Testing Physics Engines with Live Robot Programming

Johan Fabry (DCC UChile)
Stephen Sinclair (Inria Chile)

3 Things

1. Physics Engines

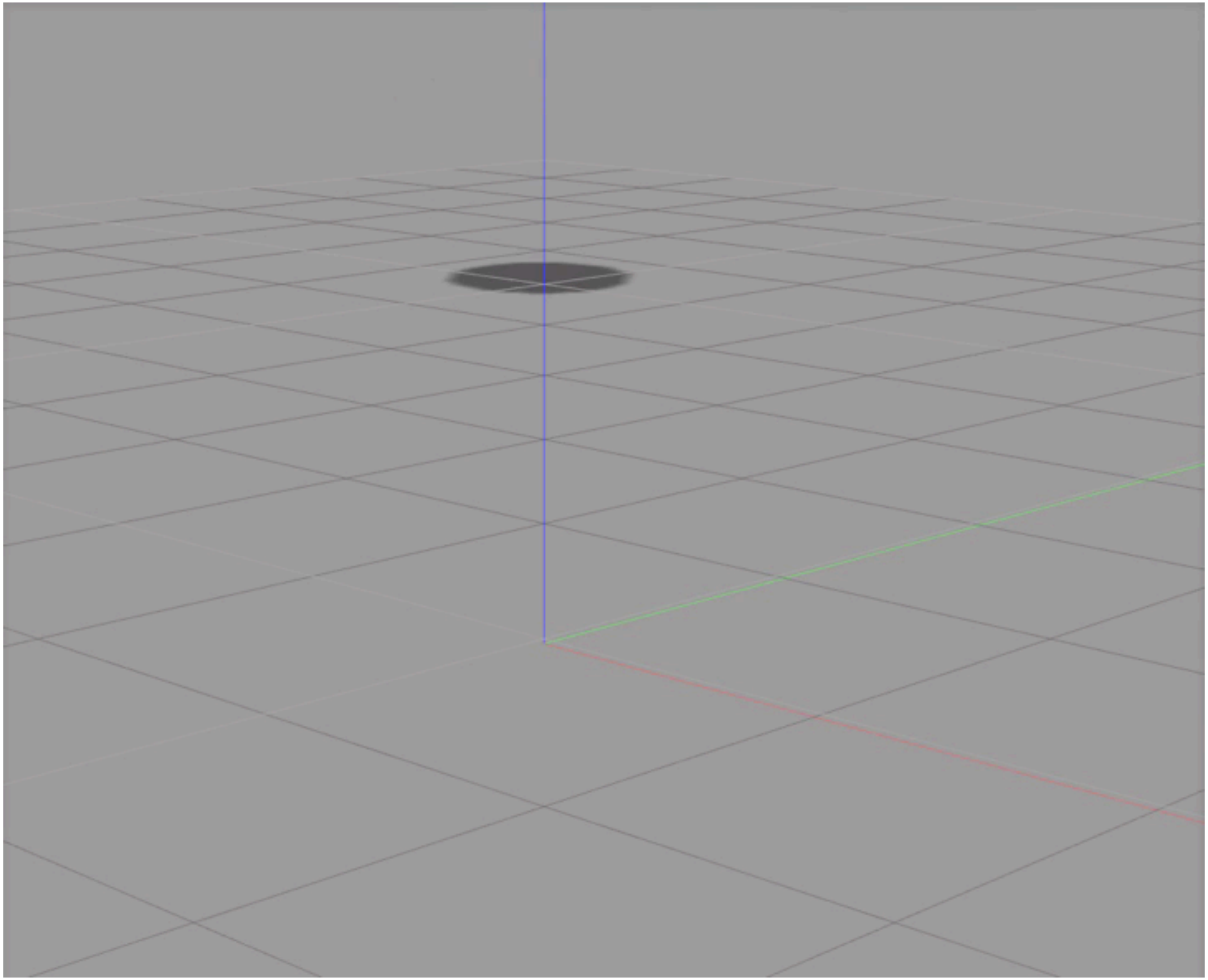
2. LRP

3. Unit tests

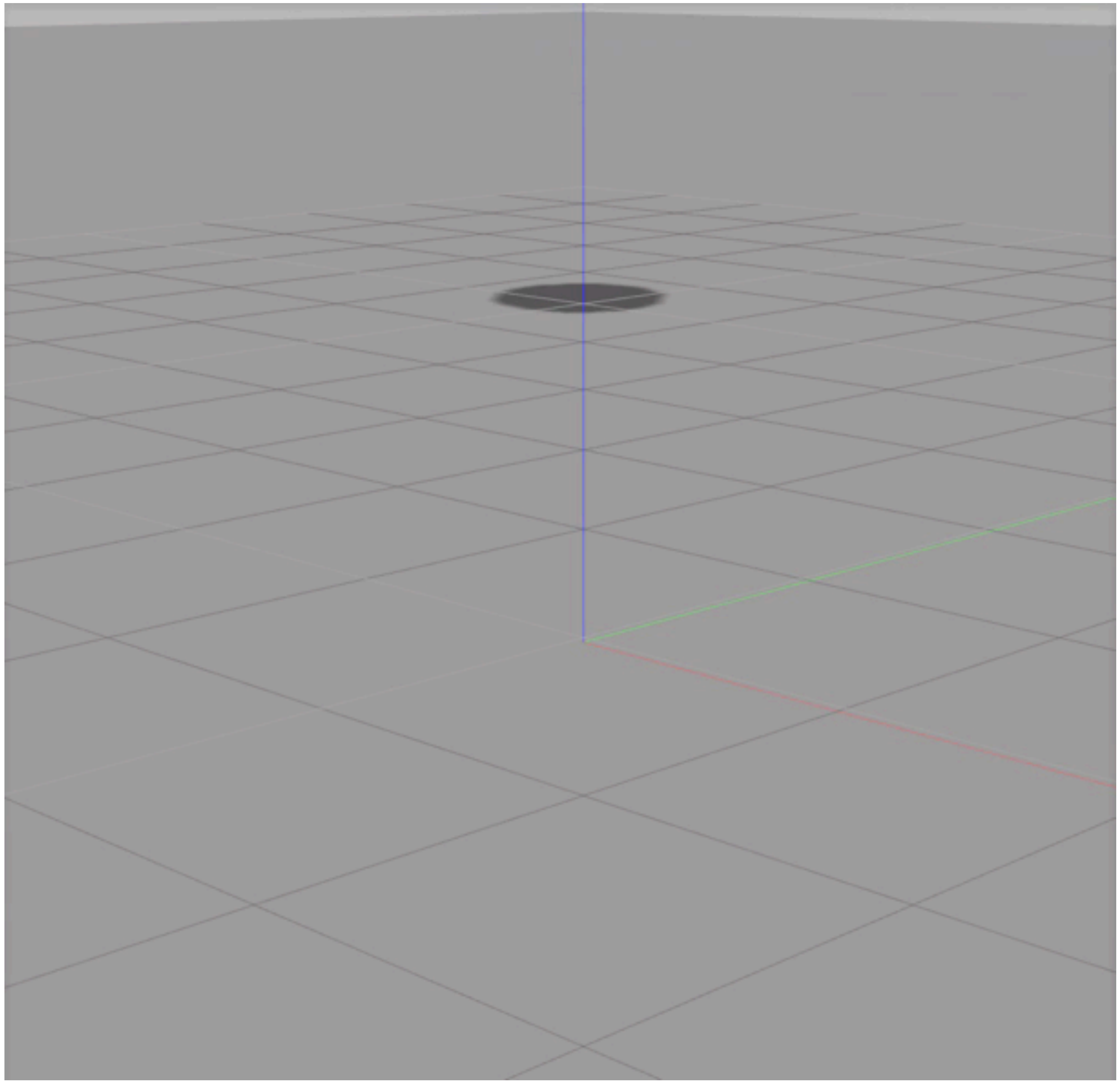
Physics Engines

What happens in Gazebo when you drop a ball?

