End-to-End driving from vision with Deep Imitation Learning and Deep Reinforcement Learning

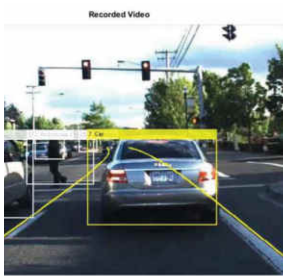
Marin TOROMANOFF[1,2], Emilie WIRBEL[1] & Pr. Fabien MOUTARDE[2]

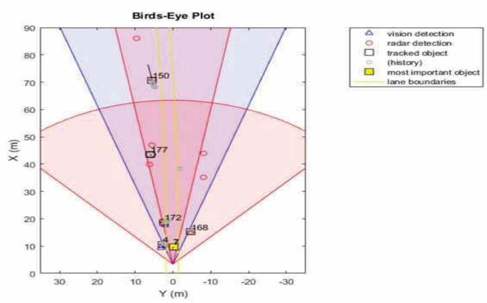
[1]Valeo Driving Assistance
[1]Center for Robotics, MINES ParisTech, PSL Université

Idea of end-to-end driving

Sensors (cameras, radar, LIDAR, etc…) → Perception → Trajectory Planning → Control → Driving wheel, Acceleration or braking

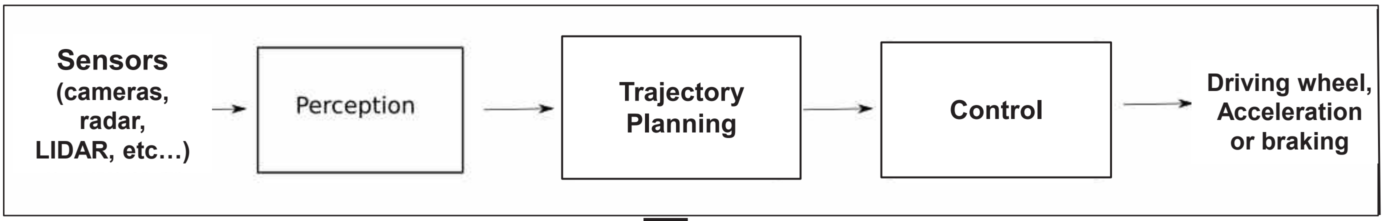
Current architecture for automated driving

