Deep Reinforcement Learning for Energy-Efficient Control of Autonomous Electric Vehicles

Dr. Andrea Pozzi

Applied Machine Learning Days at EPFL
March 26-30, 2022 STCC, Lausanne, Switzerland

Agenda

- **Introduction**
- Eco-Driving Task
- Overview on Reinforcement Learning
- Problem Formulation
- Simulative Results

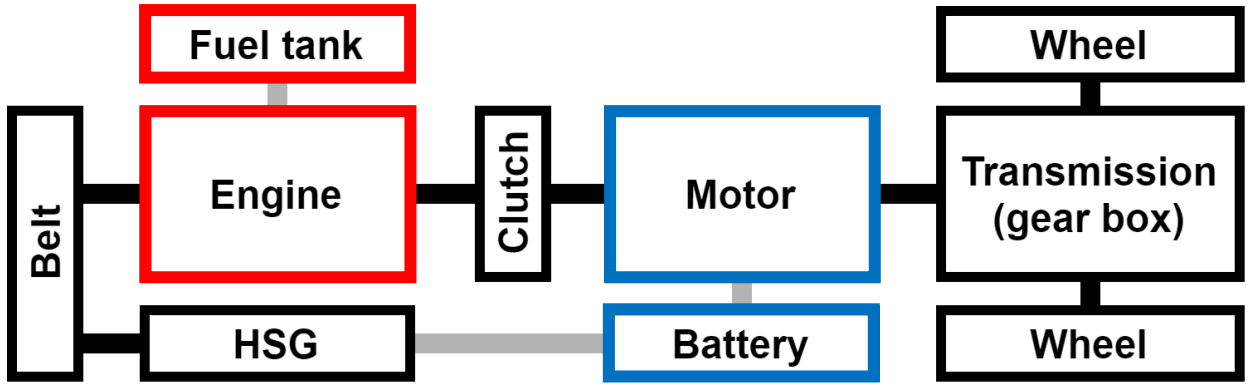
Hybrid and Electric Vehicles

The **growth in the (hybrid) electric vehicles market** is due to both the need for sustainable mobility and pollution reduction.

Why Hybrid Electric Vehicles?

HEVs are characterized by the fact that they combine an **electric motor** with an **internal combustion engine**.

Sciarretta et al., 2007



Autonomous Vehicles

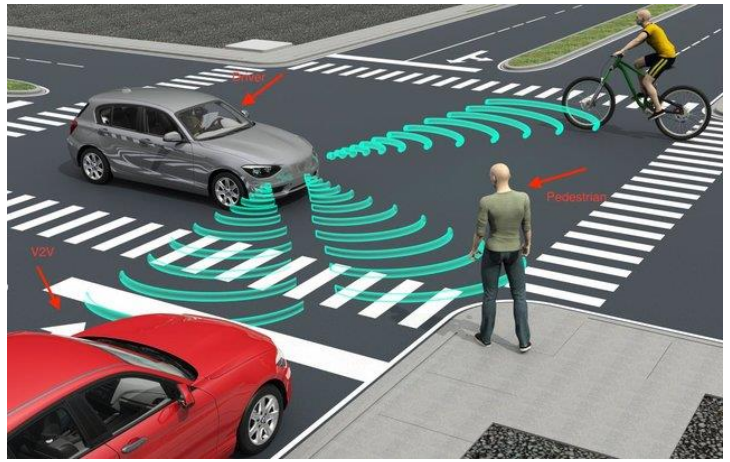
In parralel to HEVs, a huge effort of the research community has also been devoted to the development of **autonomous vehicles.**

Guanetti et al., 2018

By removing the need for a human driver is expected to **reduce both the traffic and the frequency of car accidents**.

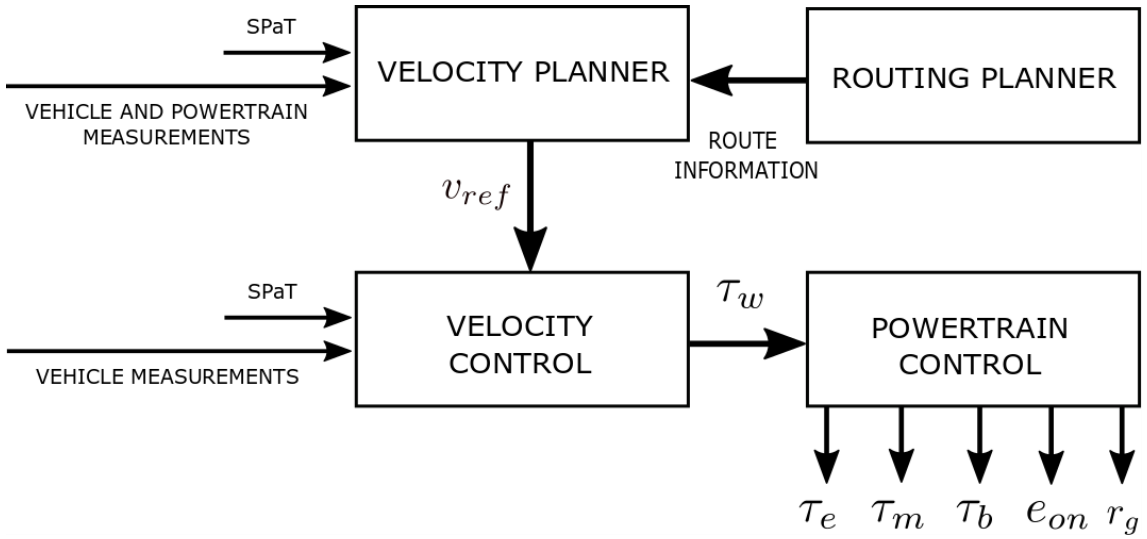
Connected Autonomous Vehicles (CAVs)

An autonomous vehicle requires to be **connected with the surrounding environment** (other vehicles, traffic lights, etc) in order to avoid collisions.



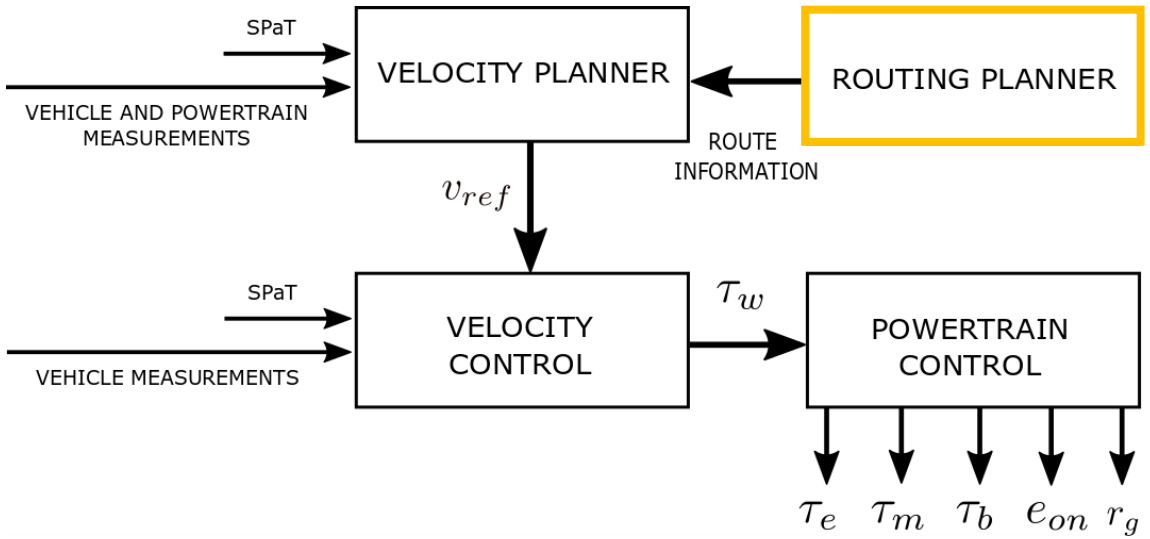
Hierarchical Control for Autonomous HEVs

