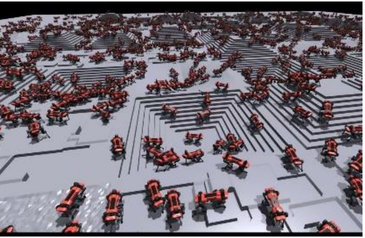
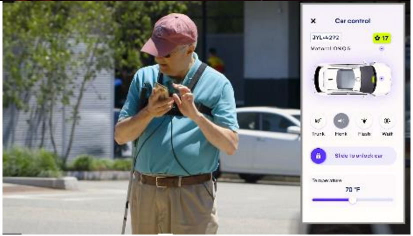
# EC500: Robot Learning and Vision for Navigation



Eshed Ohn-Bar


Feb 6, 2023



# Reminders

- Homework are expected to be individual assignments

- Project can be done individually or in groups of two (expectation are in relative to the group size)

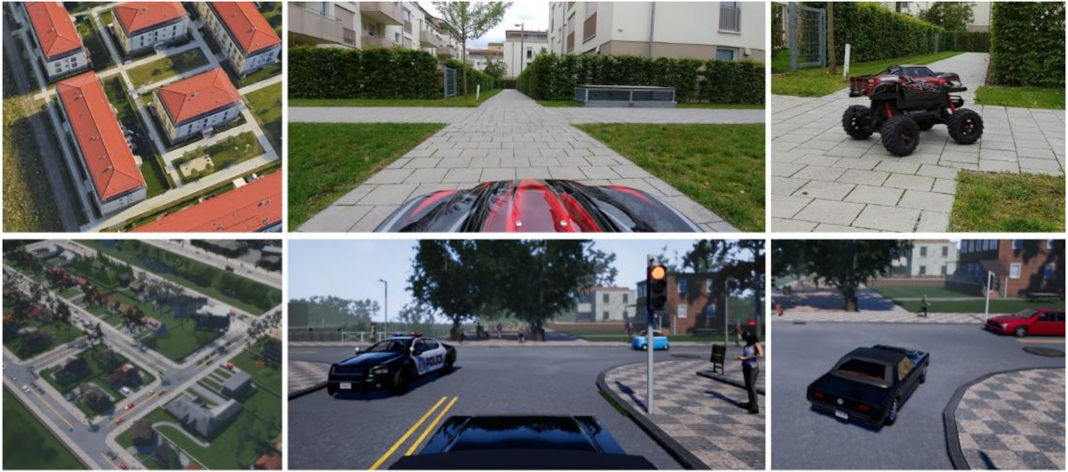
# Previously…

- Paradigms for Sensorimotor Agents

- Behavior Cloning, Covariate Shift, Dagger, Dagger + Sampling

$$\underset{\theta}{\operatorname{argmin}} \underbrace{\mathbb{E}_{(s^*,a^*)\sim P^*}\left[\mathcal{L}\left(a^*, \pi_\theta(s^*)\right)\right]}_{=\sum_{i=1}^{N} \mathcal{L}\left(a_i^*, \pi_\theta(s_i^*)\right)}$$

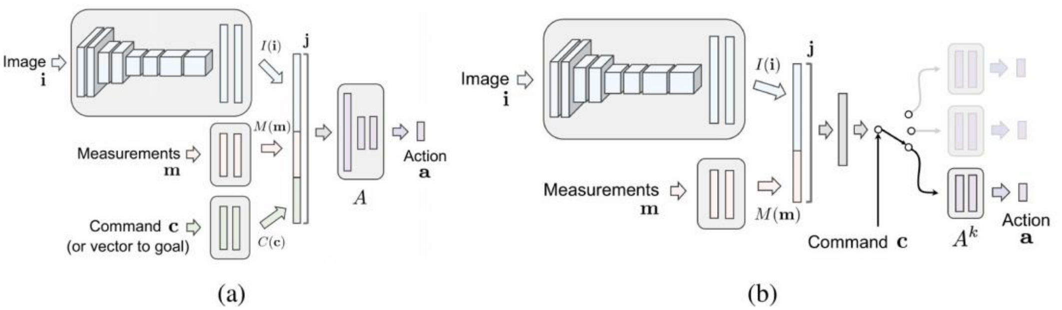
# Conditional Imitation Learning




(a) Aerial view of test environment (b) Vision-based driving, view from onboard camera (c) Side view of vehicle

Codevilla, Müller, López, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

# Conditional Imitation Learning: Network Architecture

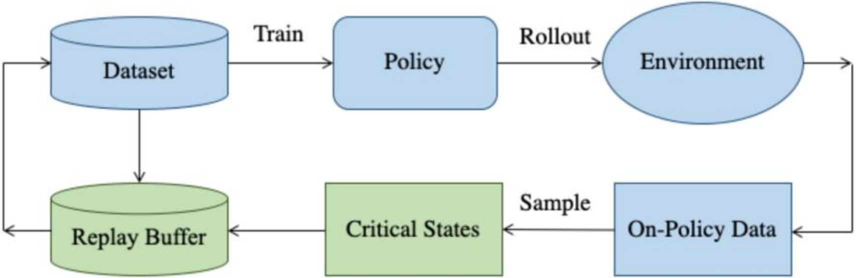



- This paper proposes two network architectures:
  - (a) Extract features $C(c)$ and concatenate with image features $I(\mathbf{i})$
  - (b) Command $c$ acts as switch between specialized submodules
- Measurements $m$ capture additional information (here: speed of vehicle)

Codevilla, Muller, Lopez, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

Dagger is very slow to converge in practice, training+testing (roll-out) iterations are inefficent

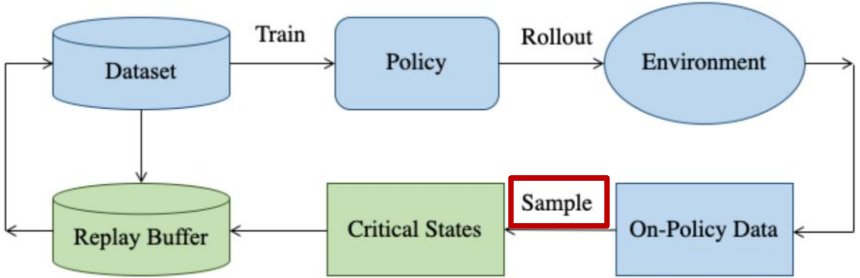
# DAGGER with Critical States and Replay Buffer




**Sampling Strategies:**

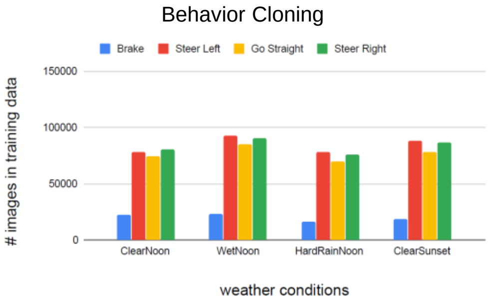
- Task-based: Sample uniformly from “left”, “right”, “straight”
- Policy-based: Use test-time dropout to estimate epistemic uncertainty
- Expert-based: Highest loss or deviation in brake signal wrt. expert

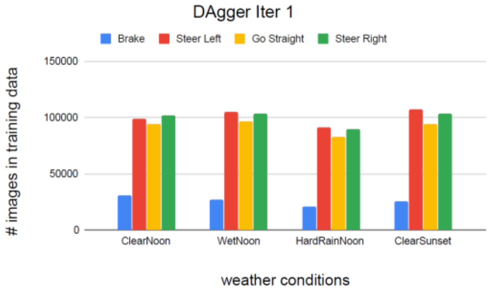
# DAGGER with Critical States and Replay Buffer

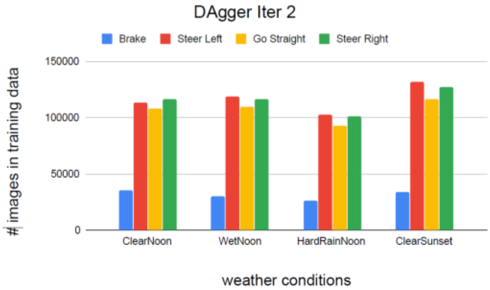


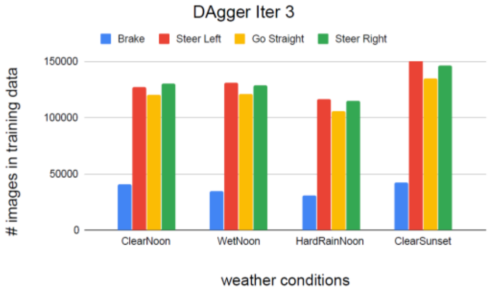

**Sampling Strategies:**

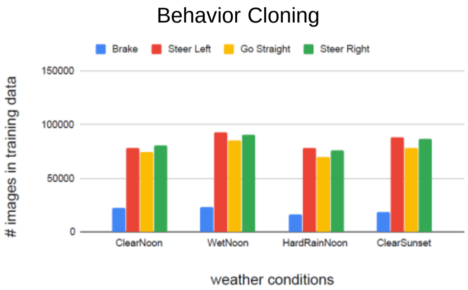
- Task-based: Sample uniformly from “left”, “right”, “straight”
- Policy-based: Use test-time dropout to estimate epistemic uncertainty
- Expert-based: Highest loss or deviation in brake signal wrt. expert