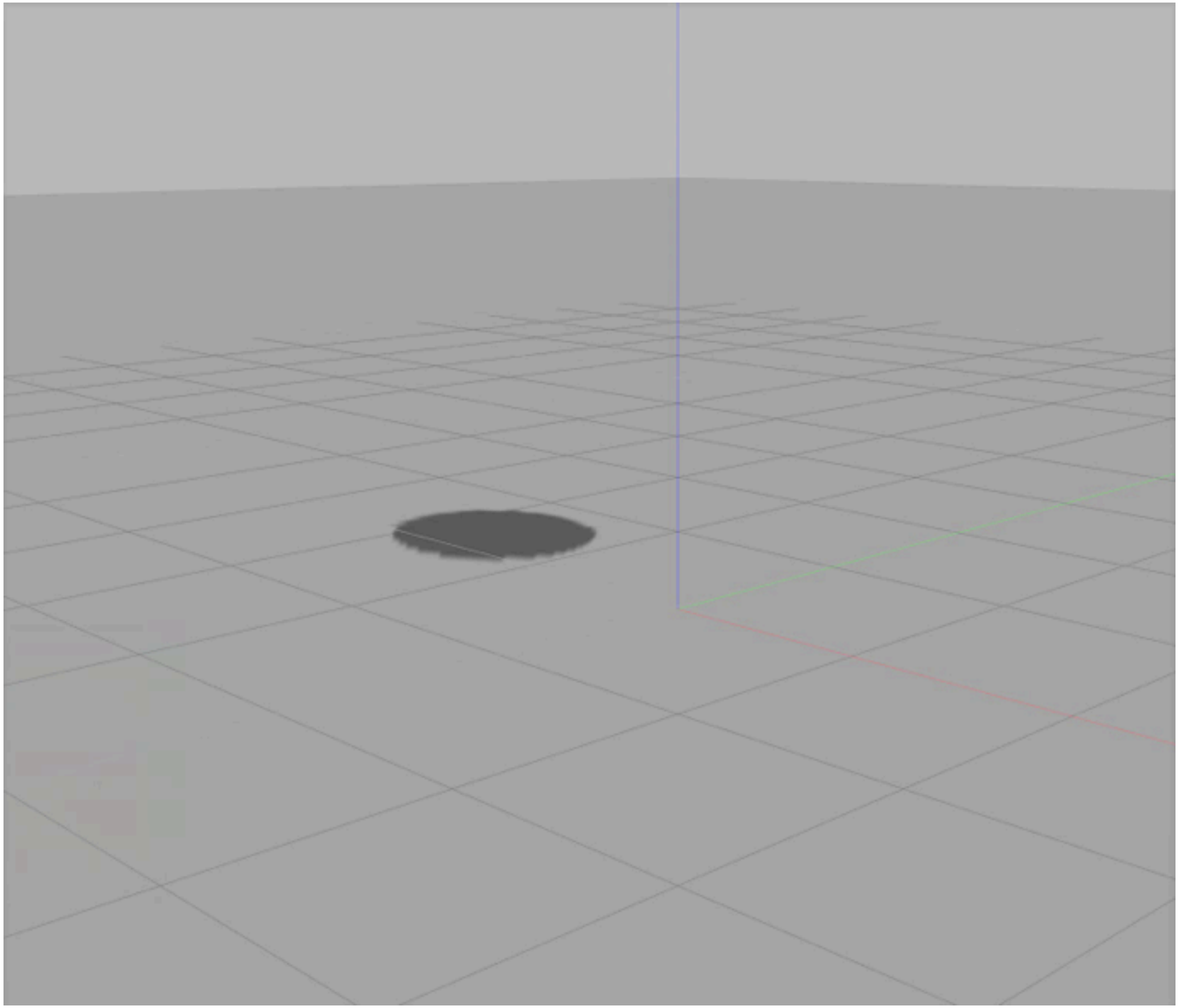
Gazebo - Bullet



Gazebo - ODE



Gazebo - Siconos



Phys engines don’t always match real world behavior

How do you test this?

