Deep Reinforcement Learning for Autonomous Systems

Designing a control system to exploit model-free deep reinforcement learning algorithms to solve a real-world autonomous driving task of a small robot.

Candidate: Piero Macaluso

Supervisors: Prof. Pietro Michiardi — EURECOM, France
Prof. Elena Baralis — Politecnico di Torino, Italy

March 12, 2020

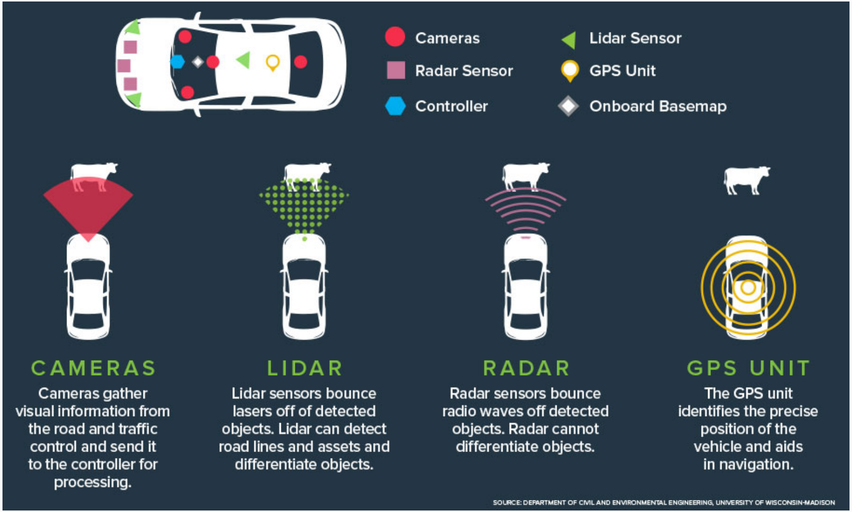
This work of this thesis was developed at EURECOM (Sophia Antipolis, France)
in collaboration with

Prof. Pietro Michiardi (EURECOM)
Prof. Elena Baralis (Politecnico di Torino)

Table of contents

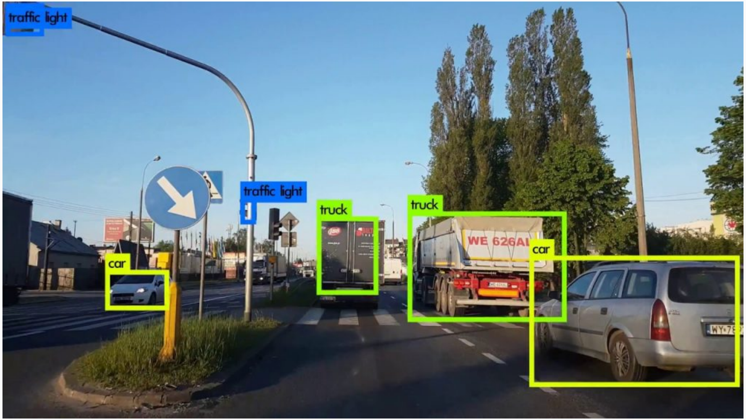
Background

State-of-the-art Autonomous Driving Systems



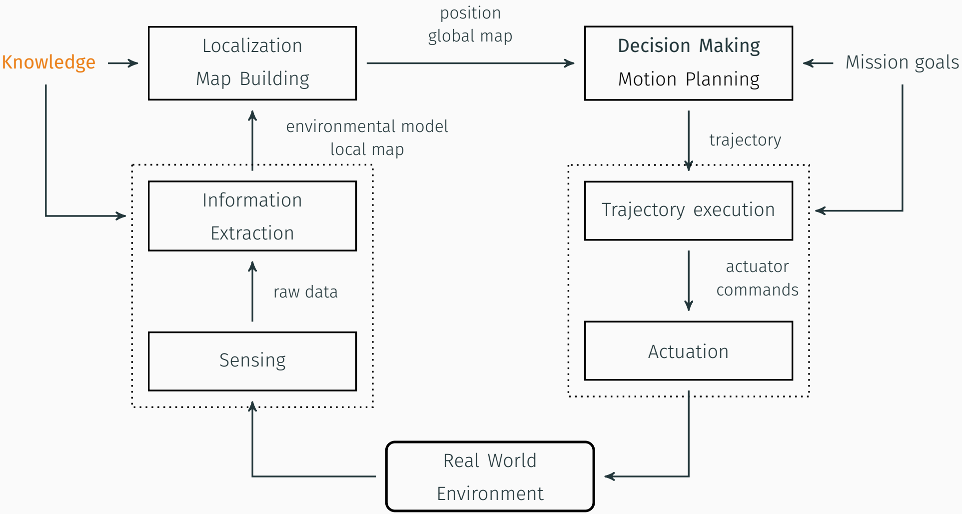
GovTech: Government Technology, *Autonomous Vehicles: Coming to a Road Near You*.