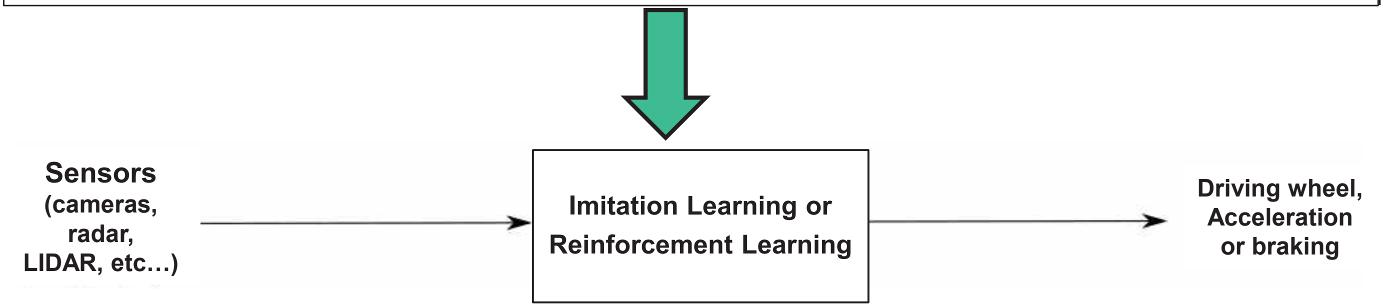


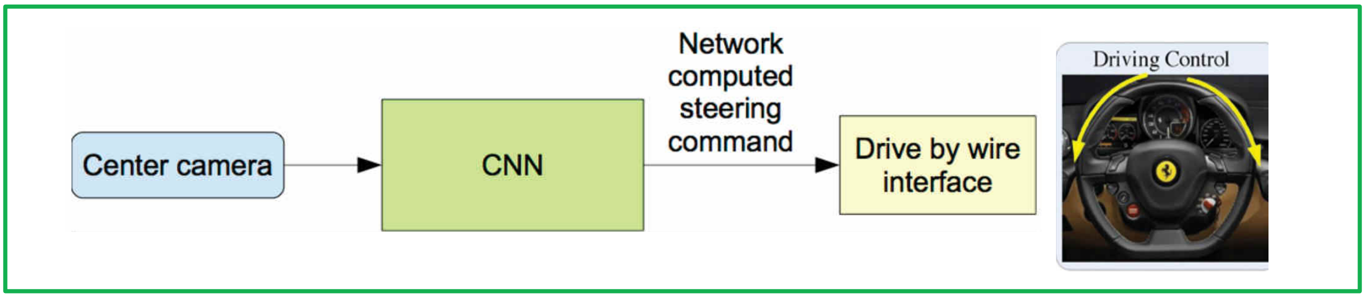


vs. HUMAN driving: turn/accelerate-brake by just looking in front (*“intelligent” <u>visual servoing</u>*)

Principle of end-to-end driving

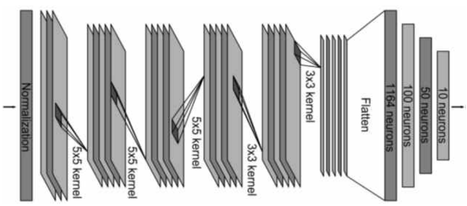




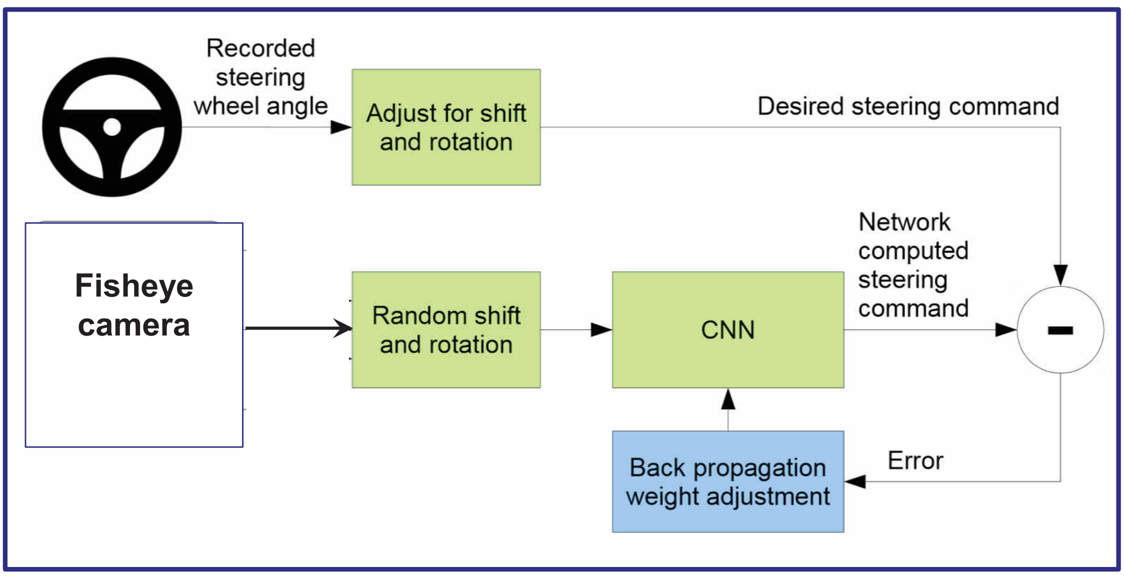


Imitation Learning: "copying" human driver

ConvNet input: Cylindrical projection of fisheye camera



ConvNet output: steering angle



Real human driving data

Training+testing dataset = <u>10,000 km</u> (<u>200 hours</u>) of human driving in openroad (highways, urban streets, country roads, etc…) under various weather conditions

