Generative Model on Inverse Reinforcement Learning

Inverse Optimal Control via Langevin Sampling

Yifei Xu

Advisor: Ying Nian Wu

Department of Statistics

University of California, Los Angeles

March 12th, 2019

UCLA

Overview

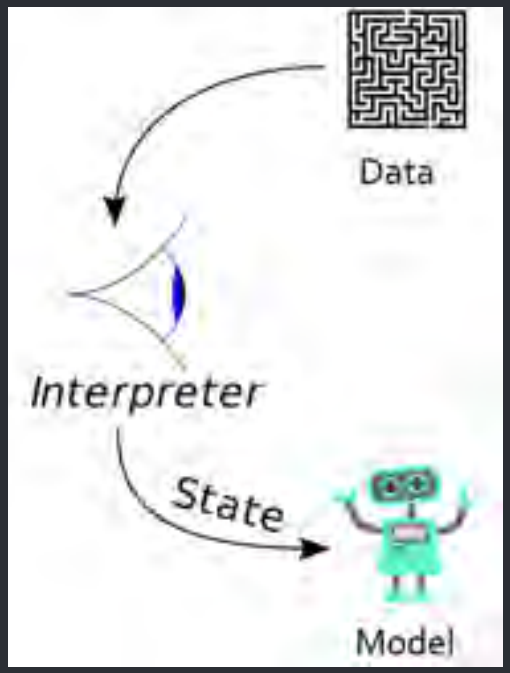
1. **Topic proposal**
   - (Inverse) Reinforcement Learning
   - Controling, Planning and Autonomous Driving
2. **Literature Review**
   - Energy-based Model
   - Maximum entropy IRL
   - Continuous Inverse Optimal Control
3. **Inverse Optimal Control via Langevin Sampling**
   - Sampling, Dynamic and cost function
   - Experiment Result
4. **Future Plan**
   - Multi-agent Model, Maneuver Conditioned Model
   - Refined Langevin Sampling, A Tale of Two Net

Outline for Section 1

1. Topic proposal
   - 1.1 (Inverse) Reinforcement Learning
   - 1.2 Controling, Planning and Autonomous Driving
2. Literature Review
   - 2.1 Energy-based Model
   - 2.2 Maximum entropy Inverse Reinforcement Learning
   - 2.3 Continuous Inverse Optimal Control
3. Inverse Optimal Control via Langevin Sampling
   - 3.1 Sampling, Dynamic and Cost Function
   - 3.2 Experiment Result
4. Future Plan
   - 4.1 Multi-agent Model, Maneuver Conditioned Model
   - 4.2 Refined Langevin Sampling, A Tale of Two Net

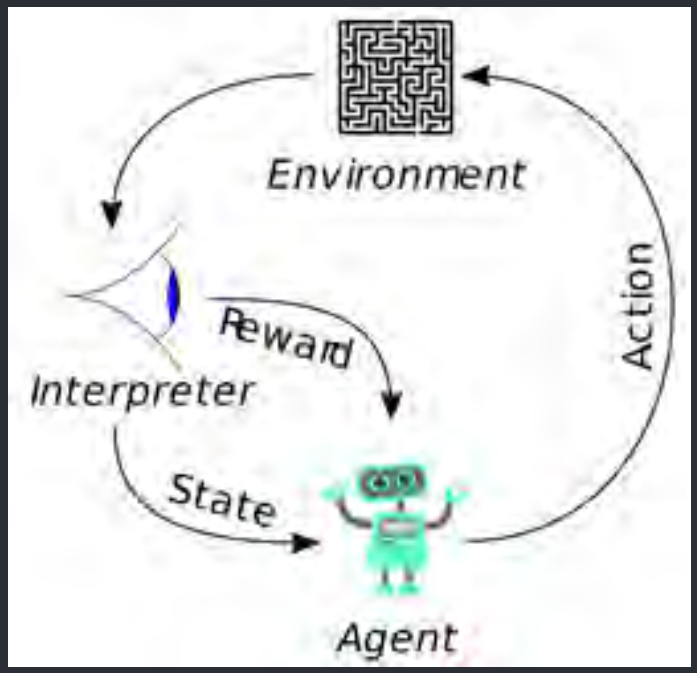
Reinforcement Learning

Regression / Classification



- Load from data
- Calculate loss
- Update model

Reinforcement Learning



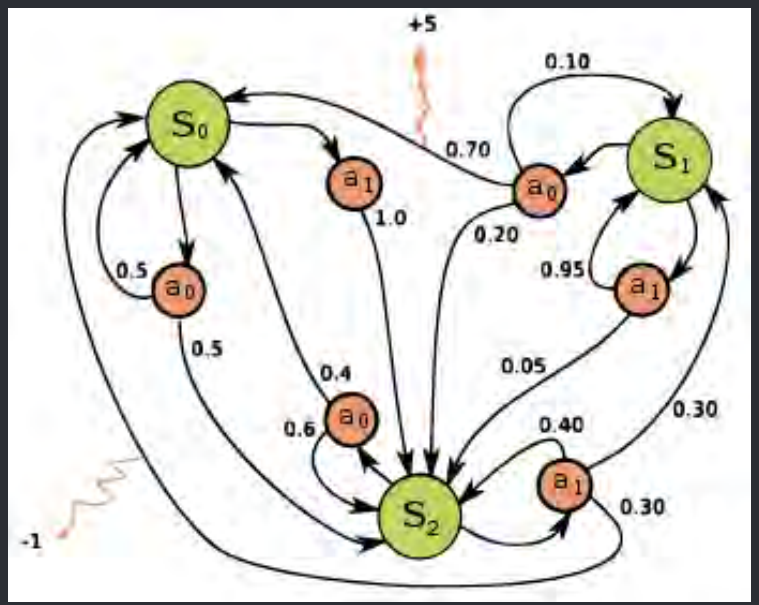
- Feed action from model
- receive feedback / reward
- update model

Reinforcement Learning

Markov Decision Process

$MDP =< X, U, D, C >$[1]

- X : State
- U : action (control)
- D : dynamic function : $x_{t+1} = D(x_t, u_t)$
- C : cost function (negative reward)



[1] Another notations widely used in RL is <S, A, P, R>

Reinforcement Learning

Typical Reward

Go

Win : very big

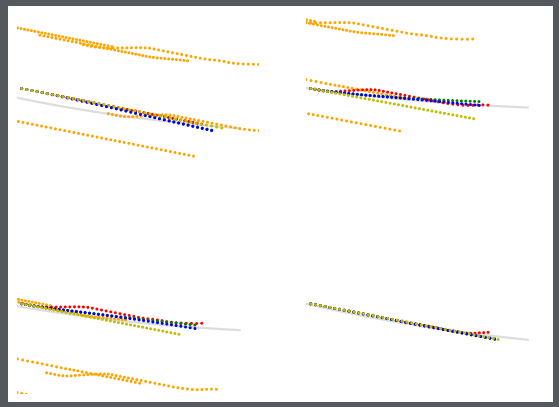
other : 0

Robotic

Ball in bin : big

Ball falls : -1

Autonomous



Collision : -1000

keep lane : 1

Reinforcement Learning

Policy Method

- Policy: $u_t = \pi(x_t)$: Given State, policy tell you the next action.

$$\pi^*(x_t) = \arg\max_{u_t} Q(x_t, u_t),$$

- Q-function : Long-term reward of specific state and action

$$Q(x_t, u_t) = \sum_{t}^{\infty} \gamma^t R(x_t, u_t),$$

- Value function : $V(x) = \max_u Q(x, u)$,
- Bellman Equations:

$$V(x_t) = \sum_{u_t} \Big[R(x_t, u_t) + \gamma V^{\pi}(D(x_t, u_t)) \Big].$$

Inverse Reinforcement Learning

- RL : Given full MDP, output a policy
- Inverse RL : Given part MDP and policy, output reward function

What if we do not know the policy?

- Learn from expert behavior
- Assume expert behavior has high reward
- Try to learn a reward system met the assumption

Controlling and planning

- Value function $V(x_t) = \sum_t^{\infty} \gamma^t R(x_t, u_t)$ is hard to approximate.
- **Controlling** : Find the best control sequence which have minimum cost
  - Not infinite long, typically next couple second.
  - Overall cost is sum of cost in each frame.
  - Sequence of control and its result is called trajectory.
- Controlling problem always have a given dynamic function.

Model-based & Model-free

Here, the model means we try to model an RL problem to an MDP.