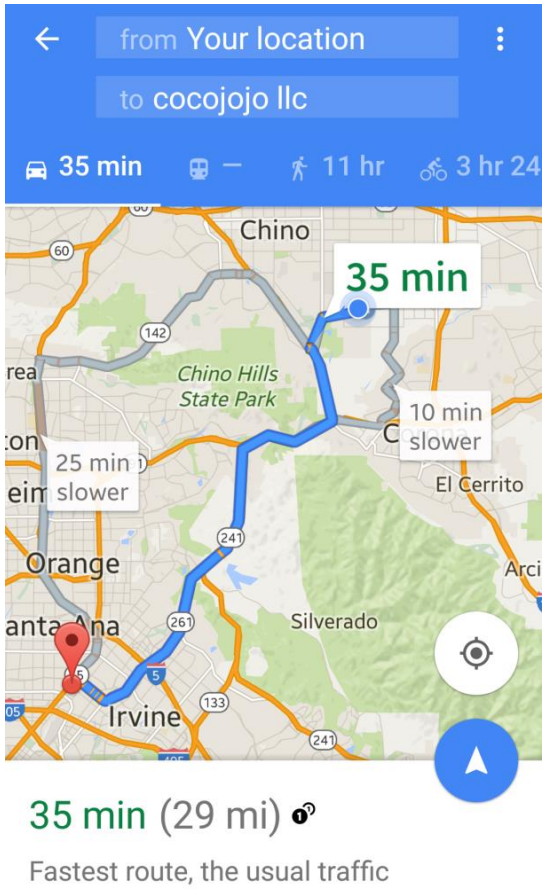
The control of autonomous HEVs is organized in a **hierarchical control scheme** and involves different tasks.



Route Planner

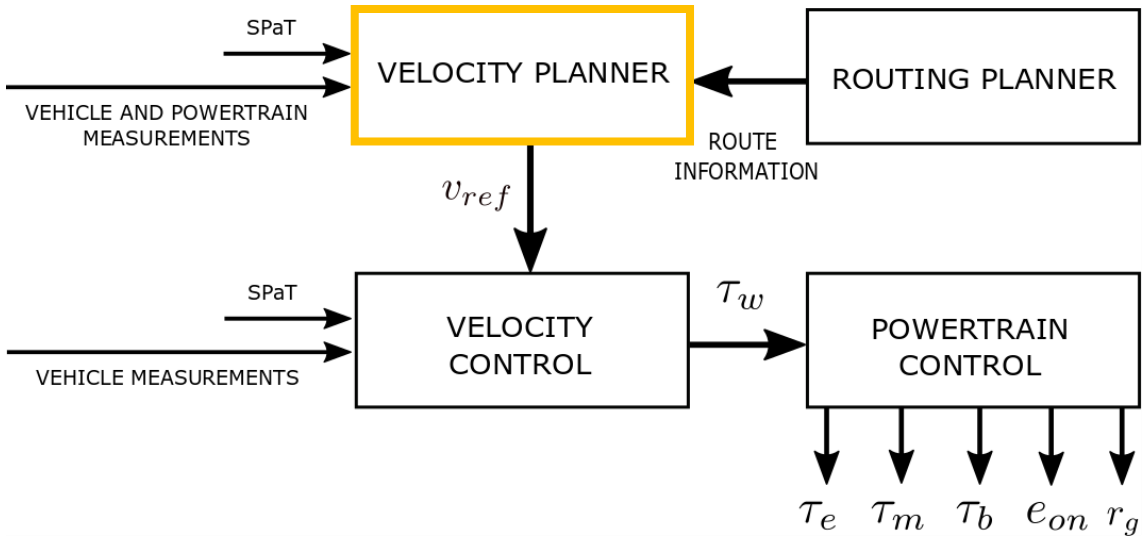
The route planner finds the **shortest path between the current position and the destination.**





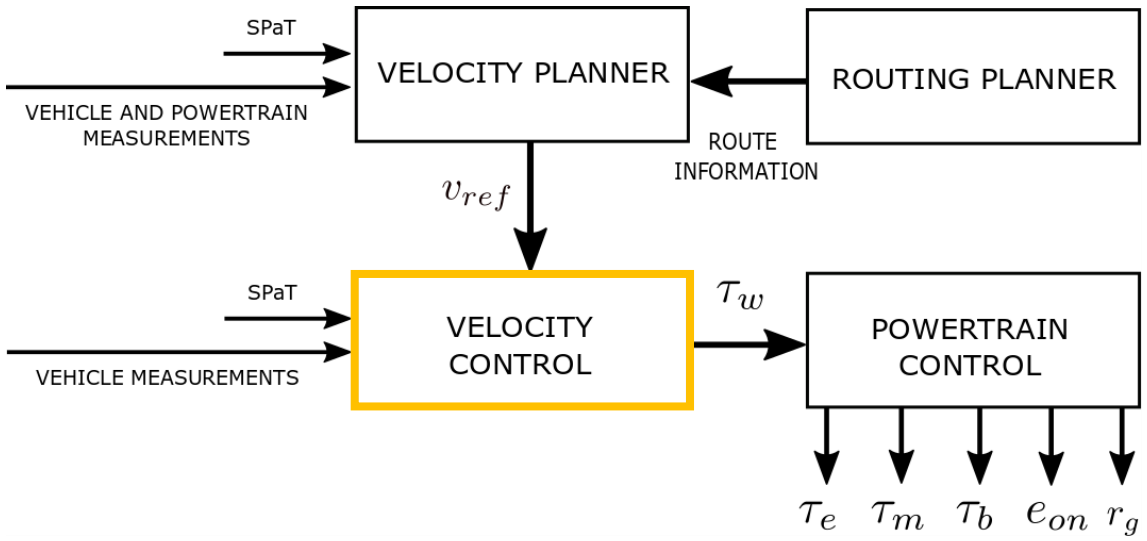
Velocity Planner

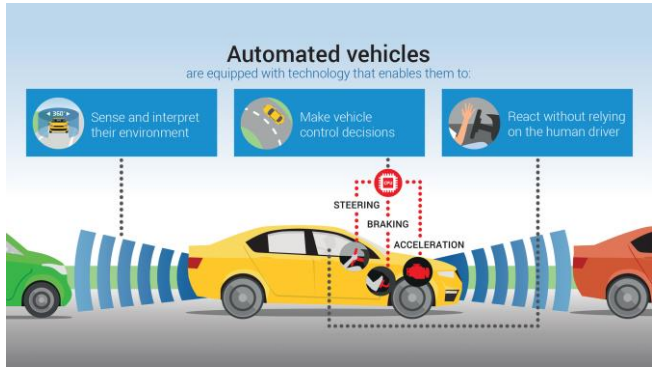
It **finds the velocity profile** along a specific path which deals with the trade-off between **travel time and energy efficiency**.



Velocity Control

The velocity control **tracks the reference velocity while ensuring safety** and avoiding collisions by designing the wheel torque.

