# Robot Navigation and Obstacle Avoidance

Robotics Y5 – Graduate Course Project

By: Georgios Voudiotis & Nefeli E. Sextou

Academic Year 2023-2024

University of Ioannina – Dept. Of Computer Science & Engineering

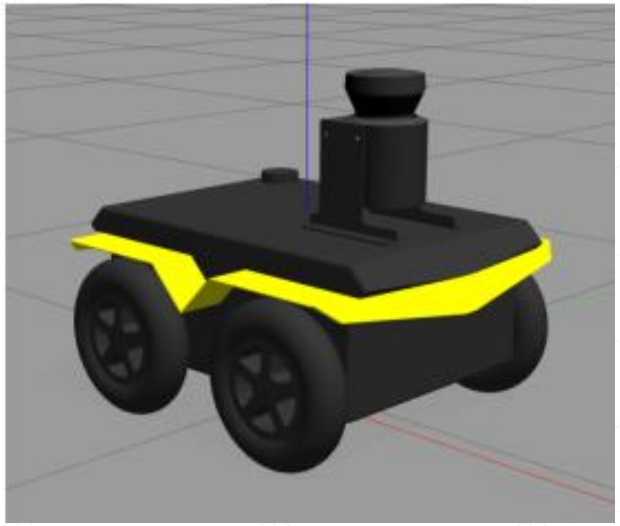
# Goal:

Program a robot’s movement in a simulation environment

Program the robot’s movement in a real environment

# The Robot:

- **Jackal** by Clearpath Robotics
- Mobile - Differential Drive
- Equipped with LiDAR



# Software:

- **ROS 1 Noetic** on Ubuntu 20.04 Focal Fossa
- **Gazebo**
- **rViz**

# Problem Description:

- Workspace: **6 x 4.5** m
- Set randomly chosen **Start** and **Goal** configurations within the workspace
- Set 3 **cubic** obstacles at random points within the workspace

Configuration Representation:

$$q = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \theta \end{bmatrix}$$

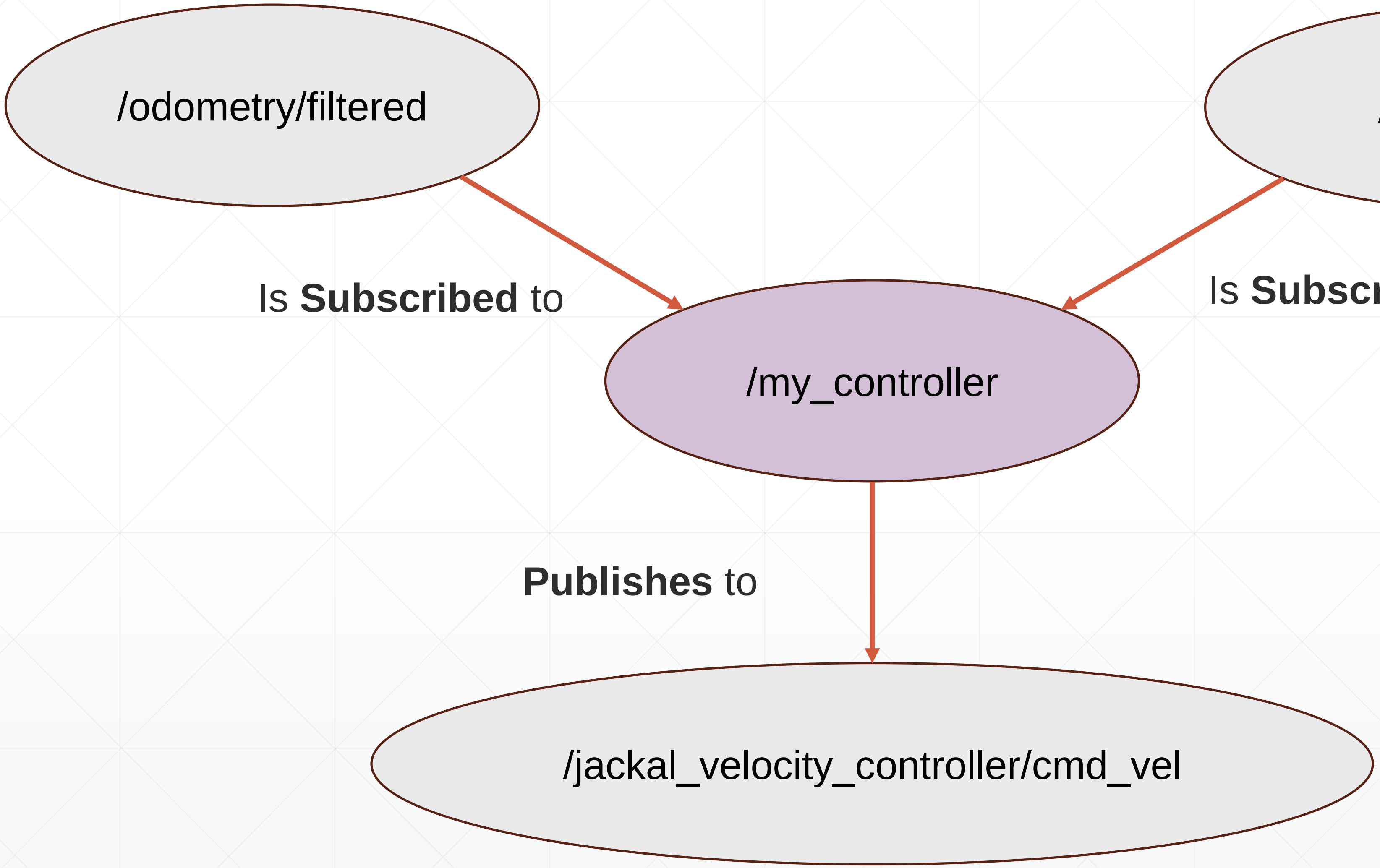
**position**

orientation

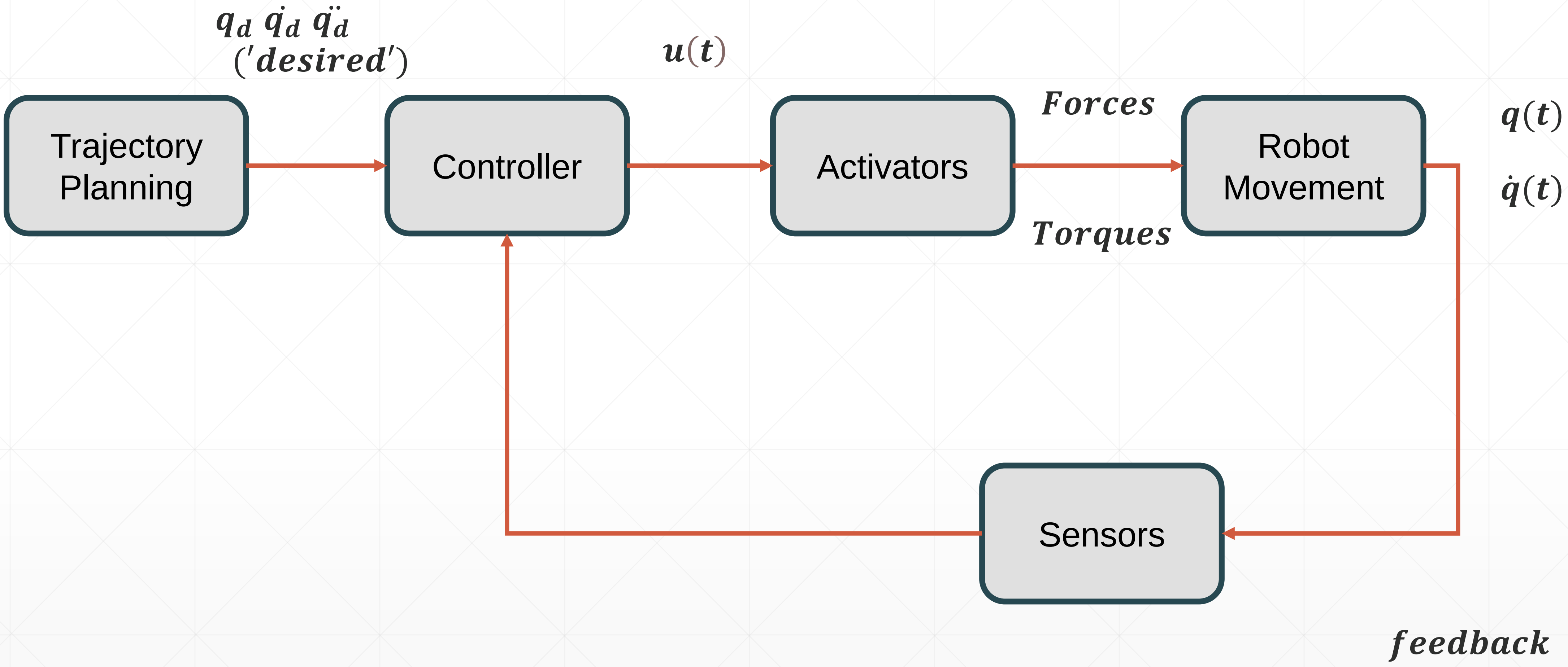
# Methodology:

- New topic/s for a Controller
- Artificial Potential Fields : Navigation Functions
- Program scripts
- Gazebo-rViz simulation and collect data
- Run on real robot and collect data
- Compare results