Model-based

- We try to infer(or given) the whole MDP include dynamic / reward function.
- Based on that model, choose best policy.
- More explainable.
- Most model in controlling are model-based.

Model-free

- Part of MDP is unknown.
- Learn optimal policy in one step
- End-to-end learning.
- Very hard RL problem typically use model-free model.

Control problem in Autonomous Driving

- Goal : predict next control which has minimum future cost (typically in next 3 second)
- State x = (position, angle, velocity, last control)
- Control u = (next steering, next acceleration)
- Environment State x_{env} = (lane, other vehicle, speed limit, ...)
- Trajectory $\tau = (x_t, u_t)$ for t in 1:T
- Cost function : Avoid collision, keep in lane, Smooth turn, ...

Outline for Section 2

1. Topic proposal
   - 1.1 (Inverse) Reinforcement Learning
   - 1.2 Controling, Planning and Autonomous Driving
2. Literature Review
   - 2.1 Energy-based Model
   - 2.2 Maximum entropy Inverse Reinforcement Learning
   - 2.3 Continuous Inverse Optimal Control
3. Inverse Optimal Control via Langevin Sampling
   - 3.1 Sampling, Dynamic and Cost Function
   - 3.2 Experiment Result
4. Future Plan
   - 4.1 Multi-agent Model, Maneuver Conditioned Model
   - 4.2 Refined Langevin Sampling, A Tale of Two Net

Energy-based Model

Formulation

Assume the data distribution p are defined as

$$p(x;\theta) = \frac{1}{Z}\exp(-\mathrm{E}_\theta(x))q(x),$$

where q is the reference distribution of data, typically a Gaussian white noise distribution, $x\ N(0, \sigma^2 I_p)$.

$$Z = \int \exp(-\mathrm{E}_\theta(x))q(x)dx = E_q\left[\exp(-\mathrm{E}_\theta(x))\right],$$

is the normalization term, which attends to the constrain $\int p(x;\theta) = 1$.

Energy-based Model

Training

Fit model by maximum likelihood:

$$\theta = \arg\max_{\theta} L_p(\theta) \quad = \quad \frac{1}{n} \sum_{i=1}^{n} \log p(X_i; \theta),$$

Equivalent to minimize $\mathrm{KL}(P_{data} | p_\theta)$, P_{data} is the true distribution of data.

Use gradient descent to train the parameter θ

Energy-based Model

Examples

Image generation

Video generation

3D generation

[2] A Theory of Generative ConvNet

[3] Learning Dynamic Generator Model by Alternating Back-Prop Through Time

[4] Learning Descriptor Networks for 3D Shape Synthesis and Analysis

Maximum entropy Inverse Reinforcement Learning

What if the energy term become reward function?

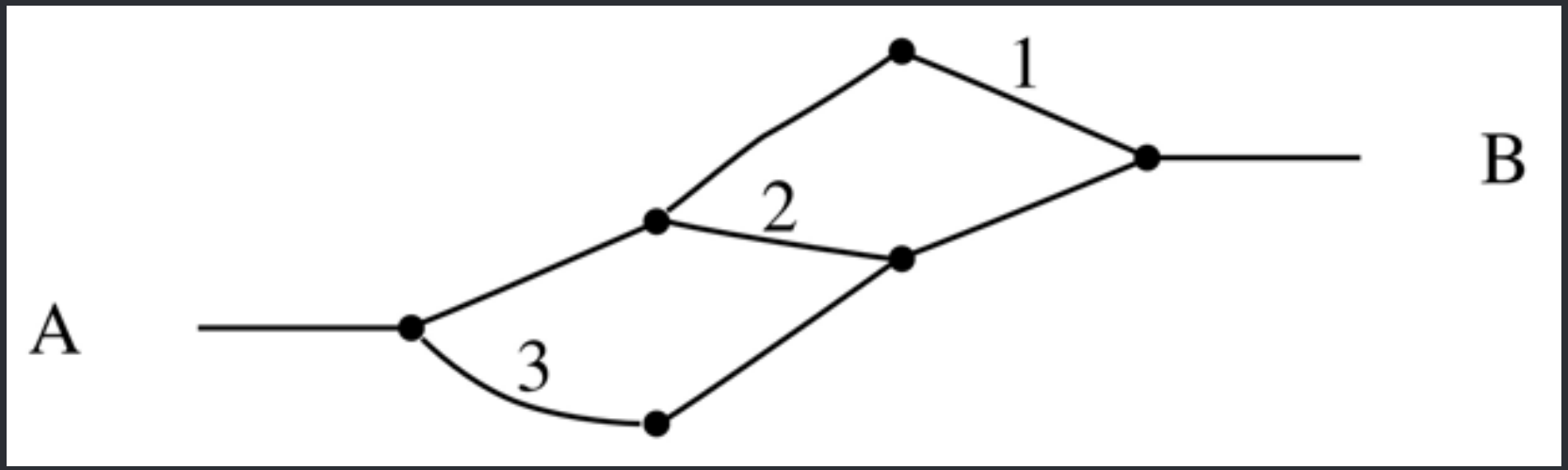
$$p(\tau;\theta) = \frac{1}{Z}\exp(-C_\theta(\tau)),$$

- The policy is to choose highest probability:
  $\tau^* = \arg\max_\tau P(\tau)$
- The training data is expert trajectory.
- Maximum likelihood = minimum cost for expert trajectory

Maximum entropy Inverse Reinforcement Learning

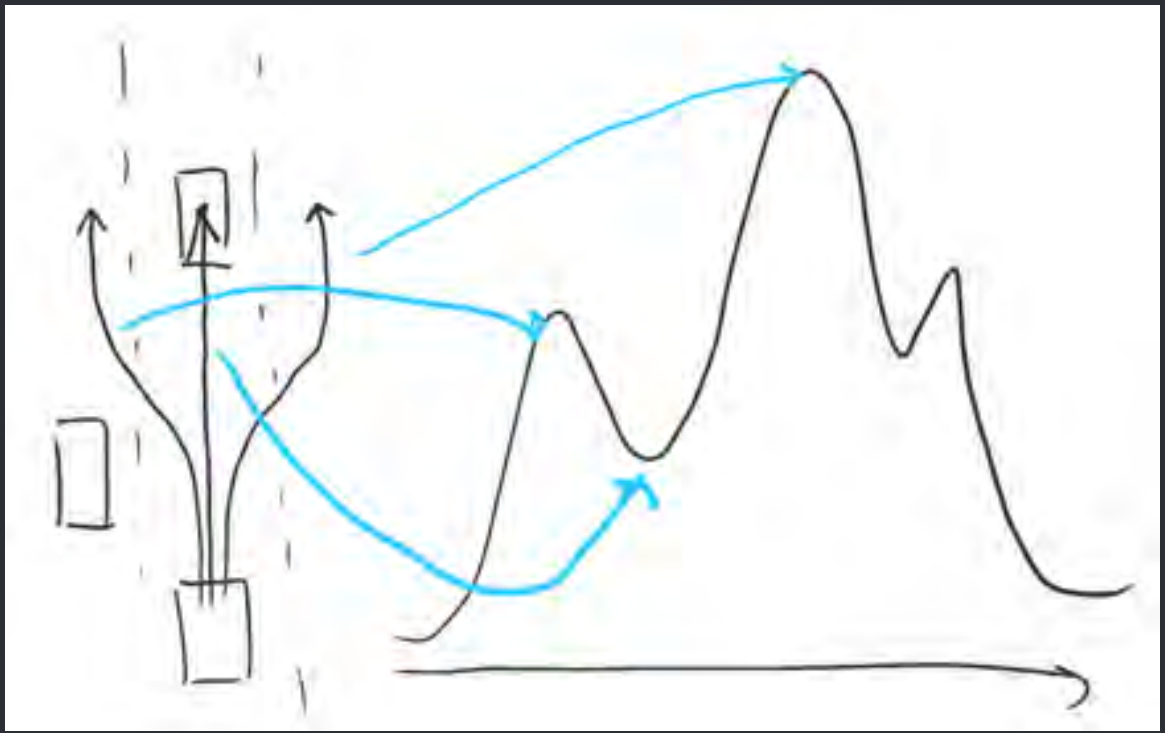
How to calculate Z?

This paper only support very small graph. They use dynamic programming to calculate all different trajectories



Maximum entropy Inverse Reinforcement Learning

If it become continuous?



