Imitation Learning with F1Tenth

Xiatao Sun

Penn Engineering

A Benchmark Comparison of Imitation Learning-based Control Policies for Autonomous Racing

Xiatao Sun, Mingyan Zhou, Zhijun Zhuang, Shuo Yang, Johannes Betz, Rahul Mangharam

Penn Engineering

Motivation

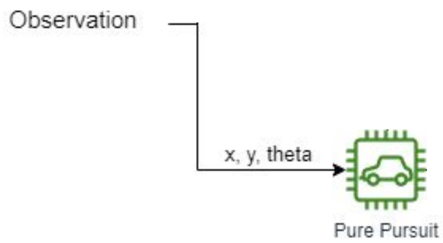
- Autonomous racing gains increasing popularity recently
- Most prior works focus on human-engineered or reinforcement learning methods
- Our work applies and compares various imitation learning methods in autonomous racing scenarios

Source code: https://github.com/mlab-upenn/f1tenth_il

Demo

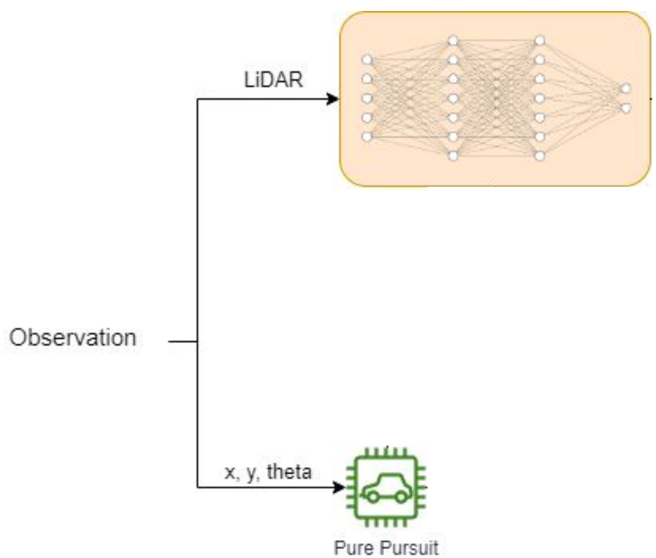
Implementation

- Expert: pure pursuit algorithm



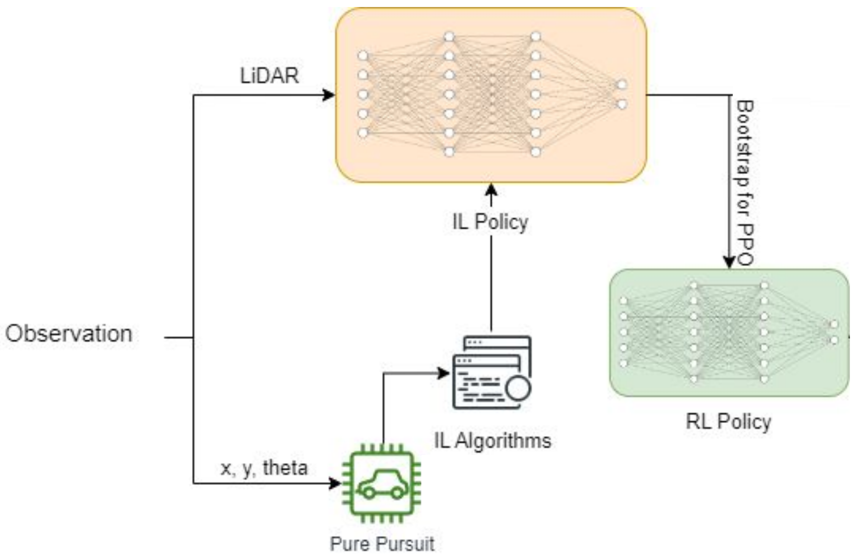
Implementation

- Expert: pure pursuit algorithm
- Learner: 2-layer MLP with 256 hidden units



