Partially Observable Markov Decision Process (POMDP)

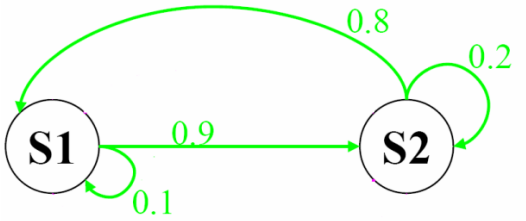
What is a Markov Property

- Finite number of discrete states (S_t)
- Probabilistic transitions between states (P)
- Next state determined only by the current state
- This is the Markov property

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, \ldots, S_t]$$

$$\boldsymbol{P}_{ss'} = P[S_{t+1} = s'|S_t = s]$$

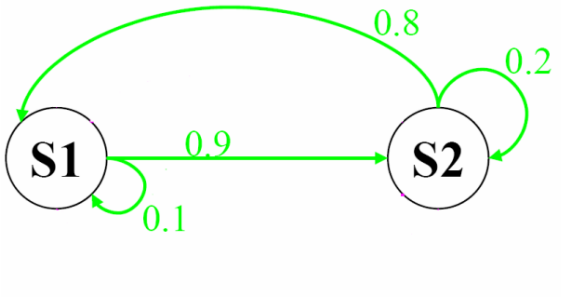
$$\boldsymbol{P} = \begin{bmatrix} P_{11} & P_{12} & \ldots & P_{1n} \\ P_{21} & P_{22} & \ldots & P_{2n} \\ \ldots & & & \\ \ldots & & & \\ P_{n1} & P_{n2} & \ldots & P_{nn} \end{bmatrix}$$



Rewards: S1 = 10, S2 = 0

What is a Markov Chain

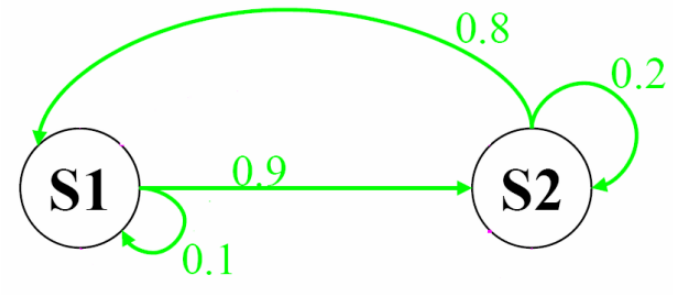
- Markov Chain, including $< S, P >$
- S: Finite number of discrete states;
- P: Probabilistic transitions between states;
- Markov property;
- P is not changing by time;



Rewards: S1 = 10, S2 = 0

What is a Hidden Markov Model?

- Finite number of discrete states
- Probabilistic transitions between states
- Next state determined only by the current state
- ***We're unsure which state we're in***
- **The current states emits an observation**



Rewards: S1 = 10, S2 = 0
Do not know state:
S1 emits O1 with prob 0.75
S2 emits O2 with prob 0.75

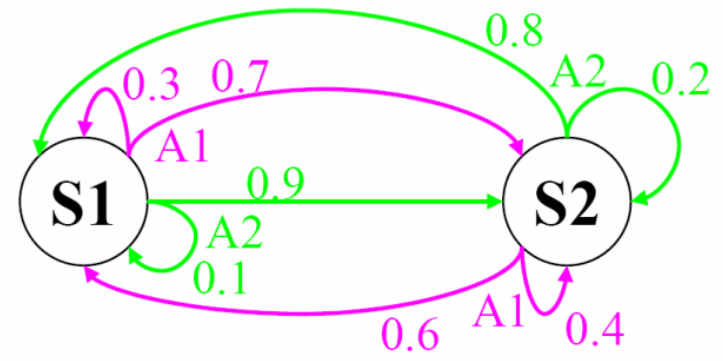
What is a Markov Decision Process?

