First Milestone Presentation

Moving 2

Technische Universität Berlin

SESE

Structure

1. Introduction to Reinforcement Learning
2. Project Requirements
3. Basic Approach
4. First Results
5. Schedule
6. Sources

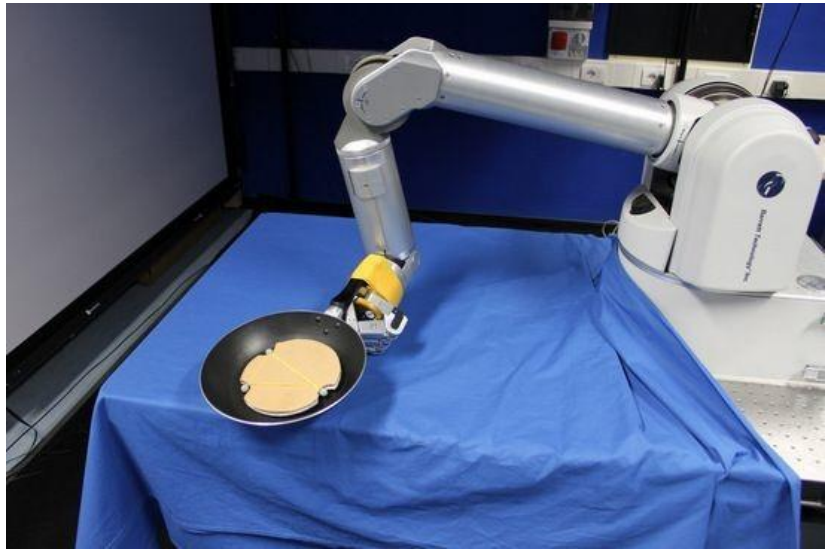
Introduction to Reinforcement Learning

Why Reinforcement Learning in Robotics?

Why don’t we just program robots?

- more “natural” movement with less programmer knowledge
- possibility to find solutions beyond what developers think is optimal
- allows adaptation to changes (Kormushev, 2013)

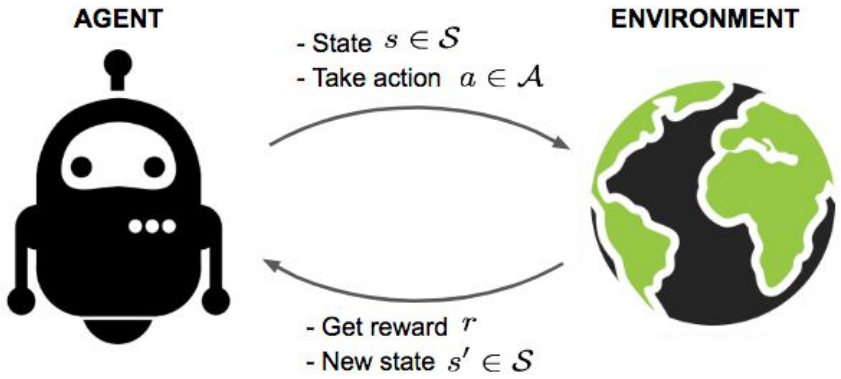
1: Teach Pendant



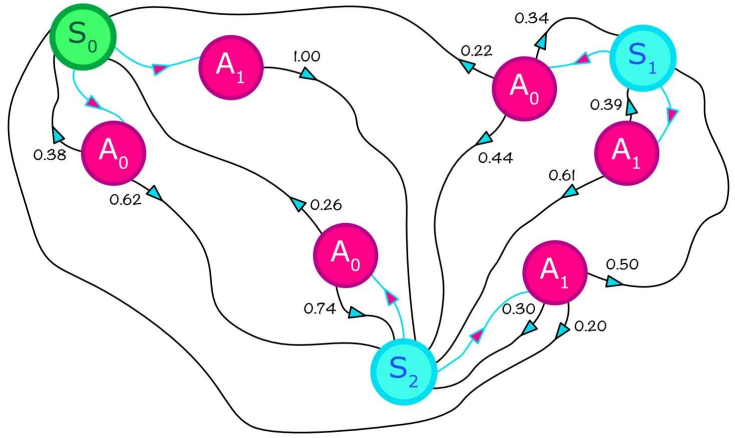
2: Pancake Robot

Markov Decision Process

- from a state, an agent can take possible actions
- only current state relevant
- action outcomes in specific states are probabilistic
- states are a representation of the environment, based on an observation of the environment