Implementation

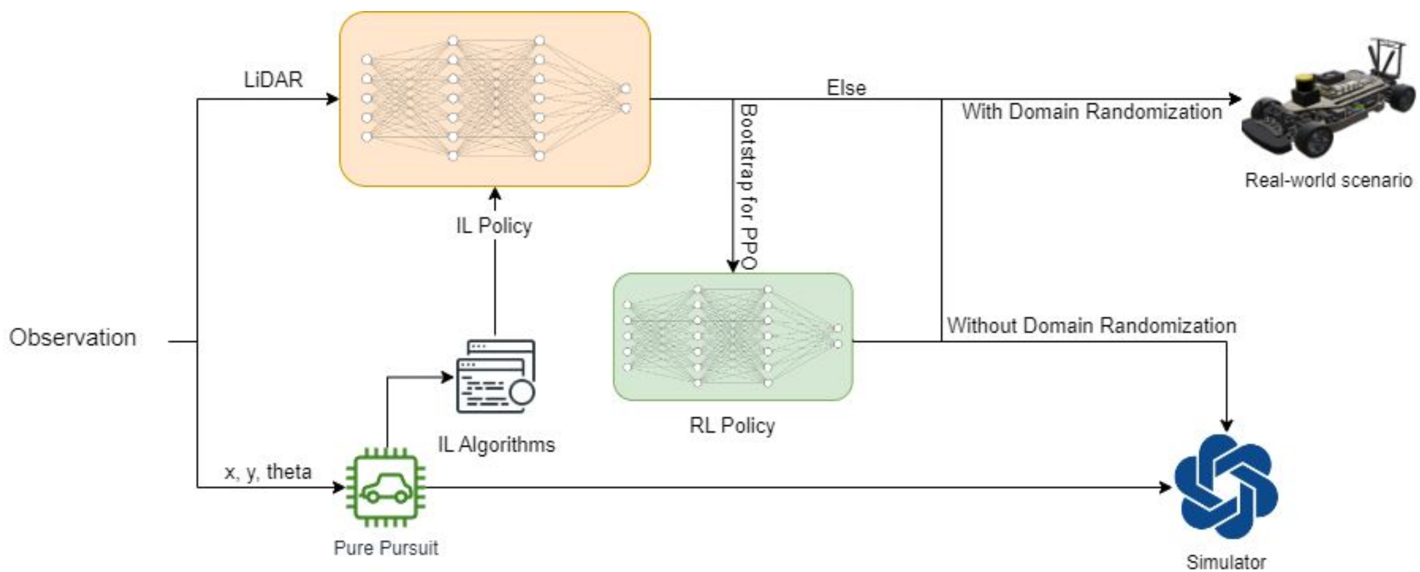
- Expert: pure pursuit algorithm
- Learner: 2-layer MLP with 256 hidden units
- Learning algorithms: BC, DAGGER, HG-DAGGER, EIL



Algorithm	Intervention Rule	Loss Function
EIL	Intervene if $(s, a) \notin \mathcal{G}$	$\ell_B(\cdot) + \lambda \ell_C(\cdot)$
BC	Expert in control	$\ell_C(\cdot)$
DAGGER	Learner in control	$\ell_C(\cdot)^\dagger$
HG-DAGGER	Intervene if $(s, a) \notin \mathcal{G}$	$\ell_C(\cdot)$

Implementation

- Expert: pure pursuit algorithm
- Learner: 2-layer MLP with 256 hidden units
- Learning algorithms: BC, DAGGER, HG-DAGGER, EIL

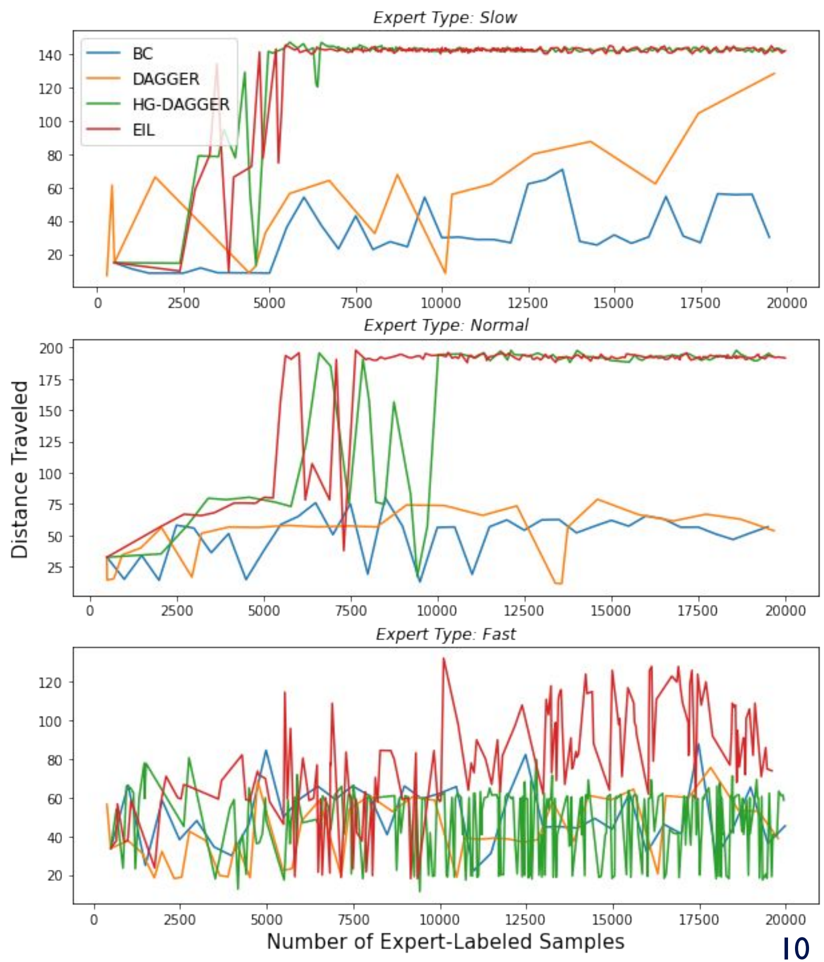


Comparison and Evaluation

- Sample and bootstrapping efficiency
- Performance and generalizability
- The experiments are performed in both simulation and real-world environments

