



MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

# MuJoCo for Advanced Physics Simulation: From manipulators to autonomous vehicles

Duc Cuong Vu

Motion Control and Applied Robotics Laboratory  
School of Electrical and Electronic Engineering,  
Hanoi University of Science and Technology

Email: [vdcuong2002@gmail.com](mailto:vdcuong2002@gmail.com)

Site: [dc-vu.github.io](https://dc-vu.github.io)

PDF available at [github.com/dc-vu/seminar1-hust.git](https://github.com/dc-vu/seminar1-hust.git)

Version: 1.0

Last updated: August 3, 2025



# Seminar on MuJoCo Simulation Framework

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## Motivation:

- High-performance physics simulation for complex, dynamic robotics.
- **MuJoCo**: lightweight, real-time, accurate multi-body dynamics with efficient contact handling.

## Seminar Highlights:

- Introduction to MuJoCo and comparison with traditional simulators (e.g., MATLAB Simulink, Simscape).
- Live demos of:
  - 7-DOF Serial Manipulator (kinematics & control)
  - Stewart Platform (constraint handling & stabilization)
  - Autonomous 3D Robots (AUVs & UAVs: motion planning, optimization, real-time control)

**Control Topics:** System stabilization, real-time implementation, motion planning, optimization-based decision making.

**Duration:** 90 minutes

**Date:** xx/06/2025



# Table of Contents

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## 1 What is MuJoCo?

## 2 Configuration

## 3 Examples

- 7-DOF serial manipulator
- 6-DOF parallel mechanisms
- Unmanned Aerial Vehicle (UAV)
- Autonomous Underwater Vehicle (AUV)

## 4 Q&A





MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

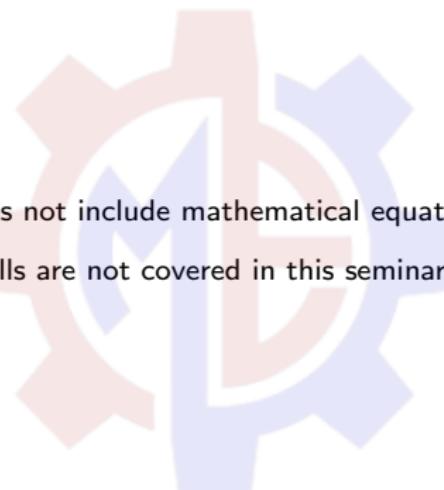
7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A



- This seminar does not include mathematical equations.
- Programming skills are not covered in this seminar.



# Table of Contents

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## 1 What is MuJoCo?

## 2 Configuration

## 3 Examples

- 7-DOF serial manipulator
- 6-DOF parallel mechanisms
- Unmanned Aerial Vehicle (UAV)
- Autonomous Underwater Vehicle (AUV)

## 4 Q&A





# What is MuJoCo?

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

*"MuJoCo is a free and open source physics engine that aims to facilitate research and development in robotics, biomechanics, graphics and animation, and other areas where fast and accurate simulation is needed."* <sup>1, 2</sup>



google-deeprl / mujoco

Code Issues Pull requests Discussions Actions Security Insights

mujoco Public

3 main · 3 Branches · 39 Tags · Go to file · Add file · Code · About

Google DeepMind and copybara GitHub Apply metallic and roughness scaling · 1 commit · yesterday · 1.044 Commits

Fix GitHubActions builds · last month

Prepare for v0.3.1 release · 2 months ago

Bump version to 2.3.3 following the 2.3.2 release · 2 months ago

Apply metallic and roughness scaling factors · yesterday

Add wj\_copies for copying real-valued arrays from mujoco · last week

Enable using sensor and SDF plugins in MJCF · last week

Add texture to rotating cylinders in pulley example · 2 days ago

Remove the Shell plugin and integrate it into the engine · 3 weeks ago

mujoco.org

Readme Apache 2.0 license Activity Custom properties 9.7k stars 115 watching 3k forks

<sup>1</sup><https://mujoco.org/>

<sup>2</sup>Todorov, E., Erez, T., & Tassa, Y. (2012, October). Mujoco: A physics engine for model-based control. In 2012 IEEE/RSJ international conference on intelligent robots and systems (pp. 5026-5033). IEEE.



# Why MuJoCo?

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

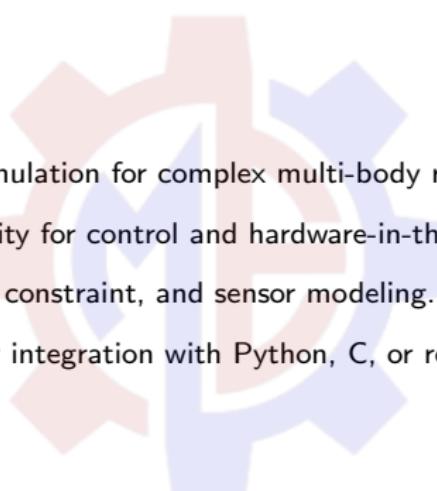
Examples

7-DOF serial  
manipulator  
6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A



- Fast, accurate simulation for complex multi-body robotic systems.
- Real-time capability for control and hardware-in-the-loop (HIL) testing.
- Efficient contact, constraint, and sensor modeling.
- Lightweight: easy integration with Python, C, or real hardware controllers.



# Comparison

MuJoCo -  
Advanced  
Physics  
Simulation  
D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## Comparisons of Simscape, MuJoCo and IsaacSim from 2015

