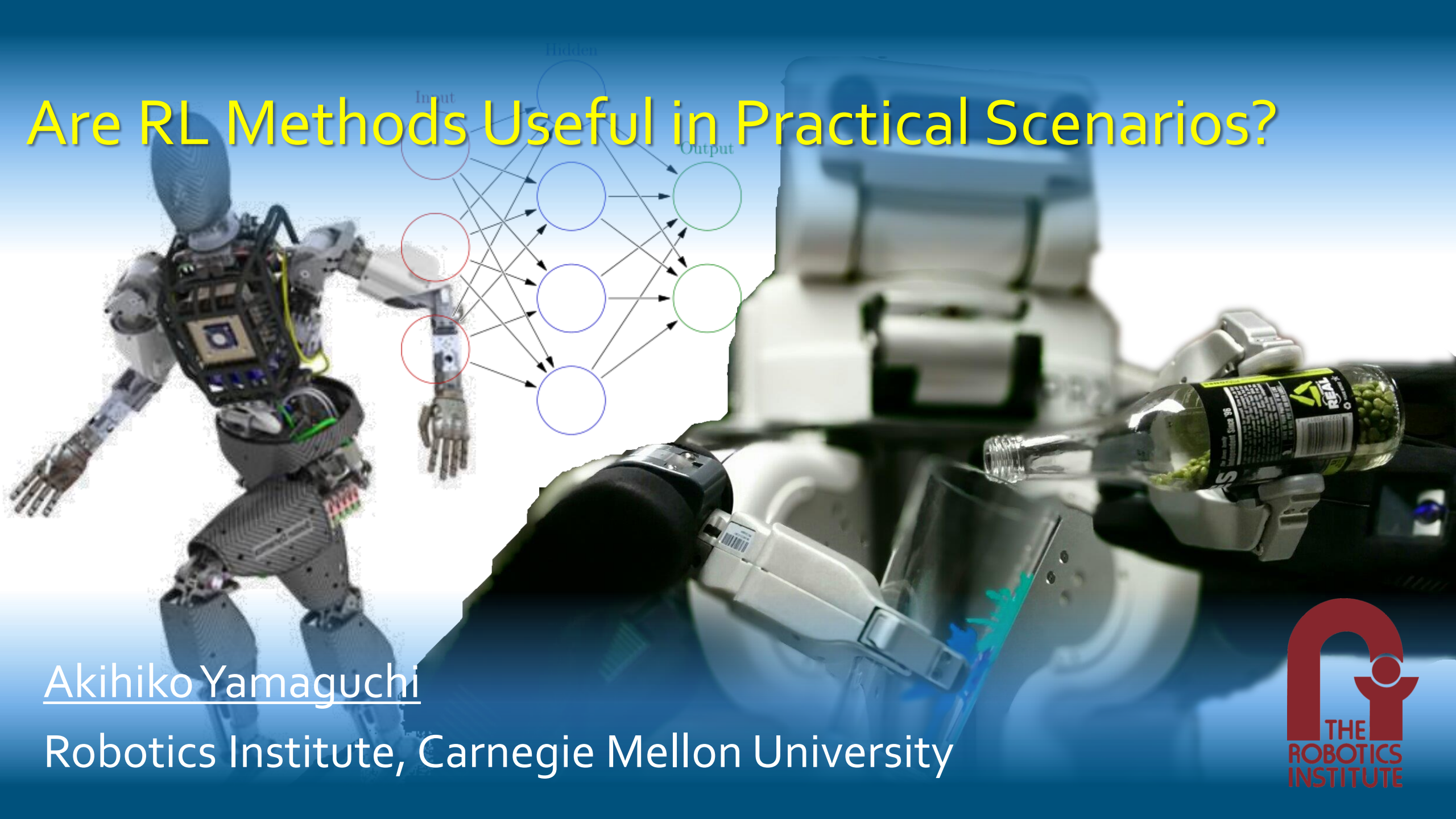
Are RL Methods Useful in Practical Scenarios?



Akihiko Yamaguchi

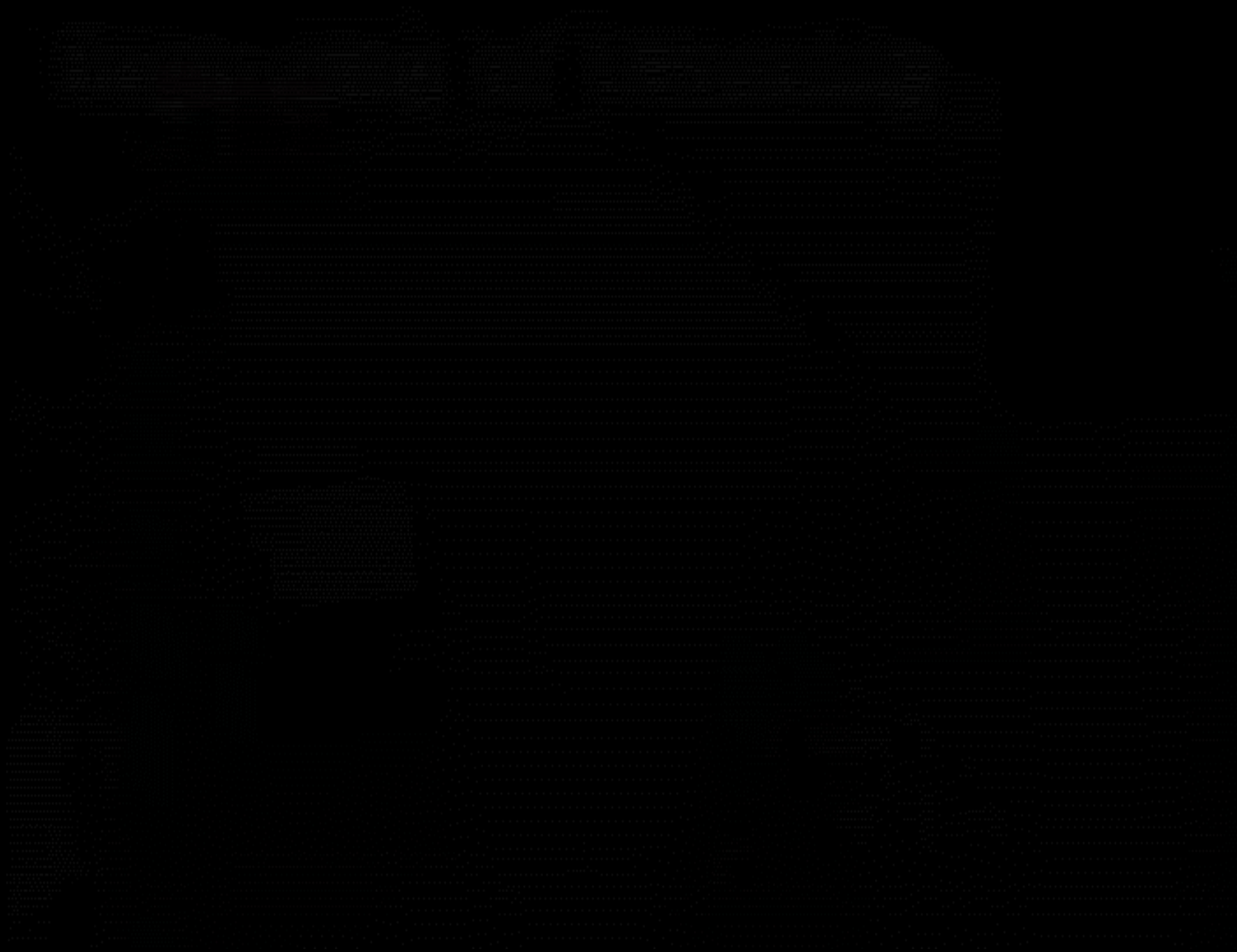
Robotics Institute, Carnegie Mellon University

THE ROBOTICS INSTITUTE

Who am I?

- …-2006: Kyoto U (Prof. T. Matsuyama)
- 2006-2008: NAIST, ATR CNS (Prof. M. Kawato)
- 2008-2011: NAIST, Robotics Lab (Prof. Ogasawara)
- 2011-2015: NAIST, Assistant Prof.
- 2014-current: CMU, Visitor/PostDoc working w Prof. Chris G. Atkeson
- Research interests:
  Robot learning, machine learning, robotics, artificial intelligence, motion planning, manipulation, …
- http://akihikoy.net/

Yamaguchi et al. "DCOB: Action space for reinforcement learning of high DoF robots", Autonomous Robots, 2013
https://www.youtube.com/playlist?list=PL41MvLpqzOg8FFoxekWT9NXCdjzN_8PUS



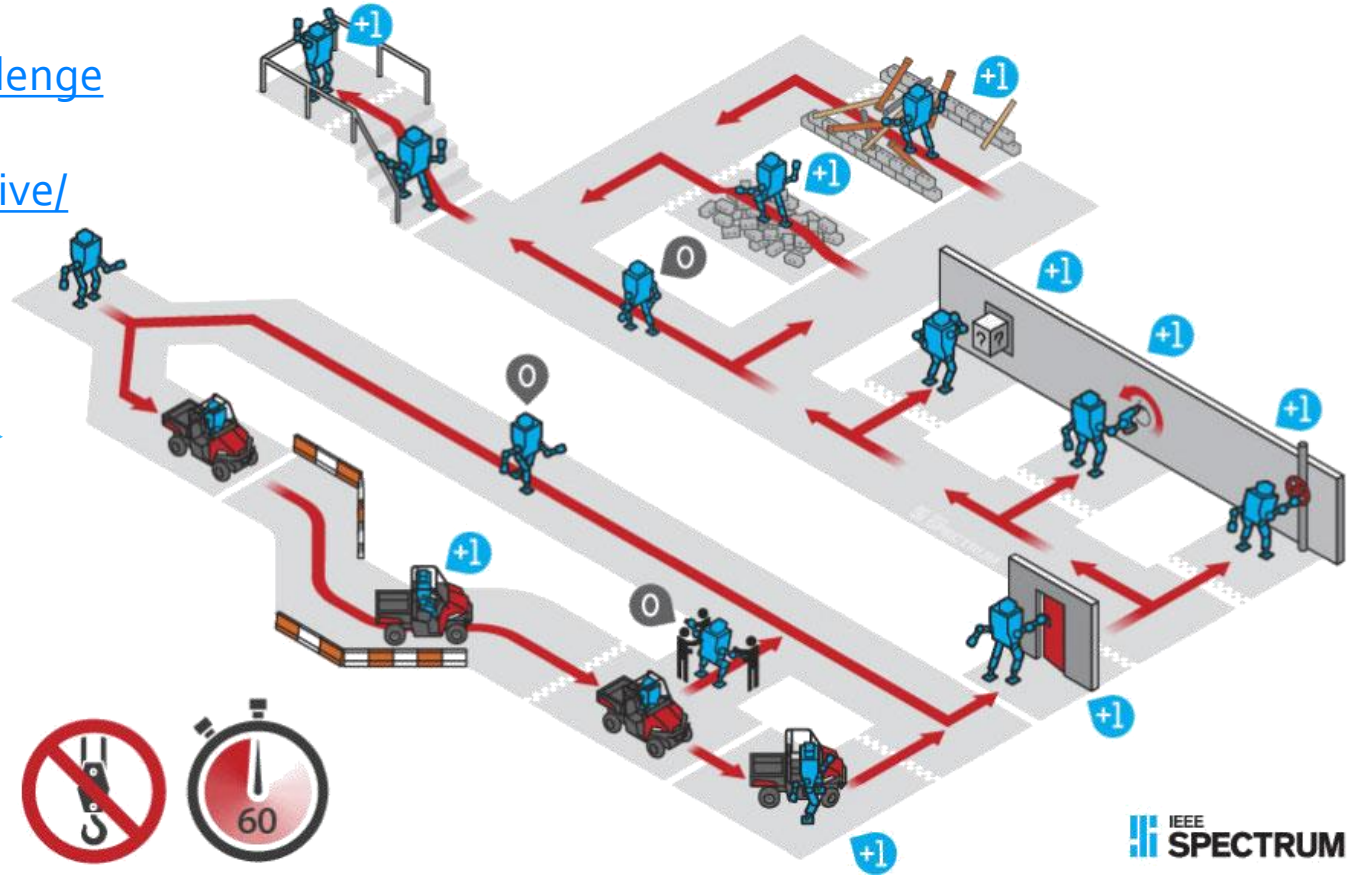
Yamaguchi et al. "DCOB: Action space for reinforcement learning of high DoF robots", Autonomous Robots, 2013
https://www.youtube.com/playlist?list=PL41MvLpqzOg8FFoxekWT9NXCdjzN_8PUS

