

# **Robotics – Tutorial for Assignment #1**

---

By Előd Páll

# The Puma560 Simulator

---

- ▶ Software that simulates the Kinematics, Dynamics, Friction etc. of the Puma560 robot
- ▶ *pumasim* binary or Virtual-Box
- ▶ Provides a GUI for controlling, monitoring and configuring the simulation



# Running the simulator natively

---

- ▶ Known to work on Ubuntu 20.04 and 18.04
- ▶ Download the pumasimulator-xxxx.tgz

```
tar -xzvf pumasimulator-xxxx.tgz
cd pumasimulator
```
- ▶ Read the *Readme.md* for installation instructions

## Running the simulator with a VM image

---

- ▶ Install Oracle **Virtualbox 6.1**  
<http://virtualbox.org/>
- ▶ Download the Virtual Machine image (.ova) from ISIS
  - Tip: Do it on the campus net (ca. 2.6GB)
- ▶ Start the machine
- ▶ Login: student
- ▶ Password: student
- ▶ Open a terminal and type *pumasim*

# Simulator internals

---

- ▶ Controller are called every 2 ms
- ▶ Predefined names
- ▶ The pumasim executable does not contain controllers, but loads them from the shared library *controlDLL.so*
- ▶ Pumasim first looks for the library in the current working directory, then in /opt/pumasim
- ▶ You only need to compile controlDLL.so

# control.cpp

---

- ▶ Here you implement your robot controller
- ▶ **Important:**
  - File must also be compatible on the Real-Time-PC running QNX.
- ▶ `init..()` functions are called when you click on „Start“ (controller).
- ▶ `..control()` functions are called periodically in the servo loop with 500Hz.
- ▶ A lot of global variables are declared via the structure `gv`: they contain the simulator/robot's state

# data.mat

---

- ▶ Plain ASCII text file

- ▶ Each line is a timepoint:

time q(1..n) dq(1..n) qd(1..n) tau(1..n) x(1..m) dx(1..m) xd(1..m)

- ▶  $n = \text{DOF}$

- ▶  $m = 7$  in „6-DOF (quaternions)“ mode  
else  $m = \text{DOF}$

- ▶ You can import it into Excel, Matlab, gnuplot, Octave, matplotlib, a.s.o. and make nice graphs

## gains\_\*.txt

---

- ▶ Text file containing separate gains for all controllers for a specific robot mode
  - gains\_1.txt = gains during 3DOF mode
  - gains\_6.txt = gains during 6DOF quaternion mode
  - a.s.o.
- ▶ Please do not edit the text file manually, but use *Store gains* and *Load gains in the GUI*



# Before you start coding

---

- ▶ **Read** *Notes and Restrictions on Coding.pdf* (available on ISIS)
- ▶ Information about available math library (vector, matrix, etc.), global variables, etc.

# P-controller

---

- ▶ Important variables for the P-controller in the `gv` struct:

`tau` : joint torques

`q` : joint position

`kp` : position gains for the current control mode

`qd` : desired joint position

- ▶ You can tune your controller via the GUI
- ▶ You can visualize signals by writing them to a text file (.mat)

# Compile System for controlDLL

---

- ▶ Cmake based compile template:

```
cd 1/
```

```
mkdir build
```

```
cd build
```

```
cmake ..
```

```
make
```

- ▶ This creates a controlDLL.so from control.cpp
- ▶ To use your controller, call *pumasim* in the *build/* directory:  

```
pumasim
```

# Q&A

---

- ▶ ISIS discussion forum
- ▶ [teaching@robotics.tu-berlin.de](mailto:teaching@robotics.tu-berlin.de)



# Puma Simulator

The screenshot shows the Stanford Puma Simulator interface. On the left, a 3D model of a Puma robot arm is visible. On the right, the control panel is divided into several sections: Control, Settings, Virtual Scene, and Plotting. The Control section includes a mode selection dropdown (set to 'float'), a table for position gains (kp and kv), and buttons for 'Store gains' and 'Load gains'. The Output Files section has buttons for 'Start' and 'Stop' data gathering, a filename field (data.mat), and a 'not running' status. The Status section displays a table of joint positions (q), velocities (dq), and torques (tau), along with end effector position (x) and force (F). At the bottom, there are buttons for 'Stop CV', 'Start CV', 'FLOAT', and 'EXIT', along with a dropdown for simulation type (set to '6-DOF (axis angle)').