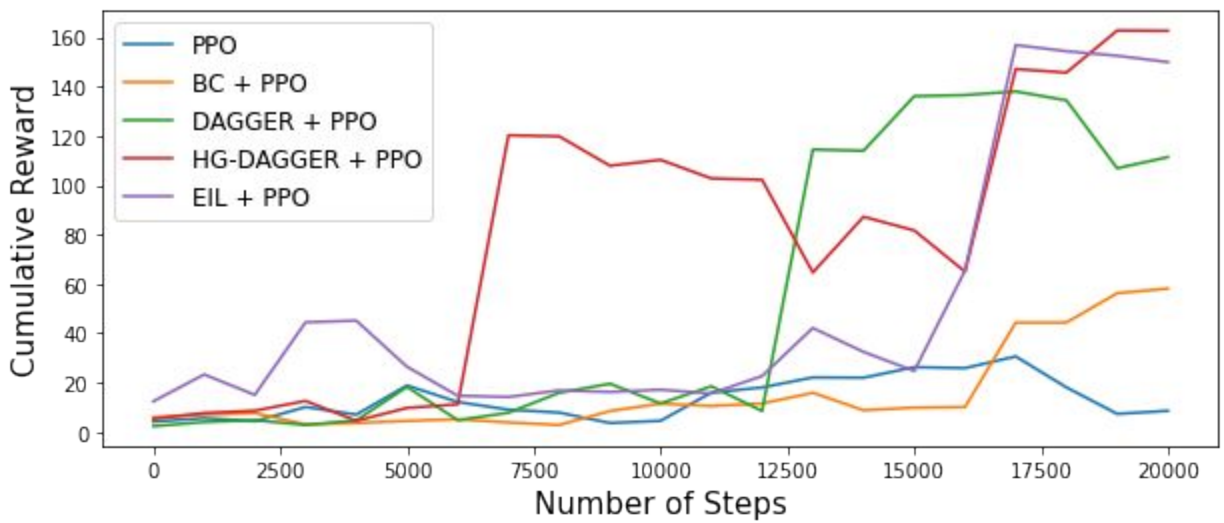
Sample Efficiency

- Training in simulation with respect to 3 different expert
  - Slow: 4.79m/s
  - Normal: 6.39 m/s
  - Fast: 8.24 m/s
- Overall interactive IL demonstrate better sample efficiency



Bootstrapping efficiency

- Using any IL algorithms can help PPO converge to a better policy
- Interactive IL has better bootstrapping efficiency



Performance in training map

- Data aggregation mitigates the problem of compounding errors
- EIL has the best performance and is the only one successfully completed one lap when the speed is fast

Expert Type	Expert	BC	DAGGER	HG-DAGGER	EIL
Slow	33.07 s	Failed	34.34 s	33.78 s	33.50 s
Normal	25.04 s	25.35 s	25.85 s	25.06 s	25.22 s
Fast	19.69 s	Failed	Failed	Failed	20.40 s

Elapsed time of IL policies trained with different experts

Generalizability

- The combinations of IL and PPO significantly outperform policies only using IL or PPO, and can efficiently converge to a more generalized policy
- Interactive IL can train policies that are more similar to expert behavior than non-interactive IL and PPO

Method	BC	DAGGER	HG-DAGGER	EIL	PPO	BC+PPO	DAGGER+PPO	HG-DAGGER+PPO	EIL+PPO
Distance Traveled (m)	7.84	8.90	12.34	15.89	12.69	151.23	86.49	155.88	150.15
Complete 1 Lap	No	No	No	No	No	Yes	No	Yes	Yes
Bhattacharyya Distance	0.77	0.60	0.12	0.24	1.09	0.59	0.59	0.47	0.43