Category	Search Query	Year	Title	Author(s)	Description
Simscape	simscape	2015	Modeling and simulation of robotic hand pressure sensor in Simscape	J. Matela, G. Dume, E. Bascholl	... quantity value changes in Simscape and vice versa. Simscape and Simulink can be used in the realization of schemes, for this Simulink uses conversion blocks to migrate to Simscape. ...
	2023	Modeling and simulation of mechatronic systems using Simscape	J. Matela, G. Dume, E. Bascholl	... quantity value changes in Simscape and vice versa. Simscape and Simulink can be used in the realization of schemes, for this Simulink uses conversion blocks to migrate to Simscape. ...	
MuJoCo	mujoco	2015	Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx	T. Egeci, Y. Tasse, E. Todorov	... the developers of the MuJoCo engine which is included in the comparisons, and indeed a secondary goal of this paper is to identify the strengths and weaknesses of MuJoCo. We have... ...
	2023	Mujoco haptic: A virtual reality system for hand manipulation	V. Kumar, F. Todorov	... 2015 IEEE-RAS 15th International Conference on Humanoid Robots, 2015 - ieeexplore.ieee.org	
IsaacSim	isaac sim	2015	Detection for non-technical loss by smart energy theft with intermediate monitor meter in smart grid	JY Kim, YM Hyung, YG Sung, I. Sim, DJ Kim	... In this work, to detect non-technical loss (NTL) of meter manipulating/manufacturing and bypassing at the same time, we propose the intermediate monitor meter (IMM)-based power... ...
	2023	Non-technical loss detection using deep reinforcement learning for feature cost efficiency and imbalanced dataset	GD, GD	...	

from 2015 - present

from 2023 - present

**IsaacSim is the trend, but ...**



# Resources

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## Mobile Manipulators

Model	Preview
Google Robot	
Hello Robot stretch 2	

## Anybots ANYmal C

Model	Preview
Google Bipedal v0	
Google Bipedal v8	
Boston Dynamics Spot	
Unreal AI	

## Arms

Model	Preview
Franka Emika Panda	
Franka FR3	
Universal Robots UR16e	
Universal Robots UR10e	

## Bipeds

Model	Preview
Ability Cossie	

## Humanoids

Model	Preview
Unreal GI	
Unreal HI	

## Quadrupeds

Model	Preview
Unreal AI	

## Drones

Model	Preview
Skydio X2	
Bitcraze Crazyflie 2	

## Biomechanical

Model	Preview
Fruitfly	

## Grippers & Hands

Model	Preview
Shadow E3M5	
RoboIQ 2F-05	

## Arms



# Model description

MuJoCo -  
Advanced  
Physics  
Simulation  
D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

MuJoCo can load XML model files in its native MJCF format, as well as in the popular but more limited URDF format.

```
1 <mujoco>
2   <default class="main">
3     <geom rgba="1 0 0 1"/>
4     <default class="sub">
5       <geom rgba="0 1 0 1"/>
6     </default>
7   </default>
8
9   <worldbody>
10    <geom type="box"/>
11    <body childclass="sub">
12      <geom type="ellipsoid"/>
13      <geom type="sphere" rgba="0 0 1 1"/>
14      <geom type="cylinder" class="main"/>
15    </body>
16  </worldbody>
17</mujoco>
```



# Programming

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

MuJoCo has a C API and is intended for researchers and developers. The runtime simulation module is tuned to maximize performance and operates on low-level data structures that are preallocated by the built-in XML compiler.

The Python bindings are distributed as the `mujoco` package on PyPI. These are low-level bindings that are meant to give as close to a direct access to the MuJoCo library as possible.

```
scripts > ⚡ validate_kinematics.py > ...
* 1  import mujoco as mj
* 2  from mujoco.glfw import glfw
* 3  import numpy as np
* 4  import os
* 5  from scipy.spatial.transform import Rotation
* 6
* 7  class MujocoSim:
* 8
* 9      def __init__(self, xml_name, time_step, simulation_time, fps):
10
11          #get the full path
12          dirname = os.path.dirname(__file__)
13          abspath = os.path.join(dirname + "/" + xml_name)
14
15          #MuJoCo data structures
16          self.model = mj.MjModel.from_xml_path(abspath) # MuJoCo model
17          if time_step is not None:
18              self.model.opt.timestep = time_step
19          self.data = mj.MjData(self.model) # MuJoCo data
20          self.cam = mj.MjvCamera() # Abstract camera
21          self.opt = mj.MjvOption() # visualization options
22
23          self.xml_path = abspath
24          self.simulation_time = simulation_time
25          self.fps = fps
26
27          # Print camera config
28          self.print_camera_config = False #set to True to print camera config
29          | #this is useful for initializing view of the model)
30
```



# Learning

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

The documents about MuJoCo could be found at <sup>3</sup>.

The screenshot shows the MuJoCo documentation website. The top navigation bar includes links for Overview, Computation, Modeling, XML Reference, Programming, API Reference, Python, MJX, Unity Plug-in, Model Gallery, Changelog, LINKS, and GitHub. The main content area features a large blue arrow graphic pointing upwards. The title "Overview" is prominently displayed in blue. Below it is a section titled "Introduction" with a brief description of what MuJoCo is and its capabilities. A sidebar on the right lists various topics under "ON THIS PAGE" such as Key features, Model instances, Examples, Model elements, Options, Assets, and many others.

**Overview**

## Introduction

MuJoCo stands for Multi-Joint dynamics with Contact. It is a general purpose physics engine that aims to facilitate research and development in robotics, biomechanics, graphics and animation, machine learning, and other areas that demand fast and accurate simulation of articulated structures interacting with their environment. Initially developed by RobotiC LLC, it was acquired and made freely available by DeepMind in October 2021, and open sourced in May 2022. The MuJoCo codebase is available at the [google-deepmind/mujoco](https://github.com/deepmind/mujoco) repository on GitHub.

MuJoCo is a C/C++ library with a C API, intended for researchers and developers. The runtime simulation module is tuned to maximize performance and operates on low-level data structures which are preallocated by the built-in XML parser and compiler. The user defines models in the native MJCF scene description language – an XML file format designed to be as human readable and editable as possible. URDF model files can also be loaded. The library includes interactive visualization with a native GUI, rendered in OpenGL. MuJoCo further exposes a large number of utility functions for computing physics-related quantities.

MuJoCo can be used to implement model-based computations such as control synthesis, state estimation, system identification, mechanism design, data analysis through inverse dynamics, and parallel sampling for machine learning applications. It can also be used as a more traditional simulator, including for gaming and interactive virtual environments.

## Key features

MuJoCo has a long list of features. Here we outline the most notable ones.

**Generalized coordinates combined with modern contact dynamics**

Physics engines have traditionally separated in two categories. Robotics and biomechanics engines use efficient and accurate recursive algorithms in generalized or joint coordinates.

<sup>3</sup><https://mujoco.readthedocs.io/en/stable/overview.html>



# Table of Contents

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## 1 What is MuJoCo?