- Finite number of discrete states (S_t)
- Probabilistic transitions between states and controllable actions in each state
- This is s Next state determined only by the current state and current action till the Markov property.

$$P(S_{t+1}|S_t, A_t)$$

- Reward:

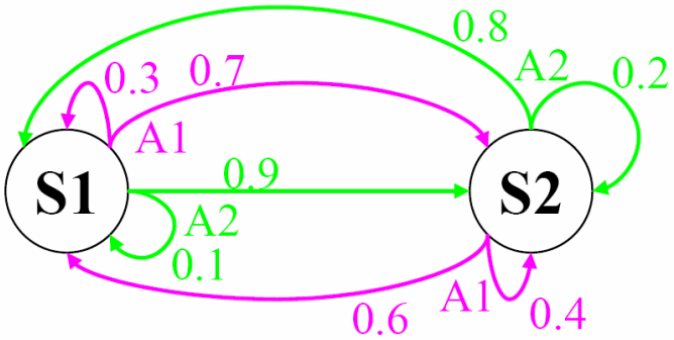
$$P(R_t|S_t, A_t)$$



Rewards: S1 = 10, S2 = 0

What is a Partially Observable Markov Decision Process?

- Finite number of discrete states
- Probabilistic transitions between states and controllable actions
- Next state determined only by the current state and current action
- We're unsure which state we're in
  - The current state emits observations



Rewards: S1 = 10, S2 = 0
Do not know state:
S1 emits O1 with prob 0.75
S2 emits O2 with prob 0.75

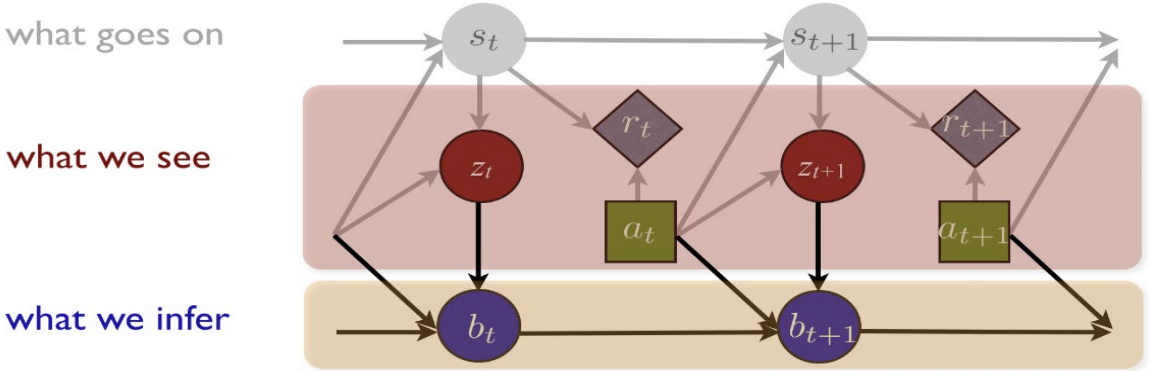
A very helpful chart

Markov Models		Do we have control over the state transitions?	
		No	Yes
Are the states completely observable?	Yes	Markov Chain	MDP Markov Decision Process
	No	HMM Hidden Markov Model	POMDP Partially Observable Markov Decision Process

Partially Observable Markov Decision Process (POMDP)

Solves for an optimal policy by sampling in **continuous** belief space (b = p(s), prob. distribution)

- Expensive, susceptible to “Curse of Dimensionality” ($|A|^h|Z|^h$)
  - Limit to small size discrete action space (|A|) and hidden state space (observed by |Z|)
  - Limit to small number of discrete time steps (h, h = 1 step = 1 action/obs, depth of policy search)
- Hidden states may include other agent intention, also influenced by ego action
  - Mixed Observability, some state attributes still fully observable (MOMDP)
- Active research area to improve efficiency of solvers (SARSOP, DESPOT, et al)



MDP in Autonomous Vehicle

MDP can be defined as a tuple:

$$(S, A, P, R, \gamma)$$

1. S represents the finite state space, i.e. it consists of lane, environment and world information
2. A represents vehicle decision space, i.e. the combination of the behavior space in all status, including car following, lane changing, turning left/right and stop.
3. $P(S_{t+1}|S_t) = P(S_{t+1}|S_t, A_t)$ is a conditional probability, representing the probability of reaching the next state with current state ***s*** and action ***a*** of the vehicle
4. $R(S_{t+1}|S_t, A_t)$ is a reward function, representing the reward to reach the next state s_{t+1} with current state S_t and action A_t of the vehicle
5. $\gamma \in (0,1)$ is a discount factor . i.e. if current reward coefficient is 1 and next sample is γ and then, in the following sample it becomes γ^2. And the future reward coefficient can be deduced in this way.