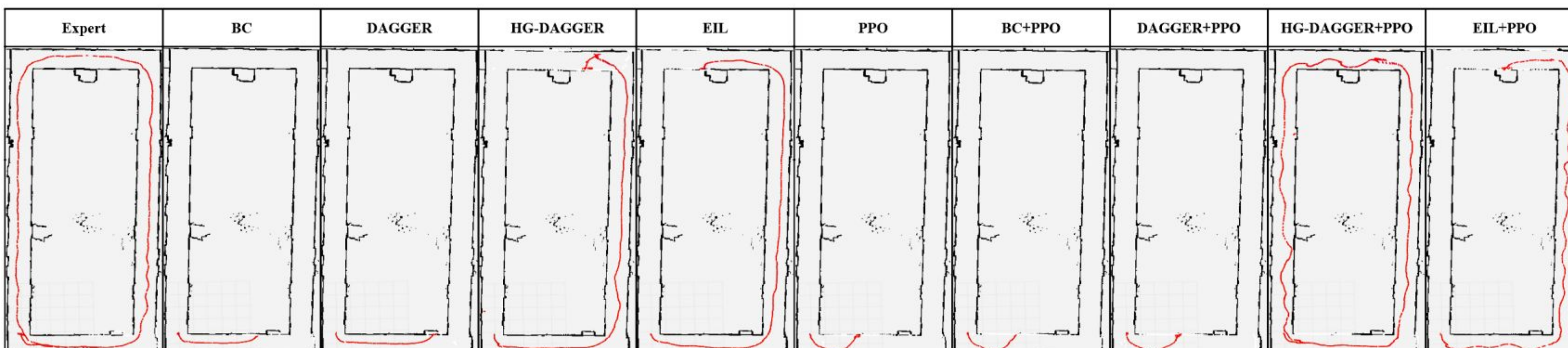
Evaluations of different learned policies in an unseen simulation environment

Experiments in real-world environment

- Policies from interactive IL can travel further distances
- PPO policies have more warbling compared with IL policies
- HG-DAGGER + PPO has the best performance

Method	Expert	BC	DAGGER	HG-DAGGER	EIL	PPO	BC+PPO	DAGGER+PPO	HG-DAGGER+PPO	EIL+PPO
Distance Traveled (m)	61.44	6.44	8.49	37.74	38.04	5.27	6.44	6.29	64.08	39.5
Complete 1 Lap	Yes	No	No	No	No	No	No	No	Yes	No

Evaluations of different learned policies in the real-world environment.



The movement trajectories of the F1TENTH vehicle on the map under the control of each policy

Conclusion

- IL algorithms can train or bootstrap high-performance policies for autonomous racing
- Interactive mechanism improves sample efficiency and bootstrapping efficiency of IL
- Combining IL and RL can achieve fast convergence and better generalizability

MEGA-DAgger: Imitation Learning with Multiple Imperfect Experts

Xiatao Sun*, Shuo Yang*, Rahul Mangharam

*Authors contributed equally.

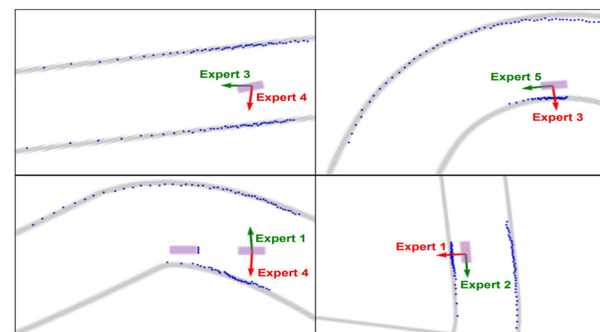
Penn Engineering

Overview

- Existing interactive imitation learning algorithms requires **one optimal expert**
- Real-world scenarios: **imperfect experts**

- **Challenges:**
  - Undesired demonstrations
  - Conflicted demonstrations

Reduced-scale vehicle controlled by policy learned using HG-DAgger



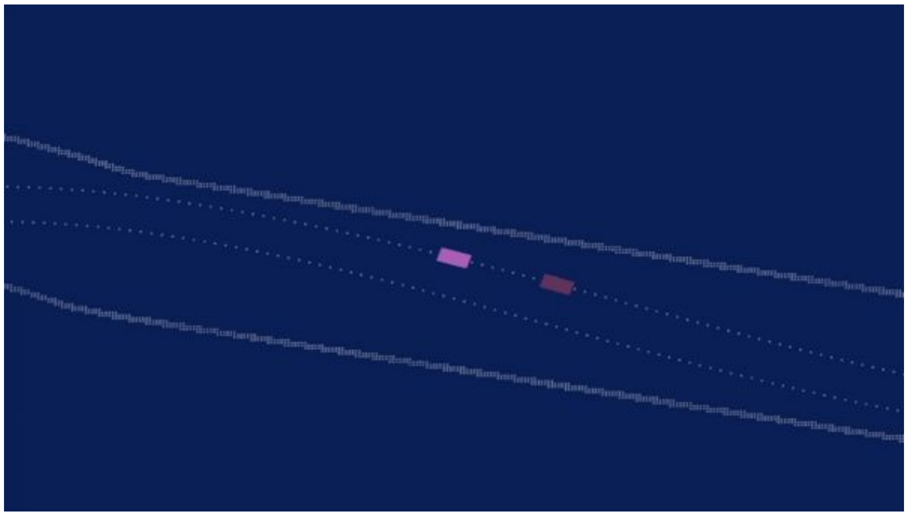
Example of conflicted demonstrations from different experts

Xiatao Sun, Mingyan Zhou, Zhijun Zhuang, Shuo Yang, Johannes Betz, and Rahul Mangharam. A benchmark comparison of imitation learning-based control policies for autonomous racing. arXiv preprint arXiv:2209.15073, 2022.