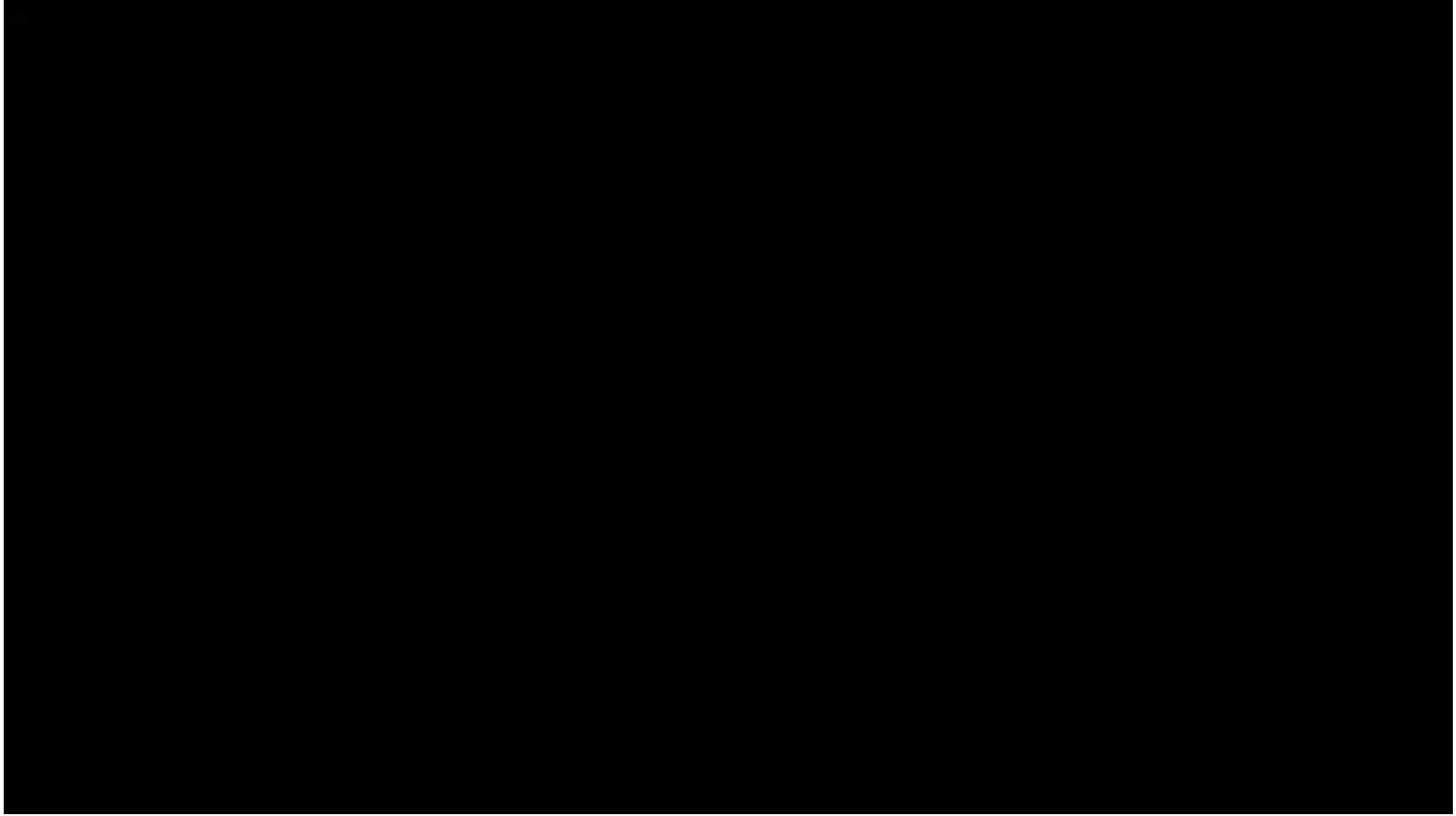
https://www.youtube.com/playlist?list=PL41MvLpqzOg8FFoxekWT9NXCdjzN_8PUS

DARPA Robotics Challenge (DRC)

- ✓ DARPA Robotics Challenge Finals: Rules and Course http://spectrum.ieee.org/automaton/robotics/humanoids/drc-finals-course
- ✓ DARPA Robotics Challenge (DRC) http://www.darpa.mil/program/darpa-robotics-challenge
- ✓ DRC Trials http://archive.darpa.mil/roboticschallengetrialsarchive/
- ✓ DRC Finals http://www.theroboticschallenge.org/

Operator room





WPI-CMU DRC Finals Day 1: Time Lapse X20
https://www.youtube.com/watch?v=AvyGzqwOPSM

[A Compilation of Robots Falling Down at the DARPA Robotics Challenge](https://www.youtube.com/watch?v=goTaYhjpOfo)
https://www.youtube.com/watch?v=goTaYhjpOfo

A technology behind (AIST-NEDO)

Shin'ichiro Nakaoka, Mitsuharu Morisawa, Kenji Kaneko, Shuuji Kajita and Fumio Kanehiro, "Development of an Indirect-type Teleoperation Interface for Biped Humanoid Robots", 2014
http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7028105

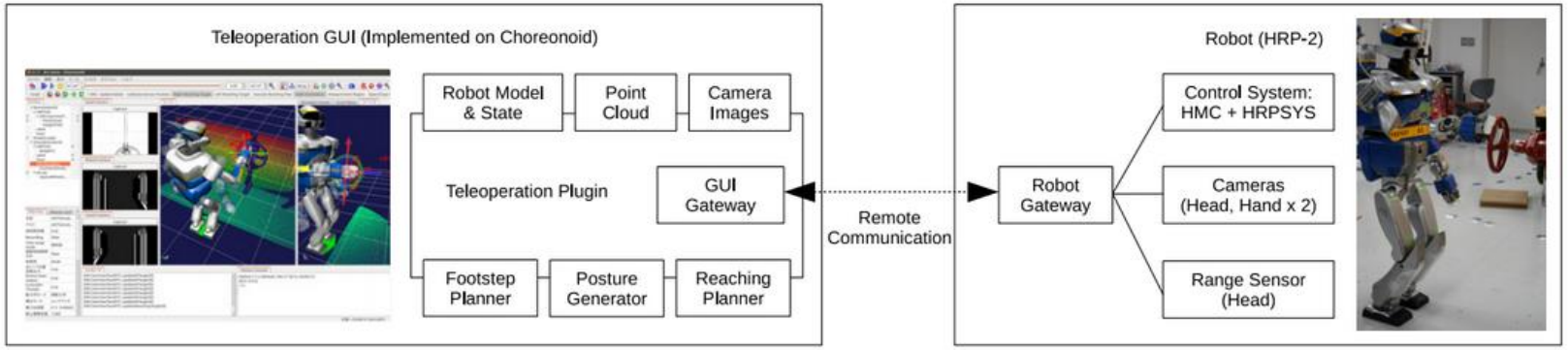
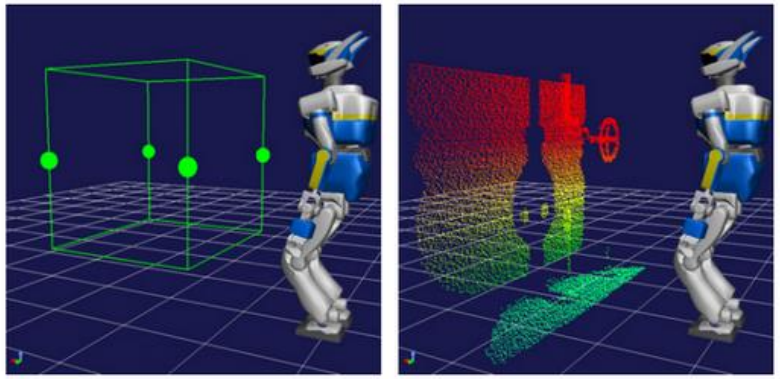


Fig. 1. Composition of the teleoperation system

the GUI when they are requested, and handles the internal controller as per the commands from the GUI. In this way, all the communications between the GUI and the robot are managed by the gateways, and only the information needed at the time is transmitted. This design makes it easy to manage the timing and the amount of the communications, which is important especially when sufficient bandwidth for communication is not ensured.

The robot used in our system is the humanoid robot HRP-2 [11]. The original six-degree-of-freedom (DOF) arms are extended to be seven-DOF, and the grippers are also modified



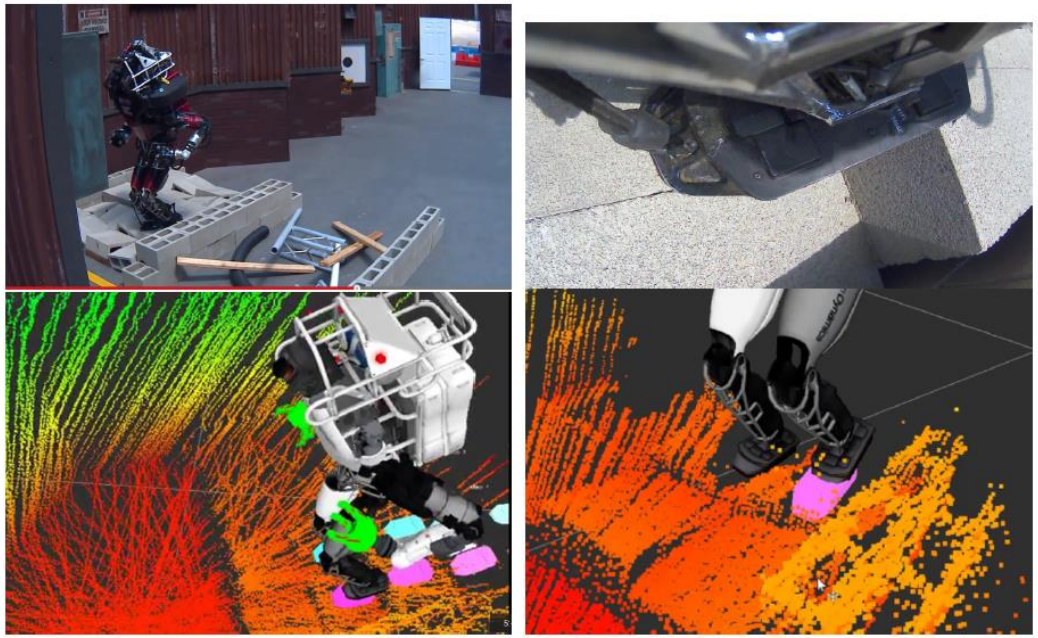
(a) (b)

A technology behind (WPI-CMU)