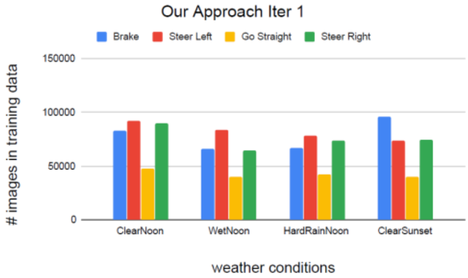

Behavior Cloning
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions

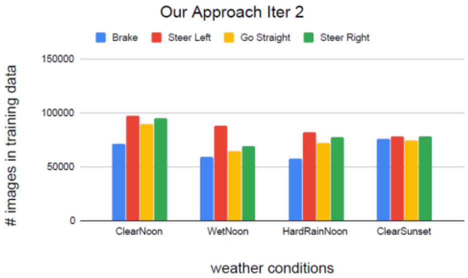

DAgger Iter 1
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions

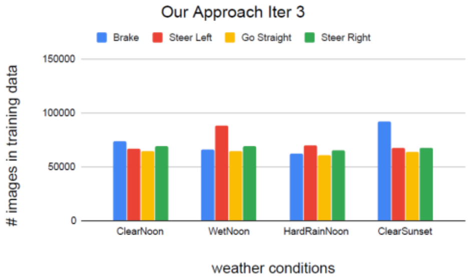

DAgger Iter 2
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


DAgger Iter 3
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions

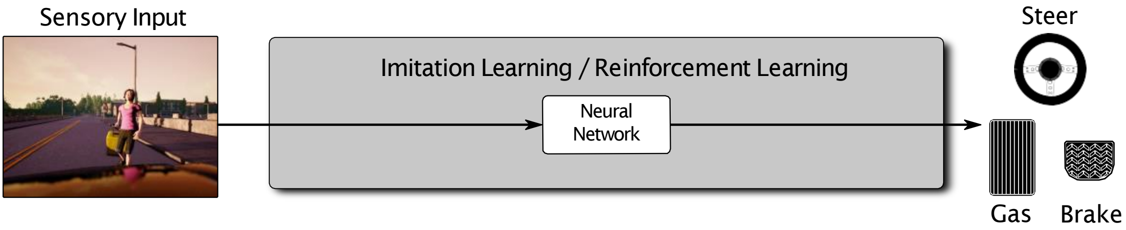

Behavior Cloning
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


Our Approach Iter 1
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


Our Approach Iter 2
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


Our Approach Iter 3
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
# images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions

# Learning Sensorimotor Policies is Difficult

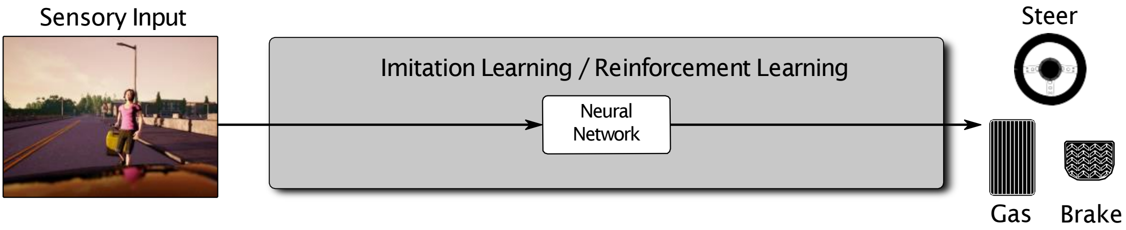



Agent has to learn

- Perception
- Prediction
- Planning
- Control

How can we decompose this problem?

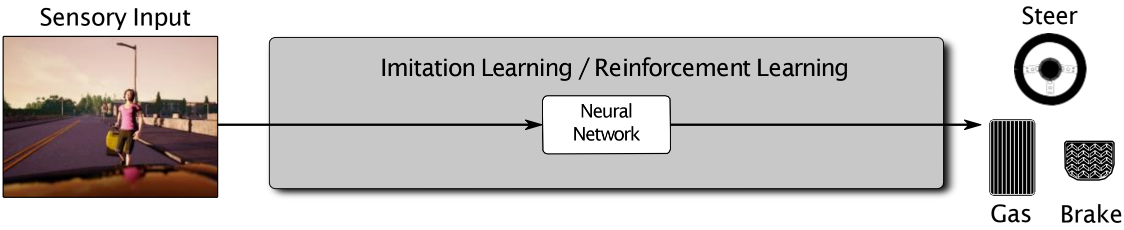
# Learning Sensorimotor Policies is Difficult




Agent has to learn

- Perception
- Prediction
- Planning
- Control

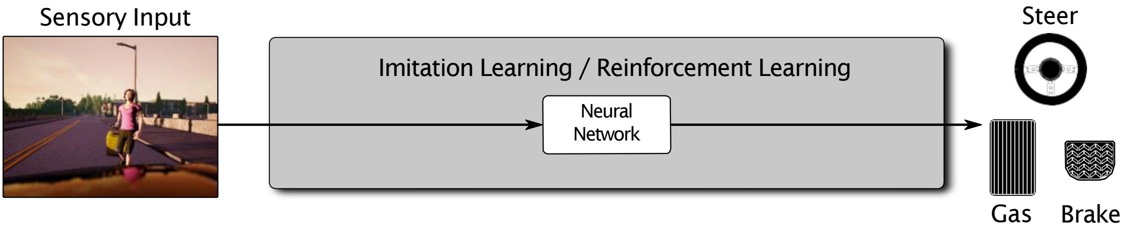
# Learning Sensorimotor Policies is Difficult




Agent has to learn

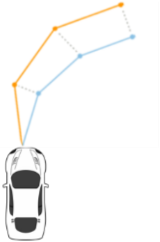
- Perception
- Prediction
- Planning
- Control => PID Controller

# Learning Sensorimotor Policies is Difficult

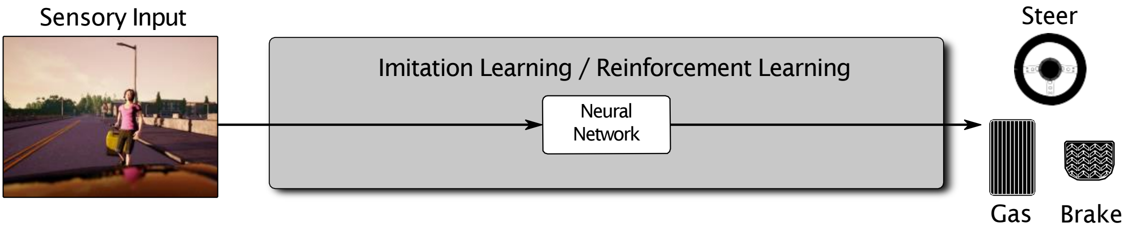



Agent has to learn

- Perception
- Prediction
- Planning => Future waypoints
- Control => PID Controller

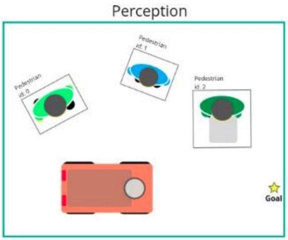


# Learning Sensorimotor Policies is Difficult




Agent has to learn

- Perception=>BEV (Bird's Eye View)
- Prediction
- Planning => Future waypoints
- Control => PID Controller



Learning by Cheating, Chen et al., Conference on Robot Learning, 2019


Trained with imitation learning from human experts

Trained with imitation learning from the privileged agent

But isn’t it the same supervised learning problem? why would learning a model from simplified input help?

But isn’t it the same supervised learning problem? why would learning a model from simplified input help?

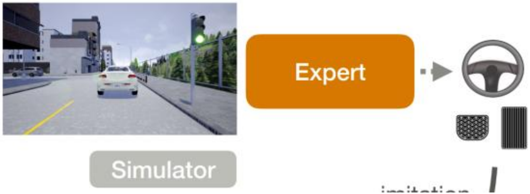
Efficient data augmentation

# Augmentation from the Expert

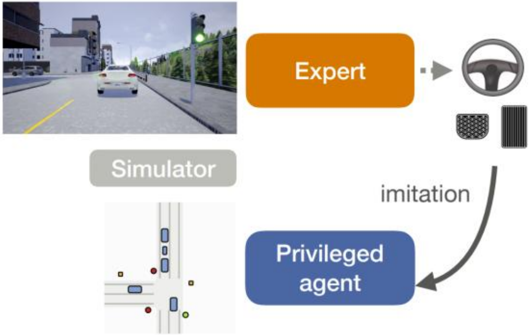
(a) Road map (b) Rotation and shift aug.

