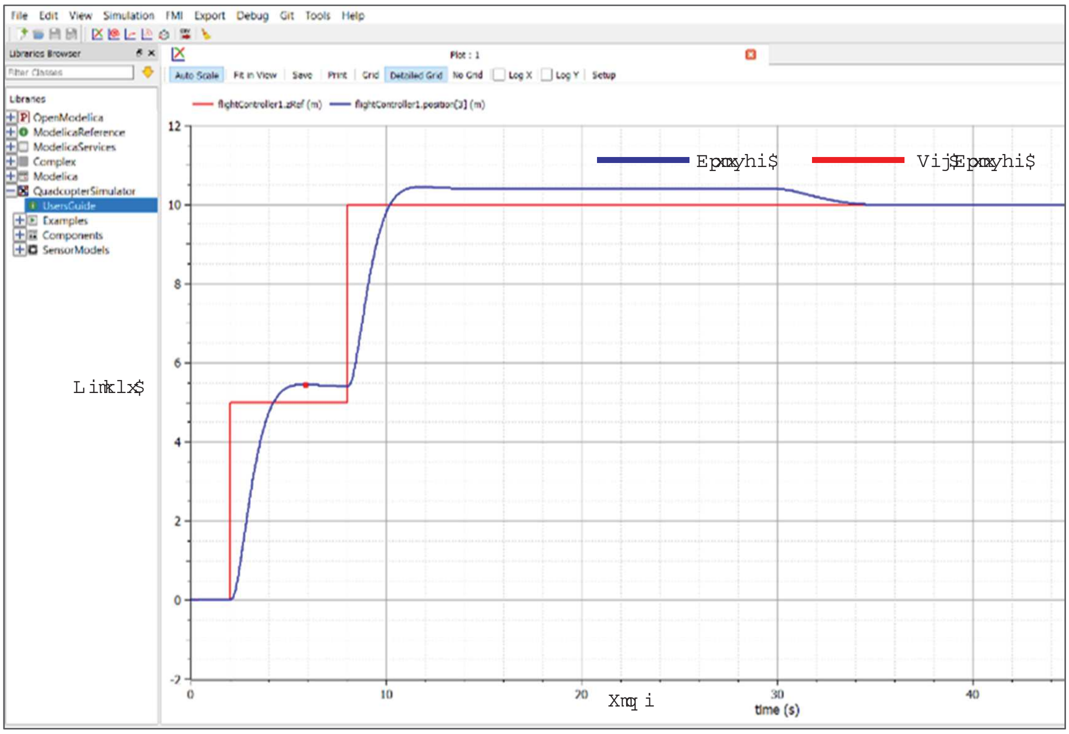
Software-In-The-Loop Simulation in OpenModelica



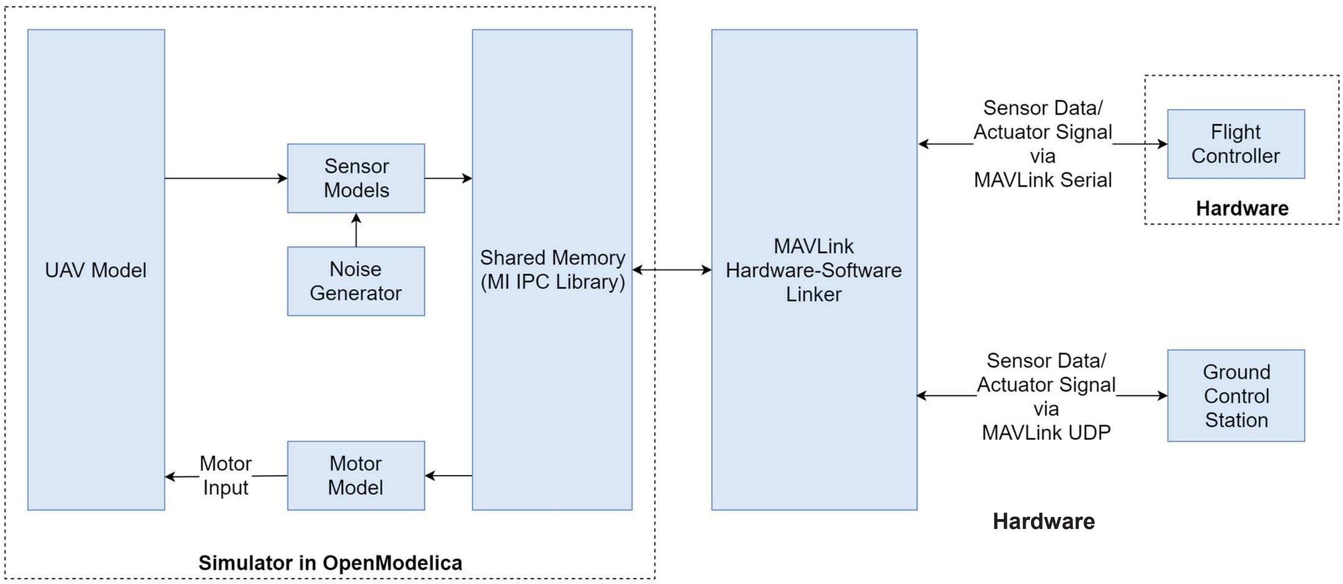
Software-In-The-Loop Demo
(UAVSimulator - GroundControlStation)



