64-424
Intelligent Robotics

https://tams.informatik.uni-hamburg.de/lectures/2019ws/vorlesung/ir

Marc Bestmann / Michael Görner / Jianwei Zhang

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics

Technical Aspects of Multimodal Systems

Winterterm 2019/2020

Outline

1. Machine Learning in Robotics

Machine Learning in Robotics

- Machine Learning
  - non-linear structure extraction from data
  - robust to (sensor) noise
  - requires big data sources
- Intelligent Robotics
  - interprets complex sensor data
  - requires expert roboticists
  - data acquisition in the real world
  - "intelligent behavior" is no well-defined function

©Andrew Ng @ Coursera

Both fields can support each other, but come with their own challenges

Machine Learning

Machine Learning is a field of computer science that gives computers the ability to perform tasks without being explicitly programmed.

ML addresses the problem of optimizing parameters θ of a parameterized family of functions f_θ, such that an error function $E_f(\theta)$ is minimized.

This includes ...

- the definition of the space of permissible functions f_θ
- the definition of the error function E_f
- appropriate optimization methods

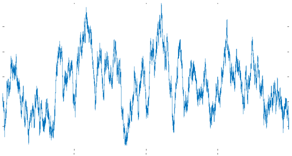
Function Optimization

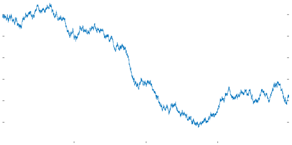
The relevant optimization problem is defined as estimating

$$\theta^* = \underset{\theta}{argmin}(E_f(\theta))$$

- any optimization method can be applied
- most practical methods exploit the gradient ∇E_f
- this is computed
  - analytically
  - by automatic differentiation
  - stocastically (particle techniques)

- smooth(er) gradients allow better optimization





Define the problem

- intelligent robots should be able to perform everyday activities
- most commonsense concepts are hard to formulate in function notation
- most tasks/goals are hard to define with reasonable gradients

The common approach:

- Instead of trying to solve everything as one problem, many things are programmed and a restricted, but crucial, component is learnt
  - use traditional robotics to solve what is hard to learn
  - use ML to solve what is hard to program

Define the function

Here are some classes of learnable functions:

- label-classification of sensory stimuli
  - understand what is there
- grade quality of candidates
  - predict probability of success
- real-world dynamics for simulation
  - approximate physics instead of modelling
- imitate functions from different inputs (*reparameterization*)
  - compute on RGB images instead of $SE(3)$
- traditional functions from Reinforcement Learning: $Q(s, a)$, $V(s)$, $A(s, a)$, $\pi(s)$
  - choose domains for states (or observations) s and actions a

Image Classification / Class Prediction

...is probably the best-understood subfield of applied ML


[1]

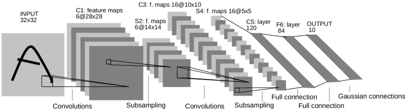
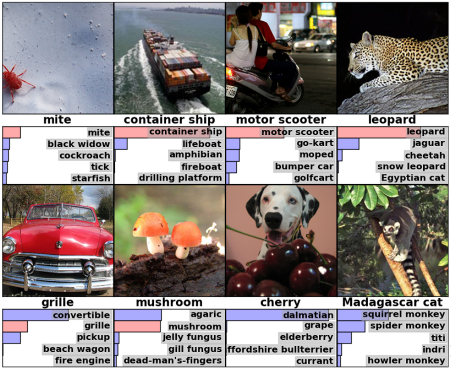
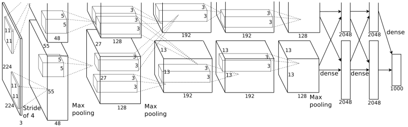


Image Classification / Class Prediction

. . . is probably the best-understood subfield of applied ML

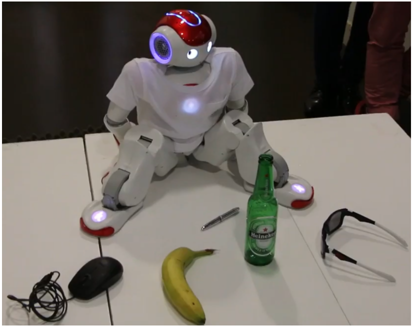




[2]

Image Classification - How to apply it?

- Classification yields one class label per image
- Robots generate camera streams that can be classified



2016 Social Robotics Hackathon

Video

Video

Image Classification - How to apply it?

