Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Institute for AI and Autonomous Systems

Learning for Robot Manipulation

An Overview

Dr. Alex Mitrevski
Master of Autonomous Systems

Structure

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

Reinforcement learning in robotics: A survey

The International Journal of Robotics Research 32(11) 1238–1274 © The Author(s) 2013 Reprints and permissions: sagepub.co.uk/journalsPermissions.nav DOI: 10.1177/0278364913495721 ijr.sagepub.com SAGE

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

Robotics and Autonomous Systems 156 (2022) 104224

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

A survey of robot manipulation in contact

Markku Suomalainen[a,*], Yiannis Karayiannidis[b], Ville Kyrki[c]

- Why learning for robot manipulation
- Overview of learning for manipulation
- State representation
- Manipulation policy learning
- Transition model learning

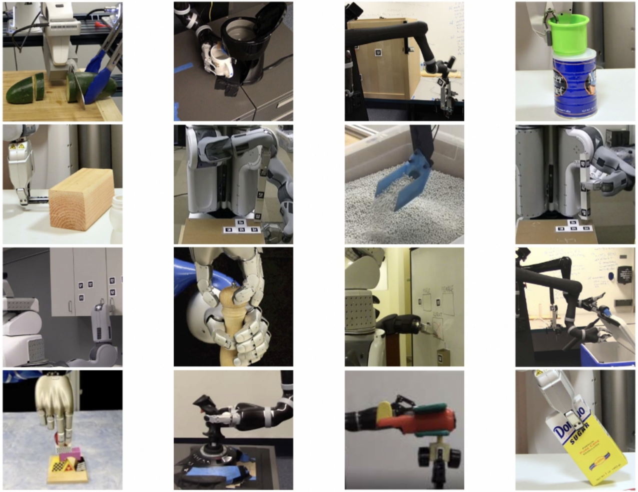
Why Learning for Robot Manipulation

Manipulation Skill Examples

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms

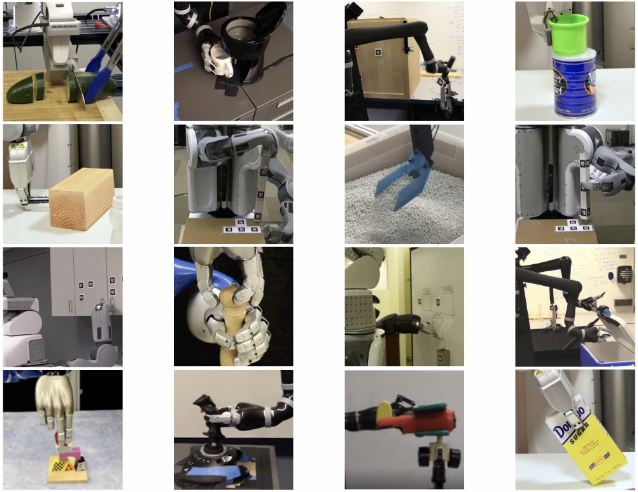
Oliver Kroemer* Scott Niekum George Konidaris



- In everyday environments, there is a large variety of useful manipulation skills, which require varying degrees of dexterity

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

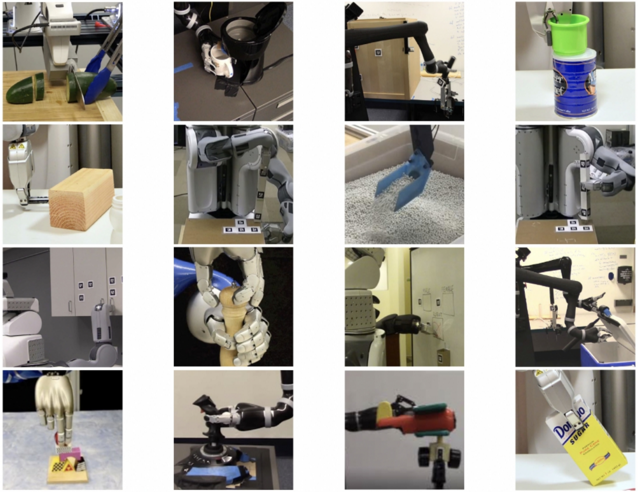
Manipulation Skill Examples



- In everyday environments, there is a large variety of useful manipulation skills, which require varying degrees of dexterity
- Many such skills can be designed using model-based techniques, but many others require flexibility that can be tricky to model explicitly

Manipulation Skill Examples

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris



- In everyday environments, there is a large variety of useful manipulation skills, which require varying degrees of dexterity
- Many such skills can be designed using model-based techniques, but many others require flexibility that can be tricky to model explicitly
- An alternative approach is to allow a robot to acquire such skills (semi-)autonomously

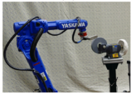
Learning for Contact-Heavy Interactions


Robotics and Autonomous Systems 156 (2022) 104224
Contents lists available at ScienceDirect
Robotics and Autonomous Systems
journal homepage: www.elsevier.com/locate/robot
A survey of robot manipulation in contact
Markku Suomalainen a,*, Yiannis Karayiannidis b, Ville Kyrki c

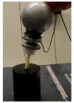

(a) Wiping [18]


(b) Wood planing [19]


(c) Scooping [20]

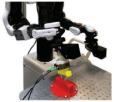


(d) Grinding [21]


(e) Classic (rounded) Peg-in-hole [22]


(f) Folding assembly [23]

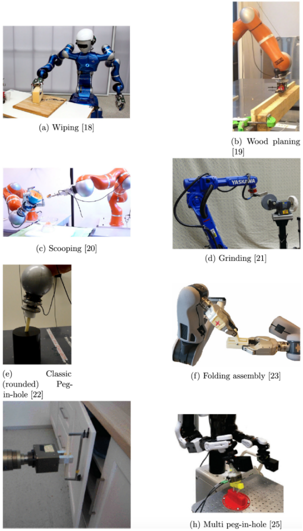

(g) Door opening [24]


(h) Multi peg-in-hole [25]

- Learning can be particularly useful to consider for manipulation tasks that involve prolonged or precise contacts with the environment

Learning for Contact-Heavy Interactions

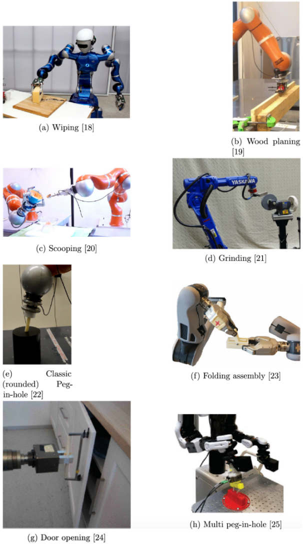

Robotics and Autonomous Systems 156 (2022) 104224
Contents lists available at ScienceDirect
Robotics and Autonomous Systems
journal homepage: www.elsevier.com/locate/robot
A survey of robot manipulation in contact
Markku Suomalainen[a,*], Yiannis Karayiannidis[b], Ville Kyrki[c]


(a) Wiping [18]
(b) Wood planing [19]
(c) Scooping [20]
(d) Grinding [21]
(e) Classic (rounded) Peg-in-hole [22]
(f) Folding assembly [23]
(g) Door opening [24]
(h) Multi peg-in-hole [25]

- Learning can be particularly useful to consider for manipulation tasks that involve prolonged or precise contacts with the environment
- This is because, in principle, **contact-heavy interactions can be challenging to model in sufficient detail**

Learning for Contact-Heavy Interactions


Robotics and Autonomous Systems 156 (2022) 104224
Contents lists available at ScienceDirect
Robotics and Autonomous Systems
journal homepage: www.elsevier.com/locate/robot
A survey of robot manipulation in contact
Markku Suomalainen, Yiannis Karayiannidis, Ville Kyrki


(a) Wiping [18]
(b) Wood planing [19]
(c) Scooping [20]
(d) Grinding [21]
(e) Classic (rounded) Peg-in-hole [22]
(f) Folding assembly [23]
(g) Door opening [24]
(h) Multi peg-in-hole [25]

- Learning can be particularly useful to consider for manipulation tasks that involve prolonged or precise contacts with the environment
- This is because, in principle, **contact-heavy interactions can be challenging to model in sufficient detail**
- Instead, it is sensible to **allow a robot to learn an appropriate interaction policy**