Results - Altitude Control



Hardware-In-The-Loop Simulation in OpenModelica

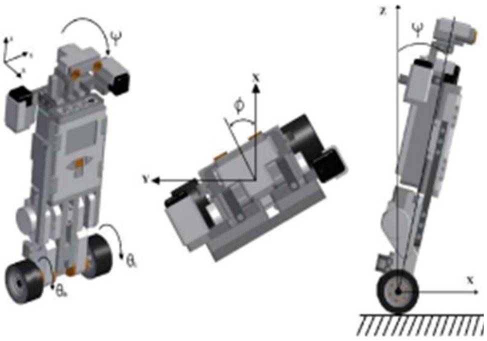


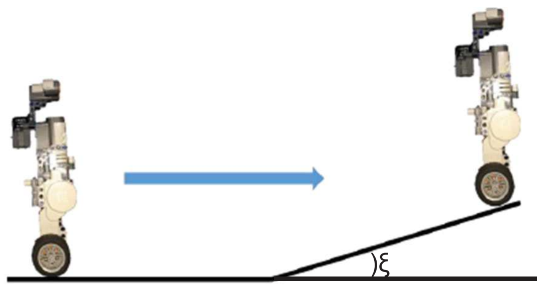
UAV
Movie demo

2. Digital Twin for Self-Balancing Robot

Introduction

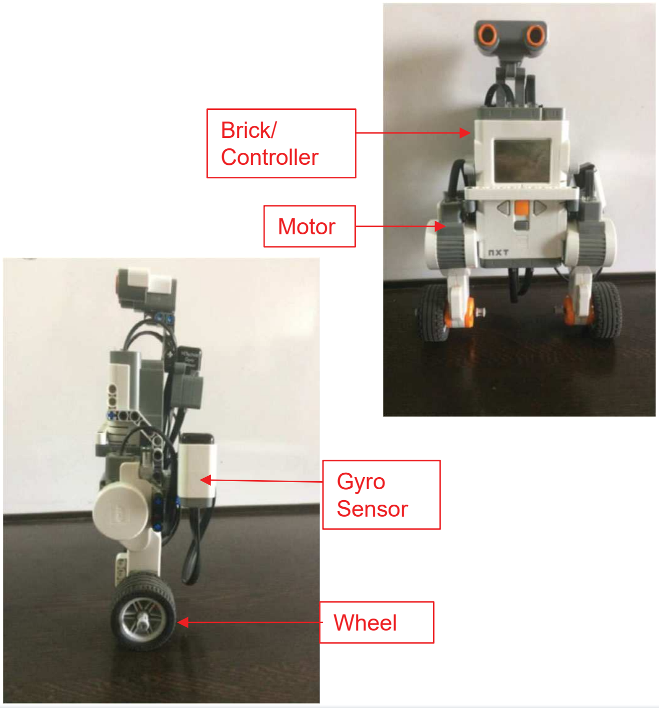
1. Walking **robot** - Lego Mindstorms based bot
2. **Two driving wheels** for position control and fast motion - operated synchronously
3. **Gyro sensor** for **rate** and **angle** of bot inclination ($der(\psi)$, ψ)
4. **Electric actuators (motors)** have angular position sensors (encoders) to measure angular position (θ) of the wheels and spee
5. Angle of **inclination** for the ramp is (ξ)
6. Under-actuated system