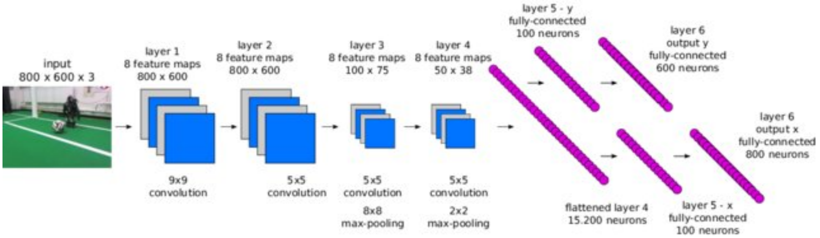
But . . .

- how should the robot respond if it observes the label “table”, “floor”, “human”, or “blue cylinder”?
- useful if the robot *already knows the object’s position*
  e.g. preceding segmentation, prior knowledge

- Even if a label is *not* selected, it might be accurate
- Robot camera streams do not have capture bias, so the object is usually *not* in the image center
- For most applications *localization* is required

Image Localization - Object Coordinates

- We can train networks to directly get the X,Y image coordinates of an object
- Typically using first convolution layers and then some fully connected ones
- This gives us no information of the size

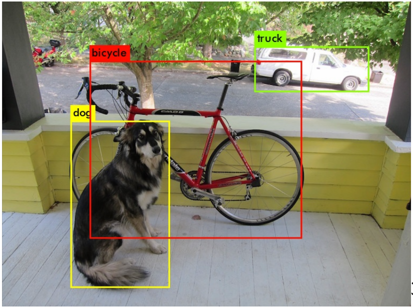


Speck et al., Ball Localization for Robocup Soccer Using Convolutional Neural Networks, RoboCup 2016

Video

Video
https://www.youtube.com/watch?v=buPWpBkR4aU&t=199
(3:20-)

Image Localization - Bounding Boxes



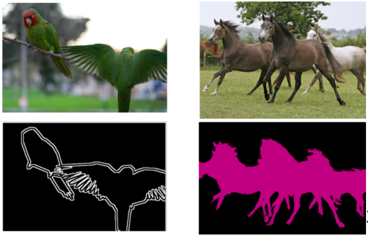
3

- box-localization in image space
- much progress with R-CNN, Fast R-CNN, Faster R-CNN, YOLO
- bounding boxes do not describe object shape
  - sufficient to look towards a face
  - insufficient to pull a door handle

Video

Video
https://www.youtube.com/watch?v=tZnBZsUAVqs&t=7s (0:07 - 1:00)

Image Localization - Pixel Labeling

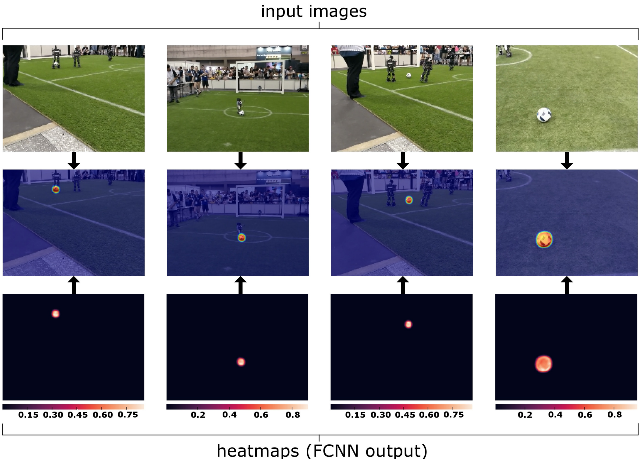

[3]


[4]

- upsampled conv-net output
- or masks for regions
- too much information
  - object boundaries are usually not pixel-accurate
- too few information

- needs further processing to get coordinates
- a pixel-mask of a known mug does not show the orientation of its handle
- interaction often requires *6D pose estimation*

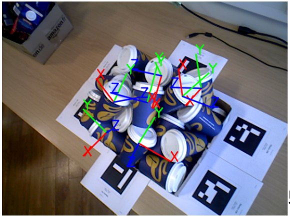
Image Localization - Pixel Labeling - Example



Daniel Speck, Marc Bestmann, Pablo Barros: Towards Real-Time Ball Localizationusing CNNs, 2018

Object Pose Estimation

- need to estimate full pose of the observed object
- accurate estimation remains challenging
- successful pipelines often combine
  - visual learning
  - feature matching
  - pose tracking
  - local model fitting
- there are many impressive demos but no silver bullet



5

6

Video

Example video
https://www.youtube.com/watch?v=yVGViBqWtBI

1. Machine Learning in Robotics

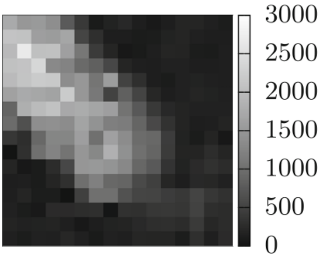
Tactile Classification - Task

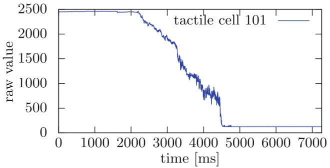
- **Task**: detect slippage of held objects online
- differentiate between rotational & translational slippage
- classification for other modalities