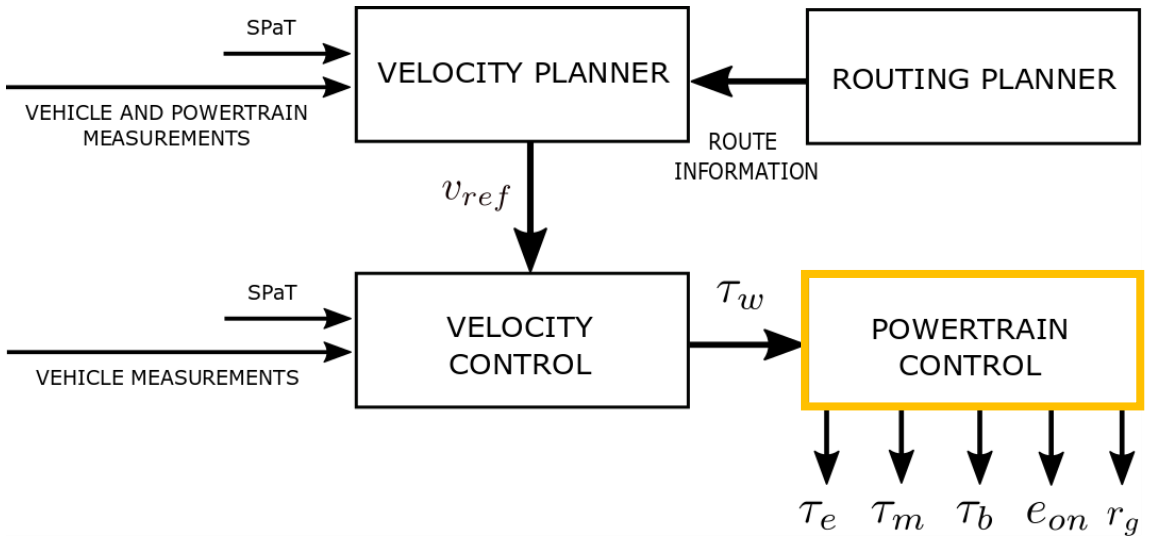


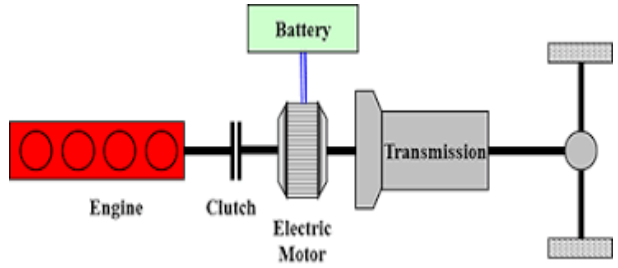


Powertrain Control

The powertrain control **splits the demanded power** between the electric motor and the engine.

Liu et al., 2008



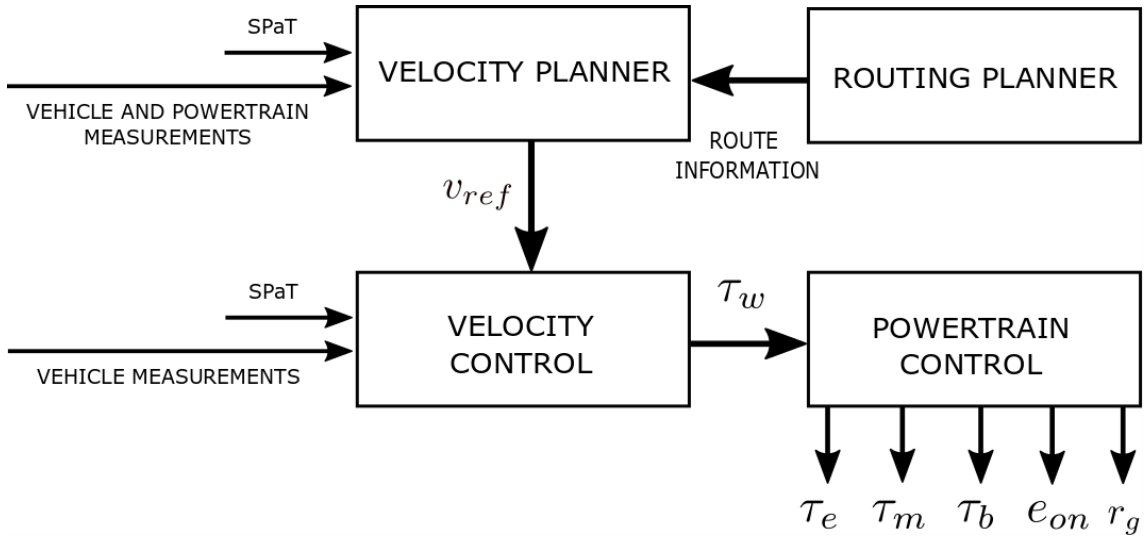


Goal of the Work

In this presentation we focus on the design of an **ecological velocity planning** for autonomous HEVs along a given path.

Previous works in this field rely on **model-based approaches**

- *Dib et al., 2014*
- *Sciarretta et al., 2015*
- *Han et al., 2018*

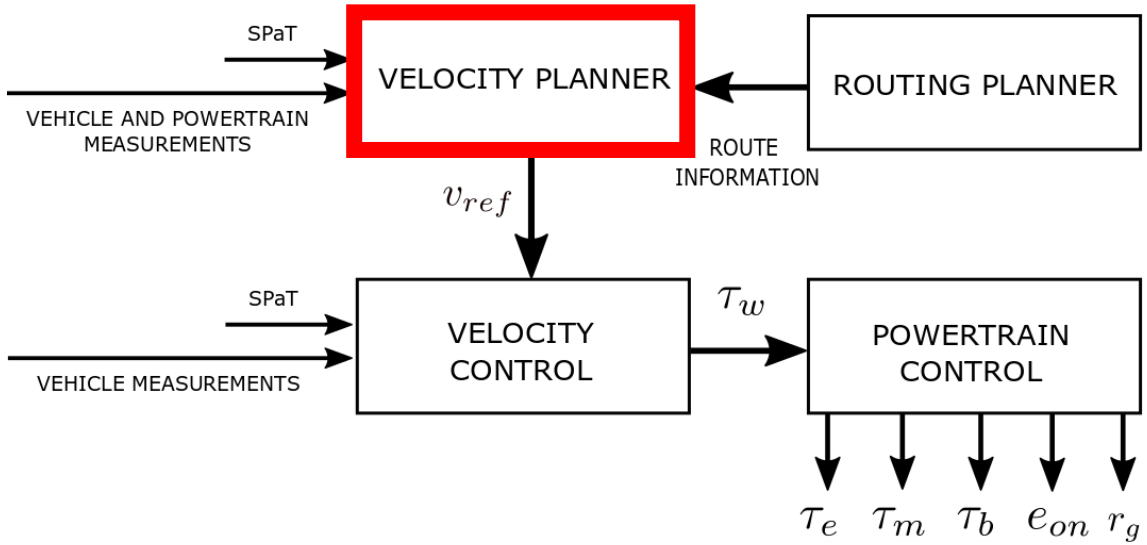


Goal of the Work

In this presentation we focus on the design of an **ecological velocity planning** for autonomous HEVs along a given path.

Previous works in this field rely on **model-based approaches**

- *Dib et al., 2014*
- *Sciarretta et al., 2015*
- *Han et al., 2018*



Agenda

- Introduction
- **Eco-Driving Task**
- Overview on Reinforcement Learning
- Problem Formulation
- Simulative Results

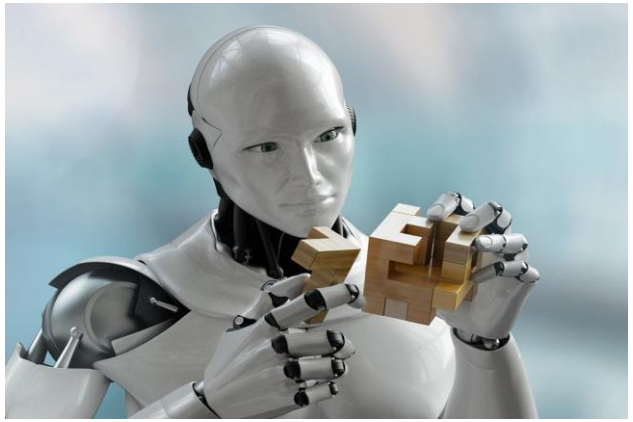
Issues of model-based eco-driving

Model-based ecological velocity planning usually exhibits the following **issues**:

- **high computational cost**
- **curse of dimensionality**
- **need for an accurate model**

RL-Based Velocity Planner

The **solution** can be the use of **model-free reinforcement learning** for online velocity planning, in order to minimize travel time and fuel consumption.



RL-Based Velocity Planner

RL has already been exploited in the context of **power management** for HEVs.

Liu and Murphey, 2019

However, its **use for optimal eco-driving constitutes a source of novelty**.

MPC for Safe Crossing

RL alone is not suitable for safety-critical problems, such as autonomous driving.

Garcia and Fernandez, 2015

Therefore, a **velocity control module** based on **MPC** is used to ensure safe crossing at the intersections.

