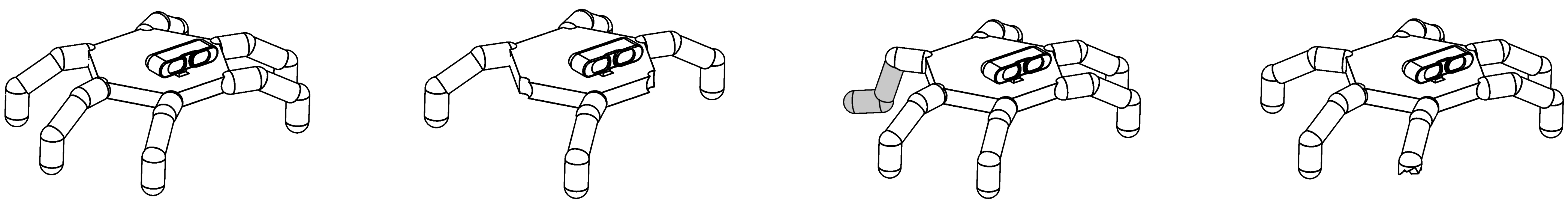
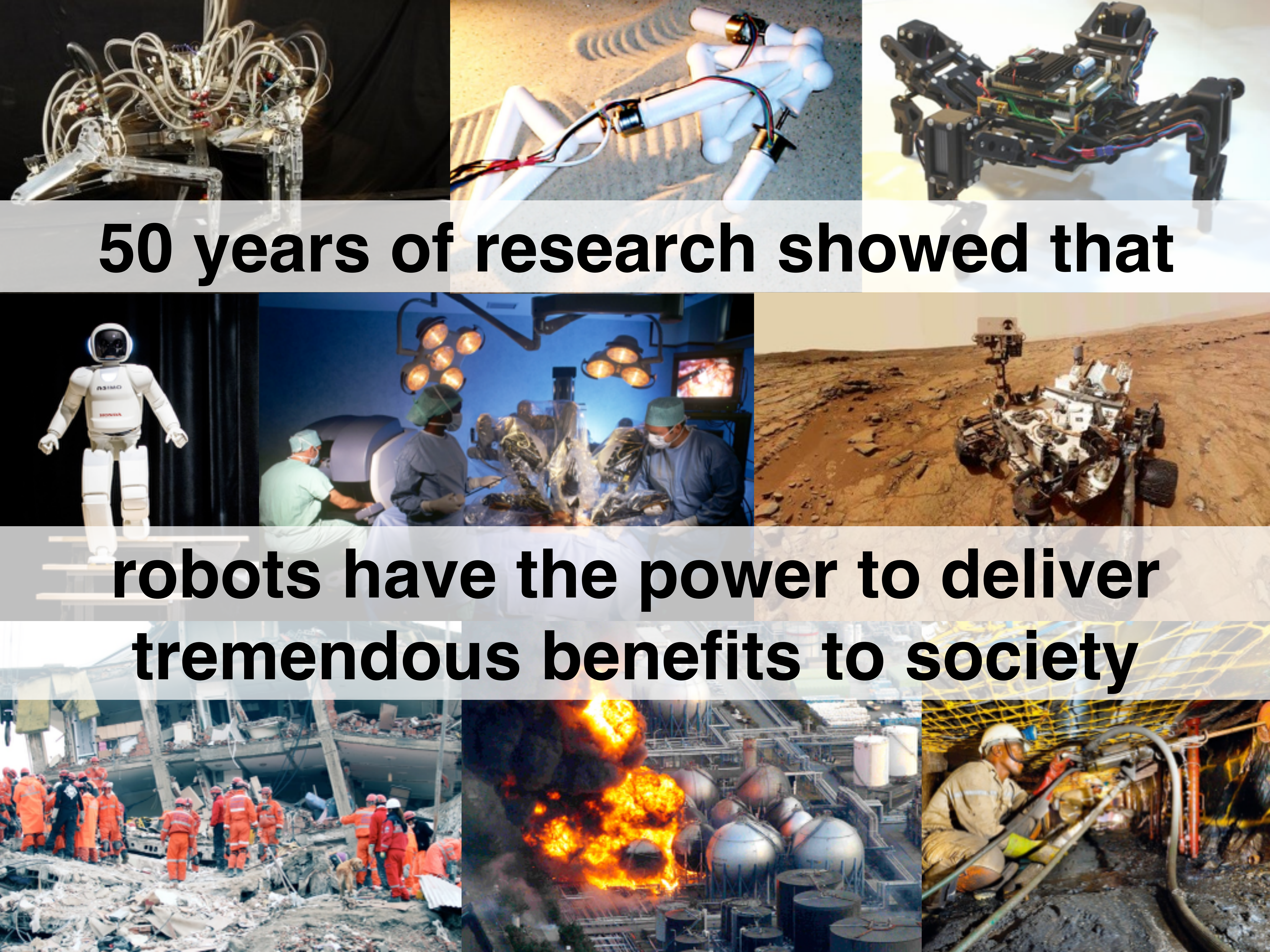
Creative Adaptation through Learning



Antoine CULLY

thesis director:
Stéphane DONCIEUX
thesis supervisor:
Jean-Baptiste MOURET

ANR DGA ISIR UPMC SORBONNE UNIVERSITÉS CNRS


50 years of research showed that
robots have the power to deliver
tremendous benefits to society


Some robots start to spread in society
1348
1348
1374
1375
GARDENA
They operate in relatively simple
environments with few actions:
It’s a good start

Sophisticated and versatile robots

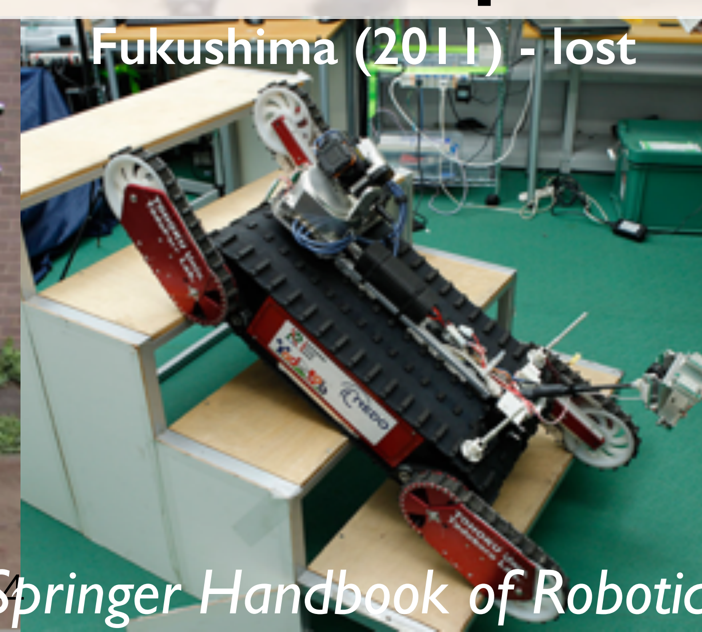
Darpa Robotics Challenge

Robots should continue their mission, even when damaged

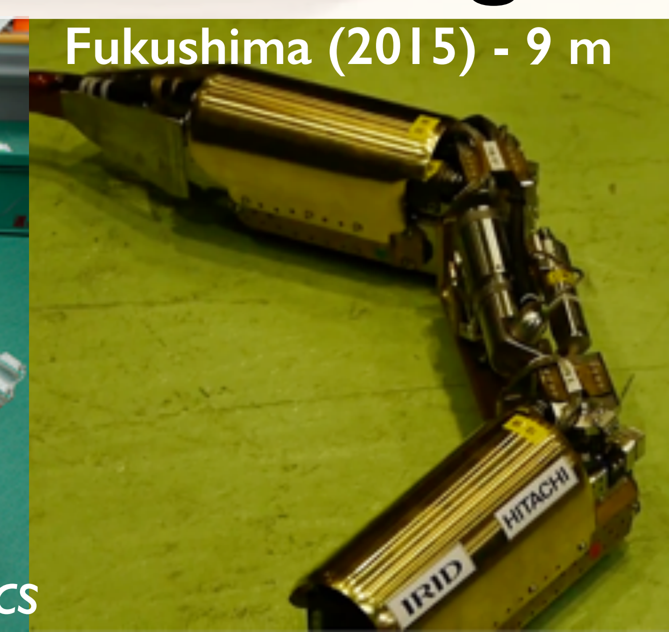
are complex to control … … and prone to damages

La Conchita, mudslide (2005) - 2 minutes

Sago mine (2006) - 700 m



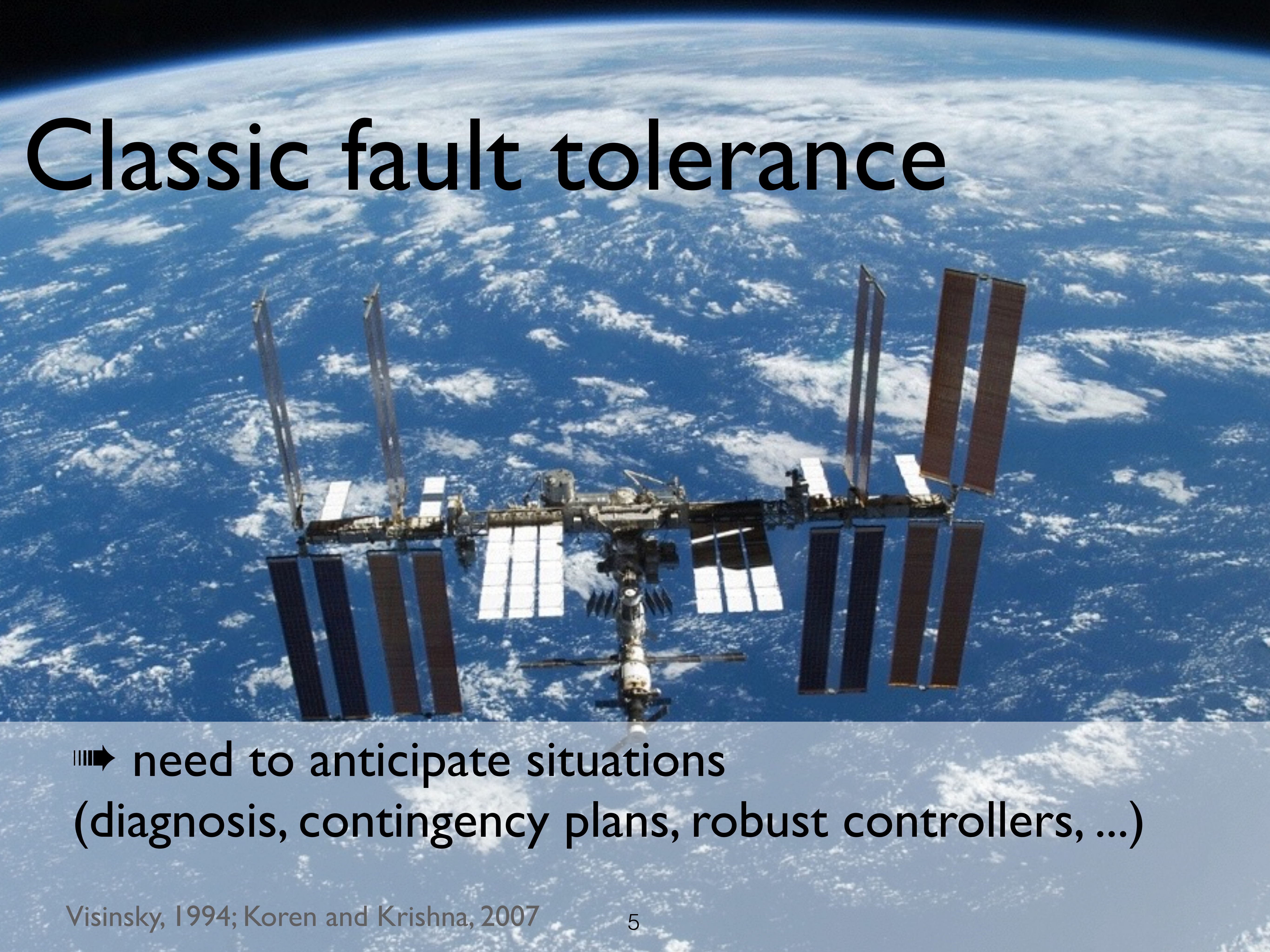
Fukushima (2011) - lost



Fukushima (2015) - 9 m

Murphy et al.(2008). Search and rescue robotics. In *Springer Handbook of Robotics*

Classic fault tolerance



➠ need to anticipate situations
(diagnosis, contingency plans, robust controllers, ...)

Visinsky, 1994; Koren and Krishna, 2007

Unexpected situation ?

Adaptation through learning …

… like animals

Wolpert,D.M.,Ghahramani,Z.&Flanagan, J. R. (2001) Trends Cogn. Sci.

Learning in an unforeseen situation

Reinforcement learning problem

"Reinforcement learning is learning what to do so as to maximize a numerical reward signal."

Sutton, R. S. and Barto, A. G. (1998). *Introduction to Reinforcement Learning*.

Traditional RL algorithms

Discrete state and action spaces

Policy search Algorithms

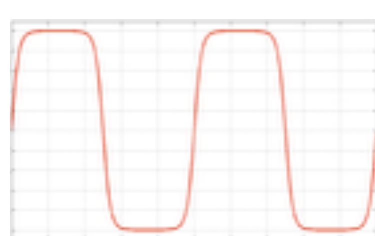
continuous search space

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} f(\mathbf{x})$$

Commonly used for motor skills learning

$\mathbf{x}$

Set of parameters



Motor Commands (controller)

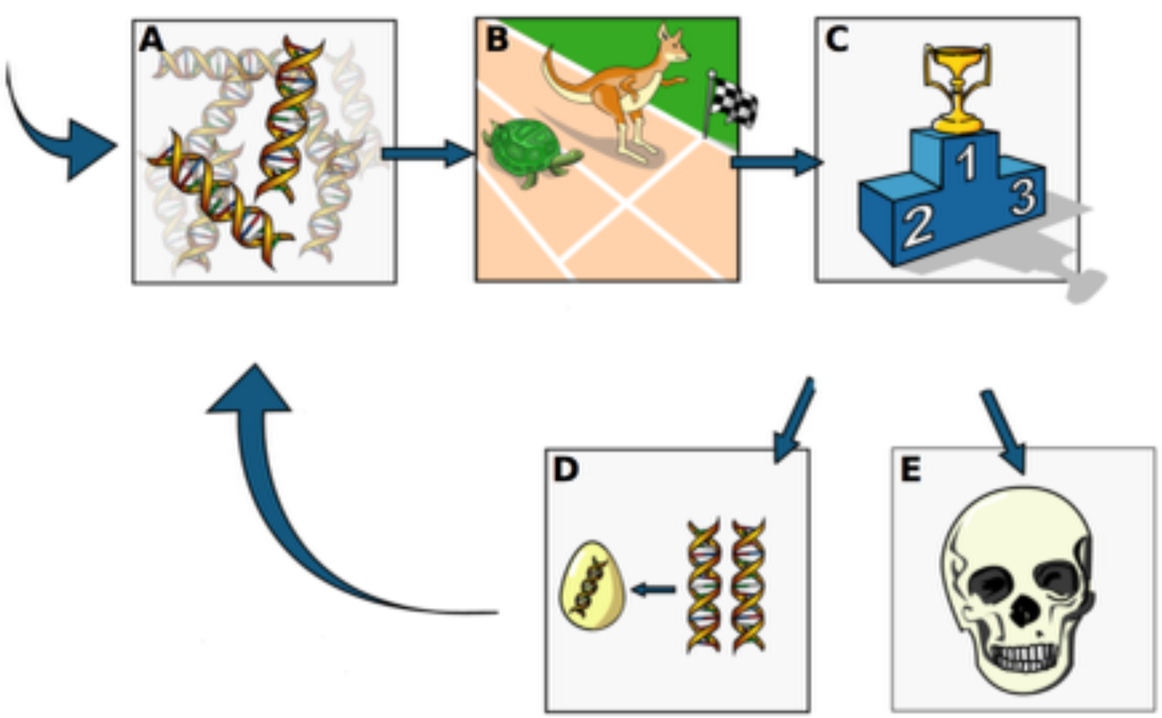
Assessed via sensors

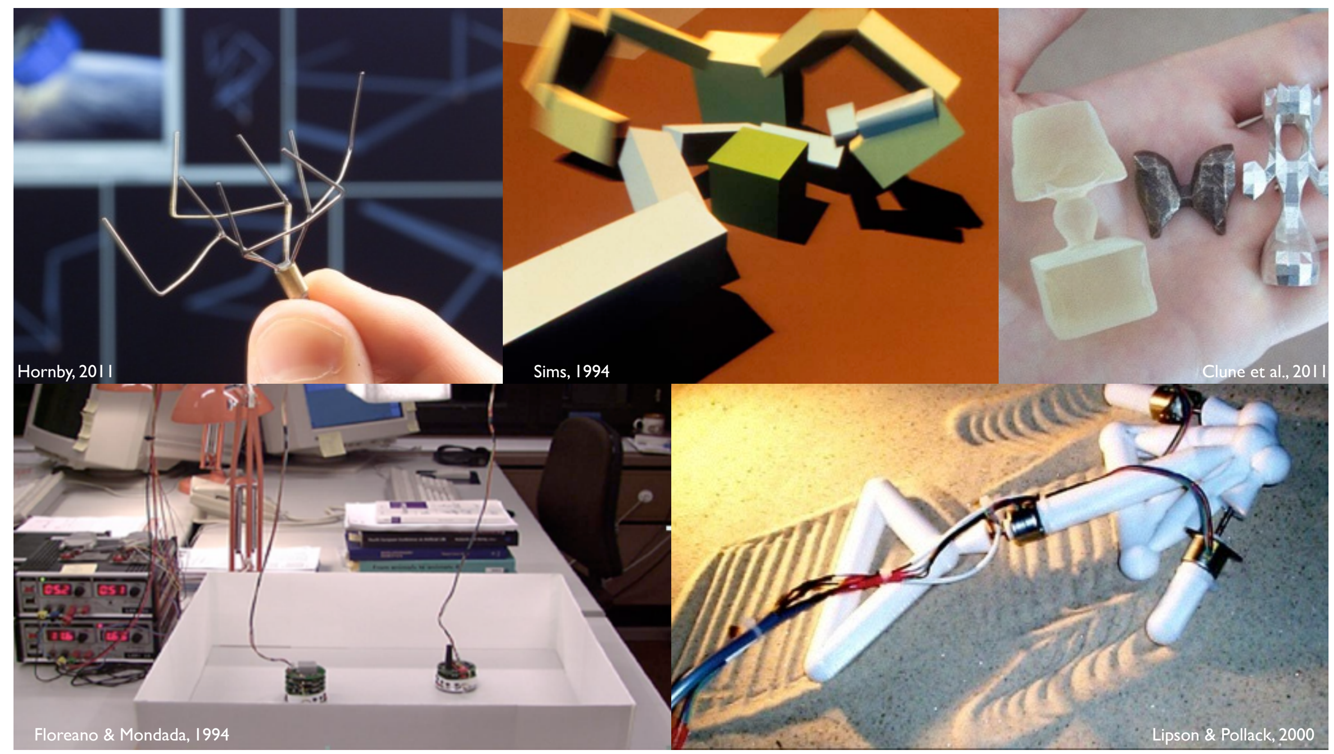
$f(\mathbf{x})$

Measured performance

Kober, J., Bagnell, J. A., and Peters, J. (2013). *The International Journal of Robotics Research*