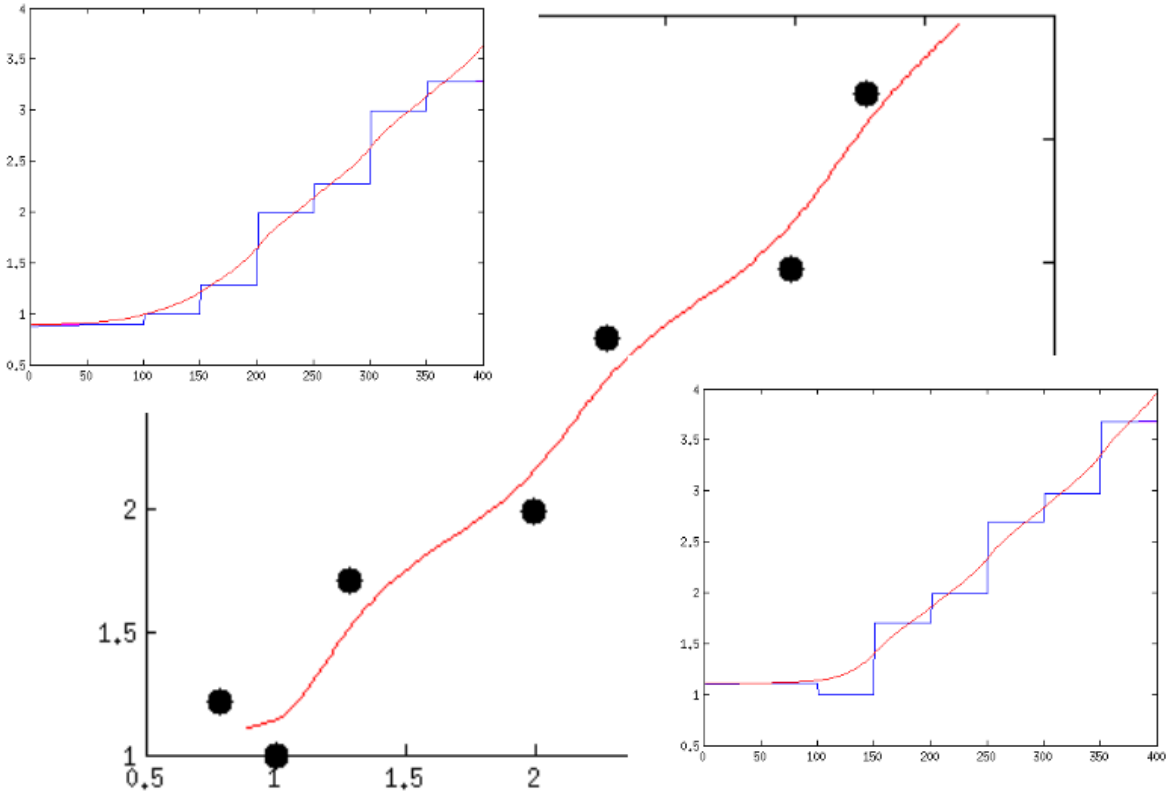
- Did well (14/16 points over 2 days, drill)
- Did not fall
- Did not require physical human intervention

Walking control:

- Detect rough terrain with LRF
- Multi-level optimization
  - Footstep optimization
  - Trajectory optimization
  - Optimization-based inverse dynamics



LIPM Trajectory Optimization



Team WPI-CMU: Darpa Robotics Challenge
http://www.cs.cmu.edu/~cga/drc/
(cmu-drc-final-public.zip, dw1.pptx)

DRC and AI/RL/ML

- How much autonomy the robots had?
  - At least, they could compensate the delay and the limited bandwidth between the operator rooms
  - Many teams used motion planning
  - But operators' directions were important (where to go, what to do e.g. grasping a valve, roughly pointing a target object e.g. rough valve position)
- How much learning methods were used?
  - Robot learning professionals were there (e.g. C. G. Atkeson, R. Tedrake, P. Kormushev, …)
  - C. G. Atkeson et al.: "NO FALLS, NO RESETS: Reliable Humanoid Behavior in the DARPA Robotics Challenge" http://www.cs.cmu.edu/~cga/drc/paper14.pdf
    "The absence of horizontal force and yaw torque sensing in the Atlas feet limited our ability to avoid foot slip, reduce the risk of falling, and optimize gait using **learning**."
    "**Learning to plan better is hard**. To be computationally feasible for real-time control, a cascade of smaller optimizations is usually favored, which for us and many others eventually boiled down to reasoning about long term goals using a simplified model and a one-step constrained convex optimization that uses the full kinematics and dynamics"
- How many general/versatile AI methods were used?
  - Motion planning (e.g. RRT-like)
  - Optimization algorithms
- Where is RL???
  - C. G. Atkeson: "As far as I know, no rl actually used; Sent from my iPhone"

Why many robots failed?

- Software bug
  - Zero division (AIST)
  - State machine implementation (WPI-CMU)
- Vision failure
  - Error to detect rough terrain (e.g. approx. 4cm error, AIST)
  - Error to detect the valve (JSK JAXON)
    Actually the robot was not grasping the valve, and then its elbow hit the valve
- Hardware trouble
  - Overheat of an arm actuator and stop (WPI-CMU)
- Operator's failure
  - IHMC, CHIMP, MIT, WPI-CMU, …

ref. What Happened at the DARPA Robotics Challenge?
http://www.cs.cmu.edu/~cga/drc/events/

ref. DARPA Robotics Challenge Finals 2015
http://akihikoy.net/notes/?article%2FDRC-finals-2015

Can we improve by RL or ML?

- Potentially to say, YES
- But not easy since many factors are combined
  - Modeling error of robots → ML is useful (Many robot learning methods are focusing on)
  - Overheat of joint actuators and stop → We may be able to learn the overheat model + avoid the overheat by planning
    (Note: better solution may be improving the hardware; cf. JAXON is using liquid-cooling system)
  - Vision error → actually error probabilistic distribution is complicated
  - Programming bug → generally it is complicated to handle
  - …
- Gathering many "failure samples" is difficult
  - Falling down cost is very expensive

Lessons from DRC

- No use of railings, walls, door frame for supporting the body
  - Recently, multi contact planning is a hot topic
    cf.
    - ✓ Humanoids 2015 technical program: http://www.humanoids2015.org/sub/sub03_12.asp
    - ✓ Humanoids 2015 Workshop on Whole-Body Multi-Task Multi-Contact Humanoid Control http://cs.stanford.edu/groups/manips/humanoid2015/index.html
- HRI (operators – robot) is important
  - Operators (professionals) made many errors!
  - Software should be able to detect operators' errors
- Sensor & state estimation are important (more than AI/control?)
  - Add wrist and knee cameras
- Thermal management is important (SCHAFT/JAXON: liquid, Hubo: air, Atlas: electric wrist motor always overheating)
- Design considering error recovery is important

ref. Humanoids 2015 Panel: Lessons Learned, http://www.cs.cmu.edu/~cga/drc/

Another problem: variation

Small variation in DRC
(tasks are pre-defined)
Variation in real world is large
(e.g. pouring)

http://reflectionsintheword.files.wordpress.com/2012/08/pouring-water-into-glass.jpg

http://schools.graniteschools.org/edtech-canderson/files/2013/01/heinz-ketchup-old-bottle.jpg

http://old.post-gazette.com/images2/20021213hosqueeze_230.jpg

http://img.diytrade.com/cdimg/1352823/17809917/0/1292834033/shampoo_bottle_bodywash_bottle.jpg

http://www.nescafe.com/upload/golden_roast_f_711.png

Why is there such a big gap?

- Many problems both in RL/AI/ML and Robotics
  - State-action space is large
  - Variety of constraints
  - Difficulty to obtain samples

  RL/AI/ML researchers are thinking

  - Inverse kinematics issues
  - Modeling contact force
  - Modeling & manipulation of soft objects
  - Multi contact planning:
    - ✓ With whole body control
    - ✓ With soft objects (e.g. sofa)
  - Handling variations
  - …

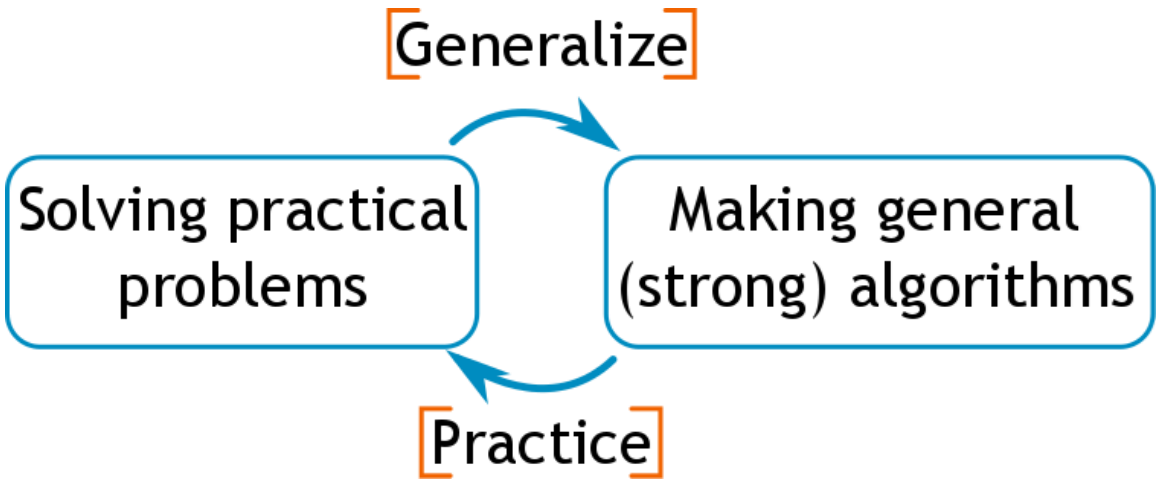
  Robotics researchers are thinking

- A perfect autonomous AI for robots needs to solve *everything above*
- What is a key solution? RL? → We do *not* know
- What we know: there is *no magic*

My philosophy

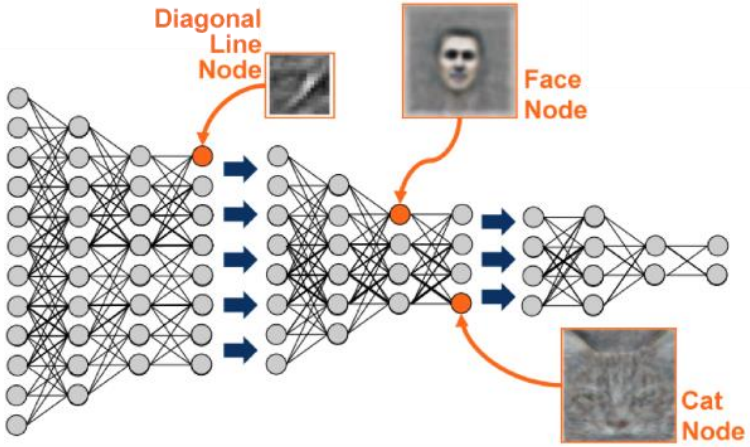
- A standard science approach is important: gathering many examples (solving challenging problems), and generalizing the solutions, until we find a strong AI/RL/ML algorithm
- Why deep learning (DL) was successful?

  - Manipulation of soft objects
  - Multi contact planning
  - Handling variations
  - DRC tasks
  - Pouring, folding towels, opening things, …
  - …



Deep learning

- Neural networks (with deeper? hidden layers)
- Won in many ML competitions; applications: image classification, voice recognition, translation, …
- Still unclear: why deeper is better? (really?)
  - Cf. Deep v.s. shallow:
    Lei Jimmy Ba, Rich Caruana: Do Deep Nets Really Need to be Deep?, NIPS 2014.
- What are the essence of success?
  - Convolution
  - Dropout (probabilistically ignore output of hidden layers) → avoid over-fitting
  - ReLU (Rectified Linear Unit; max(x,0)) was good? / Nonlinear activation funcs
  - LSTM (for RNN)
  - (Pre-training, Auto Encoder → learning technique for deeper nets)
  - Big data; e.g. ImageNet
- What are great things?
  - So far: Image → Feature detection → Neural net/SVM/…
  - DNN (Deep Neural Network): Image → Neural net
  - I.e. designing feature detection is not needed (but still need a technique like convolution layers?)