Agenda

- Introduction
- Eco-Driving Task
- **Overview on Reinforcement Learning**
- Problem Formulation
- Simulative Results

MDP Framework

Consider a **Markov Decision Process** (MDP), where the state transition and the collected reward are defined through the following probability distributions

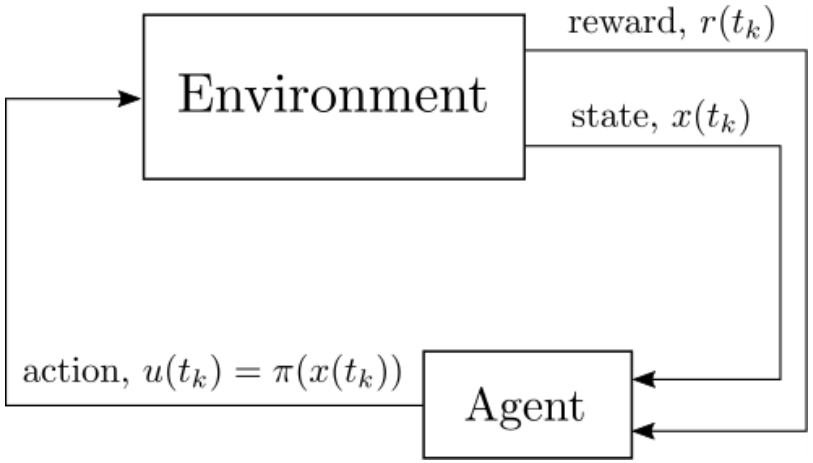
Sutton and Barto, 1998

$$p_x(x(t_{k+1})|x(t_k), u(t_k))$$

$$p_r(r(t_k)|x(t_{k+1}), x(t_k), u(t_k))$$

with the action given by

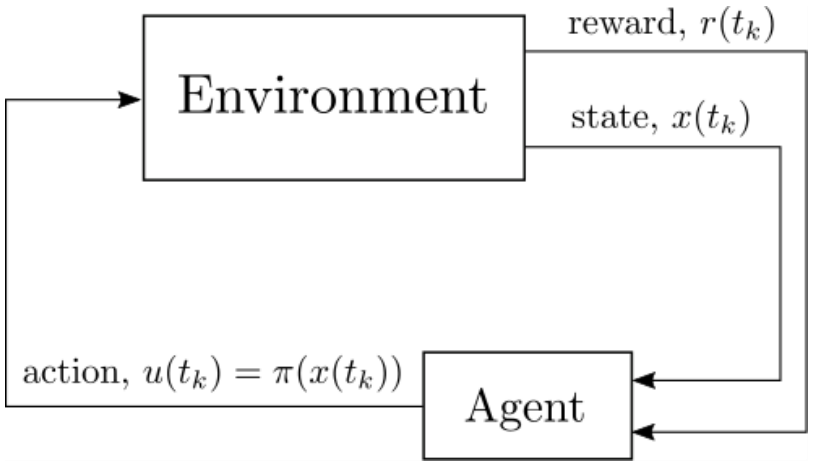
$$u(t_k) = \pi(x(t_k))$$



Solve an MDP

Solving an MDP consists in selecting the policy π: $X \rightarrow U$ which **maximizes the long term expected return**

$$\max_{\pi} \quad E\left[\sum_{\overline{k}=k}^{\infty} \gamma^{\overline{k}} r(t_{\overline{k}}) | x(t_k)\right]$$



The reward function has to be designed by the control engineers **in order to accomplish a desired task**.

RL Definitions

The long term expected return of a policy π is also known as the **state value function**

$$V^{\pi}(x(t_k)) = E\left[\sum_{\overline{k}=k}^{\infty} \gamma^{\overline{k}} r(t_{\overline{k}}) | x(t_k)\right]$$

The **state-action value function** of a policy π corresponds to the long-term expected return when action $u(t_k)$ is taken in state $x(t_k)$ and then the policy π is followed henceforth:

$$Q^{\pi}(x(t_k), u(t_k)) = E\left[\sum_{\overline{k}=k}^{\infty} \gamma^{\overline{k}} r(t_{\overline{k}}) | x(t_k), u(t_k)\right]$$

RL Definitions

The following equations define the optimum for the state-action value function and its relationship with the optimal state value one

$$Q^\star(x(t_k), u(t_k)) = \max_\pi Q^\pi(x(t_k), u(t_k))$$

$$V^\star(x(t_k)) = \max_{u(t_k)\in\mathcal{U}} Q^\star(x(t_k), u(t_k))$$

The optimal policy is then defined as

$$u^\star(t_k) = \arg\max_{u(t_k)\in\mathcal{U}} Q^\star(x(t_k), u(t_k))$$

The RL-agent builds the function $Q^\star$ by interacting with the environment.

Reinforcement Learning Algorithms

The main RL algorithms can be divided in two main groups:

- **Tabular methods**: cumulative reward expressed using tables with states and actions. -> Suitable **for small and discrete spaces** (curse of dimensionality).

- **Approximate Dynamic Programming (ADP)**: cumulative reward represented via approximators (e.g., neural networks). For instance we recall:

    - Deep Q-learning: discrete set of actions

    - **Deep Deterministic Policy Gradient: continuous set of actions**

Deep Deterministic Policy Gradient (DDPG)

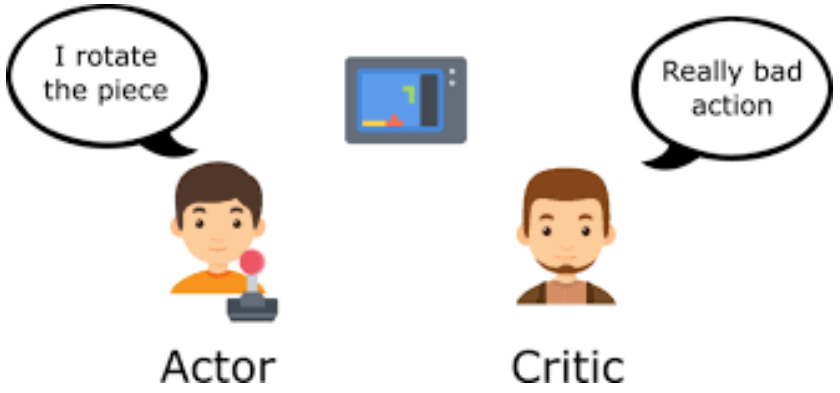
DDPG relies on the **actor-critic paradigm**: the actor learns the parametrized policy and the critic learns the parametrized Q-function.

Lillicrap et al., 2015

The information provided by the critic is used as a reinforcing signal for the actor.

Features:

- Deep neural networks
- Random sampling from replay buffer
- Target networks for stable learning



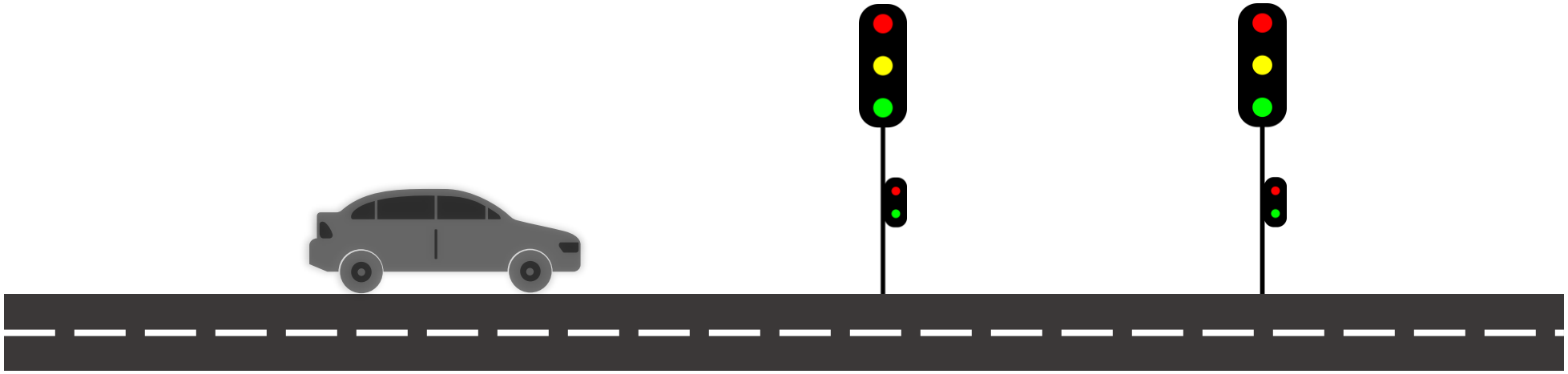
Agenda

- Introduction
- Eco-Driving Task
- Overview on Reinforcement Learning
- **Problem Formulation**
- Simulative Results

Task and Settings

Task: **ecological velocity planning** in urban context.