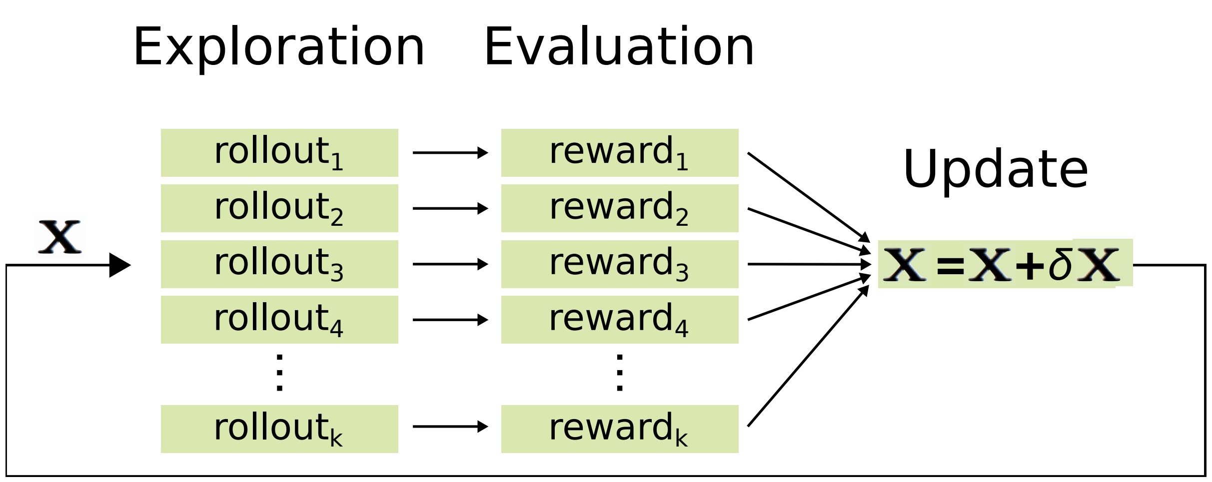
Existing Approach: Evolutionary Algorithms





Evolutionary Algorithm	Starting beh.	Learning time	robot	DOFs	Param.	reward
Chernova and Veloso (2004)	random	5 h	quadruped	12	54	external
Zykov et al. (2004)	random	2 h	hexapod	12	72	external
Berenson et al. (2005)	random	2 h	quadruped	8	36	external
Hornby et al. (2005)	non-falling	25 h	quadruped	19	21	internal
Mahdavi and Bentley (2006)	random	10 h	snake	12	1152	external
Barfoot et al. (2006)	random	10 h	hexapod	12	135	external
Yosinski et al. (2011)	random	2 h	quadruped	9	5	external

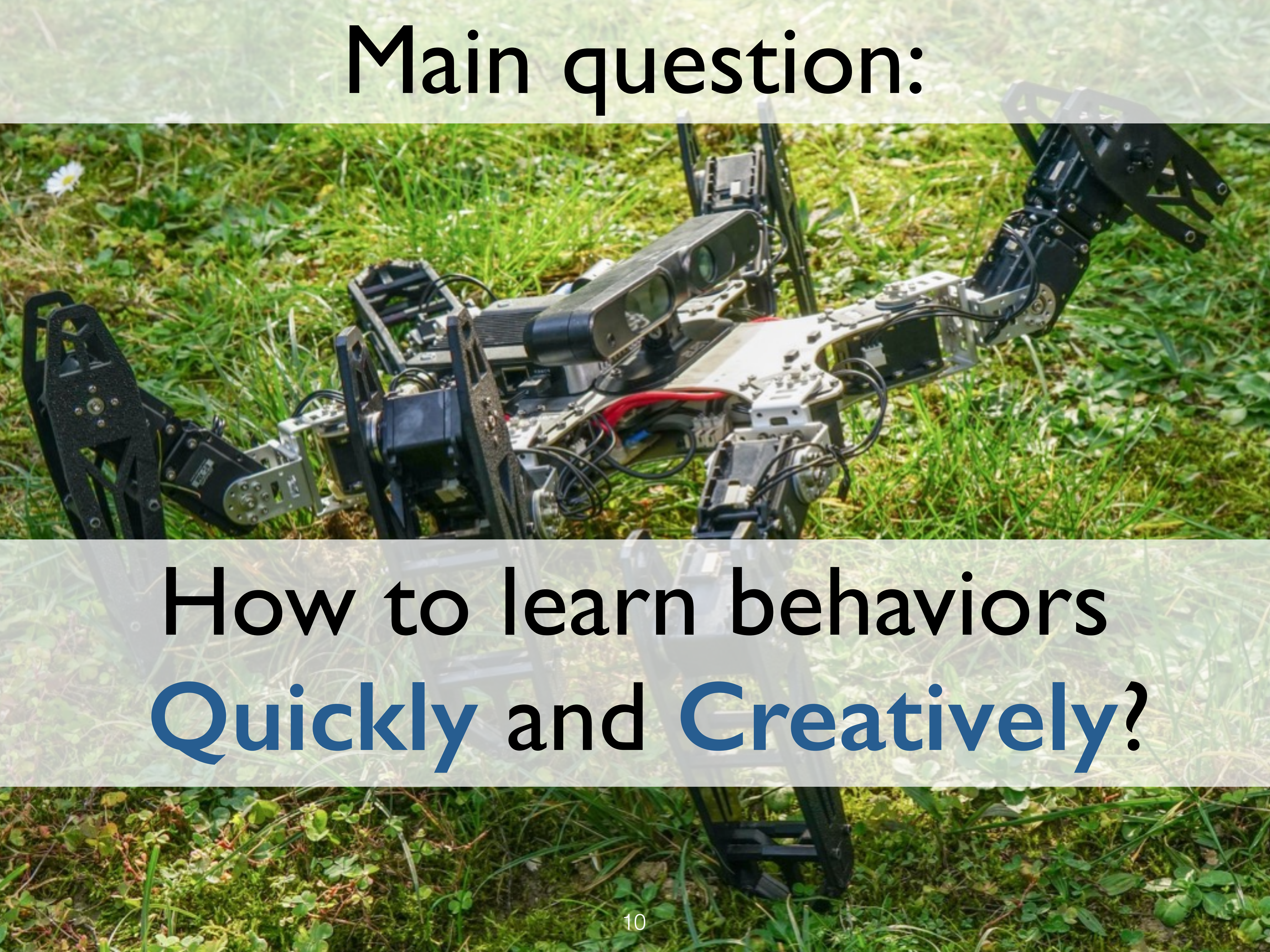
Existing Approaches: Policy Search



Fast but:

- Local search methods
- Small search space

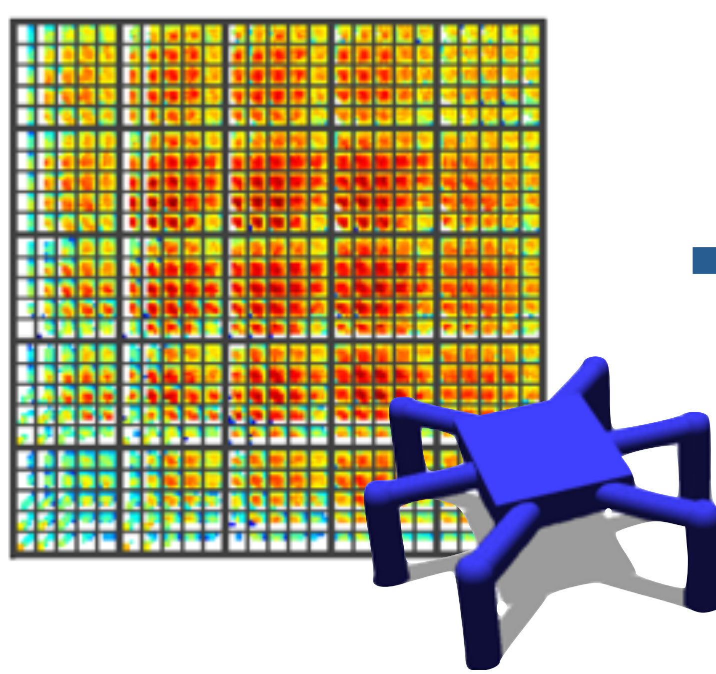
Policy Search Methods	Starting beh.	Learning time	robot	DOFs	Param.	reward
Kimura et al. (2001)	no info	80 min.	quadruped	8	72	internal
Kohl and Stone (2004)	walking	3 h	quadruped	12	12	external
Lizotte et al. (2007)	center	2h	quadruped	12	15	internal
Calendra et al. (2014)	random	46 min. / 6-9h	biped	4	4	external
Weingarten et al. (2004)	walking	> 15 h	hexapod	6	8	external
Sproewitz et al. (2008)	random	60 min.	quadruped	8	5	external
Hemker et al. (2009)	walking	3-4 h	biped	24	5	external
Barfoot et al. (2006)	random	1h	hexapod	12	135	external


Main question:
How to learn behaviors
Quickly and Creatively?

How to learn behaviors Quickly and Creatively?

Generating Behavioral Repertoires

Long Creative process



Fast adaptation

Quick process
based on Behavioral repertoires

Contribution 1

Contribution 2

Generating Behavioral Repertoires

Contribution 1

First one observation:

Learning one behavior

Classic approaches (EA,PS): optimize a single function

Learning to walk: **max(speed)** or **min(distance(target))**

Robots have to be able to perform a large variety of different behaviors

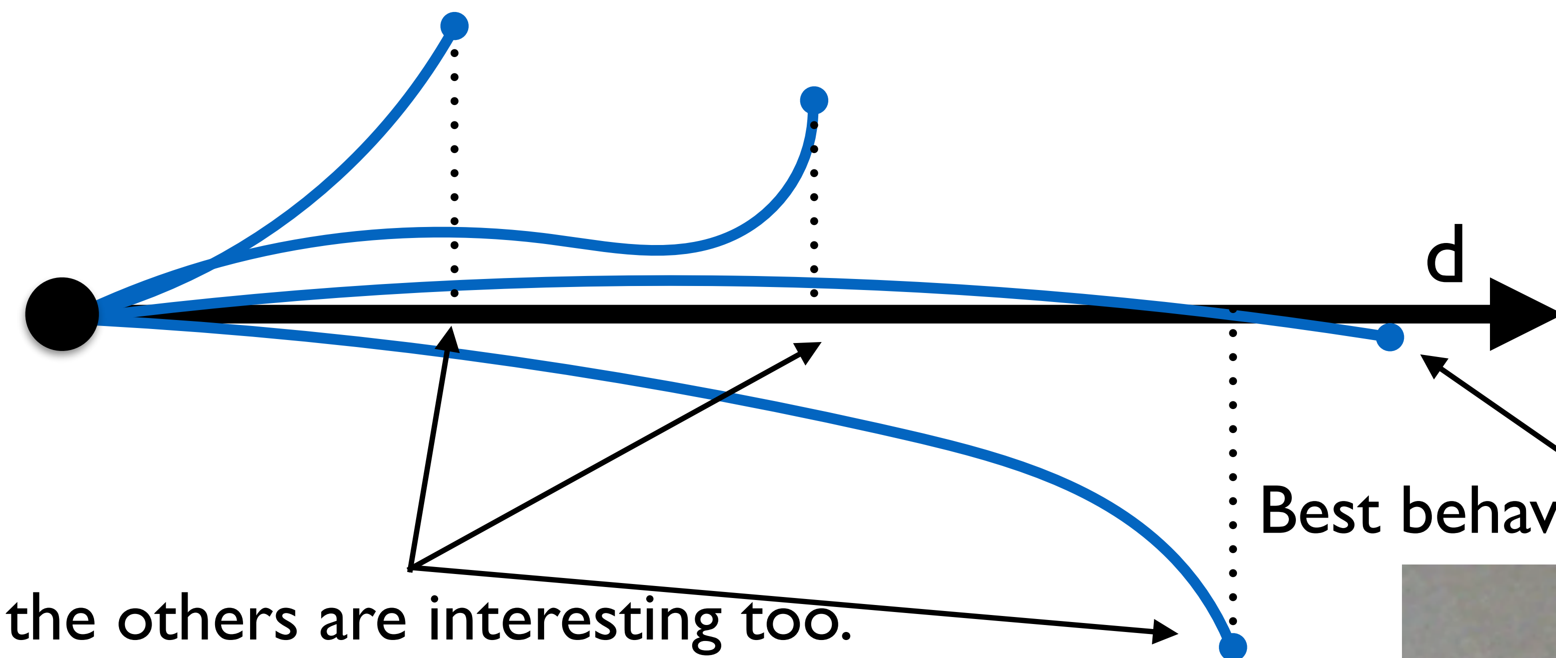
Learning them one by one?
Increases the learning time by a factor equal to the number of behaviors

Objective: Learning a large variety of actions quickly

Main Idea:

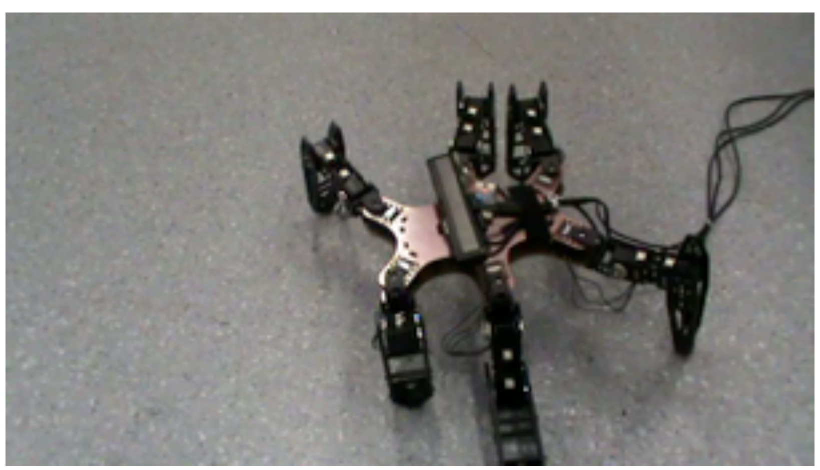
Learning all the behaviors of the repertoire simultaneously

During a classic learning process: **max(d)**



Best behavior according to **max(d)**.

But the others are interesting too.



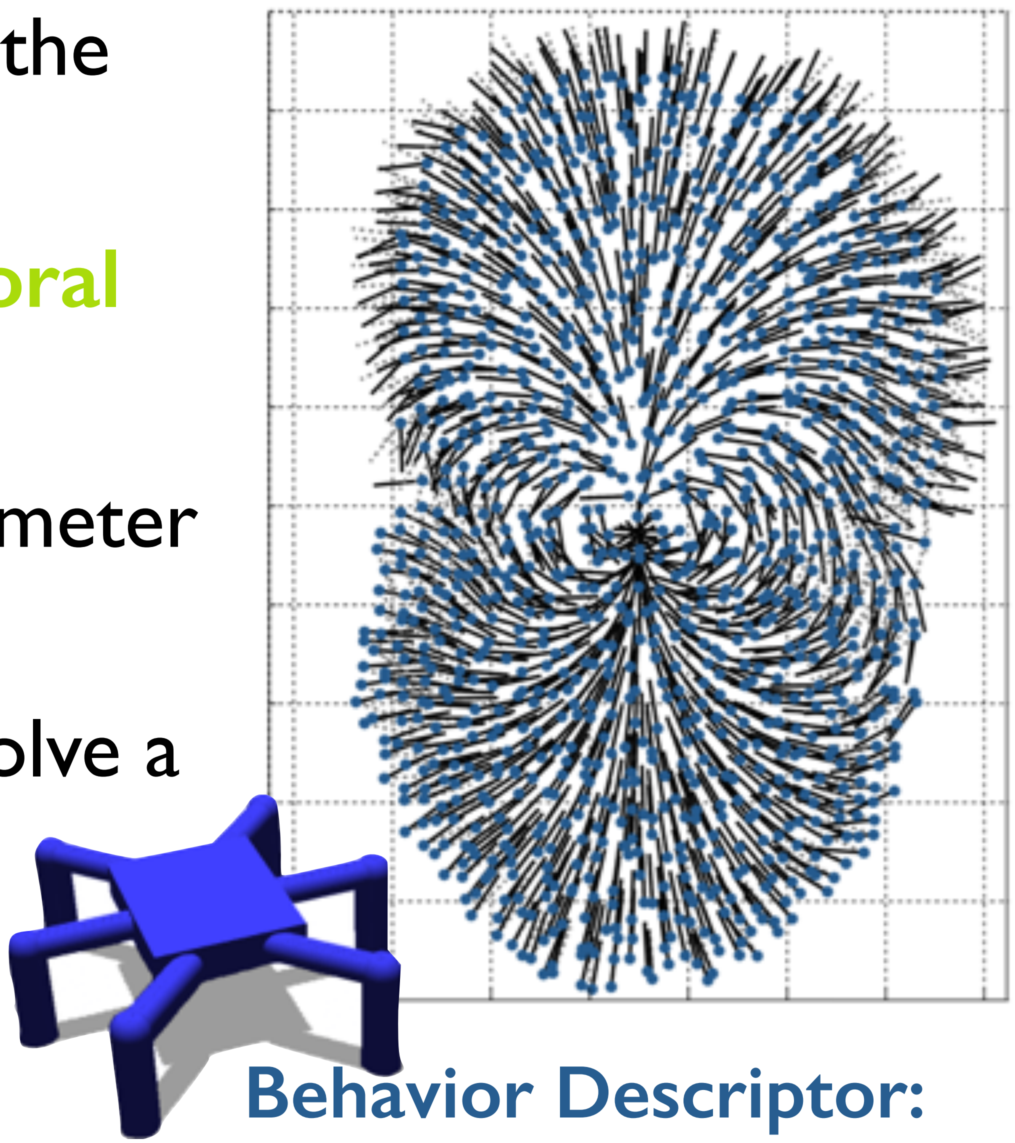
Classic algorithms find interesting behaviors but **discard** them…

... **Instead of improving them**

Recycling the creativity of algorithms

Behavioral Repertoire

- Containing (all) the possible behaviors of the robot
- Collection of behaviors sorted by a behavioral descriptor
- Behavioral descriptors are mapped to parameter sets of controller (like an inverse model)
- Can be used by higher level algorithms to solve a task. (e.g. planning algorithm)