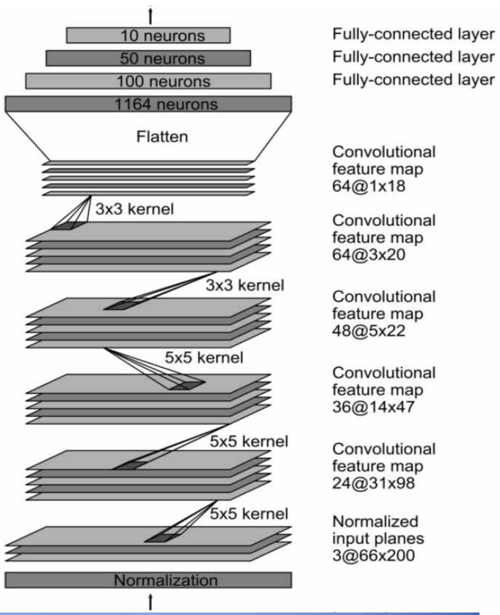
TrainingSet = videos with 10 millions images + driving-wheel angle

TestSet = videos with 3 millions images + driving-wheel angle

Scenario	Urban	Highways	Sharp turns
Image count	100000	70000	15000
Duration (min)	56	39	8

End-to-end trained convNet

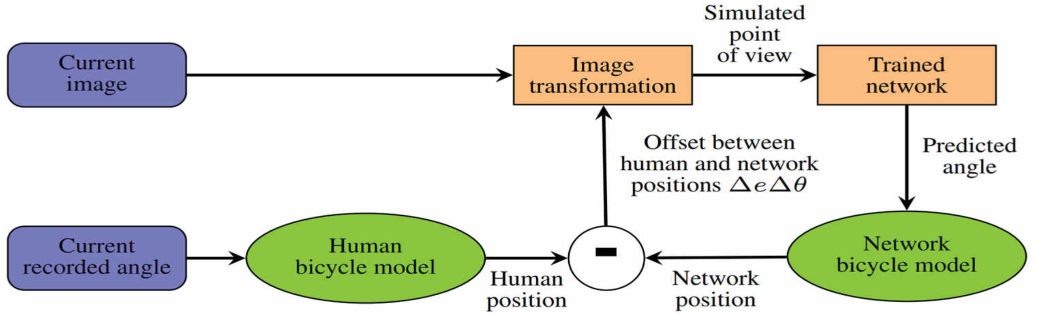
ConvNet <u>output</u>: steering angle



Same ConvNet architecture as in Nvidia work

ConvNet <u>input</u>: Cylindrical projection of fisheye camera

Closed-loop "simulator" *with real images*



"ConvNet in-the-loop" with real images

End-to-end driving: closed loop evaluation

TABLE V: Autonomy (%) and mean absolute distance (MAD, in cm) according to data distribution and validation scenario, the baseline is just going straight.

Scenario	Urban		Highways		Sharp turns	
Metric	Aut. (%)	MAD (cm)	Aut. (%)	MAD (cm)	Aut. (%)	MAD (cm)
Original	99.3	16	**98.7**	**19**	**73.7**	**30**
Sel. #1	98.9	**15**	97.7	25	83.7	**27**
Sel. #2	**99.5**	16	97.2	24	**87.5**	28
Oversamp.	**98**	**18**	**91.8**	**29**	82.5	29
Baseline	8	36	14	41	0	35

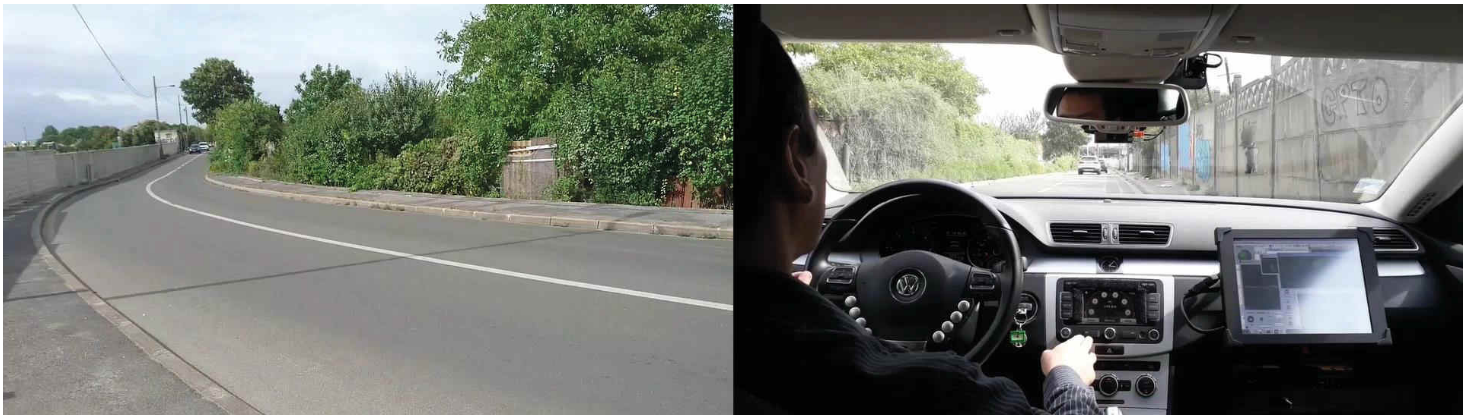
TABLE VI: Comparison of performance between individual networks and bagging