## 2 Configuration

## 3 Examples

- 7-DOF serial manipulator
- 6-DOF parallel mechanisms
- Unmanned Aerial Vehicle (UAV)
- Autonomous Underwater Vehicle (AUV)

## 4 Q&A





# MATLAB Simscape multi-body traditional approach

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

- First step, install the 50GB of MATLAB (include 20GB for installer and 30GB for unzip files).
- Secondly, what if you don't have the budget to purchase a license???
- If you are a rich kid, in the next step, you spend 1 minute to 5 minutes for starting the MATLAB, and about more than one minutes for open the first Simulink windows.
- Finally, designing is quite easy, but ..., compile time and simulation time are significant! If a block of Matlab Function is modified, MATLAB re-compiles the simulation.



# Installation

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A



```
1 pip install mujoco
```



# Configuration

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

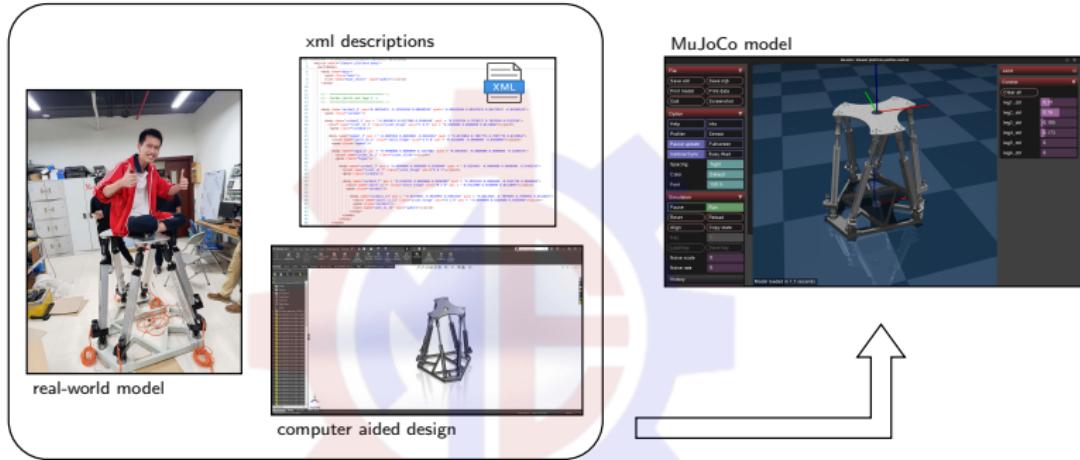
7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A



Process of replicating a real-world Stewart platform model into a MuJoCo simulation environment: Starting from the real-world model, the system is first designed in a computer-aided design (CAD) software (SolidWorks, Inventor). The CAD design is then translated into XML descriptions compatible with MuJoCo. Finally, the structured XML is used to construct the MuJoCo model with the actuators, sensors, ..., enabling high-fidelity physics-based simulation and control testing.



# Table of Contents

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## 1 What is MuJoCo?

## 2 Configuration

## 3 Examples

- 7-DOF serial manipulator
- 6-DOF parallel mechanisms
- Unmanned Aerial Vehicle (UAV)
- Autonomous Underwater Vehicle (AUV)

## 4 Q&A





# 7-DOF serial manipulator

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

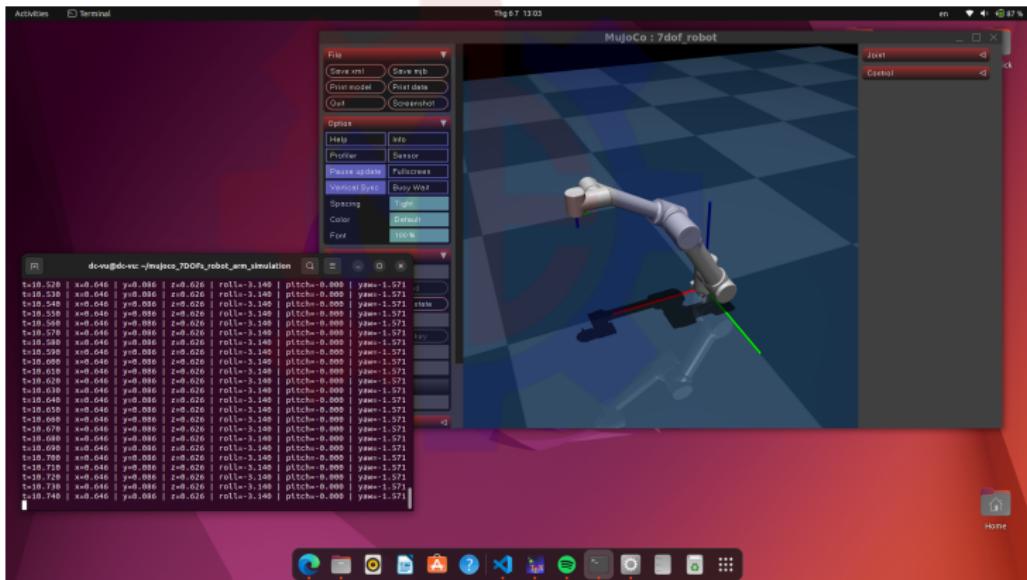
6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

7-DOF serial manipulator simulation in MuJoCo, the open-source is available at <sup>4</sup>  
(Contributor: Duc-Cuong Vu)



<sup>4</sup>[https://github.com/dc-vu/mujoco\\_7DOFs\\_robot\\_arm\\_simulation.git](https://github.com/dc-vu/mujoco_7DOFs_robot_arm_simulation.git)