Learning and Robot Control

- Particularly when considering manipulation motor skills, **the learning problem is very related to that solved by classical control theory**: make a robot act so that a certain objective is satisfied

Learning and Robot Control

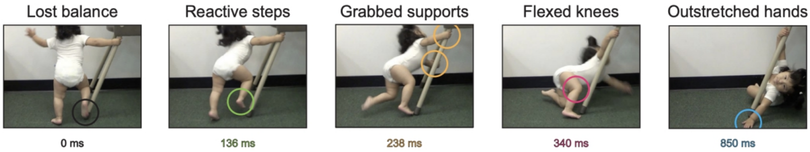
- Particularly when considering manipulation motor skills, **the learning problem is very related to that solved by classical control theory**: make a robot act so that a certain objective is satisfied
- The approaches are, however, conceptually different:
  - Control theory **models systems and controllers explicitly**
  - Learning enables robots to **optimise controllers through direct experience**

Learning and Robot Control

- Particularly when considering manipulation motor skills, **the learning problem is very related to that solved by classical control theory**: make a robot act so that a certain objective is satisfied
- The approaches are, however, conceptually different:
  - Control theory **models systems and controllers explicitly**
  - Learning enables robots to **optimise controllers through direct experience**
- Depending on the nature of the learning problem, **a combination of control theory and learning is both possible and reasonable** (e.g. learning can be used to optimise the parameters of an explicitly modelled controller)

Lessons from Natural Systems

- **The perspective on the previous slides is a practical one** — explicitly programming robot skills is often challenging or inflexible, so we use learning techniques instead
- **Learning is also interesting to look at from a cognitive developmental point of view** — after all, biological creatures acquire most of their skills via learning
- Robots that have learning and adaptation capabilities that are similar to those of biological creatures are likely to be **most useful in our complex, regularly changing environments**



D. Han and K. E. Adolph, "The impact of errors in infant development: Falling like a baby," *Developmental Science*, vol. 24, no. 5, pp. e13069:1–14, 2020.

Overview of Learning for Manipulation

What to Learn for Manipulation?

Learning in the manipulation context can be concerned with multiple aspects, for instance:

Object models

Manipulation tasks generally involve handling objects, whose models (e.g. visual recognition models or part models) can be learned

What to Learn for Manipulation?

Learning in the manipulation context can be concerned with multiple aspects, for instance:

Object models

Manipulation tasks generally involve handling objects, whose models (e.g. visual recognition models or part models) can be learned

Policy parameters

In many cases, we want a robot to execute a motor policy $\pi_{\boldsymbol{\theta}}$ that is defined by well-defined parameters $\boldsymbol{\theta}$, which need to be learned

What to Learn for Manipulation?

Learning in the manipulation context can be concerned with multiple aspects, for instance:

Object models

Manipulation tasks generally involve handling objects, whose models (e.g. visual recognition models or part models) can be learned

Policy parameters

In many cases, we want a robot to execute a motor policy $\pi_{\boldsymbol{\theta}}$ that is defined by well-defined parameters $\boldsymbol{\theta}$, which need to be learned

Skill models

In a more general case, a complete skill can be learned (a policy as well as the skill's initiation and termination conditions)

What to Learn for Manipulation?

Learning in the manipulation context can be concerned with multiple aspects, for instance:

Object models

Manipulation tasks generally involve handling objects, whose models (e.g. visual recognition models or part models) can be learned

Policy parameters

In many cases, we want a robot to execute a motor policy $\pi_{\boldsymbol{\theta}}$ that is defined by well-defined parameters $\boldsymbol{\theta}$, which need to be learned

Skill models

In a more general case, a complete skill can be learned (a policy as well as the skill's initiation and termination conditions)

Skill hierarchies

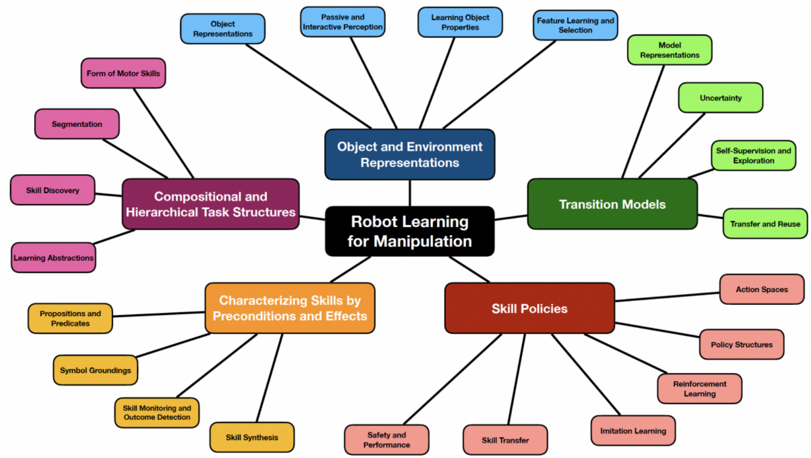
When multiple (primitive) skills are available, it can be useful to learn how the skills can be combined for solving complex tasks

Learning for Manipulation Overview

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris



State Representation

Why Does the State Representation Matter?

- The overall objective of robot manipulation is to enable robots to **perform purposeful actions in the world**, namely actions that change the environment so that some desired goal can be achieved

Why Does the State Representation Matter?

- The overall objective of robot manipulation is to enable robots to **perform purposeful actions in the world**, namely actions that change the environment so that some desired goal can be achieved
- **The manner in which the environment changes based on a robot's actions can be captured by a change of state**; thus, the state representation should be able to capture **relevant changes in the environment**

Why Does the State Representation Matter?

- The overall objective of robot manipulation is to enable robots to **perform purposeful actions in the world**, namely actions that change the environment so that some desired goal can be achieved
- **The manner in which the environment changes based on a robot's actions can be captured by a change of state**; thus, the state representation should be able to capture **relevant changes in the environment**
- In addition, an appropriate state representation is often responsible for **simplifying otherwise intractable learning problems**

Robot and Environment State

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- **Robot actions have an effect both on the robot itself and on the robot's environment**
- A general state representation thus has to capture both of these aspects and has the following form:

$$S = S_r \cup S_e$$

where

- S_r is a representation of the robot's internal state
- S_e represents the state of the task environment

Object-Centric Environment Representation

- As manipulation is typically concerned with handling objects, **the environment state is typically modelled through the states of individual objects of interest**

Object-Centric Environment Representation

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- As manipulation is typically concerned with handling objects, **the environment state is typically modelled through the states of individual objects of interest**
- Let O_j represent the state of some object o_j and n be the number of objects of interest for a task; then

$$S_o = \bigcup_{j=1}^{n} O_j$$

Object-Centric Environment Representation

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- As manipulation is typically concerned with handling objects, **the environment state is typically modelled through the states of individual objects of interest**

- Let O_j represent the state of some object o_j and n be the number of objects of interest for a task; then

$$S_o = \bigcup_{j=1}^{n} O_j$$

- In many cases, it can also be useful to capture some general information S_w about the environment; thus, **the complete environment state can be seen as a combination of the general environment state and the object-specific states**:

$$S_e = S_w \cup S_o = S_w \cup \left(\bigcup_{j=1}^{n} O_j \right)$$

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

Generalisation over Contexts

- When a robot learns an execution policy, the policy is typically **specific for certain environmental parameters** (e.g. for a specific object mass) **that remain constant during the execution**

Generalisation over Contexts

- When a robot learns an execution policy, the policy is typically **specific for certain environmental parameters** (e.g. for a specific object mass) **that remain constant during the execution**
- The dependence on such parameters can be made explicit by representing them as an **execution context vector** $\boldsymbol{\tau} \in C$

Generalisation over Contexts

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- When a robot learns an execution policy, the policy is typically **specific for certain environmental parameters** (e.g. for a specific object mass) **that remain constant during the execution**
- The dependence on such parameters can be made explicit by representing them as an **execution context vector** $\boldsymbol{\tau} \in C$
- The execution context can then serve as information that **conditions the execution policy**:

$$\pi : S \times C \rightarrow A$$

Task Family

- When modelling learning problems, we can often define a **task family**, which is a collection of tasks $T_i, 1 \leq i \leq \mathrm{t}$ that
  - have the same action space A
  - but each of them has its own state space S_i, context space C_i, a transition function $\mathcal{T}_i$, as well as a reward function R_i