Figure 3: **(a)** Map $M$ provided to the privileged agent. One channel each for road (light grey), lane boundaries (grey), vehicles (blue), pedestrians (orange), and traffic lights (green, yellow, and red). The agent is centered at the bottom of the map. The agent's vehicle (dark red) and predicted waypoints (purple) are shown for visualization only and are not provided to the network. **(b)** The map representation affords simple and effective data augmentation via rotation and shifting.

# Learning by Cheating, Chen et al., Conference on Robot Learning, 2019

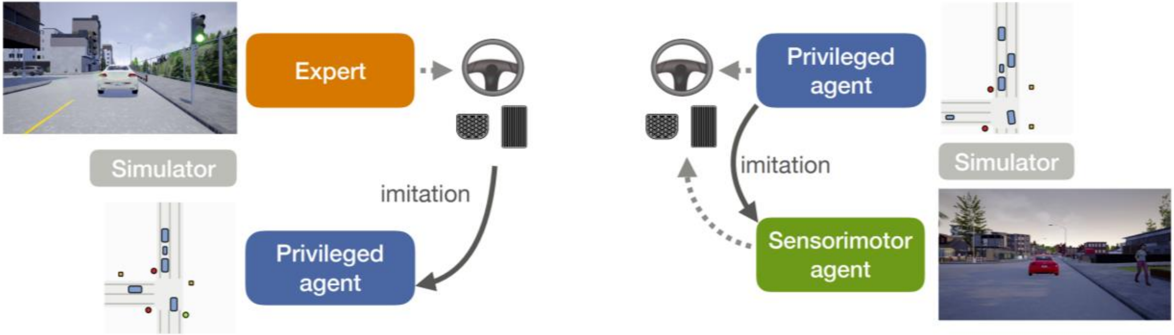


# Learning by Cheating, Chen et al., Conference on Robot Learning, 2019




(a) Privileged agent imitates the expert

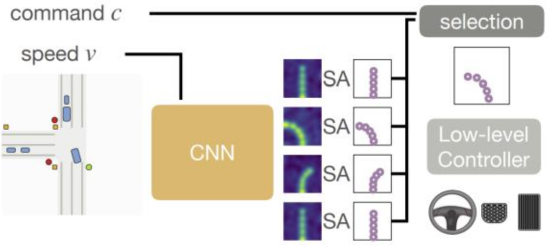
Learning by Cheating, Chen et al., Conference on Robot Learning, 2019




(a) Privileged agent imitates the expert

(b) Sensorimotor agent imitates the privileged agent

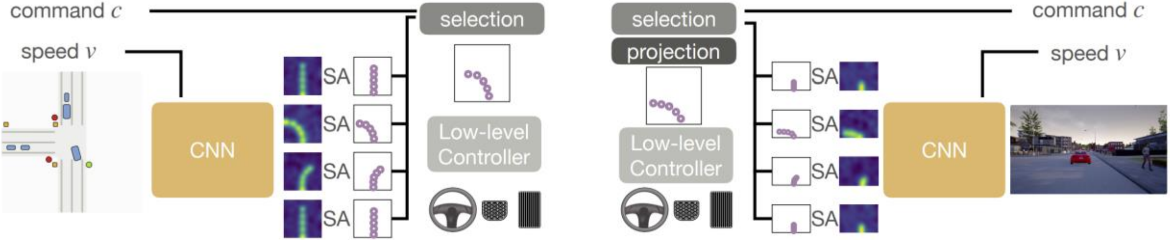
# Learning by Cheating, Chen et al., Conference on Robot Learning, 2019




(a) Privileged agent

$$\text{softargmax}(\boldsymbol{y}) = \sum_i \frac{e^{y_i/T}}{\sum_j e^{y_j/T}} i$$

# Learning by Cheating, Chen et al., Conference on Robot Learning, 2019




(a) Privileged agent (b) Sensorimotor agent

$$\text{softargmax}(\boldsymbol{y}) = \sum_i \frac{e^{y_i/T}}{\sum_j e^{y_j/T}} i$$

# Augmentation from the Expert

| Supervision | white-box | on-policy | |
|---|---|---|---|
| Direct | | | 20 |
| Two stage | | | 16 |
| Two stage | | ✓ | 64 |
| Two stage | ✓ | | 96 |
| Two stage | ✓ | ✓ | 100 |

Table 1: Ablation study on the *CoRL2017* benchmark (CARLA 0.9.5, “navigation” condition, test town, test weather). Two key advantages of the presented decomposition – white-box supervision and on-policy trajectories – each substantially improve performance and together achieve 100% success rate on the benchmark.

# Augmentation from the Expert

| Supervision | white-box | on-policy | |
|---|---|---|---|
| Direct | | | 20 |
| Two stage | | | 16 |
| Two stage | | ✓ | 64 |
| Two stage | ✓ | | 96 |
| Two stage | ✓ | ✓ | 100 |

Table 1: Ablation study on the *CoRL2017* benchmark (CARLA 0.9.5, "navigation" condition, test town, test weather). Two key advantages of the presented decomposition – white-box supervision and on-policy trajectories – each substantially improve performance and together achieve 100% success rate on the benchmark.

# Augmentation from the Expert

| Supervision | white-box | on-policy | |
|---|---|---|---|
| Direct | | | 20 |
| Two stage | | | 16 |
| Two stage | | ✓ | 64 |
| Two stage | ✓ | | 96 |
| Two stage | ✓ | ✓ | 100 |

Table 1: Ablation study on the *CoRL2017* benchmark (CARLA 0.9.5, "navigation" condition, test town, test weather). Two key advantages of the presented decomposition – white-box supervision and on-policy trajectories – each substantially improve performance and together achieve 100% success rate on the benchmark.

# Learning by Cheating

| | | CARLA ≤0.9.5 | | | | CARLA 0.9.6 | | |
|---|---|---|---|---|---|---|---|---|
| Task | Weather | CIL[6] | CAL[20] | CILRS[7] | LBC | LBC | PV | AT |
| Empty | | $48 \pm 3$ | $36 \pm 6$ | $51 \pm 1$ | $\mathbf{100} \pm 0$ | $100 \pm 0$ | $100 \pm 0$ | $100 \pm 0$ |
| Regular | train | $27 \pm 1$ | $26 \pm 2$ | $44 \pm 5$ | $\mathbf{96} \pm 5$ | $94 \pm 3$ | $95 \pm 1$ | $99 \pm 1$ |
| Dense | | $10 \pm 2$ | $9 \pm 1$ | $38 \pm 2$ | $\mathbf{89} \pm 1$ | $51 \pm 3$ | $46 \pm 8$ | $60 \pm 3$ |
| Empty | | $24 \pm 1$ | $25 \pm 3$ | $90 \pm 2$ | $\mathbf{100} \pm 2$ | $70 \pm 0$ | $100 \pm 0$ | $100 \pm 0$ |
| Regular | test | $13 \pm 2$ | $14 \pm 2$ | $87 \pm 5$ | $\mathbf{94} \pm 4$ | $62 \pm 2$ | $93 \pm 2$ | $99 \pm 1$ |
| Dense | | $2 \pm 0$ | $10 \pm 0$ | $67 \pm 2$ | $\mathbf{85} \pm 1$ | $39 \pm 8$ | $45 \pm 10$ | $59 \pm 6$ |

Table 3: Comparison of the success rate of the presented approach (LBC) to the previous approaches on the *NoCrash* benchmark in the test town. (The supplement provides results on the training town.) PV denotes the performance of the privileged agent, AT is the performance of the built-in CARLA autopilot. Since the graphics and simulator behavior changed significantly with CARLA 0.9.6, we evaluate and compare our method on CARLA 0.9.5. CILRS was also run on this version of CARLA. Our approach outperforms prior work by significant factors, achieving 100% success rate in the "Empty" condition and reaching 85% success rate or higher in other conditions.