Continuous Inverse Optimal Control

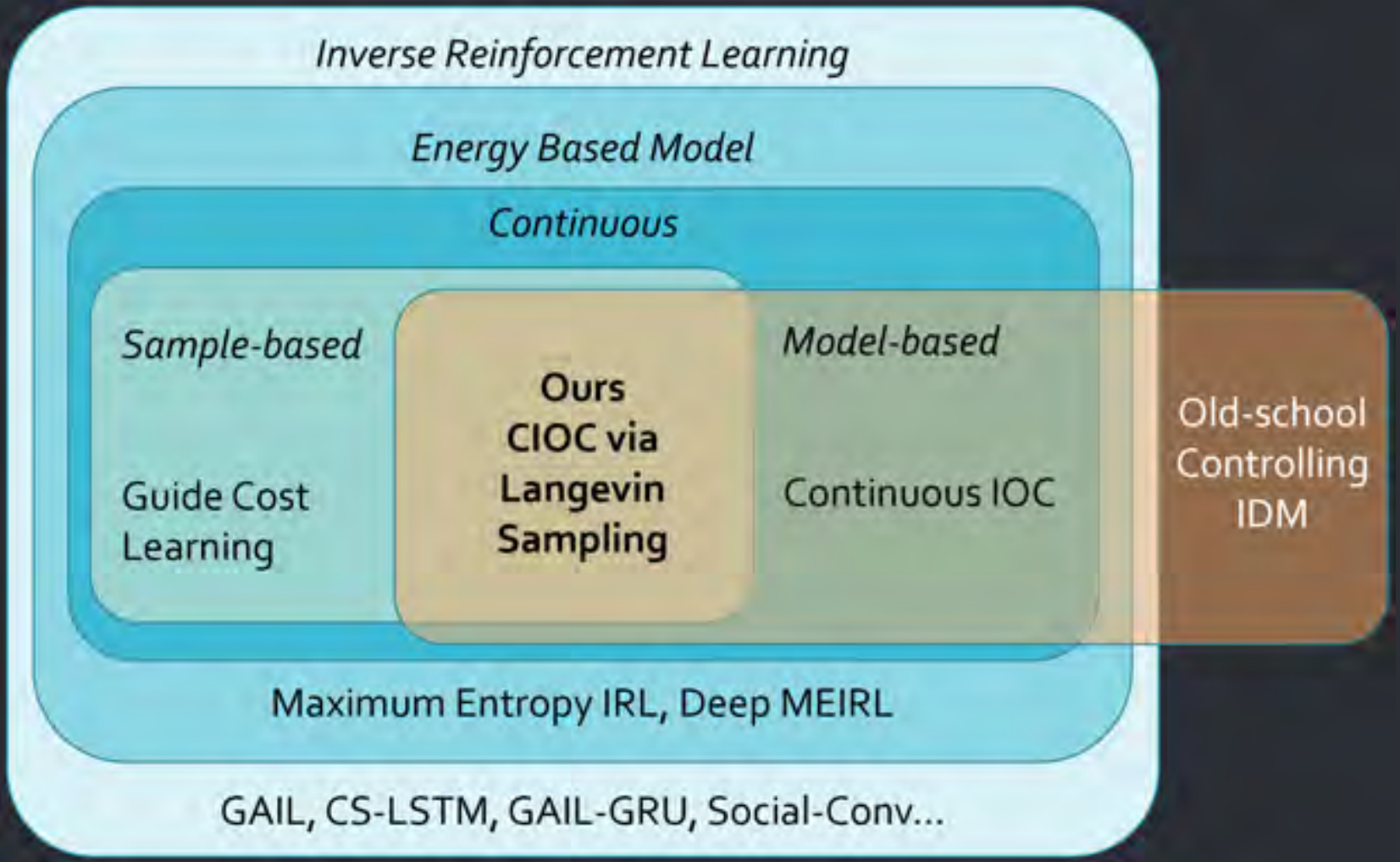
- Z is intractable on continuous and high-dim.
- Use Laplace approximation.
- Second order Taylor expansion.

$$Cost_\theta(\tilde{U}) \approx C(U) + (\tilde{U} - U)C_u + \frac{1}{2}(\tilde{U} - U)^2 C_{uu}$$

$$likelihood(\theta) \approx \frac{1}{2} C_U^T C_{UU}^{-1} C_U + \frac{1}{2} \log |- C_{UU}| - \frac{d_u}{\log}(2\pi),$$

where $c_U = \frac{\partial C_\theta}{\partial U}$, $c_{UU} = \frac{\partial^2 C_\theta}{\partial U^2}$.

Related Works



Contributions

- Introduce sample-based energy-based model to controlling
  - Avoid Laplace assumption (compare to CIOC)
- Introduce Langevin Sampling to sample data.
  - Can handle non-linear cost function (iLQR cannot)
  - Do not need to calculate second-order derivative (iLQR need)

Outline for Section 3

1. Topic proposal
   - 1.1 (Inverse) Reinforcement Learning
   - 1.2 Controling, Planning and Autonomous Driving
2. Literature Review
   - 2.1 Energy-based Model
   - 2.2 Maximum entropy Inverse Reinforcement Learning
   - 2.3 Continuous Inverse Optimal Control
3. Inverse Optimal Control via Langevin Sampling
   - 3.1 Sampling, Dynamic and Cost Function
   - 3.2 Experiment Result
4. Future Plan
   - 4.1 Multi-agent Model, Maneuver Conditioned Model
   - 4.2 Refined Langevin Sampling, A Tale of Two Net

Inverse Optimal Control via Langevin Sampling

Goal

Recall

$$P(\tau;\theta) = \frac{1}{Z}\exp(-c_\theta(\tau)),$$

The probability of taking a trajectory is small if the corresponding cost is big. The goal of IOC is to find a distribution that best fits the expert control. In other word, we maximize the log-likelihood on expert trajectories ($\tau_i \in Traj_{obs}$),

$$l(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log P(\tau_i;\theta) = \frac{1}{n}\sum_{i=1}^{n} \left(-c_\theta(\tau_i) - log(Z)\right).$$

Inverse Optimal Control via Langevin Sampling

Learning Algorithm

Sample-based approach : the gradient is,

$$\frac{\partial}{\partial \theta} l(\theta) = \frac{1}{n} \sum \frac{\partial}{\partial \theta} - c_\theta(\tau_i) - E_{P(\tau;\theta)}[\frac{\partial}{\partial \theta} - c_\theta(\tau_i)],$$

because

$$\begin{aligned}
\frac{\partial}{\partial \theta} \log(Z) &= \frac{1}{Z} \frac{\partial}{\partial \theta} \int \exp(-c_\theta(\tau)) d\tau \\
&= \int \frac{1}{Z} \exp(-c_\theta(\tau)) \frac{\partial}{\partial \theta} - c_\theta(\tau) d\tau \\
&= \int \frac{\partial}{\partial \theta} - c_\theta(\tau) P(\tau;\theta) d\tau \\
&= E_{P(\tau;\theta)}[\frac{\partial}{\partial \theta} - c_\theta(\tau_i)].
\end{aligned}$$

Inverse Optimal Control via Langevin Sampling

Learning Algorithm

We approximate the expectation term by sampling,

$$\frac{\partial}{\partial \theta} l(\theta) = \frac{1}{\tilde{n}} \sum \frac{\partial}{\partial \theta} c_\theta(\tilde{\tau}_i) - \frac{1}{n} \sum \frac{\partial}{\partial \theta} c_\theta(\tau_i),$$

where $\tilde{\tau}$ is the sampled trajectories and $\tilde{n}$ is the number of samples.

Inverse Optimal Control via Langevin Sampling

Sampling Algorithm

The state in our model can be divided into two parts, vehicle status x_v and environment x_e. Modifying control affects the status but not the environment. For each expert trajectory, we synthesize one trajectory based on the associated environment. In other words, we sample from the conditional distribution with fixed environment and maximize the conditional likelihood,

$$l(\theta) = \frac{1}{n} \sum_{i=1}^{N} \log P(\tau_i | X_e = x_e; \theta).$$

Our sampling algorithm only updates the control, which leads to a change in status x_v.