Scenario	Urban		Highways		Sharp turn	
Metric	Aut. (%)	MAD (cm)	Aut. (%)	MAD (cm)	Aut. (%)	MAD (cm)
Weights #1	**99.5**	16	97.2	24	**87.5**	28
Weights #2	98.9	15	97.7	25	83.7	**27**
Weights #3	99.3	16	**98.7**	**19**	**73.7**	**30**
Weights #4	98.6	**18**	**92**	**26**	85	29
Weights #5	**98.4**	15	96.4	21	83.7	28
Bagging	**99.5**	**13**	**98.7**	**19**	**87.5**	**27**

End-to-end driving learnt on <u>*real*</u> data & tested in GTA <u>simulator</u>

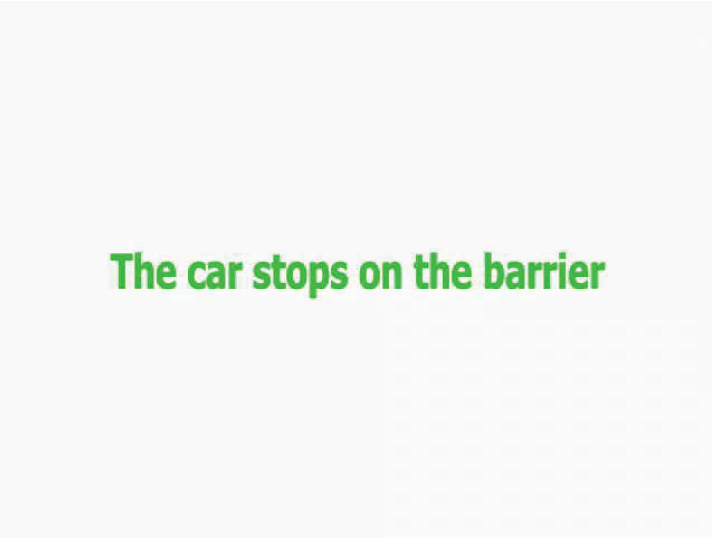
Very good behavior in simulator of convNet learnt ONLY ON REAL DATA → Generality/transferability of driving model

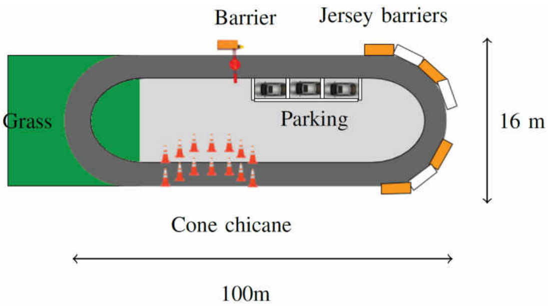
Real vehicle end-to-end driving (learnt by Imitation)



[Work by Valeo using ConvNet trained by my CIFRE PhD student Marin Toromanoff]

Demonstration at CES'2018





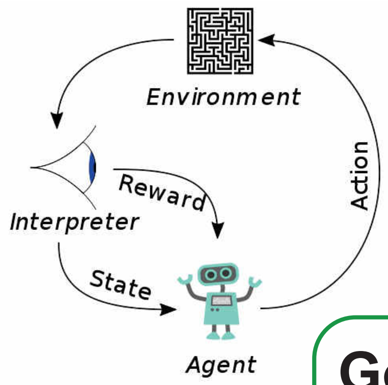
"End to End Vehicle Lateral Control Using a Single Fisheye Camera", Marin Toromanoff, Emilie Wirbel, Frédéric Wilhelm, Camilo Vejarano, Xavier Perrotton et Fabien Moutarde, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018), Madrid, Spain, 1-5 oct. 2018.