# Learning by Cheating

Dian Chen, Brady Zhou,
Vladlen Koltun, Philipp Krähenbühl

The University of Texas at Austin
Computer Science

intel

Very difficult to do in the real-world, either Dagger or Learning by Cheating

# ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst

**Mayank Bansal** MAYBAN@WAYMO.COM
*Waymo Research*
*Mountain View, CA 94043, USA*

**Alex Krizhevsky*** AKRIZHEVSKY@GMAIL.COM

**Abhijit Ogale** OGALE@WAYMO.COM
*Waymo Research*
*Mountain View, CA 94043, USA*

Dec 2018

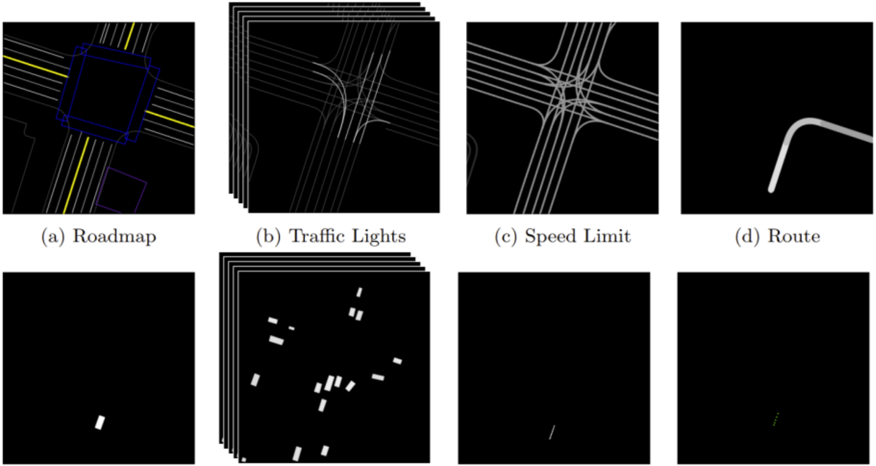
https://arxiv.org/pdf/1812.03079.pdf

- 30 million samples from 60 days of driving

- Structured input from perception, not raw pixels

- Augmentation to handle distribution shift (compounding errors)

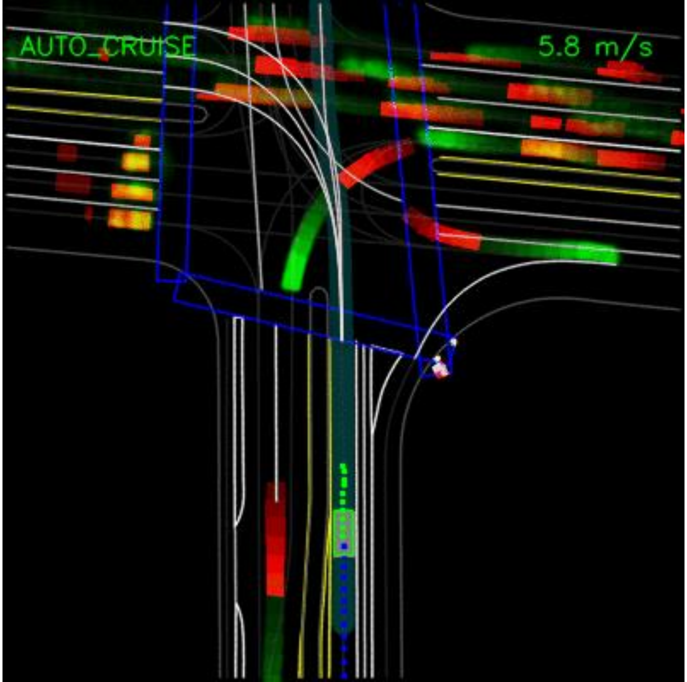
# Input Representation – Easier to Learn From



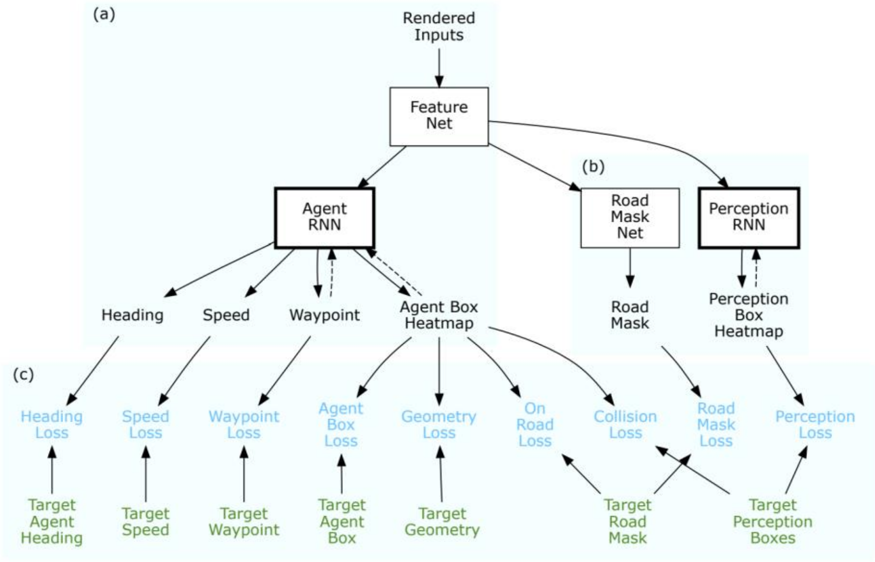
(a) Roadmap (b) Traffic Lights (c) Speed Limit (d) Route

(e) Current Agent Box (f) Dynamic Boxes (g) Past Agent Poses (h) Future Agent Poses

Figure 1: Driving model inputs (a-g) and output (h).


AUTO_CRUISE
5.8 m/s

# Architecture and Auxiliary Tasks



# Output Representation – Trajectory of Waypoints

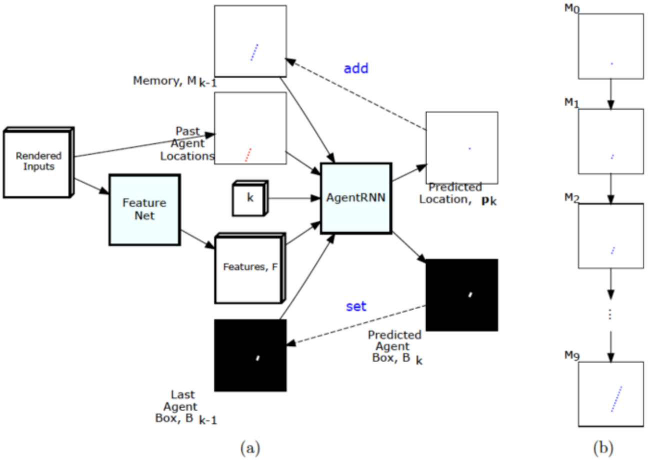


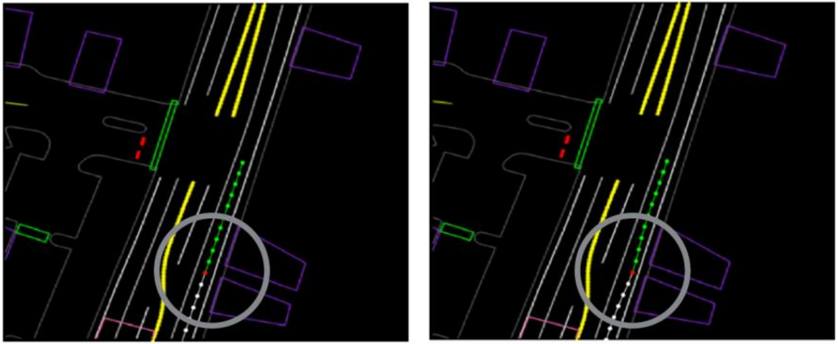

Figure 3: (a) Schematic of ChauffeurNet. (b) Memory updates over multiple iterations.

# Issue with Temporal Imitation Learning

*"During training, the model is provided the past motion history as one of the inputs. Since the past motion history during training is from an expert demonstration, the net can learn to cheat by just extrapolating from the past rather than understanding the underlying causes of the behavior."*

Addressed with past motion dropout

# Beyond Pure Imitation Learning –
# Synthetic Data for Handling Distribution Shift



(a) Original (b) Perturbed

Figure 5: Trajectory Perturbation. (a) An original logged training example where the agent is driving along the center of the lane. (b) The perturbed example created by perturbing the current agent location (red point) in the original example away from the lane center and then fitting a new smooth trajectory that brings the agent back to the original target location along the lane center.