# Topics:


/odometry/filtered
/front/scan
Is **Subscribed** to
Is **Subscribed** to
/my_controller
**Publishes** to
/jackal_velocity_controller/cmd_vel

# Closed Loop Controller:



# Velocity Control:

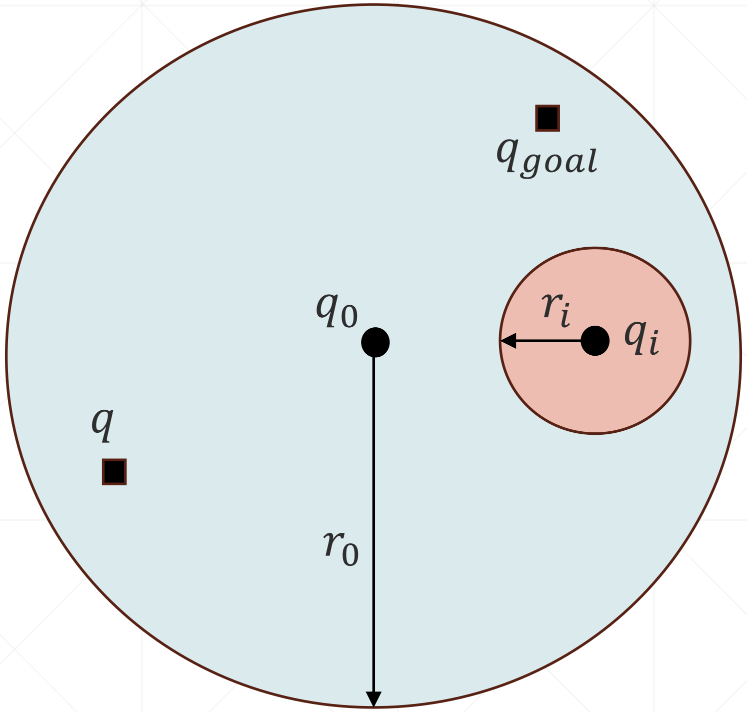
In the Navigation Function methodology, the velocity command is

$$\boxed{\dot{\boldsymbol{q}} = \boldsymbol{u} = -\mathbf{K_v}\, \boldsymbol{\nabla\phi}(\boldsymbol{q})}$$

Where, $\boldsymbol{\nabla\phi}(\boldsymbol{q})$ is the gradient of the Navigation Function
and $\mathbf{K_v}$ is a *gain* parameter

- The gradient has a magnitude and an orientation
- The robot is a mobile, therefore it only has a linear velocity on axis x and an angular velocity on axis z (yaw)
- The linear velocity must be proportional to the magnitude of the gradient
- The angular velocity must be adjusted according to the gradient’s orientation

# Navigation Functions:




Distance of obstacles

$$\beta_i(q) = \begin{cases} -d^2(q, q_0) + r_0^2 \\ d^2(q, q_i) - r_i^2 \end{cases}$$

Repulsive Potential

$$\beta(q) = \prod_{i=0}^{n} \beta_i(q)$$

Attractive Potential

$$\gamma_k(q) = \left(d(q, q_{goal})\right)^{2k}$$

Navigation Function (NF)

$$\phi = \frac{\gamma_k(q)}{\beta(q) + \gamma_k(q)}$$

$$\phi_{morse} = \frac{d(q, q_{goal})^2}{\left(d(q, q_{goal})^{2k} + \beta(q)\right)^{\frac{1}{k}}}$$

**For 2d vectors :** $d(x, y) = ||x - y|| = \sqrt{(x_2 - x_1)^2 + (x_2 - y_1)^2)}$

# Navigation Functions:

Gradient $\nabla\phi(q) = \gamma(q) = \dfrac{2d(q,q_{goal})\nabla d(q,q_{goal})\left(d(q,q_{goal})^{2k}+\beta(q)\right)^{\frac{1}{k}} - d(q,q_{goal})^{2}\nabla\left(d(q,q_{goal})^{2k}+\beta(q)\right)^{\frac{1}{k}}}{\left(d(q,q_{goal})^{2k}+\beta(q)\right)^{\frac{2}{k}}}$

Where

$$\nabla d(q,q_{goal}) = \frac{q-q_{goal}}{d(q,q_{goal})}$$

$$\nabla\left(d(q,q_{goal})^{2k}+\beta(q)\right)^{\frac{1}{k}} = \frac{1}{k}\left(d(q,q_{goal})^{2k}+\beta(q)\right)^{\frac{1}{k-1}}\left(2k\, d(q,q_{goal})^{2k-1}\nabla d(q,q_{goal}) + \nabla\beta(q)\right)$$

$\nabla\beta(q) = \sum_{i=0}^{n}\nabla\beta_i(q)\prod_{j=0,j\neq i}^{n}\beta_j(q)$ where $\nabla\beta_i(q) = \begin{cases} -2(q-q_0) \\ 2(q-q_i) \end{cases}$