# Robot definition

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## Robot XML definition

```
1  <asset>
2      <mesh name="base_mesh" file="base.stl" scale="0.001 0.001 0.001"/>
3      <mesh name="motor110_mesh" file="motor110.stl" scale="0.001 0.001 0.001"/>
4      <mesh name="motor70_mesh" file="motor70.stl" scale="0.001 0.001 0.001"/>
5      <mesh name="link2_mesh" file="link2.stl" scale="0.001 0.001 0.001"/>
6      <mesh name="link3_mesh" file="link3.stl" scale="0.001 0.001 0.001"/>
7  </asset>
8
9
10 <!-- Link 1 -->
11   <body name="link1">
12     <joint name="joint1" type="hinge" axis="0 0 1"/>
13     <geom type="mesh" mesh="motor110_mesh"/>
14
15   <!-- Link 2 -->
16   <body name="link2">
17     <joint name="joint2" type="hinge" axis="0 0 1" />
18     <geom type="mesh" mesh="motor110_mesh"/>
19
20       <!-- Robot tree-based definition => easy -->
21
22 <!-- Sensing -->
23 <sensor>
24   <framepos name="ee_pos" objtype="body" objname="ee"/>
25   <framequat name="ee_quat" objtype="body" objname="ee"/>
26 </sensor>
```





# Inverse kinematics validation

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration  
Examples

7-DOF serial  
manipulator

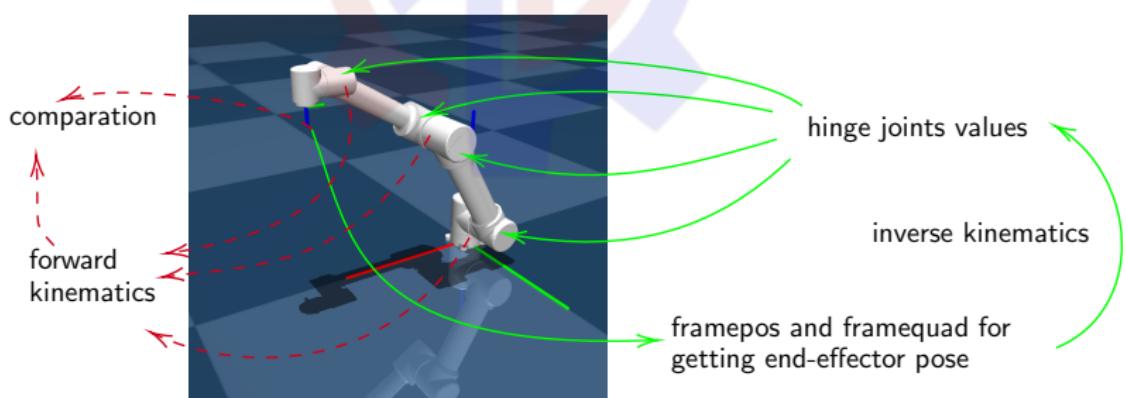
6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

```
1 ee_pos = data.body("ee").xpos
2 eex, eey, eez = ee_pos
3 ee_xmat_flat = data.body("ee").xmat
4 ee_R = np.array(ee_xmat_flat).reshape(3, 3)
5 ori = R.from_matrix(ee_R).as_euler('xyz')
6 print(f"t={t:.3f} | x={eex:.3f} | y={eey:.3f} | z={eez:.3f} | "
7       f"roll={ori[0]:.3f} | pitch={ori[1]:.3f} | yaw={ori[2]:.3f}")
8 # Inverse kinematics calculation here => do not straightforward
9 for i in range(7):
10     data.joint(f"joint{i+1}").qpos = joint_demand[i] - angles_offset[i]
```





# Remaining problems

MuJoCo -  
Advanced  
Physics  
Simulation  
D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

- The inverse kinematics is not real-time, because an optimization problem is used for solving the joint states

```
1 def levenberg_marquardt_ik_dh(q_init, desired_pose):  
2     result = least_squares(  
3         ik_loss,          # the cost function  
4         q_init,           # initial guess  
5         args=(desired_pose,),  
6         method='trf',    # Change to 'lm' if you want to use Levenberg-Marquardt  
7         ftol=1e-6,        # Tolerance for the cost function  
8         xtol=1e-6,        # Tolerance for the solution  
9         gtol=1e-6,        # Tolerance for the gradient  
10        max_nfev=200    # Maximum number of function evaluations  
11    )  
12    return result.x, result.cost
```

This function is implemented in Python, the solver is trust region method or Levenberg-Marquardt. Thus the real-time purpose is not guaranteed. This problem could be addressed by utilizing the C code in the remaining part of this talk.

- Need actuators



# 6-DOF parallel mechanisms - Stewart platform

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

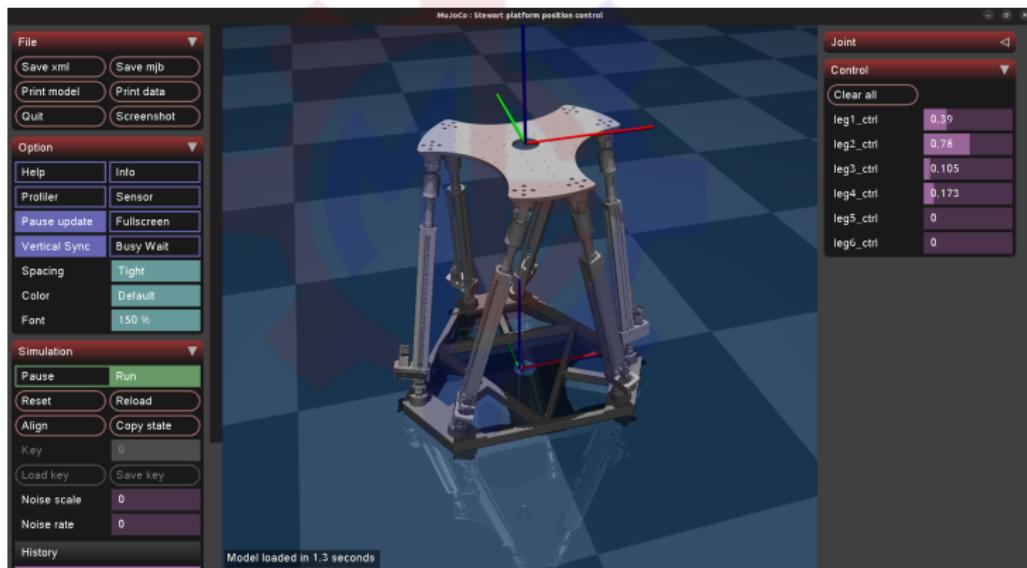
6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

Stewart platform simulation in MuJoCo, the open-source is not available.  
(Contributors: Viet Khanh Nguyen, Duc Cuong Vu)





