Prototyping of a Robot Arm Controller: getting the hands dirty to learn new technologies

Enrique Llerena Dominguez

mimacom

Eclipsecon Europe 2019

$whoami

- Enrique Llerena Dominguez
- Software Engineer / Consultant @ mimacom
- Experience developing software for the finance, retail, pharma, and automotive industry

- Professional Interests: Software Architecture, Cloud Computing, Artificial Intelligence

- Free Time: Spending time with my kids, watching football, learning german, and developing nice things

MIMACOM

Motivation

Learning!

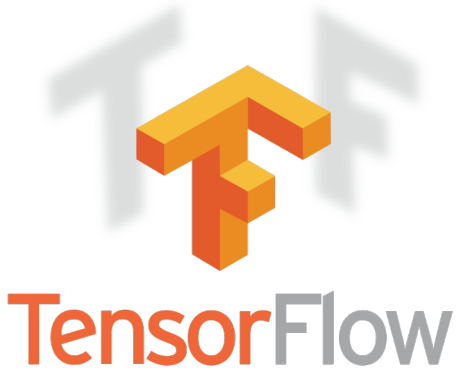


Image Sources: cited in slide „Image Sources“

Motivation

Learning!

APACHE kafka®
A distributed streaming platform

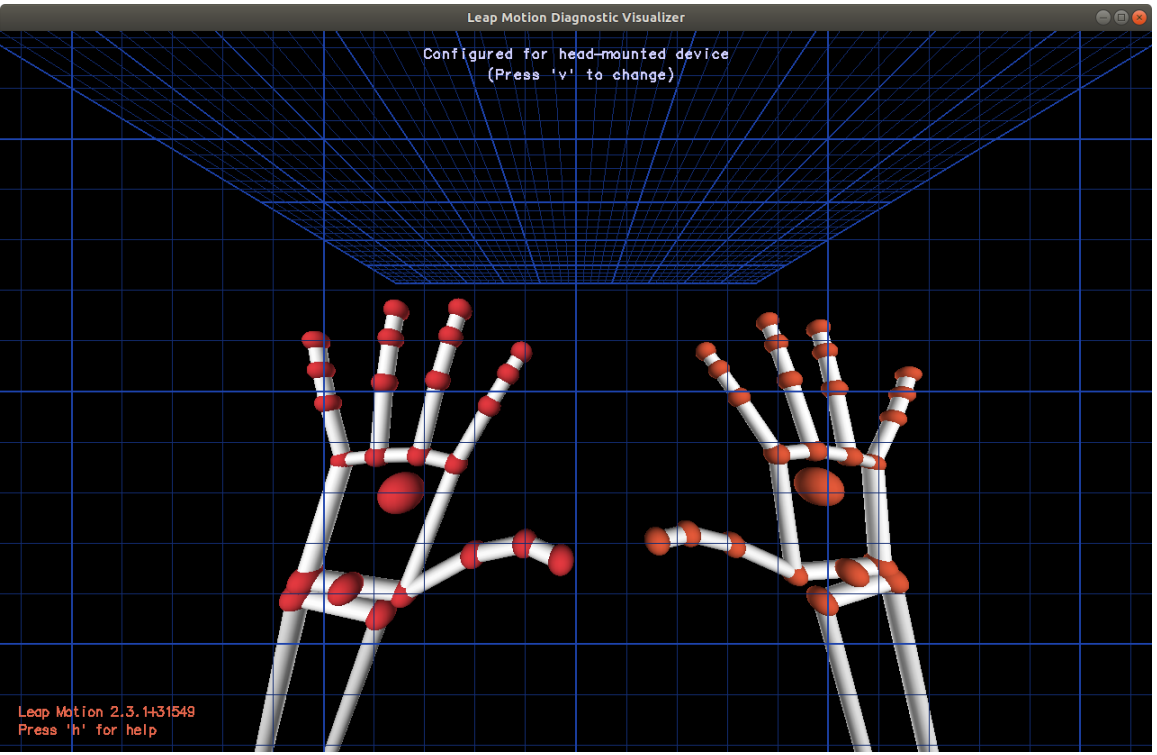
GAZEBO

Image Sources: cited in slide „Image Sources“

Source: https://cdn.pixabay.com/photo/2015/10/30/12/24/questions-1014060_960_720.jpg

Image source: Photo by Alex Jones on Unsplash

What?



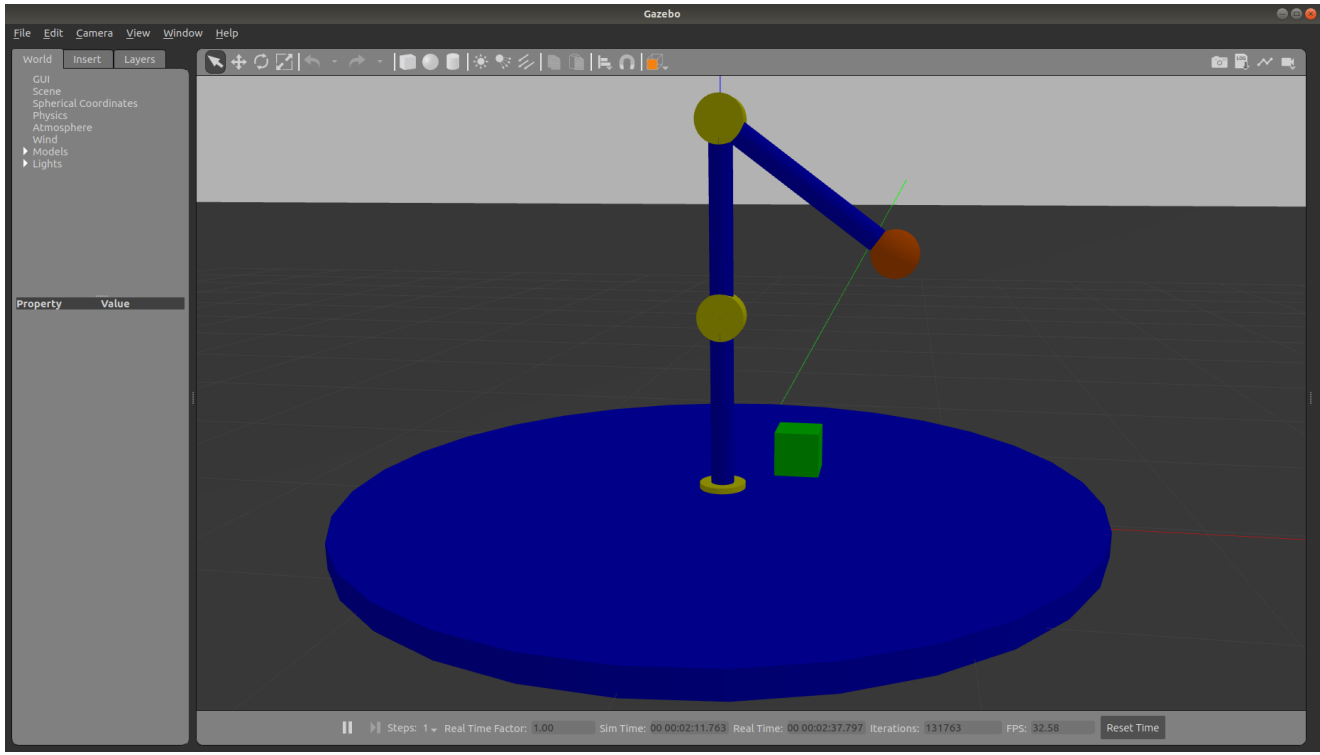


Image Sources: cited in slide „Image Sources“

Why is this a match?

- It is a data stream.

- This is a classification problem.

Image Sources: cited in slide „Image Sources“

Objectives

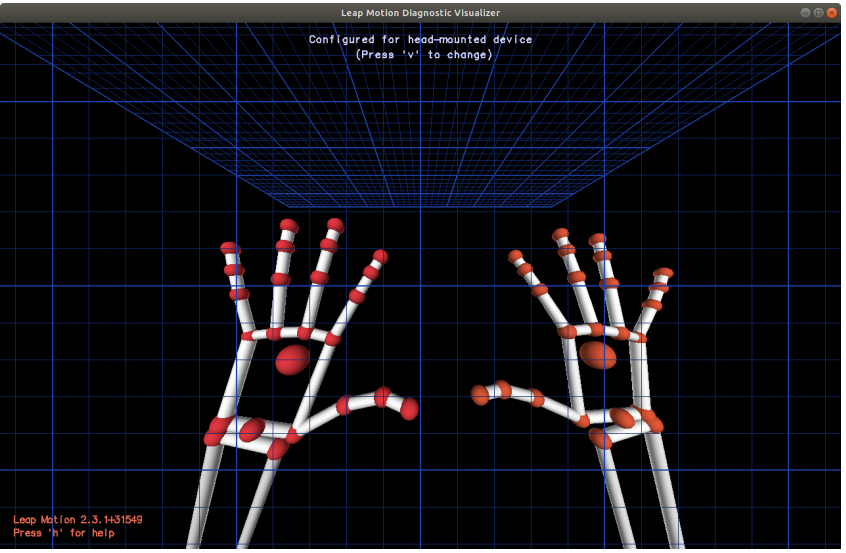
- Control the simulated robotic arm with the hands over the sensor.
  - How? Map a hand position to a movement instruction

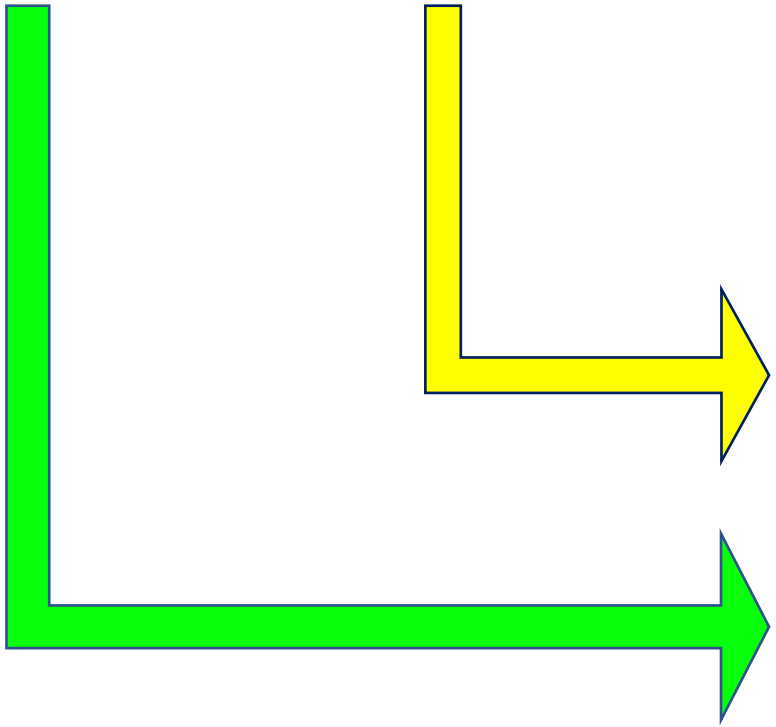
- Train a model so that everyone could define an individualized mapping

Image source: Photo by David Fanuel on Unsplash

What does „Map a hand position to a movement instruction“ mean?