We consider a straight and flat **road with signalized intersections** and **uncertain traffic lights information**. Note that surrounding traffic is neglected.



Reward Function

The reward is designed in order to minimize the battery equivalent fuel consumption, the ICE fuel consumption and the travel time:

$$r(t_k) = -\alpha_1(\Delta G_m(t_k) + \Delta G_e(t_k)) - \alpha_2 \Delta t_k$$

The coefficients α_1 and α_2 are suitably chosen weights with the aim of **regulating the trade-off between the different objectives**.

Episodic or Continuous Task? Just a Matter of Convention

The reward defined above suits perfectly with the concept of **episodic task**:

- the agent will try to minimize the total time required to travel a certain road (and cross a certain number of intersections)

The same task, however, can be formulated also in a **continuous** fashion:

- maximize the travelled distance in a time step

$$r(t_k) = -\alpha_1(\Delta G_m(t_k) + \Delta G_e(t_k)) + \alpha_2 \Delta s_k$$

Uncertainties in the Vehicle Parameters

We assume that **some vehicle parameters are uncertain**: e.g., the air drag coefficients, the rolling resistance, the air density and the frontal area.

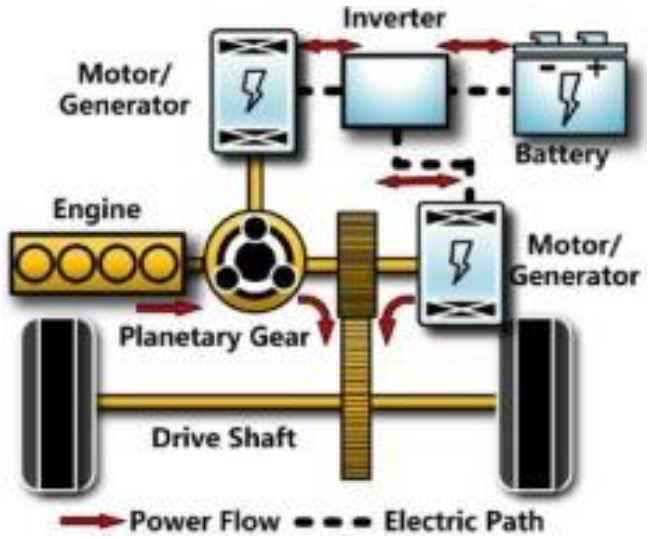
Uncertain parameters can be addressed by considering a **partially observable** RL framework, with the use of past measurements in the state vector.

By designing a training process in which such uncertain parameters are varied for each episode within a range of interest, it is possible to obtain an **agent robust to the considered uncertainties**.

Uncertainties in the Powertrain Parameters

The same approach is used also to overcome the uncertainty in the **powertrain parameters and in the powertrain controller.**

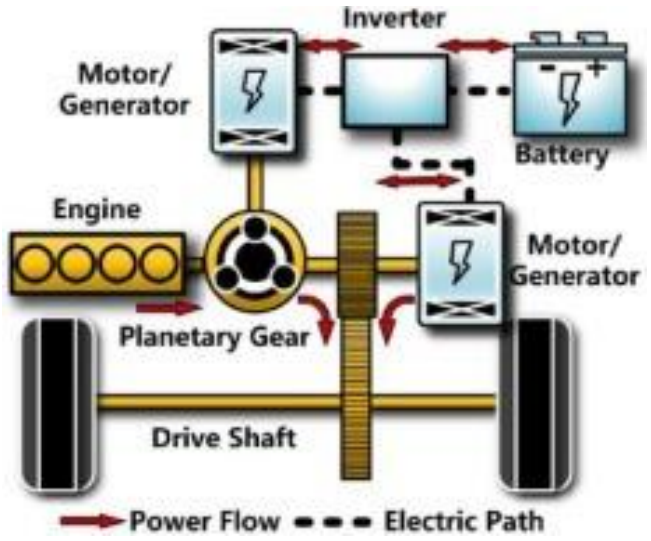
The powertrain controller in fact is usually considered as a black-box which varies from vehicle to vehicle and it is tuned to minimize an energy related cost function.



Availability of Powertrain Measurements

While the powertrain itself is considered a black box, we assume that the battery state of charge, the motor and engine powers, and their energy consumptions are measurable for state feedback control.

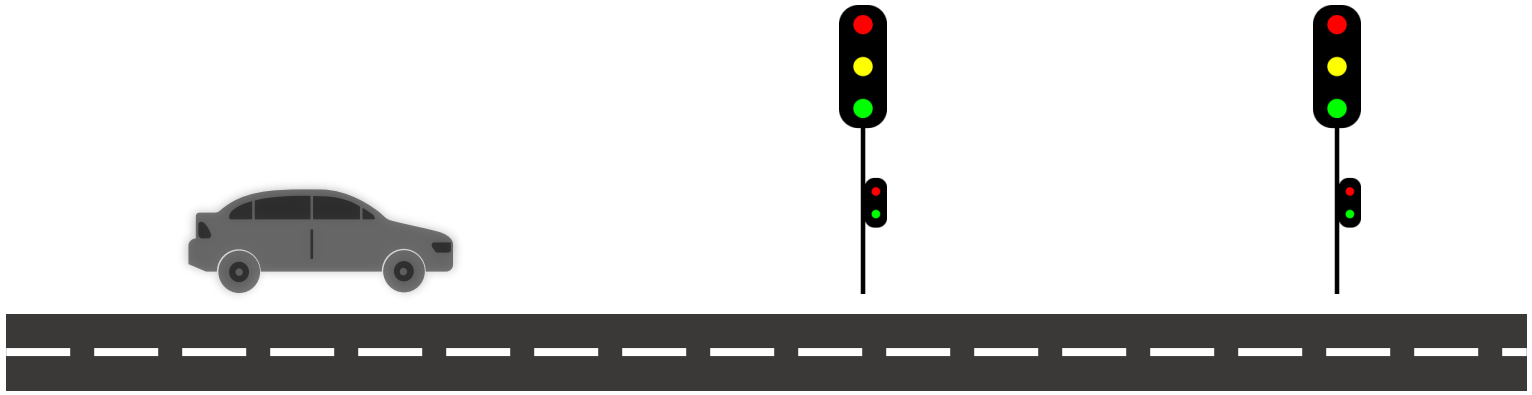
In this work, the **energy consumption measurement is necessary** since this represents the reward signal for the RL agent.



Availability of Signal Phase and Timing Measurements

The optimal velocity profile depends on the distance and the phase of the upcoming intersections.

Such measurements are **available within a specific range**. Outside this range they can only be approximated by historical values.



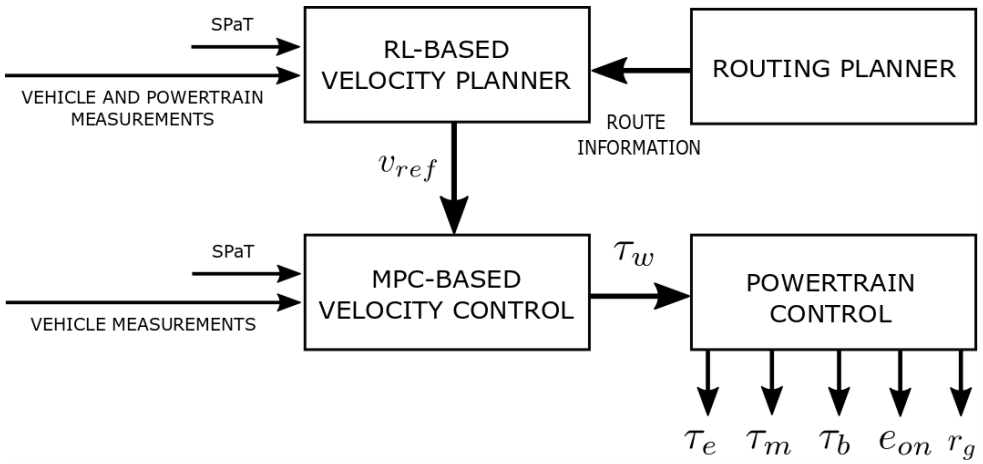
Safety Crossing Module

It finds the optimal wheel torque in order to track the RL reference velocity while guaranteeing **safe crossing (i.e. with green light)** at the intersections.