# Beyond Pure Imitation Learning – Auxiliary Losses

$$\mathcal{L}_{collision} = \frac{1}{WH} \sum_{x} \sum_{y} B_k(x, y) \ . \ Obj_k^{gt}(x, y)$$

$$\mathcal{L}_{onroad} = \frac{1}{WH} \sum_{x} \sum_{y} B_k(x, y) \ . \ (1 - Road^{gt}(x, y))$$

$B_k$- predicted agent box (k - timestep)
Obj – Ground truth boxes
Road – road regions

# Imitation Dropout

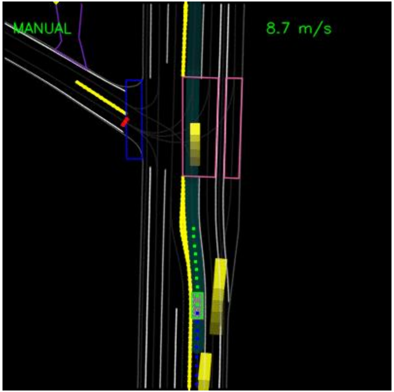
set imitation loss to be 0 randomly

$$\mathcal{L} = w_{imit} \sum_{\ell \in \mathcal{L}_{imit}} \ell + w_{env} \sum_{\ell \in \mathcal{L}_{env}} \ell$$

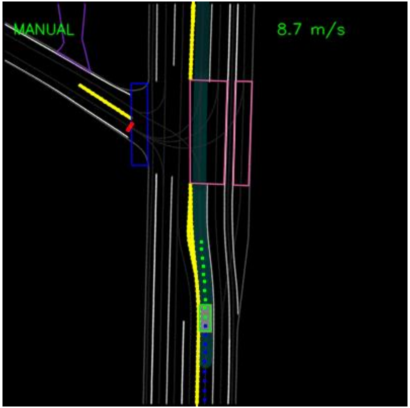
With Stop Signs Rendered


No Stop Signs Rendered

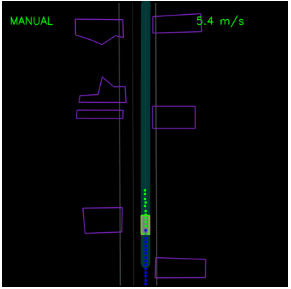
# With Box Rendered



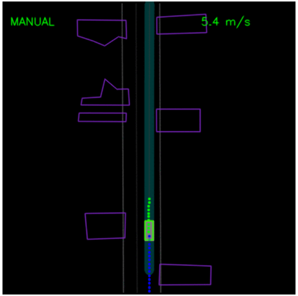

# No Box Rendered



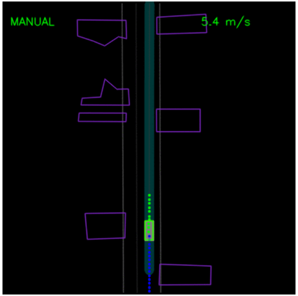
# Nudging Around a Parked Car



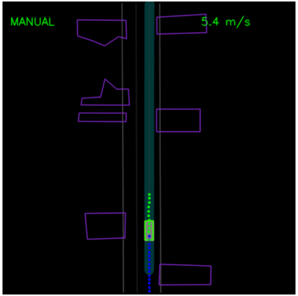

M0 = Imitation with Past Dropout




M1 = M0 + Traj Perturbation

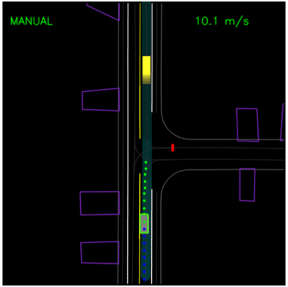



M2 = M1 + Environment Losses

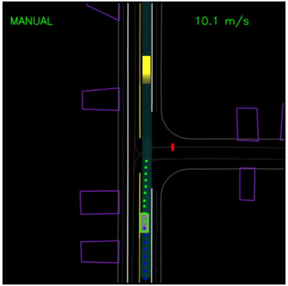



M4 = M2 + Imitation Dropout

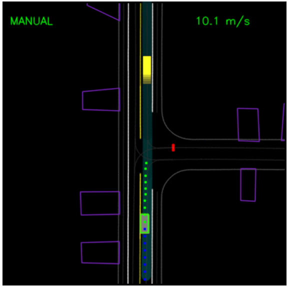
# Slowing down for a Slow Car



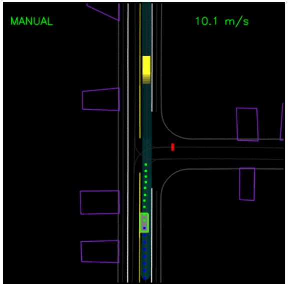

M0 = Imitation with Past Dropout




M1 = M0 + Traj Perturbation

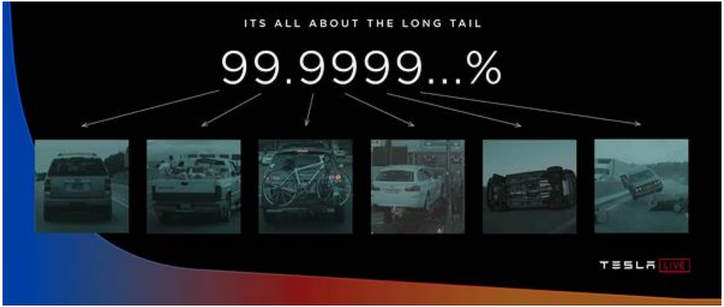



M2 = M1 + Environment Losses




M4 = M2 + Imitation Dropout

# (Human Expert) Data is Essential



# (Human Expert) Data is Essential




# One Company to Rule Them All?


yahoo!finance

**Tesla has 'key advantage' over other automakers, analyst says**

Emily McCormick · Reporter
January 13, 2020

ars TECHNICA SUBSCRIBE SIGN IN

MEET THE NEW BOSSES —

**Waymo CEO John Krafcik steps down**

Waymo ordered "up to" 82,000 vehicles in 2018. Today, it has "well over 600."

TIMOTHY B. LEE - 4/3/2021, 8:30 AM

Waymo ordered "up to 82,000" vehicles in 2018. Today, it has "well over 600"

Forbes

Jul 3, 2020, 10:10am EDT | 63,113 views

**Tesla: King Of Self-Driving Cars? Unbelievable**

Trefis Team Contributor
Great Speculations Contributor Group
Markets

electrek

Exclusives Autos Alt. Transport Autonomy Energy Tesla Shop

OCTOBER 24, 2020

**Tesla is collecting insane amount of data from its Full Self-Driving test fleet**

Fred Lambert - Oct. 24th 2020 3:09 pm ET @FredericLambert

# Underlying Development Process is Inefficient

Difficult for one entity to capture all modes (***unsafe***)
Bulk of data is kept private (***inaccessible, unsafe***)
Redundancy and high cost (***slow progress, urgent application***)

# Underlying Development Process is Inefficient

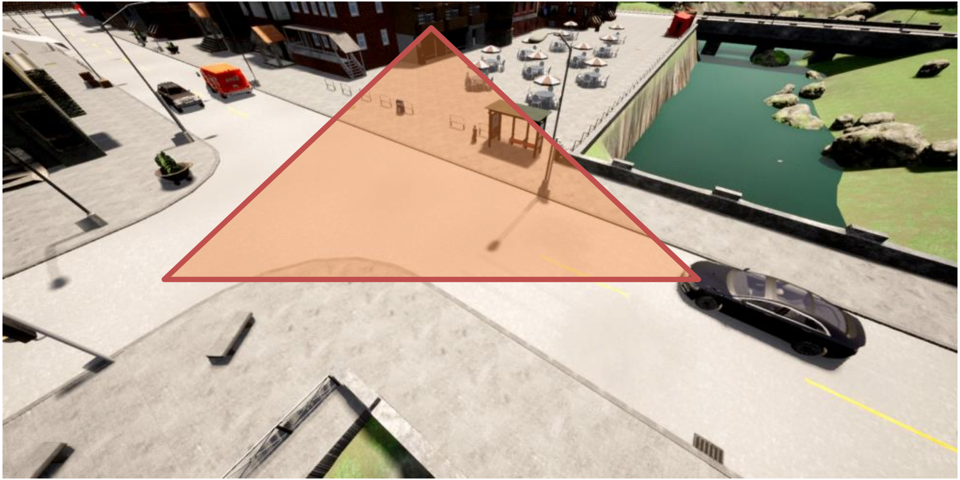
Difficult for one entity to capture all modes (***unsafe***)
Bulk of data is kept private (***inaccessible, unsafe***)
Redundancy and high cost (***slow progress, urgent application***)

Q: How to ***efficiently*** learn a navigation policy?

A: ***Learn by watching (third-person) agents*** to leverage all available demonstration sources in a scene.

Zhang and Ohn-Bar, Learning by Watching, CVPR 2021

# Learning from All Humans in the Scene



# Learning to Drive by Watching

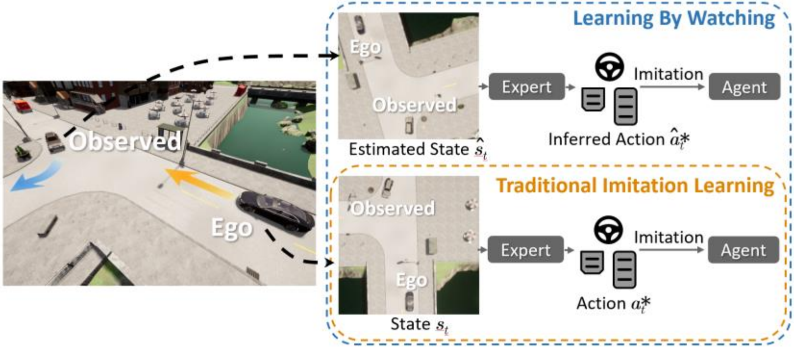
Observations: $\boldsymbol{s} = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $\boldsymbol{a} \in \mathcal{A} = [-1, 1]^2$

Policy: $\pi(\boldsymbol{a} | \boldsymbol{s})$

Trajectories: $\boldsymbol{\tau}^{observed} = \{(\hat{\boldsymbol{s}}_t, \hat{\boldsymbol{a}}_t^*)\}_{t=1}^{T}$




Trajectories: $\boldsymbol{\tau}^{expert} = \{(\boldsymbol{s}_t, \boldsymbol{a}_t^*)\}_{t=1}^{T}$

# Imitation in Humans vs. Imitation in Robotics Today

No direct knowledge of expert actions or state

Jones SS. The development of imitation in infancy. 2009
Bandura A. Social Learning Theory.