Instruction = servoId, delta





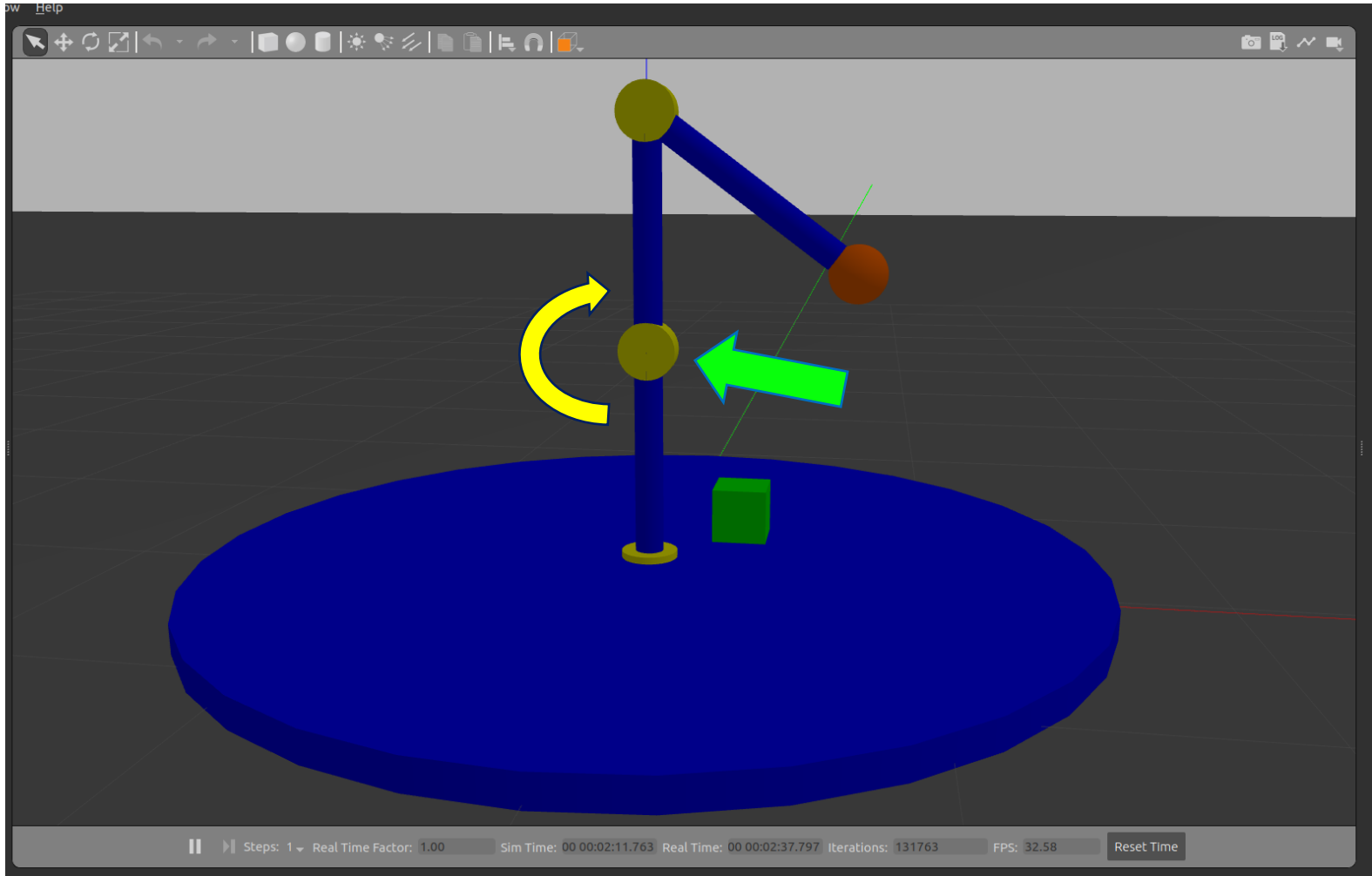
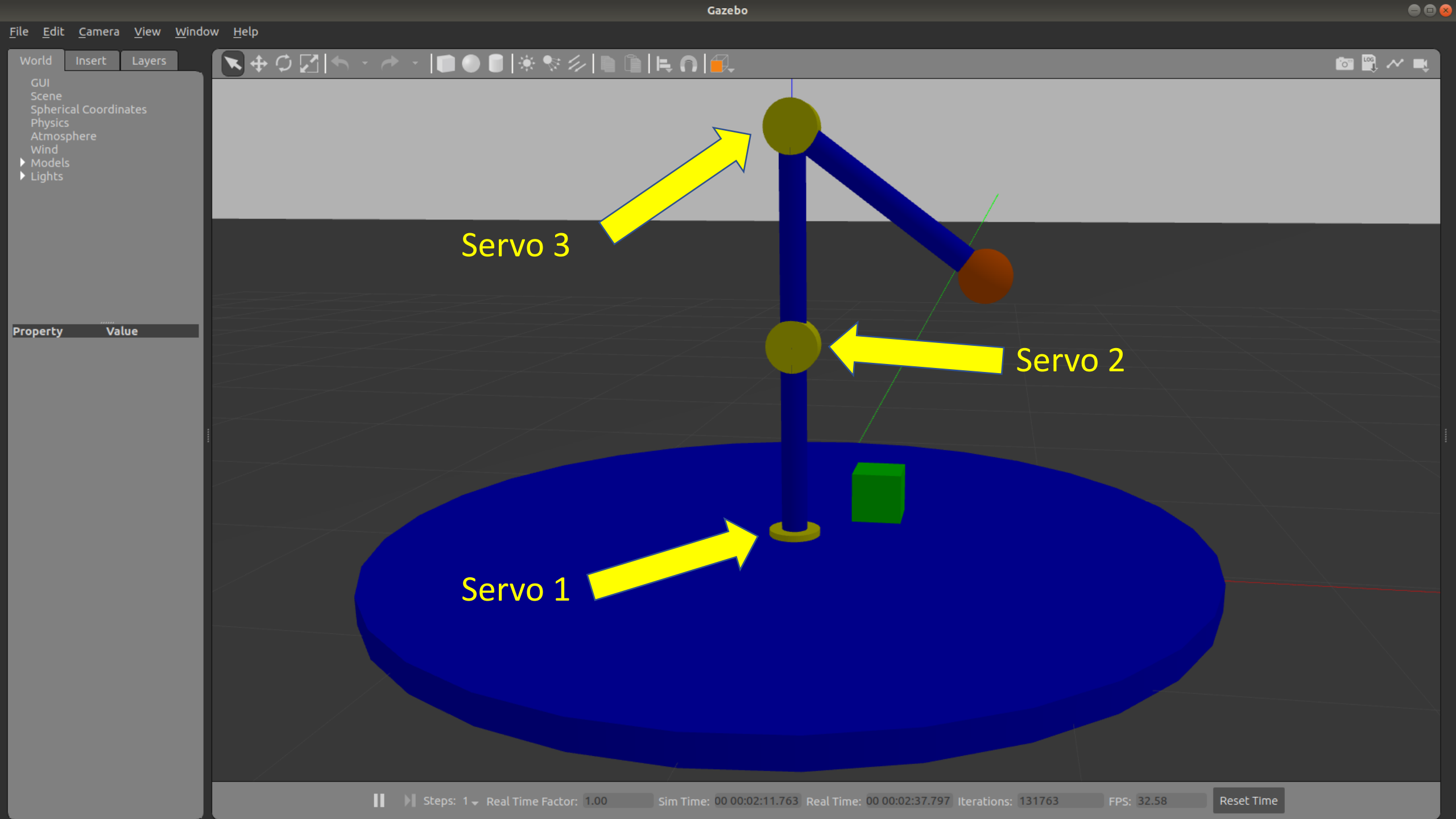
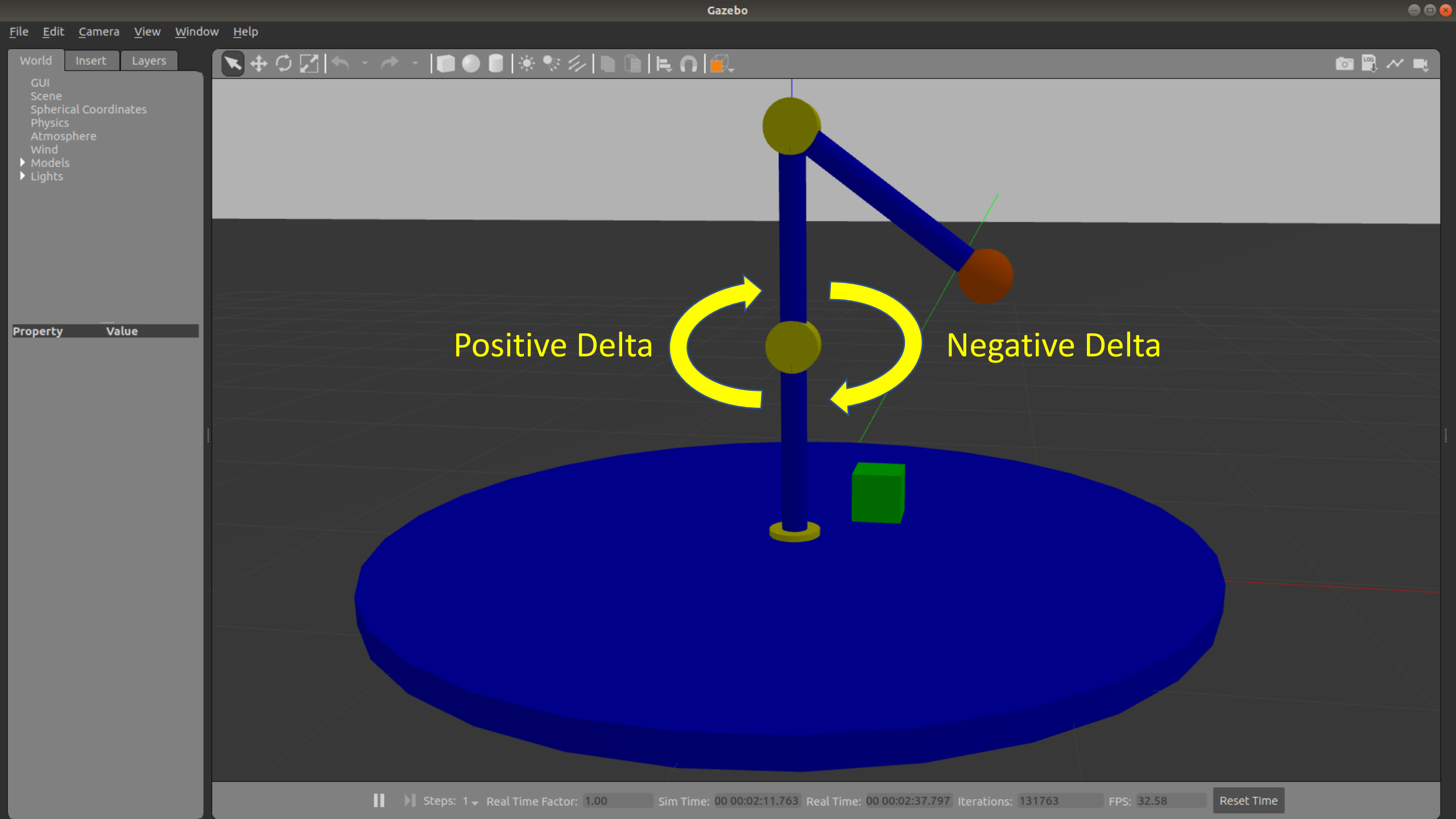