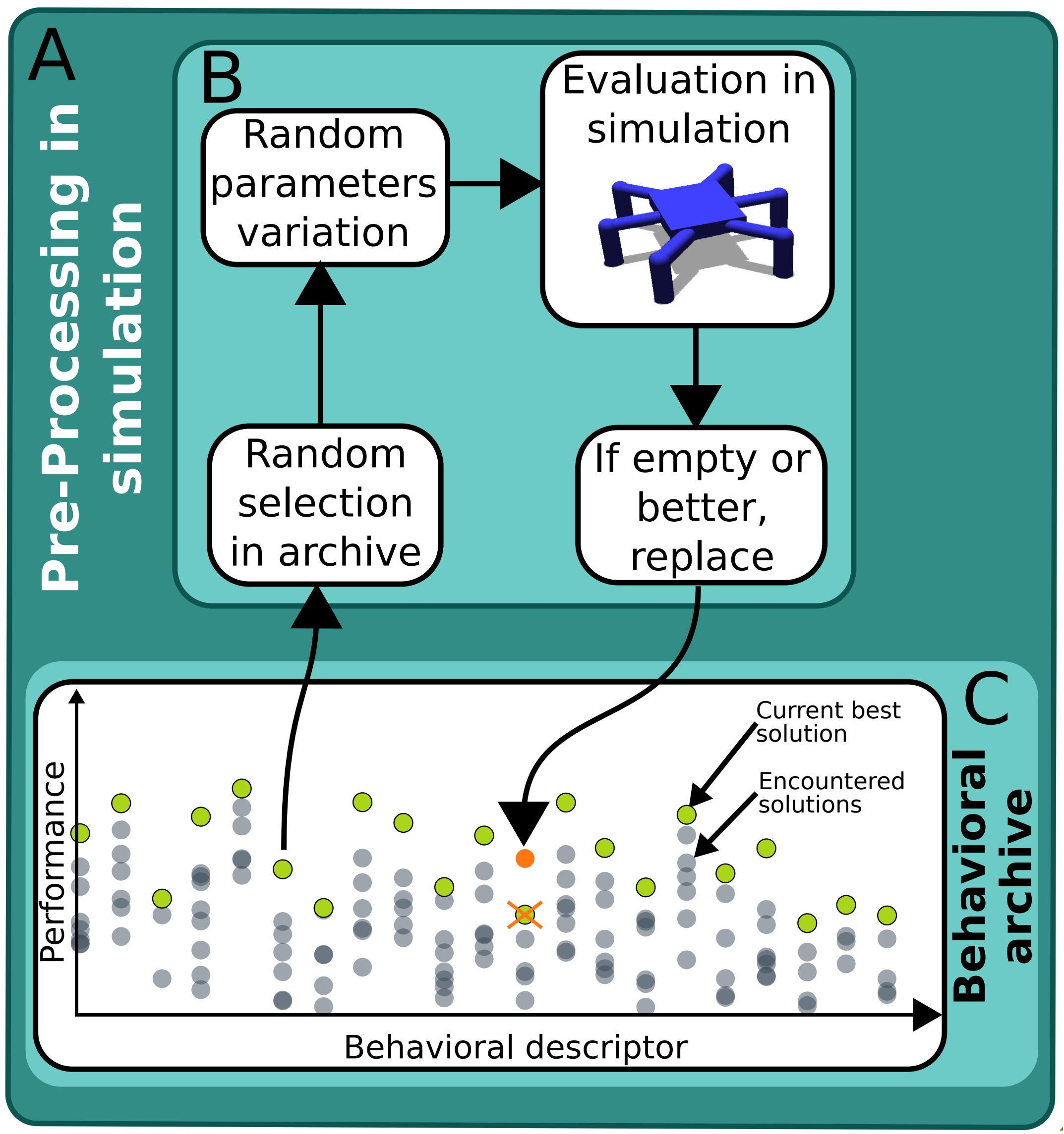


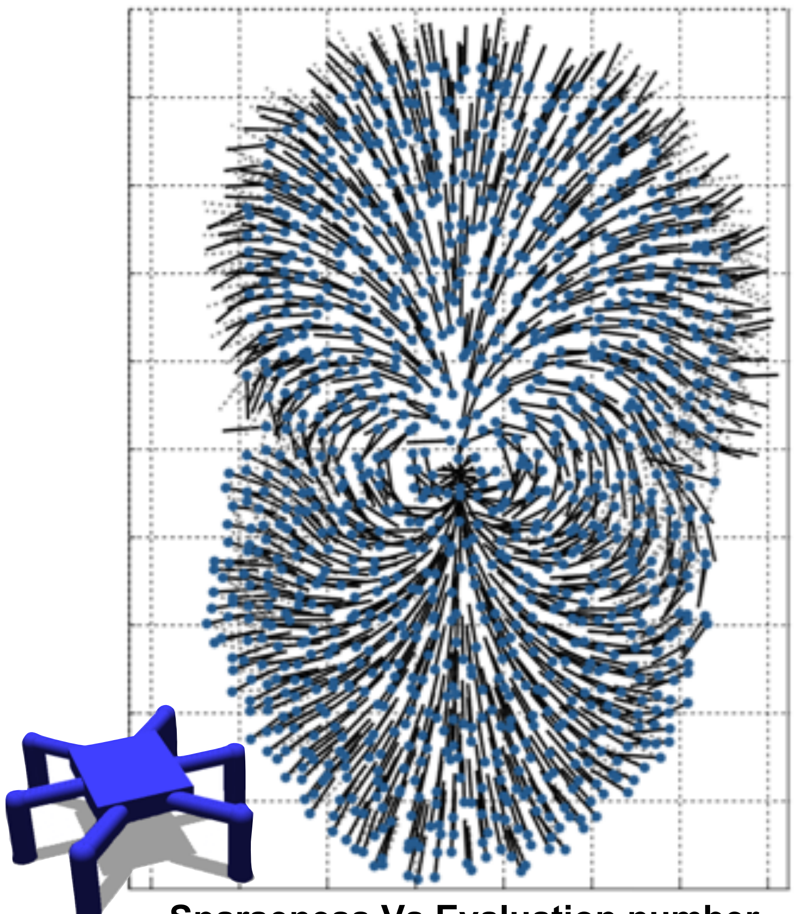
Behavior Descriptor:
(x,y) position
Performance:
Orientation Error

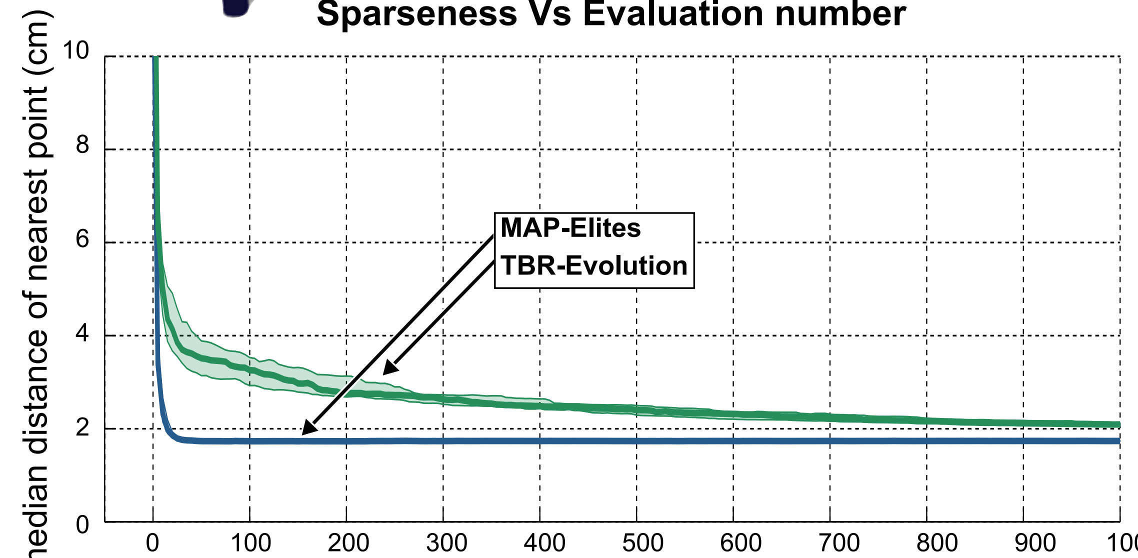
The MAP-Elites Algorithm[1]

initially used to generate plots,

can be used to generate Behavioral Repertoires[2]



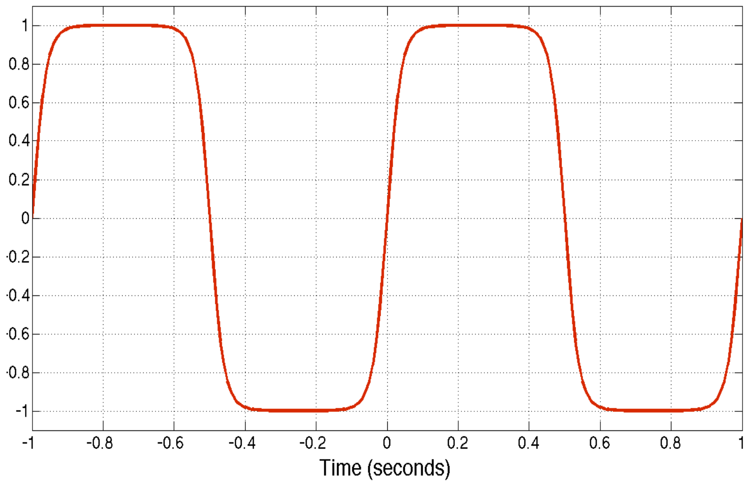




nb replicates: 40 / algorithm

[1] Mouret, J.-B. & Clune, J. (2015). ArXiv
[2] **Cully, A.**, Clune, J., Tarapore, D. & Mouret, J.-B. (2015). Nature

Finding several different ways to walk



Controller:

- 2 degrees of freedom per leg
- Amplitude, Phase, Duty Cycle
- 36 dimensions

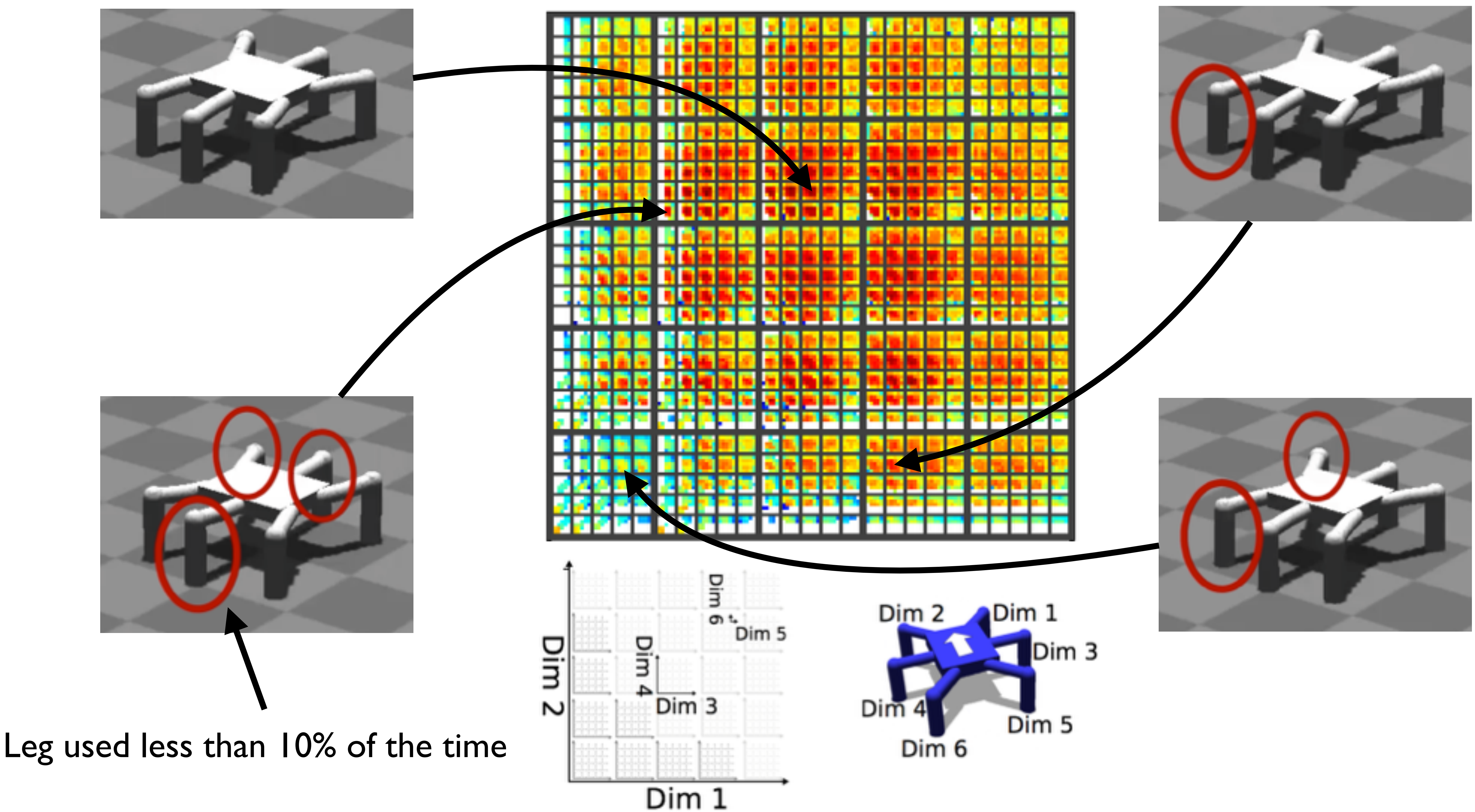
Behavioral Descriptor:

- The proportion of time that each leg touches the ground
- Discrete (0%, 25%, 50%, 75%, 100%)
- 6 dimensions

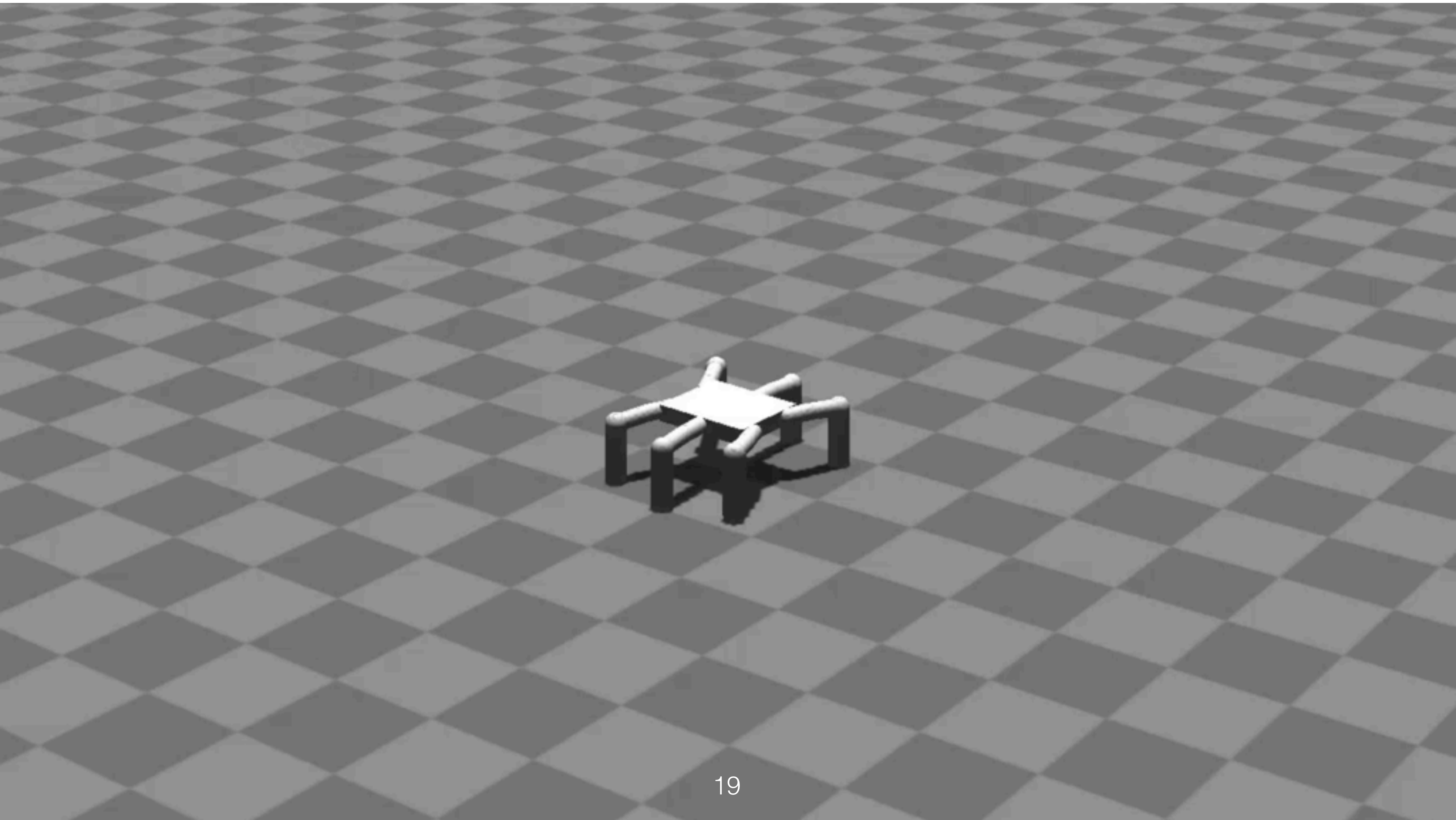
Performance function: $f(x) = \frac{pos_{front}(robot(x), t = 5s)}{5}$