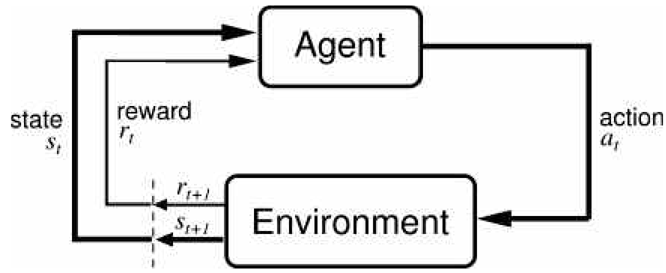
Then, why research also on Reinforcement Learning?

- Intermediate conclusion: Imitation Learning works well, but requires lots of real human driving data in most possible driving situations

- Very difficult to collect real driving data covering enough variety of driving configuration
  ➔ need to use driving simulations
  ➔ need to learn without human ground truth

Reinforcement Learning (RL)





Goal: find a "policy" $a_t=\pi(s_t)$ that

Maximizes $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}, \gamma \in [0,1[$

<u>Deep</u> Reinforcement Learning (<u>DRL</u>) if Deep NeuralNet used as model (for policy and/or its "value"): DQN, Actor-Critic A3C

End-to-end driving: policy π searched as ConvNet(front-image)

3 families of RL algorithms

- **Policy-based** $\pi_\theta \approx \pi^*$
  optimize a parameterized policy
- **Value-based** $Q(s,a,\theta) \simeq Q^{\pi^*}(s,a)$
  find the optimal (parameterized) Q-value

Model-free

- **Model-based**
  $m(s_t, a_t, \theta') \approx s_{t+1}, r_{t+1}$

Value and Q-function

- **Value of a policy (from a given state)**

$$V_\pi(s) = \mathbb{E}_\pi[R_t | s_t = s] = \mathbb{E}_\pi[\sum_{k=0}^{T} \gamma^t r_{t+k} | s_t = s]$$

- **Q-function of a policy**

$$Q_\pi(s,a) = \mathbb{E}_\pi[R_t | s_t = s, a_t = a] = \mathbb{E}_\pi[\sum_{k=0}^{T} \gamma^t r_{t+k} | s_t = s, a_t = a]$$

THERE ALWAYS EXISTS A *DETERMINISTIC* OPTIMAL POLICY π^*

$$\forall \pi, \forall s \in S, V_{\pi^*}(s) \geqslant V_\pi(s)$$

Deep Q Network (DQN)

- **Q-learning:** $Q^{new}(s_t,a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t,a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1},a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$

- **Optimal policy deduced from optimal Q-value**

$$\pi^*(s) = arg \max_a Q_{\pi^*}(s,a)$$

- **DQN [1]: if too many possible states, approximate Q as a neural network, and learn Q* using SGD with loss from Bellman equation**