Image Sources: cited in slide „Image Sources“

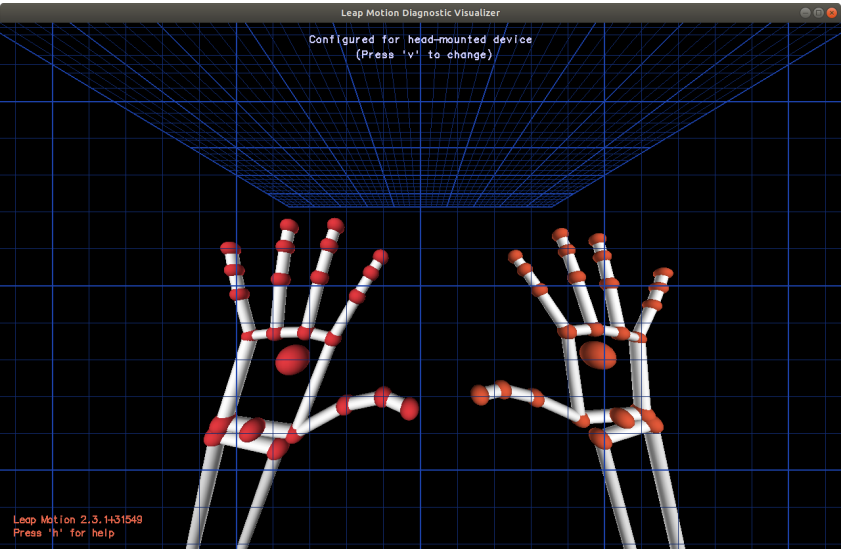

Gazebo
File
Edit
Camera
View
Window
Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property
Value
Servo 3
Servo 2
Servo 1
Steps: 1
Real Time Factor: 1.00
Sim Time: 00 00:02:11.763
Real Time: 00 00:02:37.797
Iterations: 131763
FPS: 32.58
Reset Time

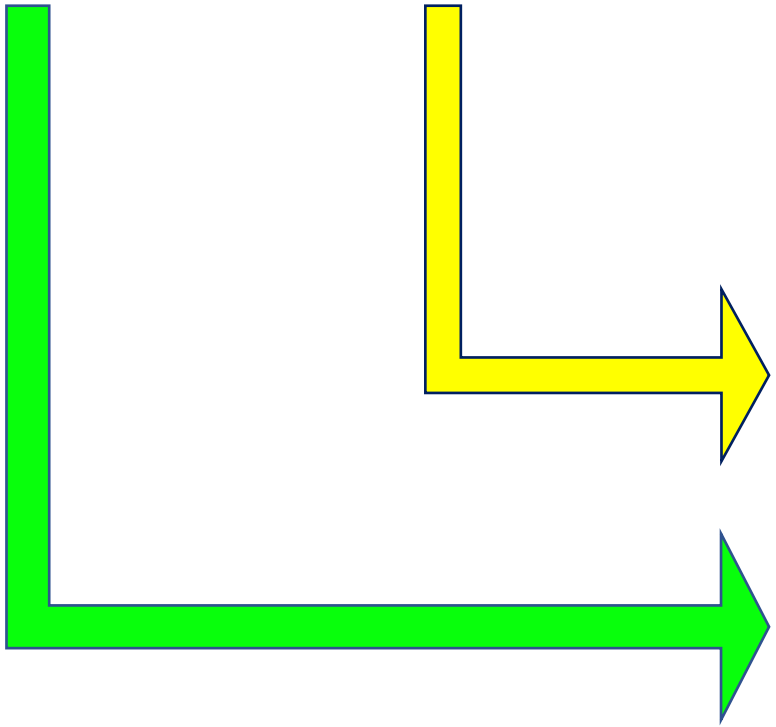

Gazebo
File
Edit
Camera
View
Window
Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property
Value
Positive Delta
Negative Delta
Steps: 1
Real Time Factor: 1.00
Sim Time: 00 00:02:11.763
Real Time: 00 00:02:37.797
Iterations: 131763
FPS: 32.58
Reset Time

What does „Map a hand position to a movement instruction“ mean?

Instruction = servoId, delta

Left Hand wide open = servo 2
Right Hand wide open = positive delta





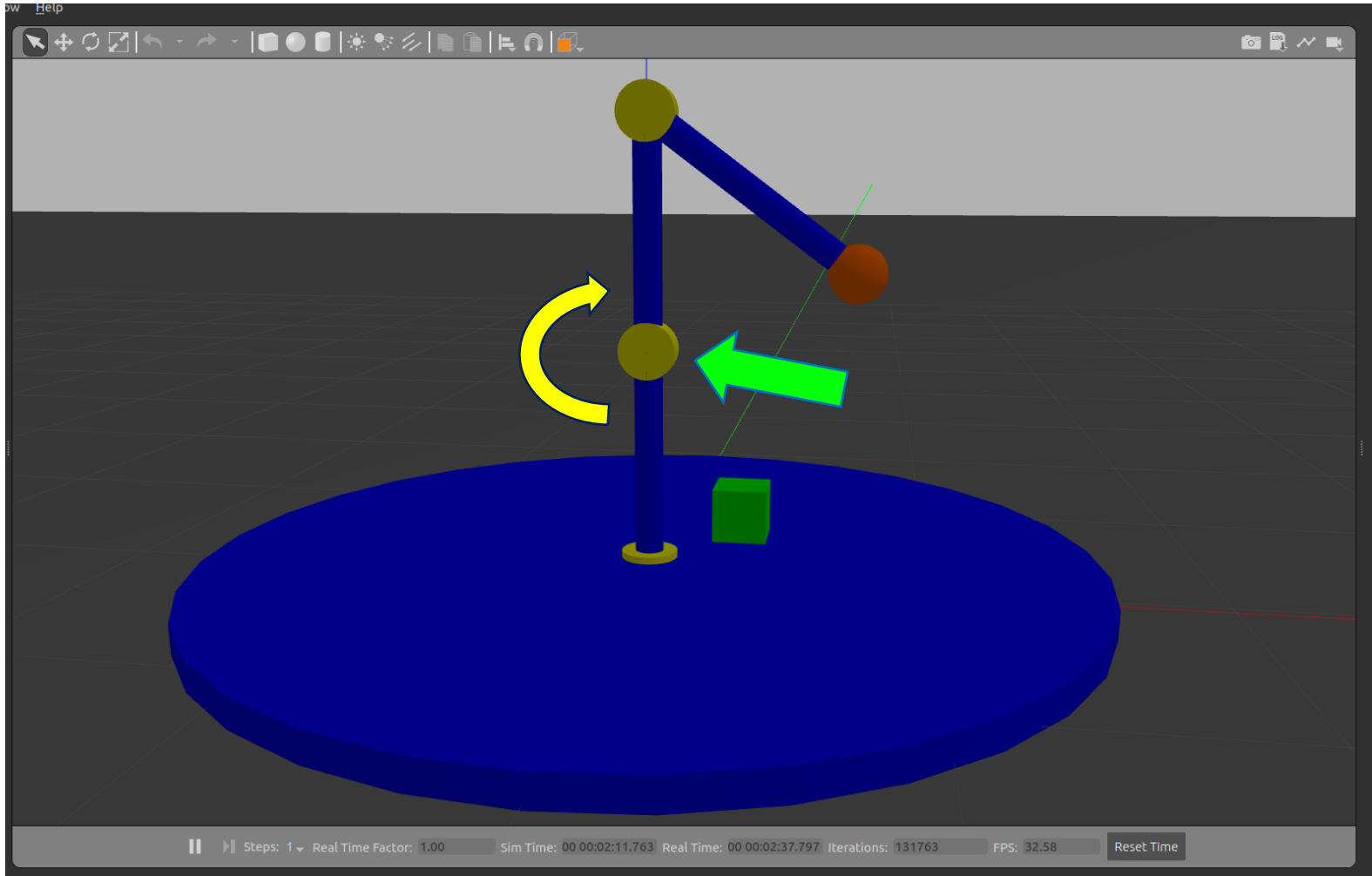


Image Sources: cited in slide „Image Sources“

Strategy

- Minimum Viable Product
- Small iterations to create small pieces with clearly testable deltas
- Tackle the riskiest things first

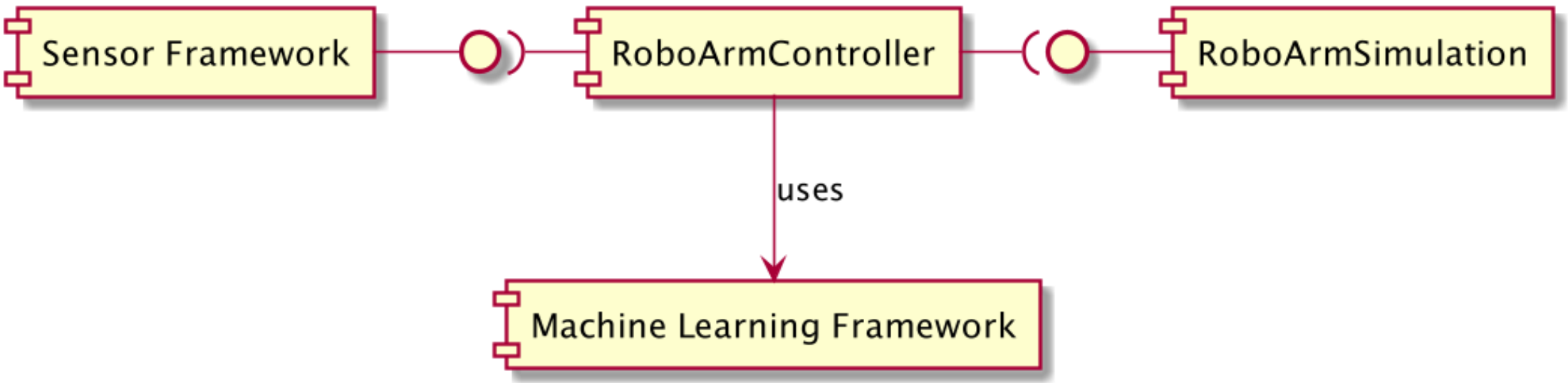
This Photo by Unknown Author is licensed under CC BY-SA-NC . Source: http://www.sportsbabel.net/diagram.jpg