# Programming NFs

- Partition LiDAR data considering objects every n samples -> m obstacles
- Keep min reading from each of the m obstacles
- Combine with angle increment information to find the distances between the robot and the obstacles ( $\beta(q)$ )
- Obstacle buffer zone = Robot Length + Virtual Obstacle Radius = 0.2+0.5
- Linear Velocity:
  - If the obstacle is within the robot's "safe zone":

$$linear.x = K_v * ||\nabla\phi(q)|| \qquad angular.z = K_a * \left(\arctan\left(\frac{-\frac{d\nabla\phi(q)}{dq_y}}{-\frac{d\nabla\phi(q)}{dq_x}}\right) - yaw\right)$$

  - If there are no objects in the "safe zone":

$$linear.x = K_v * ||\nabla\gamma_k(q)|| \qquad angular.z = K_a * \left(\arctan\left(\frac{-\frac{d\nabla\gamma_k(q)}{dq_y}}{-\frac{d\nabla\gamma_k(q)}{dq_x}}\right) - yaw\right)$$

$K_v\,, K_a :\ gains\ (> 0)$

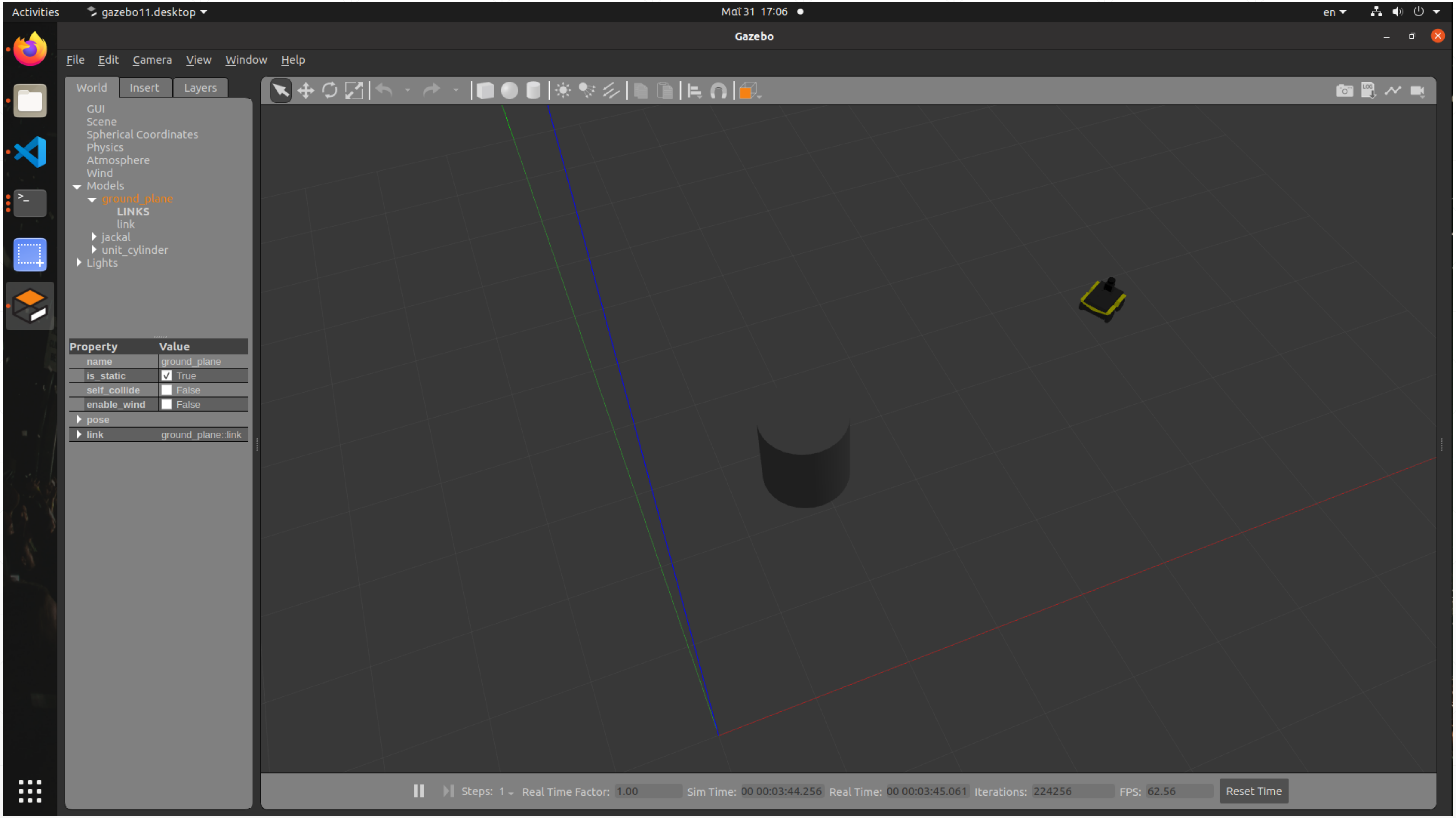
Attractive Potential Only

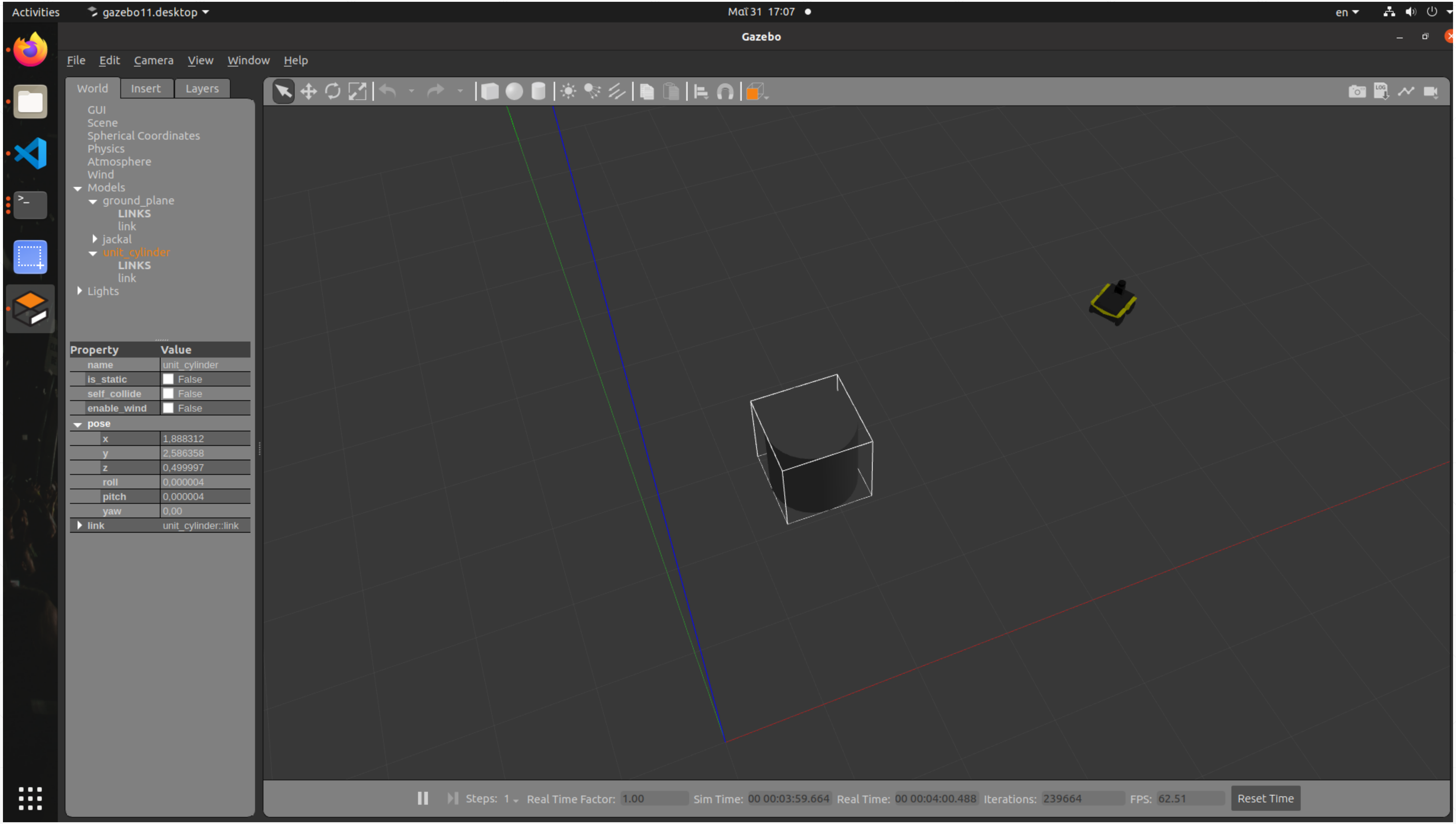
# Simulation Scenario

- **We place a cylindrical obstacle at random (could be any shape)**
- **We set the goal configuration as:**