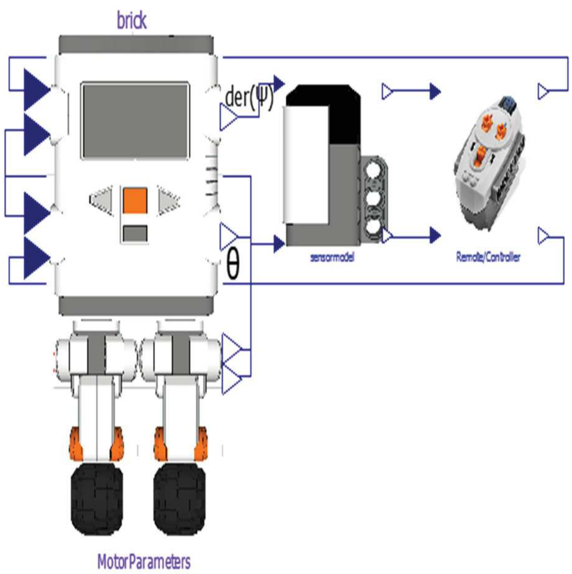


Architecture

Physical System



Digital Twin in OpenModelica

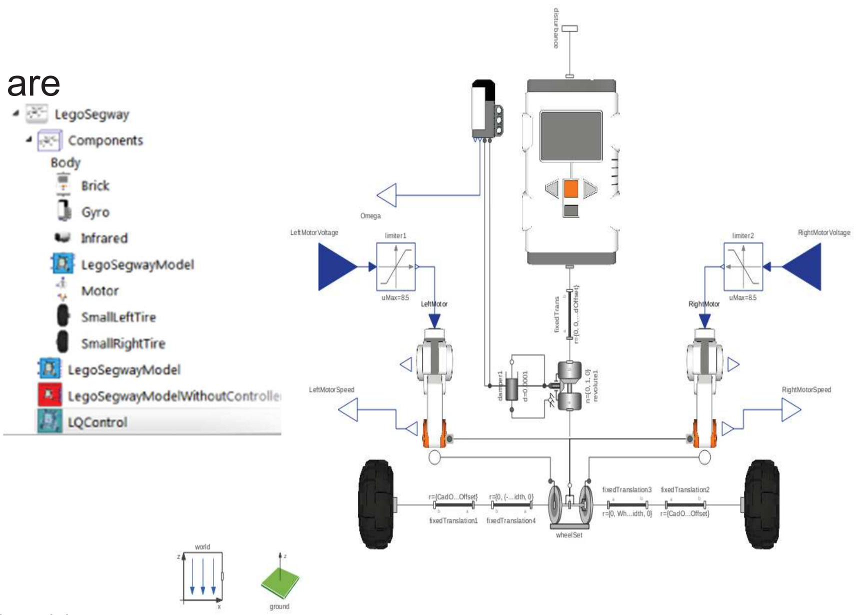


Inspiration from Segway Modelica Model
(Example made by Wolfram for System Modeler)