- Don’t care about Root Main Square
- Test real-world situations!
- Different engines, different parameters
- So easy that a roboticist can write them

3 Things

1. Physics Engines need real world tests

2. LRP

3. Unit tests

Live Robot Programming

Software Engineering For Robotics

Robotics =
Problem complexity +
Technology complexity

Waste less time in
incidental complexity

Use time on
fundamental complexity

Software Example

“But **why** is the robot executing this behavior now?”

(What is the internal state of the algorithm)

“**What would happen** if I change epsilon to 5 ?”

(What are the correct parameters for the algorithm)

Spend brainpower on the complexity of the task

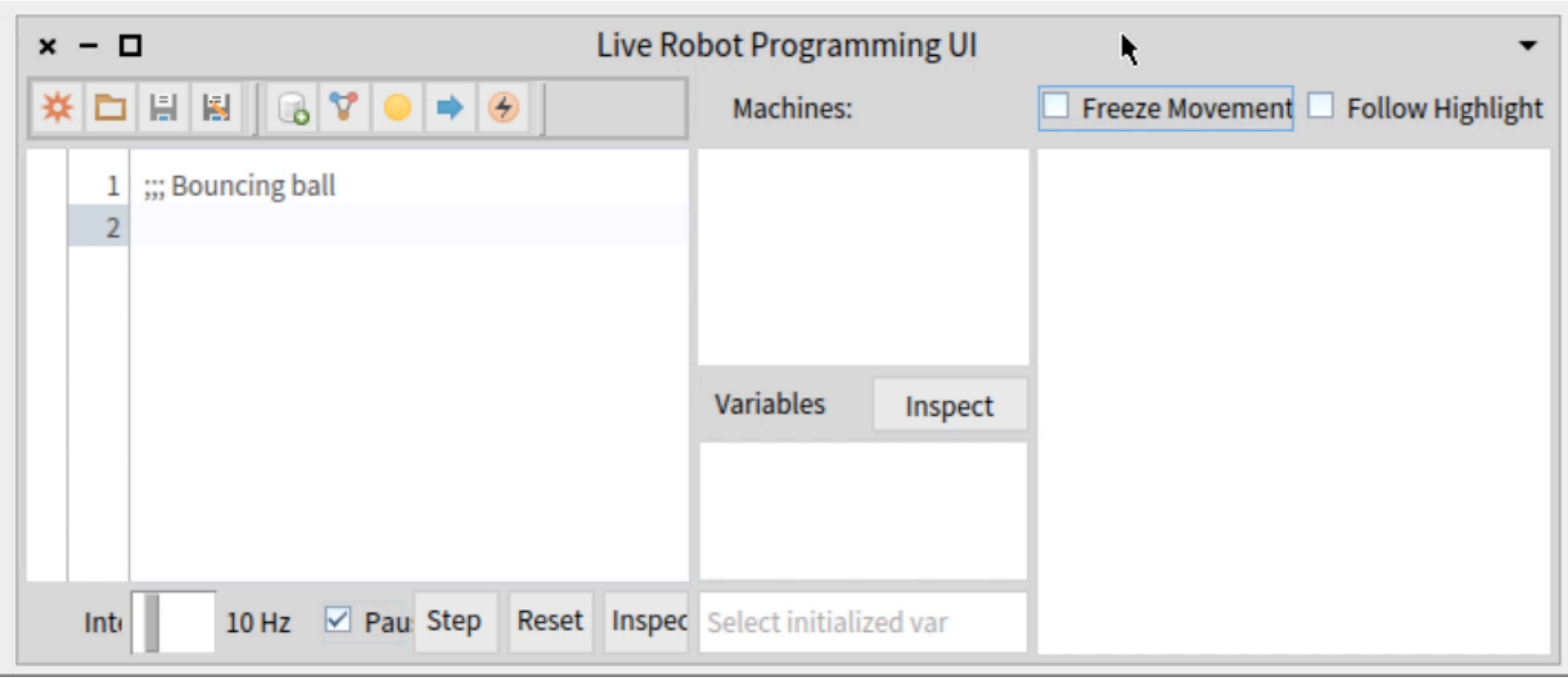
Have an immediate connection to the behavior