Task Family

- When modelling learning problems, we can often define a **task family**, which is a collection of tasks $T_i, 1 \leq i \leq \mathrm{t}$ that
  - have the same action space A
  - but each of them has its own state space S_i, context space C_i, a transition function $\mathcal{T}_i$, as well as a reward function R_i

- **The relation between tasks of a task family can be expressed through the reward function** R_i, which can be modelled as

$$R_i = G_i - E$$

where G_i **represents a task-specific goal** and E **is a common cost function**

Task Family

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- When modelling learning problems, we can often define a **task family**, which is a collection of tasks $T_i, 1 \leq i \leq \mathrm{t}$ that
  - have the same action space A
  - but each of them has its own state space S_i, context space C_i, a transition function $\mathcal{T}_i$, as well as a reward function R_i

- **The relation between tasks of a task family can be expressed through the reward function** R_i, which can be modelled as

$$R_i = G_i - E$$

where G_i **represents a task-specific goal** and E **is a common cost function**

- Overall, a task family of t tasks can be represented as **a collection of Markov Decision Processes (MDPs)**:

$$P(\mathcal{M}) = \{(S_i, A, \mathcal{T}_i, R_i, C_i, \gamma) \mid 1 \leq i \leq \mathrm{t}\}$$

Object Representations

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**

Object Representations

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**
- Concretely, there are different hierarchy levels at which objects can be observed:

Object Representations

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**
- Concretely, there are different hierarchy levels at which objects can be observed:

Point level

Individual object points (e.g. pixels) are identified

Object Representations

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**
- Concretely, there are different hierarchy levels at which objects can be observed:

Point level

Individual object points (e.g. pixels) are identified

Part level

Objects are represented through their individual parts (e.g. a cup has a container and a handle)

Object Representations

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**

- Concretely, there are different hierarchy levels at which objects can be observed:

Point level

Individual object points (e.g. pixels) are identified

Part level

Objects are represented through their individual parts (e.g. a cup has a container and a handle)

Object level

Objects are represented as a whole (e.g. through a bounding box)

Object Representations

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**

- Concretely, there are different hierarchy levels at which objects can be observed:

Point level

Individual object points (e.g. pixels) are identified

Part level

Objects are represented through their individual parts (e.g. a cup has a container and a handle)

Object level

Objects are represented as a whole (e.g. through a bounding box)

- Each of these hierarchies can be useful for different tasks:

Object Representations

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**

- Concretely, there are different hierarchy levels at which objects can be observed:

Point level
Individual object points (e.g. pixels) are identified

Part level
Objects are represented through their individual parts (e.g. a cup has a container and a handle)

Object level
Objects are represented as a whole (e.g. through a bounding box)

- Each of these hierarchies can be useful for different tasks:
  - A point-level representation is suitable when specific points of an object are relevant during an interaction (e.g. the prongs of a fork)

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

Object Representations

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**

- Concretely, there are different hierarchy levels at which objects can be observed:

Point level

Individual object points (e.g. pixels) are identified

Part level

Objects are represented through their individual parts (e.g. a cup has a container and a handle)

Object level

Objects are represented as a whole (e.g. through a bounding box)

- Each of these hierarchies can be useful for different tasks:
  - A point-level representation is suitable when specific points of an object are relevant during an interaction (e.g. the prongs of a fork)
  - Parts can be useful to look at for task-oriented grasping

Object Representations

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**

- Concretely, there are different hierarchy levels at which objects can be observed:

Point level

Individual object points (e.g. pixels) are identified

Part level

Objects are represented through their individual parts (e.g. a cup has a container and a handle)

Object level

Objects are represented as a whole (e.g. through a bounding box)

- Each of these hierarchies can be useful for different tasks:
  - A point-level representation is suitable when specific points of an object are relevant during an interaction (e.g. the prongs of a fork)
  - Parts can be useful to look at for task-oriented grasping
  - An object-level representation aids scene understanding, but is also required for ensuring that actions are performed on an object of interest

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

Object Representations

- Objects are often part of the state representation in robot manipulation, but **there is no unique way in which objects themselves are modelled**

- Concretely, there are different hierarchy levels at which objects can be observed:

Point level	Part level	Object level
Individual object points (e.g. pixels) are identified	Objects are represented through their individual parts (e.g. a cup has a container and a handle)	Objects are represented as a whole (e.g. through a bounding box)

- Each of these hierarchies can be useful for different tasks:
  - A point-level representation is suitable when specific points of an object are relevant during an interaction (e.g. the prongs of a fork)
  - Parts can be useful to look at for task-oriented grasping
  - An object-level representation aids scene understanding, but is also required for ensuring that actions are performed on an object of interest

- Ideally, **the hierarchical levels are used by different skills that can be composed** to solve a specific task

Passive vs. Interactive Perception

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- **A robot needs to perceive the environment to acquire information about objects and the overall scene**

Passive vs. Interactive Perception

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- **A robot needs to perceive the environment to acquire information about objects and the overall scene**
- There are two overall perception strategies depending on whether the robot is simply passively observing or actively investigating the scene:

Passive vs. Interactive Perception

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- **A robot needs to perceive the environment to acquire information about objects and the overall scene**
- There are two overall perception strategies depending on whether the robot is simply passively observing or actively investigating the scene:

Passive perception

A scene is observed passively based on received sensory data (e.g. camera images)

Passive vs. Interactive Perception

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- **A robot needs to perceive the environment to acquire information about objects and the overall scene**
- There are two overall perception strategies depending on whether the robot is simply passively observing or actively investigating the scene:

Passive perception

A scene is observed passively based on received sensory data (e.g. camera images)

Interactive perception

A robot performs actions to collect information about certain environmental aspects (e.g. touching an object to find out its material)

Passive vs. Interactive Perception

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- **A robot needs to perceive the environment to acquire information about objects and the overall scene**
- There are two overall perception strategies depending on whether the robot is simply passively observing or actively investigating the scene:

Passive perception

A scene is observed passively based on received sensory data (e.g. camera images)

Interactive perception

A robot performs actions to collect information about certain environmental aspects (e.g. touching an object to find out its material)

- **Many aspects of the environment are not observable using passive perception only** (e.g. the mass of an object), so interactive perception is often essential for successful task completion

Manipulation Policy Learning

Execution Policies Revisited

- In the last lecture, we defined a skill by an execution policy together with initiation and termination conditions

Execution Policies Revisited

- In the last lecture, we defined a skill by an execution policy together with initiation and termination conditions
- A policy $\pi : S \rightarrow A$ models a robot's behaviour, and **a particularly large effort in robot learning is put on how to actually acquire such a policy**

Execution Policies Revisited

- In the last lecture, we defined a skill by an execution policy together with initiation and termination conditions
- A policy $\pi : S \rightarrow A$ models a robot's behaviour, and **a particularly large effort in robot learning is put on how to actually acquire such a policy**
- There are a few general ways in which this can be done:

Execution Policies Revisited

- In the last lecture, we defined a skill by an execution policy together with initiation and termination conditions
- A policy $\pi : S \rightarrow A$ models a robot's behaviour, and **a particularly large effort in robot learning is put on how to actually acquire such a policy**
- There are a few general ways in which this can be done:

Reinforcement learning

A policy is learned using direct interactions with the world

Execution Policies Revisited

- In the last lecture, we defined a skill by an execution policy together with initiation and termination conditions
- A policy $\pi : S \rightarrow A$ models a robot's behaviour, and **a particularly large effort in robot learning is put on how to actually acquire such a policy**
- There are a few general ways in which this can be done:

Reinforcement learning	Imitation learning
A policy is learned using direct interactions with the world	Learning is done based on expert observations

Execution Policies Revisited

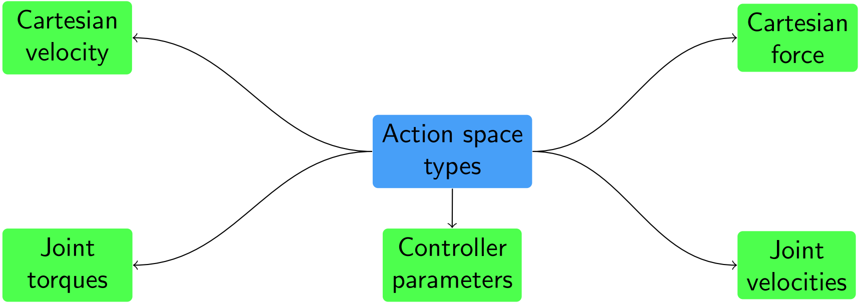
- In the last lecture, we defined a skill by an execution policy together with initiation and termination conditions
- A policy $\pi : S \rightarrow A$ models a robot's behaviour, and **a particularly large effort in robot learning is put on how to actually acquire such a policy**
- There are a few general ways in which this can be done:

Reinforcement learning	Imitation learning	Transfer learning
A policy is learned using direct interactions with the world	Learning is done based on expert observations	Previously learned policies are used to guide the learning process

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

Action Spaces

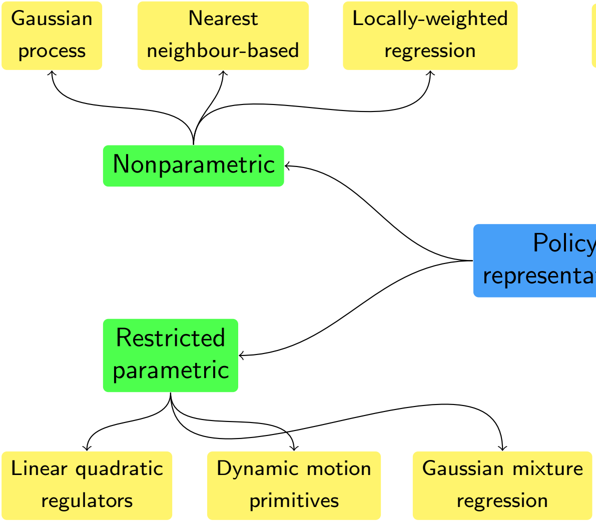
- Execution policies can have a variety of action spaces, which are illustrated below:



- Note that policy outputs are typically not directly used as actuator commands, but are **processed by a low-level robot controller**

Policy Representations

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris



Deterministic vs. Stochastic Policies

- Regarding how actions are selected from a policy, we can distinguish between **deterministic and stochastic policies**

Deterministic vs. Stochastic Policies

- Regarding how actions are selected from a policy, we can distinguish between **deterministic and stochastic policies**

Stochastic policy

Actions are selected by sampling from the distribution of actions given a state

$$\boldsymbol{a}_t \sim \pi(\boldsymbol{a}|\boldsymbol{s}_t)$$

Deterministic vs. Stochastic Policies

- Regarding how actions are selected from a policy, we can distinguish between **deterministic and stochastic policies**

Stochastic policy

Actions are selected by sampling from the distribution of actions given a state

$$\boldsymbol{a}_t \sim \pi(\boldsymbol{a}|\boldsymbol{s}_t)$$

Deterministic policy

Actions are selected by a deterministic function of the current state

$$\boldsymbol{a}_t = \pi(\boldsymbol{s}_t)$$

Parameterised Policies and Trajectories


Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- In robotics, **policies are often represented by parameters $\boldsymbol{\theta}$**, so we denote the policy as $\pi_{\boldsymbol{\theta}}$

Parameterised Policies and Trajectories


Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- In robotics, **policies are often represented by parameters θ**, so we denote the policy as $\pi_{\boldsymbol{\theta}}$
- A policy is used to define a **trajectory (also called episode or rollout)**

$$\tau = (\boldsymbol{s}_0, \boldsymbol{a}_0, \boldsymbol{s}_1, ..., \boldsymbol{a}_n, \boldsymbol{s}_{n+1})$$

Parameterised Policies and Trajectories


Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- In robotics, **policies are often represented by parameters θ**, so we denote the policy as $\pi_{\boldsymbol{\theta}}$
- A policy is used to define a **trajectory (also called episode or rollout)**

$$\tau = (\boldsymbol{s}_0, \boldsymbol{a}_0, \boldsymbol{s}_1, ..., \boldsymbol{a}_n, \boldsymbol{s}_{n+1})$$

- Given a policy π, the probability of a trajectory can be found to be

$$P_\pi(\boldsymbol{s}_0, \boldsymbol{a}_0, \boldsymbol{s}_1, ..., \boldsymbol{a}_n, \boldsymbol{s}_{n+1}) = P(s_0) \prod_{i=0}^{n} P_\pi(\boldsymbol{a}_i | \boldsymbol{s}_i) P(\boldsymbol{s}_{i+1} | \boldsymbol{s}_i, \boldsymbol{a}_i)$$

Reinforcement Learning Objective

- When using reinforcement learning for acquiring a policy, the objective is to find a policy π^* that maximises the robot's expected return:

$$\pi^* = \arg\max_{\pi} \underset{\tau \sim \pi}{E}\left[\sum_t r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] = \arg\max_{\pi} \int P(\tau|\pi) R(\tau) d\tau$$

Reinforcement Learning Objective

- When using reinforcement learning for acquiring a policy, the objective is to find a policy π^* that maximises the robot's expected return:

$$\pi^* = \arg\max_{\pi} \mathop{E}_{\tau \sim \pi} \left[\sum_t r(\boldsymbol{s}_t, \boldsymbol{a}_t) \right] = \arg\max_{\pi} \int P(\tau|\pi) R(\tau) d\tau$$

- If we are given a parameterised policy $\pi_{\boldsymbol{\theta}}$, the learning objective is that of finding a set of parameters θ^* that maximise the expected return

$$\theta^* = \arg\max_{\boldsymbol{\theta}} \mathop{E}_{\tau \sim \pi_{\boldsymbol{\theta}}} \left[\sum_t r(\boldsymbol{s}_t, \boldsymbol{a}_t) \right] = \arg\max_{\boldsymbol{\theta}} \int P(\tau|\pi_{\boldsymbol{\theta}}) R(\tau) d\tau$$

Exploration vs. Exploitation

Reinforcement Learning

- During learning, a robot has to balance **exploration and exploitation**

Exploration vs. Exploitation

Reinforcement Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

- During learning, a robot has to balance **exploration and exploitation**

Exploitation

Acting according to the best policy available to the robot (so far)

Exploration vs. Exploitation

Reinforcement Learning


Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- During learning, a robot has to balance **exploration and exploitation**

Exploitation

Acting according to the best policy available to the robot (so far)

Exploration

Acting by trying out actions that may not be the most optimal under the current best policy

Exploration vs. Exploitation

Reinforcement Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- During learning, a robot has to balance **exploration and exploitation**

Exploitation

Acting according to the best policy available to the robot (so far)

Exploration

Acting by trying out actions that may not be the most optimal under the current best policy

- There is always a trade-off between exploitation and exploration:
  - if the robot exploits too much too early, it risks converging to a suboptimal policy
  - the robot's policy should eventually converge however; too much exploration can prevent that from happening

Model-Free Learning

Reinforcement Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- When learning manipulation policies, a robot usually does not have a transition model (or even a reward model) of the environment, but **has to explore the environment during learning**

Model-Free Learning

Reinforcement Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- When learning manipulation policies, a robot usually does not have a transition model (or even a reward model) of the environment, but **has to explore the environment during learning**
- In such instances, **model-free reinforcement learning needs to be used**

Model-Free Learning

Reinforcement Learning


Reinforcement learning in robotics:
A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- When learning manipulation policies, a robot usually does not have a transition model (or even a reward model) of the environment, but **has to explore the environment during learning**
- In such instances, **model-free reinforcement learning needs to be used**
- The model-free learning setup is that we have m trajectories $\tau_i, 1 \leq i \leq m$ and the accompanying rewards along the trajectories, such that **an optimal policy has to be found from these experiences**

Model-Free Learning

Reinforcement Learning


Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com

- When learning manipulation policies, a robot usually does not have a transition model (or even a reward model) of the environment, but **has to explore the environment during learning**
- In such instances, **model-free reinforcement learning needs to be used**
- The model-free learning setup is that we have m trajectories $\tau_i, 1 \leq i \leq m$ and the accompanying rewards along the trajectories, such that **an optimal policy has to be found from these experiences**
- There are two major families of model-free algorithms:

Model-Free Learning

Reinforcement Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- When learning manipulation policies, a robot usually does not have a transition model (or even a reward model) of the environment, but **has to explore the environment during learning**
- In such instances, **model-free reinforcement learning needs to be used**
- The model-free learning setup is that we have m trajectories $\tau_i, 1 \leq i \leq m$ and the accompanying rewards along the trajectories, such that **an optimal policy has to be found from these experiences**
- There are two major families of model-free algorithms:

Temporal difference learning

Performs value / policy updates at every step (i.e. after the execution of every action)

Hochschule Bonn-Rhein-Sieg University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Institute for AI and Autonomous Systems

Model-Free Learning

Reinforcement Learning


Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- When learning manipulation policies, a robot usually does not have a transition model (or even a reward model) of the environment, but **has to explore the environment during learning**
- In such instances, **model-free reinforcement learning needs to be used**
- The model-free learning setup is that we have m trajectories $\tau_i, 1 \leq i \leq m$ and the accompanying rewards along the trajectories, such that **an optimal policy has to be found from these experiences**
- There are two major families of model-free algorithms:

Temporal difference learning	Monte Carlo learning
Performs value / policy updates at every step (i.e. after the execution of every action)	Estimates the return from complete trajectories and then performs value / policy updates

Temporal Difference — TD(λ) — Learning and Q-Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

Reinforcement Learning

- **The TD(λ) learning algorithm attempts to bring the value function $V(s_t)$ closer to the reward function, while preventing myopic updates**

Temporal Difference — TD(λ) — Learning and Q-Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

Reinforcement Learning

- **The TD(λ) learning algorithm attempts to bring the value function $V(s_t)$ closer to the reward function, while preventing myopic updates**
- The parameter λ controls the amount of prediction during learning — if $\lambda > 0$, older states are considered during learning

Temporal Difference — TD(λ) — Learning and Q-Learning


Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions: sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- **The TD(λ) learning algorithm attempts to bring the value function $V(s_t)$ closer to the reward function, while preventing myopic updates**
- The parameter λ controls the amount of prediction during learning — if $\lambda > 0$, older states are considered during learning
- **For TD(0), only a single-step prediction is done**, with α a learning rate

$$V(\boldsymbol{s}_t) = V(\boldsymbol{s}_t) + \alpha \left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma V(\boldsymbol{s}_{t+1}) - V(\boldsymbol{s}_t) \right)$$

Temporal Difference — TD(λ) — Learning and Q-Learning

Reinforcement learning in robotics: A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

- **The TD(λ) learning algorithm attempts to bring the value function $V(s_t)$ closer to the reward function, while preventing myopic updates**
- The parameter λ controls the amount of prediction during learning — if $\lambda > 0$, older states are considered during learning
- **For TD(0), only a single-step prediction is done**, with α a learning rate

$$V(\boldsymbol{s}_t) = V(\boldsymbol{s}_t) + \alpha\left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma V(\boldsymbol{s}_{t+1}) - V(\boldsymbol{s}_t)\right)$$

- A popular temporal difference RL algorithm is Q-learning, which estimates a state-action value function $Q(\boldsymbol{s}_t, \boldsymbol{a}_t)$

Temporal Difference — TD(λ) — Learning and Q-Learning


Reinforcement learning in robotics:
A survey

Jens Kober[1,2], J. Andrew Bagnell[3] and Jan Peters[4,5]

The International Journal of
Robotics Research
32(11) 1238–1274
© The Author(s) 2013
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913495721
ijr.sagepub.com
SAGE

- **The TD(λ) learning algorithm attempts to bring the value function $V(s_t)$ closer to the reward function, while preventing myopic updates**
- The parameter λ controls the amount of prediction during learning — if $\lambda > 0$, older states are considered during learning
- **For TD(0), only a single-step prediction is done**, with α a learning rate

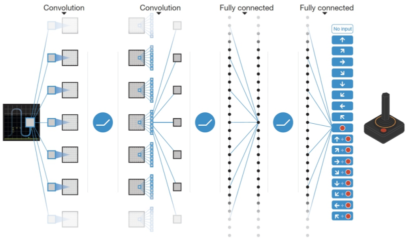
$$V(\boldsymbol{s}_t) = V(\boldsymbol{s}_t) + \alpha\left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma V(\boldsymbol{s}_{t+1}) - V(\boldsymbol{s}_t)\right)$$

- A popular temporal difference RL algorithm is Q-learning, which estimates a state-action value function $Q(\boldsymbol{s}_t, \boldsymbol{a}_t)$
- The Q-learning update rule is given by

$$Q(\boldsymbol{s}_t, \boldsymbol{a}_t) = Q(\boldsymbol{s}_t, \boldsymbol{a}_t) + \alpha\left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \max_{\boldsymbol{a}} Q(\boldsymbol{s}_{t+1}, \boldsymbol{a}) - Q(\boldsymbol{s}_t, \boldsymbol{a}_t)\right)$$

Deep Q-Learning

Reinforcement Learning

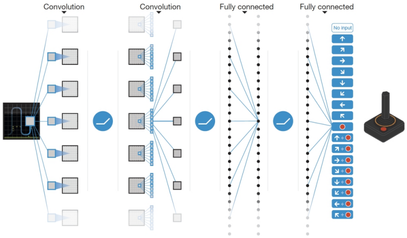


V. Mnih et al. "Human-level Control Through Deep Reinforcement Learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

- Q-learning as seen on the previous slide is defined for discrete action spaces; however, **using a function approximator (e.g. a neural network), it can be extended to continuous state spaces**

Deep Q-Learning

Reinforcement Learning



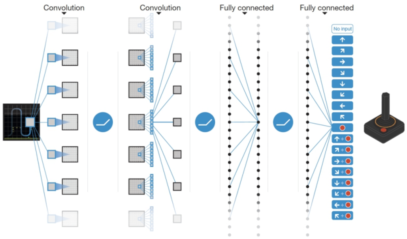
V. Mnih et al. "Human-level Control Through Deep Reinforcement Learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

- Q-learning as seen on the previous slide is defined for discrete action spaces; however, **using a function approximator (e.g. a neural network), it can be extended to continuous state spaces**
- In deep Q-learning, Q-function is represented using a deep neural network, and the objective function that is being minimised here is often of the form

$$\mathcal{L}(\boldsymbol{\theta}) = E\left[Q(\boldsymbol{s}_t, \boldsymbol{a}_t) - \left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \max_{\boldsymbol{a}} Q(\boldsymbol{s}_{t+1}, \boldsymbol{a})\right)\right]$$

Deep Q-Learning

Reinforcement Learning



V. Mnih et al. "Human-level Control Through Deep Reinforcement Learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

- Q-learning as seen on the previous slide is defined for discrete action spaces; however, **using a function approximator (e.g. a neural network), it can be extended to continuous state spaces**
- In deep Q-learning, Q-function is represented using a deep neural network, and the objective function that is being minimised here is often of the form

$$\mathcal{L}(\boldsymbol{\theta}) = E\left[Q(\boldsymbol{s}_t, \boldsymbol{a}_t) - \left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \max_{\boldsymbol{a}} Q(\boldsymbol{s}_{t+1}, \boldsymbol{a})\right)\right]$$

- Q-learning can be fairly unstable when a neural network is used to represent the value function (and may not even converge) — but there are practical tricks to improve the convergence

- Value-based algorithms, such as TD(λ) and Q-learning, derive the policy from the value function

Policy Search

Reinforcement Learning

- Value-based algorithms, such as TD(λ) and Q-learning, derive the policy from the value function
- Policy search circumvents the need for the value function by **optimising in the policy space directly**

Policy Search

- Value-based algorithms, such as TD(λ) and Q-learning, derive the policy from the value function
- Policy search circumvents the need for the value function by **optimising in the policy space directly**
- **Policy search algorithms are useful in robotics since they allow incorporating prior knowledge about the policy**

Policy Gradients

Reinforcement Learning

- Policy gradient methods represent one popular family of policy search

Policy Gradients

Reinforcement Learning

- Policy gradient methods represent one popular family of policy search
- Given a parameterised policy $\pi_{\boldsymbol{\theta}}$, a **policy gradient algorithm estimates the gradient of the expected return and modifies the parameters $\boldsymbol{\theta}$** using the update rule

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \nabla J(\boldsymbol{\theta}) = \boldsymbol{\theta} + \nabla \int R(\tau) P(\tau | \boldsymbol{\theta}) d\tau$$

Policy Gradients

Reinforcement Learning

- Policy gradient methods represent one popular family of policy search
- Given a parameterised policy $\pi_{\boldsymbol{\theta}}$, a **policy gradient algorithm estimates the gradient of the expected return and modifies the parameters θ** using the update rule