1. Based on **multibody library** for the bot components.

2. 3D Forces and voltages are supplied as inputs

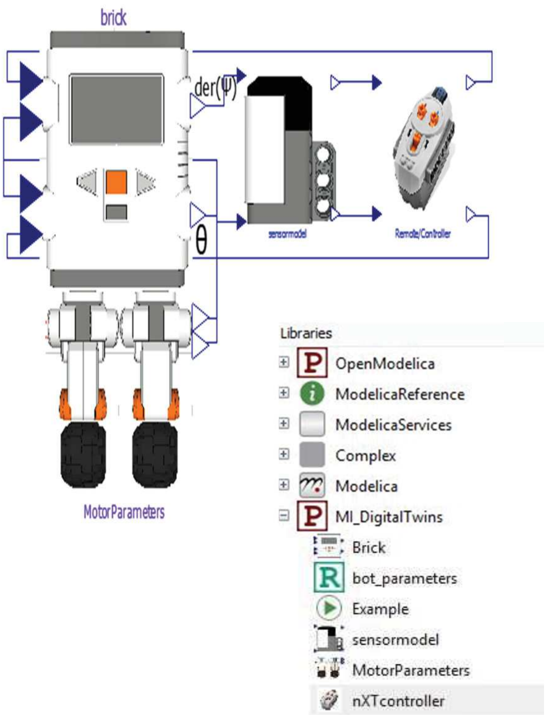
3. Maneuvers the bot on a **flat surface**.



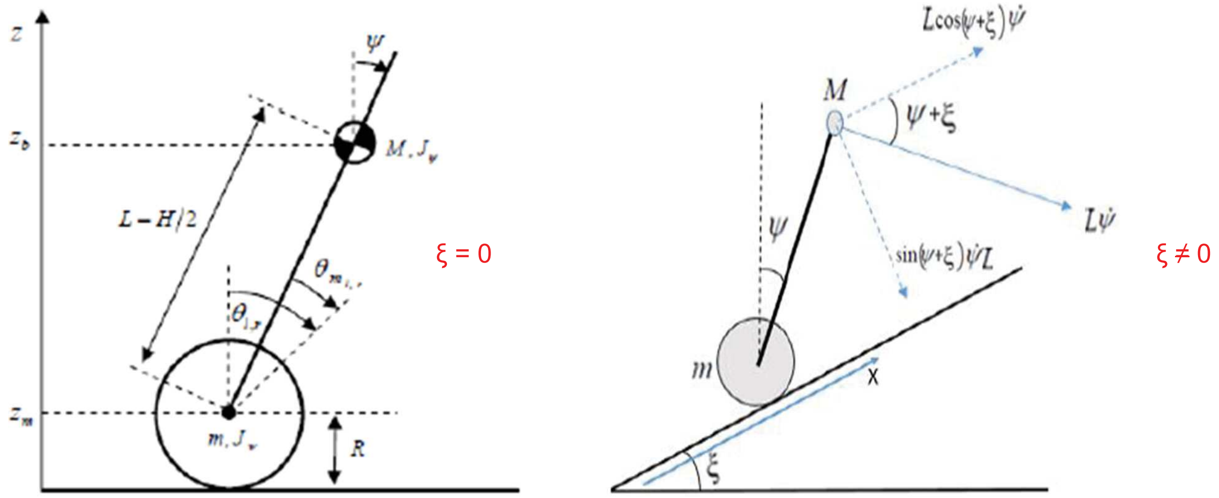
Reference: http://www.wolfram.com/system-modeler/examples/mechanical-engineering/inverted-pendulum-control.html#alert-1

Segway Modelica Model
(Made Using OpenModelica)

1. Based on **Lagrangian equations:**
   a. Energy Conservation (Kinetic and Potential Energy)
2. Captures the **pitching** and wheel forces
3. Maneuvers the bot on an **inclined surface**