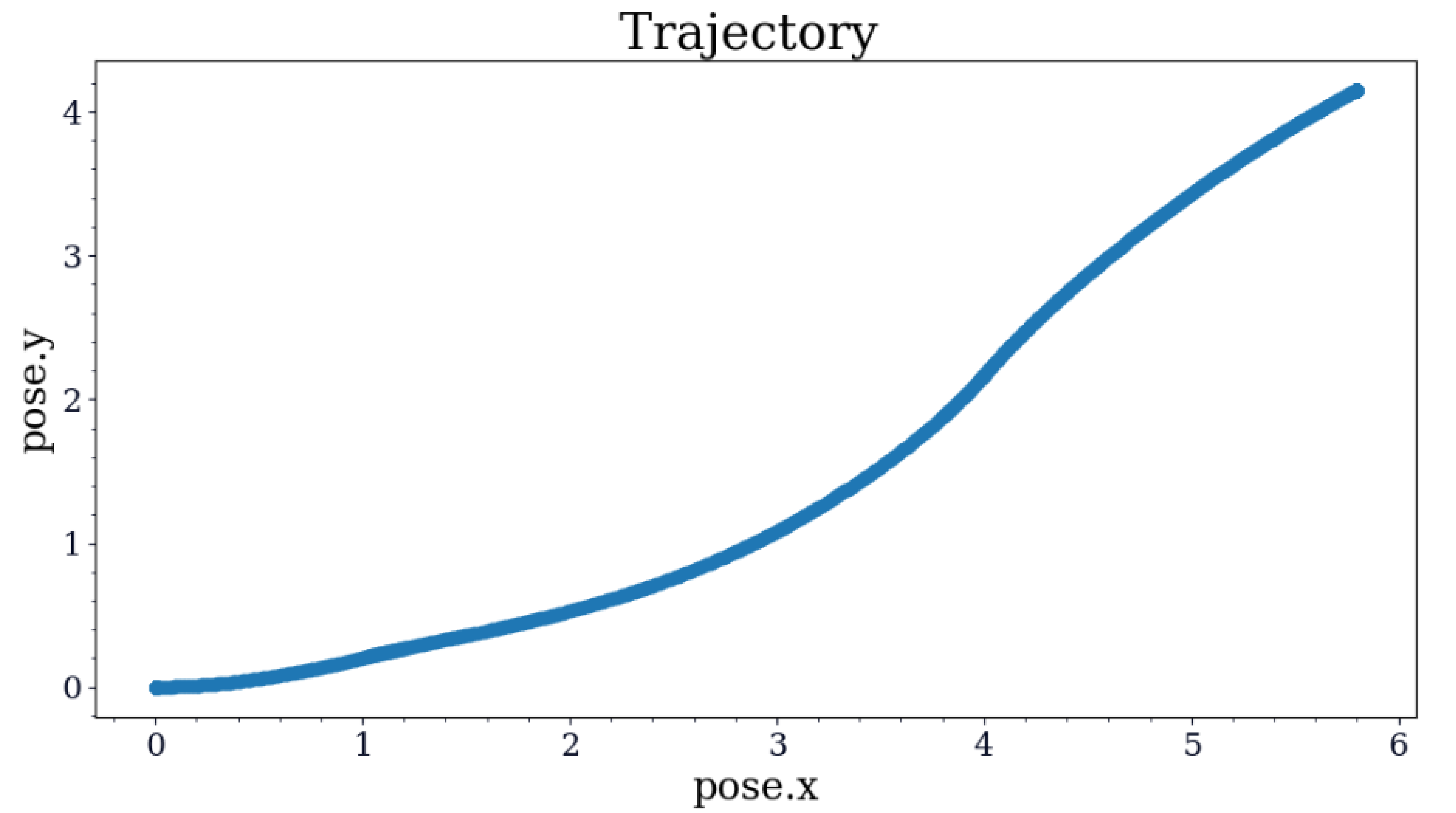
$$\mathbf{q}_{goal} = \begin{bmatrix} 6.5 \\ 4.0 \\ 0.0 \end{bmatrix}$$