7

Tactile Classification - Method

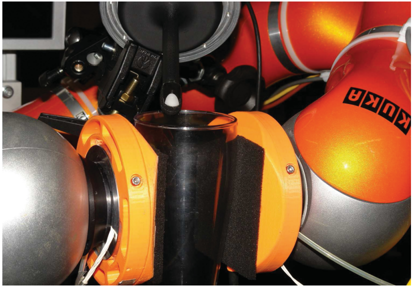
- measure tactile arrays, 16x16 taxels at 1.9kHz
- crop to center 12x12 taxels
- consider 64ms windows
- compute short time Fourier transforms
- **Input**: 12x12 images with 32 amplitude channels
- convolutional NN with 3-5 layers
- **Output**: $\{stable, translation, rotation\}$
- test accuracy 97.89% at 125 Hz





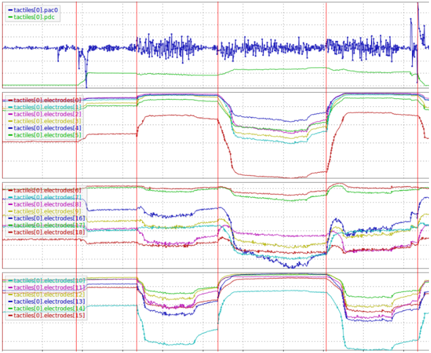
Tactile Classification - Data Acquisition

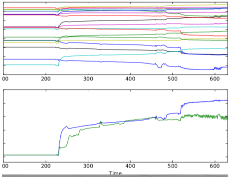
- user places object between arms, defines rotational / translational slippage
- controllers relax force until slippage occurs
- detect slippage with an orthogonal sensor
- episode runs until no contact is detected
- data augmentation: rotate inputs to account for gravity

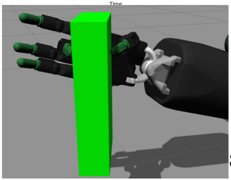


Tactile Simulation - Task

- **Task**: simulate raw readings of a complex tactile sensor



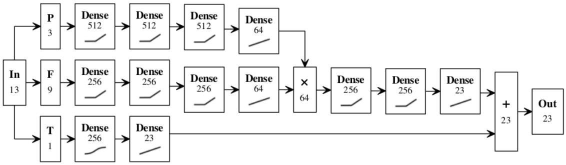




8

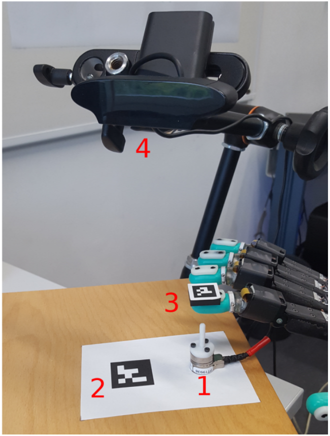
Tactile Simulation - Method

- restrict to single-contact cases
- **Input**: [contact point, 3x force vectors, sensor temperature]
- input is available in physics simulators at each time step
- **Output**: 23 sensor channels



Tactile Simulation - Data Acquisition

- fiducial-based tracking of sensor and a contact pin
- contact pin mounted on 6-axis force-torque sensor
- compensate for visual inaccuracies by optimizing setup model based on sensor data
- record 300.000 tactile readings with varying contacts



1. Machine Learning in Robotics

Grasping Objects

- Picking up objects is a common task for robots
- It is unclear where an object should be grasped
- This is influenced by
  - What kind of gripper does the robot have?
  - How stable would the resulting grasp be?
  - Has the object handle-like structures?
  - Why is the object being picked up?
- If all objects are modeled
  - Feasible grasps can be annotated in models
  - The problem becomes pose estimation
  - If the pose is known, grasps can be looked up
- Most research focuses on 2-finger parallel grippers

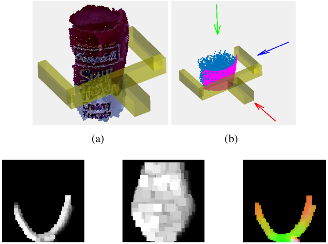
Grasp Pose Generator - Task

- **Task**: perform successful 6dof grasps from cluttered scenes of unknown objects
- generate “good” grasp points
- based on RGB-D camera
- learn scoring function for candidate grasps
- 93% grasp success in experiments

[9]

Grasp Pose Generator - Method

- sample 6dof grasps, s.t.
  - gripper encloses at least one point
  - gripper is not in collision with the point cloud
- extract multi-image representation from enclosed point cloud and observed/occluded volume



(a) (b) (c) (d) (e)