Challenges

- Get data from the sensor
- Model the simulated robotic arm, control it, and communicate it with the exterior
- Transform sensor data into movements: Process data and communicate both components

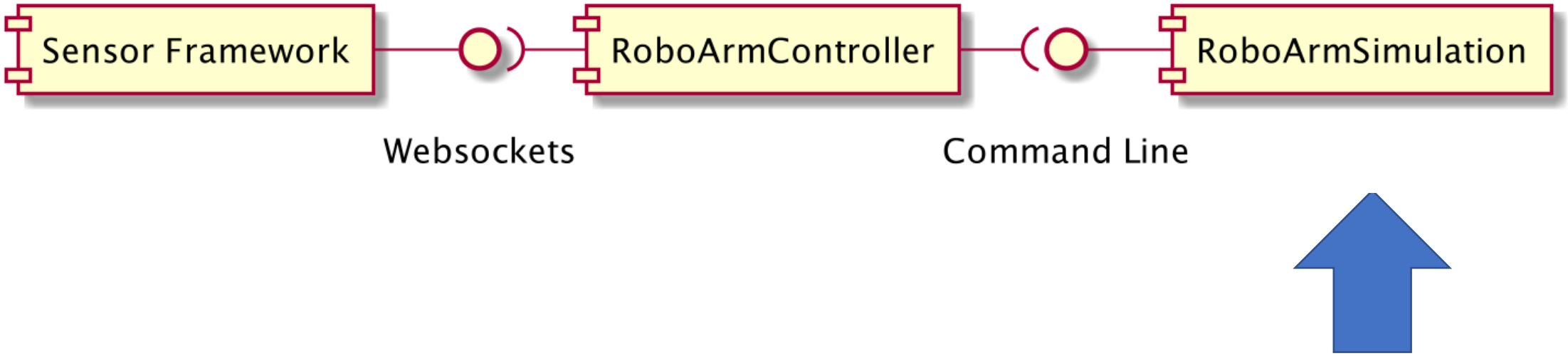
Image Source: Photo by Muyuan Ma on Unsplash