# Imitation in Humans vs. Imitation in Robotics Today

No direct knowledge of expert actions or state

Jones SS. The development of imitation in infancy. 2009
Bandura A. Social Learning Theory.

Full access to sensory state, ego-expert actions

# Imitation in Humans vs. Imitation in Robotics Today

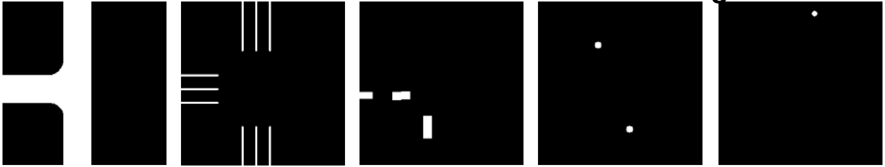
No direct knowledge of expert actions or state

Jones SS. The development of imitation in infancy. 2009
Bandura A. Social Learning Theory.

Full access to sensory state, ego-expert actions

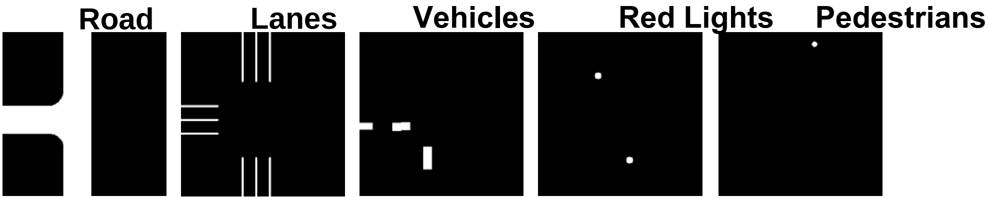
(1) Humans transform observations to their points of view
(2) Infer others' expert actions

# Compact Bird's-Eye-View (BEV) State Representation

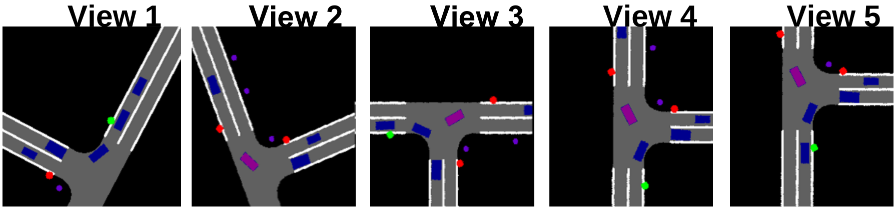



Enables relatively straightforward ***perspective transform***

# Compact Bird's-Eye-View (BEV) State Representation

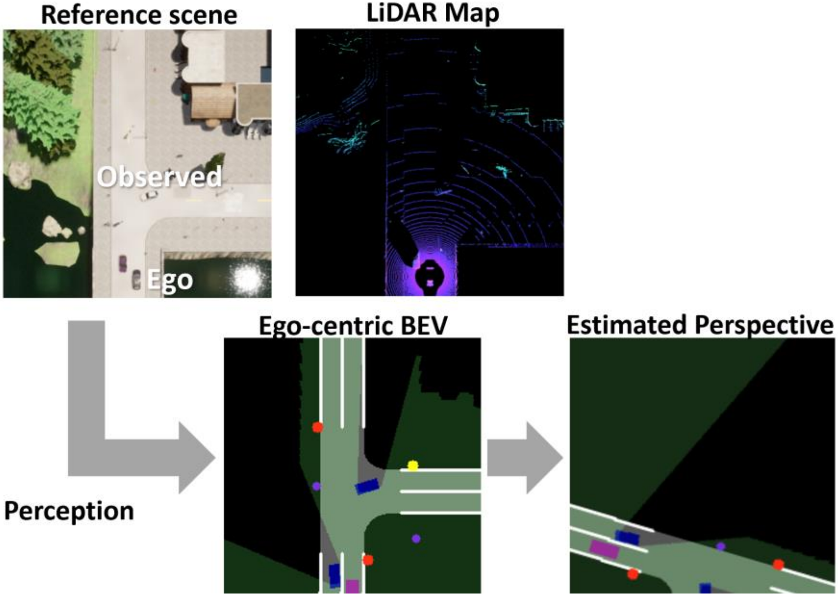



Enables relatively straightforward ***perspective transform***




Missing information regrading the ***original perspective***!

# Handling Occluded Regions With Visibility Map

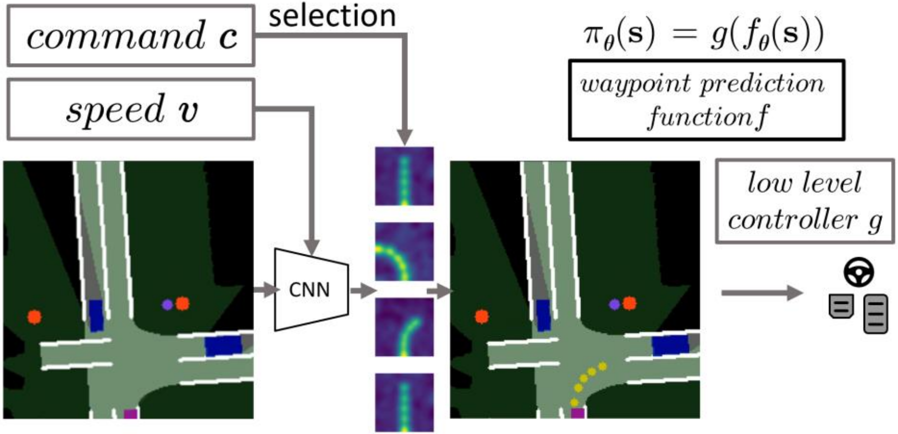


# Waypoint-Based Action Representation

$$\tau^{observed} = \{(\hat{\mathbf{s}}_t,\ \hat{\mathbf{a}}^*_t)\}_{t=1}^{T} \rightarrow \tau^{observed} = \{(\hat{\mathbf{s}}_t,\ \hat{\mathbf{w}}^*_t)\}_{t=1}^{T}$$

# Waypoint-Based Action Representation

$$\tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{a}}^*_t)\}_{t=1}^{T} \rightarrow \tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{w}}^*_t)\}_{t=1}^{T}$$




$$\mathcal{L}_{\text{behavior-cloning}} = \mathbb{E}_{(\mathbf{s},\mathbf{w})\sim\mathbf{D}}[\ell_1(\mathbf{w}, f_\theta(\mathbf{s}))]$$

# Low-Data Regime on CARLA Benchmark

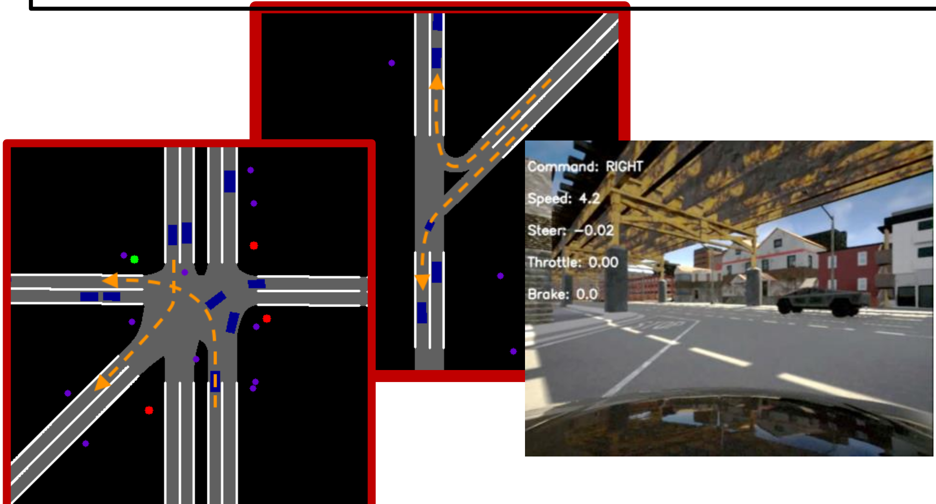
Town 1: Training

Town 2: Testing

***<u>Metric</u>***: Percentage of successfully completed episodes (**success rate**)

# Low-Data Adaptation Benchmark

Town 3: Novel routes, intersection types, 5-lane junctions, roundabouts, tunnels



# Results

| Model | One Hour Training | 10 Minutes Training | New Town |
|---|---|---|---|
| Ego (Baseline) | 46 | 24 | 40 |
| LbW | 64 | 34 | 60 |
| LbW + Visibility (Early) | 52 | 28 | 60 |
| LbW + Visibility (Late) | **92** | **52** | **100** |