MDP in Autonomous Vehicle

Finding optimal strategy

- The vehicle decision making problem is essentially a problem of finding optimal strategy π, where *S → A.* The objective is to maximize the total reward from present.

  $\sum_{t=0}^{\infty} \gamma^t R(S_{t+1}|S_t)$, where action is determined by policy π: $a = \pi(s)$

- The solution can be found via ***Dynamic Programming***. Assuming the state transition probability matrix ***P*** and the reward function ***R*** are known, the process of finding the optimal solution is a process of iterating the following:

$$\pi(s_t) \leftarrow \underset{a}{\operatorname{argmax}} \left\{ \sum_{s_t+1} P(S_{t+1}|S_t) \left(R(S_{t+1}|S_t) + \gamma V(s_{t+1})\right) \right\}$$

$$V(s_t) \leftarrow \sum_{s_{t+1}} P_{\pi(s_t)}(S_{t+1}|S_t)(R_{\pi(s_t)}(S_{t+1}|S_t) + \gamma V(s_{t+1}))$$

where $V(s_t)$ represents accumulated discount in reward, $\pi(s_t)$ represents the strategy

The process of finding the optimal strategy is to iterate between s_t and s_{t+1} until they converge.

MDP in Autonomous Vehicle

Challenges in MDP

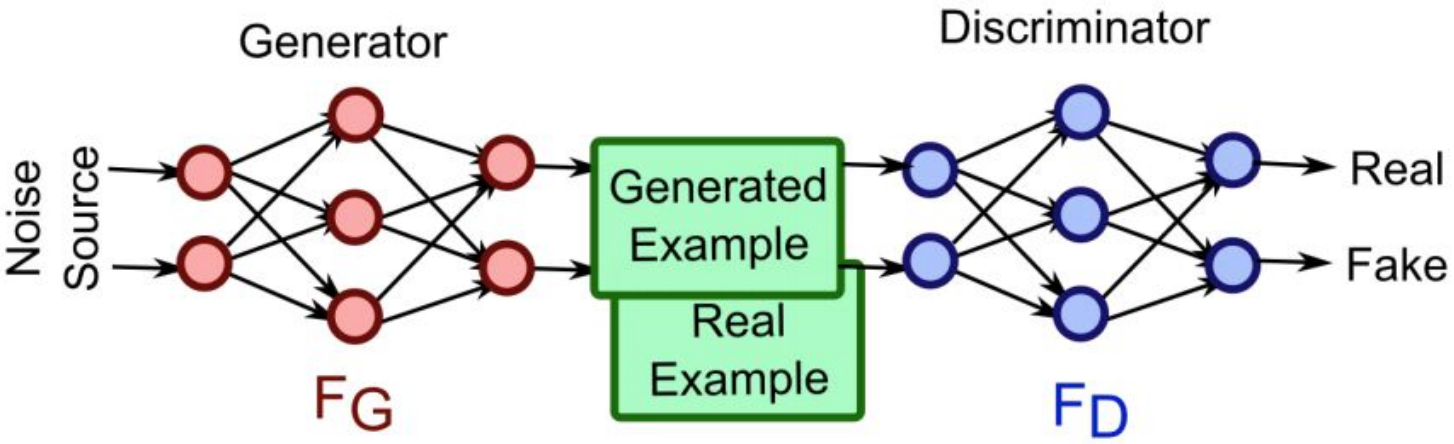
The key to find the optimal strategy is the selection of the Reward function. The following factors are to be considered when designing the Reward function:

1. Reaching the destination: Penalty needs to be given if the vehicle deviates from the given route
2. Safety: it should be rewarded when a trajectory is able to avoid obstacles
3. Comfort: smooth trajectories should be rewarded