- **Input**: 15 constructed images
- **Output**: grasp probability
- choose best-ranking samples for execution

Grasp Pose Generator - Data Acquisition

- based on object dataset pairing object views & 55 object meshes
- generate random grasp candidates for views
- label based on geometric antipodal force-closure criterium
- simulated data (3DNET) produced inferior results due to simulation discrepancy

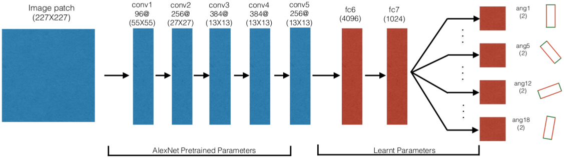
Learning Grasps - Supersizing Self-Supervision

- **Task**: predict 3dof grasp poses (x, y, φ)
- first paper to collect 500h experience for grasping
- collection procedure:
  - Move gripper above sampled ROI
  - Sample grasp point and orientation $\theta \in \{0°, 10°, \ldots 170°\}$
  - Pick object and check gripper force sensor for success


[10]

- 66% success rate on test objects
  73% for known objects

Learning Grasps - Supersizing Self-Supervision - Method



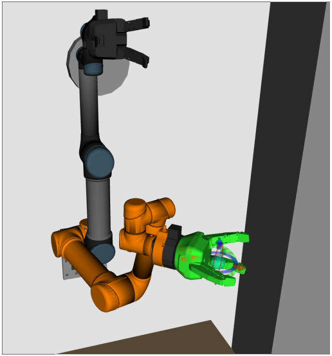
10

- adapt AlexNet and retrain last layers
- output softmax layer with 18 possible angles φ
- at runtime,
  - sample image patches on ROI
  - evaluate them according to network
  - execute highest-scoring one

Video

Video

Inverse Kinematics



example IK solution for a UR5 arm

- IK is one of the oldest problems in robotics
- given a robot kinematic design, find a function that maps the Cartesian workspace of the tool frame to joint configurations
- a single pose often maps to multiple joint configurations
- for many existing robot arms fast analytical solutions exist

Inverse Kinematics (2)

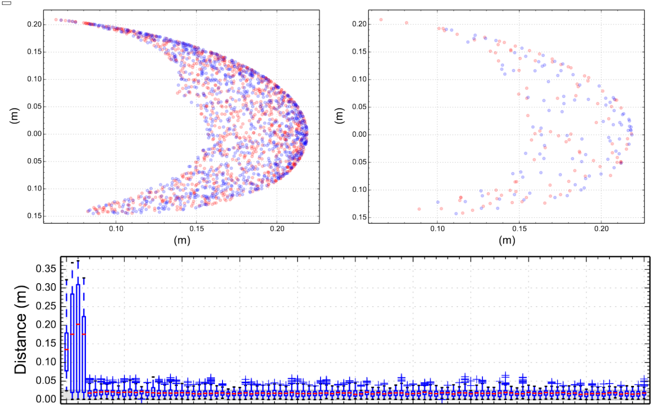
- "well-defined" function
- easy to generate samples
- but ...
  - applications require very high accuracy solutions position error $< 10^{-5}$m
  - multiple solutions disrupt training gradient

- training a network for 6dof can succeed
- demonstrated for continuous sub-workspaces with current joint state as input
- online optimization performs better

[11]

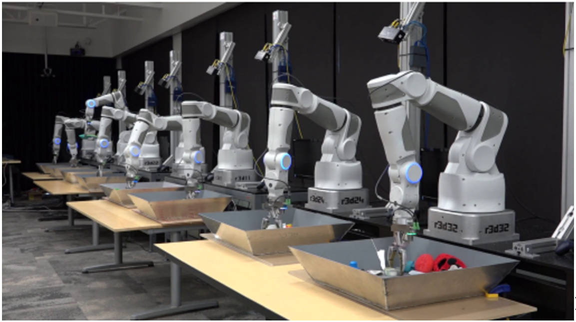
IK to absurdum

- Learning IK for 2dof with 2.5cm accuracy



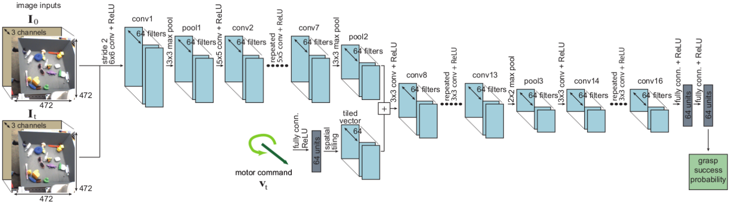
1. Machine Learning in Robotics

Learning Hand-Eye Coordination - Task


12

- **Task**: pick objects from cluttered container
- learn "*Q*-like" function by supervised learning