Live Robot Programming

- Nested State machines
- Direct manipulation of **running** code
- Direct manipulation of vars **in memory**
- Direct manipulation of the machine
- (Not linked to specific API/middleware)

Immediate Connection

A bouncing ball machine



Live Robot Programming

- Direct manipulation of **running** code
- Direct manipulation of vars **in memory**
- Direct manipulation of the machine

Immediate Connection

Software Example

“But **why** is the robot executing this behavior now?”

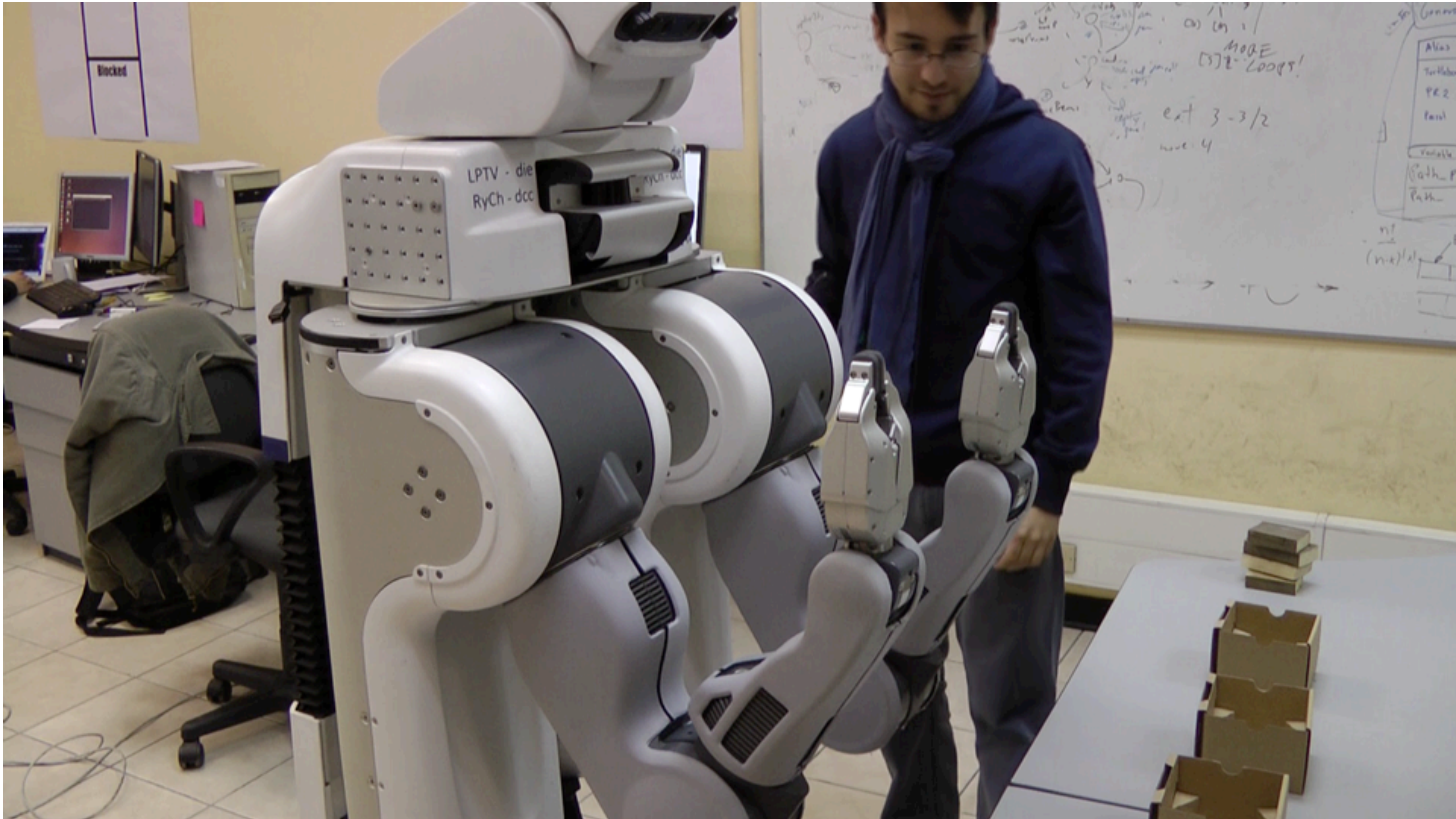
(What is the internal state of the algorithm)