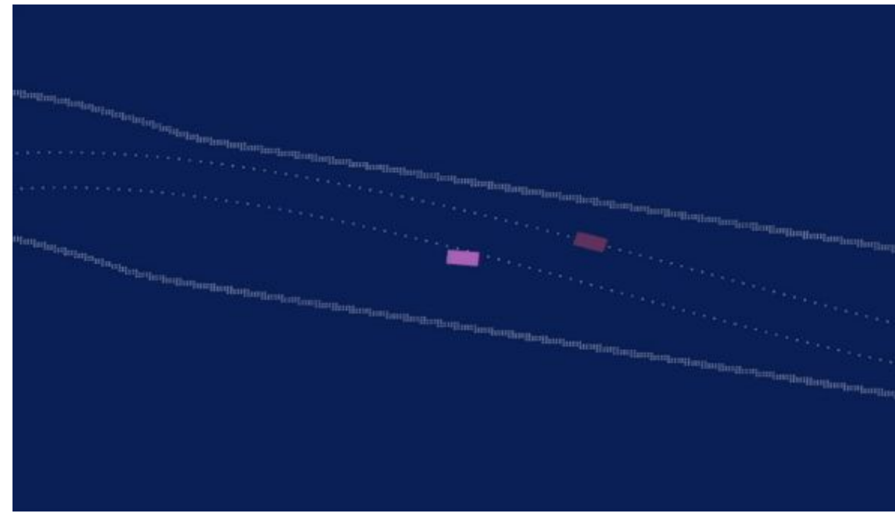
Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In 2019 International Conference on Robotics and Automation (ICRA), pages 8077–8083. IEEE, 2019.

Comparison: HG-DAgger & MEGA-DAgger

HG-DAgger



MEGA-DAgger (ours)



Comparison: Experts & MEGA-DAgger

Experts 5

MEGA-DAgger (ours)

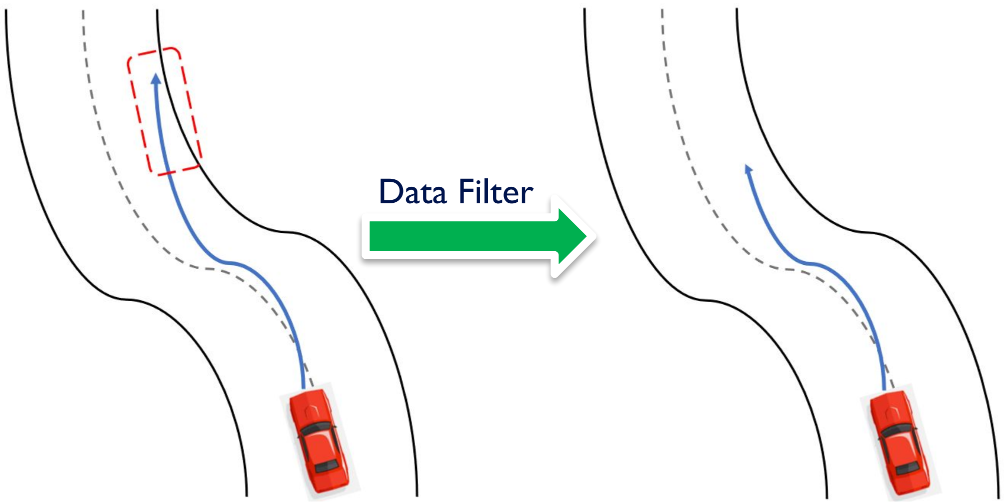
MEGA-DAgger: **M**ultiple **E**xperts **GA**ted **DAgger**

π_{exp}^{j}	Expert policies
D	Global dataset
π_{N_0}	Initial novice policy
Π	All possible policies
o	Observation
a	Action
D_j	Dataset of current rollout
σ_t	Safety score
π_{N_i}	Novice policy of current iteration
K	Total number of iterations
M	Total number of rollouts for each iteration
T	Total number of timesteps

Algorithm 1 MEGA-DAgger

procedure MEGA-DAGGER($\pi_{exp}^{1:M}$)
    Initialize $\mathcal{D} \leftarrow \emptyset$
    Initialize π_{N_0} to any policy in Π
    for $\texttt{iteration}\ i = 1 : K$ **do**
        for $\texttt{rollout}\ j = 1 : M$ with expert π_{exp}^{j} **do**
            for $\texttt{timestep}\ t \in T$ of $\texttt{rollout}\ j$ **do**
                if π_{exp}^{j} takes control **then**
                    $o \leftarrow \texttt{rollout}_{i,j}^{t}$
                    $a \leftarrow \pi_{exp}^{j}(o)$
                    $D_j \leftarrow o, a$
                    $D_j, \sigma_t \leftarrow$ DATA FILTER(D_j)
                end if
            end for
            $D_j \leftarrow$ CONFLICT RESOLUTION(D_j, D, σ_t)
            $D \leftarrow D \cup D_j$
        end for
        Train π_{N_i} on $\mathcal{D}$
    end for
end procedure