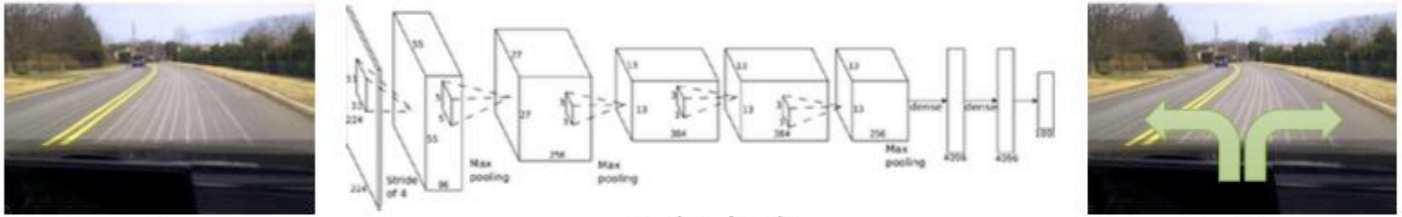
The design of MDP needs to be designed carefully, such as state space, state transition probability matrix, and reward function. It is not commonly used in industry. It’s a bit hard to find the relationship and reward function.

Imitation Learning

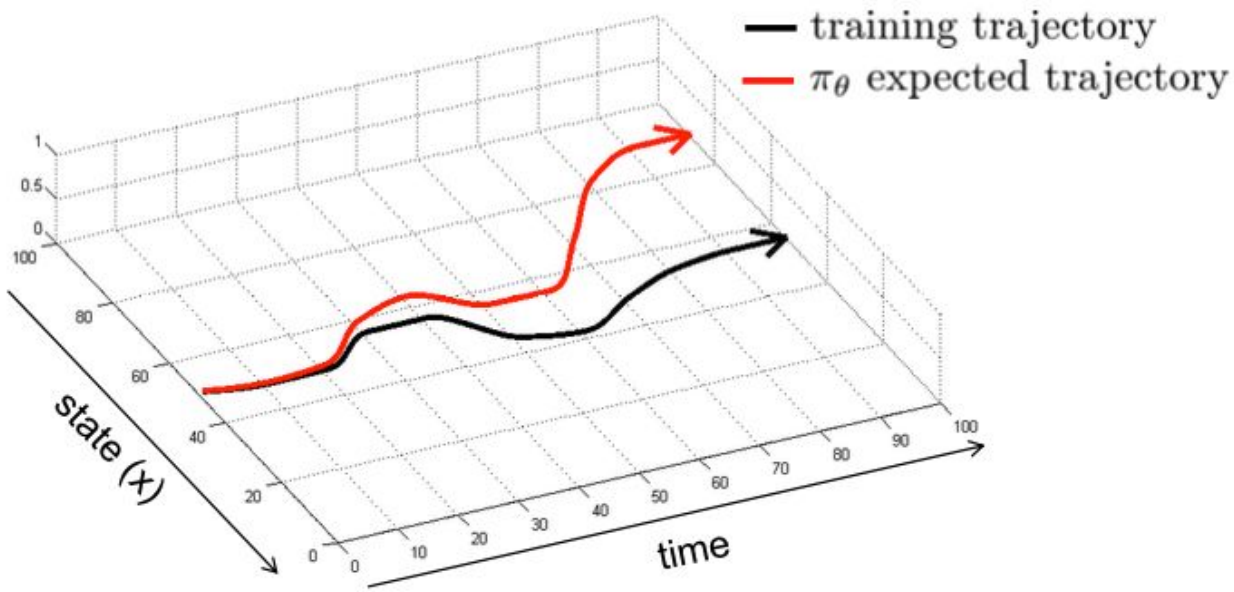
Imitation Learning



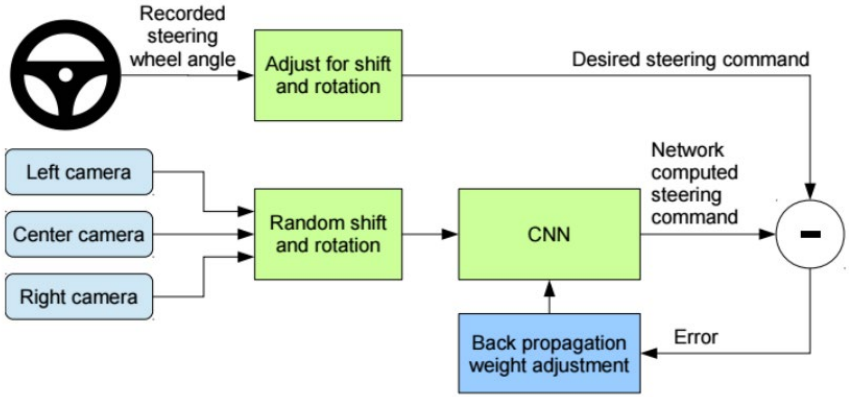
Imitation Learning

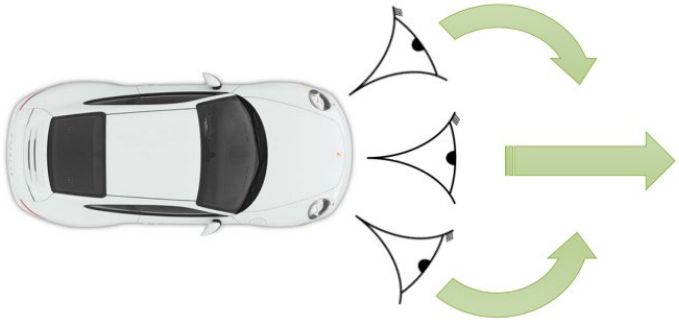


Imitation Learning



Imitation Learning: Data Augmentation





Imitation Learning: DAgger (Dataset Aggregation)

Dataset AGGregation: bring learner's and expert's trajectory distributions closer by iteratively labelling expert action for states generated by the current policy

1. Train $\pi_\theta(u_t|o_t)$ from human data $D_{\pi^*} = \{o_1, u_1, \dots, o_N, u_N\}$
2. Run $\pi_\theta(u_t|o_t)$ to get dataset $D_\pi = \{o_1, \dots, o_M\}$
3. Ask human to label D_π with actions u_t
4. Aggregate: $D_{\pi^*} \leftarrow D_{\pi^*} \cup D_\pi$
5. GOTO step 1.

Problems:

- Execute an unsafe/partially trained policy
- Repeatedly query the expert

A reduction of imitation learning and structured prediction to No-Regret online learning, Ross el al. 2011

DAGGER (in a real platform)

Other Challenges:

1. Is hard for the expert to provide the right magnitude for the turn without feedback of his own actions!

   Solution: provide visual feedback to expert

2. The expert's reaction time to the vehicles' behaviors is large, this causes imperfect actions to be commanded

   Solution: play-back in slow motion offline and record their actions

3. Executing an imperfect policy causes accidents, crashes into obstacles.