Sampling Method

Langevin Dynamic

The iterative process for Langevin Sampling is

$$U_{\tau+1} = U_\tau - \frac{\delta^2}{2}\left[\frac{U_\tau}{\omega^2} - \frac{\partial}{\partial U}C_\theta(U_\tau)\right] + \delta Noise_\tau.$$

Notice that the state changes as the control is changed; at the same time, the change of control in the previous frame affects each cost later. Thus the derivative is calculated by chain rule,

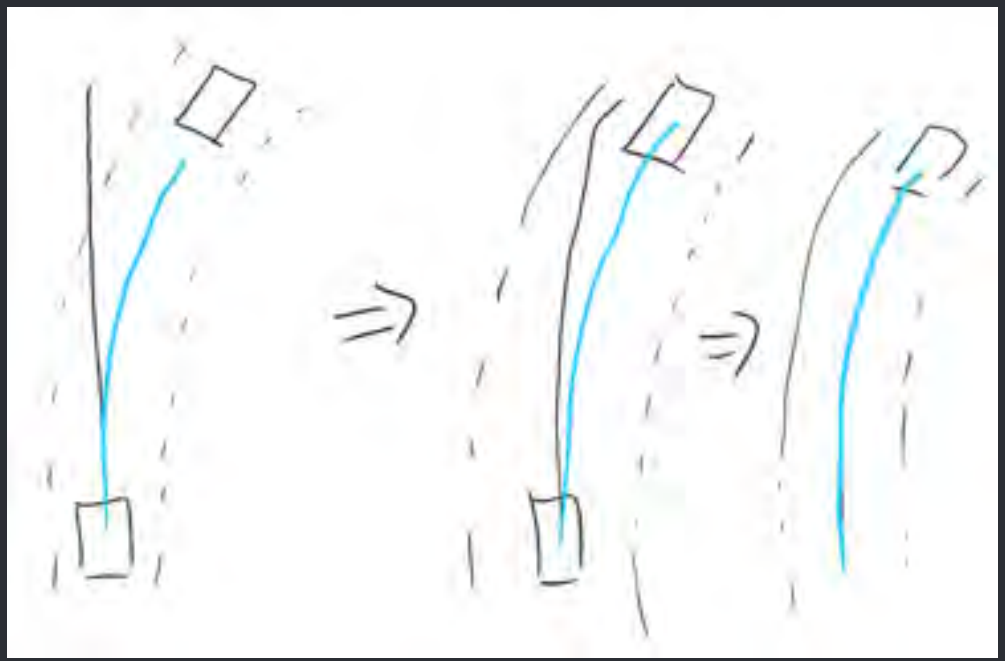
$$\frac{\partial}{\partial u_i}C_\theta(U_\tau) = \sum_{i=t}^{T}\frac{\partial C_i}{\partial u_t} = \sum_{i=t}^{T}\frac{\partial C_i}{\partial x_i}\frac{\partial x_t}{\partial u_t}\prod_{j=t}^{i-1}\frac{\partial x_{j+1}}{\partial x_j} + \frac{\partial C_t}{\partial u_t}.$$

Sampling Method

Langevin Dynamic

The iterative process for Langevin Sampling is

$$U_{\tau+1} = U_\tau - \frac{\delta^2}{2}\left[\frac{U_\tau}{\omega^2} - \frac{\partial}{\partial U}C_\theta(U_\tau)\right] + \delta Noise_\tau.$$



Sampling Method

Iterative Linear Quadratic Regulation

Given an initial trajectory, it updates the trajectory by repeatedly solving for the optimal policy under linear quadratic assumptions. Let (x_t^i, u_t^i) be the i-th iteration trajectory. The dynamic is known, $x_{t+1}^i = f(x_t^i, u_t^i)$. Define $\Delta x_t = x_{t+1} - x_t$, $\Delta u_t = u_{t+1} - u_t$, then,

$$\begin{aligned} \Delta x_{t+1} &\approx f_{x_t} \Delta x_t + f_{u_t} \Delta u_t \\ C_\theta(x_t, u_t) &\approx \Delta x_t c_{x_t} + \Delta u_t c_{u_t} + \frac{1}{2} \Delta x_t c_{x x_t} \Delta x_t \\ &+ \frac{1}{2} \Delta u_t c_{u_t} \Delta u_t + \Delta u_t c_{u x_t} \Delta x_t + C_\theta(x_{t-1}, u_{t-1}). \end{aligned}$$

where the subscripts denote the Jacobians and Hessians of the dynamic f and cost function C.

Inverse Optimal Control via Langevin Sampling

Min-max Interpretation

$$V_\theta(\tilde{\tau}) = \frac{1}{n}\sum E_\theta(\tau_i) - \frac{1}{\tilde{n}}\sum E_\theta(\tilde{\tau}_i),$$

- **Mode Shifting** Update θ Shifts the low energy mode from the current trajectories $\{\tilde{\tau}_i\}$ towards the expert trajectories $\{\tau_i\}$.
- **Mode Seeking** Resample $\tilde{\tau}$ Seek the minimum mode in the distribution.

As a result, our training process is

$$\theta = \arg\min_\theta \max_{\tilde{\tau}} V_\theta(\tilde{\tau}).$$

Inverse Optimal Control via Langevin Sampling

Cost function

- Lane Keeping cost
  - The distance to the center of the lane.
  - The heading angle to lane.
- collision cost
  - The penalty to collision to other vehicle. It is inversely proportional to distance to other vehicle.
- Smooth cost
  - The L2-norm of acceleration and steering
  - The L2-norm for difference of acceleration and steering between two frames.
  - The difference to speed limit.

Inverse Optimal Control via Langevin Sampling

Cost function

cost function for a trajectory is defined as the sum of the cost for each frame with a state-control pair,

$$Cost_\theta(\tau) = \sum_{(x,u)\in\tau} Cost_\theta(x, u).$$

Linear version:

$$Cost_\theta(x, u) = \sum_{k=1}^{K} \theta_k f_k(x, u).$$

where $f_k(x, u)$ is hand crafted based on human expertise.

Inverse Optimal Control via Langevin Sampling

Cost function: NN argmented

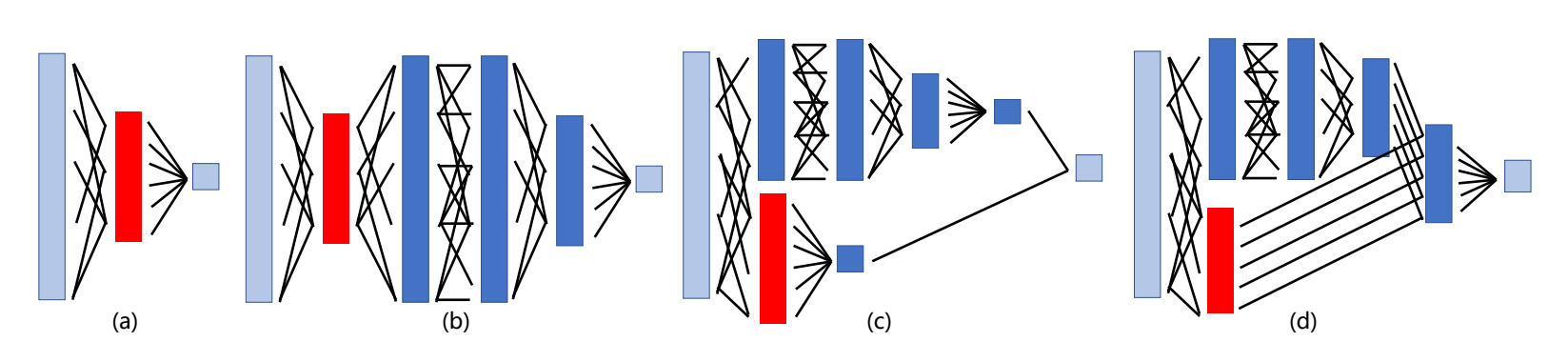
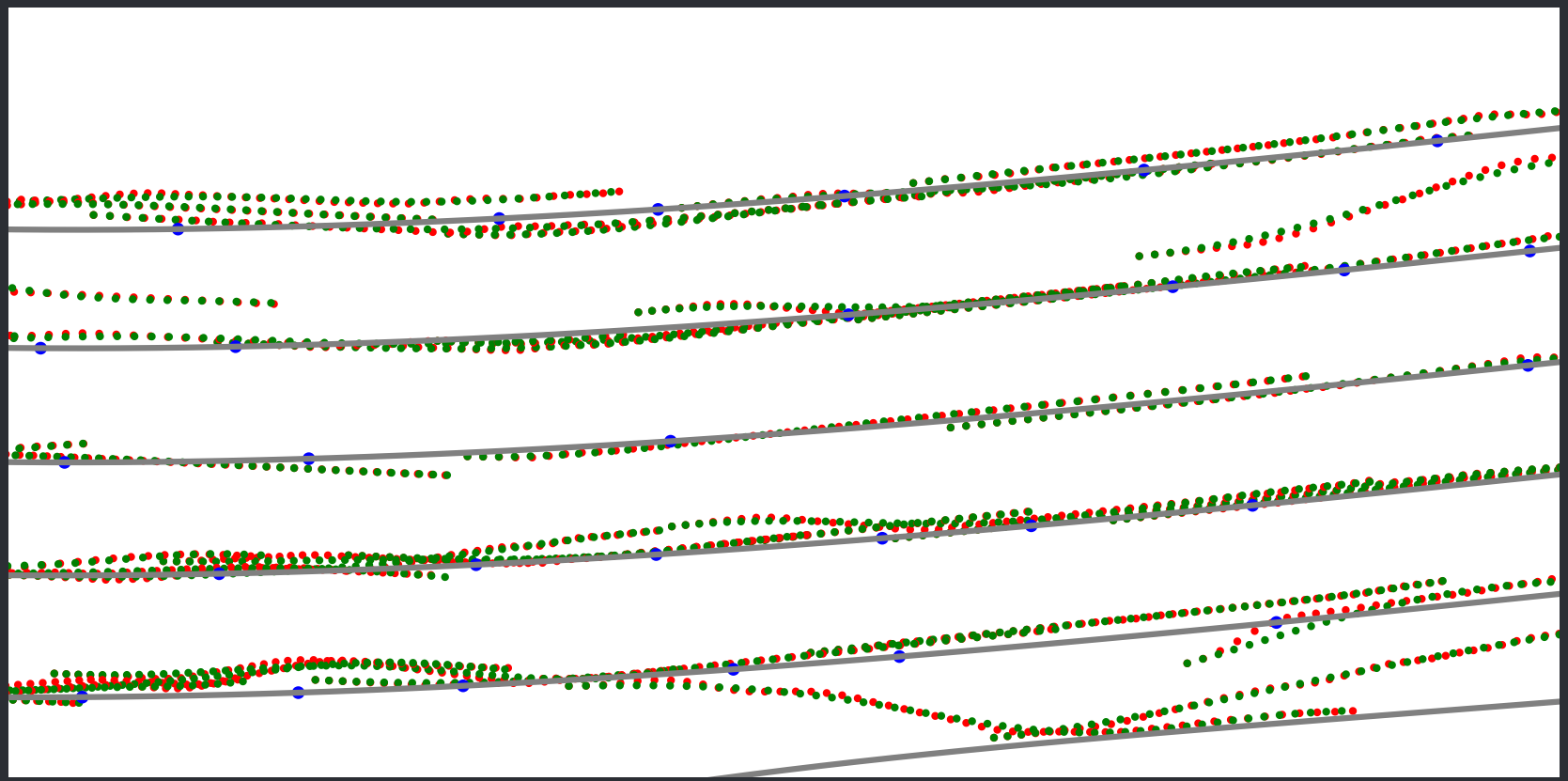