MEGA-DAgger: Data Filter



Algorithm 1 MEGA-DAgger

procedure MEGA-DAGGER($\pi_{exp}^{1:M}$)
  Initialize $\mathcal{D} \leftarrow \emptyset$
  Initialize π_{N_0} to any policy in Π
  for iteration $i = 1 : K$ **do**
    for rollout $j = 1 : M$ with expert π_{exp}^{j} **do**
      for timestep $t \in T$ of rollout j **do**
        if π_{exp}^{j} takes control **then**
          $o \leftarrow \texttt{rollout}_{i,j}^{t}$
          $a \leftarrow \pi_{exp}^{j}(o)$
          $D_j \leftarrow o, a$
          $D_j, \sigma_t \leftarrow \boxed{\text{DATA FILTER}(D_j)}$
        end if
      end for
      $D_j \leftarrow$ CONFLICT RESOLUTION(D_j, D, σ_t)
      $D \leftarrow D \cup D_j$
    end for
    Train π_{N_i} on $\mathcal{D}$
  end for
end procedure

MEGA-DAgger: Data Filter

- Control Barrier Function (CBF):

$$h(x_t^e, y_t^e) = (x_t^e - x_t^p)^2 - (y_t^e - y_t^p)^2 - \alpha^2$$

- Safety score:

$$\sigma_t = h(x_{t+1}^e, y_{t+1}^e) - (1-\gamma)h(x_t^e, y_t^e)$$

$$0 < \gamma \leq 1$$

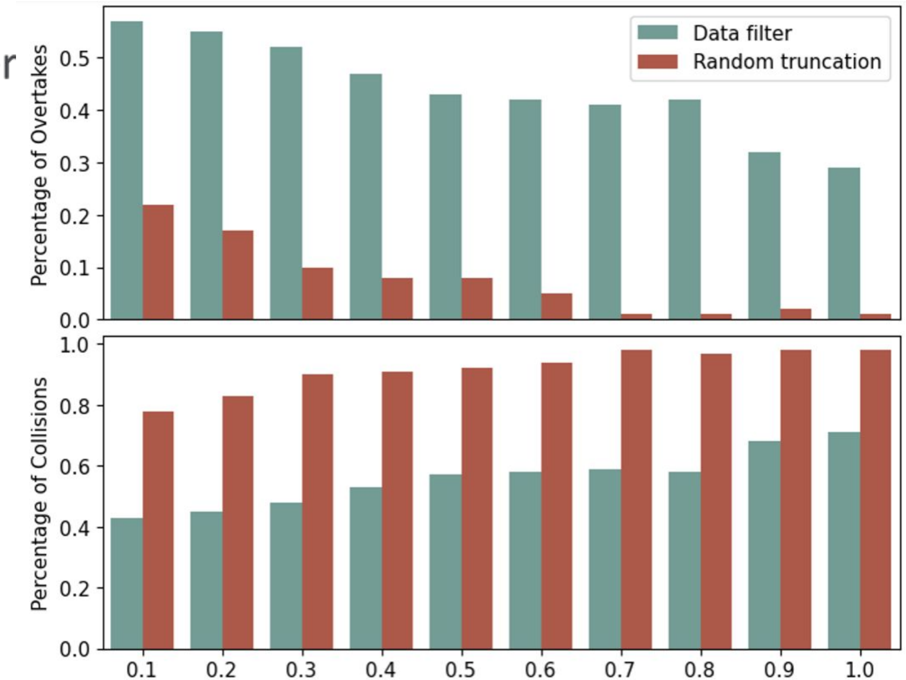
(x_t^e, y_t^e)	Current ego position
(x_t^p, y_t^p)	Current obstacle position
α	Minimal safety distance

Algorithm 1 MEGA-DAgger

procedure MEGA-DAGGER($\pi_{exp}^{1:M}$)
    Initialize $\mathcal{D} \leftarrow \emptyset$
    Initialize π_{N_0} to any policy in Π
    for `iteration` $i = 1 : K$ **do**
        for `rollout` $j = 1 : M$ with expert π_{exp}^j **do**
            for `timestep` $t \in T$ of `rollout` j **do**
                if π_{exp}^j takes control **then**
                    $o \leftarrow \texttt{rollout}_{i,j}^t$
                    $a \leftarrow \pi_{exp}^j(o)$
                    $D_j \leftarrow o, a$
                    $D_j, \sigma_t \leftarrow \boxed{\text{DATA FILTER}(D_j)}$
                end if
            end for
            $D_j \leftarrow$ CONFLICT RESOLUTION(D_j, D, σ_t)
            $D \leftarrow D \cup D_j$
        end for
        Train π_{N_i} on $\mathcal{D}$
    end for
end procedure