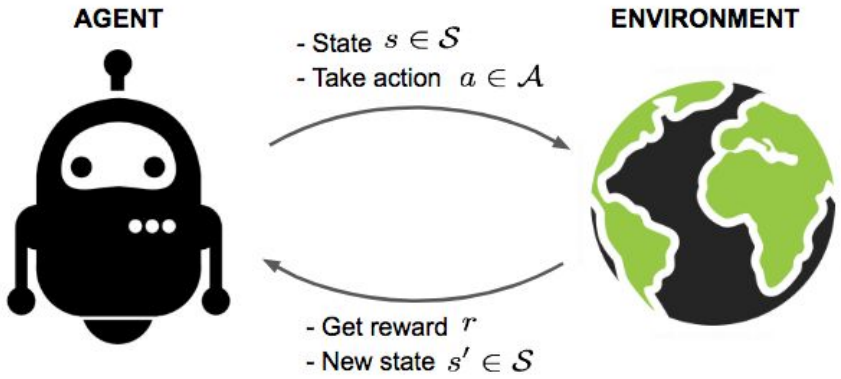
3: Fundamental processes of RL



4: States and possible actions

Reward Functions

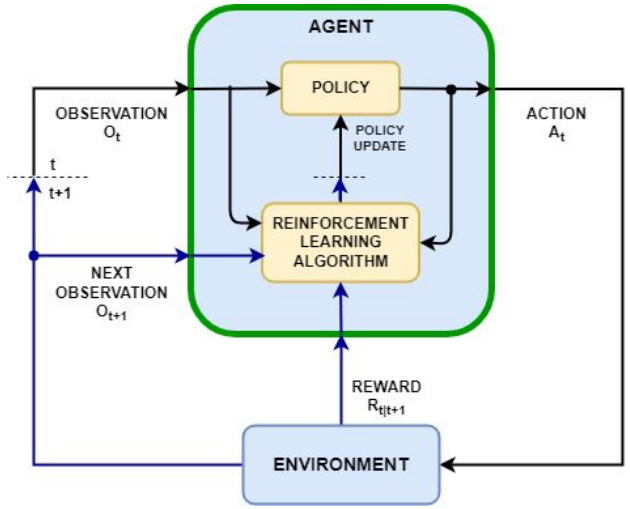
- Reward functions receive state and action and return reward
- Rewards accumulated over a trajectory are a return
- Value: return agent expects to receive from an action in a state and onwards



3: Fundamental processes of RL

Policies and Learning Algorithms in RL

- Agent decides based on a policy function, which takes the state and outputs an action -> Parameters in policy function are to be optimized!
- Direct and indirect policy mapping
- Model-free and Model-based



5: Reinforcement Learning Framework

Project Requirements

Project Requirements

- **Specified:**
  - movement forward (not driving)
  - use of at least two motors
  - use of sensors
  - learning of movement with reinforcement learning

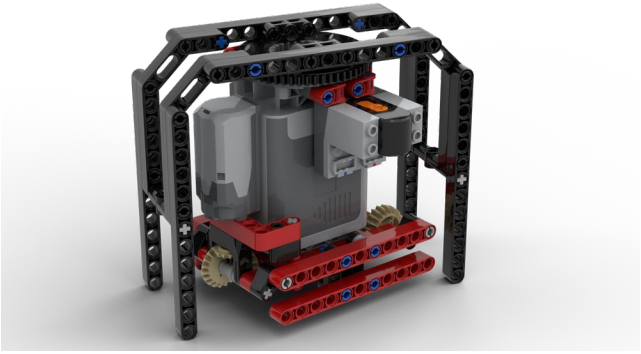
- **Safety Requirements:**
  - avoidance of collisions
  - does not fall over

Basic Approach

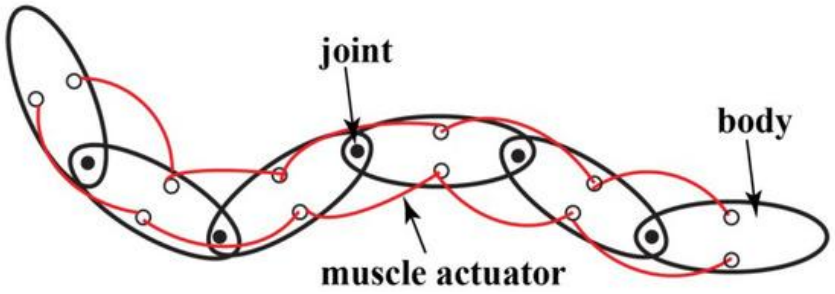
The Robot

- Possible types of movement: jumping, rolling, shuffling, walking, slithering…
- To consider:
  - stability
  - control of movement
  - mechanical complexity
  - number of motors required
  - placement of sensors