“**What would happen** if I change epsilon to 5 ?”

(What are the correct parameters for the algorithm)

Works with ROS (+others)

Great for orchestration



Works with other APIs

Pleiad RyCh

More about the language ~~in the paper~~ on the website.

http://pleiad.cl/LRP

3 Things

1. Physics Engines need real world tests

2. Live Robot Programming

3. Unit tests

Live Tests for Robotics

How do you test this?

- Don’t care about Root Main Square
- Test real-world situations!
- Different engines, different parameters
- So easy that a roboticist can write them

Design principles

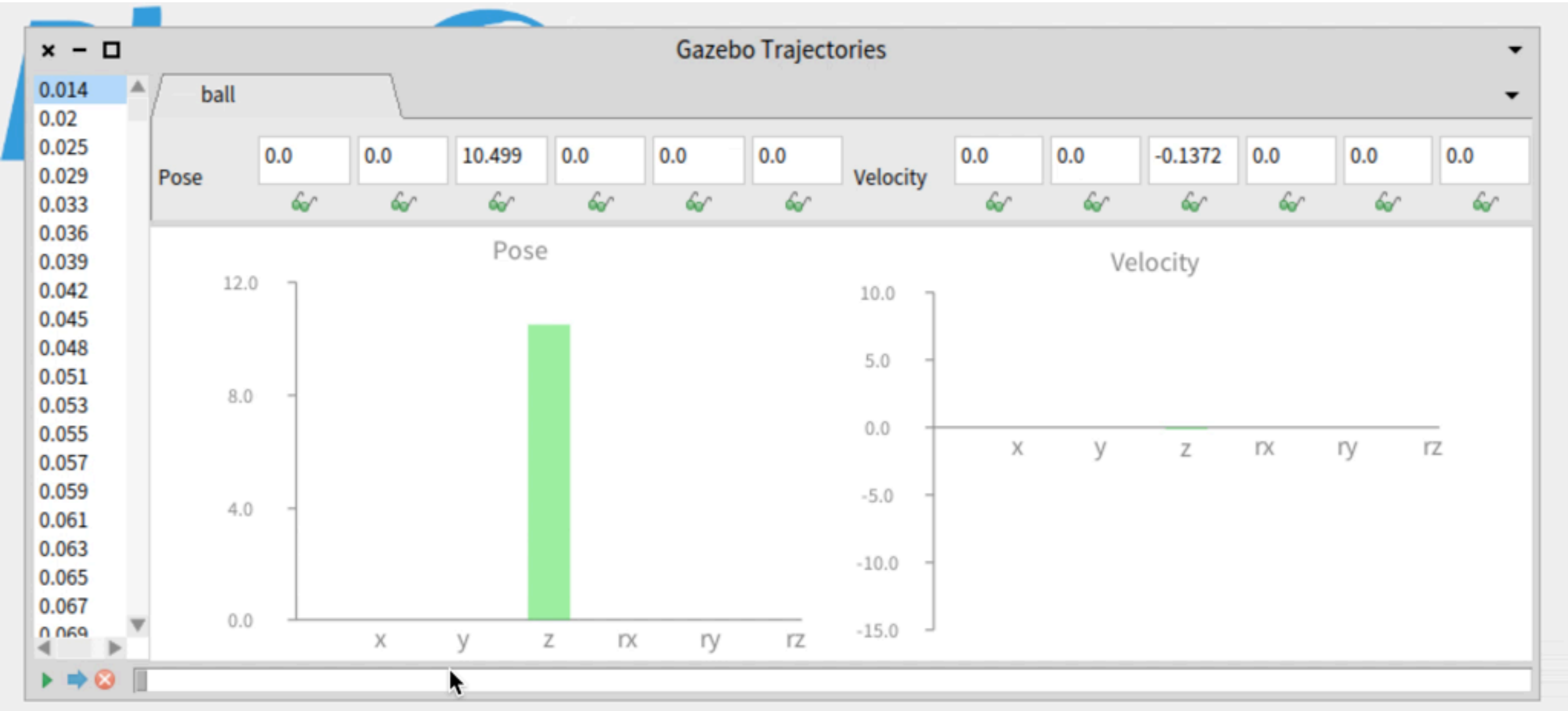
- Use trajectory logs of Gazebo
- Machine = State of the world
- Code accesses 1 snapshot of time
- Object trajectory = global variable
- High interactivity with the user