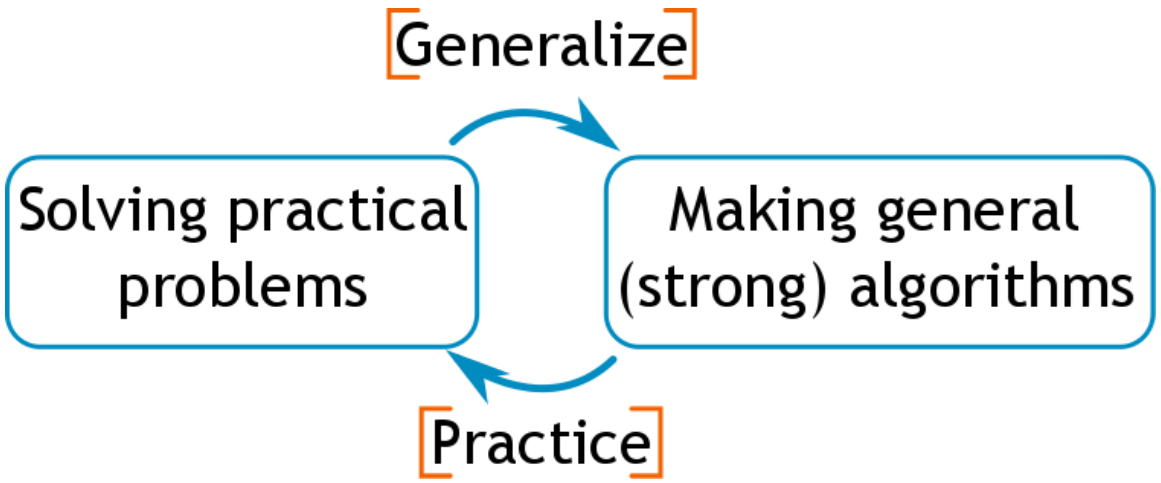


- Jurgen Schmidhuber: Deep Learning in Neural Networks: An Overview, Technical Report IDSIA-03-14 / arXiv:1404.7828 v2 [cs.NE], 2014. http://arxiv.org/abs/1404.7828
- Large Scale Deep Learning by Jeff Dean (Google) http://static.googleusercontent.com/media/research.google.com/ja//people/jeff/CIKM-keynote-Nov2014.pdf
- Hinton (talk): Brains, Sex, and Machine Learning https://youtu.be/DleXA5ADG78

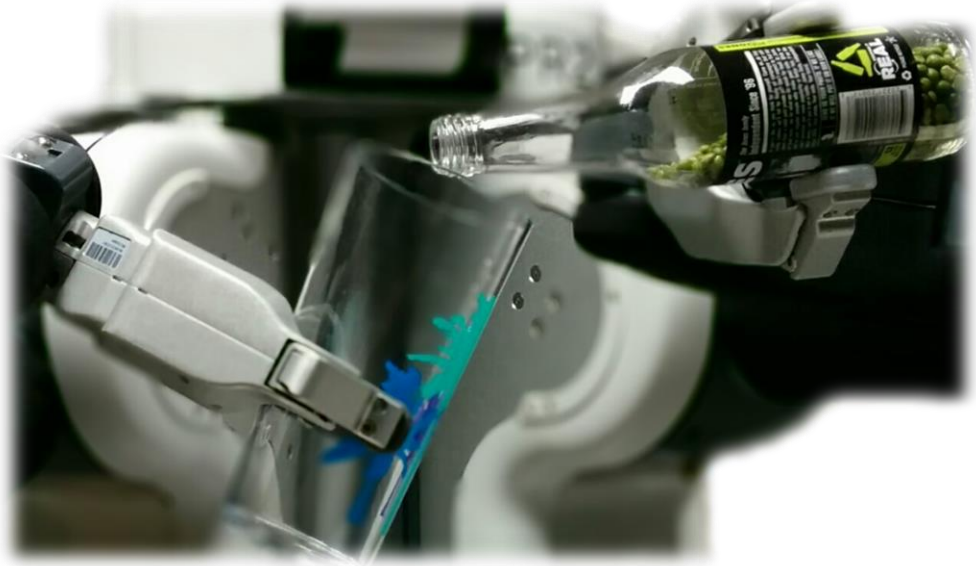
My view: background of DL's success and a lesson

- It is said that the reasons of success are progressed computers and big data, however, …
- There were many examples of image recognition researches and ML researches
  - Convolution layer can be seen as a generalization of feature detection
- and these researches are combined, which led the DL's success
- We need to make such a success in RL!

- Manipulation of soft objects
- Multi contact planning
- Handling variations
- DRC tasks
- Pouring, folding towels, opening things, …
- …



My practical example: *pouring*



Pouring oil

Why pouring?

- Many human intelligences are unified
  Such as various skills, planning, learning

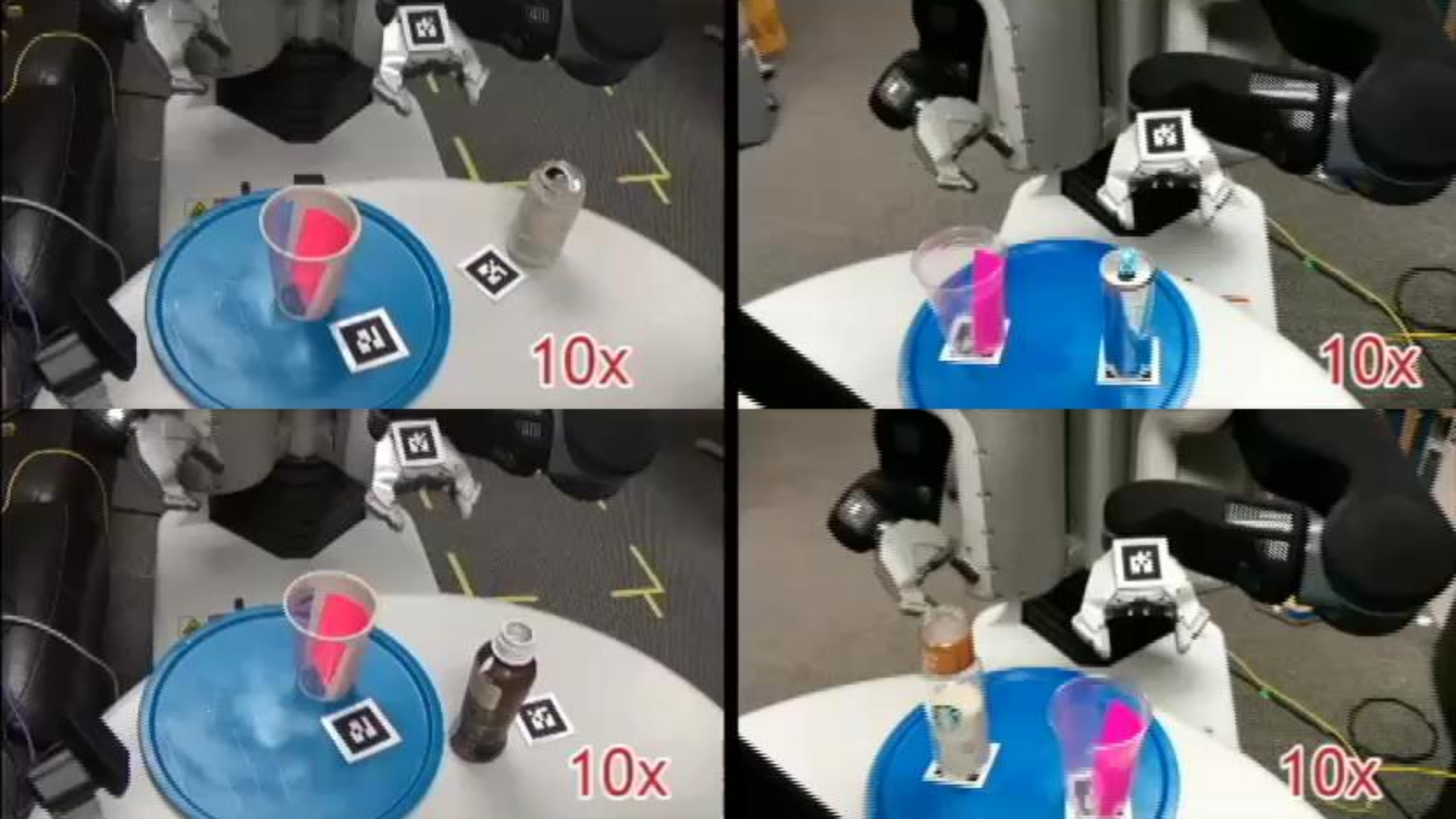
- How to solve pouring task in general?
  → More intelligent robot

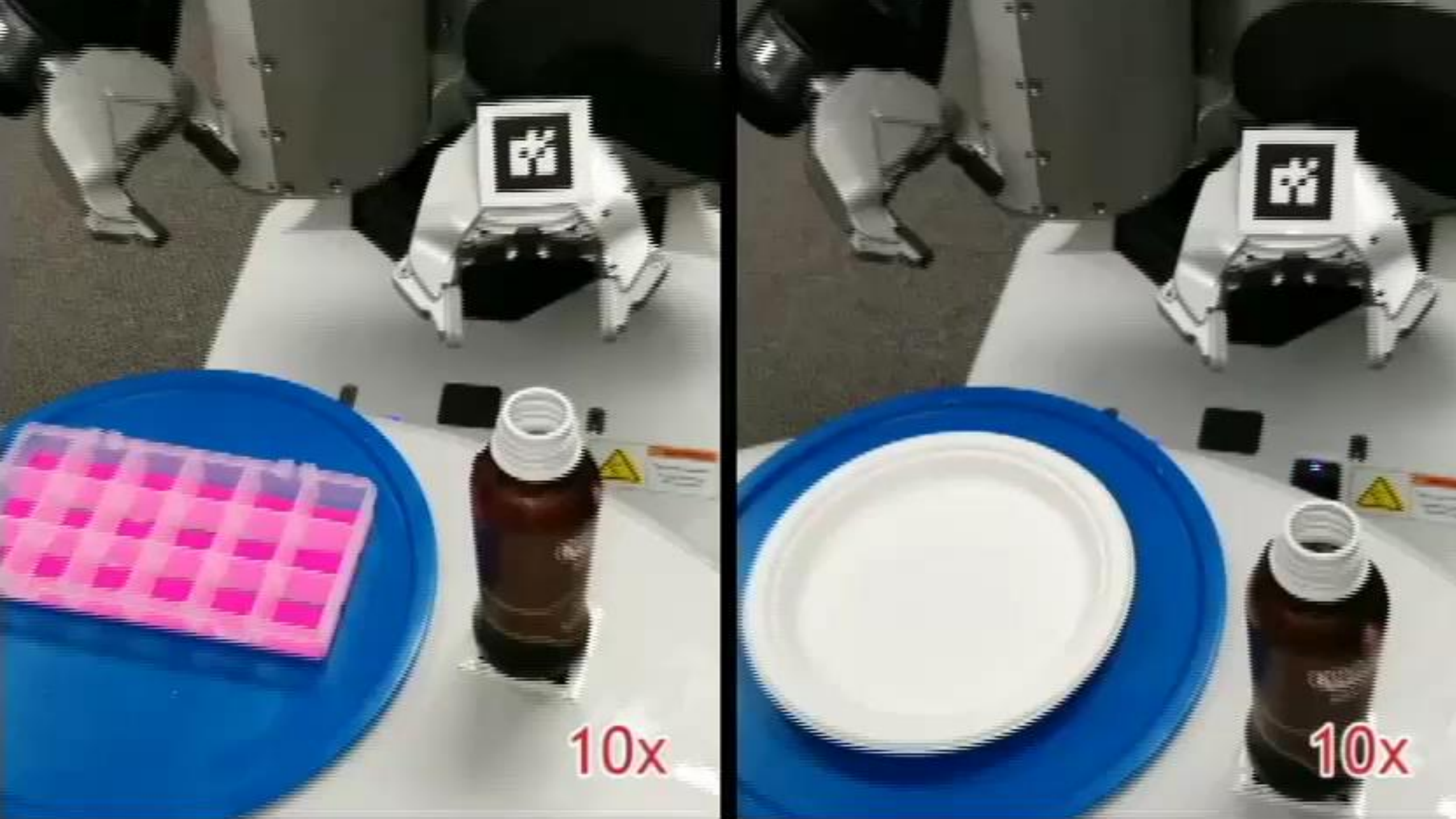
- Step 1. Make a practical & general pouring behavior
  - Learning from demonstration framework: *How humans are doing?*

    cf. Aude Billard and Daniel Grollman: Robot learning by demonstration, Scholarpedia, Vol. 8, No. 12, 2013. http://www.scholarpedia.org/article/Robot_learning_by_demonstration
  - Concentrate on *behavior* (simplify perception)
  - A. Yamaguchi, C. G. Atkeson, and T. Ogasawara: Pouring Skills with Planning and Learning Modeled from Human Demonstrations, International Journal of Humanoid Robotics, Vol.12, No.3, 2015.
- Step 2. Automate one-by-one i.e. reduce human's implementation
  - Q. What are complicated works in step 1?
  - Q. How to replace by AI/RL/ML tools?