6: Jumping robot



7: Omnidirectional walker



8: Mechanism of a snake's movement

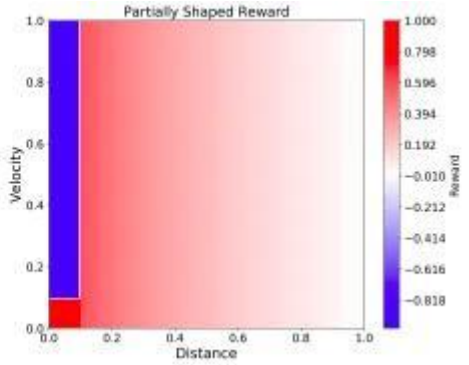
Reward Function

- receives state and action and returns reward
- Challenges and Solutions
  - sparse rewards and reward shaping
  - cobra effect and incentivizing what you intend
  - positive and negative rewards
- how to shape the function: space, time, sensors

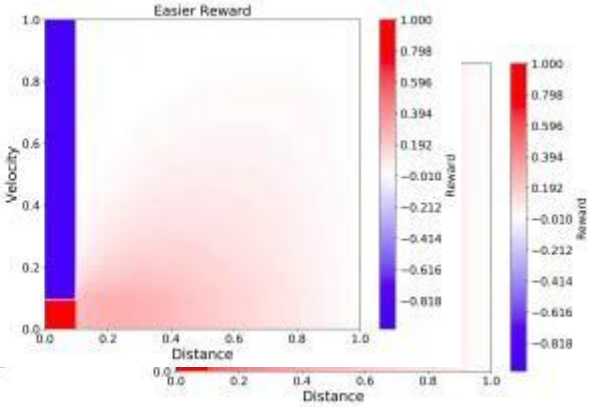
Blue == "crash"

```
# Encourages getting close,
# but goes too fast!
reward = 1 - distance^0.4
```

Red == "success"




```
# Better to also reward for going slowly
dist_reward = 1 - distance^0.4
vel_discount = (1 - max(velocity,0.1))^(
                    1/max(distance,0.1))
reward = vel_discount * dist_reward
```