**control mode selection**

**position gains for the current control mode**

**store and load gains to file**

**data output for plotting**

**Type of simulation (DOF)**

**Control** | Settings | Virtual Scene | Plotting

float Start         paste

float Start         paste

float Start         paste

float Start         paste

**Control Gains**

kp:  400.0  400.0  400.0  400.0  400.0  400.0

kv:  40.0  40.0  40.0  40.0  40.0  40.0

Store gains Load gains

Alternate parameters...

**Output Files**

Gather Data: Start Stop data.mat not running

Gripper close open

**Status**

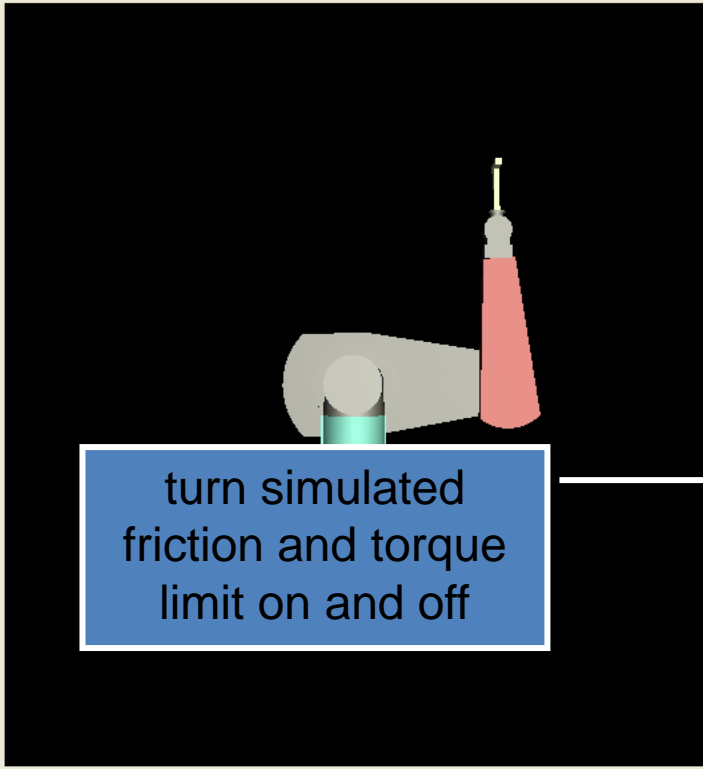
q	0.0	0.0	0.0	0.0	0.0	0.0		copy	Plot
dq	0.0	0.0	0.0	0.0	-0.0	0.0		copy	Plot
tau	0.0	-43.0	0.3	0.0	-0.0	0.0			Plot
x	0.41	0.15	0.59	0.0	1.00	0.00	0.00	copy	Plot
F	0.00	0.00	0.00	0.00	0.00	0.00			

6-DOF (axis angle) Stop CV Start CV FLOAT EXIT

robotype 6 Simulator 80.91

# Puma Simulator Settings

Stanford Puma Simulator



turn simulated friction and torque limit on and off

Control Settings

qmin	-223.0	-40.0	-100.0			
qmax	43.0	240.0	100.0			
dqmax	45.0	45.0	45.0			
ddqmax	60.0	60.0	60.0			
q0	5.0	5.0	5.0			
kj	0.10	0.10	0.10			

vmax 0.30 sbound 10.00 joint limits  
amax 10.00 cgripper 0 friction  
wmax 45.00 ogripper 2048 torque limit  
rho0 0.05 spring K 200.0  
eta 0.010 Sim Speed 1.000

Virtual Line

☐ Draw line from A to B

point A -0.50 0.00 0.00  
point B 0.50 0.00 0.00

Obstacles

x	y	z	R

add delete

Status

q	-0.8	-0.1	0.0					copy
dq	0.0	-0.0	0.0					copy
tau	-37.1	1.3	-0.0					
x	0.40	0.60	-0.9					copy
F	0.00	0.00	0.00	0.00	0.00	0.00		

3-DOF Stop CV Start CV FLOAT EXIT

qmin -223.0 -40.0 -100.0 Simulator 1141.72