Learning Hand-Eye Coordination - Method



- **Input**:
  - ***o*** - 2 RGB images of current and first state
  - ***a*** - 3D Cartesian force vector & a twist φ
- **Output**: eventual grasp success ℓ
- the network is trained to predict the eventual grasp success ℓ of the attempt
- online controller runs stochastic search to find action with good prediction

Learning Hand-Eye Coordination - Data Acquisition

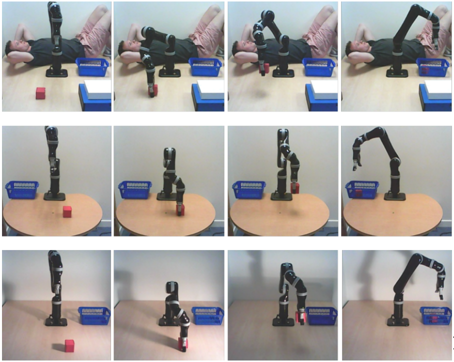
- 6-14 robots running for 2 months
- heuristics to automatically lift arm up if “not promising”
- grasp if at least 90% as successful as move
- servoing with 2-5Hz
- self-supervised exploration for 800.000 grasp attempts
- first 50% random commands (10-30% success already)

A tremendous achievement that works because the learning task is designed to be *as simple as possible*, while it still defines the core behavior.

Video

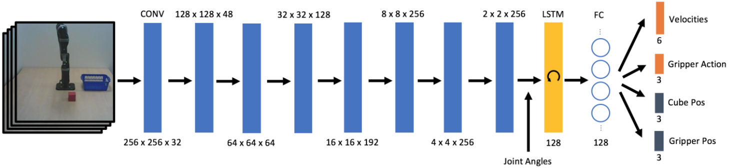
Video

End-to-End Visuomotor Control - Task


[12]

- **Task**: move red block into blue basket
- learn visuomotor policy π

End-to-End Visuomotor Control - Method



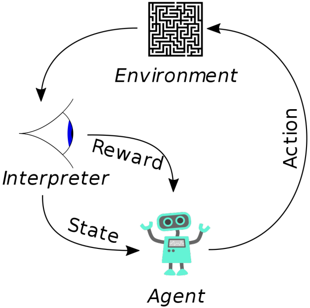
- **Input**: $\boldsymbol{o}$
  - 4 last RGB images
  - current joint angles
- **Output**: $\boldsymbol{a}$
  - 6 joint velocities
  - softmax gripper action
  - auxiliary outputs

- learn to imitate straight-line motion planner
- originally sim2real application with domain randomization
- later demonstrated to be trainable in reality (with position targets)

1. Machine Learning in Robotics

Reinforcement Learning

- Currently a hype topic in ML
  - Especially regarding robotics/motion
- Advances due to more investment
  - Most notably Google/OpenAI
- and common environments
  - OpenAI Gym
  - RoboSchool
  - PyBullet
- and baseline implementations
  - Open AI baselines
  - INRIA stable_baselines



https://en.wikipedia.org/wiki/Reinforcement_learning

Vanilla PPO

Video:
https://www.youtube.com/watch?v=hx_bgoTF7bs
Live Demo Roboschool

N. Heess et al, Emergence of Locomotion Behaviours in Rich Environments, 2017

Vanilla PPO - What is Learned

- Action: joint efforts
  - Planar walker: 9 DOF
  - Quadruped: 12 DOF
  - Humanoid: 28 DOF
  - Joints box constrained
- Observation proprioceptive
  - Joint angles and velocities
  - Velocimeter
  - Accelerometer
  - Gyroscope
  - Contact sensors at feet and leg
- Observation exteroceptive
  - Position in relation to center of the track
  - Profile of the terrain ahead

Vanilla PPO - What is Learned (cont.)

- Policy: two networks
  - Only proprioceptive observations
  - Only exteroceptive observations
  - Type of network not clear (probably MLP)
  - Somehow choose action together
- Reward
  - Forward velocity
  - Penalization for torques
  - Stay at center of track

N. Heess et al, Emergence of Locomotion Behaviours in Rich Environments, 2017

Vanilla PPO - How is it Trained

- Mujoco simulation
  - Physical parameters unknown
  - Rate of policy unknown
- PPO - Proximal Policy Optimization
  - Simplified version of TRPO - Trust Region Policy Optimization
  - Current policy is used to choose actions
  - After an episode, advantages of those actions are computed
  - The policy is updated so that good actions become more propable and vice versa
  - Updates are clipped to prevent the policy from leaving the area where it can explore senseful
- Distribution through workers
  - Each worker collects data and computes gradients
  - Batch results are processed by chief
  - New policy is distributed to workers