# Closed loop kinematics chain

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

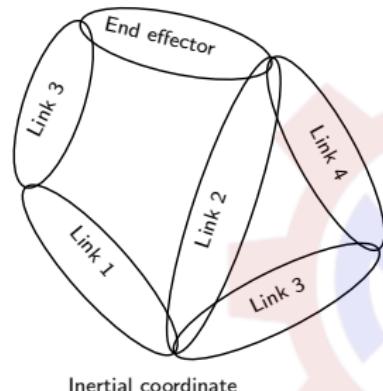
6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

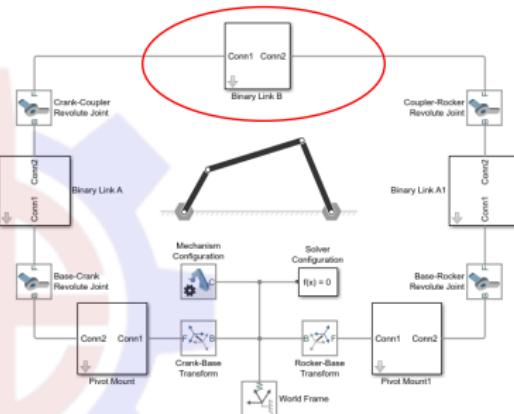
Autonomous  
Underwater  
Vehicle (AUV)

Q&A

closed loop kinematic chain



"Which ensures that the connection is valid?



The parallel mechanisms could be described in MuJoCo by equality attribute as follows

```
1 <mujoco model="Parallel mechanisms equality">
2   <equality>
3     <weld name="w_1" site1="site1" site2="site2"></weld>
4     <weld name="w_2" site1="site2" site2="site4"></weld>
5   </equality>
6 </mujoco>
```



# Optimization-based kinematics - (ctypes in Python - C code for real-time implementation)

MuJoCo -  
Advanced  
Physics  
Simulation  
D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

The algorithm is introduced in 2.2. *Kinematics analysis* of the publication <sup>5</sup>

```
kinematics > ⏷ forward_kinematics.py > ...
1 import ctypes
2 import numpy as np
3 import time
4
5 lib = ctypes.cDLL.LoadLibrary('kinematics/c_code/kinematics.so')
6
7 lib.lm_optimize.argtypes = [
8     ctypes.POINTER(ctypes.c_float),
9     ctypes.POINTER(ctypes.c_float),
10    ctypes.c_bool,
11    ctypes.POINTER(ctypes.c_int)
12 ]
13 lib.lm_optimize.restype = None
14
15 x_array = (ctypes.c_float * 6)(0, 0, 1.1, 0, 0, 0) # initial guess
16 # l_target = (ctypes.c_float * 6)(0.9287322759628296, 0.9264336228370667, 0.9265220761299133, 0.9288733601570129, 0.9278072714805603, 0.9
17 # [1.05833114 0.92127699 0.92419712 0.92959993 0.92540508 0.92480616]
18 l_target = (ctypes.c_float * 6)(1.05833114, 0.92127699, 0.92419712, 0.92959993, 0.92540508, 0.92480616) # target lengths
19
20 num_iter = ctypes.c_int(0)
21
22 start_time = time.time()
23 lib.lm_optimize(x_array, l_target, False, ctypes.byref(num_iter))
24 end_time = time.time()
25
26 x_result = [x_array[i] for i in range(6)]
27 print("Optimized pose (x):", x_result)
28 print("Number of iterations:", num_iter.value)
29 print(f"Execution time: {(end_time - start_time)*1e6:.2f} microseconds")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● (.venv) dc-vu@dc-vu:~/stewart_platform$ /home/dc-vu/stewart_platform/.venv/bin/python /home/dc-vu/stewart_platform/kinematics/forward_kinematics.py
Optimized pose (x): [-0.050542283803224564, 0.1859666407108307, 1.093039724156982, 0.07483875751495361, -0.05862480774521828, 0.26324185729026794]
Number of iterations: 2
Execution time: 26.23 microseconds
❶ (.venv) dc-vu@dc-vu:~/stewart_platform$
```

<sup>5</sup>Vu, D. C., Nguyen, T. L., & Nguyen, D. H. (2025). A novel approach of Consensus-based Finite-time Distributed Sliding Mode Control for Stewart platform manipulators motion tracking. *Results in Engineering*, 25, 103872.



## Control purpose → The upper platform track a reference

From the desired pose of the upper platform, by the inverse kinematics, the desired length of the actuators is calculated.

In the aim of this seminar, a simple PID controller is design to control the length of the actuators.

```
1 # Get data from sensors ...
2 sensor_value = np.array([
3     data.sensor('leg1_len_sensor').data,
4     data.sensor("leg2_len_sensor").data,
5     ...
6 ]).reshape(-1) + self.legs_offset
7
8 # Define the controller (PID in this case) ...
9 self.controller = StewartPlatformPIDController(Kp, Ki, Kd, self.time_step)
10
11 # Apply the control signal to the model via data.ctrl
12 data.ctrl = self.controller.solve(errors)
```



# Peripherals connection

MuJoCo -  
Advanced  
Physics  
Simulation  
D.C. Vu

What is  
MuJoCo?  
Configuration  
Examples  
7-DOF serial  
manipulator  
6-DOF  
parallel  
mechanisms  
Unmanned  
Aerial Vehicle  
(UAV)  
Autonomous  
Underwater  
Vehicle (AUV)

Q&A

Get pose of the joystick as the references values of the upper platform.