Evaluation: Data Filter

- Comparison: HG-DAgger with data filter and HG-DAgger with random data truncation
- The proposed data filter is able to effectively remove undesired demonstrations and result in a better policy

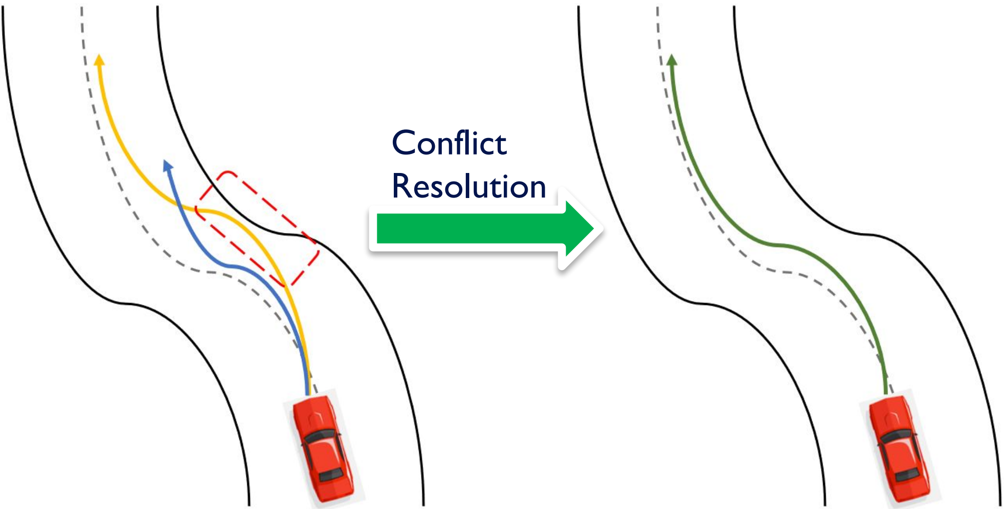


$P(U)$	0.1	0.2	0.3	0.4	0.5
r_β	0.22	0.25	0.28	0.32	0.34
$P(U)$	0.6	0.7	0.8	0.9	1.0
r_β	0.38	0.49	0.57	0.69	0.82

- $P(U)$: Undesired behavior probability
- r_β: Ratio of removed demonstrations

Penn Engineering

MEGA-DAgger: Conflict Resolution



Algorithm 1 MEGA-DAgger

procedure MEGA-DAGGER($\pi_{exp}^{1:M}$)
  Initialize $\mathcal{D} \leftarrow \emptyset$
  Initialize π_{N_0} to any policy in Π
  for $\mathtt{iteration}\ i = 1 : K$ **do**
    for $\mathtt{rollout}\ j = 1 : M$ with expert π_{exp}^{j} **do**
      for $\mathtt{timestep}\ t \in T$ of $\mathtt{rollout}\ j$ **do**
        if π_{exp}^{j} takes control **then**
          $o \leftarrow \mathtt{rollout}_{i,j}^{t}$
          $a \leftarrow \pi_{exp}^{j}(o)$
          $D_j \leftarrow o, a$
          $D_j, \sigma_t \leftarrow$ DATA FILTER(D_j)
        end if
      end for
      $D_j \leftarrow \boxed{\text{CONFLICT RESOLUTION}(D_j, D, \sigma_t)}$
      $D \leftarrow D \cup D_j$
    end for
    Train π_{N_i} on $\mathcal{D}$
  end for
end procedure

MEGA-DAgger: Conflict Resolution

- Cosine similarities:

$$\Theta = \frac{O \cdot O_j}{\|O\| \odot \|O_j\|}$$

- Evaluation score:

$$\omega_t = \frac{\|\sigma_t\| - \mathtt{min}_t\|\sigma_t\|}{\mathtt{max}_t\|\sigma_t\| - \mathtt{min}_t\|\sigma_t\|} + \frac{\|v_t\| - \mathtt{min}_t\|v_t\|}{\mathtt{max}_t\|v_t\| - \mathtt{min}_t\|v_t\|}$$

O	Observations in D
O_j	Observations in D_j
v_t	Ego velocity

Algorithm 1 MEGA-DAgger

procedure MEGA-DAGGER($\pi_{exp}^{1:M}$)
    Initialize $\mathcal{D} \leftarrow \emptyset$
    Initialize π_{N_0} to any policy in Π
    for $\mathtt{iteration}\ i = 1 : K$ **do**
        for $\mathtt{rollout}\ j = 1 : M$ with expert π_{exp}^{j} **do**
            for $\mathtt{timestep}\ t \in T$ of $\mathtt{rollout}\ j$ **do**
                if π_{exp}^{j} takes control **then**
                    $o \leftarrow \mathtt{rollout}_{i,j}^{t}$
                    $a \leftarrow \pi_{exp}^{j}(o)$
                    $D_j \leftarrow o, a$
                    $D_j, \sigma_t \leftarrow$ DATA FILTER(D_j)
                end if
            end for
            $D_j \leftarrow$ $\boxed{\text{CONFLICT RESOLUTION}(D_j, D, \sigma_t)}$
            $D \leftarrow D \cup D_j$
        end for
        Train π_{N_i} on $\mathcal{D}$
    end for
end procedure