**Baseline**

**LbW**

Driving Results on Town 2

(*10 Minutes* of Policy Training Data)

# *Conditional* Imitation Learning

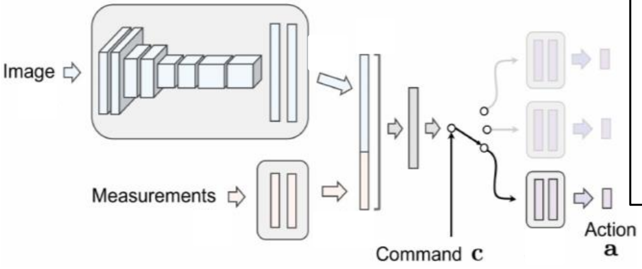
Observations: $\boldsymbol{s} = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $\boldsymbol{a} \in \mathcal{A} = [-1, 1]^2$

Policy: $\pi(\boldsymbol{a} | \boldsymbol{s}, c)$

Architecture (trained end-to-end):




Optimization?
Generalization?
Task supervision?
Safety?
Covariate shift?
Catastrophic failure?

Codevllia, et al., ICCV’19, ICRA’18

# Conditional Imitation Learning

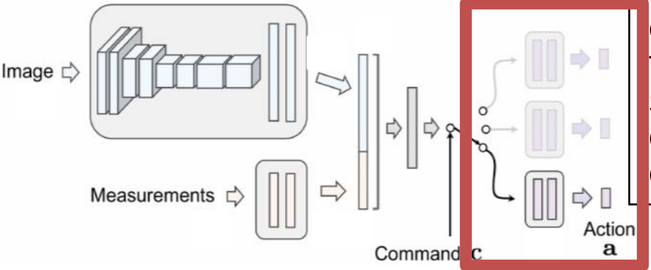
Observations: $s = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $\boldsymbol{a} \in \mathcal{A} = [-1,1]^2$

Policy: $\pi(\boldsymbol{a} | \boldsymbol{s}, c)$

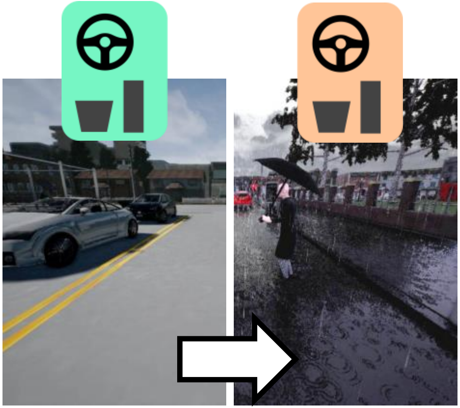
Architecture (trained end-to-end):




Optimization?
Generalization?
Task supervision?
Safety?
Covariate shift?
Catastrophic failure?

Codevllia, et al., ICCV’19, ICRA’18

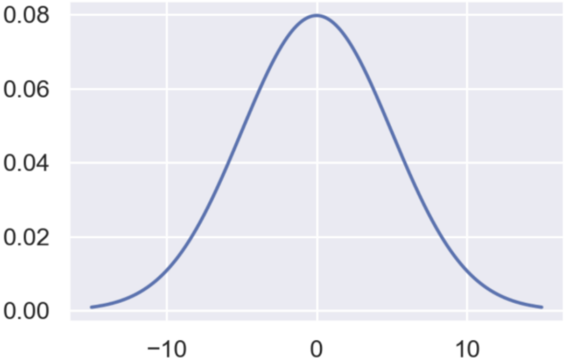
# Humans Use Situation-Specific Awareness and Strategies



M. R. Endsley. Toward a Theory of Situation Awareness in Dynamic Systems, 2000.
C. C. Macadam. Understanding and modeling the human driver. Vehicle System Dynamics, 2003.

# Most networks fit a Gaussian to labels

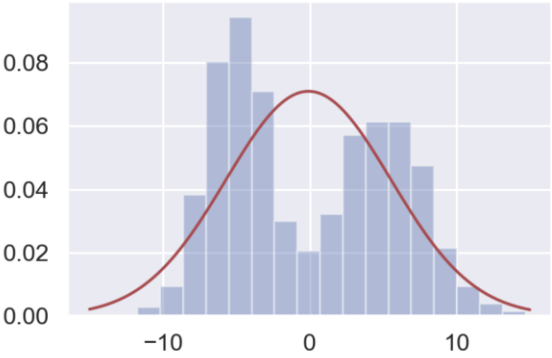
- “Standard” probability distribution
- Has two parameters:
  - mean (μ) and
  - standard deviation (σ)
- Probability Density Function:
  - $$N(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



# PROBLEM: NORMAL DISTRIBUTION MIGHT NOT FIT DATA

What if the data is complicated?

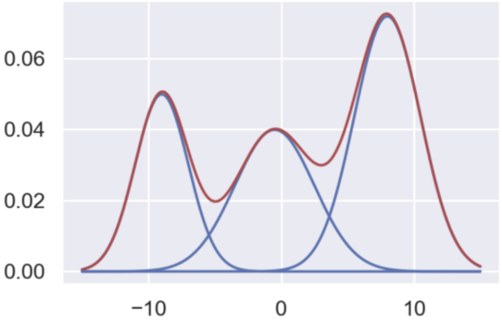
- It's easy to "fit" a normal model to any data.
  - Just calculate $\mu$ and $\sigma$
- But this might not fit the data well.



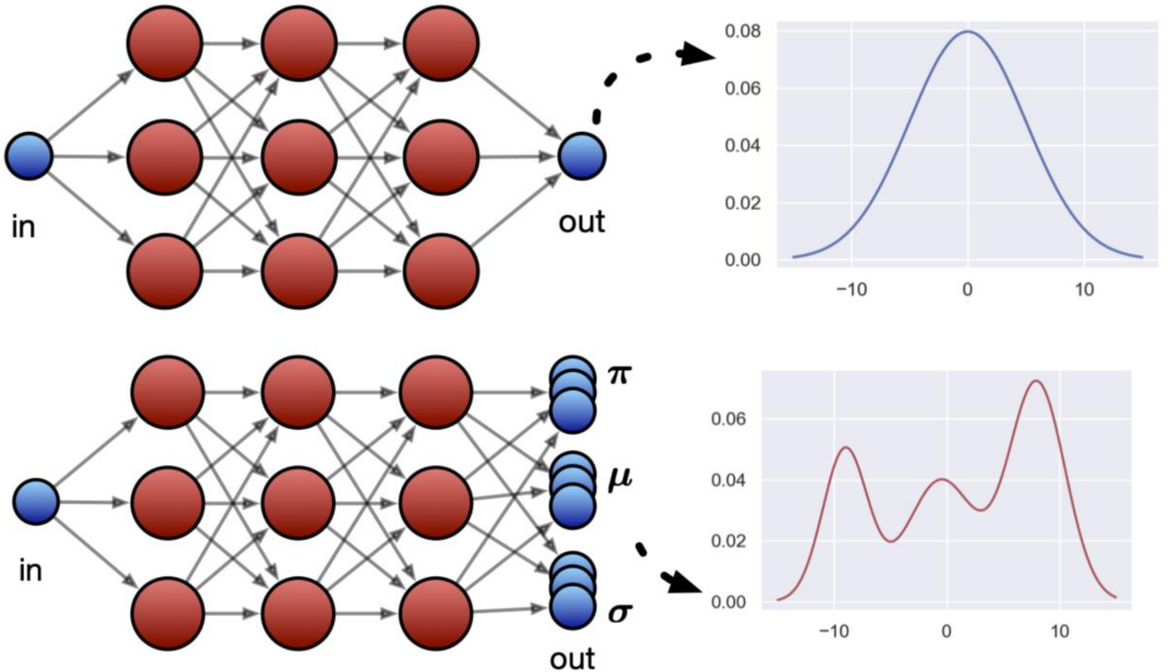
# MIXTURE OF NORMALS

Three groups of parameters:

- means ($\mu$): location of each component
- standard deviations ($\sigma$): width of each component
- Weight ($\pi$): height of each curve
- Probability Density Function:
  - $$p(x) = \sum_{i=1}^{K} \pi_i N(x \mid \mu, \sigma^2)$$