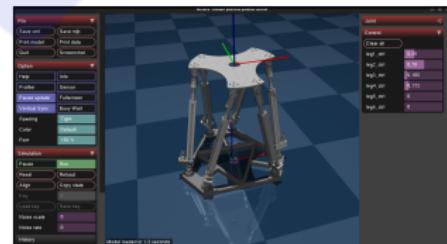
```
1 import pygame
2 pygame.init()
3 pygame.joystick.init()

4
5 roll = normalize_euler_workingspace(round_to_step(self.joystick.get_axis(0)))
6 pitch = -normalize_euler_workingspace(round_to_step(self.joystick.get_axis(1)))
7 yaw = -normalize_euler_workingspace(round_to_step(self.joystick.get_axis(5)))

8
9 z = normalize_pos_workingspace(round_to_step(self.joystick.get_axis(2)))
10 y = normalize_pos_workingspace(round_to_step(self.joystick.get_axis(3)))
11 x = normalize_pos_workingspace(round_to_step(self.joystick.get_axis(4)))
```



read data from flight joystick  
evdev  
pygame





# Peripherals connection

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

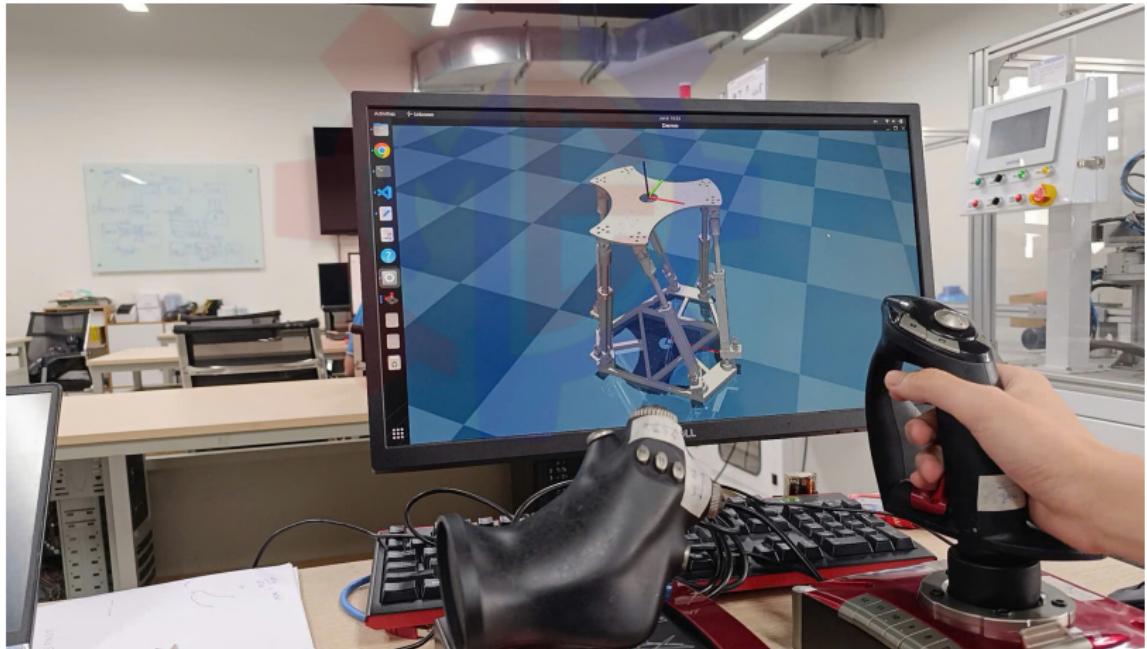
Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

The source code for this project is not publicly available.

For access or further information, please contact Viet Khanh Nguyen or Duc Cuong Vu.





# Skydio X2 model

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

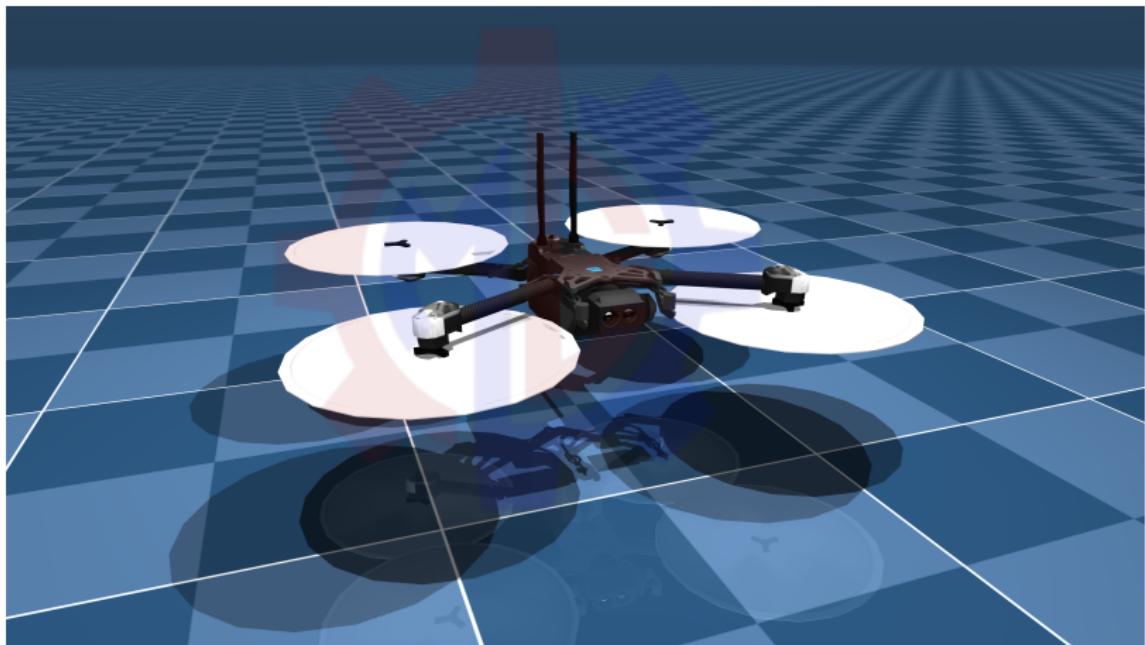
6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

The model is available at <sup>6</sup>



<sup>6</sup>text[https://github.com/google-deepmind/mujoco\\_menagerie/blob/main/skydio\\_x2/x2.xml](https://github.com/google-deepmind/mujoco_menagerie/blob/main/skydio_x2/x2.xml) ↗ ↘ ↙ ↘



# Skydio X2 model

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator  
6-DOF  
parallel  
mechanisms  
Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

The AUV is a standard model. Actuator allocation can be found in the `<actuator>` tag below.