   Solution: Safety measures which again make the data distribution matching imperfect between train and test, but good enough.

Imitation Learning

Two broad approaches:

- Direct: Supervised training of policy (mapping states to actions) using the demonstration trajectories as ground-truth (a.k.a. behaviour cloning)
- Indirect: Learn the unknown reward function/goal of the teacher, and derive the policy from these, a.k.a. Inverse reinforcement learning.

Rule Based Decision Making vs Learning Based Decision Making

Rule Based Decision Making:

- ✓ Logic is easy to understand
- ✓ Stable
- ✓ Easy to model and tune
- ✓ Less requirement on processor performance
- ✓ Easy to scale by cascade multiple layers
- x Discontinuity in vehicle behavior
- x Rules may be rendered not effective without careful design
- x Finite state machine may not cover all scenarios
- x Long tail effect

Rule Based Decision Making vs Learning Based Decision Making

Learning Based Decision Making:

- ✓ Advantage of better scenarios coverage, every scenarios can be trained with large data
- ✓ Decision making can be simplified through network
- ✓ Some learning algorithms is capable to extracting environment features and decision making properties.
- ✓ Not necessary to iterate through all scenarios, which will be trained/improved as more data is collected.
- x Difficult to understand why certain decision is made
- x Difficult to modify model
- x Different scenario may end up totally different model
- x Large experimental data is needed as samples for training
- x Performance highly depends on the quality of the data

深蓝学院
shenlanxueyuan.com

Thanks for Listening !