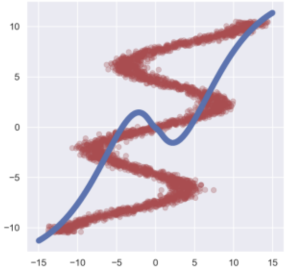


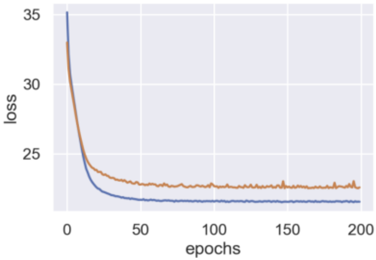

Mixture Density Networks, Bishop, 1994



in
out
0.08
0.06
0.04
0.02
0.00
−10
0
10
π
μ
σ
in
out
0.06
0.04
0.02
0.00
−10
0
10

# FEEDFORWARD MSE NETWORK

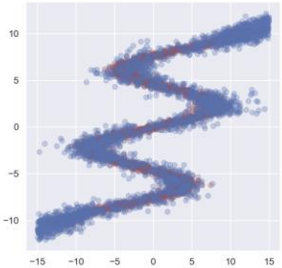


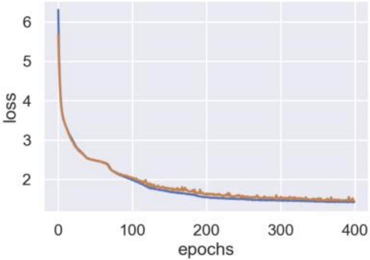




Here's a simple two-hidden-layer network (286 parameters), trained to produce the above result.

```
model = Sequential()
model.add(Dense(15, batch_input_shape=(None, 1), activation='tanh'))
model.add(Dense(15, activation='tanh'))
model.add(Dense(1, activation='linear'))
model.compile(loss='mse', optimizer='rmsprop')
model.fit(x=x_data, y=y_data, batch_size=128, epochs=200, validation_split=0.15)
```

# FEEDFORWARD MDN SOLUTION



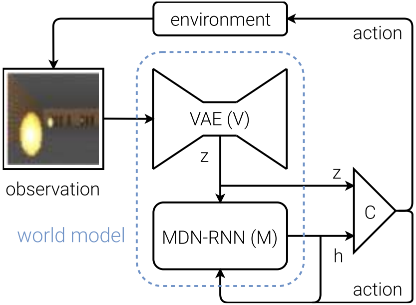




And, here's a simple two-hidden-layer MDN (510 parameters), that achieves the above result! Much better!

```
N_MIXES = 5

model = Sequential()
model.add(Dense(15, batch_input_shape=(None, 1), activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(mdn.MDN(1, N_MIXES)) # here's the MDN layer!
model.compile(loss=mdn.get_mixture_loss_func(1,N_MIXES), optimizer='rmsprop')
model.summary()
```

# Inspiration: World Models (Didn't Work Very Well...)




<u>Step 1</u>: Learn **generative model** of game environment (VAE)

<u>Step 2</u>: Learn dynamics and control models in **latent space** (CMA-ES)

| Method | Score |
|---|---|
| DQN | 343 |
| A3C | 652 |
| World Models | **906** |

Fails beyond simplest settings. ***<u>Mixture Density Network (MDN) is key</u>***.

Ha and Schmidhuber, Recurrent World Models Facilitate Policy Evolution , NeurIPS 2018

# Proposed ***Learning Situational Driving (LSD)*** Framework

Observations: $\boldsymbol{s} = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $\boldsymbol{a} \in \mathcal{A} = [-1, 1]^2$

Policies: $\Pi = \{\pi_\theta^1, \ldots, \pi_\theta^K\}$

Parameters: $\Theta = \{\theta, \phi, \boldsymbol{\Psi}\}$

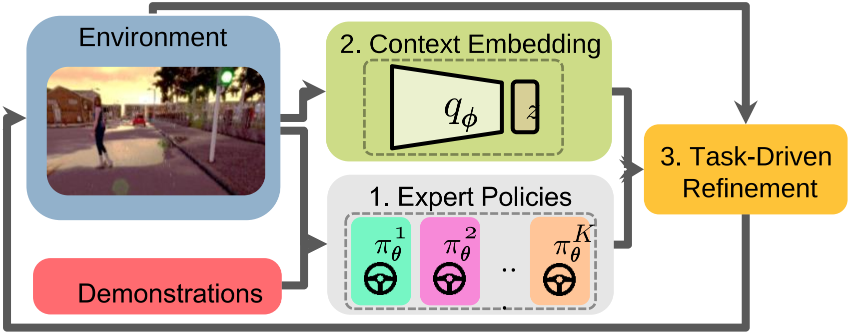
$$\pi_\Theta(\boldsymbol{a}|\boldsymbol{s}, c) = \sum_{k=1}^{K} \underbrace{\alpha_\theta^k(\boldsymbol{s}, c)}_{\text{Mixture Weights}} \underbrace{\pi_\theta^k(\boldsymbol{a}|\boldsymbol{s}, c)}_{\textit{Expert Models}} + \boldsymbol{\Psi} \underbrace{\begin{bmatrix} q_\phi(\mathbf{I}) \\ v \\ c \end{bmatrix}}_{\text{Context Embedding}}$$

$$\pi_\theta^k(\boldsymbol{a}|\boldsymbol{s}, c) = \mathcal{N}\left(\boldsymbol{a}|\boldsymbol{\mu}_\theta^k(\boldsymbol{s}, c), \mathrm{diag}(\boldsymbol{\sigma}_\theta^k(\boldsymbol{s}, c)^2)\right)$$

**Key Idea**

Learn **specialized experts** and combine in a context-dependent

# Multistep Model Training and Optimization




<u>Step 1</u>: Learn **mixture of experts** via imitation $\mathcal{L}_{\text{MoE}} = -\log[\sum_{k=1}^{K} \alpha_\theta^k \pi_\theta^k]$

<u>Step 2</u>: Learn a general purpose context encoder $q_\phi$ as a $\beta$-VAE

$$\mathcal{L}_{\text{VAE}} = \beta \ \mathrm{KL}(q_\phi(\boldsymbol{z}|\mathbf{I}) \ || \ \mathcal{N}_0) + ||d_\phi(\boldsymbol{z}) - \mathbf{I}||^2$$

<u>Step 3</u>: **Task-driven** policy refinement with CMA-ES for $\mathbb{E}_{\pi_\Theta}[\sum_t r_t]$

# CARLA Benchmark

***<u>Metric</u>***: Percentage of successfully completed episodes (**success rate**)

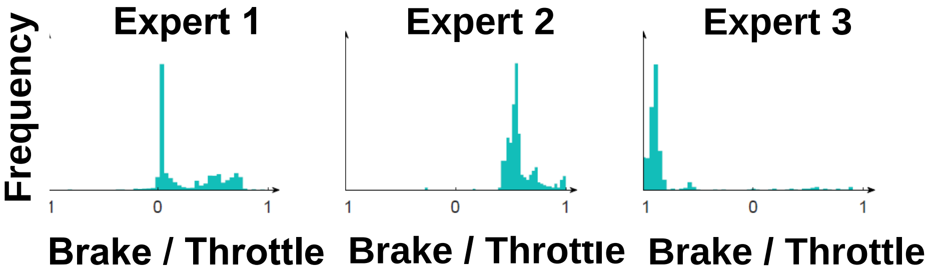
Town 1: Training

Town 2: Testing

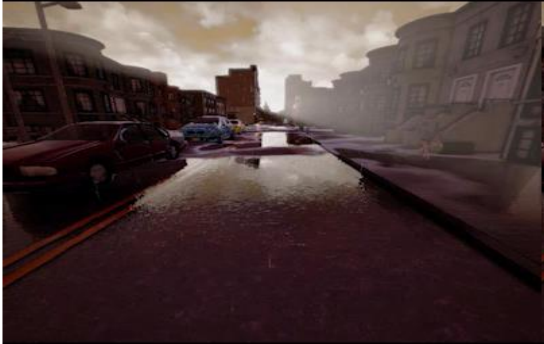
# Results & Emergent Driving Behavior Modes

| Driving Task | CIRL (ECCV18) | CILRS (ICCV19) | LSD (ours, step 1) | LSD+Reward (ours, steps 1-3) |
|---|---|---|---|---|
| Straight (Static) | **100** | 96 | **100** | **100** |
| One Turn (Static) | 71 | 84 | **100** | **100** |
| Navigation (Static) | 53 | 69 | **98** | **100** |
| Dense Traffic | 41 | 66 | 92 | **98** |

Acceleration distribution of three different experts during testing



LSD
(***Times-out*** due to reflection)



LSD+Reward
(***Success***)

# Further Readings

- Learning by Cheating, Conference on Robot Learning, 2019
- Learning by Watching, CVPR, 2021
- Learning monocular reactive UAV control in cluttered natural environments, ICRA 2013
- Learning Situational Driving, CVPR, 2020