In addition, an IMU sensor is used in this model.

```
1 <actuator>
2   <motor class="x2" name="thrust1" site="thrust1" gear="0 0 1 0 0 -.0201"/>
3   <motor class="x2" name="thrust2" site="thrust2" gear="0 0 1 0 0 .0201"/>
4   <motor class="x2" name="thrust3" site="thrust3" gear="0 0 1 0 0 .0201"/>
5   <motor class="x2" name="thrust4" site="thrust4" gear="0 0 1 0 0 -.0201"/>
6 </actuator>
7
8 <sensor>
9   <gyro name="body_gyro" site="imu"/>
10  <accelerometer name="body_linacc" site="imu"/>
11  <framequat name="body_quat" objtype="site" objname="imu"/>
12 </sensor>
```



# Peripherals connection for UAV control

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

The source code for this project is not publicly available.

For access or further information, please contact Viet Khanh Nguyen.





# Omni-directional Intelligent Navigation model

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

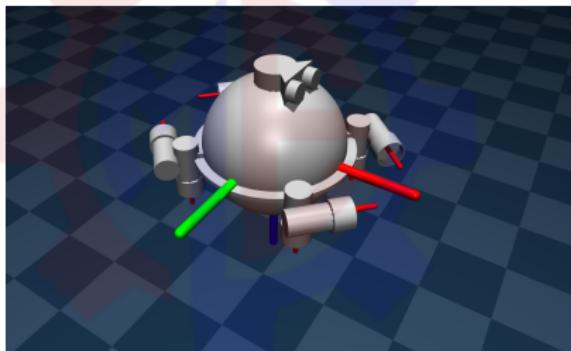
6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

This repository presents a simulation framework for an Autonomous Underwater Vehicle (AUV) that combines Model Predictive Control (MPC), Control Barrier Functions (CBF) for robust path tracking and obstacle avoidance in dynamic underwater environments. The theories are shown in the publication <sup>7</sup>



The source code for this project is not publicly available.

For access or further information, please contact Duc Cuong Vu.

---

<sup>7</sup>Pham, M. D., Vu, D. C., Nguyen, T. T. H., Nguyen, T. V. A., Vu, M. N., & Nguyen, T. L. (2025). CBFs-based Model Predictive Control for Obstacle Avoidance with Tilt Angle Limitation for Ball-Balancing Robots. IEEE Access.



# ODIN actuators

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

The ODIN is driven by 8 thruster, it could be defined as follows

```
1 <actuator>
2   <general name = "thruster1" site="thruster1_site" gear="0 0 1 0 0 0" />
3   <general name = "thruster2" site="thruster2_site" gear="0 0 1 0 0 0" />
4   <general name = "thruster3" site="thruster3_site" gear="0 0 1 0 0 0" />
5   <general name = "thruster4" site="thruster4_site" gear="0 0 1 0 0 0" />
6   <general name = "thruster5" site="thruster5_site" gear="0 0 1 0 0 0" />
7   <general name = "thruster6" site="thruster6_site" gear="0 0 1 0 0 0" />
8   <general name = "thruster7" site="thruster7_site" gear="0 0 1 0 0 0" />
9     <general name = "thruster8" site="thruster8_site" gear="0 0 1 0 0 0" />
10 </actuator>
```

and the general force and moment of underwater environment

```
1 <actuator>
2   <general name="Force_X" site="odin_site" gear="1 0 0 0 0 0" />
3   <general name="Force_Y" site="odin_site" gear="0 1 0 0 0 0" />
4   <general name="Force_Z" site="odin_site" gear="0 0 1 0 0 0" />
5   <general name="Moment_K" site="odin_site" gear="0 0 0 1 0 0" />
6   <general name="Moment_M" site="odin_site" gear="0 0 0 0 1 0" />
7   <general name="Moment_N" site="odin_site" gear="0 0 0 0 0 1" />
8 </actuator>
```



# Optimization-based decision-making

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

Following the SNAME(1950) notation, a general underwater vehicle could be controlled by consider the control input as the forces and moments at the origin. Thus, the force allocation for thruster could be achieved by an energy-optimization decision.

```
from scipy.optimize import minimize, Bounds  
  
def objective(x): # Objective: minimize  $x^T x$   
    return np.dot(x, x)  
  
def eq_constraint(x): # Equality constraint:  $E x = A$   
    return force_allocation(x) - FandM  
  
bounds = Bounds([0, 0, 0, 0, -np.inf, -np.inf, -np.inf, -np.inf], [np.inf]*8)  
linear_constraint = {'type': 'eq', 'fun': eq_constraint}  
  
x0 = thurstter_prev # Initial guess  
  
result = minimize(objective, x0, method='SLSQP',  
                  constraints=[linear_constraint],  
                  bounds=bounds)
```



# Model Predictive Control with CasADI

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

```
1 import casadi as ca
2 for j in range(N):
3     # Apply dynamics to get next state
4     x_next = rk4_step(x[:, j], u[:, j], dtc)
5     opti.subject_to(x[:, j+1] == x_next) # Dynamics constraint
6
7     # Tracking error: difference between current position and reference
8     e = x[0:3, j] - QR[:, j]
9     # Add to cost (only position error considered)
10    cost += ca.mtimes([e.T, Q, e])
11    # Note: control effort term ( $u.T * R * u$ ) can be added here
12
13    dist = obs_detected * ((x_next[0] - x_obs)**2 +
14                           (x_next[1] - y_obs)**2 + (x_next[2] - z_obs)**2 - r_obs**2)
15
16    # Update obstacle position based on velocity
17    x_obs += current_vel[0]*dtc
18    y_obs += current_vel[1]*dtc
19    z_obs += current_vel[2]*dtc
20
21    # Change in distance between time steps
22    diff_dist = dist - dist_before
23    dist_before = dist # Update for next step
24
25    # CBF constraint to ensure distance grows fast enough
26    opti.subject_to(diff_dist >= -gamma*dist)
```