Target Architecture

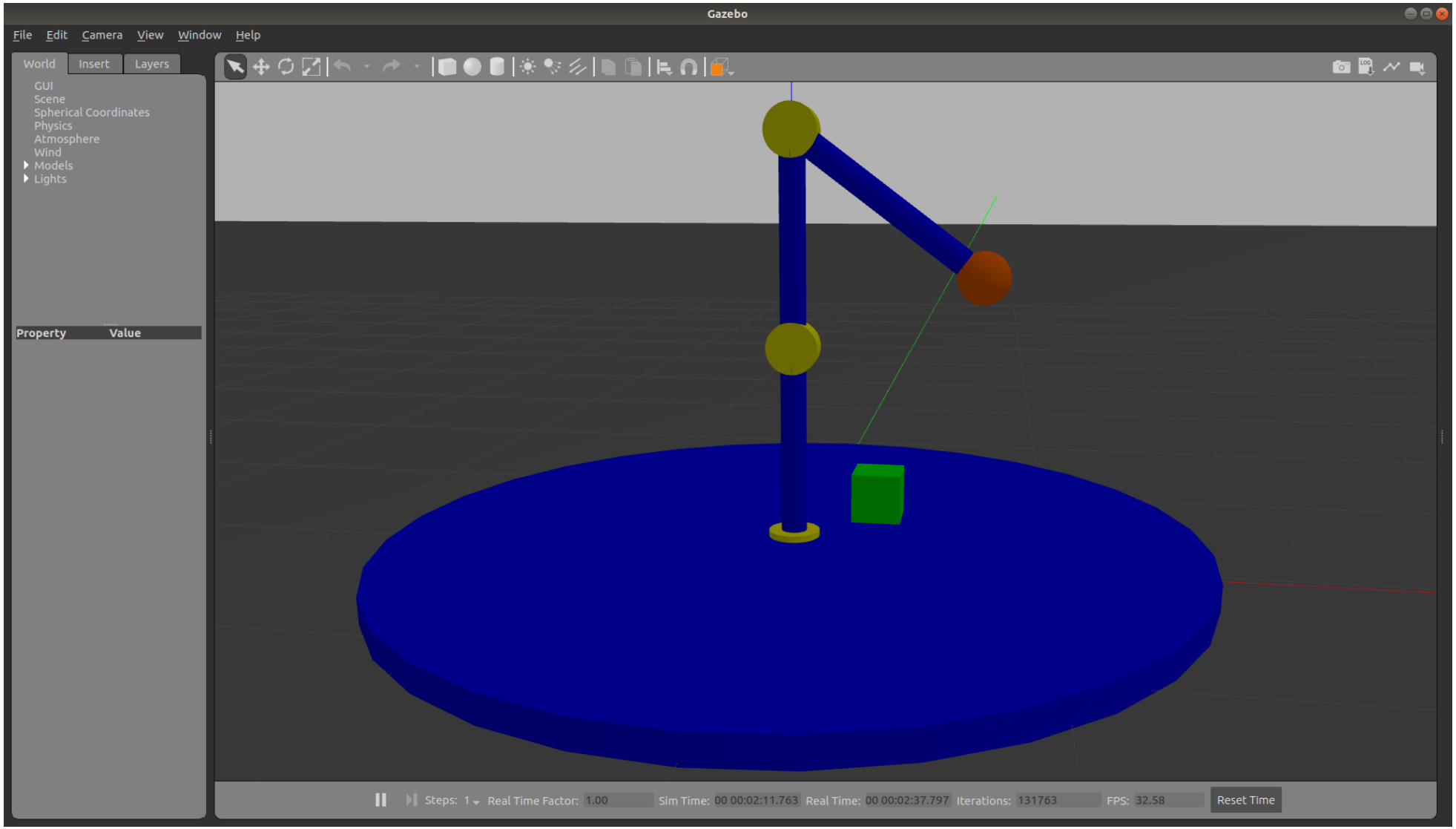


First Prototype: Architecture

The first piece: the simulation



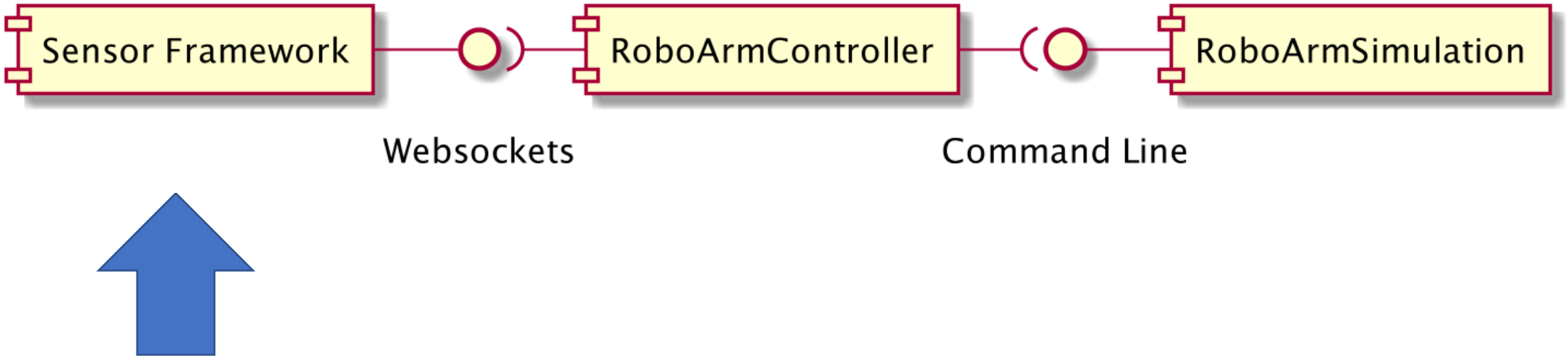
The first piece: the simulation



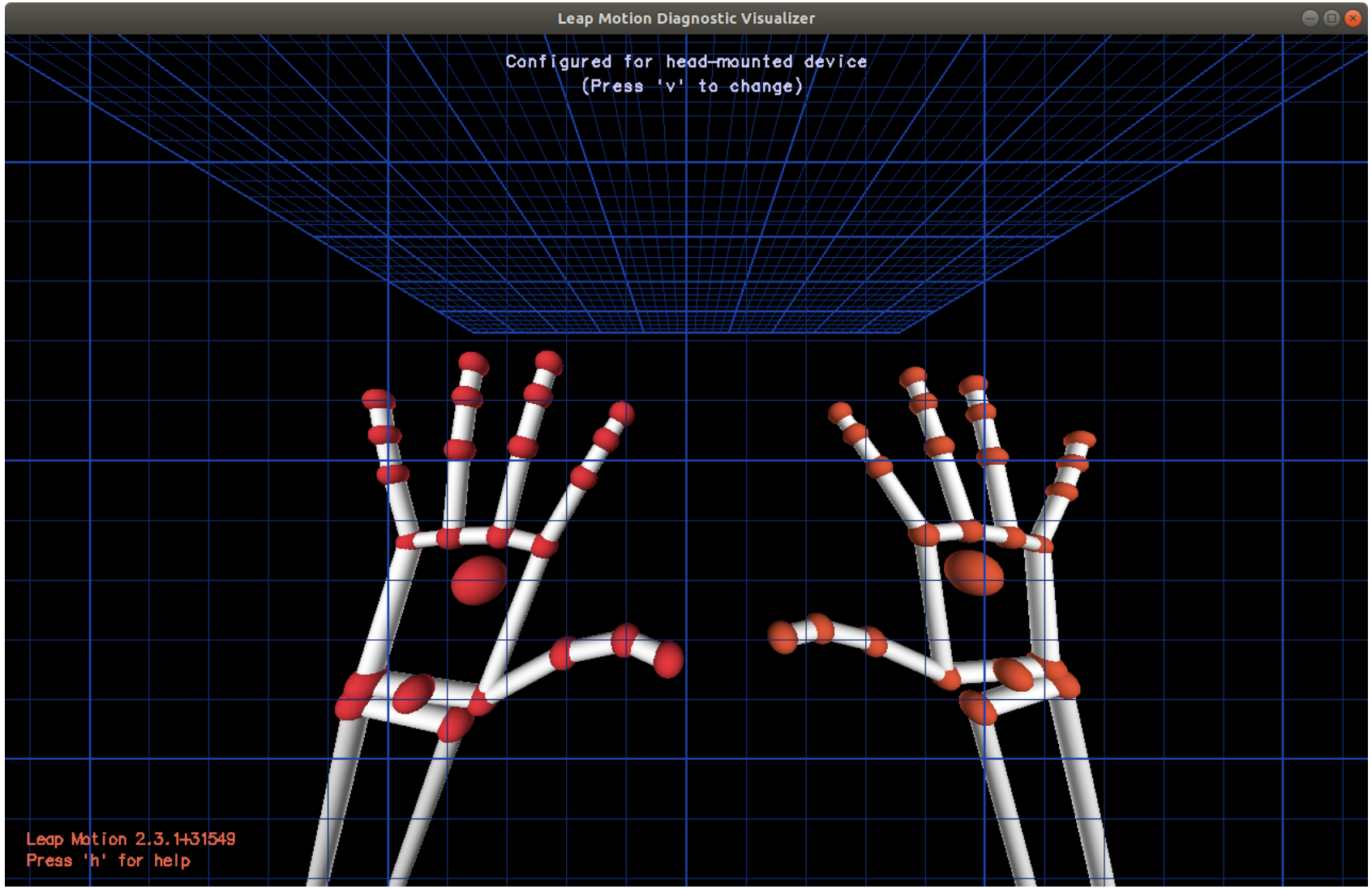
GAZEBO

Image Sources: cited in slide „Image Sources“

The second piece: Getting the data from the sensor



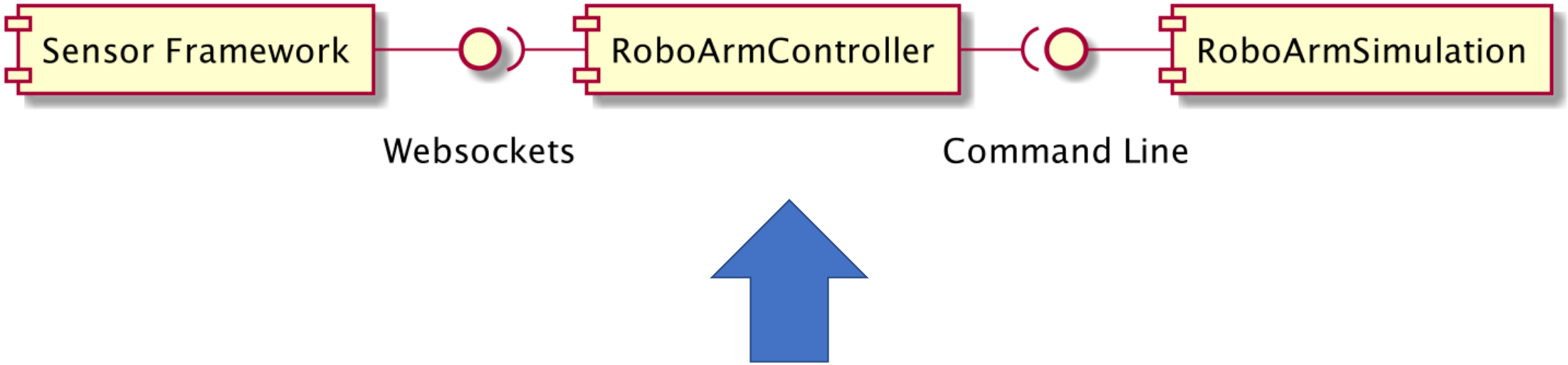
The second piece: Getting the data from the sensor



LEAP MOTION

Image Sources: cited in slide „Image Sources“

The third piece: The controller

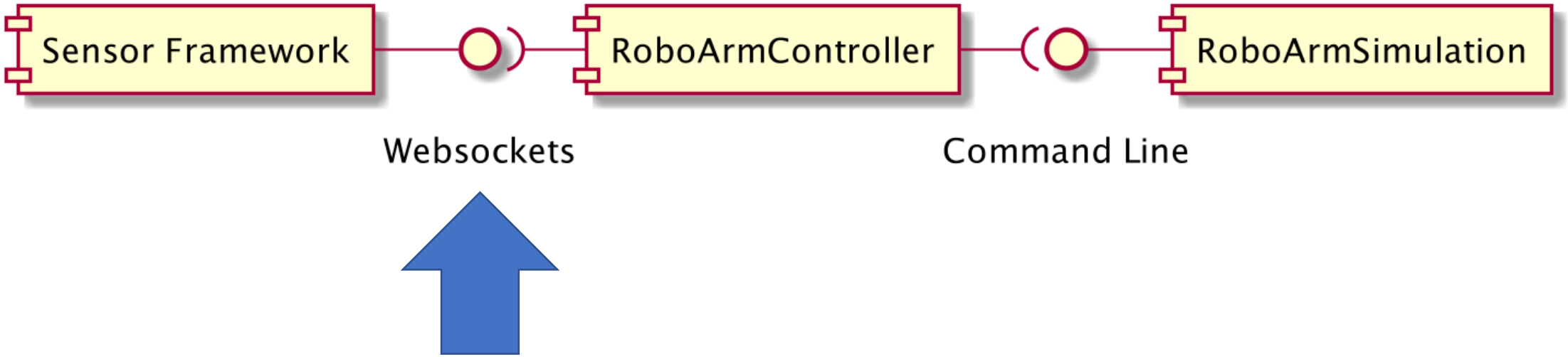


The third piece: The controller

- Communicate the hands sensor with the simulation
- Transform the hands data into a movement instruction

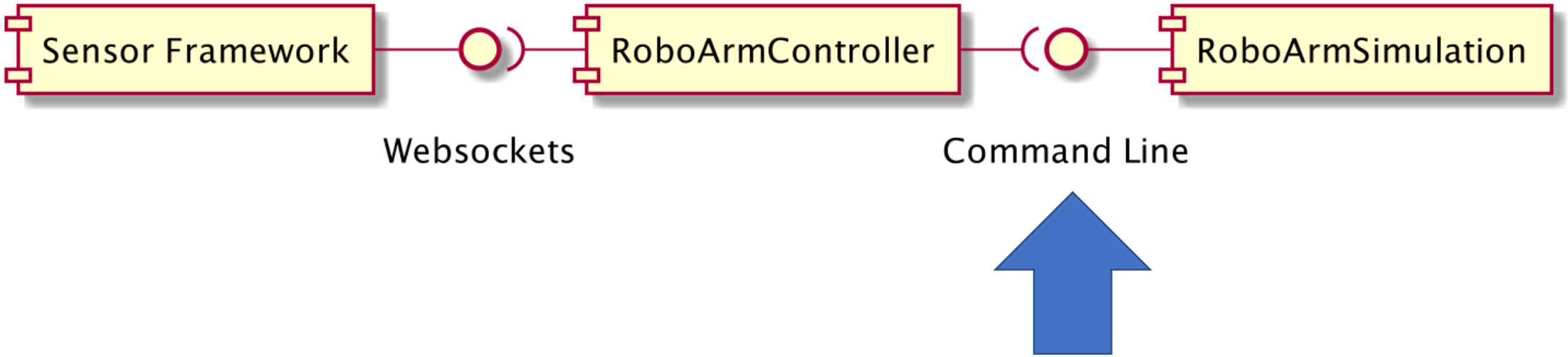
Image Sources: cited in slide „Image Sources“

Communication interfaces



```java
import org.java_websocket.client.WebSocketClient;
// ... more imports omitted

@Component
public class LeapMotionWebSocketListener extends WebSocketClient {
    // ... constructor omitted
    @PostConstruct
    public void startListener() {
        log.info("Trying to connect to: {}", this.getURI());
        connect();
    }

    public void onOpen(ServerHandshake serverHandshake) {
        log.info("WebSocket Opened");
        this.send("{\"focused\": true}");
    }

    public void onMessage(String json) {
        log.debug("Message Received {}", json);
        dataParser.publish(json);
    }

    public void onClose(int code, String reason, boolean remote) {
        log.info("WebSocket Closed. Code={}, Reason={}, Remote={}", code, reason, remote);
    }

    public void onError(Exception e) {
        log.info("WebSocket Error", e);
    }
}
```

Communication interfaces



```java
    public void execute(Parameters parameters) {
        try {
            String command = String.format(
              commandPattern, parameters.getServoId(), parameters.getDelta());

            log.info("Executing command, command {}", command);

            Runtime runtime = Runtime.getRuntime();
            Process process = runtime.exec(command);

            process.waitFor();
            BufferedReader b = new BufferedReader(new InputStreamReader(process.getInputStream()));

            String line;
            while ((line = b.readLine()) != null) {
                System.out.println(line);
            }
            b.close();
        } catch (Exception e) {
            log.error("Exception:{}", e.getMessage());
        }
    }

```

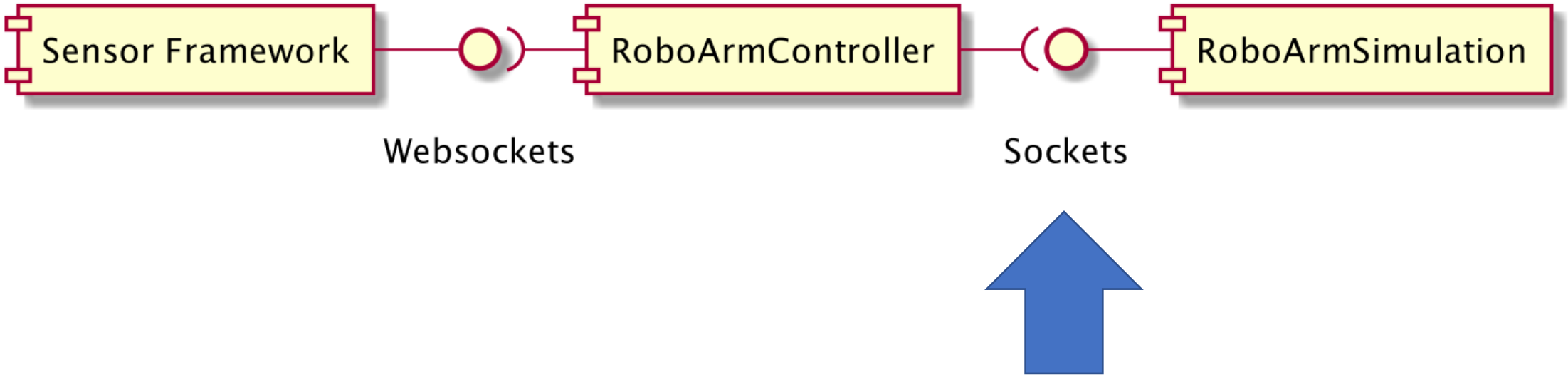
First Prototype: Result

FAILED!