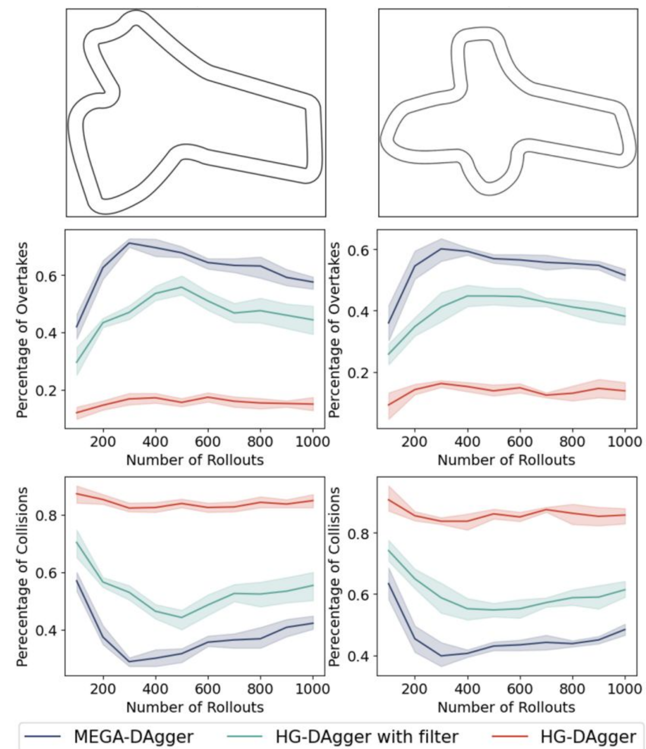
Evaluation: Comparison with HG-DAgger

- Comparison: MEGA-DAgger, HG-DAgger with and without data filter
- Perform experiments on two different maps
- Policies learned using MEGA-DAgger demonstrates better performance on both overtaking and collision avoidance!

Map 1 **Map 2**

Overtakes rate

Collisions rate



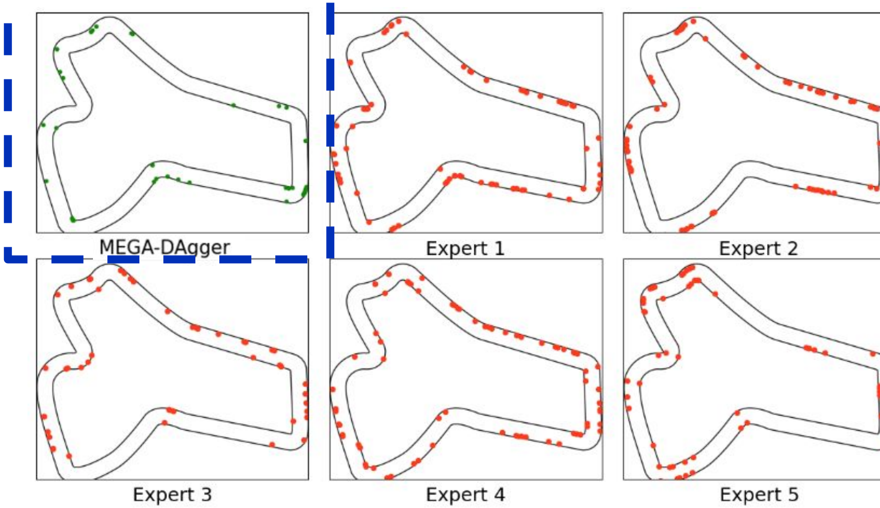
MEGA-DAgger
HG-DAgger with filter
HG-DAgger

Penn Engineering

Evaluation: Comparison with Experts

- Comparison: MEGA-Dagger and experts
- By leveraging the complementarily good demonstrations, policies learned using MEGA-Dagger is able to outperform all experts!

Collision points visualization



Metrics	MEGA-DAgger	Expert 1	Expert 2	Expert 3	Expert 4	Expert 5	Experts Cumulative
Collisions Percentage	$\mathbf{0.212 \pm 0.019}$	0.340 ± 0.025	0.401 ± 0.033	0.291 ± 0.025	0.392 ± 0.028	0.317 ± 0.032	0.348 ± 0.051
Overtakes Percentage	$\mathbf{0.781 \pm 0.016}$	0.657 ± 0.027	0.594 ± 0.036	0.706 ± 0.022	0.605 ± 0.024	0.681 ± 0.030	0.649 ± 0.051

Thank you

Source code: https://github.com/M4D-SC1ENTIST/MEGA-DAgger

Penn Engineering