40 million evaluations

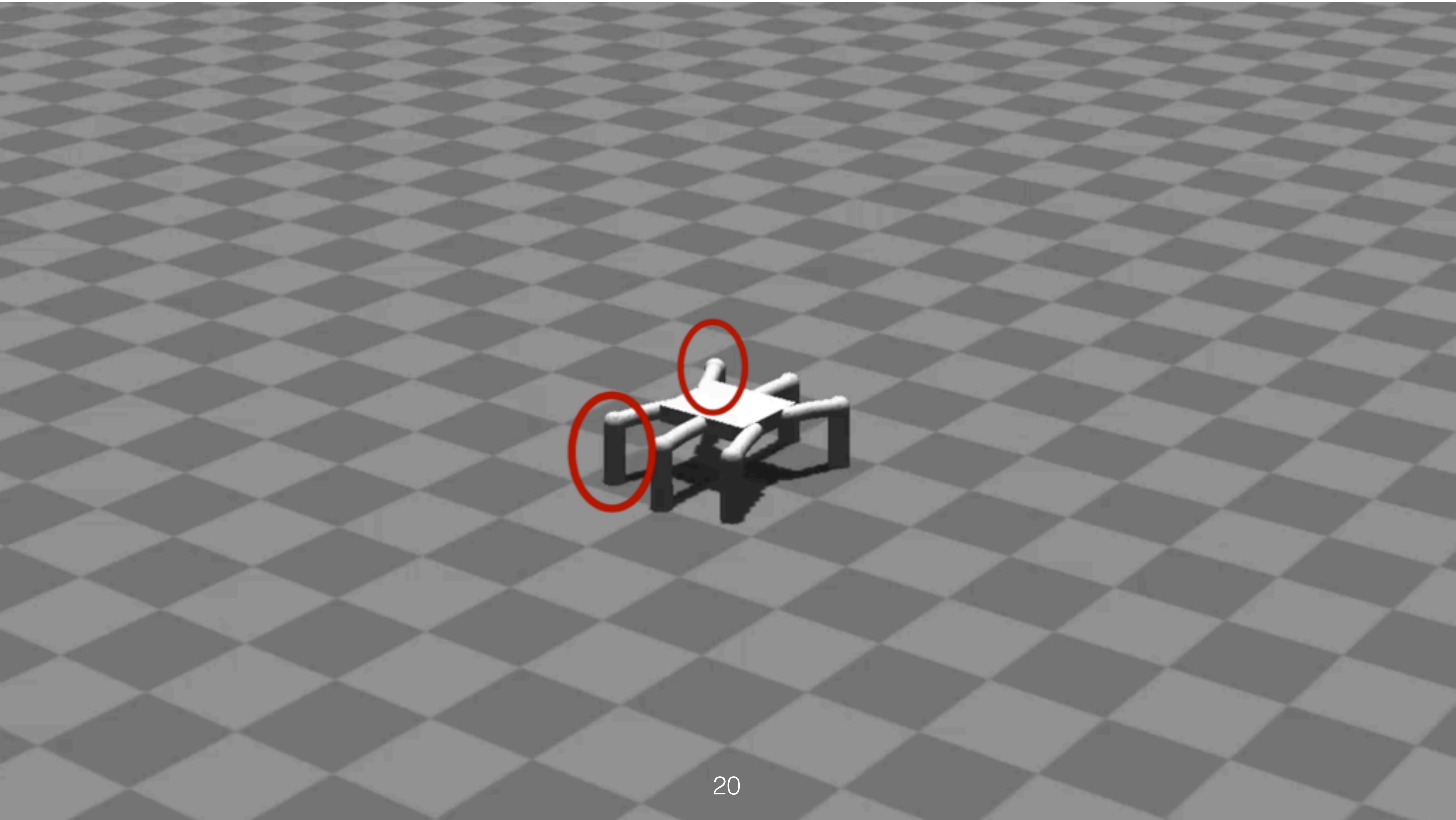
Result: 13 000 Behaviors



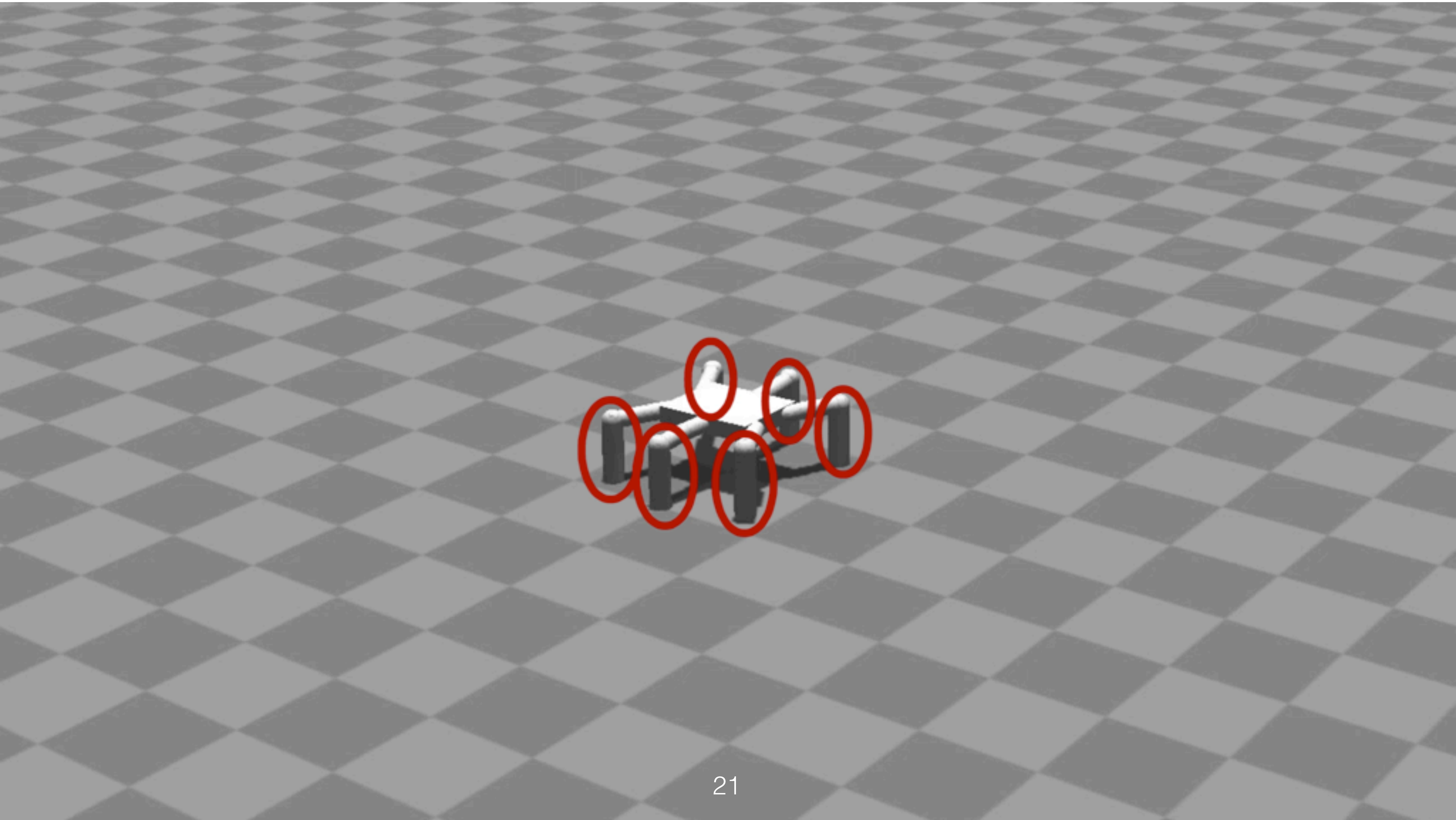
Hexapod Gait



Quadrupedal Gait

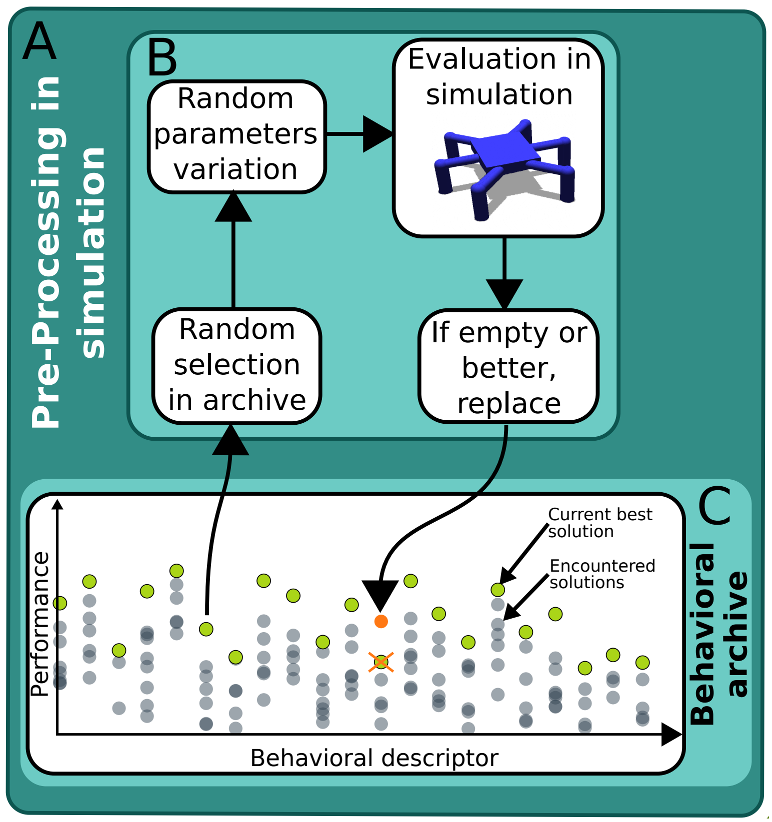


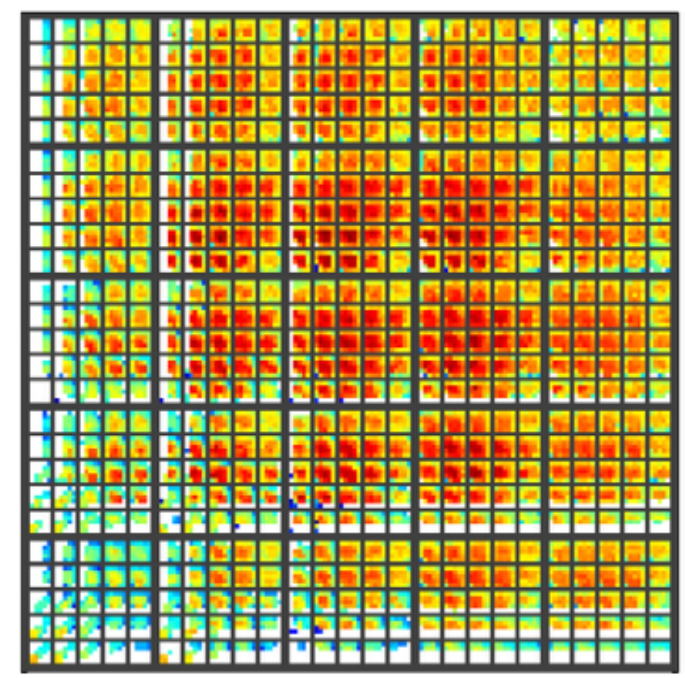
« Creative » Gait



MAP-Elites - Conclusion

MAP-Elites generates autonomously **Behavioral Repertoires** that contain **(all) the possible behaviors** of the robot



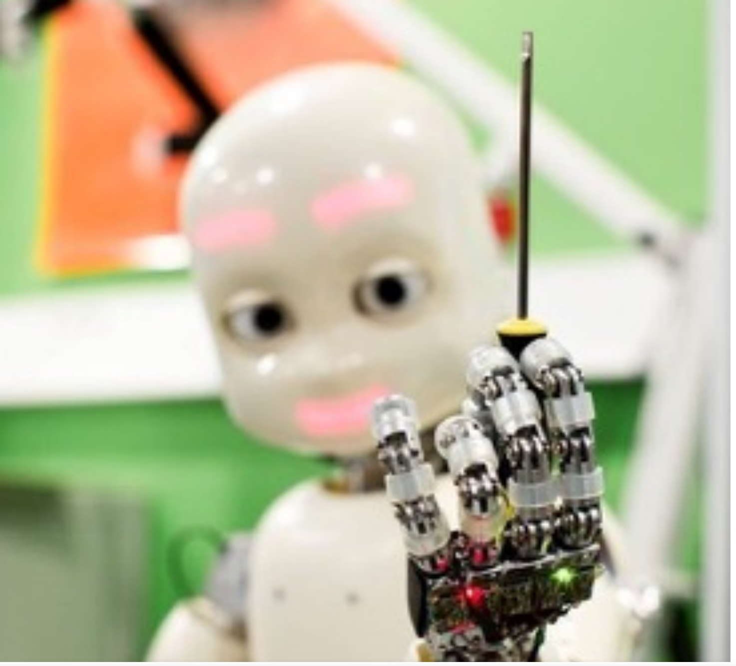


Behavioral repertoires encapsulate the **Creativity** of evolutionary algorithms

No assumption about the robot,
the controller, the behavior descriptor

Fast adaptation

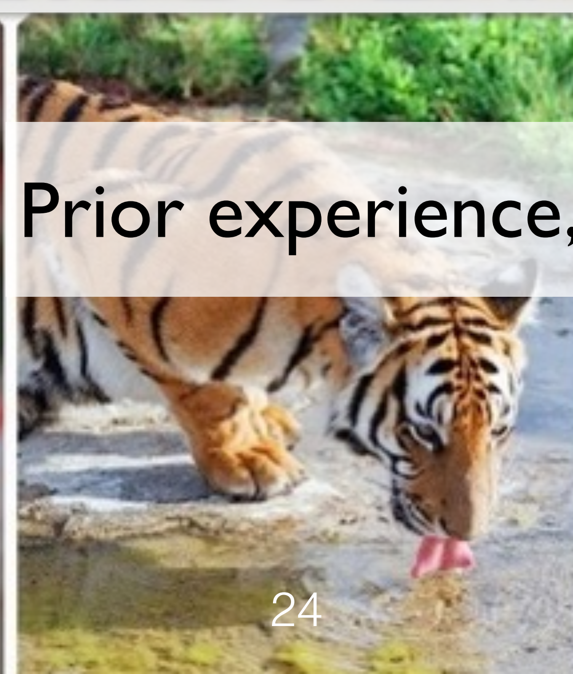
Contribution 2



While robots learn from scratch …

… animals can rely on prior knowledge

Knowledge about their body,



Prior experience,

Instincts,
Imagination,
…

Simulation can be a good source of prior knowledge

But it doesn’t take into account the situation (i.e. the damage)

Update of the simulation ~ diagnosis

How to use simulation without updating the model?

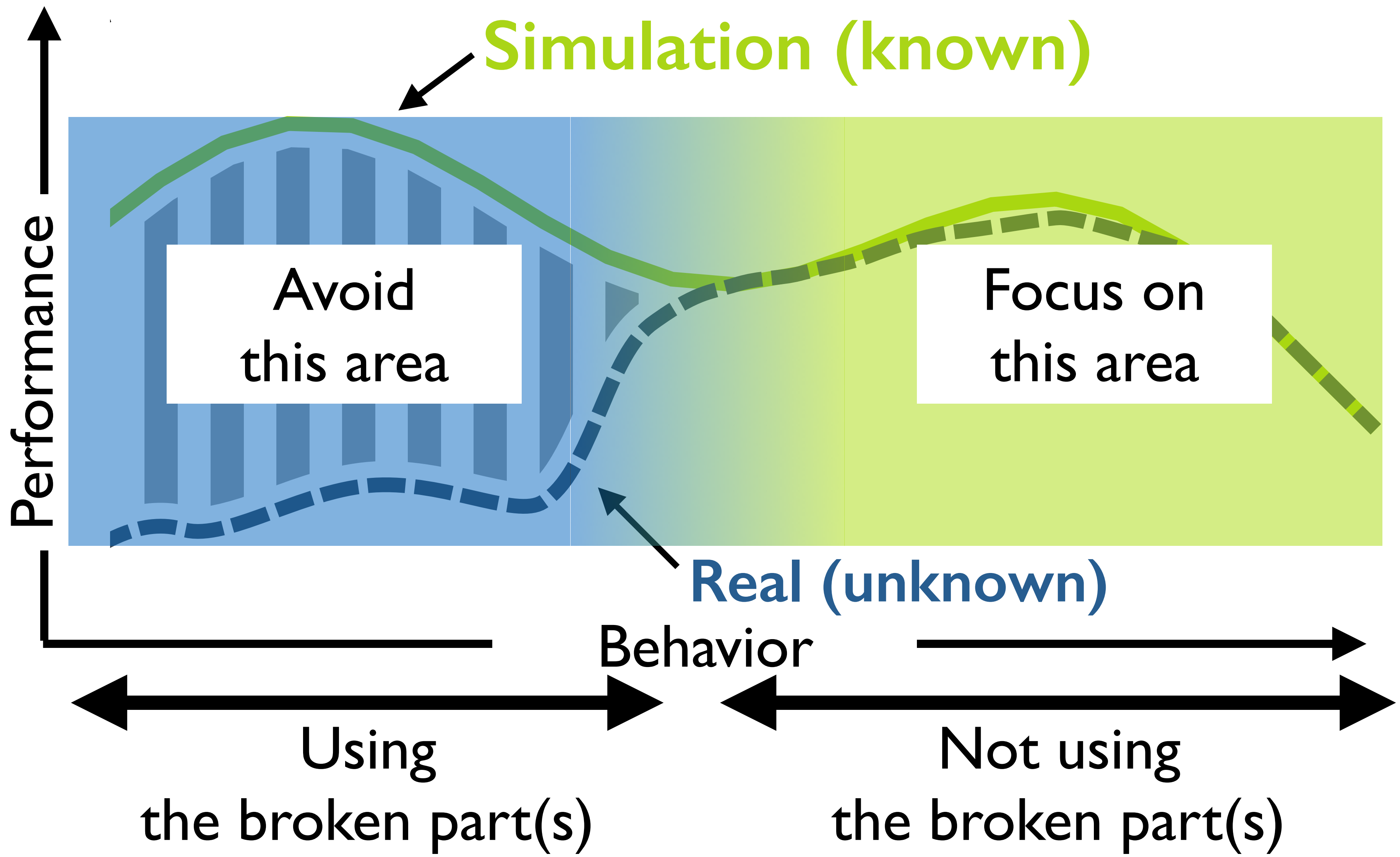
Hypothesis:

Some behaviors perform similarly on the intact robot and the damaged robot

Objective:

Finding such behaviors, in order to learn quickly without updating the simulation (no diagnosis)

Concretely



It's a simpler problem

Adaptation with behavioral repertoire

New Hypothesis:
Some behaviors work similarly on the intact robot and the damaged robot

Behavioral repertoire goal:
Gathering all the possible actions of the intact robot

Searching compensatory behaviors in the repertoire

We have some prior knowledge about this smaller search space

Searching in the repertoire using **Bayesian Optimization**[1,2,3]

- Optimization algorithm of very **expensive black-box** functions
- Method based on surrogate model and probabilistic distribution (Similar to Kriging or EGO)
- Active learning with exploitation/exploration tradeoff
- State of the art of learning (policy search) techniques

120 Evaluations[2]
(15 parameters)

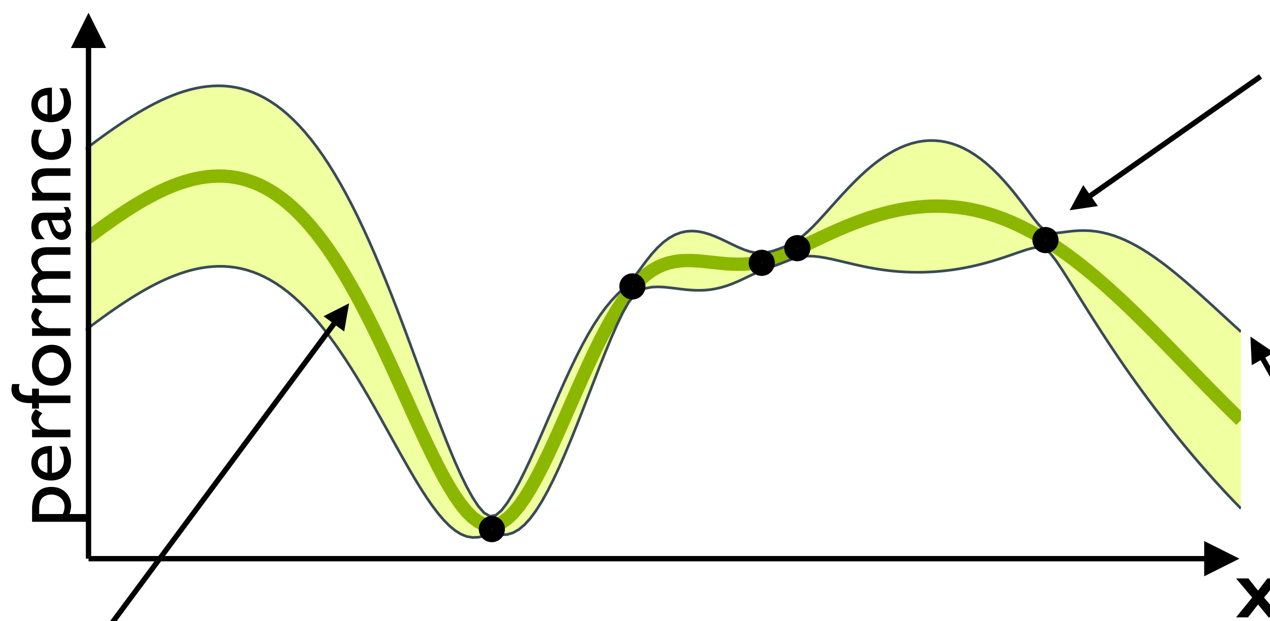
40 Evaluations[3]
(4 parameters)