Deep Mimic

Video:
https://www.youtube.com/watch?v=vppFvq2quQ0

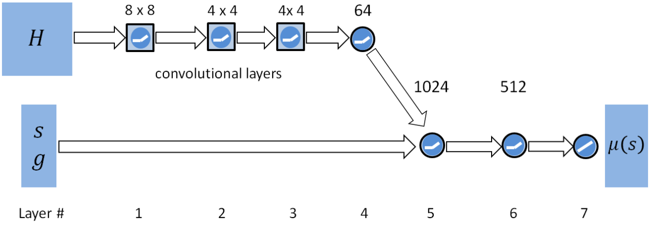
X. Peng et al., DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills, 2018

Deep Mimic - What is Learned

- Action: joint position
  - PD controllers compute effort
- "Observations" (States)
  - Relative pose of links to pelvis
  - Linear and angular velocity of links
  - Phase $\in [0, 1]$
  - Goal g
    - Target heading (walk)
    - Target position (kick, throw)
- Reward
  - $r_t = \omega^I r^I + \omega^G r^G$
  - Closeness to mocap and goal
  - $r^I = w^P r_t^P + w^v r_t^v + w^e r_t^e + w^c r_t^c$
  - Reward based on difference in joint position/velocity, end-effector position, CoM position

Deep Mimic - What is Learned (cont.)

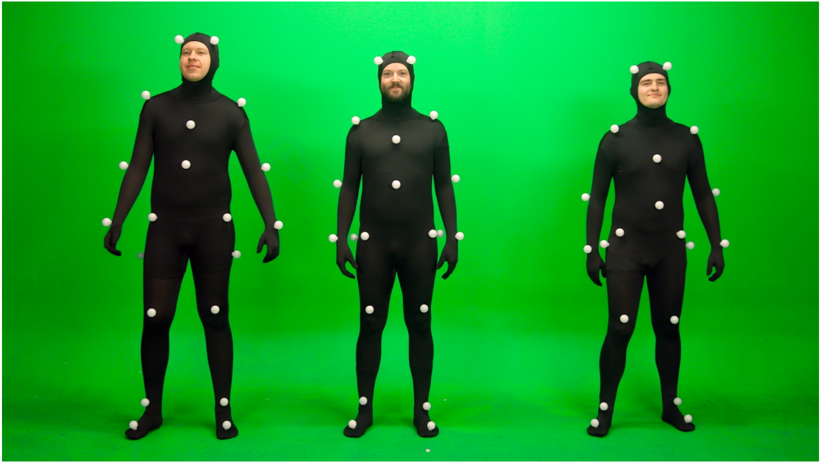
- Policy: single network
  - Input goal and state
  - 2 hidden layer with 1024 and 512 neurons
  - ReLU activation
  - Additional CNN for height map



Deep Mimic - How is it Trained

- Mujoco simulation
  - Physics parameter unknown
  - 30 Hz
- Initial state distribution
  - "Man muss immer umkehren" - Jacobi
  - Easier to learn starting from the back (backplay)
  - Reward clearer when near goal
  - Choosing initial state simple with mocap
- Early termination
  - Terminate episode if condition is reached
  - Classic for walking: head is below certain height
  - Reward for episode is set to zero
  - Further shapes reward function
  - Biases the data distribution to samples which are more favorable

Motion capture - small excursus



Motion capture - small excursus (cont.)

- Different ways to get the data
  - Infra red reflectors
  - LEDs blinking with different frequencies
  - IMUs on all links
  - Magnetic field based (hall sensor)
  - Exoskeleton measuring angles
- Pro
  - Faster learning
  - Less exploits of glitches
  - (Maybe) more useful on actual robot

Motion capture - small excursus (cont.)

- Contra
  - Expensive
  - A lot of work
  - Difficult to get data from animals, e.g. a tiger
  - You look kind of stupid while recording
  - Need to find a student which can do a round house kick
  - Need to bring student into hospital after failed round house kick

There has to be a better way!

Skills from Video

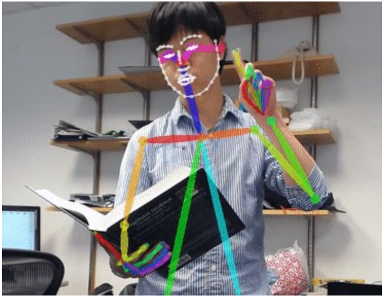
Video:
https://www.youtube.com/watch?v=4Qg5I5vhX7Q

SFV - What is Learned

- Pose estimation
  - 2D and 3D
  - 2 different estimators OpenPose and Human Mesh Recovery
- Motion reconstruction
  - Find optimal motion from single poses
  - Enforce temporal consistency to reduce jitter and glitches
- Learning of motion is similar to DeepMimic, just without goal
- $r = w^P r_t^P + w^v r_t^v + w^e r_t^e + w^c r_t^c$

SFV - How is it Trained

- Pose estimators
  - Supervised learning on single images
- Motion part
  - Similar to DeepMimic



http://www.cs.cmu.edu/ yaser/

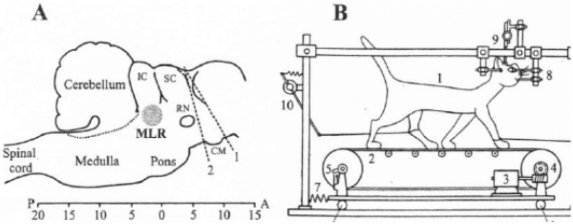
Do We Walk With Our Brain?