9: Reward function and improvement

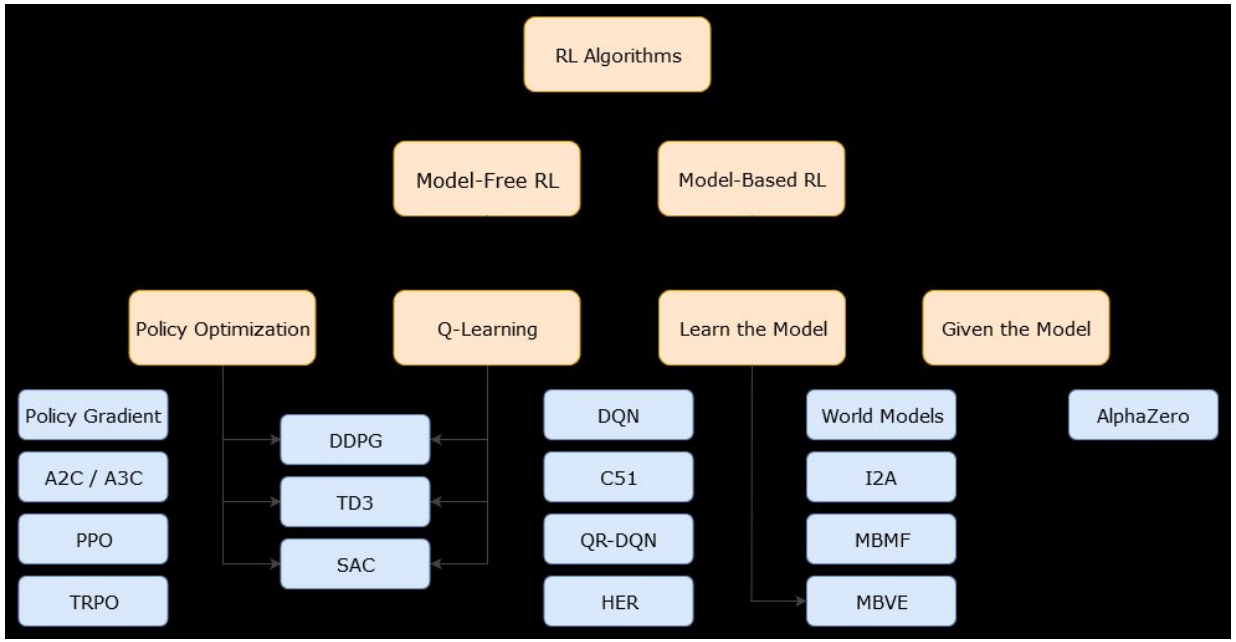
Observation and Action Space

- **Observation space**
  - data that the robot can collect from the environment
  - can be continuous and discrete

- **Action Space**
  - contains possible actions
  - can be continuous and discrete

RL Algorithm

- Model-free vs Model-based
- Whether the agent uses predictions of the environment response
- We are dealing with the real world (complex environment)

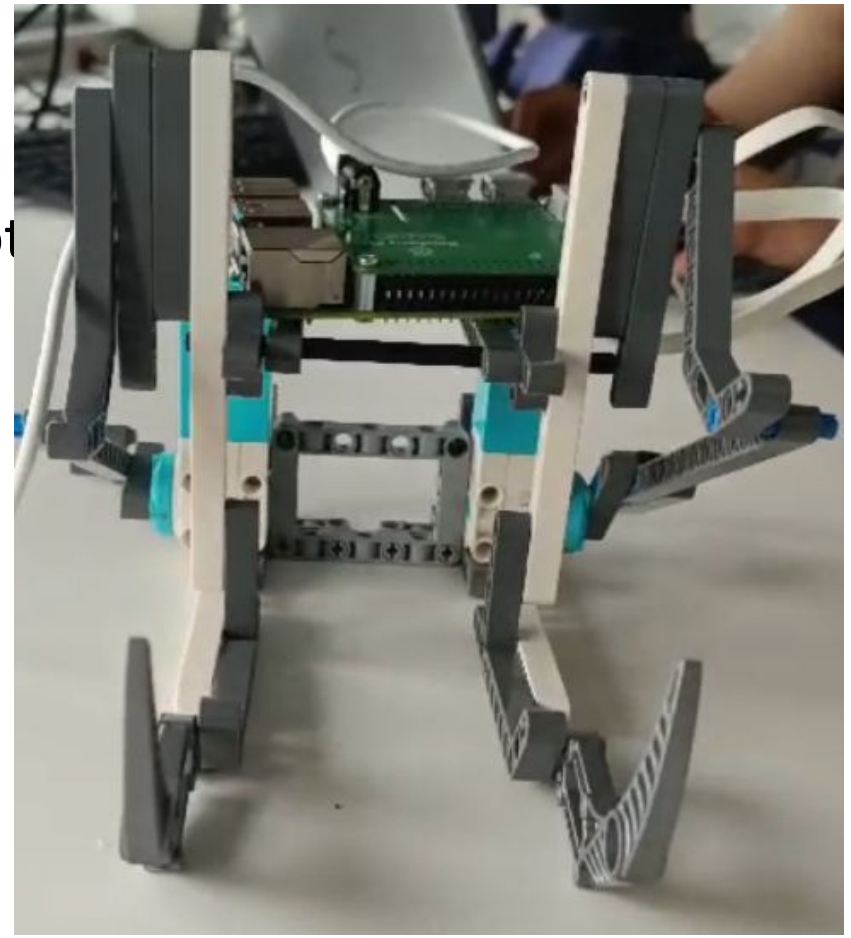


11 : RL Algorithms classification

First Results

Basic robot Design

- “Ski-Robot” :
  - uses 2 Motors
  - stable
  - moves forward most efficiently moving 2 mot
  - able to rotate by moving one motor



System Architecture

- Possibilities
  - Data collection and RL agent on RPI
  - Data collection on RPI and fed instantly to the agent in a server.
  - **Offline Data collection on RPI and feed after a few tests to RL agent.**

- Implementation
  - Hardware:
    - Raspberry Pi 4 + Lego Build Hat
    - 2 motors
    - 1 camera sensor
    - 1 distance sensor
  - Software:
    - Offline Data Collection on Raspberry Pi
    - Data Processing and RL Agent Training (Server or Computer)
    - Periodic Data Transfer to Agent over TCP/IP
    - Send Feedback from Pi/ Agent to Server and Rerun for iterative improvements