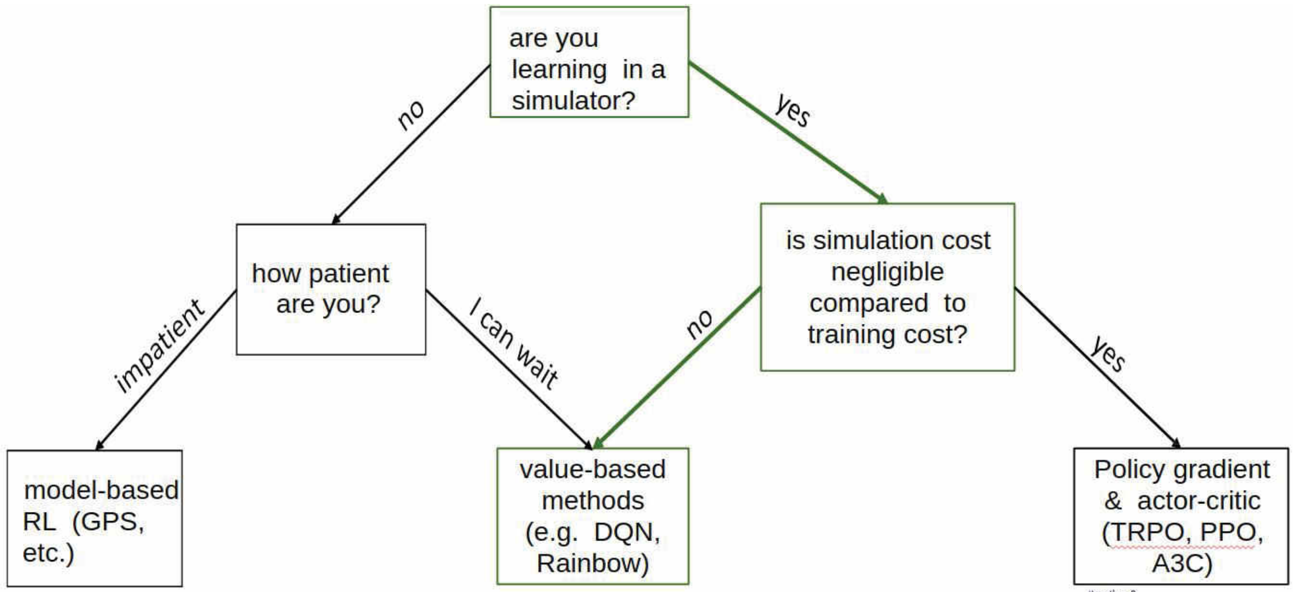
$$L(s_t, a_t, r_{t+1}, s_{t+1}, \theta) = (\underbrace{r_{t+1} + \gamma \max_a Q(s_{t+1}, a, \theta)}_{target} - Q(s_t, a_t, \theta))^2$$

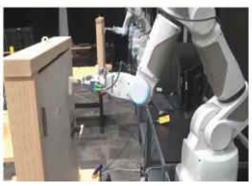
[1] V. Mnih et al : Human-level control through deep reinforcement learning (2015)

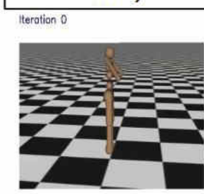
Summary of main RL algorithm types

Family	Algorithm	On/Off policy	Discrete/continuous output?
Policy based	REINFORCE	On policy	Both
	Actor-Critic (A3C)	On policy	Both
Value based	SARSA	On policy	Discrete
	Q-Learning (DQN)	Off-policy	Discrete
Model based	MCTS	Off-policy	Discrete
	iLQG	Off-policy	Continuous

How to choose RL algorithm?







[1] S. Levine: CS294 *Deep Reinforcement Learning* (2017)

RL for automated driving

- **Until recently, very few published research, and mostly in racing games:**

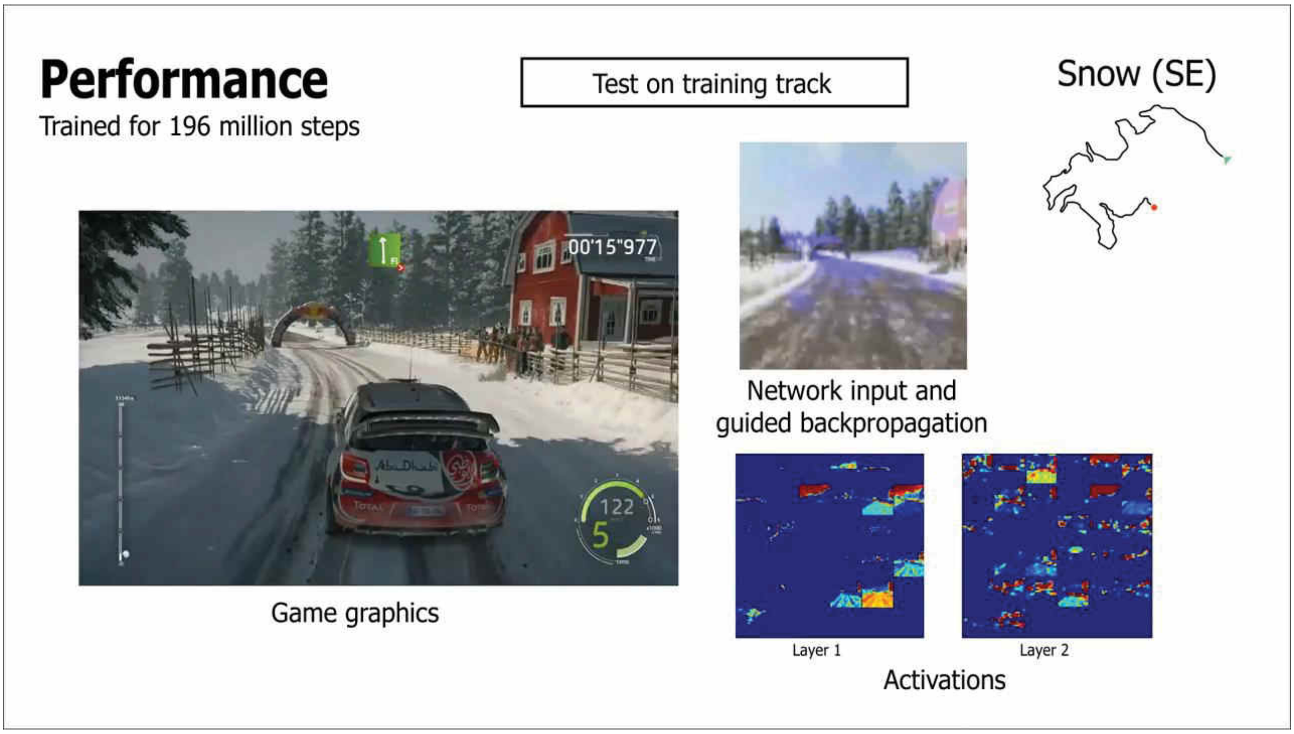
Asynchronous methods for deep reinforcement learning, V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, ICML'2016.