Lagrangian Model - Free Body Diagram

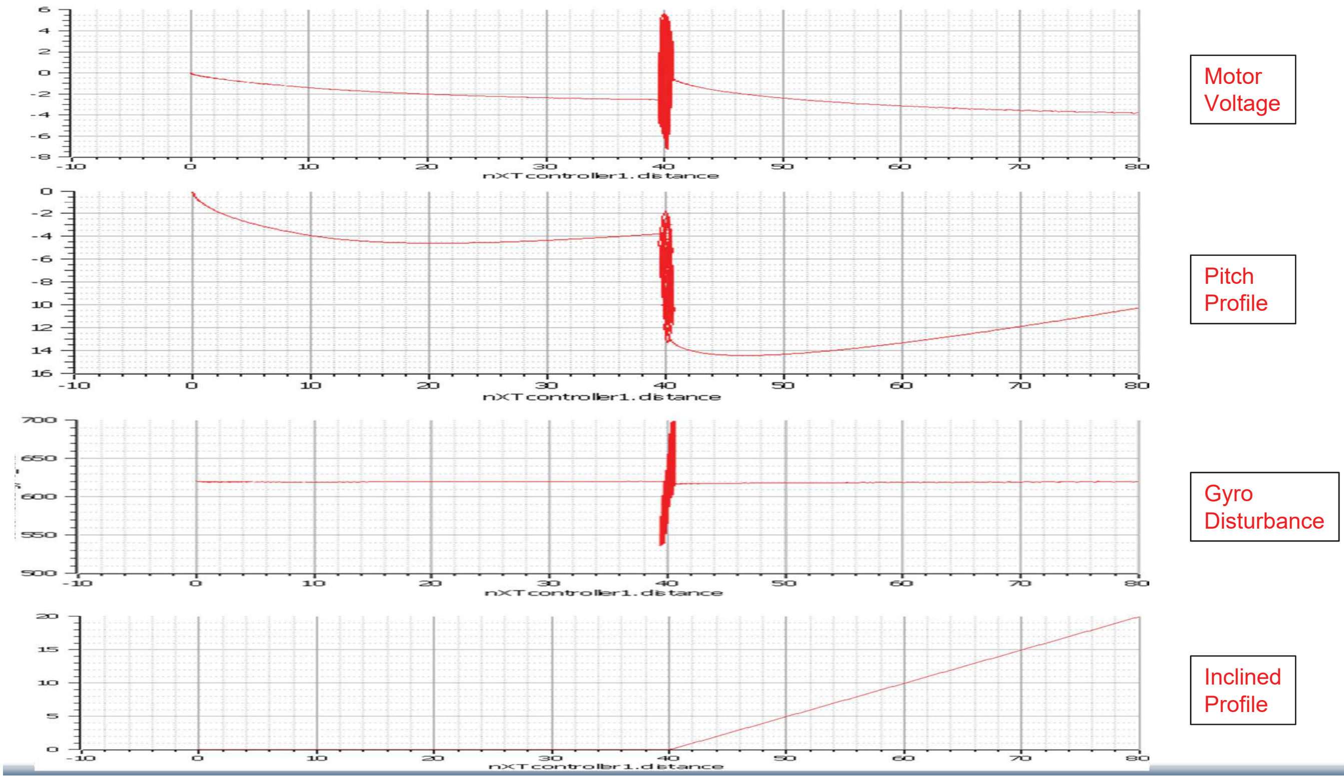


KE_T (translational) + KE_R(rotational) = PE (potential) + L (Lagrangian)