Position and velocity dynamics:

$$s(t_{k+1}) = s(t_k) + v(t_k)t_s$$

$$v(t_{k+1}) = v(t_k) + a(t_k)t_s$$



with acceleration:

$$a(t_k) = \frac{1}{m}\left(\frac{\tau_w(t_k)}{R_w} - f_{drag}(t_k) - f_{roll}(t_k)\right)$$

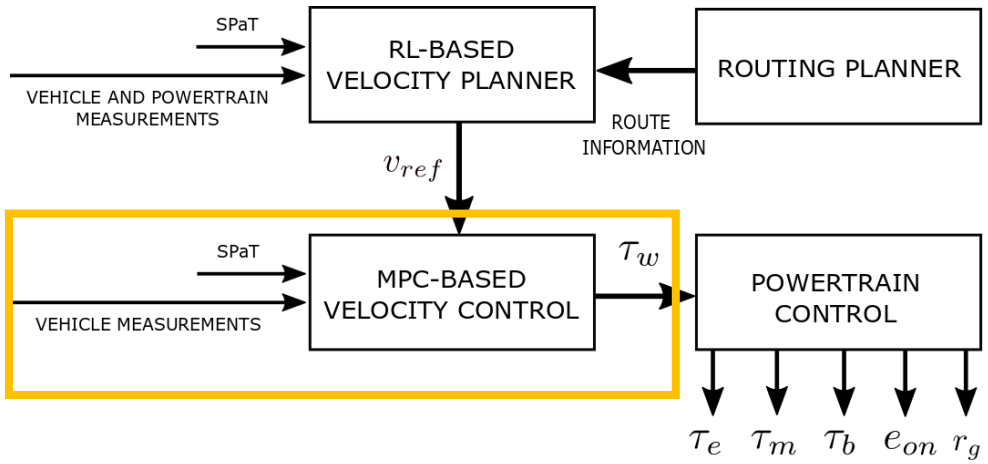
Safety Crossing Module

It finds the optimal wheel torque in order to track the RL reference velocity while guaranteeing **safe crossing (i.e. with green light)** at the intersections.

Position and velocity dynamics:

$$s(t_{k+1}) = s(t_k) + v(t_k)t_s$$

$$v(t_{k+1}) = v(t_k) + a(t_k)t_s$$



with acceleration:

$$a(t_k) = \frac{1}{m}\left(\frac{\tau_w(t_k)}{R_w} - f_{drag}(t_k) - f_{roll}(t_k)\right)$$

Agenda

- Introduction
- Eco-Driving Task
- Overview on Reinforcement Learning
- Problem Formulation
- **Simulative Results**

Benchmark Algorithms

The RL approach, is compared with two benchmark strategies:

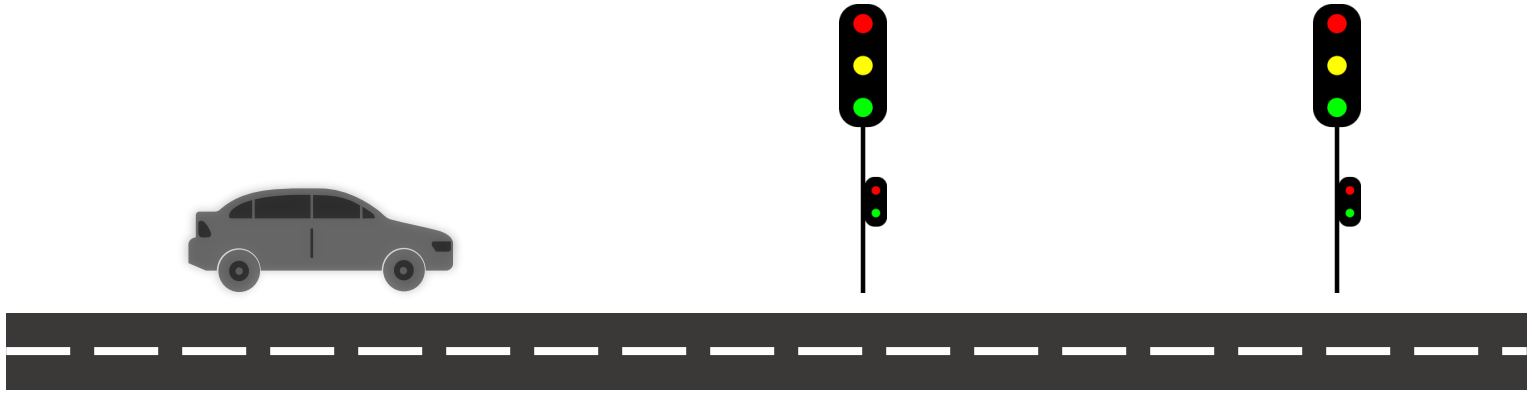
1. A **constant reference velocity**, regardless of the measurements

2. A **variable reference velocity**, according to the signal phase and timing information of the traffic lights for the upcoming intersection.

Training Process

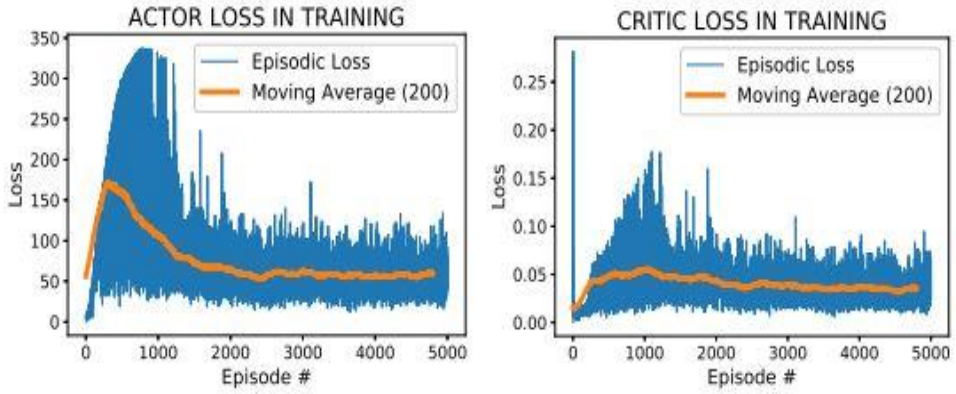
During the training process, the traffic lights configuration, as well as the vehicle and the powertrain parameters **are randomly initialized** among the different episodes.

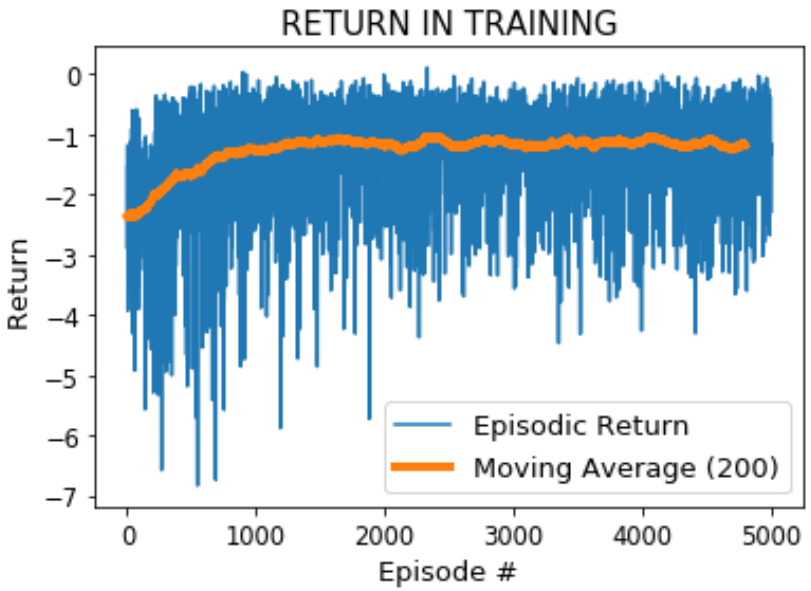
In this way, **the agent learns to find the optimal velocity in a generic scenario**, and not only for a specific configuration of the intersections.