[1] Brochu, Cora & De Freitas. arXiv 2010
[2] Lizotte, Wang, Bowling & Schuurmans. IJCAI 2007
[3] Calandra et al. ICRA 2014

Gaussian Process (GP)

- Generalize performance over the evaluated solutions (regression)
- Models the uncertainty
- Commonly used as model for Bayesian optimization

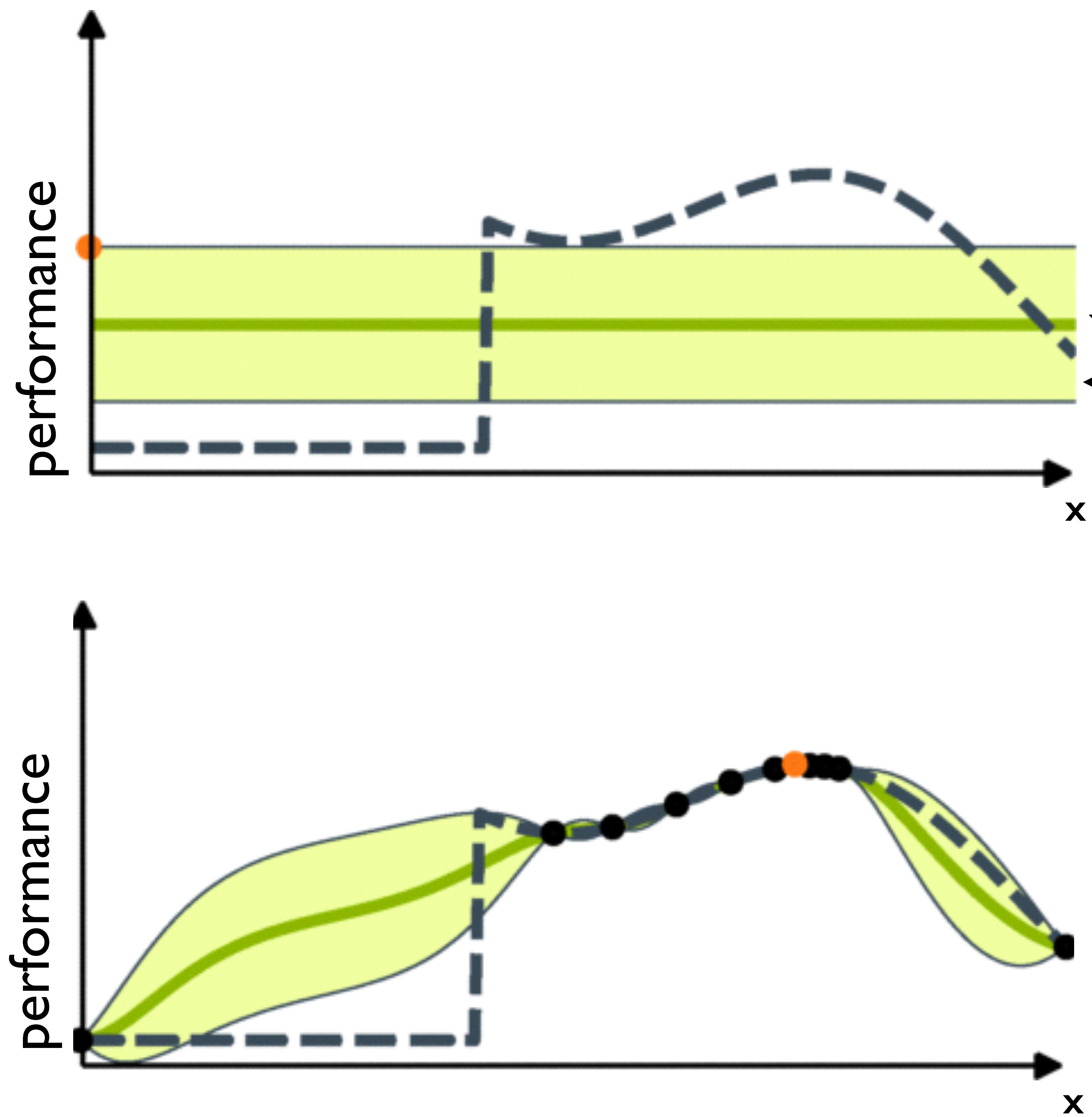
Evaluated Solution



μ(x): mean of the GP
Most expected performance

σ(x): Variance of the GP
Uncertainty of the performance

Bayesian Optimization



$$P(f(\mathbf{x})|\mathbf{P}_{1:t+1}\ ,\mathbf{x}) = \mathcal{N}(\mu_{t+1}(\mathbf{x}), \sigma^2_{t+1}(\mathbf{x}))$$

where

$$\mu_{t+1}(\mathbf{x}) = \mu_0 + \mathbf{k}^t\mathbf{K}^{-1}(\mathbf{P}_{1:t+1} - \mu_0)$$

$$\sigma^2_{t+1}(\mathbf{x}) = k(\mathbf{x},\mathbf{x}) - \mathbf{k}^t\mathbf{K}^{-1}\mathbf{k}$$

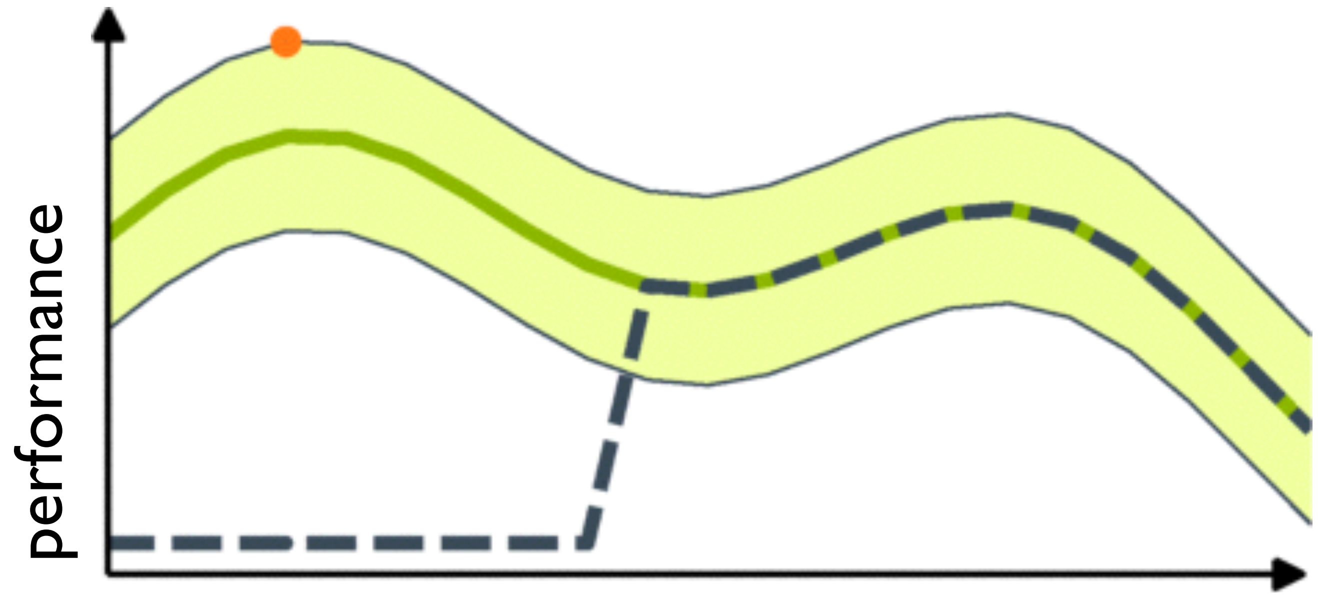
$$\mathbf{K} = \begin{bmatrix} k(\mathbf{y}_1,\mathbf{y}_1)+\sigma^2_{noise} & \cdots & k(\mathbf{y}_1,\mathbf{y}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{y}_t,\mathbf{y}_1) & \cdots & k(\mathbf{y}_t,\mathbf{y}_t)+\sigma^2_{noise} \end{bmatrix}$$

$$\mathbf{k} = \begin{bmatrix} k(\mathbf{x},\mathbf{y}_1) & k(\mathbf{x},\mathbf{y}_2) & \cdots & k(\mathbf{x},\mathbf{y}_t) \end{bmatrix}$$

Classic Kernel Function:

$$k(\mathbf{x}_i,\mathbf{x}_j) = \exp\left(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$$

Using our **prior knowledge** regarding the behavioral repertoire



$$P(f(\mathbf{x})|\mathbf{P}_{1:t+1}, \mathbf{x}) = \mathcal{N}(\mu_{t+1}(\mathbf{x}), \sigma^2_{t+1}(\mathbf{x}))$$

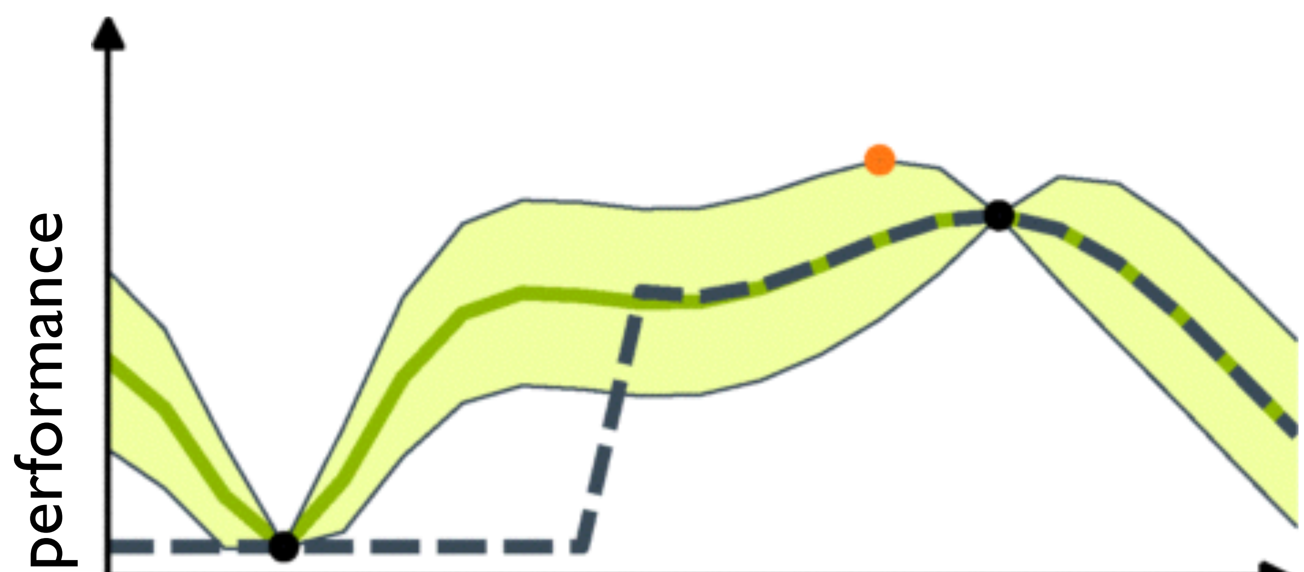
where

$$\mu_{t+1}(\mathbf{x}) = \mathcal{A}(\mathbf{x}) + \mathbf{k}^t\mathbf{K}^{-1}(\mathbf{P}_{1:t+1} - \mathcal{A}(\mathbf{y}_{1:t+1}))$$

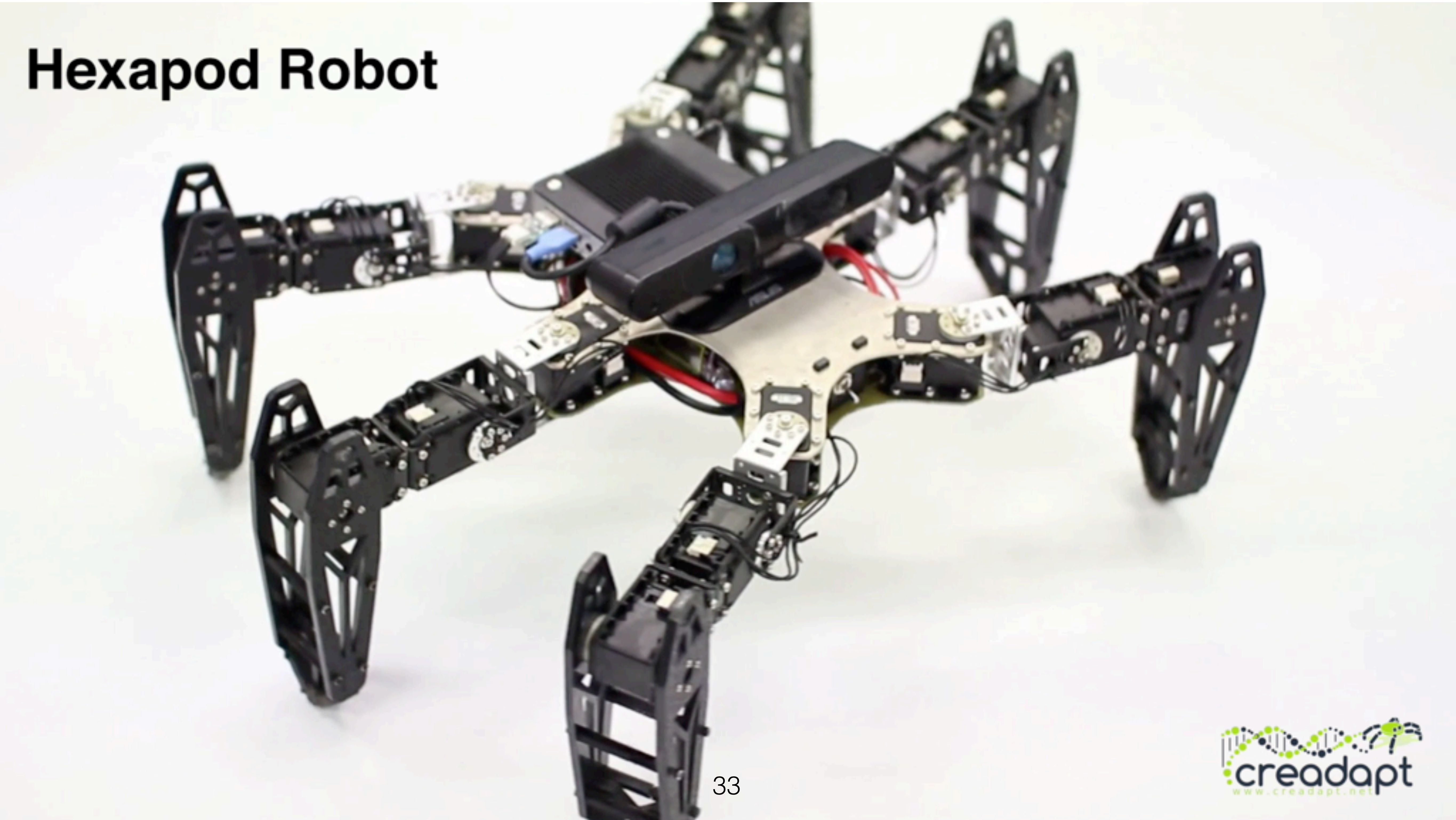
$$\sigma^2_{t+1}(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}^t\mathbf{K}^{-1}\mathbf{k}$$

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{y}_1, \mathbf{y}_1) + \sigma^2_{noise} & \cdots & k(\mathbf{y}_1, \mathbf{y}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{y}_t, \mathbf{y}_1) & \cdots & k(\mathbf{y}_t, \mathbf{y}_t) + \sigma^2_{noise} \end{bmatrix}$$

$$\mathbf{k} = \begin{bmatrix} k(\mathbf{x}, \mathbf{y}_1) & k(\mathbf{x}, \mathbf{y}_2) & \cdots & k(\mathbf{x}, \mathbf{y}_t) \end{bmatrix}$$