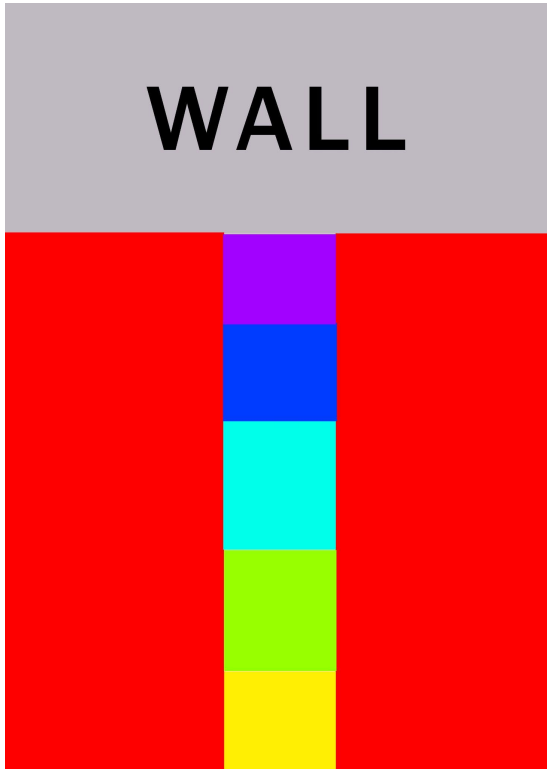
Test Environment

- Rectangular shaped room

- Floor in Testing space is covered in colored Tape :

  - Straight line in Different colors of Tape (representing levels) (or one color)

  - Sides of the line in a specific color Tape



Observation Space

- **Horizontal distance to walls :**
  - continuous space
  - measured with distance Sensor

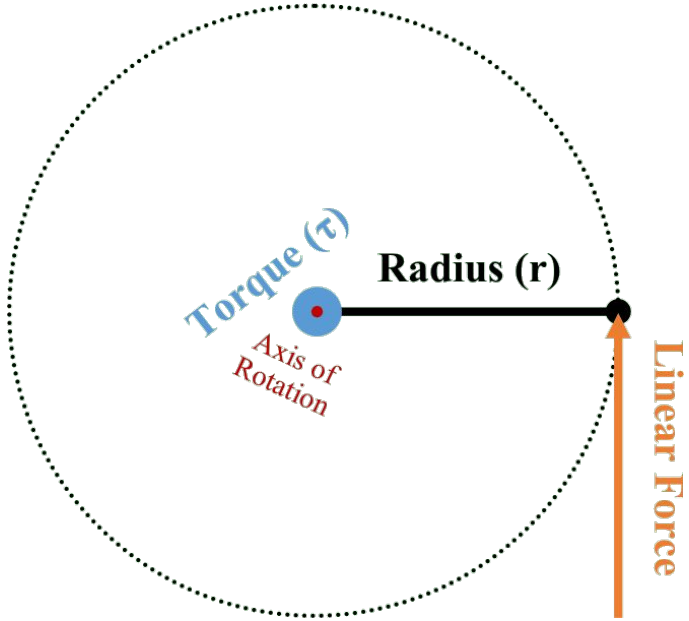
- **Color of current position :**
  - discrete space
  - measured with color sensor

- **Subsequent Action**

Action Space

- **Continuous : Motor torque**
  - continuous action space
  - for each motor

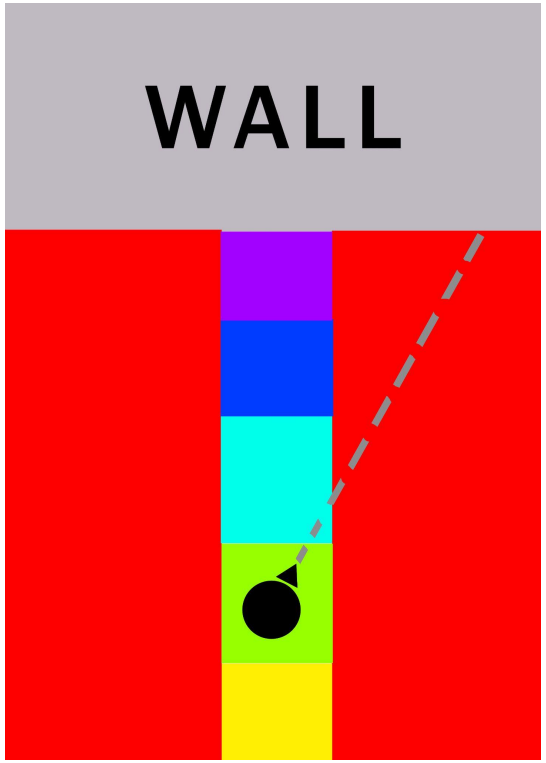
- **Discrete : pre-defined movements**
  - small, big , intermediate step ..