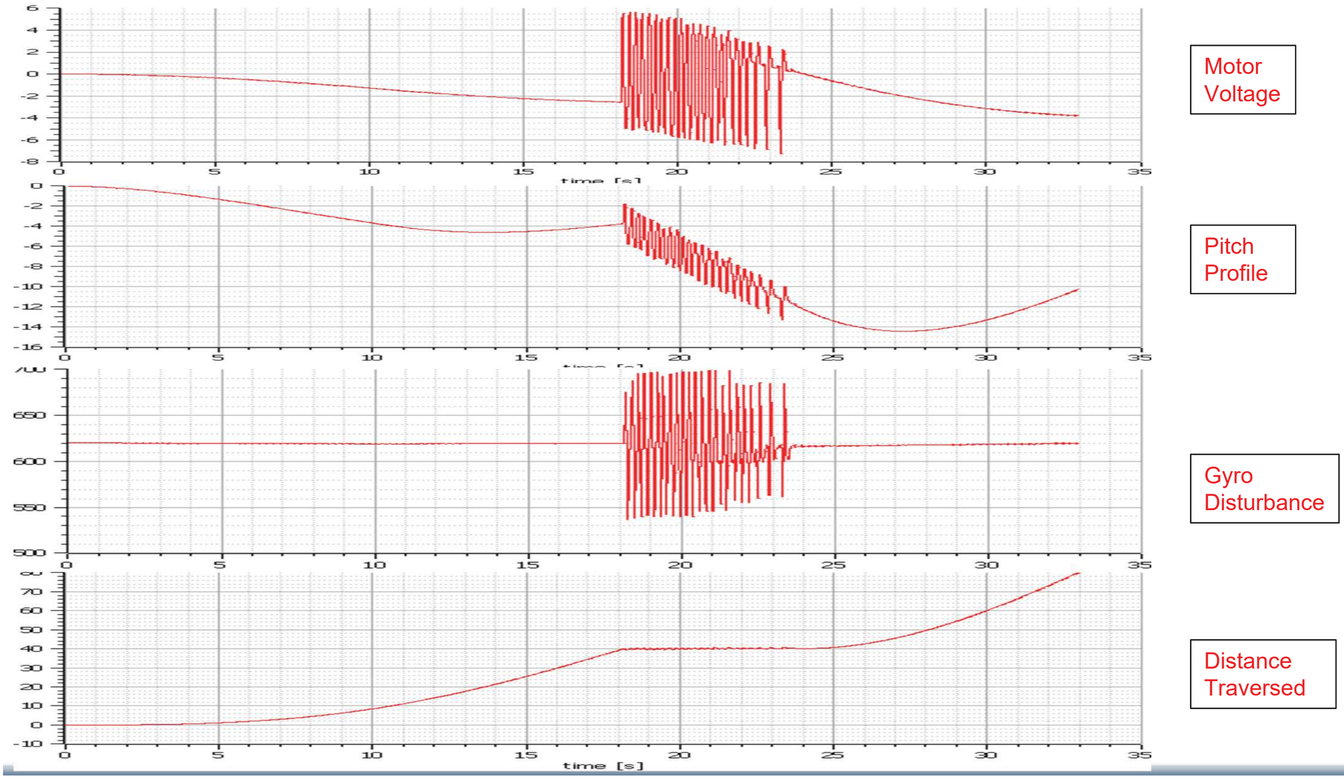
The force equations are as follows :

- $der(\frac{\partial L}{\partial \dot{\theta}})$ - $\frac{\partial L}{\partial \theta}$ = F_θ *(Force on wheel)*
- $der(\frac{\partial L}{\partial \dot{\Psi}})$ - $\frac{\partial L}{\partial \theta}$ = F_ψ *(Force on pitching)*