Communicating the controller with the simulation through the command line is too slow.

Photo by John Torcasio on Unsplash

Fixing the first Prototype



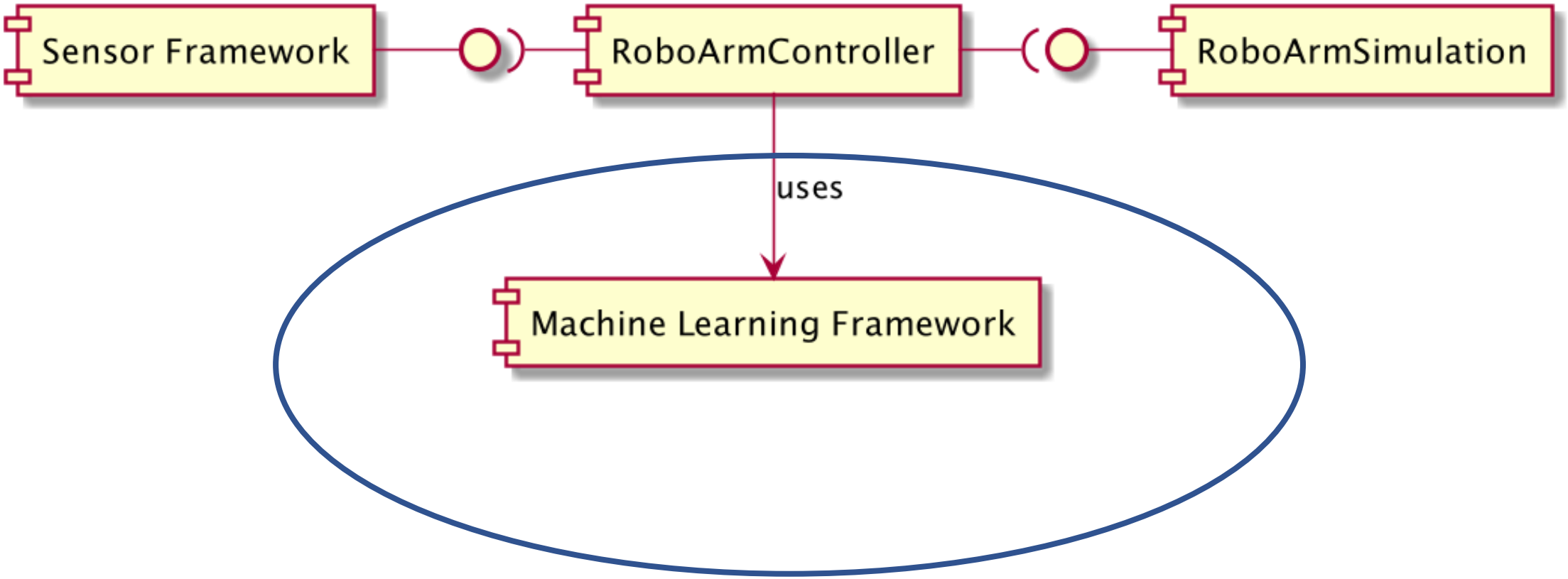
```java
@Component
public class SimulationGatewayTCP implements SimulationGateway {
    // ... fields omitted
    @Override
    public void send(Command command) {
        if (output == null) {
            this.initSocket();
        }
        intData[0] = command.getServoId();
        intData[1] = command.getDelta();
        byte[] data = convertIntArrayToByteArray(intData);
        try {
            output.write(data);
        } catch (IOException e) {
            log.error("Problem on writing to the socket");
        }
    }

    private void initSocket() {
        try {
            Socket socket = new Socket(SERVER_HOST, SERVER_PORT);
            output = socket.getOutputStream();
        } catch (IOException e) {
            log.error("Exception opening the socket", e);
            throw new RuntimeException("Exception opening the socket");
        }
    }
   // ... helper methods omitted
}
```