State-of-the-art Autonomous Driving Systems



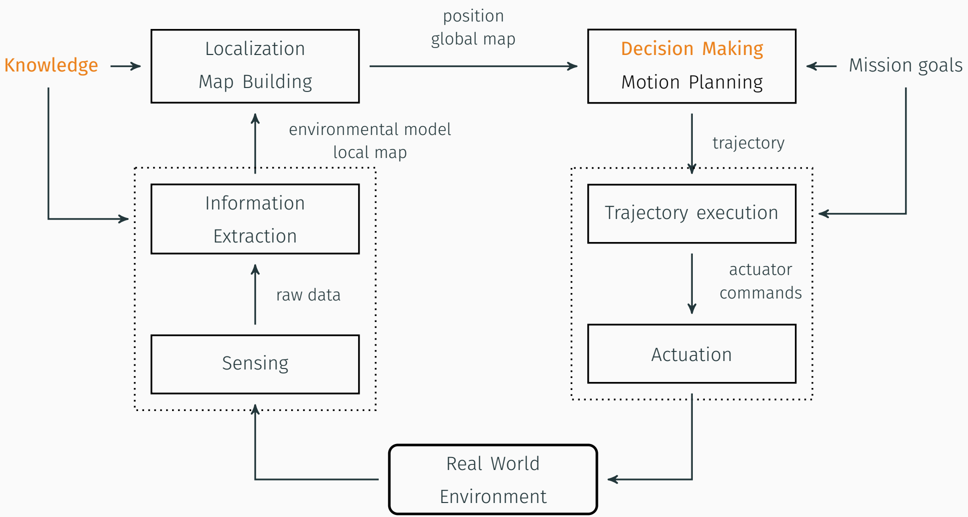
Deep Learning and **Machine Learning** are mainly exploited in **object detection** and **recognition**.

State-of-the-art Autonomous Driving Systems



Pavone, *Veicoli a guida autonoma: a che punto siamo e cosa ci aspetta?*

State-of-the-art Autonomous Driving Systems

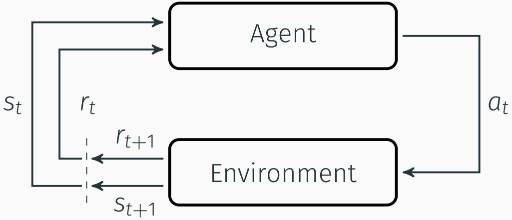


Pavone, *Veicoli a guida autonoma: a che punto siamo e cosa ci aspetta?*

Reinforcement Learning

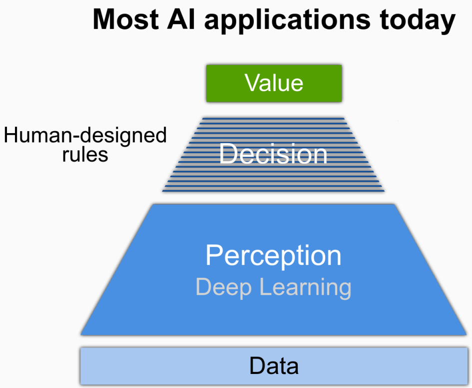
Problems involving an **agent** interacting with an **environment**, which provides numeric **reward signals**.

Goal: Learn how to take actions in order to maximize a reward function.



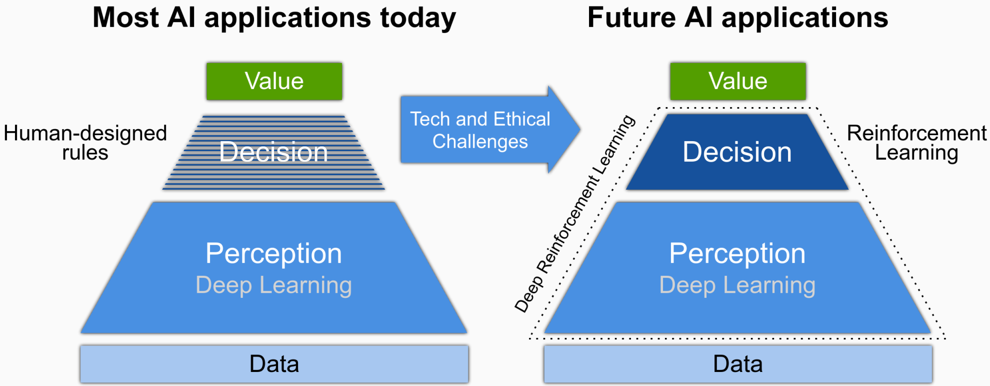
Sutton and Barto, *Reinforcement learning: An introduction*.

From Data to Value



Charafeddine, *Reinforcement Learning in the Wild and Lessons Learned*.