Figure: (a) hand-crafted Cost (b) nn as transformer (c) NN as residual (d) NN as residual to each.

Experiment Result

Experiment : Prediction



- 10Hz over a time span of 45 minutes.
- 831 total scenes with 96,000 5-second vehicle trajectories.
- Control is inferred by bicycle dynamic model.

Experiment Result

Comparison Metric

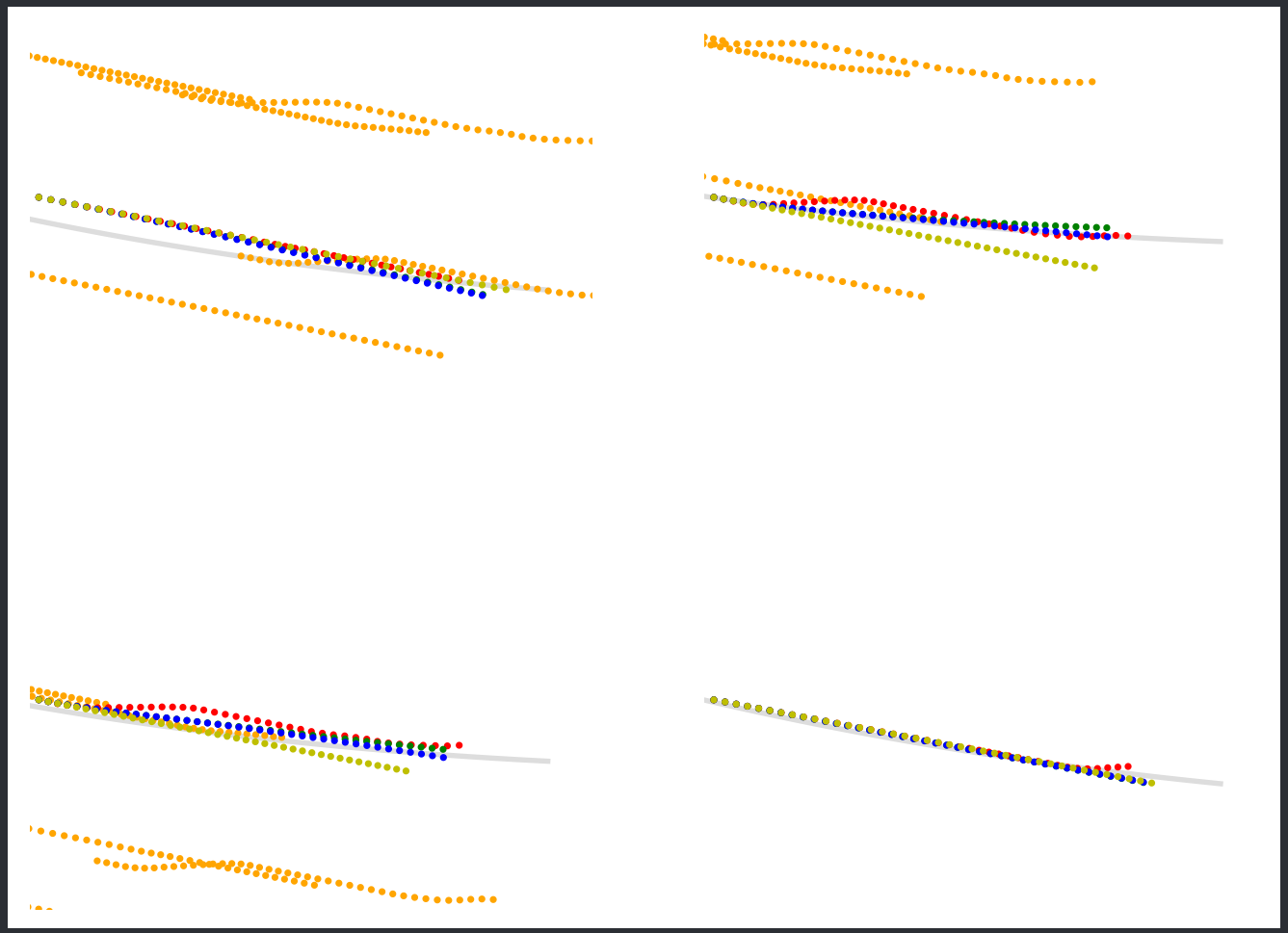
Rooted mean square error (RMSE) for i-th position is defined as:

$$
\begin{aligned}
RMSE &= \sqrt{\frac{1}{N}\sum_{k} Err(\tau^{k}_{pre'}, \tau^{k}_{obs'}, i)} \\
&= \sqrt{\frac{1}{N}\sum_{k} (x^{k}_{pre,1} - x^{k}_{obs,1})^2 + (x^{k}_{pre,2} - x^{k}_{obs,2})^2}.
\end{aligned}
$$

A small RMSE is desired.