Photo by Katrina Berban on Unsplash

The next goal: Integrating Machine Learning

Photo by Markus Spiske on Unsplash

Target Architecture

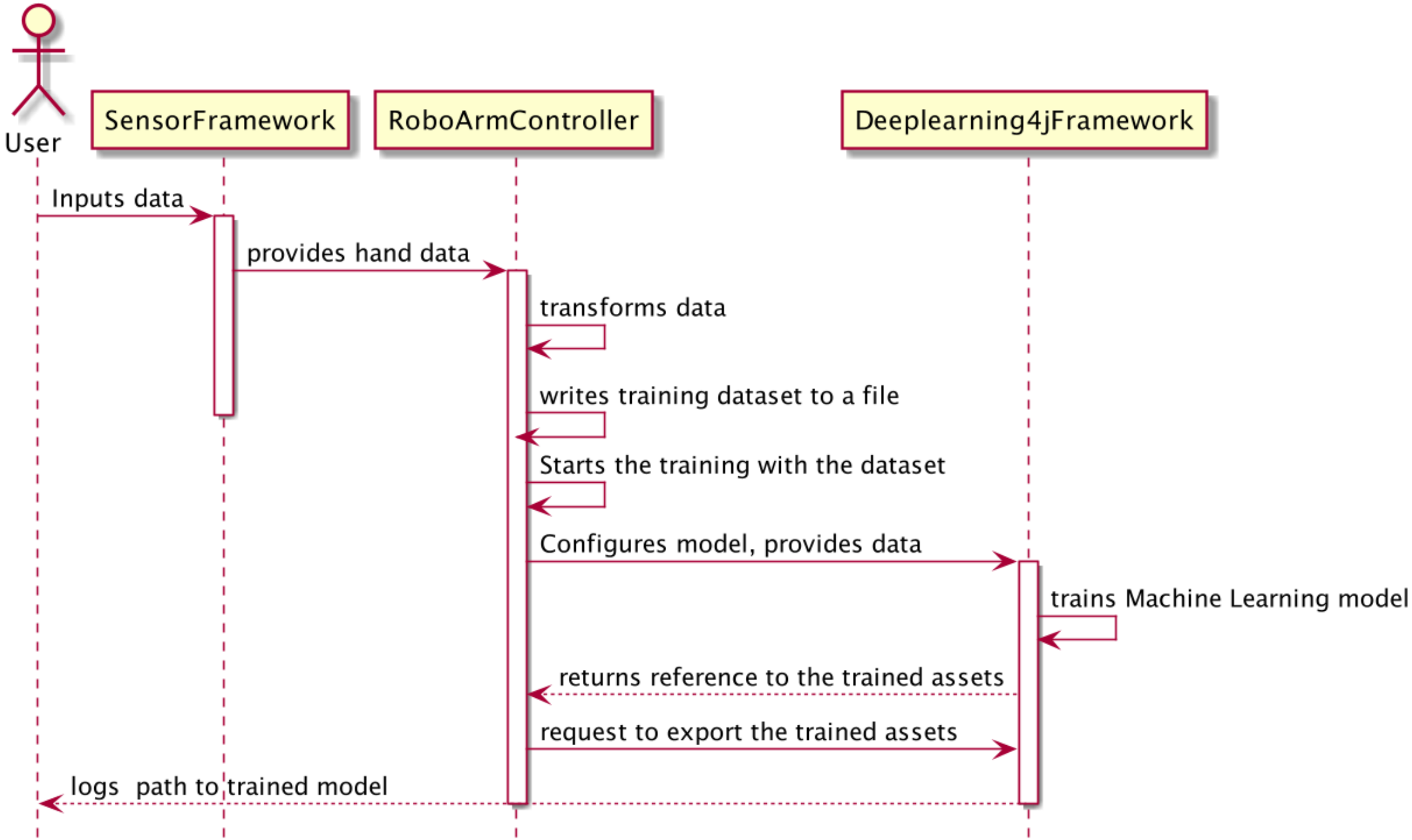


The fourth piece: Machine Learning

- 2 Phases:
  - Training
  - Controlling

- Training:
  - Data set generation
  - Machine Learning model training

Image Sources: cited in slide „Image Sources“

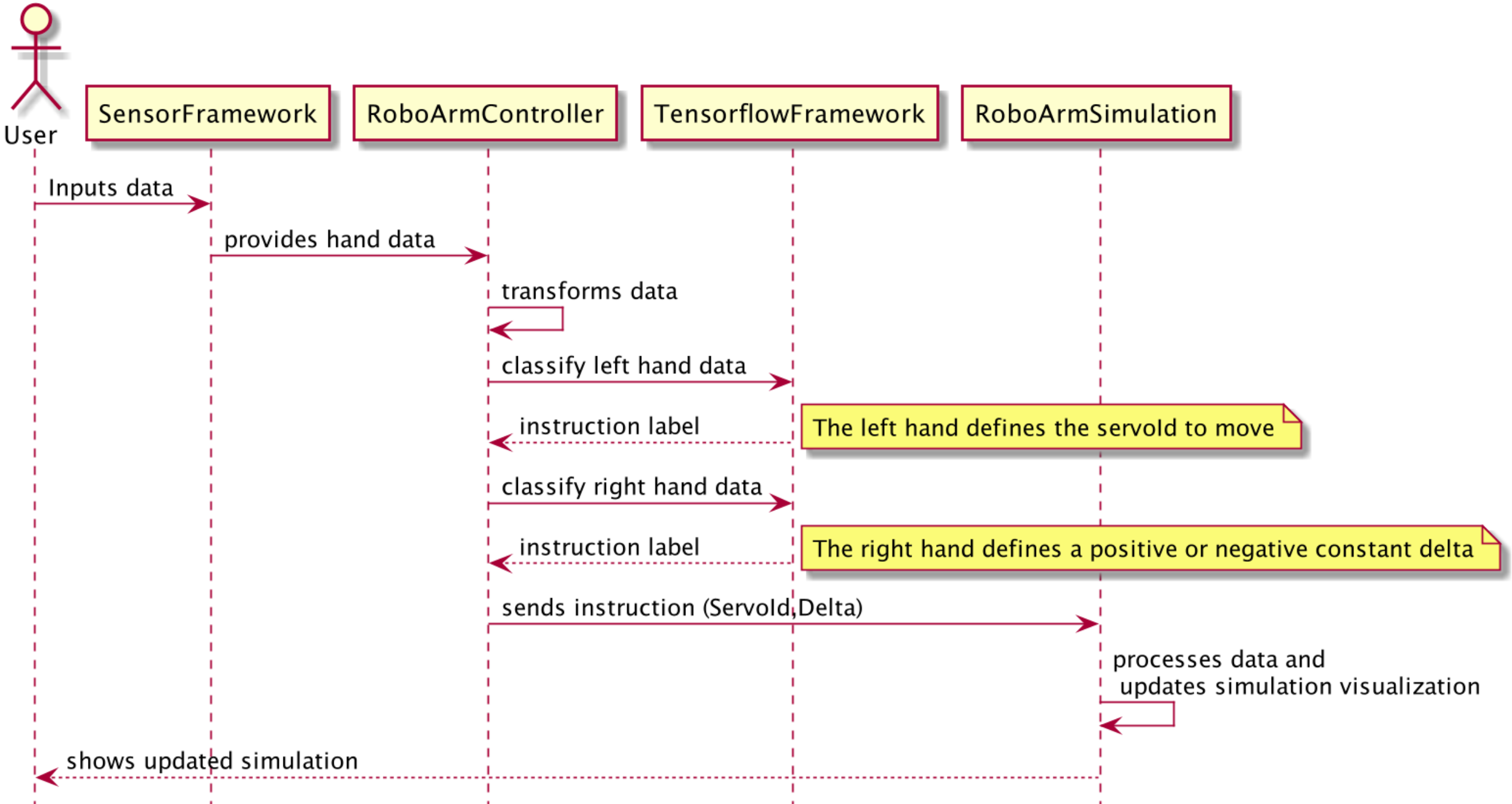

Training Process: Generate Dataset, Train and Export Model
User
SensorFramework
RoboArmController
Deeplearning4jFramework
Inputs data
provides hand data
transforms data
writes training dataset to a file
Starts the training with the dataset
Configures model, provides data
trains Machine Learning model
returns reference to the trained assets
request to export the trained assets
logs path to trained model

Hand Type	Finger1_X	Finger1_Y	Finger1_Z	…	Finger5_Z	Instruction Label
0	-226.271	202.675	999.255	…	-105.923	0
0	-22.476	202.418	995.894	…	-113.055	0
0	-726.617	110.996	136.756	…	719.333	1
0	-723.216	110.019	153.189	…	871.511	1
…	…	…	…	…	…	…
1	-210.596	148.747	320.942	…	-711.063	3
1	-234.792	147.133	32.069	…	-708.719	3
1	372.292	171.954	-631.447	…	-383.639	4
1	370.837	167.724	-606.725	…	-365.966	4

Dataset Labels

Instruction Label	Meaning
0	Servo 1
1	Servo 2
2	Servo 3
3	Positive Delta
4	Negative Delta

Controlling using Machine Learning



Enough BLA BLA! Show me the Deeplearning4j Code!

Image Sources: cited in slide „Image Sources“

Image Source: Photo by Jeffrey F Lin on Unsplash

Biggest Challenges

- Use an iterator instead of the dataset to train the Neural Network.
- The data on the NDArray for prediction needs to be a float and not a double.
- The Neural Network performs really poorly if we dont normalize it. If we do so, we need to also normalize the input for prediction.

Not so big challenges

- The empty constructor of the NDArray throws an NPE when we try to add elements to it.
- The data needs to be shuffled, or the Neural Network will perform really bad.
- The Neural Network seems to be overfitting. It has a better performance with less data.

What is missing?

Image Sources: cited in slide „Image Sources“

Open Issues

- UX is bad.
- Sockets in the simulation are blocking.
- Always using the classified data, even if it does not match a trained instruction.
- Bug: the simulation makes big jumps of state.

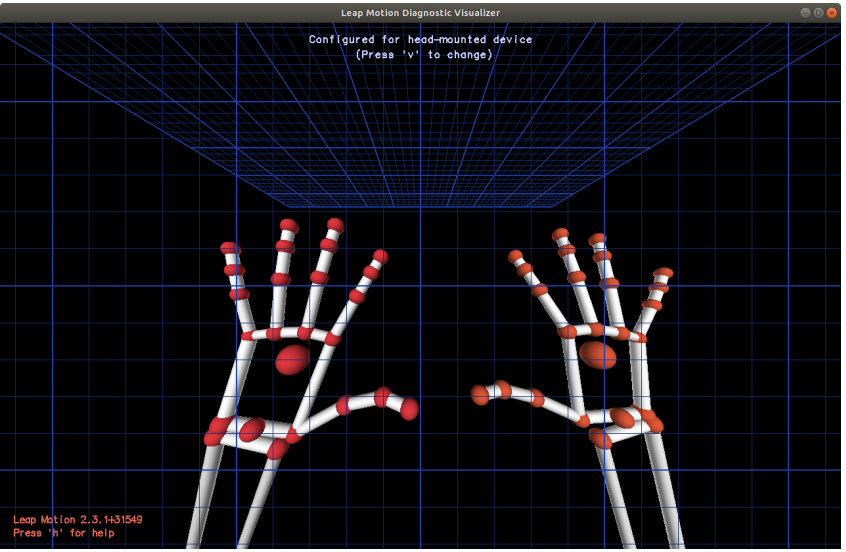
What went well?

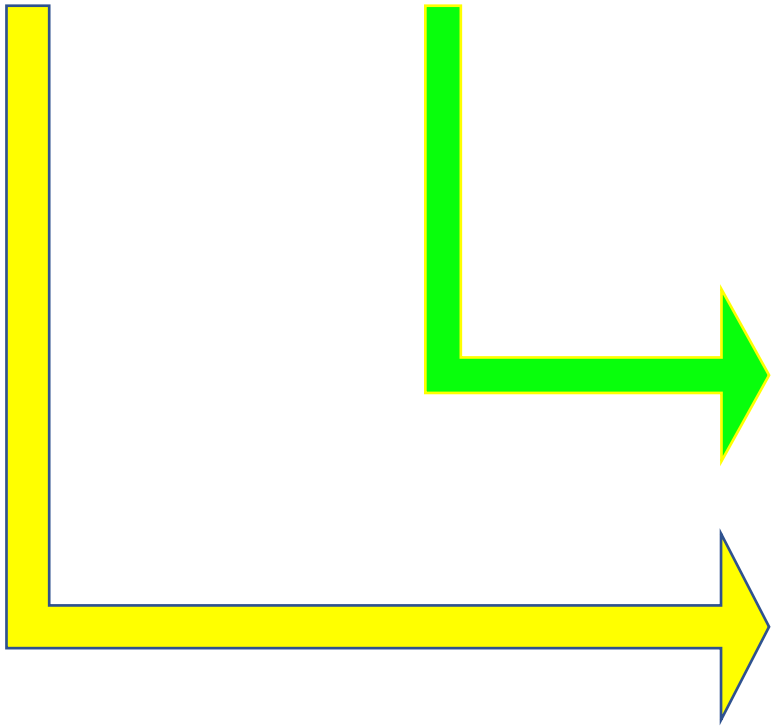
Image Sources: cited in slide „Image Sources“

Next steps / opportunities

- Evaluate which open issues are worth to solve.
- Challenge the necessity of Kafka to communicate the components.
- Implement a formal reactive approach on the controller?
- Use ROS on top of Gazebo?

DEMO!