- $K_v , K_a = 0.5$
- $\mathrm{k} = 1$

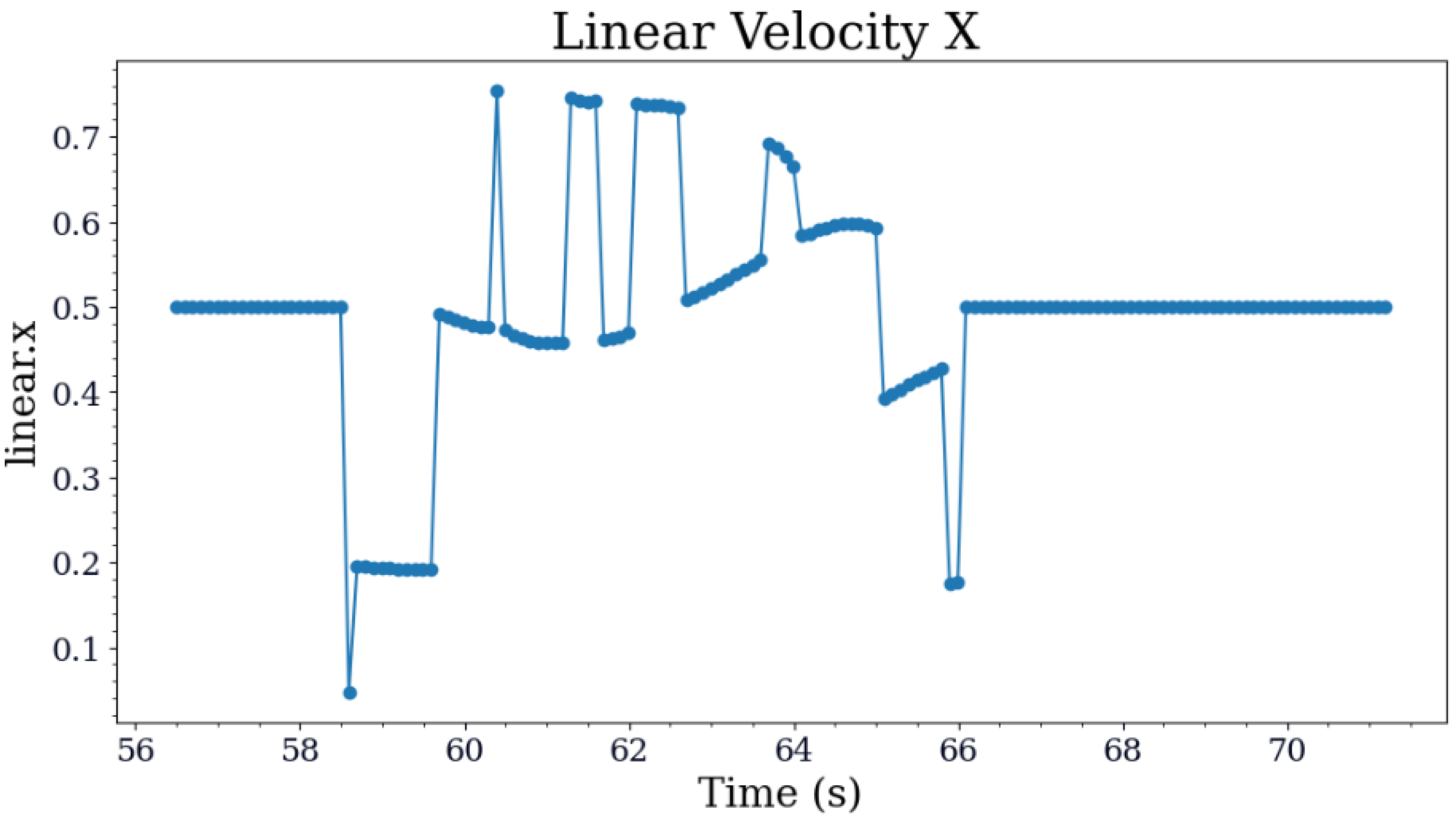

Activities
gazebo11.desktop
Маї 31 17:06
en
Gazebo
File
Edit
Camera
View
Window
Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
ground_plane
LINKS
link
jackal
unit_cylinder
Lights
Property
Value
name
ground_plane
is_static
True
self_collide
False
enable_wind
False
pose
link
ground_plane::link
Steps: 1
Real Time Factor: 1.00
Sim Time: 00 00:03:44.256
Real Time: 00 00:03:45.061
Iterations: 224256
FPS: 62.56
Reset Time


Activities
gazebo11.desktop
Маї 31 17:07
en
Gazebo
File
Edit
Camera
View
Window
Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
ground_plane
LINKS
link
jackal
unit_cylinder
LINKS
link
Lights
Property
Value
name
unit_cylinder
is_static
False
self_collide
False
enable_wind
False
pose
x
1,888312
y
2,586358
z
0,499997
roll
0,000004
pitch
0,000004
yaw
0,00
link
unit_cylinder::link
Steps: 1
Real Time Factor: 1.00
Sim Time: 00 00:03:59.664
Real Time: 00 00:04:00.488
Iterations: 239664
FPS: 62.51
Reset Time