From Data to Value



Charafeddine, *Reinforcement Learning in the Wild and Lessons Learned.*

Components of the Agent

- **Policy**: agent's behaviour function

$$\textbf{Deterministic}: \pi(s) = a$$

$$\textbf{Stochastic}: \pi(a|s) = \mathbb{P}[A_t = a | S_t = s]$$

- **Value Function**: policy evaluation function

$$\textbf{State Value}: V^{\pi}(s) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^k r_t | s_0 = s, \pi\right]$$

$$\textbf{Action Value}: Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^k r_t \middle| s_0 = s, a_0 = a, \pi\right]$$

- **Model**: agent's representation of the environment

Categorizing Reinforcement Learning agents

- Value Based
  - No Policy (implicit)
  - Value Function
- Policy Based
  - Policy
  - No value function
- Actor Critic
  - Policy
  - Value function

- Model Free
  - Policy and/or value function
  - No Model
- Model Based
  - Policy and/or value function
  - Model

Categorizing Reinforcement Learning agents

- Value Based
  - No Policy (implicit)
  - Value Function
- Policy Based
  - Policy
  - No value function
- Actor Critic
  - Policy
  - Value function

- Model Free
  - Policy and/or value function
  - No Model
- Model Based
  - Policy and/or value function
  - Model

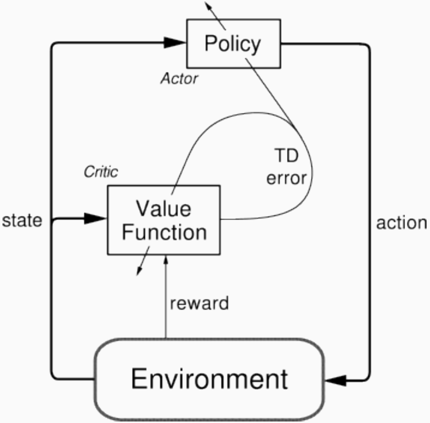
Model-Free Actor Critic methods

Critic Network

Estimates the value function. This could be the action value Q or state value V.

Actor Network

Updates the policy distribution in the direction suggested by the Critic (such as with policy gradients).



Sutton and Barto, *Reinforcement learning: An introduction*.

Deep Deterministic Policy Gradient (DDPG)

- Off-Policy:
  - Experience Replay Memory of $(s_t, a_t, r_t, s_t + 1, d_t)$ tuples
- Action space: Countinuous
- Policy: Deterministic
- Exploration:
  - Ornstein–Uhlenbeck process noise
  - Noise regulation with ϵ-decay function

Needs accurate hyper-parameters fine-tuning

Lillicrap et al., “Continuous control with deep reinforcement learning”.

Deep Deterministic Policy Gradient (DDPG) - Neural Networks

It uses **Target Networks** to minimise the instability MSBE loss

2 Local Neural Networks:

- Actor $\pi(s \mid \theta)$
- Critic $Q(s, a \mid \phi)$

2 Target Neural Networks:

- Actor $\pi'(s \mid \bar{\theta})$
- Critic $Q'(s, a \mid \bar{\phi})$

Deep Deterministic Policy Gradient (DDPG) - Learning Equations

$$
\begin{aligned}
L(\phi) &= \mathbb{E}_{s_t \sim \rho^{\beta}, a_t \sim \beta, r_t \sim E}[(Q(s_t, a_t|\phi) - y_t)^2] \\
y_t &= r(s_t, a_t) + \gamma(1 - d_t)Q'(s_{t+1}, \pi'(s_t + 1|\bar{\theta})|\bar{\phi})
\end{aligned} \tag{1}
$$

Lillicrap et al., "Continuous control with deep reinforcement learning".

Soft Actor-Critic (SAC)

- **Off-Policy**
- **Action space**: Countinuous
- **Policy**: **Stochastic**
- **Exploration**: **Temperature** Parameter
- SAC is an off-policy algorithm which exploits entropy-regularized reinforcement learning
- Auto-tune parameters: Less hyper-parameters, less tuning
- Suitable for Real-World Experiments

Haarnoja et al., "Soft actor-critic algorithms and applications".

Design of the control system

Experimental methodology and results

Conclusions and future work

Questions?

Thank you!

References

- Charafeddine, Mohamad. *Reinforcement Learning in the Wild and Lessons Learned*. 2018. URL: `https://link.medium.com/SRUZ24Itx4` (visited on 10/26/2018).
- GovTech: Government Technology. *Autonomous Vehicles: Coming to a Road Near You*. 2018. URL: `https://www.govtech.com/transportation/Autonomous-Vehicles-Coming-to-a-Road-Near-You.html`.
- Haarnoja, Tuomas et al. “Soft actor-critic algorithms and applications”. In: *arXiv preprint arXiv:1812.05905* (2018).

References ii

Lillicrap, Timothy P et al. "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971* (2015).

Pavone, Marco. *Veicoli a guida autonoma: a che punto siamo e cosa ci aspetta?* Festival della Tecnologia Conference. 2019.

Sutton, Richard S and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.