https://www.youtube.com/watch?v=GjwfbOur3CQ

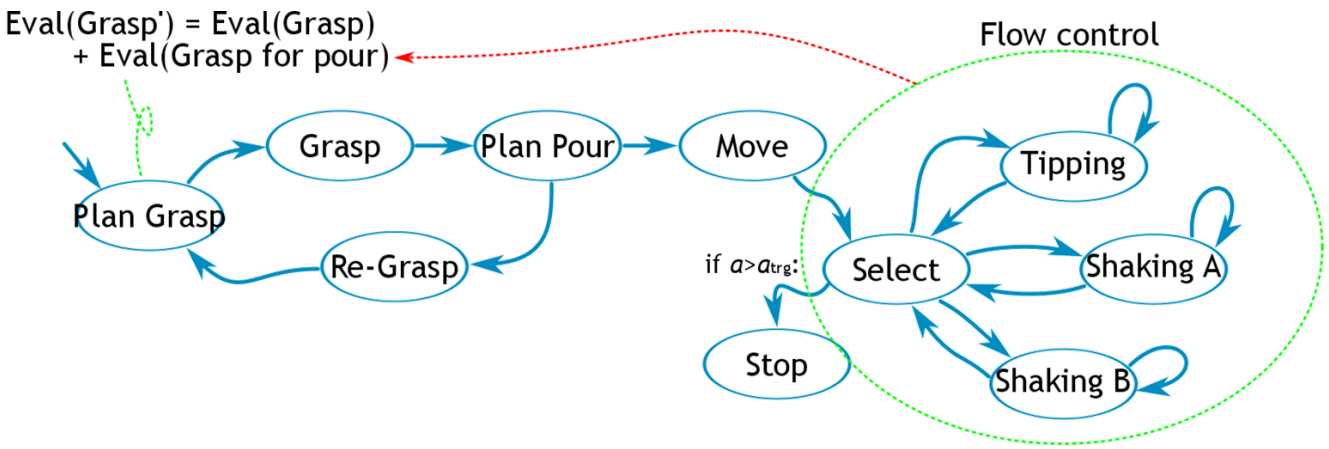
6.5x
5x
3.2x
11x


10x
10x
10x
10x


10x
10x

Essence in general pouring

- Skill library
  - flow ctrl (tip, shake, …), grasp, move arm, …
    → State machines (structure, feedback ctrl)
- Planning methods
  - grasp, re-grasp, pouring locations, feasible trajectories, …
- Learning methods
  - Improve plan quality
  - Skill selection
  - Parameter adjustment (e.g. shake axis)



Idea behind

Decomposition of entire pouring → skills

- Make general pouring behavior easily
- Make planning easy:
  - Small planning: grasping, pouring locations, …
  - Planned independently
- Make learning easy:
  - Small learning: skill selection, skill adjustment, planning impromement

How much each element contributes to generalization?

Method	Material type	Container shape	Context	Initial poses	Target amount
State machines for skills	**	**	**	*	**
Planning methods		**		**	
Learning for planning methods		*		*	
Learning for selection	**	**			
Learning for adjustment	*	*			

**: Plays an essential roll.

*: Plays an important roll in improving performance.

How to automate?

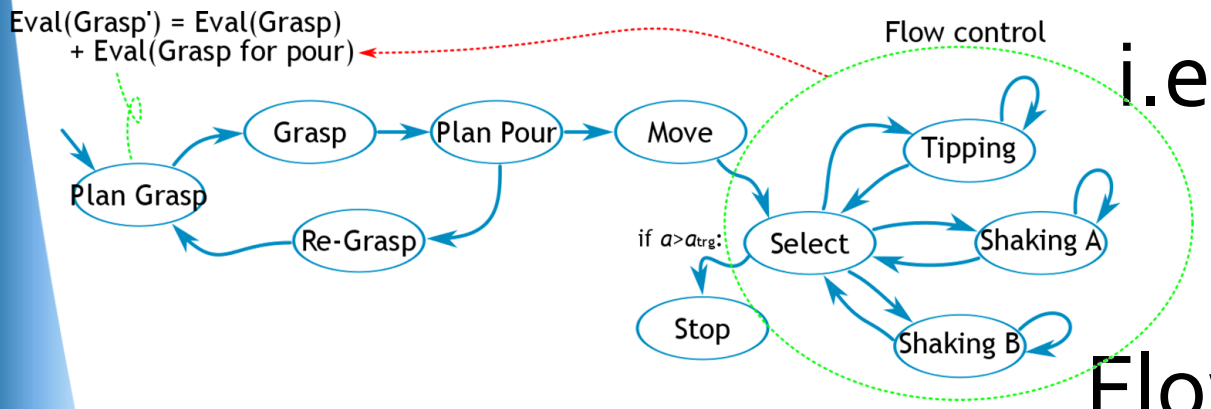
- How to find skills?
  - Human demonstrations → (ML tool) → FSMs
    e.g. [Niekum,2013]

- Unification of planning & learning
  - Designing evaluation functions: complicated
  - (Feasible) Reinforcement Learning

Why reinforcement learning?

- e.g. Evaluation func of grasping =
  Eval of grasping + Eval of *grasping for flow ctrl*

Propagation from future outcome
i.e. Decomposed planning
= *Dynamic programming*

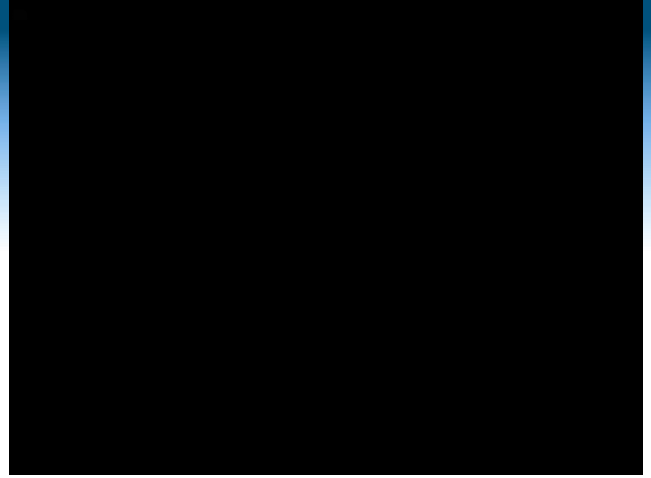


Flow is hard to model
→ *Reinforcement learning*

RL methods

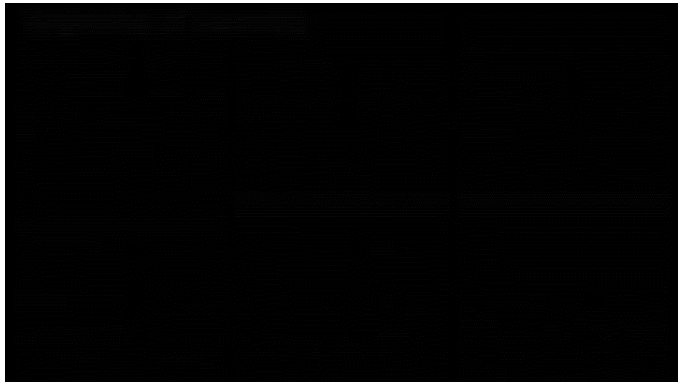
- **Model-free**
  - **Direct policy learning**
    - ✓ E.g. [Kober,Peters,2011], [Kormushev,2010]
    - ✓ Popular in RL for robotics
  - **Value function based**
    - ✓ E.g. [Yamaguchi,2013]
  - A disadvantage of model-free methods is the poor generalization ability
  - Attempt to overcome: [Kober,Peters,2012], [Levine,Abbeel,2015]
- **Model-based**
  - **Learning system dynamics + Planning**
  - Not popular in RL for robotics:
    - ✓ We need to gather many samples to construct dynamics models
    - ✓ We need to solve two problems, model learning and planning
  - Learning dynamics: LWR, GPR, Neural Net, etc.
  - Planning: DDP
    e.g. [Mayne,1966], [Tassa,Todorov,2011], [Levine,Koltun,2013]
  - Examples: [Schaal,Atkeson,1994], [Morimoto,Atkeson,2003], [Pan,Theodorou,2014]
- **Hybrid**
  - Dyna [Sutton,1990], [Sutton et al.,2008]

Q. Which is the best?

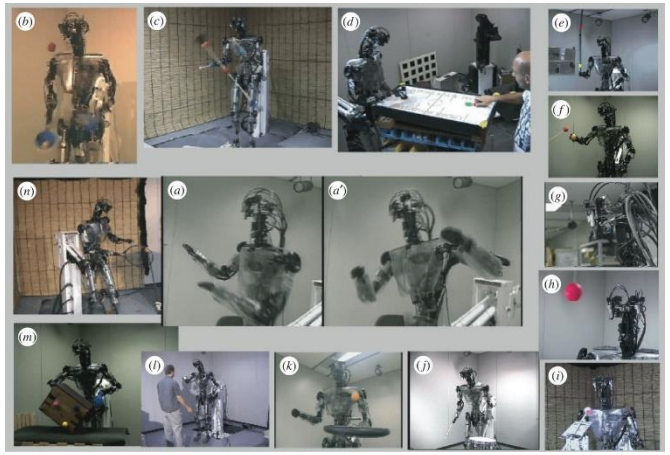


[1]

[2]



[3]



[4]