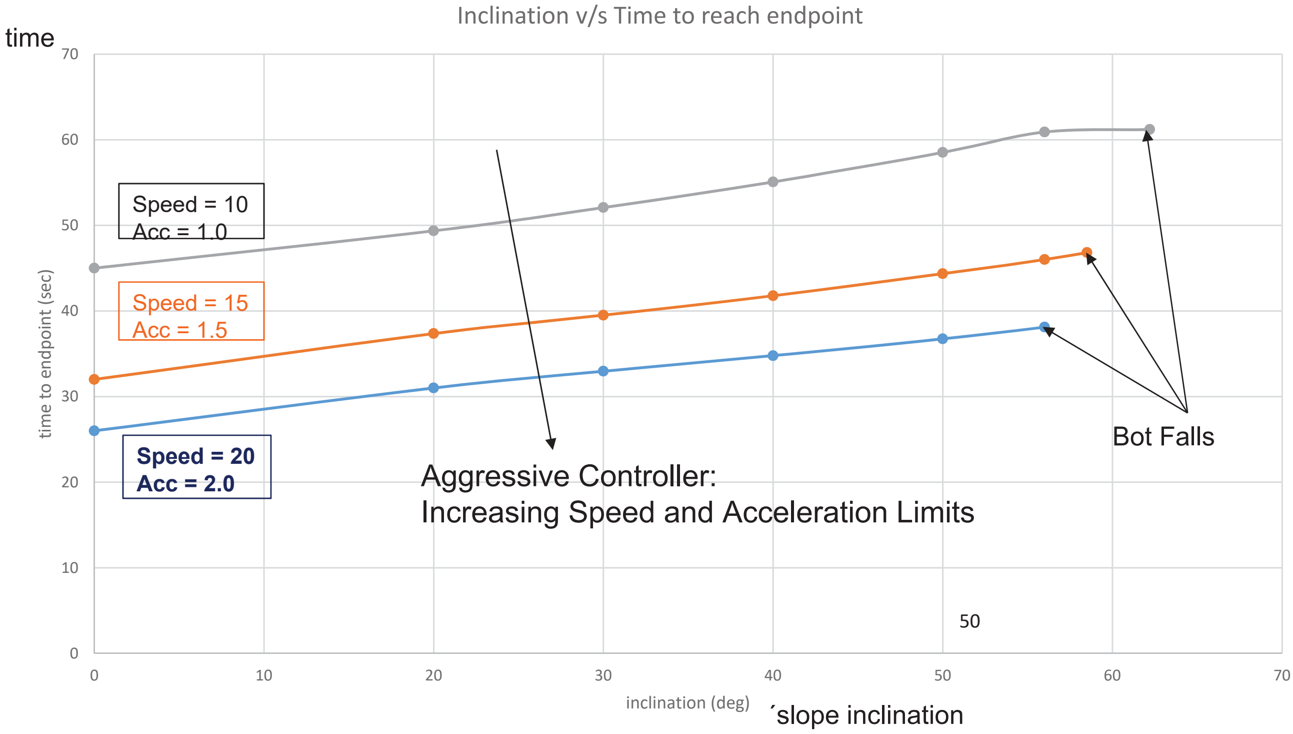
Simulation Results (with respect to Distance)



Simulation Results (with respect to Time)