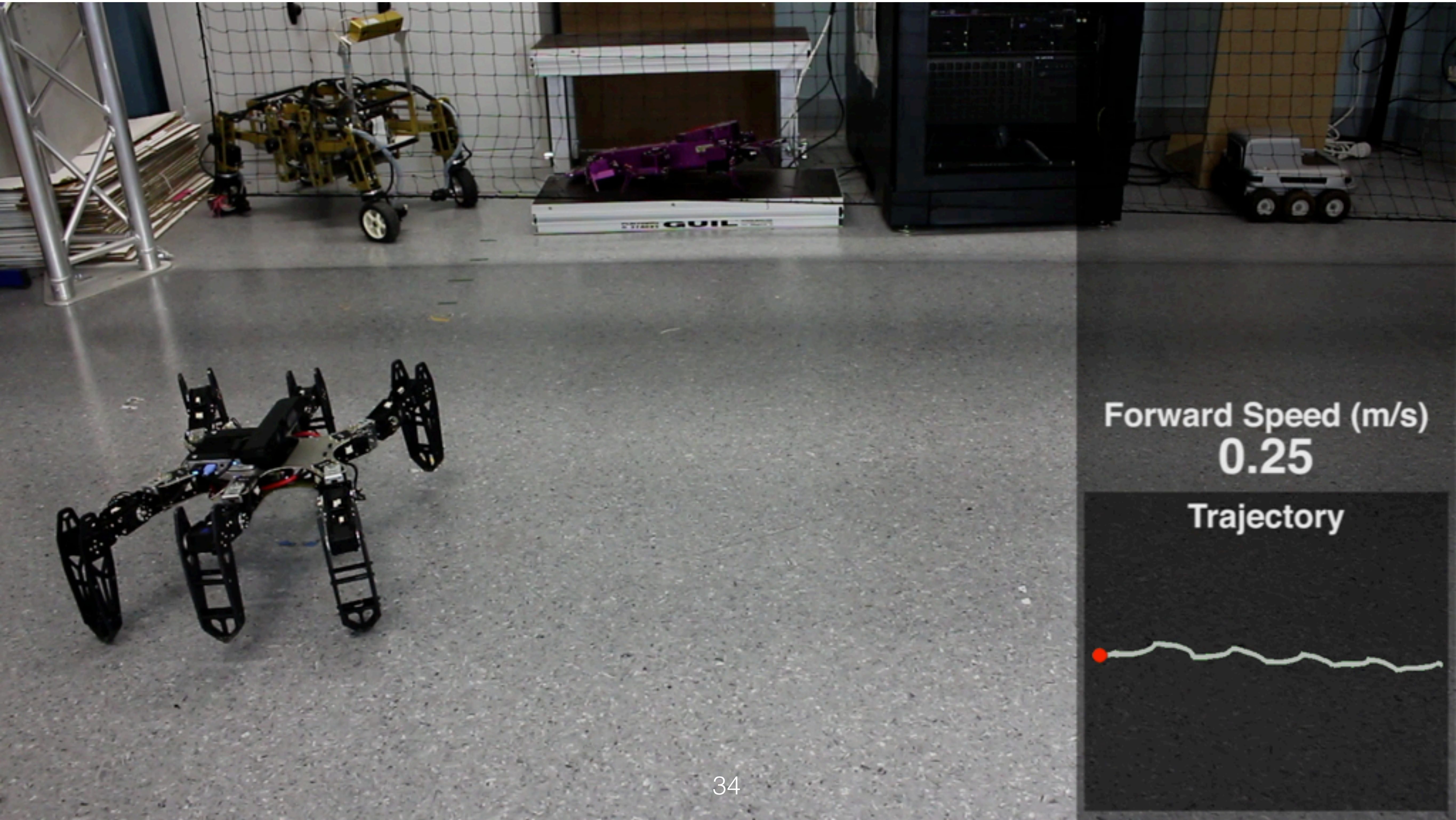


Application on a robot



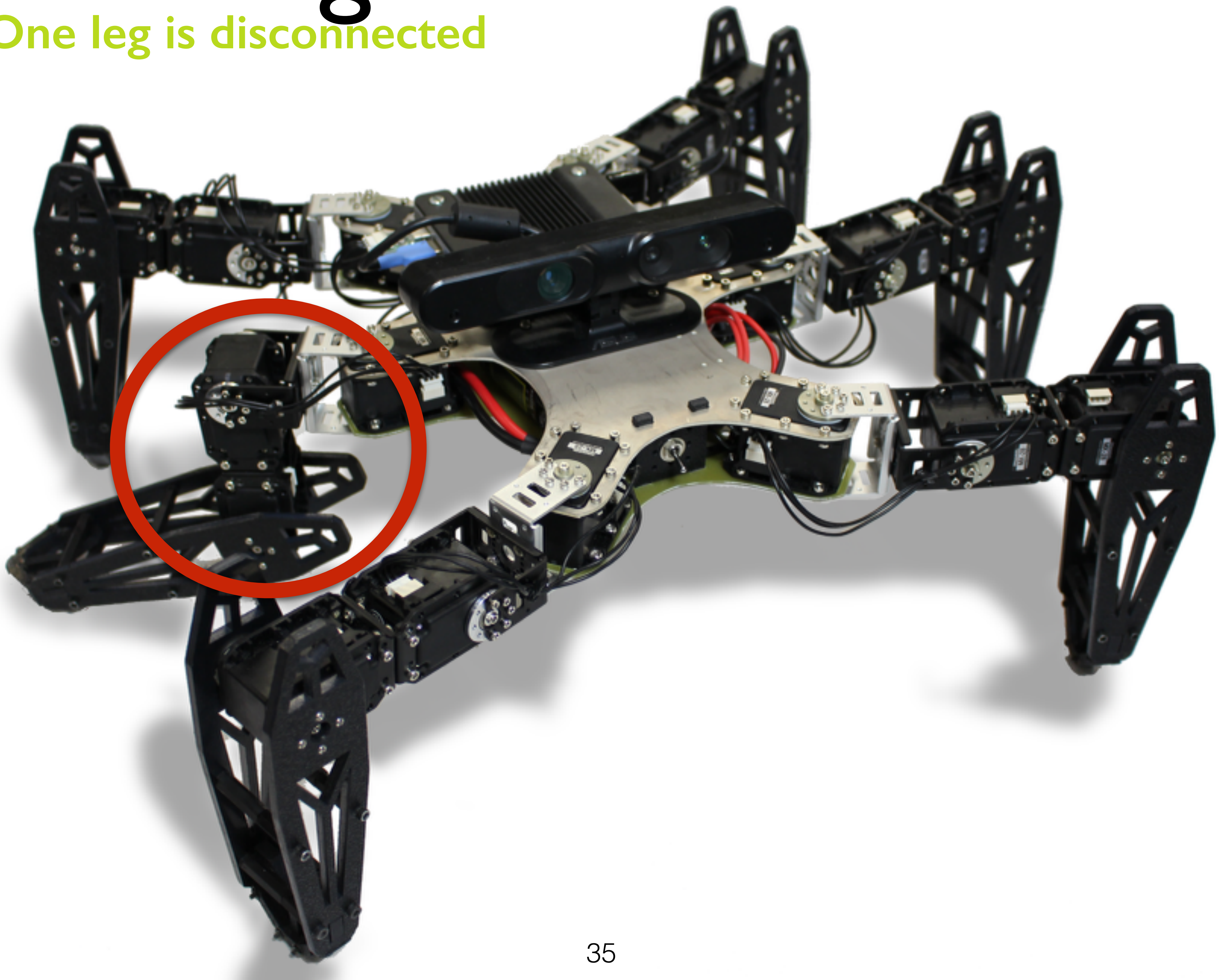
Classic Hexapod Gait

Open-loop controller - Tripod Gait

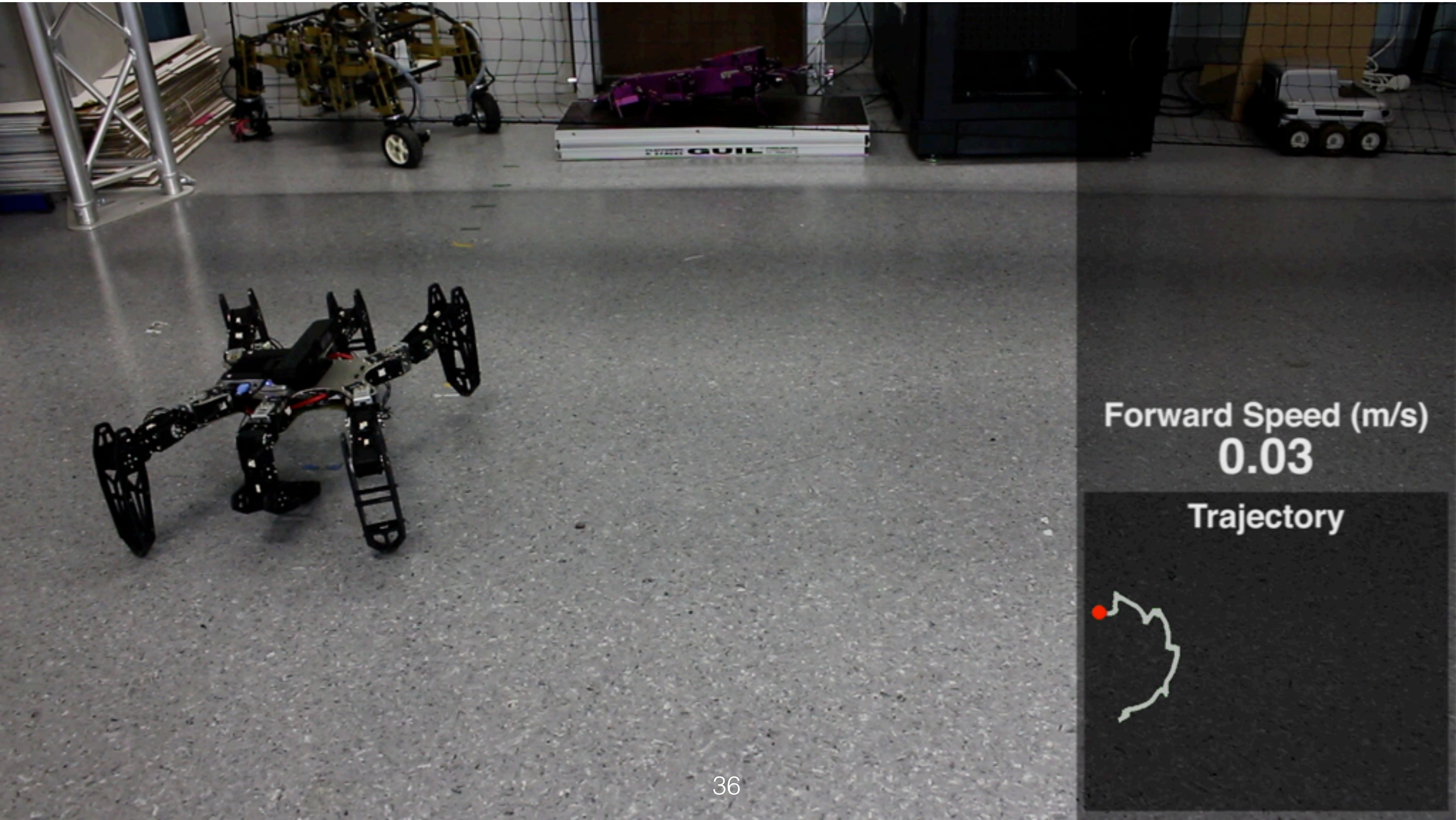


A damage occurs …

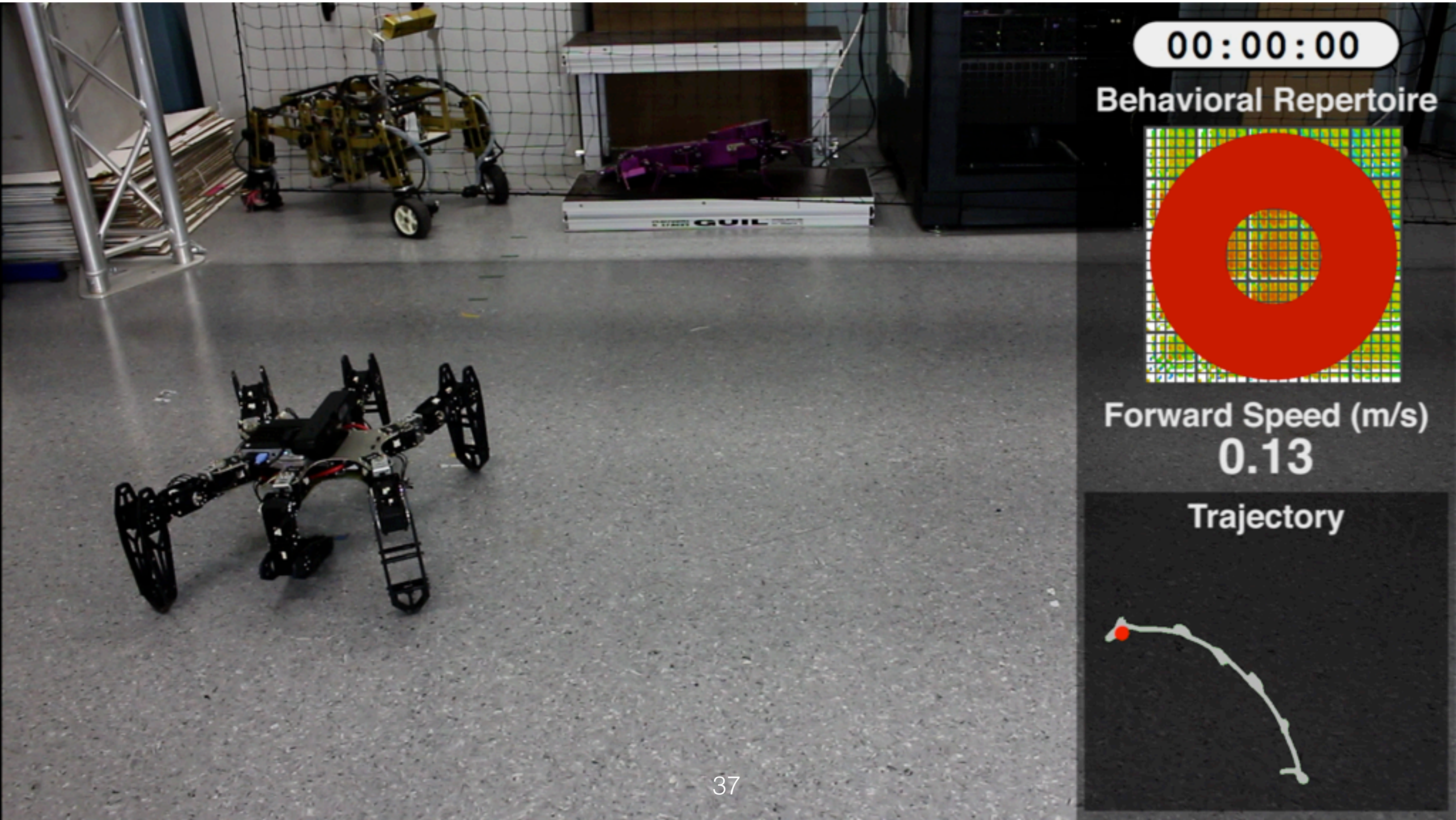
One leg is disconnected



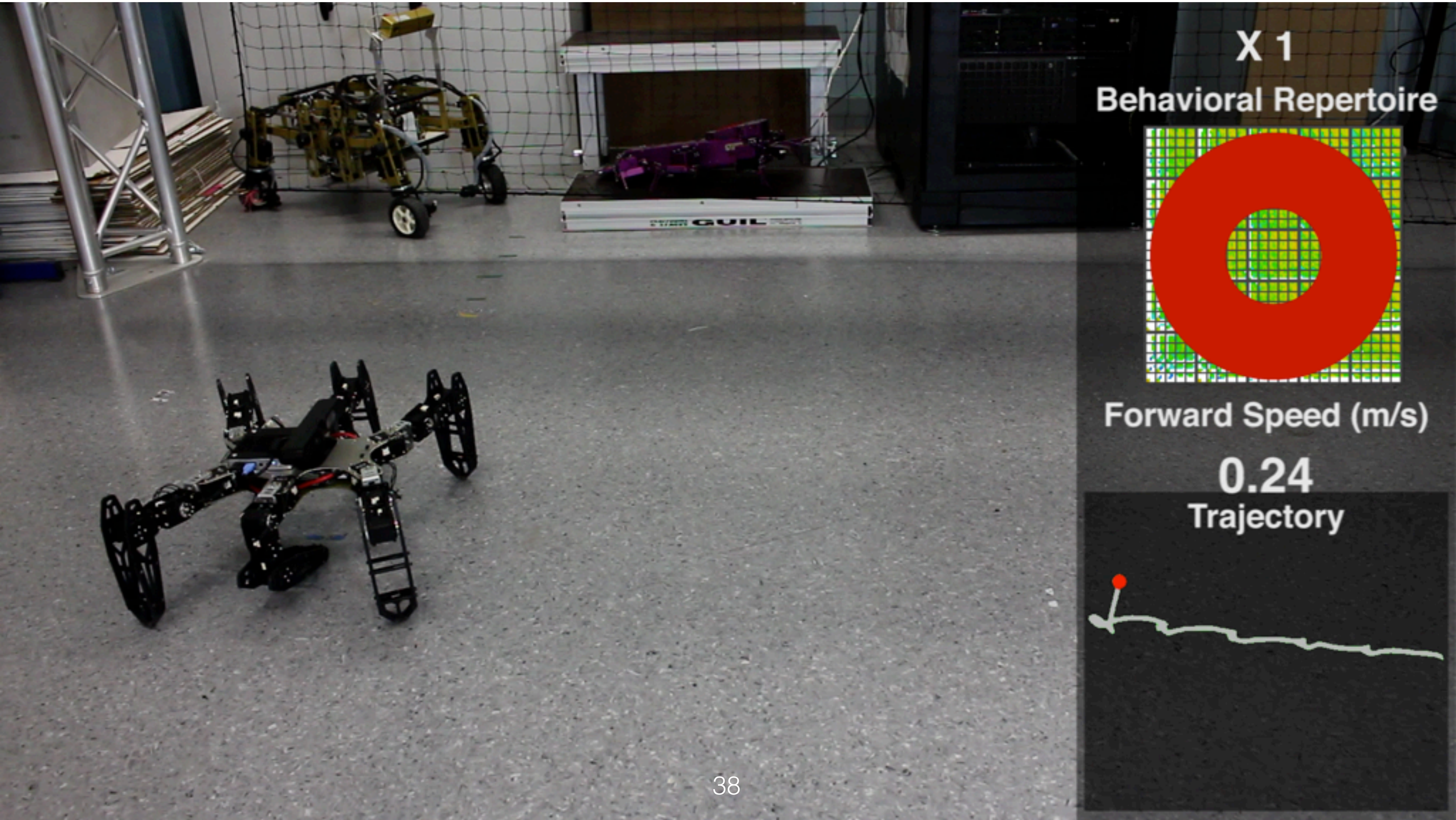
The behavior is seriously affected



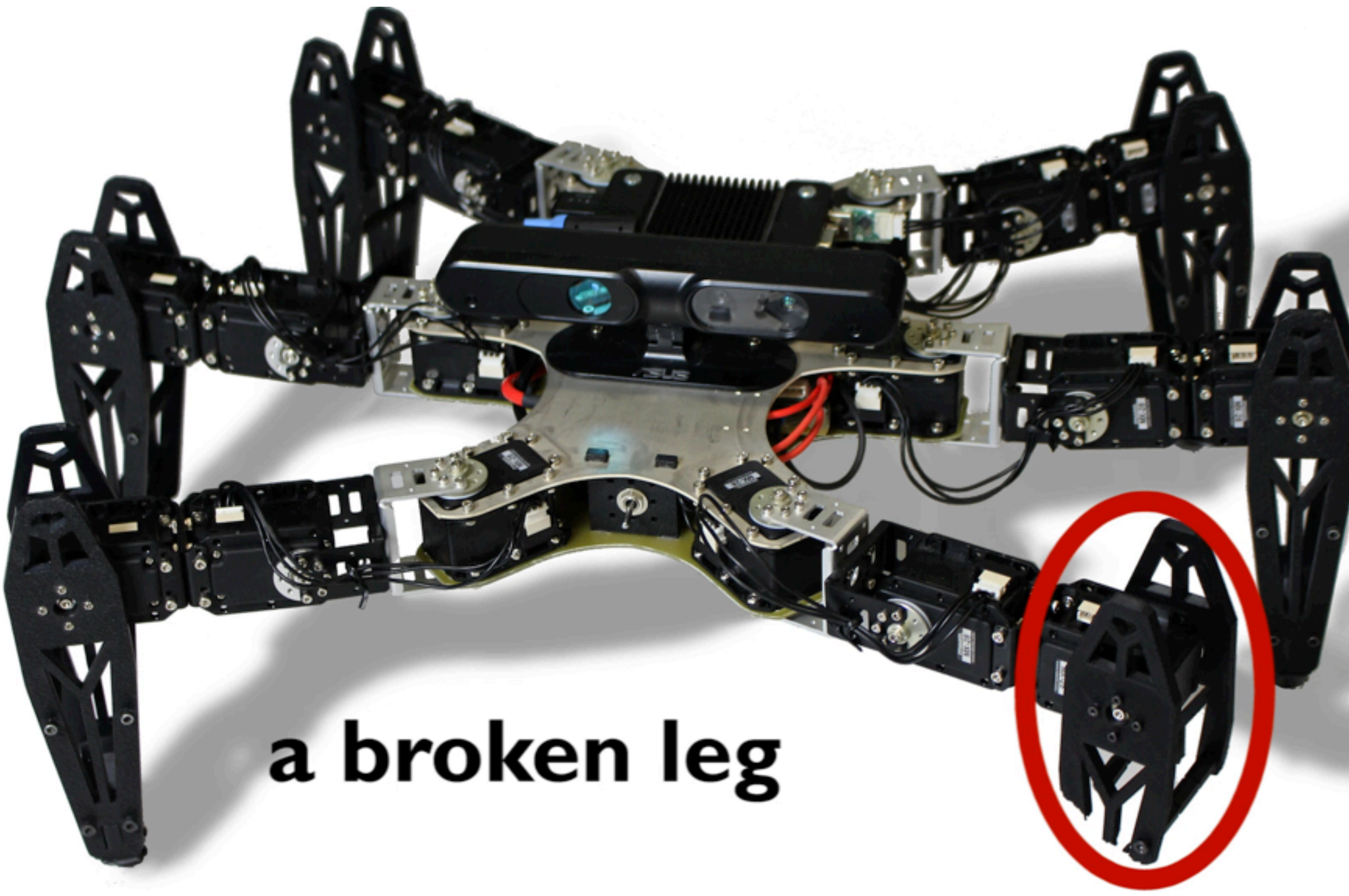
Learning process



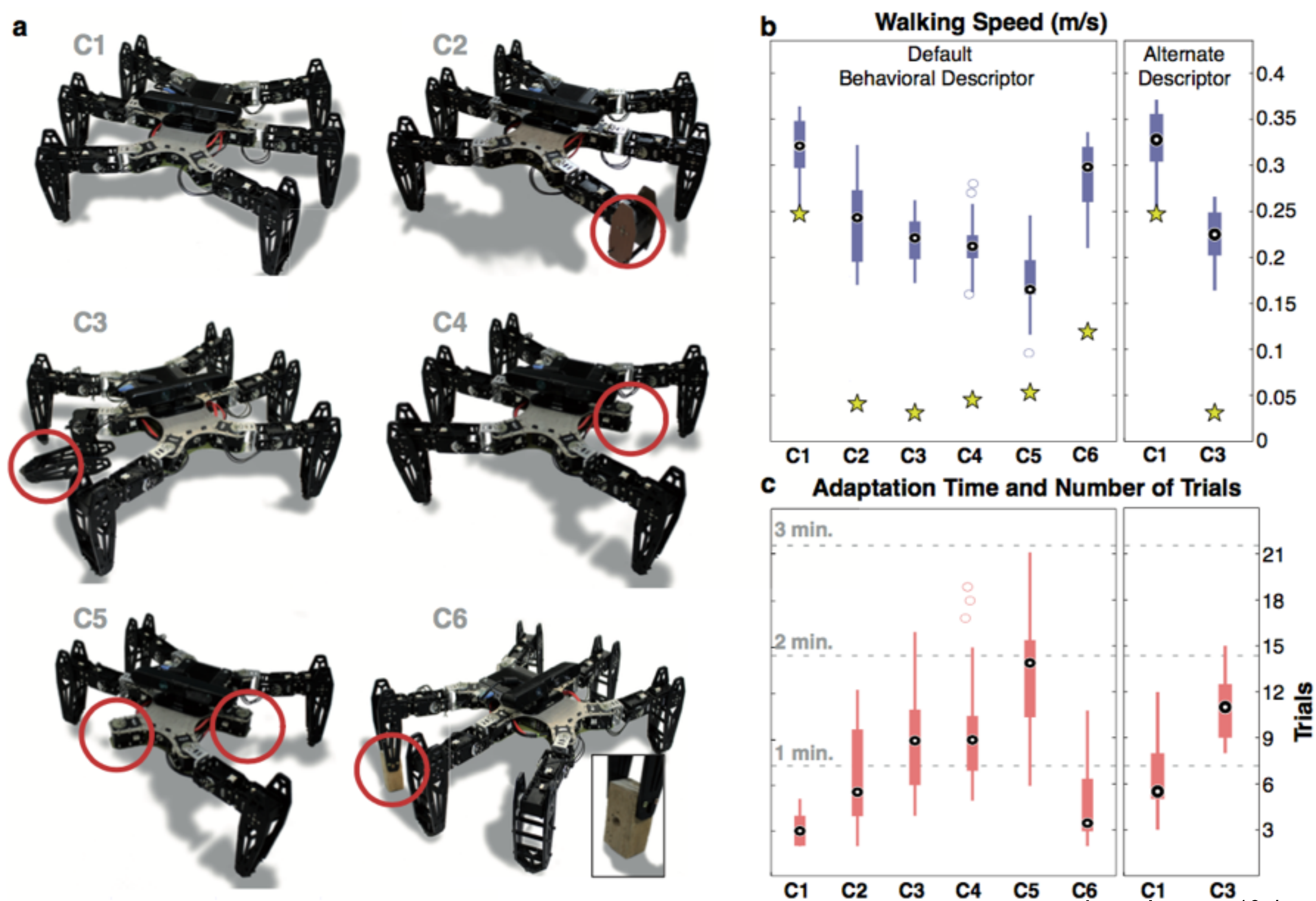
Result after 40 seconds



Other examples



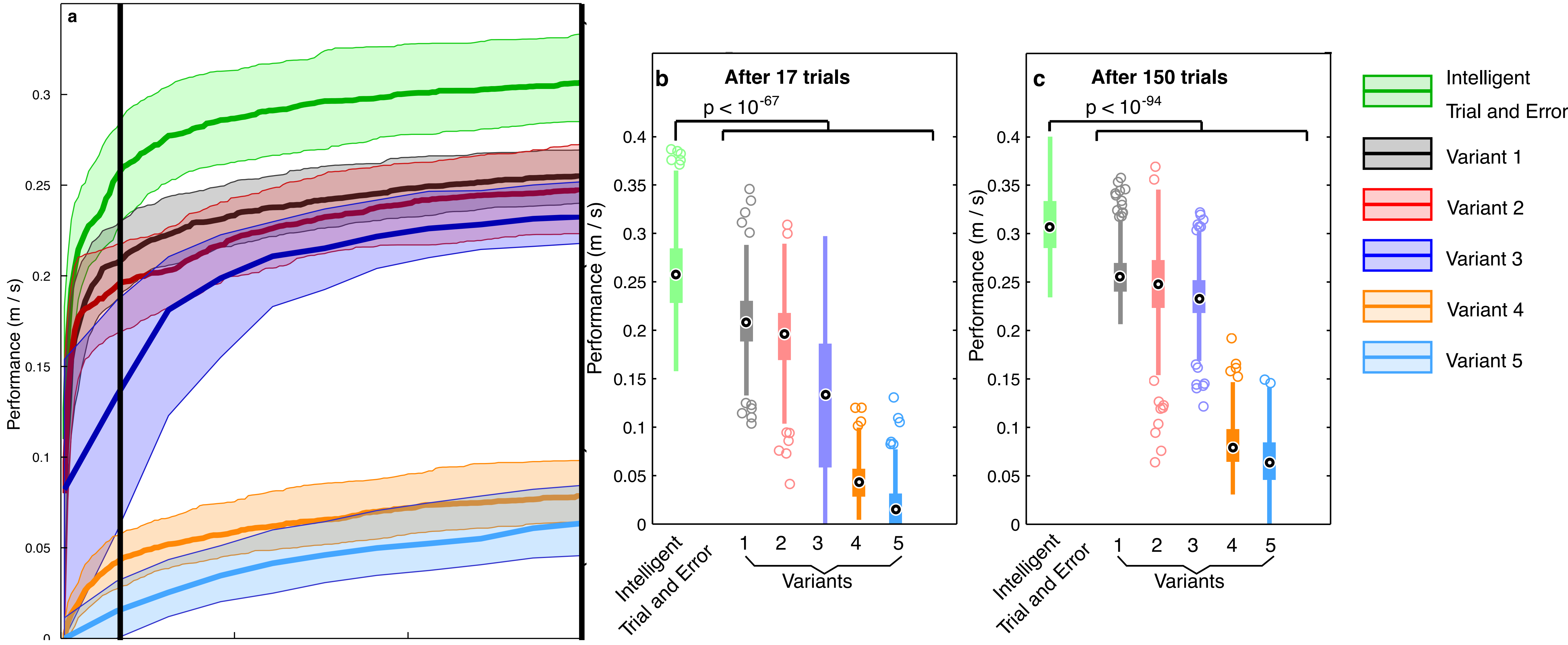
All tested scenarios