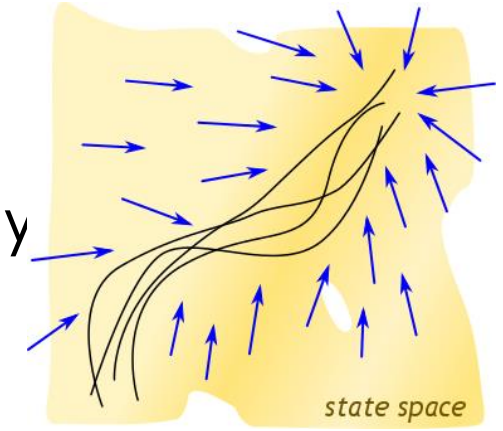
- [1] By J. Kober and J. Peters: Learning Motor Primitives for Robotics (Ball-in-cup)
  http://www.ausy.tu-darmstadt.de/Research/LearningMotorPrimitives
  https://www.youtube.com/watch?v=cNyoMVZQdYM
- [2] By P. Kormushev et al.
  Video: Robot Arm Wants Nothing More Than To Master The Art Of The Flapjack-Flip
  http://www.popsci.com/technology/article/2010-07/after-50-attempts-hard-working-flapjack-bot-learns-flip-pancakes-video
  http://programming-by-demonstration.org/showPubli.php?publi=3018
  https://vimeo.com/13387420#at=NaN
- [3] YouTube: RL_MotionLearning (by myself)
  https://www.youtube.com/playlist?list=PL41MvLpqzOg8FFoxekWT9NXCdjzN_8PUS
- [4] J. Morimoto, M. Kawato: Creating the brain and interacting with the brain: an integrated approach to understanding the brain, Interface, 2015.
  http://rsif.royalsocietypublishing.org/content/12/104/20141250

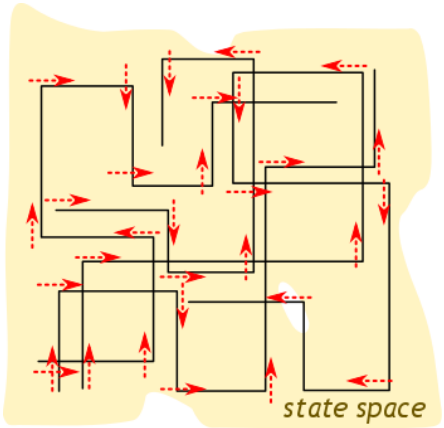
- Text
  - Sutton, Barto Reinforcement Learning: An Introduction, The MIT Press, 1998. https://webdocs.cs.ualberta.ca/~sutton/book/ebook/the-book.html
- Model-base RL
  - S. Schaal and C. Atkeson, ``Robot juggling: implementation of memory-based learning,'' in the IEEE International Conference on Robotics and Automation (ICRA'94), vol. 14, no. 1, 1994, pp. 57--71.
  - J. Morimoto, G. Zeglin, and C. Atkeson, ``Minimax differential dynamic programming: Application to a biped walking robot,'' in the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03), vol. 2, 2003, pp. 1927--1932.
- Model-free RL
  - J. Kober and J. Peters, ``Policy search for motor primitives in robotics,'' Machine Learning, vol. 84, no. 1-2, pp. 171--203, 2011.
  - E. Theodorou, J. Buchli, and S. Schaal, ``Reinforcement learning of motor skills in high dimensions: A path integral approach,'' in the IEEE International Conference on Robotics and Automation (ICRA'10), may 2010, pp. 2397--2403.
  - D. Ernst, P. Geurts, and L. Wehenkel, ``Tree-based batch mode reinforcement learning,'' Journal of Machine Learning Research, vol. 6, pp. 503--556, 2005.
  - P. Kormushev, S. Calinon, and D. G. Caldwell, ``Robot motor skill coordination with EM-based reinforcement learning,'' in the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'10), 2010, pp. 3232--3237.
  - A. Yamaguchi, J. Takamatsu, and T. Ogasawara, ``DCOB: Action space for reinforcement learning of high dof robots,'' Autonomous Robots, vol. 34, no. 4, pp. 327--346, 2013.
  - J. Kober, A. Wilhelm, E. Oztop, and J. Peters, ``Reinforcement learning to adjust parametrized motor primitives to new situations,'' Autonomous Robots, vol. 33, pp. 361--379, 2012.
  - S. Levine, N. Wagener, and P. Abbeel, ``Learning contact-rich manipulation skills with guided policy search,'' in the IEEE International Conference on Robotics and Automation (ICRA'15), 2015.
- Hybrid
  - R. S. Sutton, ``Integrated architectures for learning, planning, and reacting based on approximating dynamic programming,'' in the Seventh International Conference on Machine Learning. Morgan Kaufmann, 1990, pp. 216--224.
  - R. S. Sutton, C. Szepesv¥'{a}ri, A. Geramifard, and M. Bowling, ``Dyna-style planning with linear function approximation and prioritized sweeping,'' in Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence, 2008, pp. 528--536.

Model-free RL (my view)

- Direct policy learning
  (Policy gradient, NAC, PoWER, PI², ...)
  - ☺Sample efficient
  - ☺Work stably in continuous state-action space
  - ☹Many current methods: explore around a trajectory only
- Value function based
  (Q(λ)-learning, FQI, ...)
  - ☹Need many samples
  - ☺Good for learning from scratch
  - ☺Work stably in continuous state space
  - ☹Problematic in action space
- ☺Some methods work in POMDP
  - Partially observable Markov decision process
- ☺Small planning in run-time

☺ Good / ☹ Bad



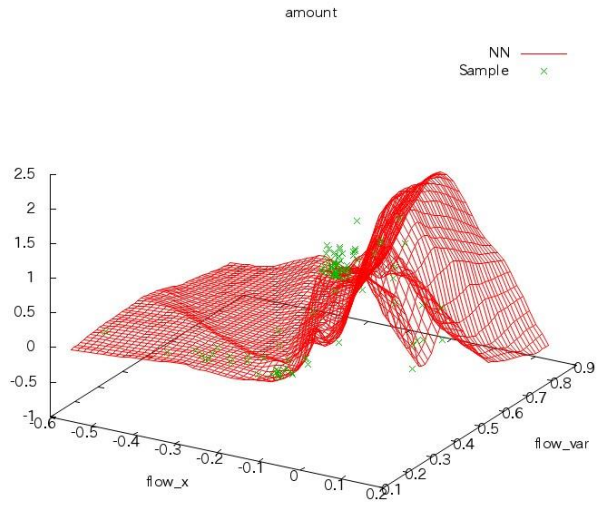


A. Yamaguchi, et al., ``DCOB: Action space for reinforcement learning of high dof robots,'' Autonomous Robots, vol. 34, no. 4, pp. 327--346, 2013.

Model-based RL (my view)

☺ Good / ☹ Bad

- ☹Need to solve two problems
  - Learning models ← ☹Need many samples to cover state-action space
  - ☹Planning (dynamic programming) in run time (or we can cache)
- ☺*Maybe* better for generalization (???)
- ☺Learning dynamics would be robust because it is supervised learning
- ☺We can combine with existing models
  e.g. collision detector
- ☺*Maybe* we can seamlessly integrate with planning methods
  e.g. Collision-avoidance planner
- Good DDP (differential dynamic programming) are proposed recently → ☺Work with continuous state-action domain
  e.g. [Tassa,Todorov,2011], [Levine,Koltun,2013]
- ☹Learned dynamics model has many local maxima
- ☹Weak in POMDP



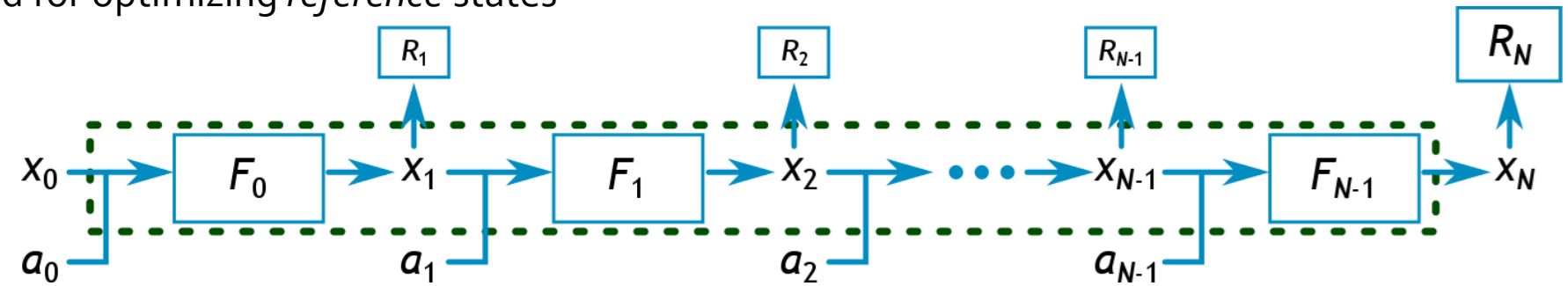
e.g. learned dynamics

Recent work

Differential Dynamic Programming with Temporally Decomposed Dynamics, Humanoids'15

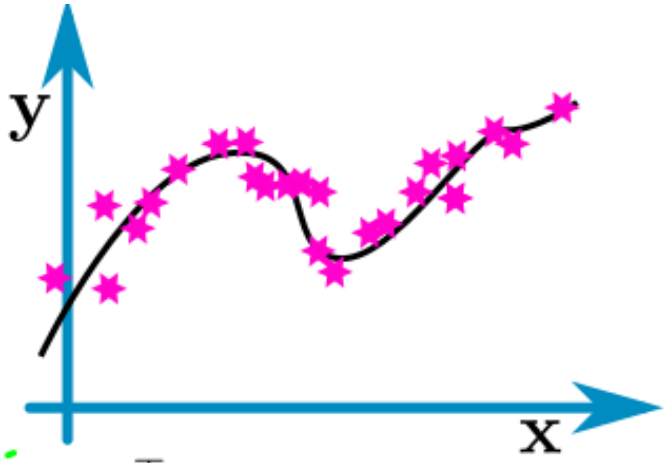
http://www.researchgate.net/publication/282157952_Differential_Dynamic_Programming_with_Temporally_Decomposed_Dynamics

- Learning dynamics models: LWR
  - Locally Weighted Regression cf. [Atkeson,Schaal,1997]
  - + Extension on expectation computation
- Planning: Stochastic DDP
  - Consider probabilistic states and evaluation function $J_n(\mathbf{x}_n, \{\mathbf{a}_n, \ldots, \mathbf{a}_{N-1}\}) = \sum_{n'=n+1}^{N} \mathrm{E}[R_{n'}(\mathbf{x}_{n'})]$
  - Since learned models have many local maxima…
    - ✓ First-order gradient algorithm with multiple gradient candidates
    - ✓ Multiple criteria evaluation function with *reference* states
- Decomposed dynamical system
  - Consider decomposition even where *no action is taken*
    - ✓ Good for learning dynamics more accurately
    - ✓ Good for optimizing *reference* states



Model learning

- Locally weighted regression (LWR)
  cf. [Atkeson,Schaal,1997]
- + Extension of expectation computation



LWR

$$\mathbf{y} = \boldsymbol{\beta}^\top \mathbf{x}$$

$$\boldsymbol{\beta} = (\mathbf{X}^\top \mathbf{W} \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{W} \mathbf{Y}$$

$$\mathbf{Q}(\mathbf{x}) = (\mathbf{X}\boldsymbol{\beta} - \mathbf{Y})^\top \mathbf{W} (\mathbf{X}\boldsymbol{\beta} - \mathbf{Y}) / (\mathrm{Tr}(\mathbf{W})(1 - D/M))$$

Learning dynamics

$$\mathbf{x}^\top = [\mathbf{x}_n^\top, \mathbf{a}_n^\top, 1]$$

$$\mathbf{y} = \mathbf{x}_{n+1}$$

$$\boldsymbol{\beta}^\top = [\mathbf{F}_{\mathrm{X}n}, \mathbf{F}_{\mathrm{A}n}, \mathbf{F}_{0n}]$$

$$\partial \mathbf{F}_{\mathrm{X}n} = \mathbf{F}_{\mathrm{X}n}^\top \qquad \partial \mathbf{F}_{\mathrm{A}n} = \mathbf{F}_{\mathrm{A}n}^\top$$

$$\bar{\mathbf{x}}_{n+1} = \boldsymbol{\beta}^\top [\bar{\mathbf{x}}_n^\top, \mathbf{a}_n^\top, 1]^\top$$

$$\boldsymbol{\Sigma}_{n+1} = \mathbf{F}_{\mathrm{X}n} \boldsymbol{\Sigma}_n \mathbf{F}_{\mathrm{X}n}^\top + \mathbf{Q}_n(\bar{\mathbf{x}}_n, \mathbf{a}_n)$$