API Examples

(It’s a DSL)

- `ball pose z`
- `ball velocity z > 0`
- `(ball pose ~ 0.02) z > top`
- `ball time >= stopTime`

User interface !

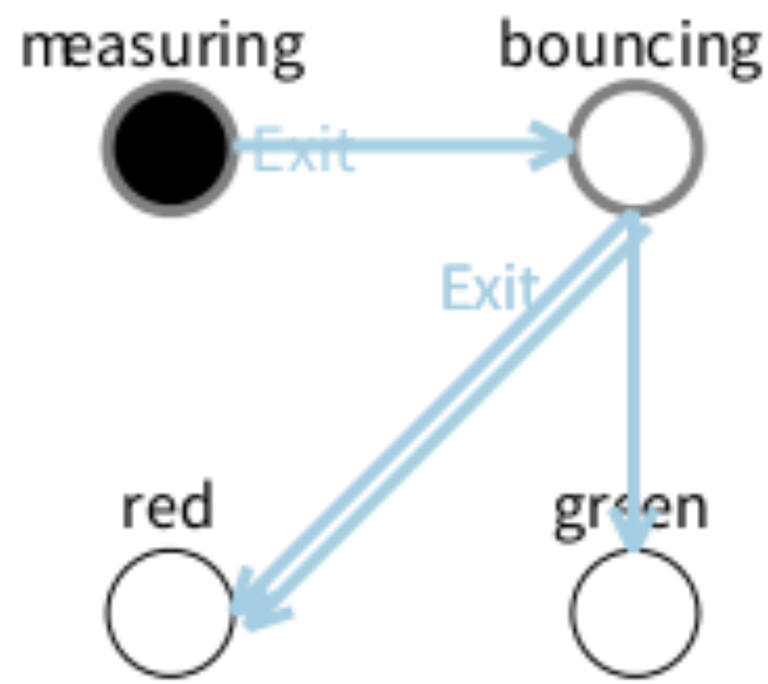


Immediate Connection

Bouncing ball test:

```
(var count :=[0])
(machine bounceCounter
    (state falling )
    (state rising (onentry [count := count + 1]))
    (on [ball velocity z > 0] falling -> rising)
    (on [ball velocity z < 0 ] rising -> falling)
    (event endWell [ ball time = stopTime and: [count = 3]])
    (state green)
    (on endWell falling->green)
    (on endWell rising->green)
    )
(spawn bounceCounter falling )
```

Sensible bounce



More example tests in the paper

Objectives achieved!

- Don’t care about Root Main Square
- Test real-world situations!
- Different engines, different parameters
- So easy that a roboticist can write them

3 Things

1. Physics Engines need real world tests

2. Live Robot Programming

3. LT4R: Unit tests for nr 1 in nr 2

http://pleiad.cl/LRP