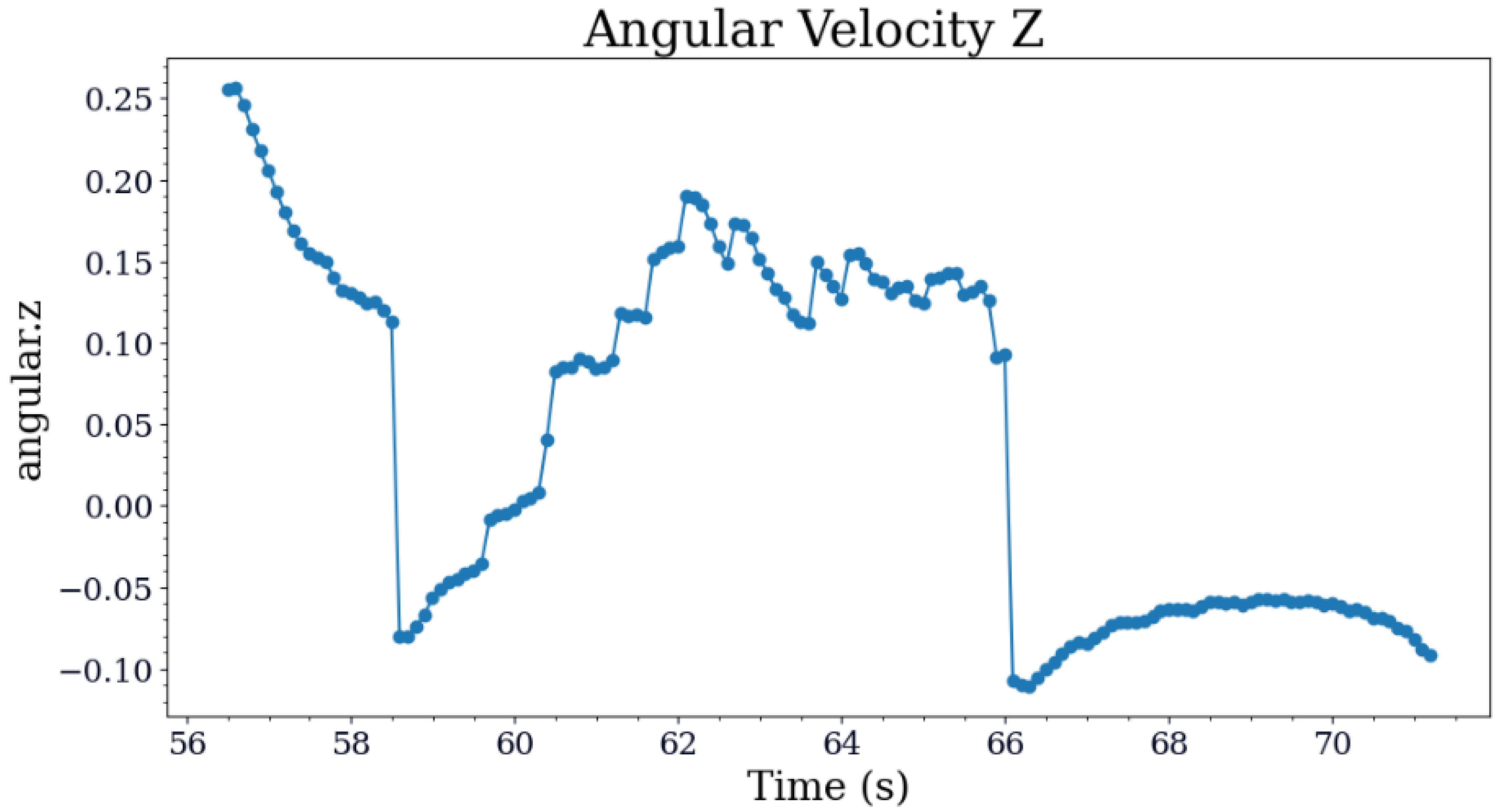
# Simulation: Odometry Results



# Simulation: Velocity Results



# Simulation: Velocity Results



# Experiments in a Real Environment

- **Experiment 1 (Simulation to real world):**
  - **Goal Configuration: x = 6.5 y = 4 θ = 0,** $K_v, K_a = 0.5, \mathrm{k} = 1$
  - **It was stopped before attempting to go through the wall but the trajectory it was on was correct**
  - **Link to video: https://drive.google.com/file/d/1-aFstjbn_DAgrovlkNKFjyuxKuny7vDY/view?usp=drive_link**
- **Experiment 2 (Goal within available lab space):**
  - **Goal Configuration: x = 4 y = 4 θ = 0,** $K_v, K_a = 0.5, \mathrm{k} = 1$
  - **Successfully reaches the goal and stops**
  - **Link to video: https://drive.google.com/file/d/1rq3qUtUcCK7O-VWkYEEWPHHpCpDoiybM/view?usp=drive_link**

Note: Google Chrome works best with the links

# Conclusion

- The simulation and real results matched
- The robot behaves as we expect it to based on the simulation
- The robot both avoids the obstacle and reaches the goal

# Sources

Course Materials

For the Gradients:

- https://www.sciencedirect.com/science/article/pii/019688589090017S
- https://ecourse.uoi.gr/pluginfile.php/395920/mod_resource/content/17/robotics_project_grads_2024_B.pdf

# Thank You!