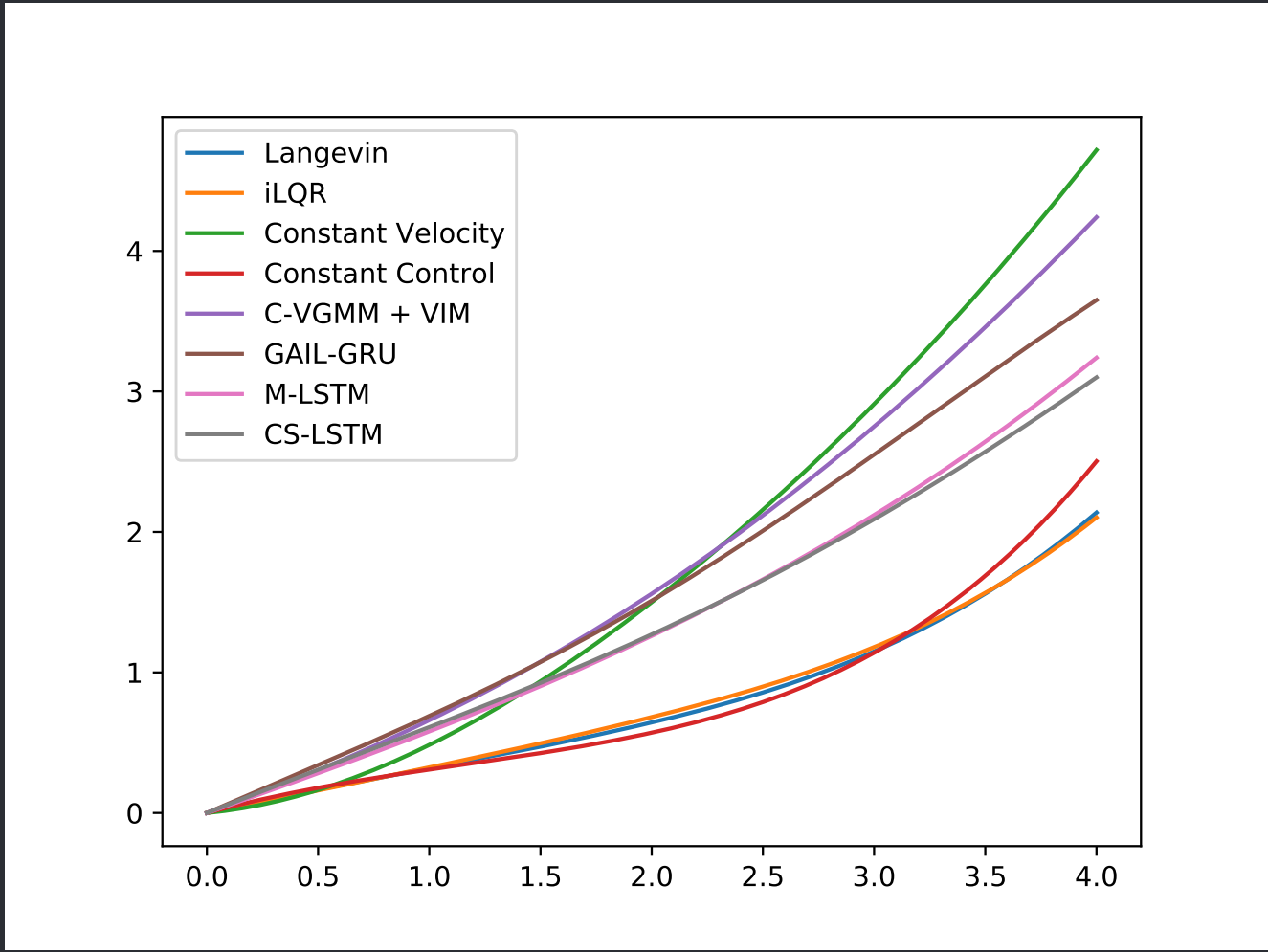
Experiment Result

Predicted Trajectory



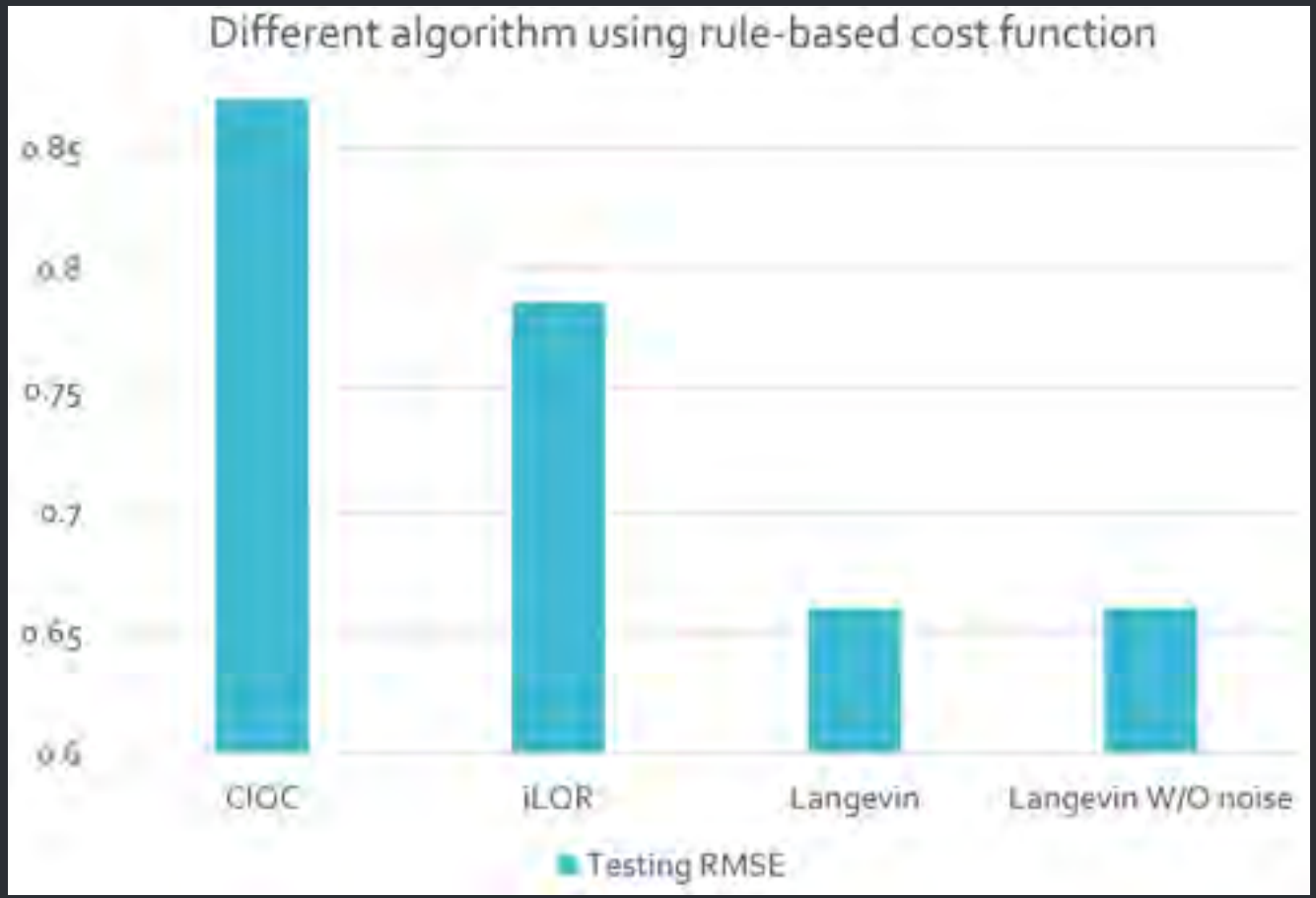
Experiment Result

RMSE Comparison



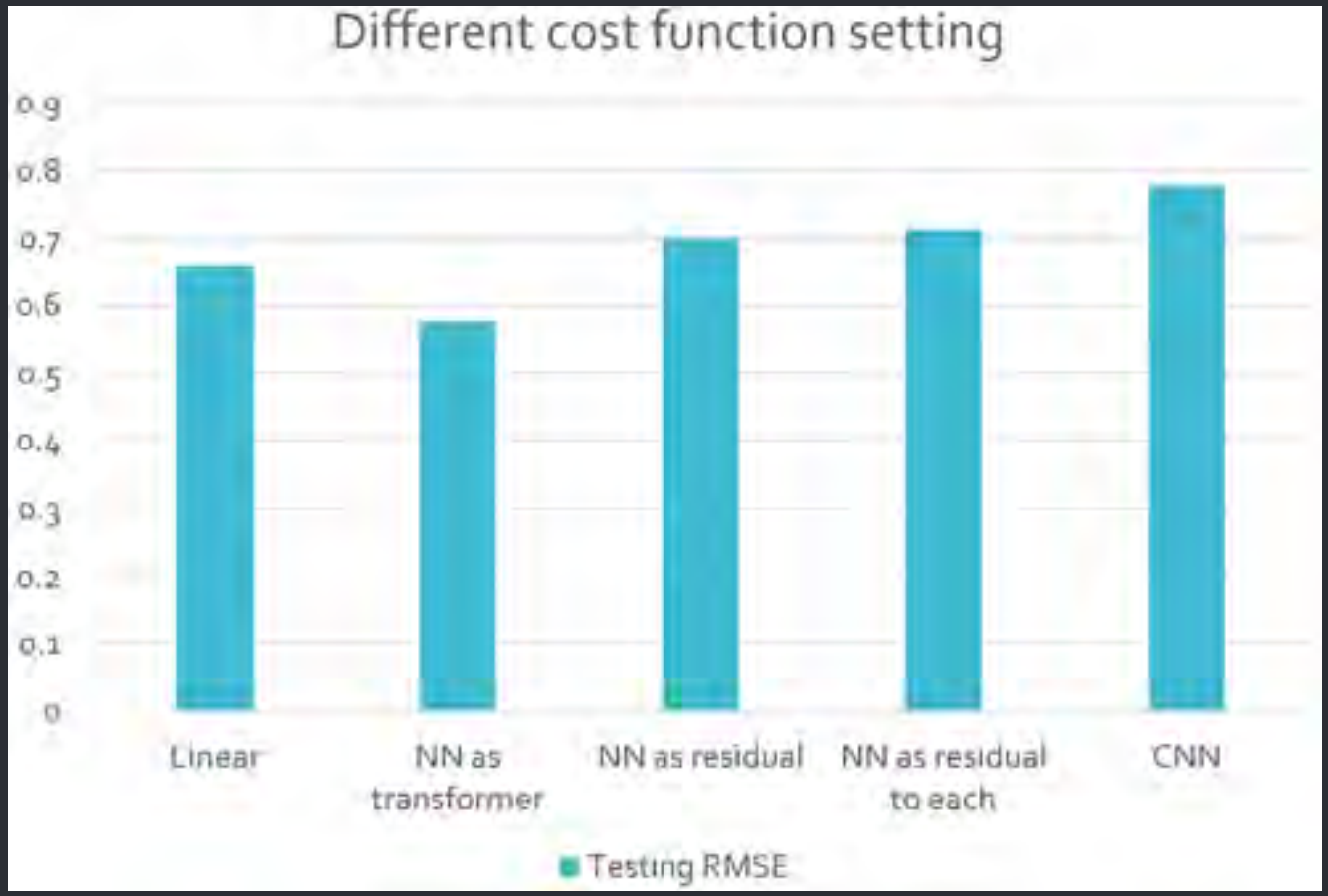
Experiment Result

Autonomous Driving Dataset : Comparing to CIOC



Experiment Result

Autonomous Driving Dataset : Comparing between different cost function



Experiment Result

Synthetic Examples

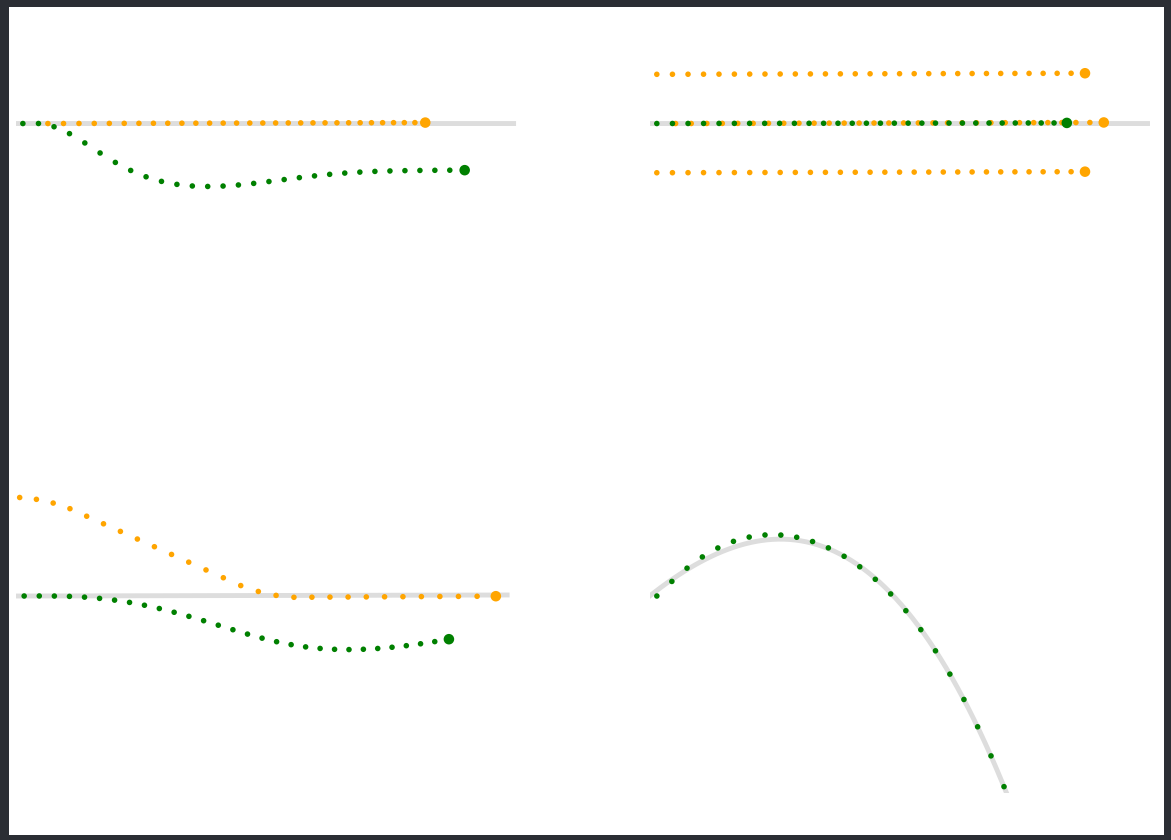


Figure: Predicted Trajectory for synthetic examples.

Conclusion

- Results shows the introduction of control greatly improves prediction.
  - Dynamic model is an important piror knowledge.
- Langevin Sampling is capable of handling non-linear cost function.
  - Langevin Sampling tend to infer smooth trajectories. (Both pro / con)
  - It do well in autonomous driving.
- Synthetic examples show good prediction on corner case.
  - We can output each subcost and explain why we need to acc/dec.

Outline for Section 4

1. Topic proposal
   - 1.1 (Inverse) Reinforcement Learning
   - 1.2 Controling, Planning and Autonomous Driving
2. Literature Review
   - 2.1 Energy-based Model
   - 2.2 Maximum entropy Inverse Reinforcement Learning
   - 2.3 Continuous Inverse Optimal Control
3. Inverse Optimal Control via Langevin Sampling
   - 3.1 Sampling, Dynamic and Cost Function
   - 3.2 Experiment Result
4. Future Plan
   - 4.1 Multi-agent Model, Maneuver Conditioned Model
   - 4.2 Refined Langevin Sampling, A Tale of Two Net

Multi-Agent Model [5]

Calculate the joint trajectory distribution for all moving agents, and sample multiple trajectories at the same time. Assume we have K agents and each of them has a trajectory τ_i, then,

$$P(\tau_1, \ldots, \tau_K; \theta) = \frac{1}{Z} exp\left(\sum_{i=1}^{K} C_\theta(\tau_i)\right).$$

The cost function of each agent shares the same parameters. Notice that the cost function for one vehicle is dependent on the information on the others.

[5] Multi-Agent Generative Adversarial Imitation Learning

Multi-Agent Model

Sampling multiple trajectory simultaneously may be a little hard. We can update it through one by one like "Cooderdinate descent".
For each scene, for i = 1:K,
When updating trajectory i, a sample from conditioned distribution:

$$P(\tau_i | \tau_{-i}; \theta).$$

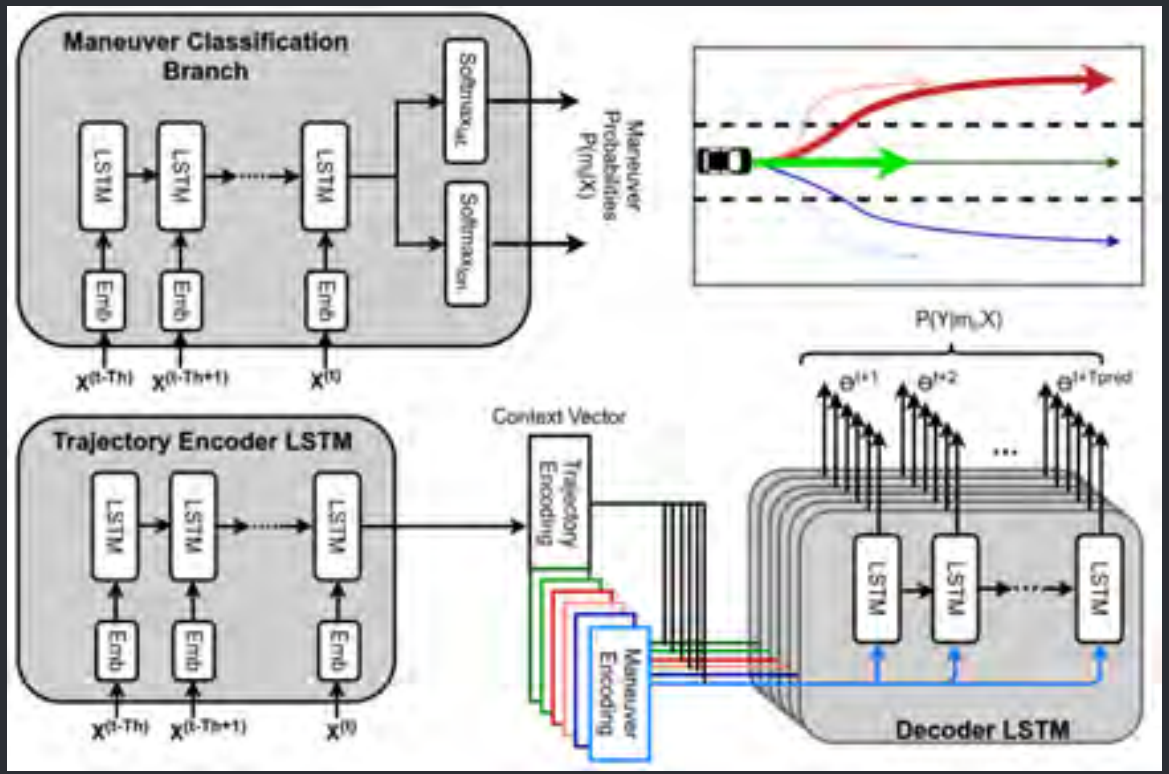
Multi-Agent Model

Cooperative and competitive

- **Cooperative** : $Cost = \sum_{i=1}^{K} C_\theta(\tau_i)$ Every agent want to minimum the overall cost.
- **Competitive** : $Cost = C_\theta(\tau_i) - \sum_{-i} C_\theta(\tau_i)$ Each car want to beat other vehicle, minimum self cost and maximum others' cost.
- **Nash equilibrium** : Every car will reach their nash equilibrium.

Maneuver Conditioned Model

- Hard to predict lane changing –> Introduce Maneuver



[6]

[6] Multi-Modal Trajectory Prediction of Surrounding Vehicles with Maneuver based LSTMs

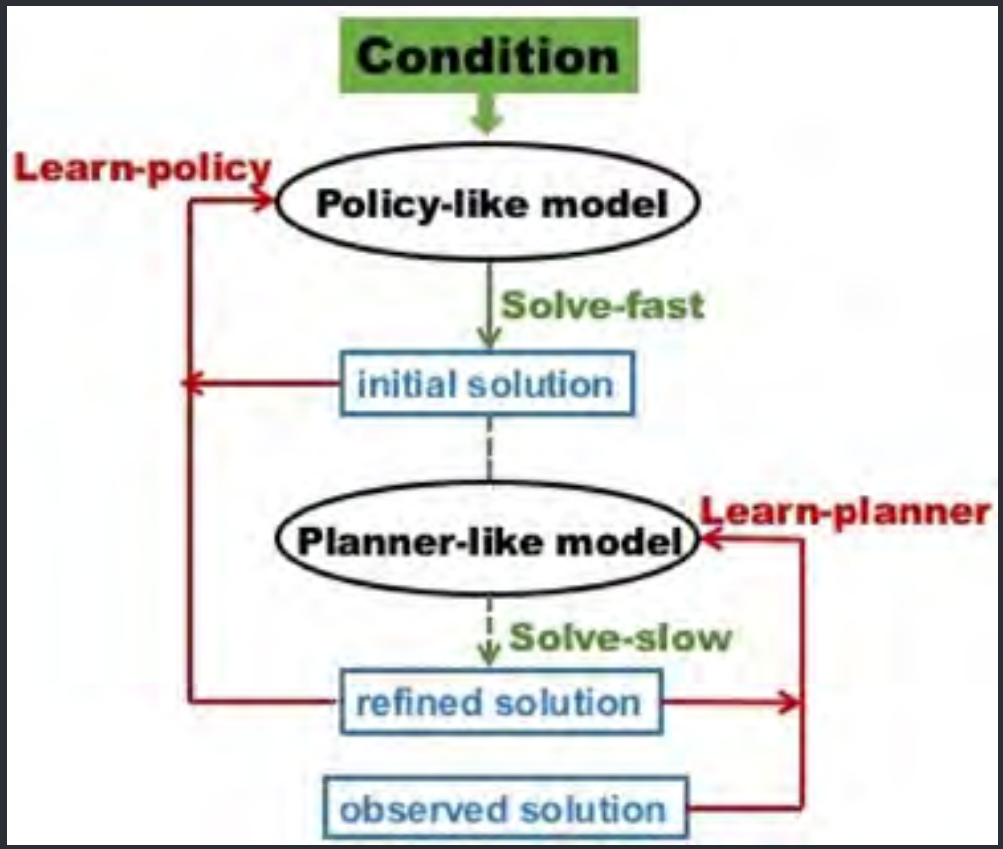
Refined Langevin Sampling

- iLQR : Find minimum point, need second order, iteratively update
- Langevin : Sample distribution, only 1st order needed, multi-step needed

-> Try to refine the Langevin Sampling by introducing second order message.[7]

[7] A Function Space HMC Algorithm With Second Order Langevin Diffusion Limit

A Tale of Two Net



Use fast thinking as an initializer and use slow thinking to find the optimal solution. [8]

[8] Multimodal Conditional Learning with Fast Thinking Policy-like Model and Slow Thinking Planner-like Model

Reference

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. 2016. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition.
[2] Georges S. Aoude, Brandon D. Luders, Joshua M. Joseph, Nicholas Roy, and Jonathan P. How. 2013. Probabilistically Safe Motion Planning to Avoid Dynamic Obstacles with Uncertain Motion Patterns. Auton. Robots 35, 1 (July 2013), 51–76. DOI:http://dx.doi.org/10.1007/s10514-013-9334-3
[3] Saurabh Arora and Prashant Doshi. 2018. A Survey of Inverse Reinforcement Learning: Challenges, Methods and Progress. arXiv preprint arXiv:1806.06877 (2018).
[4] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N Pistikopoulos. 2002. The explicit linear quadratic regulator for constrained systems. Automatica 38, 1 (2002), 3–20.
[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. 2016. End to End Learning for Self-Driving Cars. CoRR abs/1604.07316 (2016). http://arxiv.org/abs/1604.07316
[6] Lu Chi and Yadong Mu. 2017. Deep steering: Learning end-to-end driving model from spatial and temporal visual cues. arXiv preprint arXiv:1708.03798 (2017).
[7] J. Colyar and J. Halkias. 2007. US highway dataset. Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030 (2007).
[8] Nachiket Deo, Akshay Rangesh, and Mohan M Trivedi. 2018. How Would Surround Vehicles Move? A Unified Framework for Maneuver Classification and Motion Prediction. IEEE Transactions on Intelligent Vehicles 3, 2 (2018), 129–140.
[9] Nachiket Deo and Mohan M Trivedi. 2018. Convolutional Social Pooling for Vehicle Trajectory Prediction. arXiv preprint arXiv:1805.06771 (2018).
[10] Nachiket Deo and Mohan M Trivedi. 2018. Multi-Modal Trajectory Prediction of Surrounding Vehicles with Maneuver based LSTMs. arXiv preprint arXiv:1805.05499 (2018).
[11] Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. 2016. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. arXiv preprint arXiv:1611.03852 (2016).
[12] Chelsea Finn, Sergey Levine, and Pieter Abbeel. 2016. Guided cost learning: Deep inverse optimal control via policy optimization. In International Conference on Machine Learning. 49–58.

Thank you :) Question?

1. Topic proposal
   - 1.1 (Inverse) Reinforcement Learning
   - 1.2 Controling, Planning and Autonomous Driving
2. Literature Review
   - 2.1 Energy-based Model
   - 2.2 Maximum entropy Inverse Reinforcement Learning
   - 2.3 Continuous Inverse Optimal Control
3. Inverse Optimal Control via Langevin Sampling
   - 3.1 Sampling, Dynamic and Cost Function
   - 3.2 Experiment Result
4. Future Plan
   - 4.1 Multi-agent Model, Maneuver Conditioned Model
   - 4.2 Refined Langevin Sampling, A Tale of Two Net