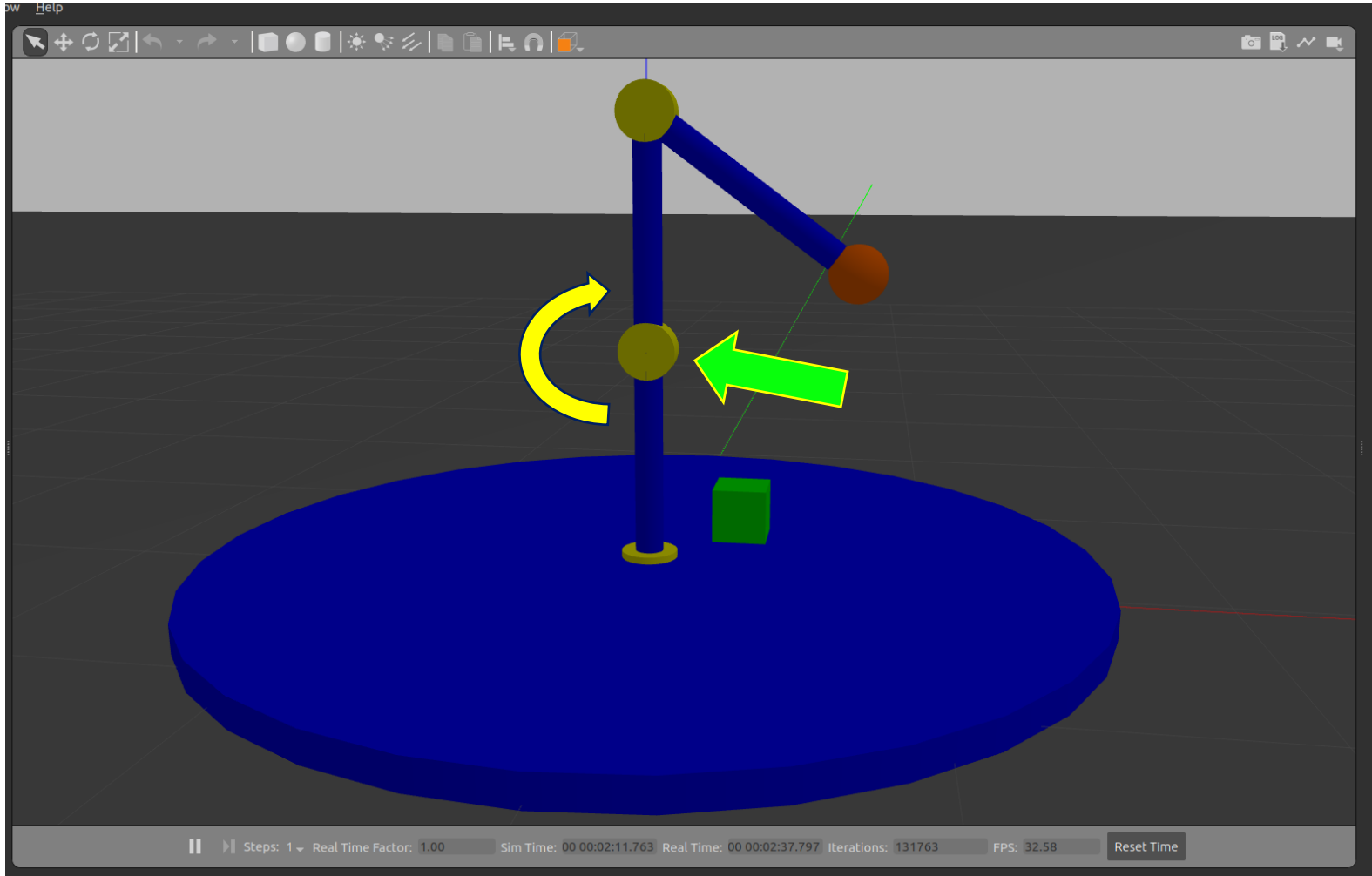
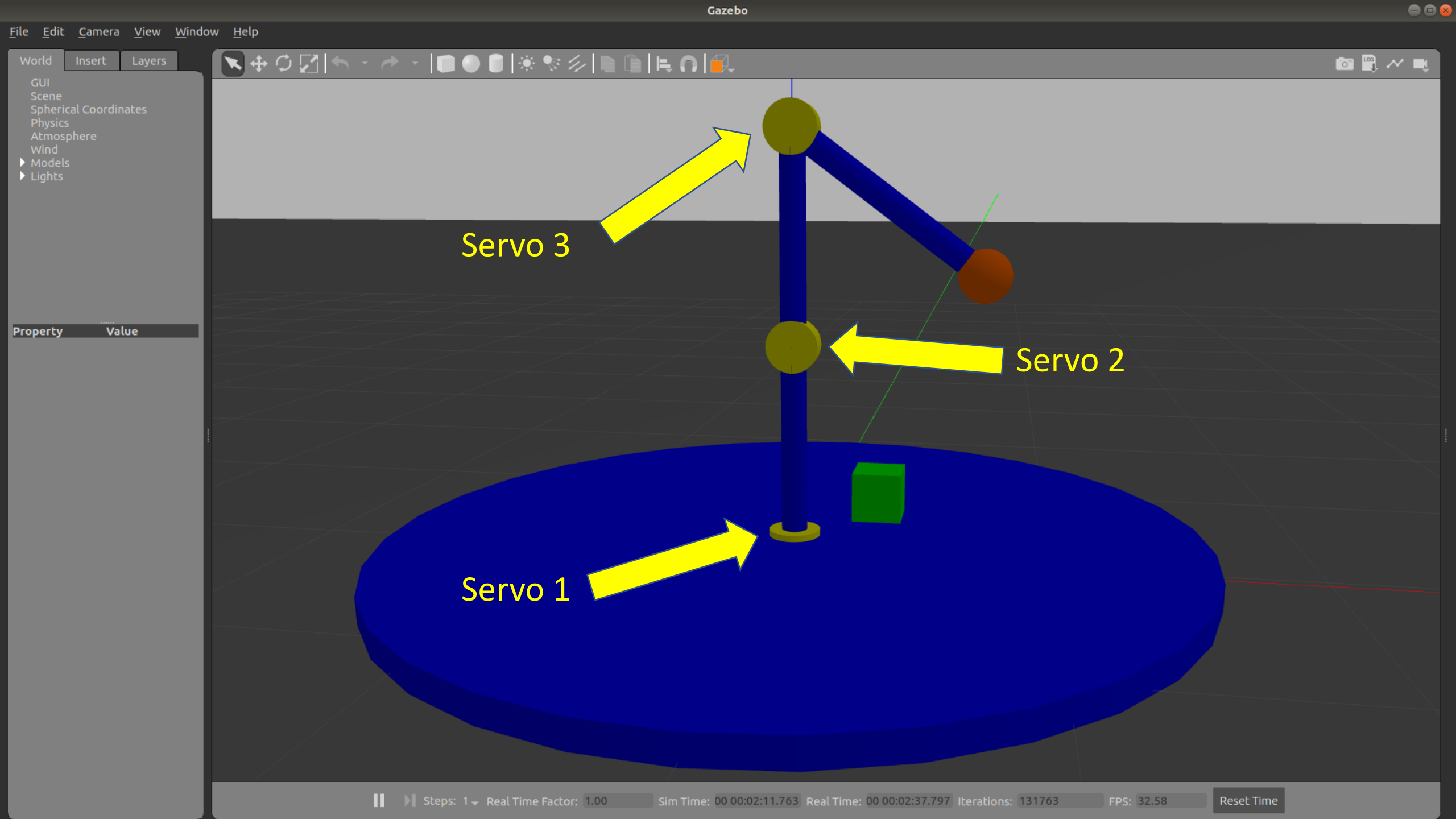
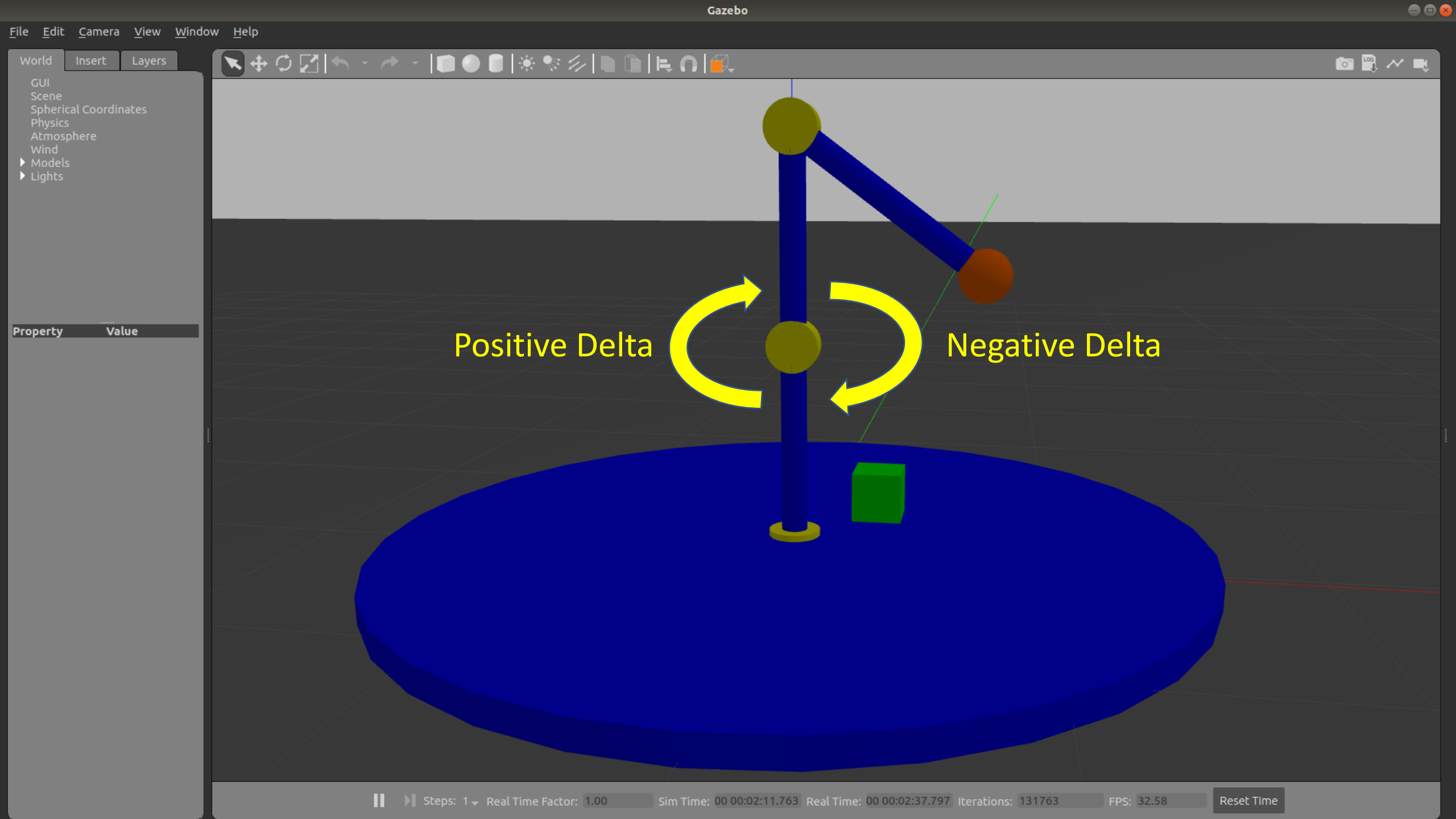


Image Sources: cited in slide „Image Sources“


Gazebo
File
Edit
Camera
View
Window
Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property
Value
Servo 3
Servo 2
Servo 1
Steps: 1
Real Time Factor: 1.00
Sim Time: 00 00:02:11.763
Real Time: 00 00:02:37.797
Iterations: 131763
FPS: 32.58
Reset Time


Gazebo
File
Edit
Camera
View
Window
Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property
Value
Positive Delta
Negative Delta
Steps: 1
Real Time Factor: 1.00
Sim Time: 00 00:02:11.763
Real Time: 00 00:02:37.797
Iterations: 131763
FPS: 32.58
Reset Time

Thanks!

Github:
ellerenad

Repo:
RoboArmController (here)

Repo:
RoboArmController-spike-deeplearning4j (here)

enrique.dominguez@mimacom.com
ellerenad@hotmail.com

Image Sources

- Apache Kafka Logo, https://kafka.apache.org/
- C++ Logo, By Jeremy Kratz - https://github.com/isocpp/logos, Copyrighted free use, https://commons.wikimedia.org/w/index.php?curid=62851110
- Gazebo Logo, http://gazebosim.org/
- Deeplearning4j Logo, https://www.facebook.com/deeplearning4j/
- Hand Data Visualization: Screenshot of the Visualizer tool of the Leap Motion SDK
- Java Logo, By Source (WP:NFCC#4), Fair use, https://en.wikipedia.org/w/index.php?curid=51298819
- Junit 5 Logo, https://junit.org/junit5/assets/img/junit5-logo.png
- Leap Motion Logo: By Leap Motion - http://2015s.hackbu.org/img/sponsors/leap-motion.svg, Public Domain, https://commons.wikimedia.org/w/index.php?curid=39152503
- Maven Logo, By Apache Software Foundation - Apache Software Foundation http://maven.apache.org/images/maventxt_logo_200.gif, Attribution, https://commons.wikimedia.org/w/index.php?curid=10365467
- PlantUML User Diagram, generated with the PlantUML tool.
- Python Logo, By www.python.org - https://www.python.org/community/logos/, GPL, https://commons.wikimedia.org/w/index.php?curid=34991637
- RoboArmSimulation: Screenshot for the simulated robotic arm, generated used the Gazebo simulator.
- Tensorflow Logo, By TensorFlow - vectors combined, edited - Begoon - https://github.com/tensorflow/tensorflow https://github.com/valohai/ml-logos/blob/master/tensorflow-text.svg https://github.com/valohai/ml-logos/blob/master/tensorflow-tf.svg , Apache License 2.0, https://commons.wikimedia.org/w/index.php?curid=65268375