# Model Predictive Control with CasADI

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

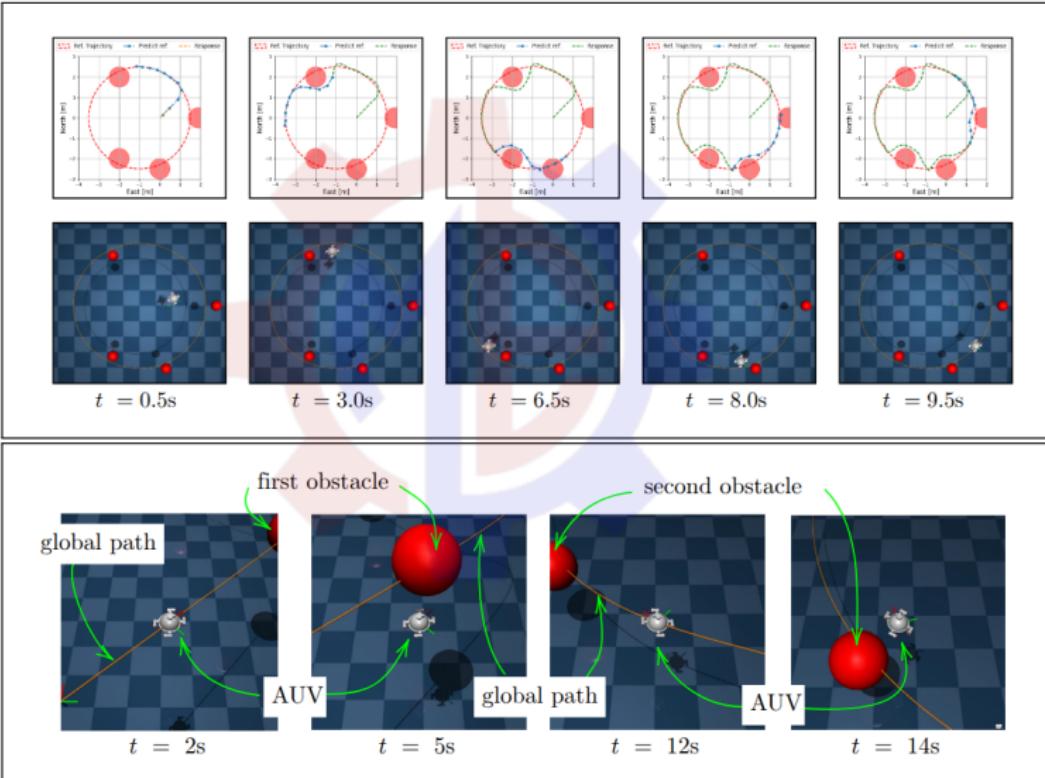
7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A





# Model Predictive Control with CasADI

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

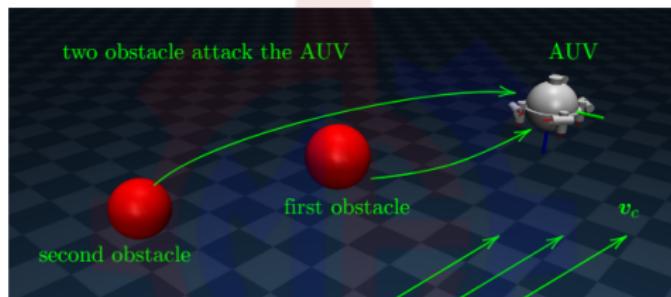
7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A





# Model Predictive Control with CasADI

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

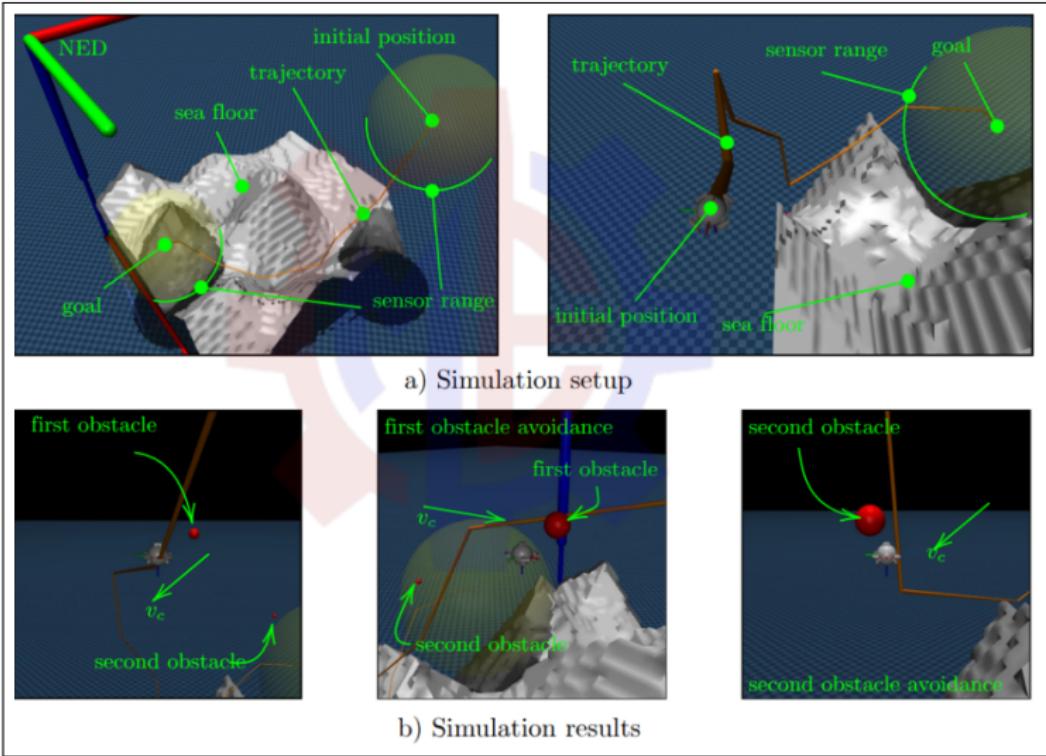
7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A





# Table of Contents

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

## 1 What is MuJoCo?

## 2 Configuration

## 3 Examples

- 7-DOF serial manipulator
- 6-DOF parallel mechanisms
- Unmanned Aerial Vehicle (UAV)
- Autonomous Underwater Vehicle (AUV)

## 4 Q&A





# Thanks for your attention! Any questions?

MuJoCo -  
Advanced  
Physics  
Simulation

D.C. Vu

What is  
MuJoCo?

Configuration

Examples

7-DOF serial  
manipulator

6-DOF  
parallel  
mechanisms

Unmanned  
Aerial Vehicle  
(UAV)

Autonomous  
Underwater  
Vehicle (AUV)

Q&A

Hope you slept comfortably!