Forward and backward propagations

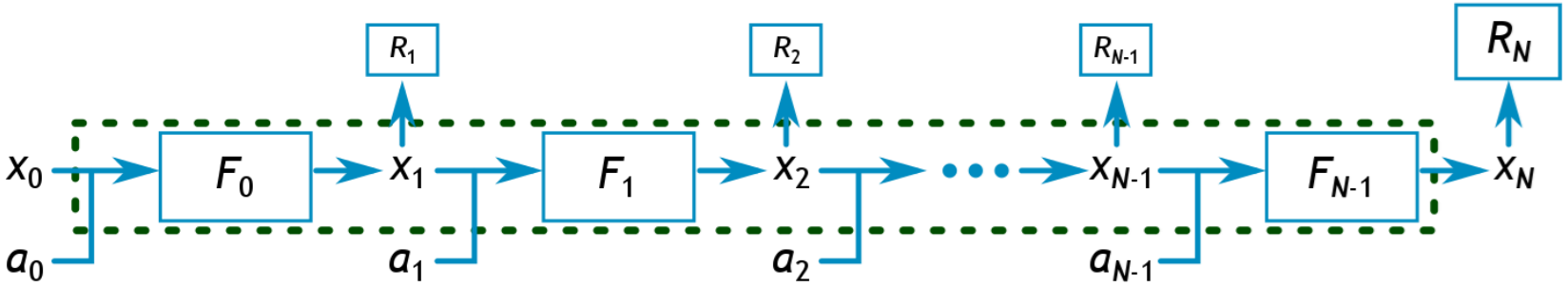
deterministic

$$\mathbf{x}_{n+1} = \mathbf{F}_n(\mathbf{x}_n, \mathbf{a}_n)$$
$$\partial \mathbf{F}_{\mathrm{X}n} = \frac{\partial \mathbf{F}_n}{\partial \mathbf{x}_n}$$
$$\partial \mathbf{F}_{\mathrm{A}n} = \frac{\partial \mathbf{F}_n}{\partial \mathbf{a}_n}$$

probabilistic

$$\mathbf{x}_n \sim \mathcal{N}(\bar{\mathbf{x}}_n, \boldsymbol{\Sigma}_n) \qquad \mathbf{x}_{n+1} \sim \mathcal{N}(\bar{\mathbf{x}}_{n+1}, \boldsymbol{\Sigma}_{n+1})$$
$$\bar{\mathbf{x}}_{n+1} = \tilde{\mathbf{F}}_n(\bar{\mathbf{x}}_n, \boldsymbol{\Sigma}_n, \mathbf{a}_n) \qquad \boldsymbol{\Sigma}_{n+1} = \mathbf{S}_n(\bar{\mathbf{x}}_n, \boldsymbol{\Sigma}_n, \mathbf{a}_n)$$
$$\partial \tilde{\mathbf{F}}_{\mathrm{X}n} = \frac{\partial \tilde{\mathbf{F}}_n}{\partial \bar{\mathbf{x}}_n} \qquad \partial \tilde{\mathbf{F}}_{\mathrm{A}n} = \frac{\partial \tilde{\mathbf{F}}_n}{\partial \mathbf{a}_n}$$

$$J_n(\mathbf{x}_n, \{\mathbf{a}_n, \ldots, \mathbf{a}_{N-1}\}) = \sum_{n'=n+1}^{N} \mathrm{E}[R_{n'}(\mathbf{x}_{n'})]$$



forward propagation

$$\mathbf{x}_{n+1} \sim \mathcal{N}(\bar{\mathbf{x}}_{n+1}, \boldsymbol{\Sigma}_{n+1}) = \mathcal{N}(\tilde{\mathbf{F}}_n(\mathbf{x}_n, \mathbf{a}_n), \mathbf{S}_n(\mathbf{x}_n, \mathbf{a}_n))$$
$$\mathrm{E}[R_n(\mathbf{x}_n)] = R_n(\bar{\mathbf{x}}_n) + \mathrm{Tr}(\mathbf{A}_n \boldsymbol{\Sigma}_n)$$
$$\mathrm{var}[R_n(\mathbf{x}_n)] = 2\mathrm{Tr}(\mathbf{A}_n \boldsymbol{\Sigma}_n \mathbf{A}_n \boldsymbol{\Sigma}_n) + \mathbf{b}_n^\top \boldsymbol{\Sigma}_n \mathbf{b}_n$$

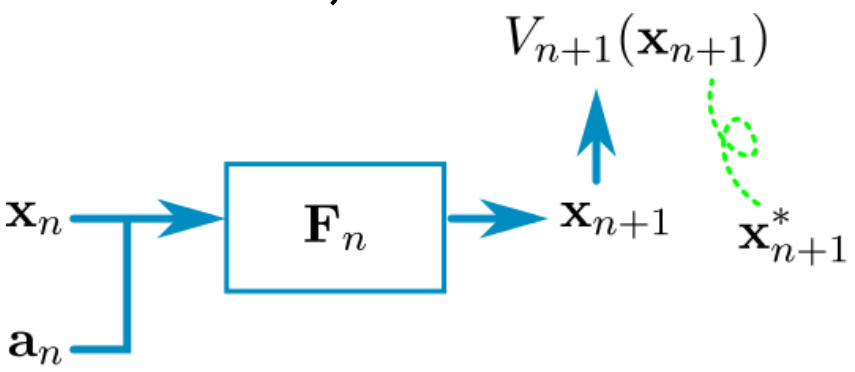
backward propagation

$$\boldsymbol{\Omega}_N = \mathbf{0} \qquad \boldsymbol{\Omega}_n = \partial \tilde{\mathbf{F}}_{\mathrm{X}n+1} \boldsymbol{\Omega}_{n+1} + \partial \tilde{\mathbf{R}}_{\mathrm{X}n+1}$$
$$\partial \mathbf{J}_{\mathrm{A}n} = \partial \tilde{\mathbf{F}}_{\mathrm{A}n} \boldsymbol{\Omega}_n$$

Multiple criteria evaluation function

- Motivation: DDP w learned models has many local maxima (sequence of dynamics)

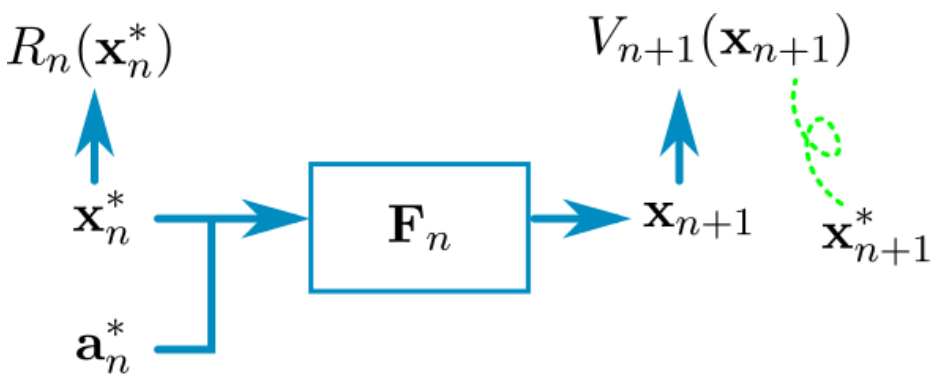
Value function with ref. state (another criterion)



$$V_n(\mathbf{x}_n) = -(\mathbf{x}^*_n - \mathbf{x}_n)^\top \mathbf{W}_{\mathrm{rs}}(\mathbf{x}^*_n - \mathbf{x}_n)$$

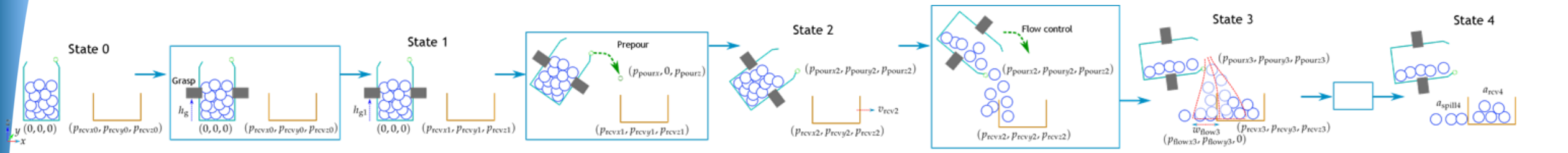
$$\mathrm{E}[V_n(\mathbf{x}_n)] = V_n(\bar{\mathbf{x}}_n) - \mathrm{Tr}(\mathbf{W}_{\mathrm{rs}}\boldsymbol{\Sigma}_n)$$

How to get reference states



$$L_n(\mathbf{x}^*_n, \mathbf{a}^*_n) = R_n(\mathbf{x}^*_n) + V_{n+1}(\tilde{\mathbf{F}}_n(\mathbf{x}^*_n, \mathbf{a}^*_n))$$

Multiple criteria: e.g. VF → VF → Jn → Jn → Jn



$\mathbf{x}_0 = (p_{\mathrm{rcv}x0}, p_{\mathrm{rcv}y0})$

$\mathbf{a}_0 = (h_{\mathrm{g}})$

$\mathbf{x}_1 = (p_{\mathrm{rcv}x1}, p_{\mathrm{rcv}y1}, h_{\mathrm{g}1})$

$\mathbf{a}_1 = (p_{\mathrm{pour}x}, p_{\mathrm{pour}z})$

$\mathbf{x}_2 = (\langle p_{\mathrm{rcv}x2} - p_{\mathrm{rcv}x1}\rangle, \langle p_{\mathrm{rcv}y2} - p_{\mathrm{rcv}y1}\rangle, \langle v_{\mathrm{rcv}2}\rangle, p_{\mathrm{pour}x2} - p_{\mathrm{rcv}x2}, p_{\mathrm{pour}z2} - p_{\mathrm{rcv}z2})$

$\mathbf{a}_2 = ()$

Penalty for rcv move

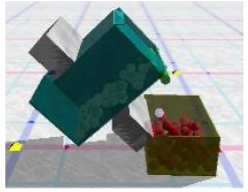
$\mathbf{x}_3 = (\langle p_{\mathrm{flow}x3} - p_{\mathrm{rcv}x3}\rangle, \langle p_{\mathrm{flow}y3} - p_{\mathrm{rcv}y3}\rangle, w_{\mathrm{flow}3}, p_{\mathrm{pour}z3} - p_{\mathrm{rcv}z3})$