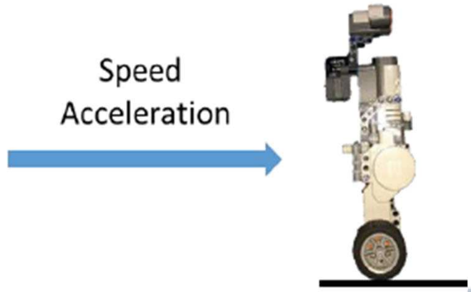
Inclination and Time Characteristics



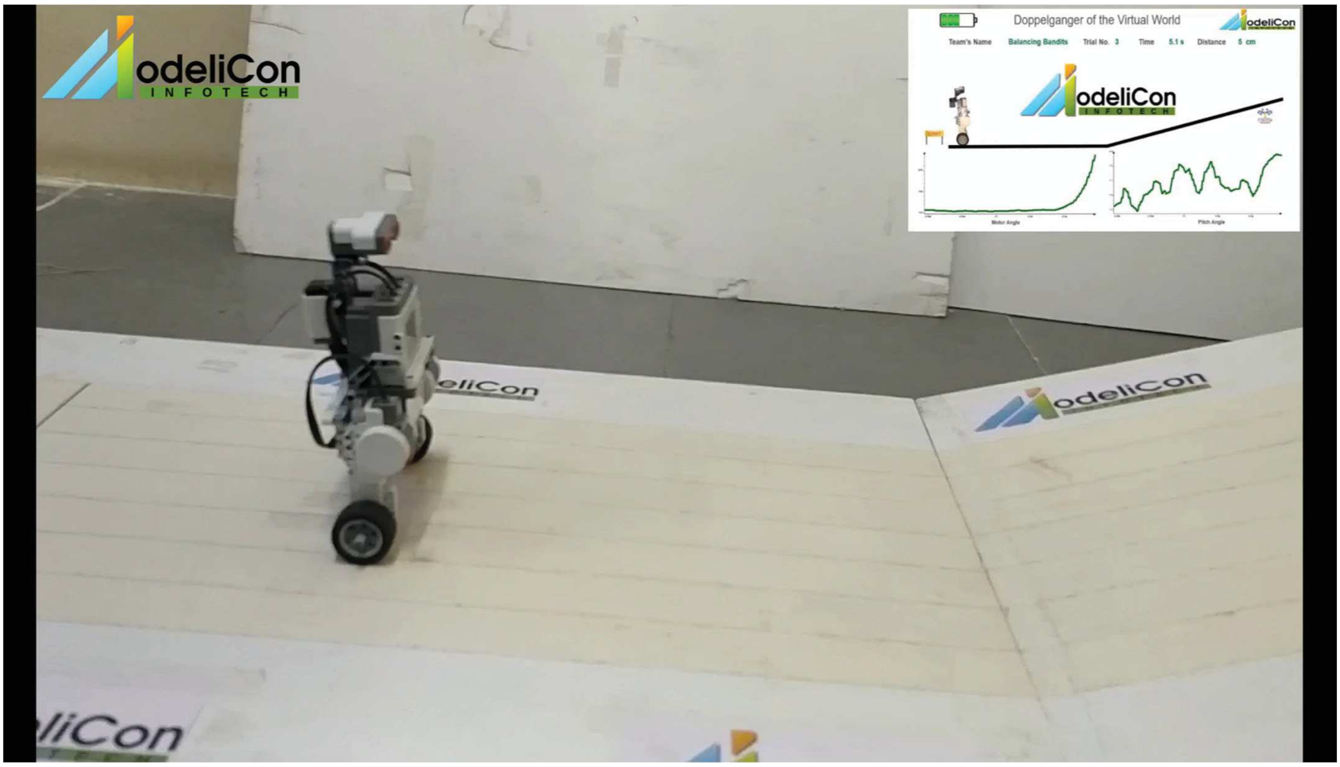
Workshop with Participants on Digital Twin

- **Objective**: To **find** the **optimum velocity** and acceleration limits for quickest travel without falling, by trying and testing
- **Task**: To **simulate** the model at **varying conditions** understanding the relationship between:
  - Velocity
  - Acceleration
  - Inclination of the track
- **Given**: Mathematical **model** (Digital Twin) of the bot

Inclination	0	10	20	30
Speed	S1	S2	S3	S4
Acceleration	A1	A2	A3	A4



Workshop on Digital Twin



Conclusions

1. Leveraged **Modelica/OpenModelica's** modelling **capabilities** and **external** interfacing capabilities for **real time** modelling and simulation of systems
2. Demonstrated **inter-process** communication using serial and shared memory
3. Demonstrated versatile **UAV** and Self-Balancing **Robot** models

Acknowledgment

The authors would like to thank:

- Vasu Goel
- Puneet AC
- Ritesh Sharma
- Ankur Gajjar
- Jal Panchal
- Sreejith Sasidharan

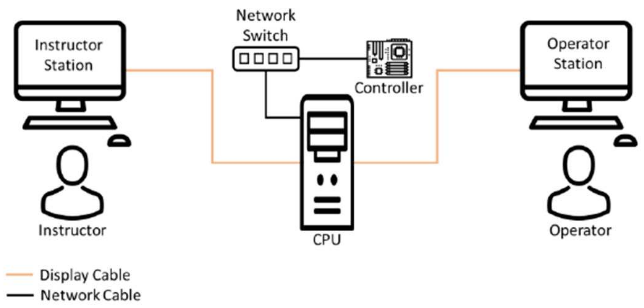
for extending their support in this work

Operator Training Simulators (OTS)

1. Computer-based training system that uses dynamic simulation models of industrial processes.
2. Integrated with emulator of the process plant control system.
3. Elements of an OTS include
   a. Dynamic simulation software
   b. Process model
   c. Instructor interface
   d. Control system integration software
   e. Control system
   f. Replica of operator station
4. An OTS provides a number of scenarios tailor-made to the plant operation. Some examples include:
   a. Runaway reaction
   b. Equipment fouling
   c. Electrical failures
5. Operator gains experience in dealing with unexpected situations arising in plant and hence gains confidence.

OTS Architecture with Modelica

Software Architecture



Hardware Architecture