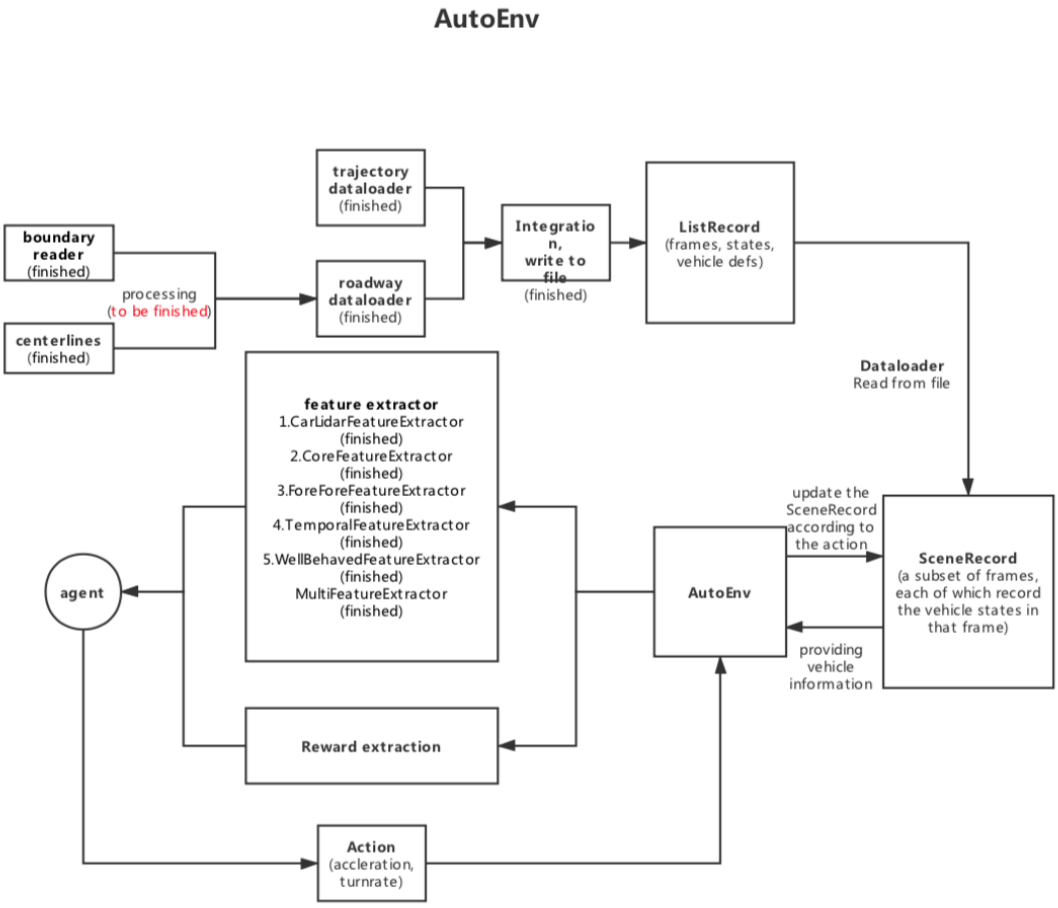
AutoEnv Guide

1. Overview

AutoEnv is a python version environment(simulator) for doing experiments on vehicle driving. The structure of this environment is built on top of rllab's basic functionalities. It is also an extendable environment for researchers to do flexible adjustments according to their goal.

2. Working Flow Introduction

AutoEnv is basically consisted of two parts, one is in charge of dealing with the raw data while the other one is an interactive interface between the agent and our vehicle world(extracted from our raw data).



2.1 Data Loader

For the raw data, we now have a restricted format but hope to have some improvement in the future. Three basic files are needed, namely boundary, centerline and trajectory file. We use boundary and centerline file to build the roadway class, which stores the information of

the road on which our data are collected. This includes line numbers, line width, line curvature, etc. We need trajectory files as our expert trajectories for future usage and store it in a NGSIMTrajdata class. Then we combine the information in these two classes which is stored in a ListRecord class, which has attributes frames, states, vehicle definitions, etc. In each frame we have a list of vehicles appear in that timestep, whose information is stored in the Vehicle class object.

2.2 Interactive Interface

To initialize our environment, we need to get a subset of the frames in ListRecord and create a SceneRecord to store those frames. We can either do random sampling or specific selection to get those frames.

We have a feature extractor to extract all kinds of features from our SceneRecord object, e.g. collision or not, distance indicated by the car lidar and so on. We also have a reward indicator to give us the reward of a specific state, this part can be flexibly designed.

Our environment has two ways to provide our agent with observations(features), **reset** and **step. Reset** is called whenever we want to re-select a new subset of frames or do another round of experiment, it can initialize our SceneRecord object and return the current observations(features) to our agent(ego vehicle). **Step** is called whenever the agent choose an action, the environment will collect the action, do the propagation, update its SceneRecord and return the observations(features).

The agent can receive observations(features) and rewards from the environment, then it can decide the next action accordingly. Notice that, our agent is not restricted to a single agent, we can have multiple agents and our action will become a list of actions which are then sent to the environment. Then it results in a loop between the agent and environment, the former one give actions while the latter one provide observations(features)

2.3 Actions

Action from a single agent is consisted of acceleration and turning rate. In a multi-agent setup, we can have an action list where the length of the list equals to the number of our agent.

The propagation function is an approximate physical simulation given the current vehicle state and its action.

3. Algorithms

3.1 AGen

AGen is an Adaptable Generative Prediction Model. It consists of two phases, an off-line training phase and an online adaption phase which adjust each individual's policy according to their individual history trajectories.