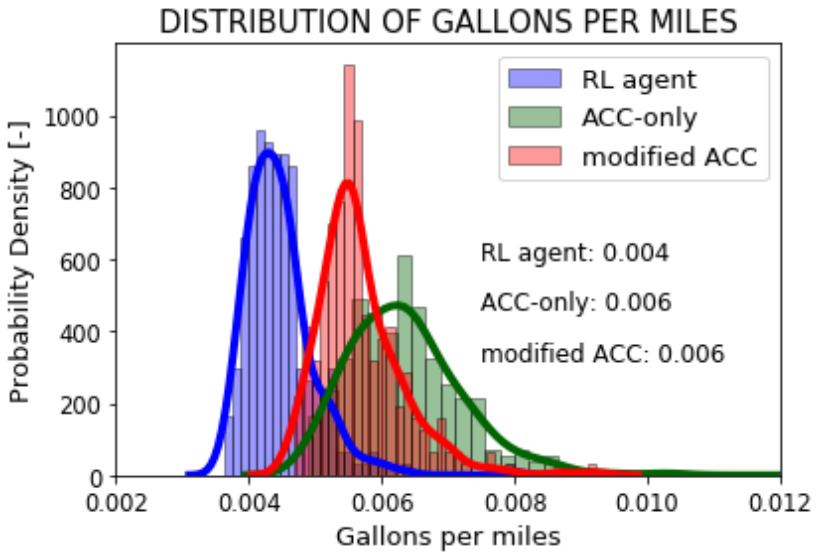
Training Results

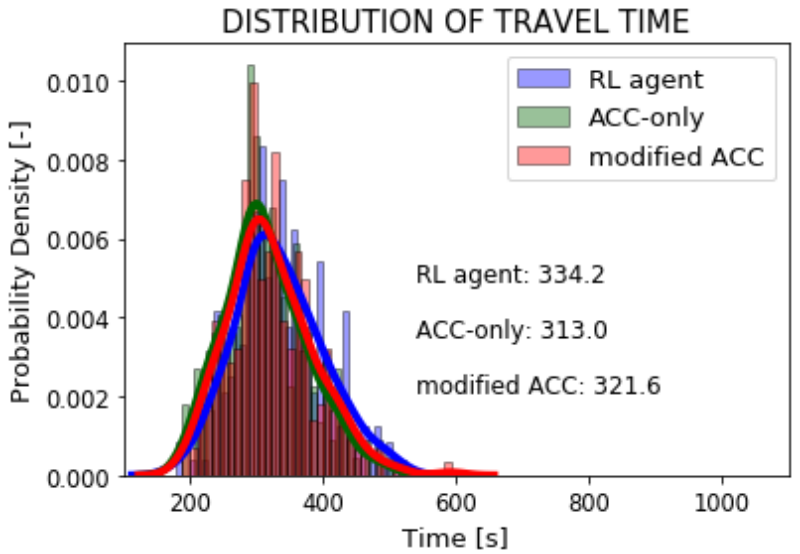
The training can be considered completed after **1500 episodes**.





Statistical Comparison





Statistical Comparison

The **RL-based return is the highest** over 250 different scenarios:

RETURN	Charge Depleting		Charge Sustaining	
	Mean	Std	Mean	Std
RL-agent	−3.199	0.396	−5.521	0.877
ACC-only	−3.653	0.546	−6.559	1.002
modified ACC	−3.492	0.457	−6.125	0.904

Statistical Comparison

The **RL-based return is the highest** over 250 different scenarios:

RETURN	Charge Depleting		Charge Sustaining	
	Mean	Std	Mean	Std
RL-agent	−3.199	0.396	−5.521	0.877
ACC-only	−3.653	0.546	−6.559	1.002
modified ACC	−3.492	0.457	−6.125	0.904

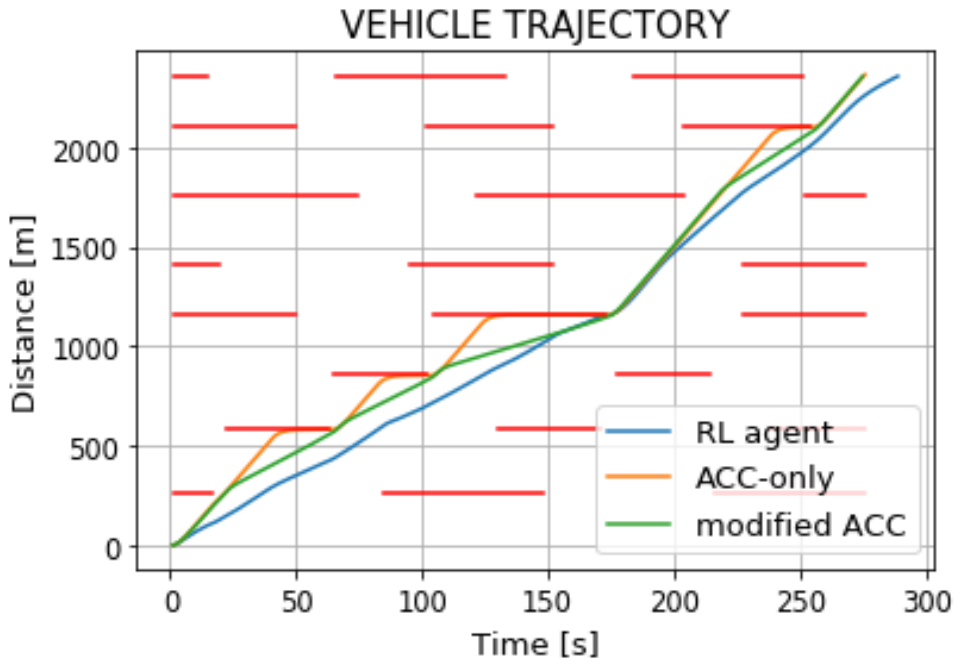
Conclusions

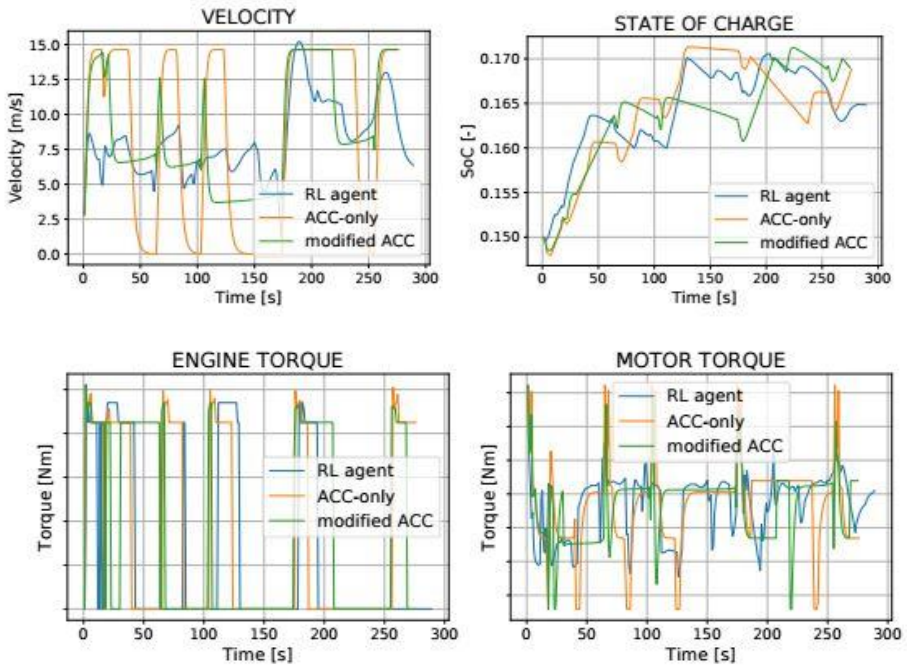
- ✓ We use **model-free reinforcement learning** to solve the ecological velocity planning problem in an **urban context** with traffic lights
- ✓ A **model predictive controller** is employed **for ensuring safety** crossing at the intersections
- ✓ The training process and the state variables are designed in order to develop **an agent which is robust** to variation in vehicle and powertrain parameters
- ✓ **Lower fuel consumption can be achieved in the same travel time** with respect to benchmark methods.