nb replicates: 40 / condition

Comparison with the State of the Art

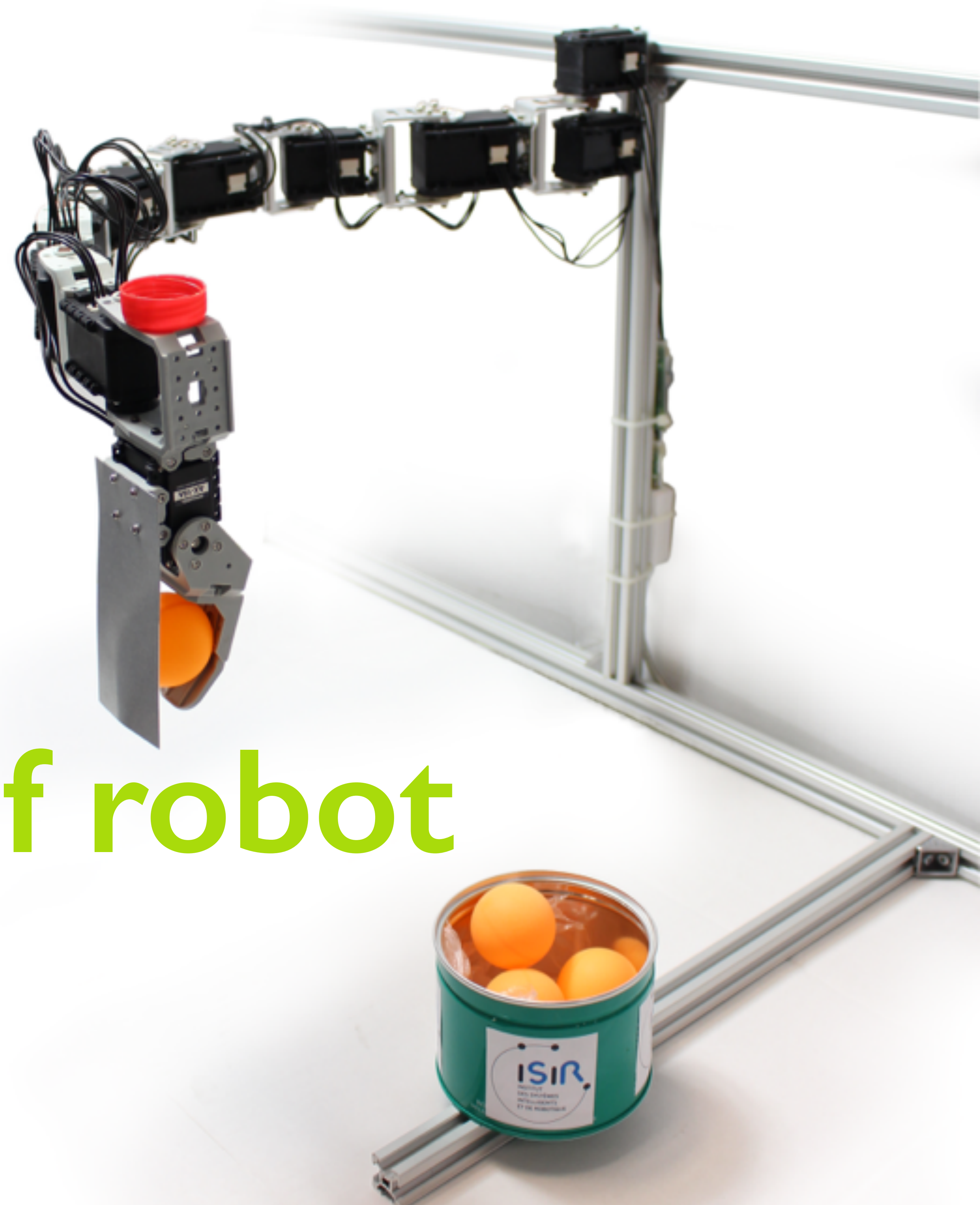
In simulation



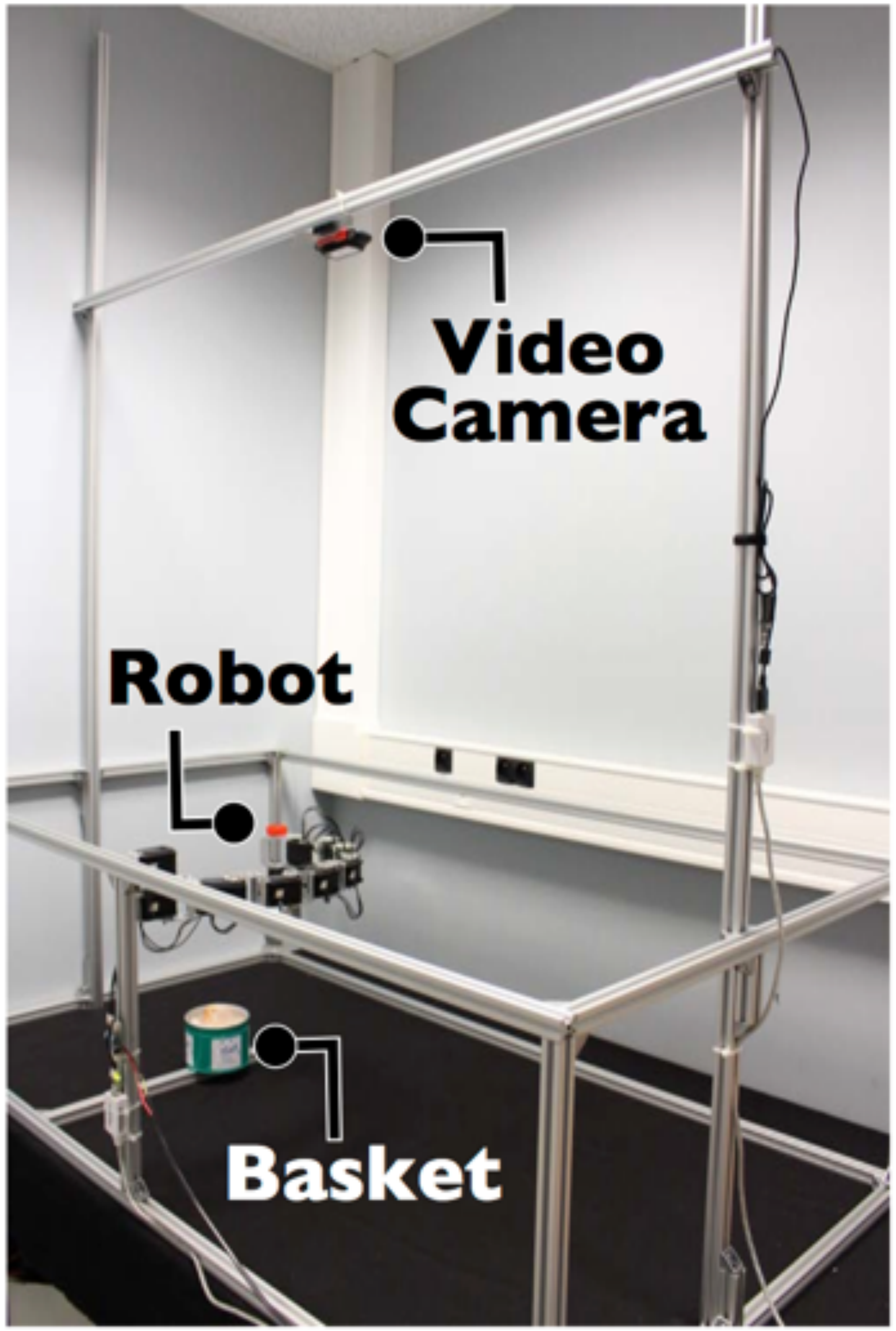
Variant	Behavior-performance map creation	Priors on performance	Search algorithm	equivalent approach
Intelligent Trial and Error	MAP-Elites	yes	Bayesian Optimization	-
Variant 1	MAP-Elites	none	Random Search	-
Variant 2	MAP-Elites	none	Bayesian Optimization	-
Variant 3	MAP-Elites	none	Policy Gradient	-
Variant 4	none	none	Bayesian Optimization	Lizotte et al. (2007)
Variant 5	none	none	Policy Gradient	Kohl et al. (2004)