End-to-End Race Driving with Deep Reinforcement Learning, **Maximilian Jaritz, Raoul De Charette, Marin Toromanoff, Etienne Perot, Fawzi Nashashibi, *ICRA 2018 - IEEE International Conference on Robotics and Automation*, Brisbane, Australia, May 2018.**

- **Up to now, only real driving with RL:**
  "*Learning to Drive in a Day*" (2018, [1])
  - **Embed DRL in a real car, and learn « *from scratch* »**
  - **But <u>*VERY SIMPLE CASE: lane keeping along 250m!*</u>**
  - **Simulation used before to design architecture and find hyper-parameters**

[1] A. Kendall et al.: Learning to Drive in a Day (2018)

End-to-end driving learnt by RL in racing-cars simulator



End-to-End Race Driving with Deep Reinforcement Learning, **Maximilian Jaritz, Raoul De Charette, Marin Toromanoff, Etienne Perot, Fawzi Nashashibi, *ICRA 2018 - IEEE International Conference on Robotics and Automation*, Brisbane, Australia, May 2018.**

RL for Automated Driving: why learn in a simulator?

- RL require huge amount of trial & error, and initial policy = very bad driving!
  ⇒ *Learn in <u>simulation</u>* (for safety + speed)
- Still few driving simulators adapted for DL and RL, and best ones not totally mature

Simulateur	GTA	DeepDrive.io	AirSim	CARLA[1]
Flexibilité	−−	++	++	++
Variété	++	−−	−	+
Complexité/Réalisme	++	−−	−	−
Objets mobiles	++	−−	−−	+
Vitesse exécution	−−	+	+	+
Multi-agent	−−	−	−	++

→ Choice of CARLA

[1] A. Dosovitskiy: CARLA: An Open Urban Driving Simulator (2017)

CARLA benchmarks

- First benchmark: navigation tasks with 4 possible orders at intersections (Left, Straight, Right, Follow_Lane) and 4 difficulty levels:
  - Straight (never turn at intersection)
  - One Turn
  - Navigation = longer path with at least 2 orders
  - Dynamic Navigation = + pedestrians and cars

  But metrics = only %arrival at destination!
- <u>CARLA challenge</u> (current): idem + respecting traffic lights, and handling lane-change
  - Evaluation metrics = Task completion & Distance between infractions
  - Final test on *unseen* city, and several unseen weathers!
  - Results (and 10.000$ for winner!) at CVPR'2019 (June 16-17)

RL used: Rainbow + IQN + ApeX

- **Rainbow** [1] **= combination of many improvements of DQN** [4] **➔ currently SoA on ATARI benchmark**
- **IQN** [2] **= learning with *<u>probability distributions</u>* rather than just expectation of average**