Thank you very much for your attention!!

Suggestions, questions and advices are welcomed!

Specific Scenario Validation





The **trajectory of the RL is much smoother**, thus avoiding useless idle time and reducing consumptions.

Deep Deterministic Policy Gradient (DDPG)

Algorithm 1 DDPG algorithm

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in R
    Sample a random minibatch of N transitions (s_t, a_t, r_t, s_{t+1}) from R
    Set $y_t = r_t + \gamma Q'(s_{t+1}, \mu'(s_{t+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_t (y_t - Q(s_t, a_t|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_t \nabla_a Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_t}$$

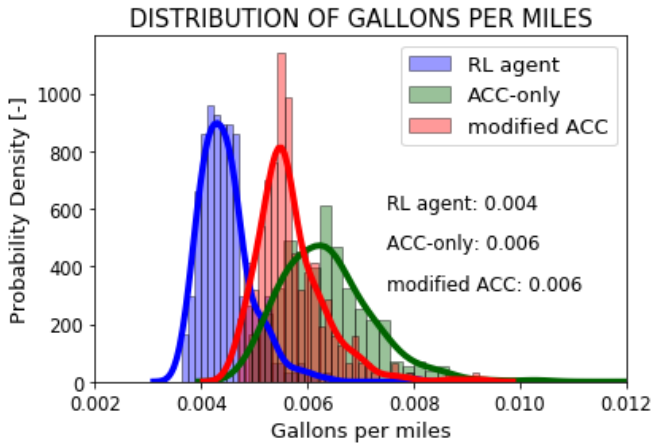
    Update the target networks:

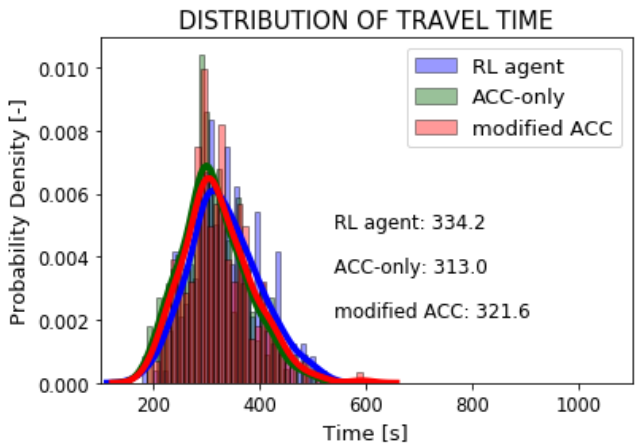
$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

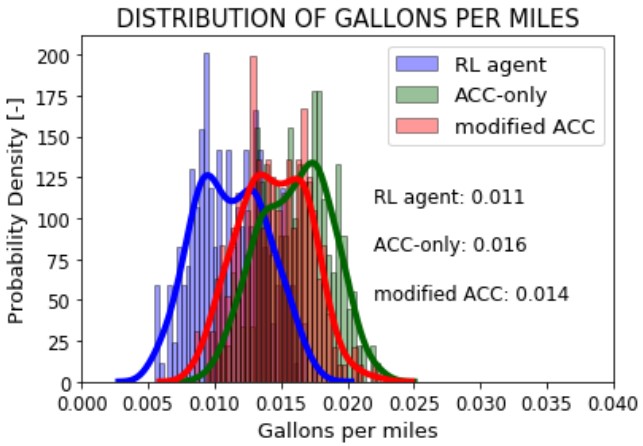
Statistical Comparison

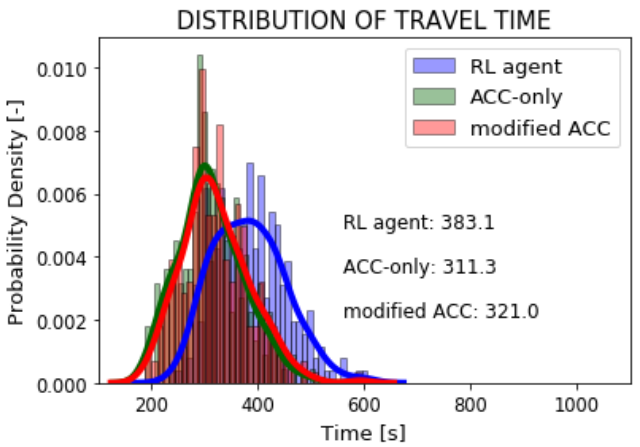
Charge Deplating





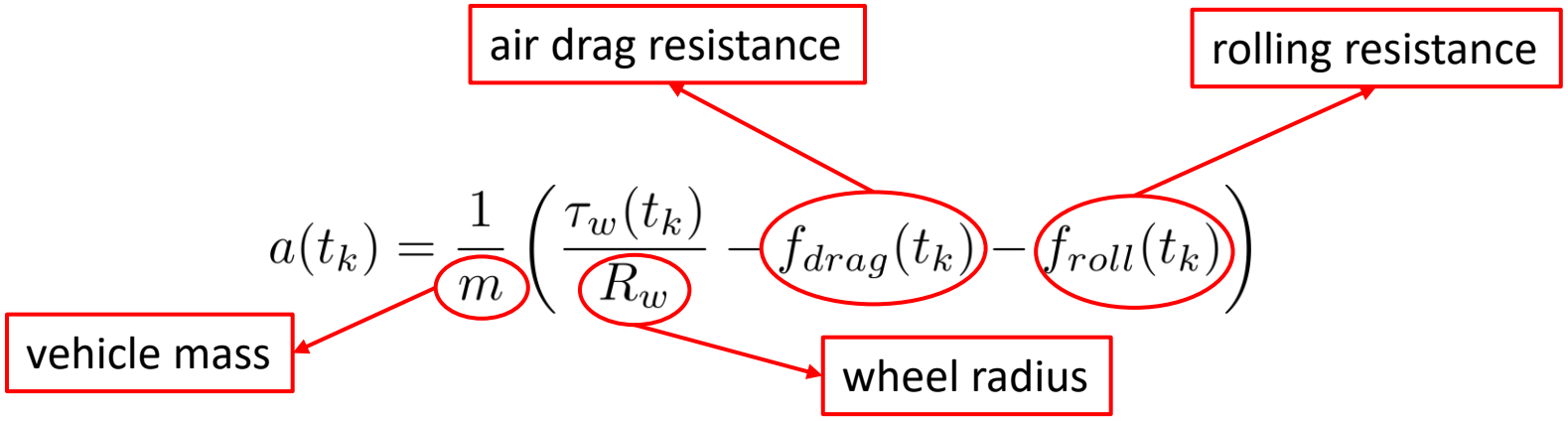
Charge Sustaining





Safety Crossing Module

The terms in the acceleration are the following



Safety Crossing Module

The velocity controller **prevents the vehicle from standing in the middle of the intersections** during phase transitions.

$$
\begin{aligned}
0 \leq d_{tl}(t_k) \quad & \text{if } p^{up}(t_k) = red \\
0 \leq d_{tl}(t_k) + \phi(t) \quad & \text{if } p^{up}(t_k) = yellow
\end{aligned}
$$

with constraints on the distance on the upcoming traffic light according to its phase. We assume that the yellow light phase is long enough for the vehicle to come to a full stop with the maximum deceleration.