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \nabla J(\boldsymbol{\theta}) = \boldsymbol{\theta} + \nabla \int R(\tau) P(\tau|\boldsymbol{\theta}) d\tau$$

- Policy gradient algorithms often make use of the so-called likelihood ratio trick

$$\nabla_{\boldsymbol{\theta}} P(\tau|\boldsymbol{\theta}) = P(\tau|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log P(\tau|\boldsymbol{\theta})$$

while estimating the gradient of J

REINFORCE Algorithm

Reinforcement Learning

- REINFORCE is an algorithm that forms the backbone of many practically used policy gradient algorithms
- The algorithm was originally formulated for neural network-based policies, but its general formulation is applicable to any differentiable policy
- A high-level overview of the algorithm is shown on the right

```
Initialise θ randomly
for i ← 1 to N do
    𝒯 ← {}
    for j ← 1 to M do
        τ ← sample(π_θ)
        𝒯 ← 𝒯 ∪ τ
    J_θ ← (1/M) Σ_{j=1}^{M} Σ_t r_t^j
    θ ← θ + ∇J_θ
```

Initialise $\boldsymbol{\theta}$ randomly
for $i \leftarrow 1$ **to** N **do**
  $\mathcal{T} \leftarrow \{\}$
  for $j \leftarrow 1$ **to** M **do**
    $\tau \leftarrow \texttt{sample}(\pi_{\boldsymbol{\theta}})$
    $\mathcal{T} \leftarrow \mathcal{T} \cup \tau$
  $J_{\boldsymbol{\theta}} \leftarrow \frac{1}{M} \sum_{j=1}^{M} \sum_t r_t^j$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \nabla J_{\boldsymbol{\theta}}$

Actor-Critic Learning

Reinforcement Learning

- Value-based algorithms can be referred to as **critic-based**, while policy search algorithm are also called **actor-based**

Actor-Critic Learning

Reinforcement Learning

- Value-based algorithms can be referred to as **critic-based**, while policy search algorithm are also called **actor-based**
- A combination of the two also exists — this forms the so-called **actor-critic family of RL algorithms, which estimate the value function and maintain a policy at the same time**

- Value-based algorithms can be referred to as **critic-based**, while policy search algorithm are also called **actor-based**
- A combination of the two also exists — this forms the so-called **actor-critic family of RL algorithms, which estimate the value function and maintain a policy at the same time**
- Actor-critic algorithms make use of a **baseline b when estimating the gradient of J**

$$\nabla_{\boldsymbol{\theta}} J(\theta) = E\left[\sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)(R_t - b_t)\right]$$

Actor-Critic Learning

Reinforcement Learning

- Value-based algorithms can be referred to as **critic-based**, while policy search algorithm are also called **actor-based**
- A combination of the two also exists — this forms the so-called **actor-critic family of RL algorithms, which estimate the value function and maintain a policy at the same time**
- Actor-critic algorithms make use of a **baseline** b **when estimating the gradient of** J

$$\nabla_{\boldsymbol{\theta}} J(\theta) = E\left[\sum_{i=0}^{T} \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}(\boldsymbol{a}_t | \boldsymbol{s}_t)(R_t - b_t)\right]$$

- The benefit of actor-critic algorithms is that the variance of policy updates is reduced

Proximal Policy Optimisation (PPO)

Reinforcement Learning

- PPO is a policy gradient algorithm that is often used as a baseline method in learning problems

Proximal Policy Optimisation (PPO)

Reinforcement Learning

- PPO is a policy gradient algorithm that is often used as a baseline method in learning problems
- The optimisation objective of PPO is maximising

$$\mathcal{L}(\boldsymbol{\theta}) = E\left[\min\left(q_t(\boldsymbol{\theta})A_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a}), \text{clip}\left(q_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon\right)A_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})\right)\right]$$

Proximal Policy Optimisation (PPO)

Reinforcement Learning

- PPO is a policy gradient algorithm that is often used as a baseline method in learning problems
- The optimisation objective of PPO is maximising

$$\mathcal{L}(\boldsymbol{\theta}) = E\left[\min\left(q_t(\boldsymbol{\theta}) A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a}), \text{clip}\left(q_t(\boldsymbol{\theta}), 1 - \epsilon, 1 + \epsilon\right) A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})\right)\right]$$

for a small ϵ where $A_{\boldsymbol{\theta}}$ is called the **advantage function**

$$A(\boldsymbol{s}, \boldsymbol{a}) = Q(\boldsymbol{s}, \boldsymbol{a}) - V(\boldsymbol{s})$$

Proximal Policy Optimisation (PPO)

Reinforcement Learning

- PPO is a policy gradient algorithm that is often used as a baseline method in learning problems
- The optimisation objective of PPO is maximising

$$\mathcal{L}(\boldsymbol{\theta}) = E\left[\min\left(q_t(\boldsymbol{\theta})A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a}), \text{clip}\left(q_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon\right) A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})\right)\right]$$

for a small ϵ where $A_{\boldsymbol{\theta}}$ is called the **advantage function**

$$A(\boldsymbol{s}, \boldsymbol{a}) = Q(\boldsymbol{s}, \boldsymbol{a}) - V(\boldsymbol{s})$$

and

$$q_t(\theta) = \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s})}{\pi_{\boldsymbol{\theta}_{old}}(\boldsymbol{a}|\boldsymbol{s})}$$

Proximal Policy Optimisation (PPO)

Reinforcement Learning

- PPO is a policy gradient algorithm that is often used as a baseline method in learning problems
- The optimisation objective of PPO is maximising

$$\mathcal{L}(\boldsymbol{\theta}) = E\left[\min\left(q_t(\boldsymbol{\theta}) A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a}), \operatorname{clip}\left(q_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon\right) A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})\right)\right]$$

for a small ϵ where $A_{\boldsymbol{\theta}}$ is called the **advantage function**

$$A(\boldsymbol{s}, \boldsymbol{a}) = Q(\boldsymbol{s}, \boldsymbol{a}) - V(\boldsymbol{s})$$

and

$$q_t(\theta) = \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s})}{\pi_{\boldsymbol{\theta}_{old}}(\boldsymbol{a}|\boldsymbol{s})}$$

$$\operatorname{clip}(x, y, z) = \begin{cases} y & \text{if } x < y \\ z & \text{if } x > z \\ x & \text{otherwise} \end{cases}$$

Proximal Policy Optimisation (PPO)

Reinforcement Learning

- PPO is a policy gradient algorithm that is often used as a baseline method in learning problems
- The optimisation objective of PPO is maximising

$$\mathcal{L}(\boldsymbol{\theta}) = E\left[\min\left(q_t(\boldsymbol{\theta})A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a}), \text{clip}\left(q_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon\right) A_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})\right)\right]$$

  for a small ϵ where $A_{\boldsymbol{\theta}}$ is called the **advantage function**

$$A(\boldsymbol{s}, \boldsymbol{a}) = Q(\boldsymbol{s}, \boldsymbol{a}) - V(\boldsymbol{s})$$

  and

$$q_t(\theta) = \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s})}{\pi_{\boldsymbol{\theta}_{old}}(\boldsymbol{a}|\boldsymbol{s})}$$

$$\text{clip}(x, y, z) = \begin{cases} y & \text{if } x < y \\ z & \text{if } x > z \\ x & \text{otherwise} \end{cases}$$

- PPO maintains a (deep) policy network (thus it is considered a deep RL algorithm) and tries to limit the amount by which the policy is updated

Imitation Learning

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- In reinforcement learning, **a robot needs to interact with its environment** (either in the real world or in a simulation) **so that it can identify an appropriate execution policy**

Imitation Learning

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- In reinforcement learning, **a robot needs to interact with its environment** (either in the real world or in a simulation) **so that it can identify an appropriate execution policy**
- If an expert is available that can show the desired behaviour, a more appropriate way is to perform **imitation learning**

Imitation Learning

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- In reinforcement learning, **a robot needs to interact with its environment** (either in the real world or in a simulation) **so that it can identify an appropriate execution policy**
- If an expert is available that can show the desired behaviour, a more appropriate way is to perform **imitation learning**
- There are various techniques to perform imitation learning:

Imitation Learning

- In reinforcement learning, **a robot needs to interact with its environment** (either in the real world or in a simulation) **so that it can identify an appropriate execution policy**
- If an expert is available that can show the desired behaviour, a more appropriate way is to perform **imitation learning**
- There are various techniques to perform imitation learning:

Behaviour cloning

An expert policy is mimicked directly based on observed states and actions

Imitation Learning

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- In reinforcement learning, **a robot needs to interact with its environment** (either in the real world or in a simulation) **so that it can identify an appropriate execution policy**
- If an expert is available that can show the desired behaviour, a more appropriate way is to perform **imitation learning**
- There are various techniques to perform imitation learning:

Behaviour cloning

An expert policy is mimicked directly based on observed states and actions

Inverse reinforcement learning

Expert demonstrations are used for extracting a reward function

Imitation Learning

- In reinforcement learning, **a robot needs to interact with its environment** (either in the real world or in a simulation) **so that it can identify an appropriate execution policy**
- If an expert is available that can show the desired behaviour, a more appropriate way is to perform **imitation learning**
- There are various techniques to perform imitation learning:

Behaviour cloning	Inverse reinforcement learning	Learning from observation
An expert policy is mimicked directly based on observed states and actions	Expert demonstrations are used for extracting a reward function	A policy is learned from raw observations, without explicit state and action labels

Behaviour Cloning

Imitation Learning

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- The simplest way to perform imitation learning is to **copy the actions performed by the demonstrator**, an approach known as behaviour cloning

Behaviour Cloning

Imitation Learning

- The simplest way to perform imitation learning is to **copy the actions performed by the demonstrator**, an approach known as behaviour cloning
- In behaviour cloning, we are given a set of c observations $X = \{(\boldsymbol{s}_i, \boldsymbol{a}_i)\}, 1 \leq i \leq \mathrm{c}$ that specifies **states and ground-truth actions taken by an expert demonstrator**

Behaviour Cloning

Imitation Learning

- The simplest way to perform imitation learning is to **copy the actions performed by the demonstrator**, an approach known as behaviour cloning
- In behaviour cloning, we are given a set of c observations $X = \{(\boldsymbol{s}_i, \boldsymbol{a}_i)\}, 1 \leq i \leq \mathrm{c}$ that specifies **states and ground-truth actions taken by an expert demonstrator**
- Given such a dataset, supervised learning can be used to acquire a policy

Behaviour Cloning

Imitation Learning

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- The simplest way to perform imitation learning is to **copy the actions performed by the demonstrator**, an approach known as behaviour cloning
- In behaviour cloning, we are given a set of c observations $X = \{(\boldsymbol{s}_i, \boldsymbol{a}_i)\}, 1 \leq i \leq \mathrm{c}$ that specifies **states and ground-truth actions taken by an expert demonstrator**
- Given such a dataset, supervised learning can be used to acquire a policy
- A policy learned using behaviour cloning can be **further improved using reinforcement learning, but also using corrective demonstrations**

Inverse Reinforcement Learning

Imitation Learning

- Another way in which expert demonstrations can be utilised is for **extracting a reward function** — this is the approach taken by inverse reinforcement learning (aka reward inference)

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

Inverse Reinforcement Learning

Imitation Learning

- Another way in which expert demonstrations can be utilised is for **extracting a reward function** — this is the approach taken by inverse reinforcement learning (aka reward inference)
- The reward in inverse RL is typically represented as **a (linear) combination of features that can be observed**

Inverse Reinforcement Learning

Imitation Learning

- Another way in which expert demonstrations can be utilised is for **extracting a reward function** — this is the approach taken by inverse reinforcement learning (aka reward inference)
- The reward in inverse RL is typically represented as **a (linear) combination of features that can be observed**
- Such a reward function can then be used to do reinforcement learning
  - Inverse RL is usually performed as an iterative process that has **an outer loop for reward extraction** (based on some optimisation metric) and **an inner loop for policy learning**

Inverse Reinforcement Learning

Imitation Learning

- Another way in which expert demonstrations can be utilised is for **extracting a reward function** — this is the approach taken by inverse reinforcement learning (aka reward inference)
- The reward in inverse RL is typically represented as **a (linear) combination of features that can be observed**
- Such a reward function can then be used to do reinforcement learning
  - Inverse RL is usually performed as an iterative process that has **an outer loop for reward extraction** (based on some optimisation metric) and **an inner loop for policy learning**
- Inverse RL is challenging because **the problem is ill-defined** — there can be many possible reward functions that optimise the metric

Policy Transfer

- One additional strategy in which a policy π for a new task T_j can be acquired is to **reuse a policy $\pi^{T_i^*}$ that has already been learned for a different task $T_i, i \neq j$**

Policy Transfer

- One additional strategy in which a policy π for a new task T_j can be acquired is to **reuse a policy $\pi^{T_i^*}$ that has already been learned for a different task $T_i, i \neq j$**
- $\pi^{T_i^*}$ can be used either directly for T_j or, more frequently, fine-tuned for T_j

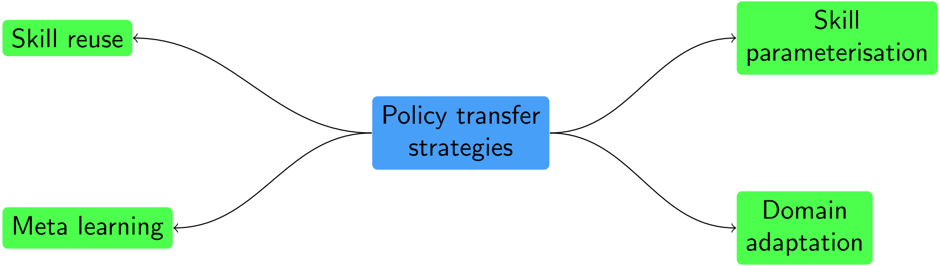
Policy Transfer

- One additional strategy in which a policy π for a new task T_j can be acquired is to **reuse a policy $\pi^{T_i^*}$ that has already been learned for a different task $T_i, i \neq j$**
- $\pi^{T_i^*}$ can be used either directly for T_j or, more frequently, fine-tuned for T_j
- This can also be achieved using a variety of strategies, which we will not discuss in further detail in this lecture

Policy Transfer

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- One additional strategy in which a policy π for a new task T_j can be acquired is to **reuse a policy $\pi^{T_i^*}$ that has already been learned for a different task $T_i, i \neq j$**
- $\pi^{T_i^*}$ can be used either directly for T_j or, more frequently, fine-tuned for T_j
- This can also be achieved using a variety of strategies, which we will not discuss in further detail in this lecture



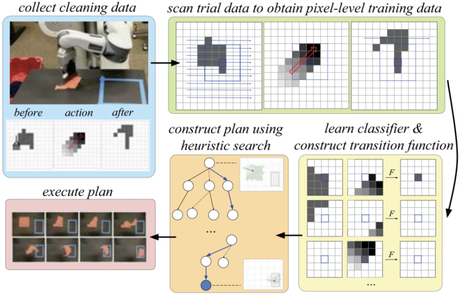
Skill Learning

- In this section, we only discussed the aspect of learning a policy
- In the previous lecture, we defined a complete skill as $\mathcal{S} = (\mathcal{S}_{\mathcal{I}}, \mathcal{S}_{\mathcal{T}}, \pi)$, namely a skill also has initiation and termination conditions — what about those, you might ask?
- The aspect of learning the initiation conditions (preconditions) and the termination condition is left out on purpose; this will be discussed in a dedicated session later in the course

Transition Model Learning

Transition Models for State Prediction

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

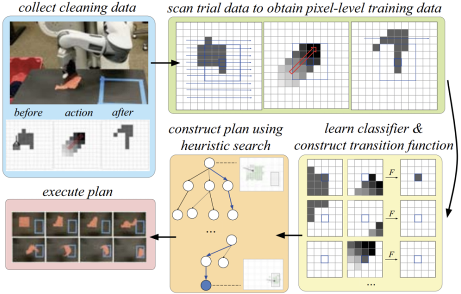


S. Elliott and M. Cakmak, "Robotic Cleaning Through Dirt Rearrangement Planning with Learned Transition Models," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 1623–1630.