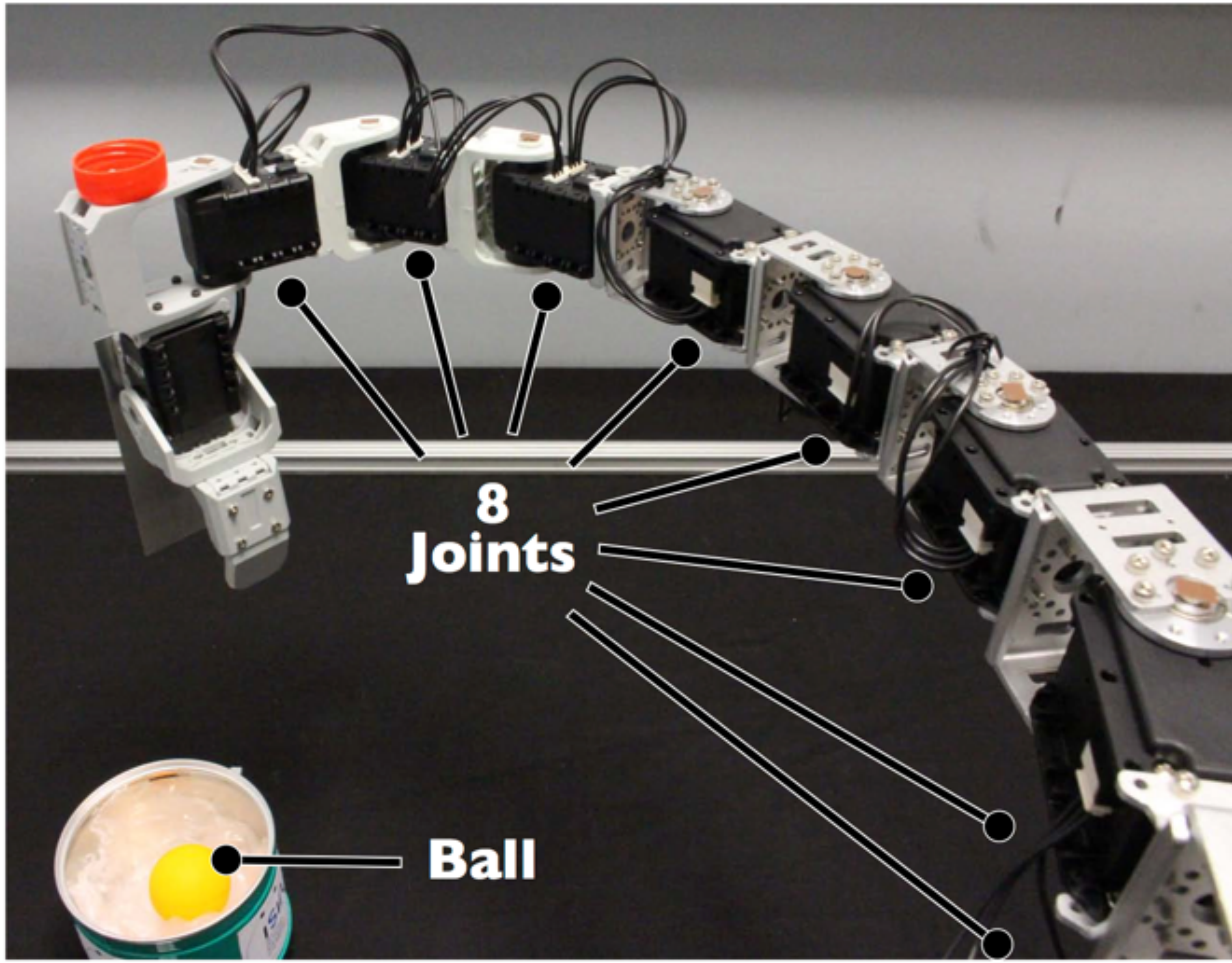
nb replicates: 480 / variant

Works on other types of robot



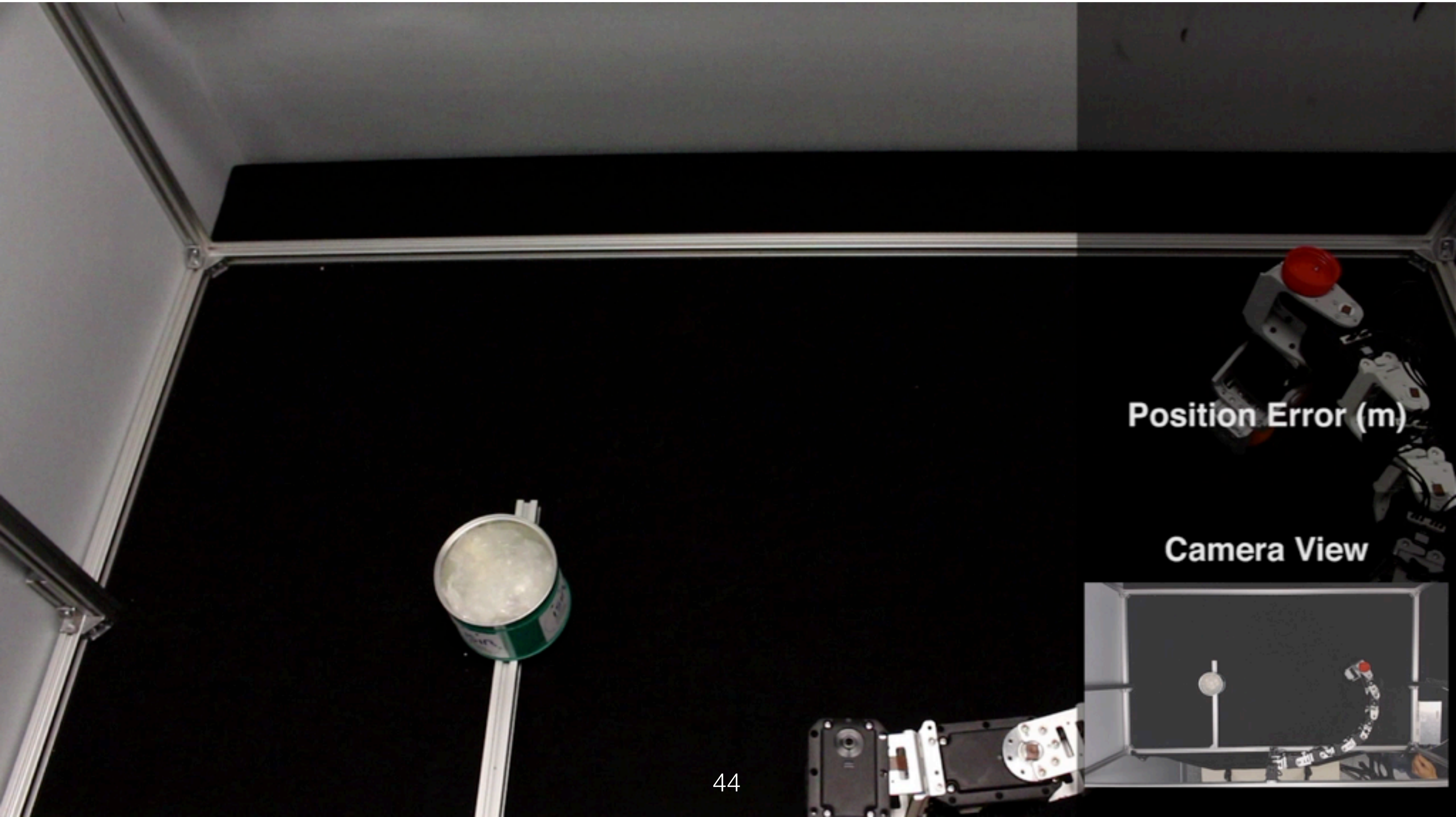
8 DOFs Arm



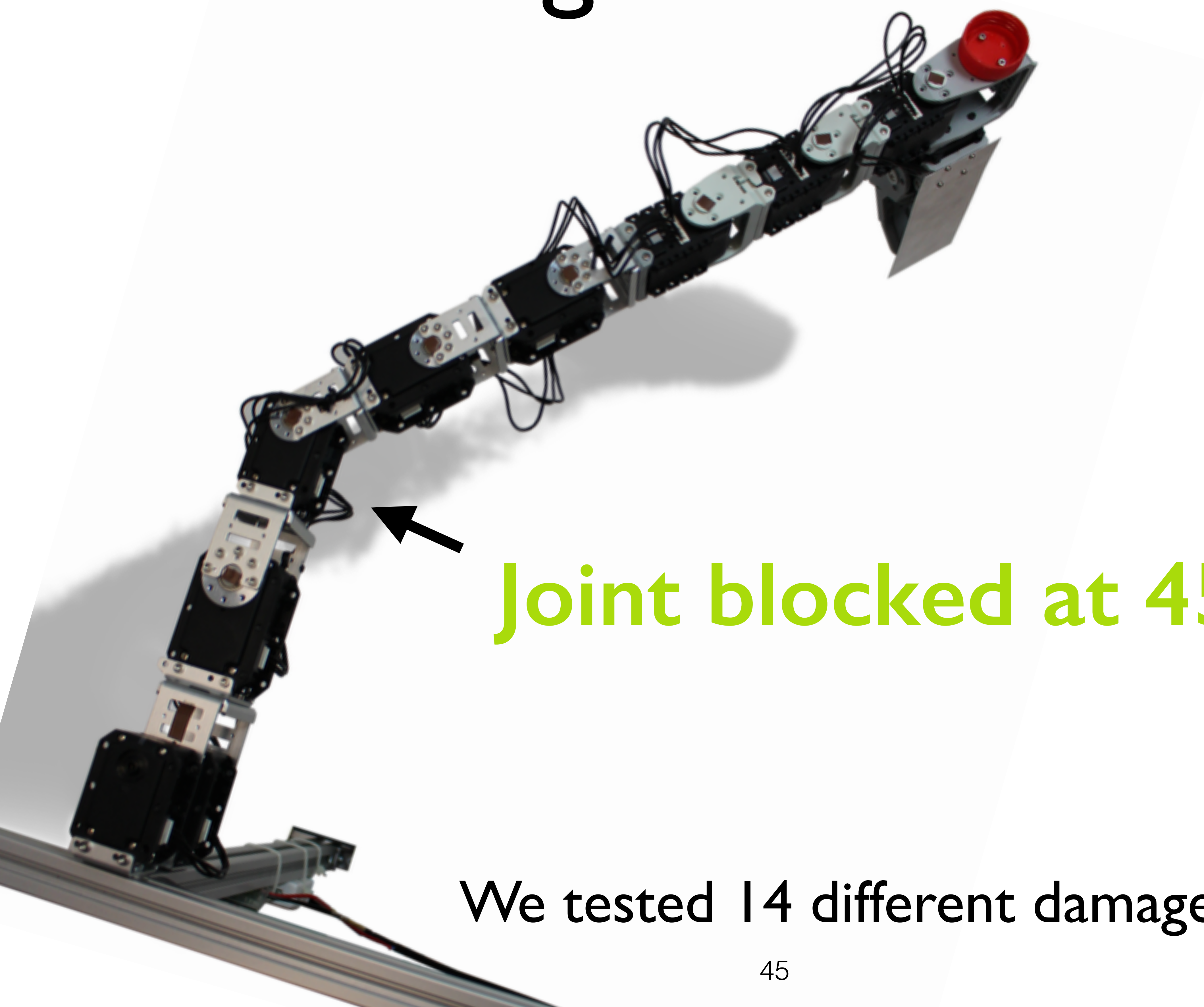


8 DOFs Arm

Open-loop controller



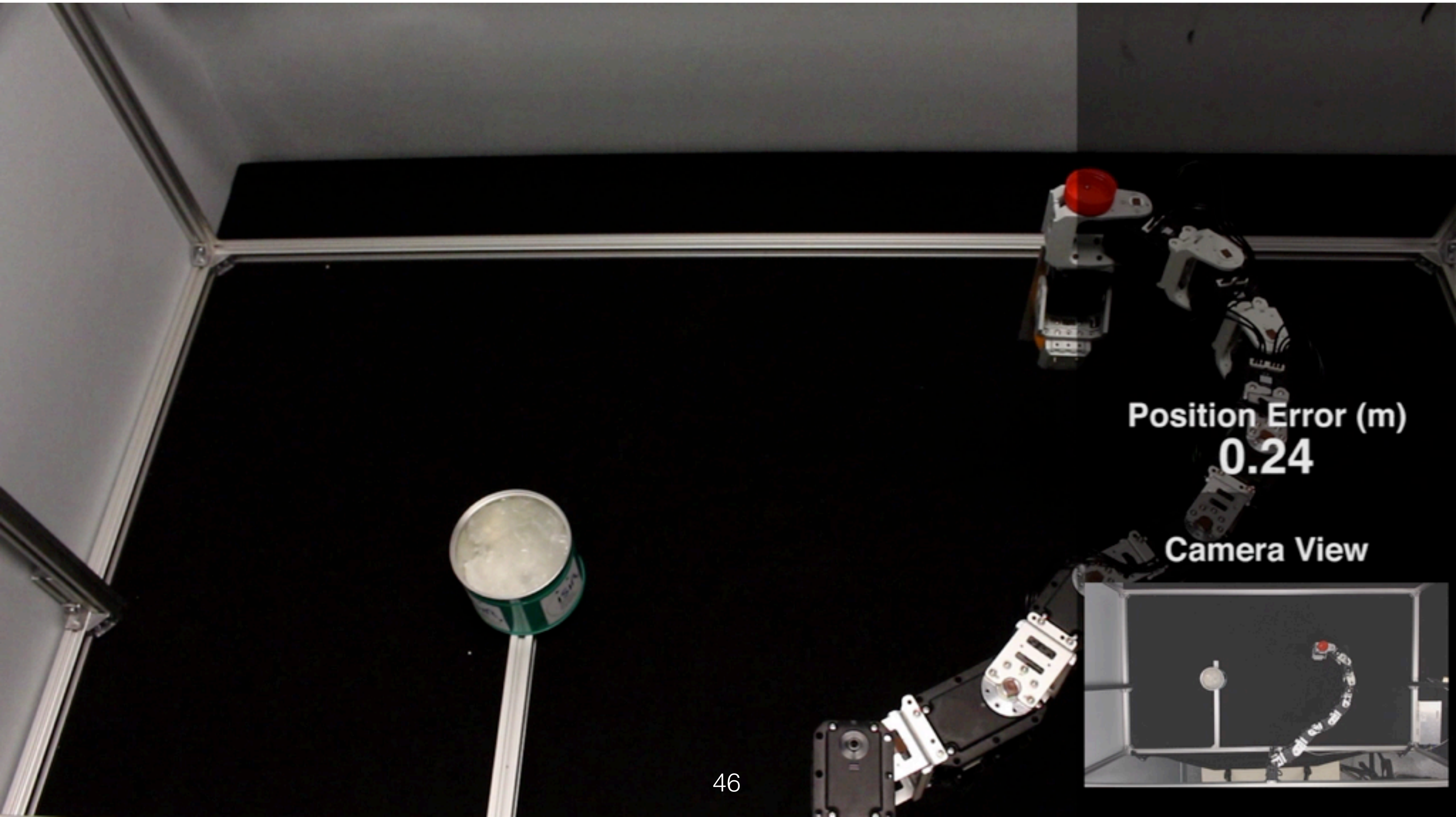
Damaged 8 DOFs Arm



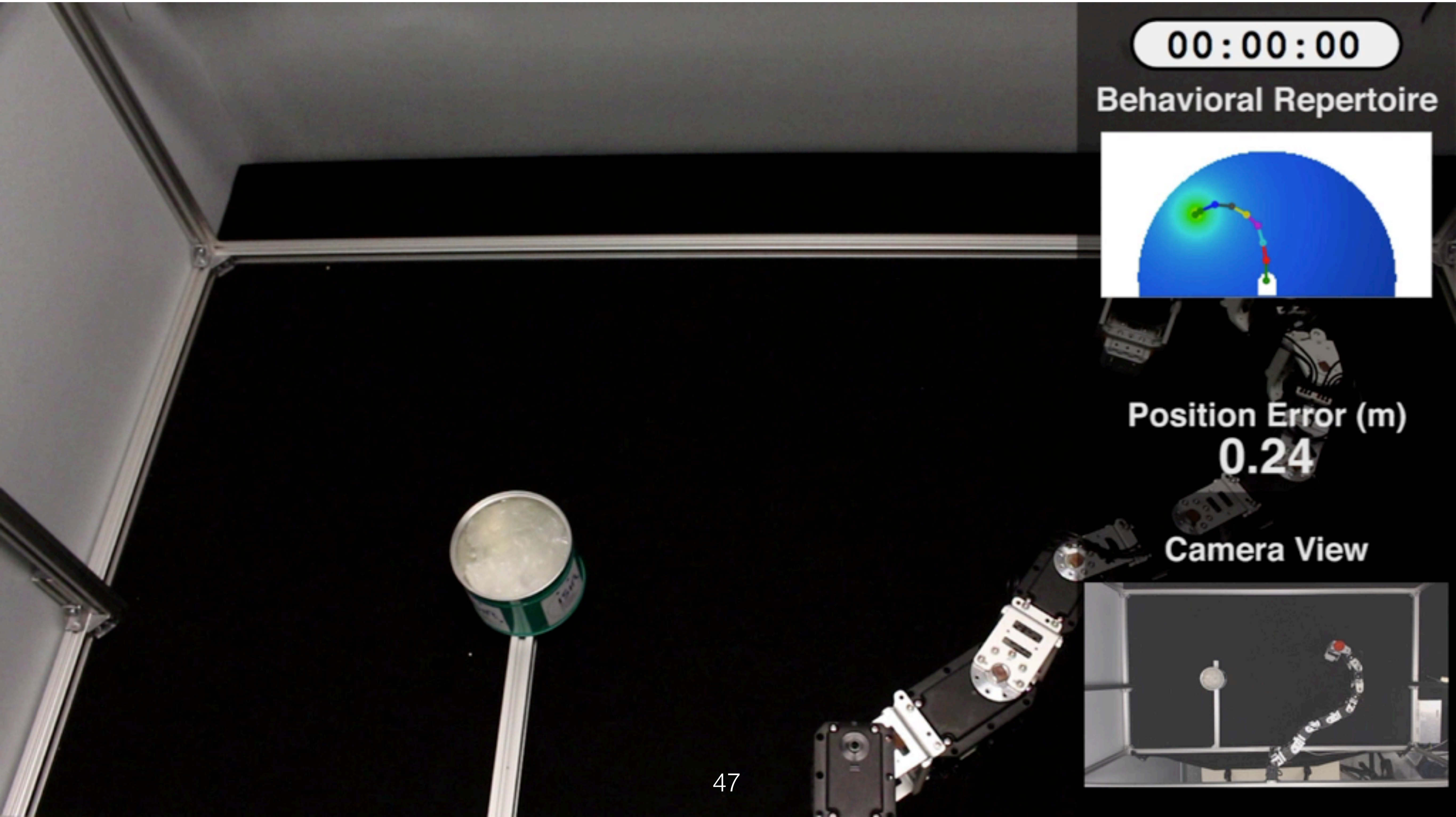
Joint blocked at 45°

We tested 14 different damage conditions

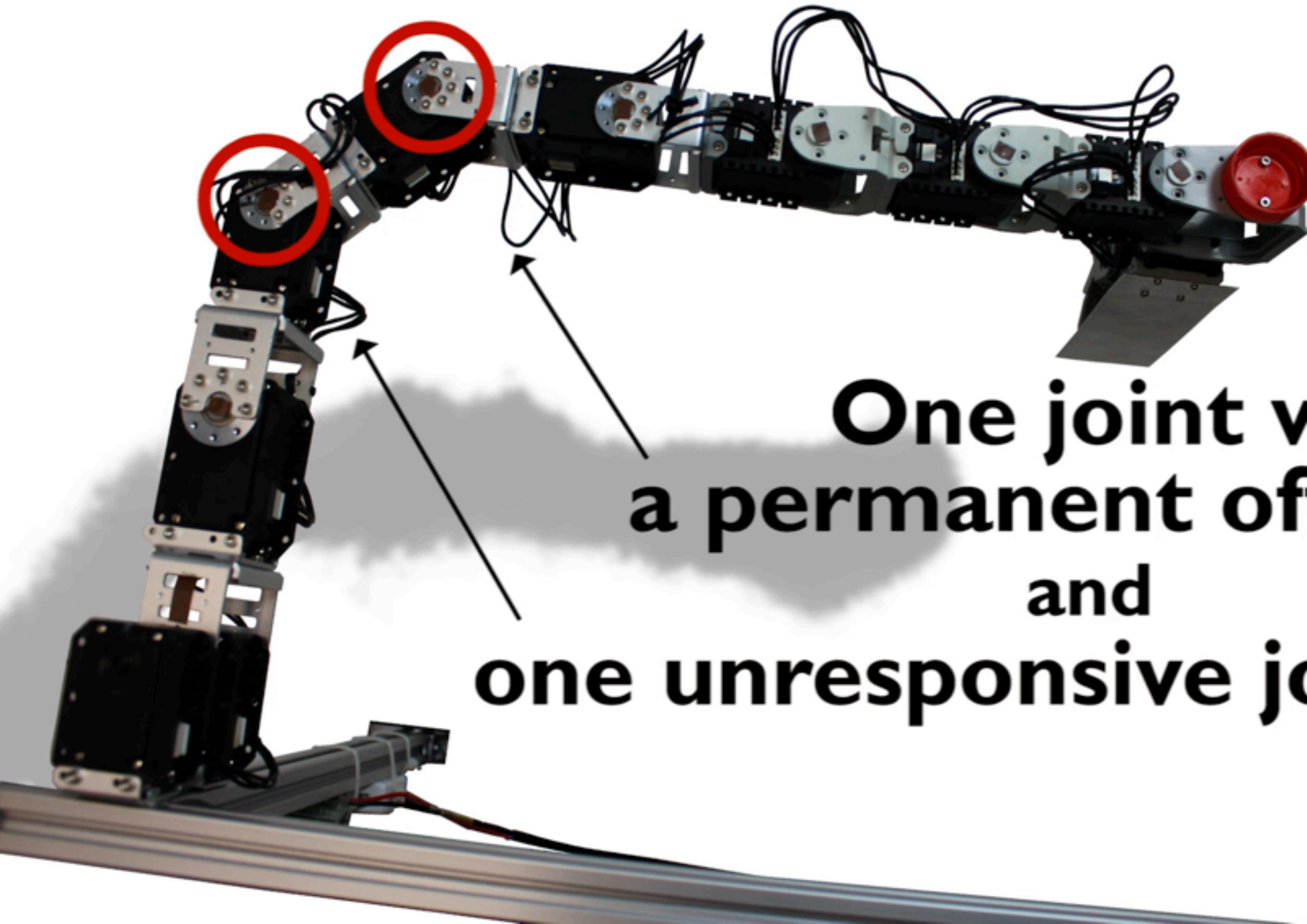
Damaged 8 DOFs Arm



Damaged 8 DOFs Arm



Other examples



One joint with a permanent offset and one unresponsive joint

Intelligent Trial and Error Conclusion

- with the **Intelligent Trial and Error algorithm,** robots can **generate** and **use prior knowledge** (simulation) to learn and adapt **quickly**.
- The Intelligent Trial and Error algorithm is at least **one order of magnitude faster** than state of the art learning algorithms.

Cully, Clune, Tarapore & Mouret (2015). Robots that can adapt like animals. Nature.

Thanks to:

My supervisors

Jean-Baptiste Mouret

Stephane Doncieux

My co-authors

Sylvain Koos

Jeff Clune

Danesh Tarapore

Thank you for your attention

Questions ?