- How do humans compute their walking?
- Chickens can run without head
- Legend of Störtebecker

http://www.neurologie.usz.ch/ueber-die-klinik/veranstaltungen/Documents/7_hirnstimulation.pdf

Central Pattern Generators

- Biological neural circuits in the spline
- Generate rhythmic output after being activated
- Used by humans for walking, breathing, swallowing, ...
- Models of this can be implemented for robots



Central Pattern Generator, Mark L. Latash et al., Biomechanics and Motor Control, pp.157-174

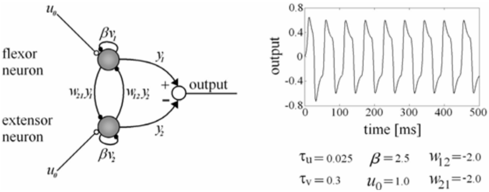
Video

Trigger warning!

Video of experiment with real cat

Central Pattern Generators

- Flexor and extensor neuron
- Activated with tonic (non-rhythmic) signal
- Different patterns with different weights



Central Pattern Generators for Gait Generation in Bipedal Robots, Almir Heralic et al., Humanoid Robots, New Developments, 2007

CPG - How is it learned

- It is not!
- Weights are hand crafted
- Learning would be possible either by direct parameter learning or RL

Evolutionary Approach

Video
https://www.youtube.com/watch?v=pgaEE27nsQw

Evolutionary Approach - What is Learned

- Muscle structure of the robot
- Parameters of FSM for leg state
- Target poses
- Force applied in relation to feedback
- Initial pose

Subject	**Parameters**
Muscle physiology	3–30 *
Muscle geometry	12–39 *
State transition	3
Target features	14
Feedback control	14–63 *
Initial character state	6

Geijtenbeek, Thomas, Michiel Van De Panne, and A. Frank Van Der Stappen. "Flexible muscle-based locomotion for bipedal creatures." ACM Transactions on Graphics (TOG) 32.6 (2013): 206.

Evolutionary Approach - How is it Trained

- Evolution approaches in general
  - Generate initial random population of parameter sets
  - Loop
    - Evaluate individuals based on fitness function
    - Pick best
    - Recombination / mutation
- Covariance matrix adaptation evolution strategy (CMA-ES)
  - Pairwise dependency between parameters is represented by covariance matrix
  - This matrix is updated to increase fitness
  - Good for ill-conditioned functions

Simulation Downsides - The Reality Gap

- Difference between simulation and reality
- Wrong models
  - Mass, inertia, size
  - Sensor noise non Gaussian
  - Actuator properties not correct
  - Change over time
  - No static values
- Friction / contact
- Soft bodies
- Environment model not correct
  - Changing lighting conditions
  - Cluttered background
  - Non static background

Simulation vs. Reality (cont.)

- Simulation physics not correct
  - Discrete approximation of continuous system
  - Simplifications due to performance bounds
  - Glitches

Bridging the Reality Gap

- What can we do?
- Improving simulation accuracy (smaller step size)
  - Not enough to bridge reality gap
- Adding sensor noise
  - Noise is not perfectly Gaussian
  - Needs noise model, which can have errors
- Domain randomization
  - Currently the most used approach
  - Simulated variability in training time to make model generalize
  - Implementation depends on the scenario

J. Tobin et al., Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World, 2017

Learning Dexterity

Video
https://www.youtube.com/watch?v=jwSbzNHGflM&t=1s

Learning Dexterity - What is Learned

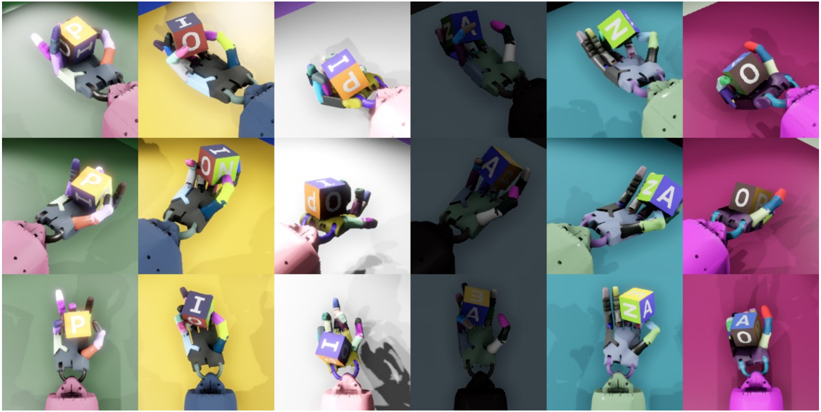
- CNN for object pose detection
  - Based on three camera inputs
- LSTM for finger actions given finger and object pose
- Both networks are concatinated