10 : Torque

Reward Function

- **Distance reward/punishment : (relative to distance)**
  - gets minor reward when distance gets smaller
  - gets minor punishment when distance gets bigger



- **Color reward/punishment :**
  - colors are ordered
  - side color being worse
  - gets major reward when going from a color to a “better” one
  - gets major punishment when going from a color to a “worse” one

Schedule

Plan

	07.05-13.05	14.05-20.05	21.05-27.05	28.05-03.06
Hardware				
Gather Information about Mechanics, robotics.. To make efficient first prototype.				
Decide Robot design (first prototype)				
Brainstorm System Architectures				
Build an efficient Prototype				
Decide System Architecture				
Refine Robot Design				

Plan

Software				
	07.05-13.05	14.05-20.05	21.05-27.05	28.05-03.06
Gather broad knowledge about RL Algorithms and how to implement them				
Understand C++ Library				
Select 2 RL Algorithms				
Implement RL Algorithms and try them out				
Improve RL Algorithm				
Brainstorm reward function				
first trial of a value function				

Plan

General				
	07.05-13.05	14.05-20.05	21.05-27.05	28.05-03.06
Define Requierments				
Define Observation Space (decide which sensors to use and how)				
Define Action Space (Define type of action)				

Sources

Reinforcement Learning:

Kormushev, Petar, Sylvain Calinon, and Darwin Caldwell. “Reinforcement Learning in Robotics: Applications and Real-World Challenges.” *Robotics* 2, no. 3 (July 5, 2013): 122–48. https://doi.org/10.3390/robotics2030122.

https://medium.com/@cedric.vandelaer/reinforcement-learning-an-introduction-part-1-4-866695deb4d1 (all parts)

https://medium.com/@BonsaiAI/deep-reinforcement-learning-models-tips-tricks-for-writing-reward-functions-a84fe525e8e0

https://de.mathworks.com/campaigns/offers/reinforcement-learning-with-matlab-ebook.html?gclid=CjwKCAjw0YGyBhByEiwAQmBEWvkDebcU6pwgRJ3Z9KIZYZpl4pIXoRO-pOeCbEH3CE8iThIlHnyo0hoCOUAQAvD_BwE&ef_id=CjwKCAjw0YGyBhByEiwAQmBEWvkDebcU6pwgRJ3Z9KIZYZpl4pIXoRO-pOeCbEH3CE8iThIlHnyo0hoCOUAQAvD_BwE:G:s&s_kwcid=AL!8664!3!642023603906!p!!g!!reinforcement%20learning&s_eid=psn_95574020043&q=reinforcement+learning&gad_source=1

(ebook)

https://de.mathworks.com/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html

Pictures

1: https://control.com/technical-articles/how-to-program-a-robot-industrial-robotic-arm-coding-basics/

2: https://www.mdpi.com/2218-6581/2/3/122 (see Source 1 in Sources)

3: https://lilianweng.github.io/posts/2018-02-19-rl-overview/

4: https://medium.com/@cedric.vandelaer/reinforcement-learning-an-introduction-part-2-4-46a1491a2451

5: https://de.mathworks.com/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html

6: https://www.sciencebuddies.org/science-fair-projects/project-ideas/Robotics_p047/robotics/rubber-band-jumping-robot

7: https://rebrickable.com/mocs/MOC-102576/2in1/omnidirectional-walker/#details

8: Lopez, Marcela, and Mahdi Haghshenas-Jaryani. “A Muscle-Driven Mechanism for Locomotion of Snake-Robots.” *Automation* 3, no. 1 (December 31, 2021): 1–26. https://doi.org/10.3390/automation3010001.

9 : https://medium.com/@cedric.vandelaer/reinforcement-learning-an-introduction-part-2-4-46a1491a2451

10: https://akotorque.com/resources/torque-101/

11: https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html