- As a robot can affect its environment with its own actions, **it can be useful for it to know how those actions affect the state** before committing to specific actions

Transition Models for State Prediction

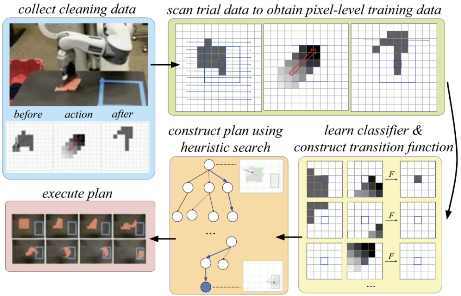
Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris



S. Elliott and M. Cakmak, "Robotic Cleaning Through Dirt Rearrangement Planning with Learned Transition Models," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 1623–1630.

- As a robot can affect its environment with its own actions, **it can be useful for it to know how those actions affect the state** before committing to specific actions
- **Prediction is also useful when a robot co-exists with other agents**

Transition Models for State Prediction

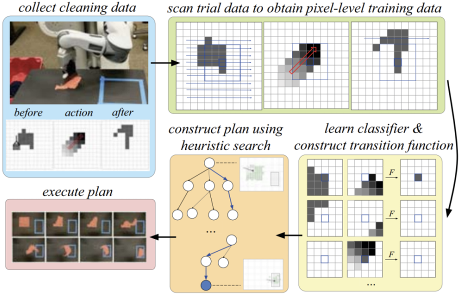


S. Elliott and M. Cakmak, "Robotic Cleaning Through Dirt Rearrangement Planning with Learned Transition Models," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 1623–1630.

- As a robot can affect its environment with its own actions, **it can be useful for it to know how those actions affect the state** before committing to specific actions
- **Prediction is also useful when a robot co-exists with other agents**
- A **transition model** $\mathcal{T}$ enables such predictions about the state evolution as a result of executed actions to be created

Transition Models for State Prediction

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris



S. Elliott and M. Cakmak, "Robotic Cleaning Through Dirt Rearrangement Planning with Learned Transition Models," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 1623–1630.

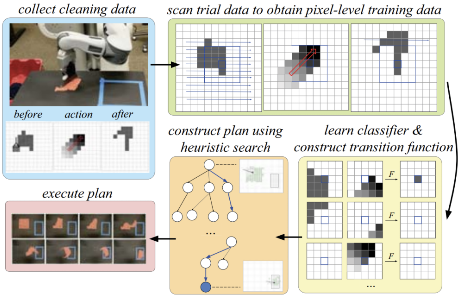
- As a robot can affect its environment with its own actions, **it can be useful for it to know how those actions affect the state** before committing to specific actions
- **Prediction is also useful when a robot co-exists with other agents**
- A **transition model** $\mathcal{T}$ enables such predictions about the state evolution as a result of executed actions to be created
- Depending on the nature of the predictive process, we can distinguish between two types of transition models:

Transition Models for State Prediction

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris



S. Elliott and M. Cakmak, "Robotic Cleaning Through Dirt Rearrangement Planning with Learned Transition Models," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 1623–1630.

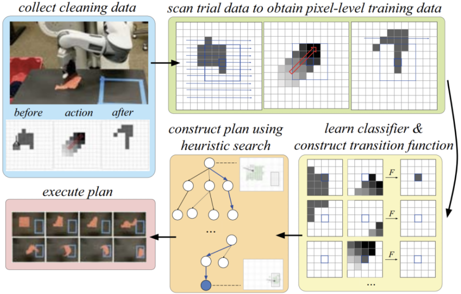
- As a robot can affect its environment with its own actions, **it can be useful for it to know how those actions affect the state** before committing to specific actions
- **Prediction is also useful when a robot co-exists with other agents**
- A **transition model** $\mathcal{T}$ enables such predictions about the state evolution as a result of executed actions to be created
- Depending on the nature of the predictive process, we can distinguish between two types of transition models:

Deterministic transition model

$$\mathcal{T} : S \times A \to S$$

Transition Models for State Prediction

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris



S. Elliott and M. Cakmak, "Robotic Cleaning Through Dirt Rearrangement Planning with Learned Transition Models," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 1623–1630.

- As a robot can affect its environment with its own actions, **it can be useful for it to know how those actions affect the state** before committing to specific actions
- **Prediction is also useful when a robot co-exists with other agents**
- A **transition model** $\mathcal{T}$ enables such predictions about the state evolution as a result of executed actions to be created
- Depending on the nature of the predictive process, we can distinguish between two types of transition models:

Deterministic transition model	Probabilistic transition model
$\mathcal{T} : S \times A \rightarrow S$	$\mathcal{T} : S \times A \times S \rightarrow \mathbb{R}$

Discrete and Continuous Transition Models

- There are two types of predictive models based on the nature of the state variables to be predicted:

Discrete and Continuous Transition Models

- There are two types of predictive models based on the nature of the state variables to be predicted:

Continuous predictive model

Used for predicting the evolution of continuous state variables

Discrete and Continuous Transition Models

- There are two types of predictive models based on the nature of the state variables to be predicted:

Continuous predictive model

Used for predicting the evolution of continuous state variables

Discrete predictive model

Used when the state space can be discretised (e.g. using a symbolic representation)

Discrete and Continuous Transition Models

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21

A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer* Scott Niekum George Konidaris

- There are two types of predictive models based on the nature of the state variables to be predicted:

Continuous predictive model

Used for predicting the evolution of continuous state variables

Discrete predictive model

Used when the state space can be discretised (e.g. using a symbolic representation)

- Discrete and continuous transition models can be combined in a **hybrid model** to enable the representation of different **manipulation modes**
  - Here, a discrete model is used to predict mode transitions
  - A continuous model is used to predict state variables within a mode

Model Uncertainty

- Probabilistic transition models have an associated uncertainty that stems from the fact that **a robot does not have perfect knowledge about the process to be predicted**

Model Uncertainty

Journal of Machine Learning Research 22 (2021) 1-82 Submitted 9/19; Revised 8/20; Published 1/21
A Review of Robot Learning for Manipulation:
Challenges, Representations, and Algorithms
Oliver Kroemer* Scott Niekum George Konidaris

- Probabilistic transition models have an associated uncertainty that stems from the fact that **a robot does not have perfect knowledge about the process to be predicted**
- In this context, we need to distinguish between two sources of uncertainty:

Model Uncertainty

- Probabilistic transition models have an associated uncertainty that stems from the fact that **a robot does not have perfect knowledge about the process to be predicted**
- In this context, we need to distinguish between two sources of uncertainty:

Aleatoric uncertainty

Inherent uncertainty in the process

Model Uncertainty

- Probabilistic transition models have an associated uncertainty that stems from the fact that **a robot does not have perfect knowledge about the process to be predicted**
- In this context, we need to distinguish between two sources of uncertainty:

Aleatoric uncertainty	Epistemic uncertainty
Inherent uncertainty in the process	Uncertainty due to a lack of process knowledge

Model Uncertainty

- Probabilistic transition models have an associated uncertainty that stems from the fact that **a robot does not have perfect knowledge about the process to be predicted**
- In this context, we need to distinguish between two sources of uncertainty:

Aleatoric uncertainty

Inherent uncertainty in the process

Epistemic uncertainty

Uncertainty due to a lack of process knowledge

- Epistemic uncertainty can be **minimised with more training data (or with interactive perception)**; this is not the case with aleatoric uncertainty, **where more data cannot help**

Inverse Models

- Predictive models as discussed so far are often called **forward models**

Inverse Models

- Predictive models as discussed so far are often called **forward models**
- In some cases, it can be useful to **know how the state was changed in a particular way** (e.g. when observing other agents performing tasks and only the state is observable)

Inverse Models

- Predictive models as discussed so far are often called **forward models**
- In some cases, it can be useful to **know how the state was changed in a particular way** (e.g. when observing other agents performing tasks and only the state is observable)
- An inverse model **makes a prediction of the action that is performed so that a certain state transition occurs**

$$\mathcal{T}^{-1} : S \times S \rightarrow A$$

Inverse Models

- Predictive models as discussed so far are often called **forward models**
- In some cases, it can be useful to **know how the state was changed in a particular way** (e.g. when observing other agents performing tasks and only the state is observable)
- An inverse model **makes a prediction of the action that is performed so that a certain state transition occurs**

$$\mathcal{T}^{-1} : S \times S \rightarrow A$$

- Inverse models can be learned similarly to predictive models