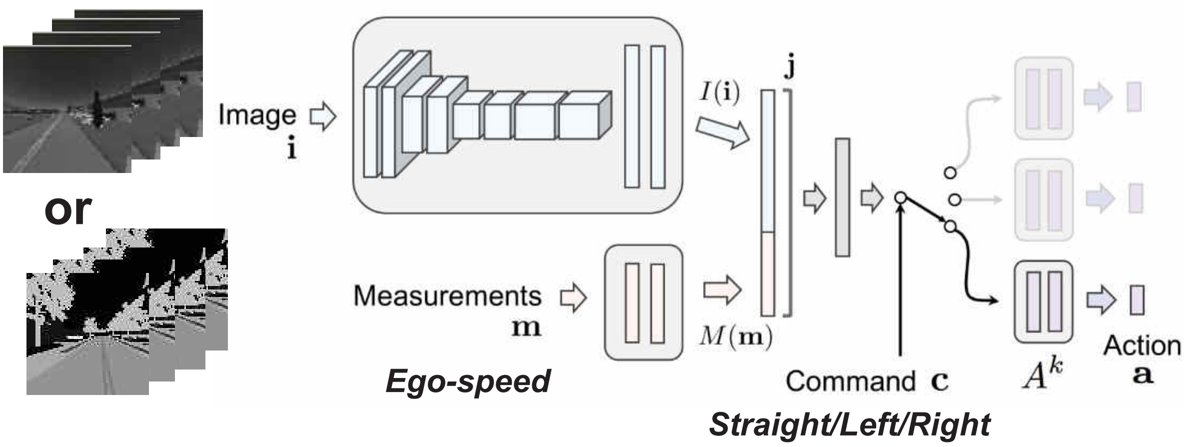
	Mean	Median	Human Gap	Seeds
DQN	228%	79%	0.334	1
PRIOR.	434%	124%	0.178	1
C51	701%	178%	0.152	1
RAINBOW	**1189%**	**230%**	0.144	2
QR-DQN	864%	193%	0.165	3
IQN	1019%	218%	**0.141**	5

- **Ape-X** [3] ***<u>multi-agent</u>* version of DQN allowing massively parallel distributed learning**
  ⇒ Largely better performance, but typically require 22 billions of frames (vs. 200 millions)

[1] M. Hessel et al : Rainbow: Combining Improvements in Deep Reinforcement Learning Matteo (2017)
[2] D. Silver et al : Implicit Quantile Networks for Distributional Reinforcement Learning (2018)
[3] B. Horgan et al : Distributed Prioritized Experience Replay (2018)
[4] V. Mnih et al : Human-level control through deep reinforcement learning (2015)

Learning on CARLA first benchmark

- **Multi-head architecture for high-level navigation goal (straight / left-or-right turn at intersections)**



Steering angle (5 values)
Acceleration (3 values)

- **Negative reward = f(distance from center-of-lane) + positive reward = g(speed – recommanded_value) [36 km/h in our initial tests]**
- **End of episod if collision or too far from lane-center**

Preliminary tests on first CARLA benchmark

TASK	*Baseline RL (Train Town)*	Rainbow-IQN (Train Town)	*Baseline RL (Test Town)*	Rainbow-IQN segmentation (Test Town)
Straight	*89%*	88%	*74%*	96%
One Turn	*34%*	80%	*12%*	76%
Navigation	*14%*	68%	*3%*	52%
Dynamic Nav.	*7%*	52%	*2%*	44%

Rainbow-IQN > Rainbow >>> Baseline RL CARLA (but metrics = only %arrival at destination!)

Example of result for "Go Straight"

Example of test result with turns

Conclusions & perspectives

- End-to-end driving by *Imitation-Learning* can work and generalize rather well, but requires human driving data of all representative situations!!
- Deep Reinforcement Learning in CARLA simulator for End-to-end driving → Very encouraging preliminary results
  - Potential improvement of driving smoothness by increasing # of discretization levels for actions
  - Currently in progress: participation to CARLA challenge → handling of Traffic Lights (etc…) *using a pre-learnt input transformation* (instead of raw images)
- Future work:
  - transferrability to real-world videos
  - Combination of Imitation-Learning and RL?

QUESTIONS?

[very first test on new CARLA challenge]