M. Andrychowicz et al., Learning Dexterous In-Hand Manipulation, 2018

Learning Dexterity - How is it Trained

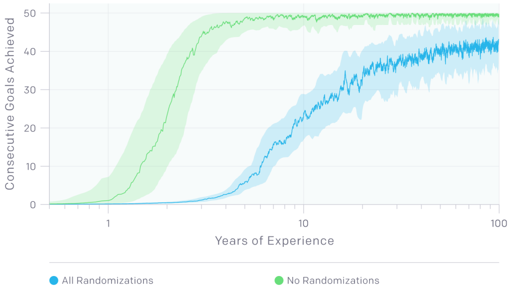
- PPO
- Domain randomization
  - Object dimensions
  - Object and finger masses
  - Surface friction coefficients
  - Robot joint damping coefficients
  - Actuator controller P term (proportional gain)
  - Joint limits
  - Gravity vector
  - Colors in simulation

Learning Dexterity - Domain Randomization



Learning Dexterity - Impact of Domain Randomization

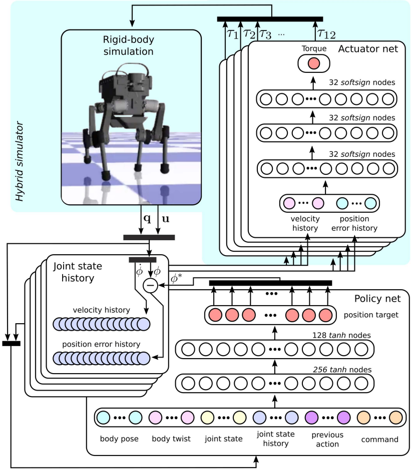
- Median number of successes
  - Without: 0
  - With: 11.5
- Training simulated time
  - Without: 3 years
  - With: 100 years



Learning Dynamic Skills

Video
https://www.youtube.com/watch?v=aTDkYFZFWug

Learning Dynamic Skills - Overview



Learning Dynamic Skills - What is Learned

- Policy Network
  - MLP 2 hidden layers 256, 128 nodes
  - Joint angles, velocities
  - Joint state history
  - Body height estimation (filtered forward kinematics)
  - Body pose, twist (IMU)
  - Previous action
  - Command
- Actuator Network
  - MLP 3 hidden layers with 32 nodes
  - Velocity history
  - Position error history

J. Hwangbo et al., Learning agile and dynamic motor skills for legged robots, Science Robotics 2018

Learning Dynamic Skills - How is it Trained

- Randomized simulator model
  - Links masses
  - CoM positions
  - Joint positions
- Policy Network
  - RL with TRPO
- Actuator Network
  - Supervised learning
  - Data collection on robot with simple walk algorithm
  - Joint Position error, Velocity, and Torque

More Information

- We only had a quick overview, here are some further information
- ML Foundations
  - Machine learning lecture next semester
  - Arxiv Insights - Youtube channel
  - R. Sutton - Reinforcement Learning, an Introduction (free)
  - Berkeley - Deep RL http://rail.eecs.berkeley.edu/deeprlcourse/
- Current advances
  - Open AI blog - https://blog.openai.com/
  - reddit.com/r/MachineLearning
  - CORL conference (open access)
- If you have some good sources, tell me!

Summary

- Intelligent Robotics encompasses many domains that can profit from ML applications, including
  - images
  - tactile data
  - inverse kinematics
  - grasp estimation
  - behavior policies
  - ...
- (uncorrelated) data is scarce, because it is collected at runtime
- to train in practice tasks require many simplifications and resources or simulation
- successful approaches reduce the learning problem
- the resulting modules can be very robust and successful

Discussion

What would you teach a robot?

Masterproject
Thesis

[allowframebreaks] Literature list

[1] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner.
Gradient-based learning applied to document recognition.
Proceedings of the IEEE, 86(11):2278–2324, 1998.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton.
Imagenet classification with deep convolutional neural networks.
In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[3] Martin Meier, Florian Patzelt, Robert Haschke, and Helge J Ritter.
Tactile convolutional networks for online slip and rotation detection.
In *International Conference on Artificial Neural Networks*, pages 12–19. Springer, 2016.

[4] Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool.
Traffic sign recognition—how far are we from the solution?
In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013.