$\mathbf{a}_3 = ()$

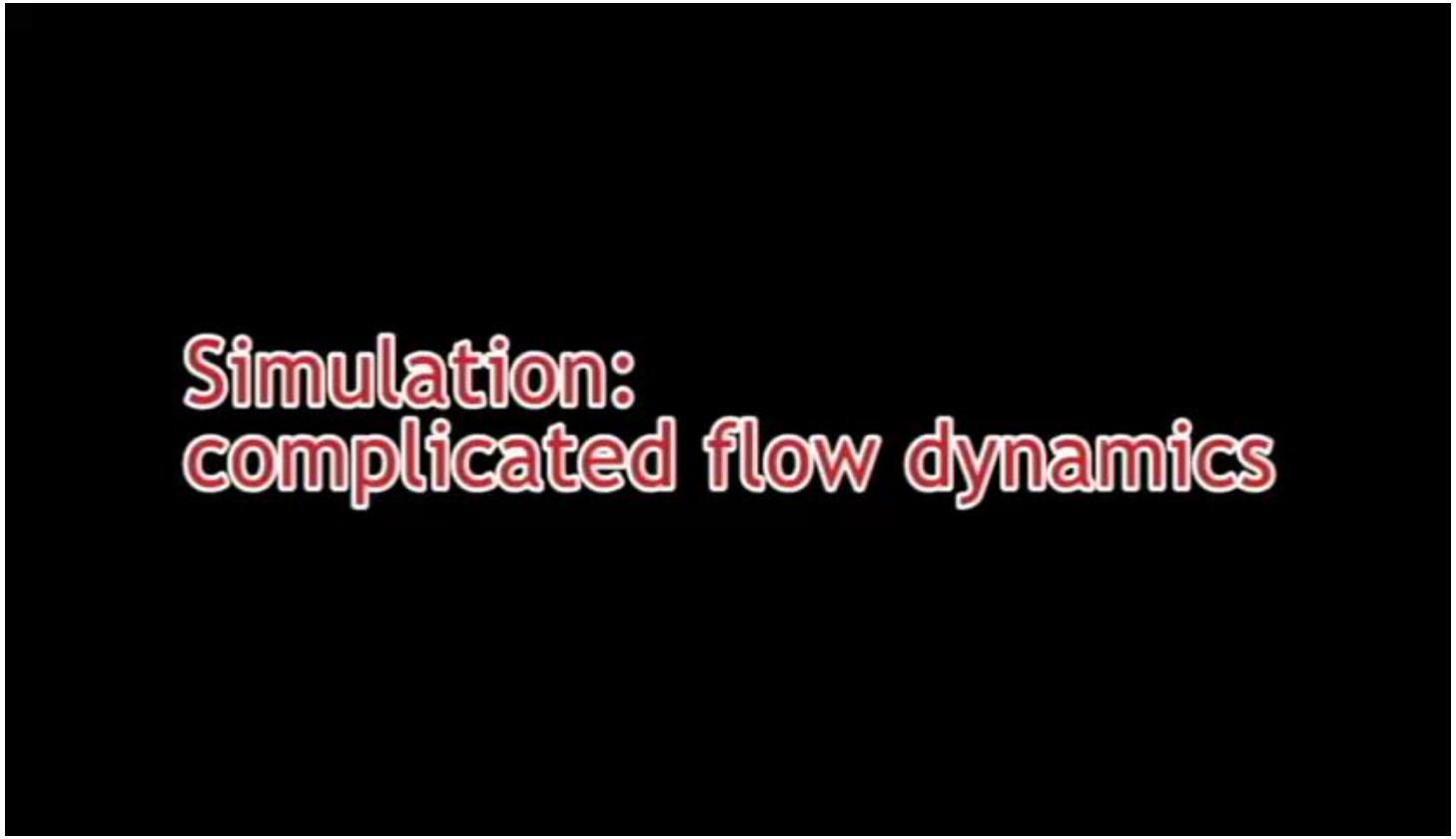
$\mathbf{x}_4 = (a_{\mathrm{rcv}4}, -a_{\mathrm{spill}4})$

Penalty for spilled

Reward for poured



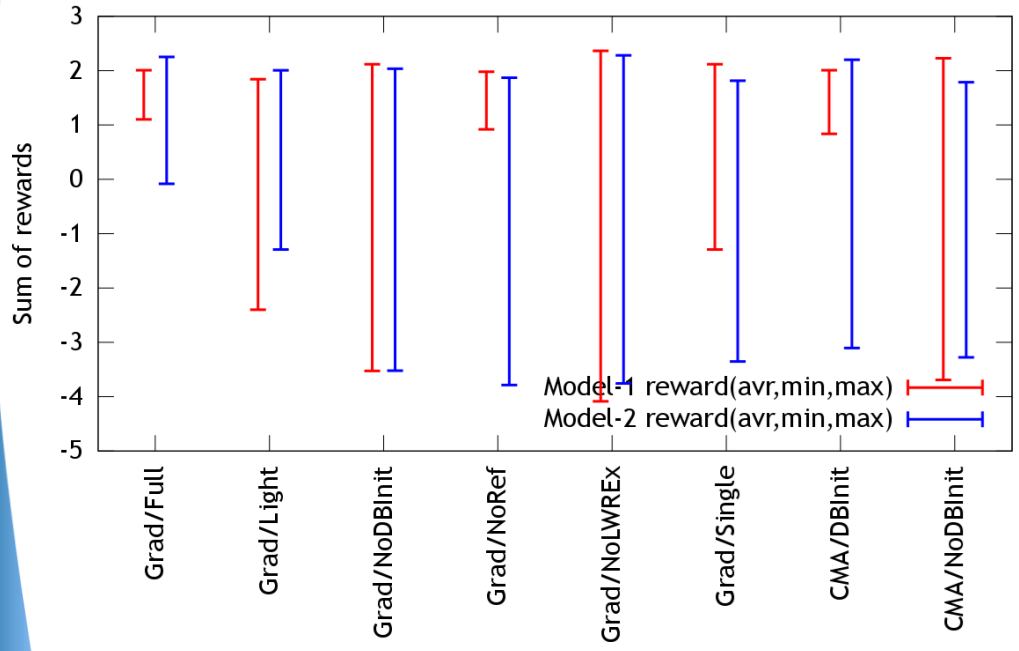
Poured material

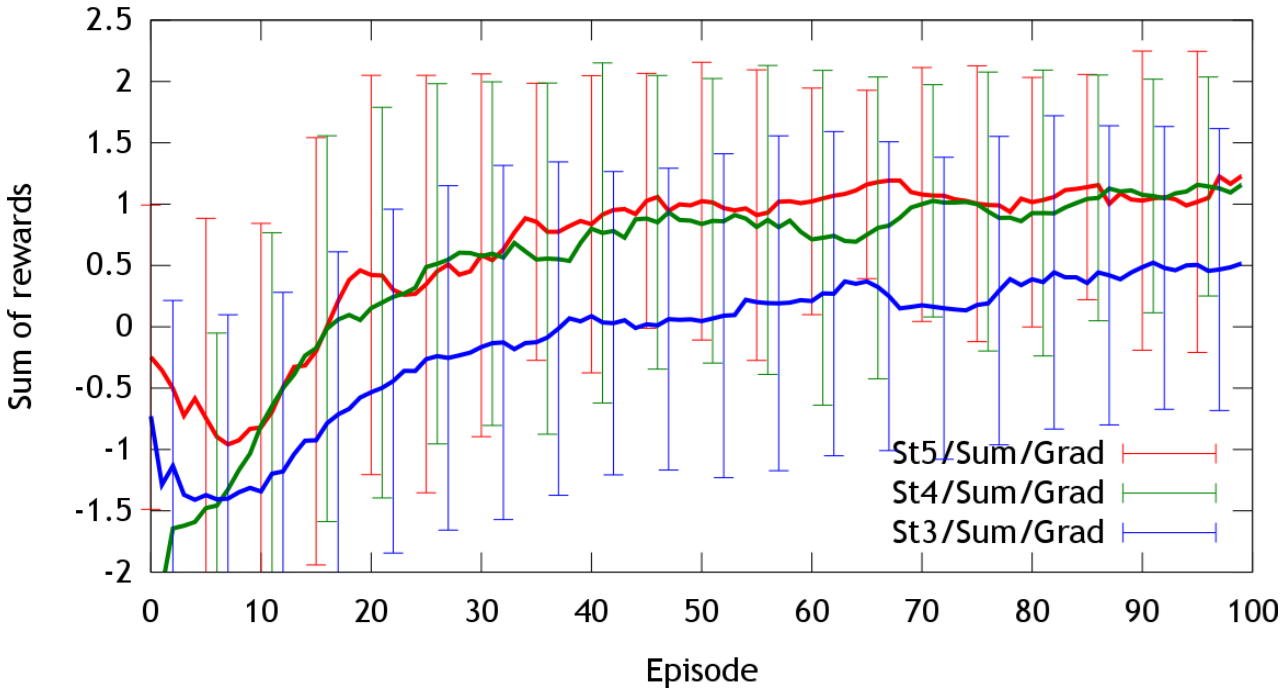


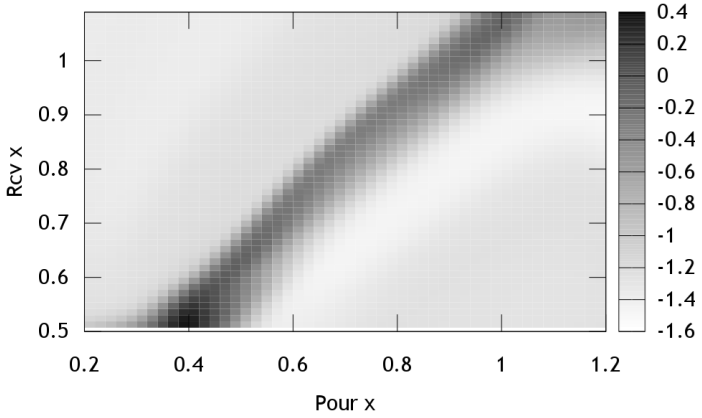
https://www.youtube.com/watch?v=OrjTHwoCHew

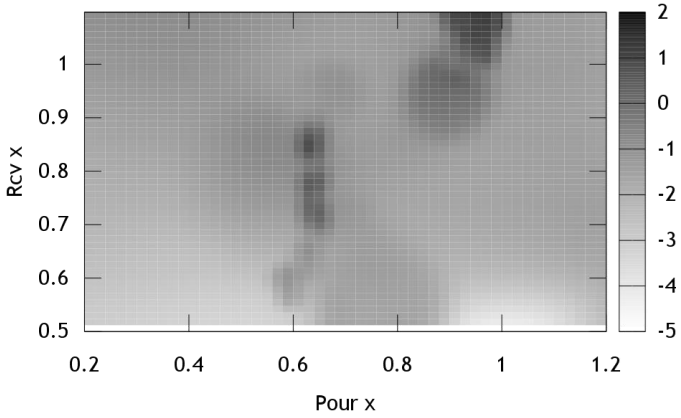
Learning dynamics
+ Dynamic programming

Is dynamics decomposition useful?





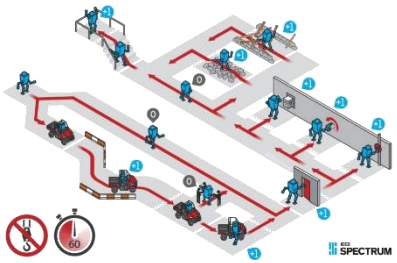




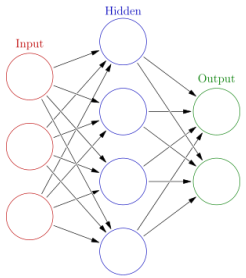
PR2 preliminary experiments

Figure sources

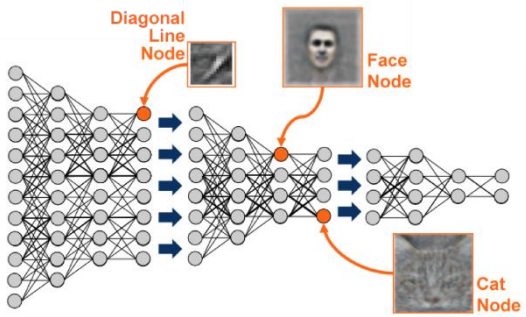
Team WPI-CMU: Darpa Robotics Challenge
http://www.cs.cmu.edu/~cga/drc/
(cmu-drc-final-public.zip)



http://spectrum.ieee.org/automaton/robotics/humanoids/drc-finals-course



Artificial neural network
https://en.wikipedia.org/wiki/Artificial_neural_network



http://scyfer.nl/wp-content/uploads/2014/05/Deep_Neural_Network.png