Outline:

1. What is a scene?

scene, a configuration of physical objects and agents.

1. Why do we need Scenic or probabilistic programming languages (PPLs)?

The traditional ML approach to these problems is to gather more data from the environment, retraining the system until its performance is adequate. 🡪 slow and expensive, and hard to target the rare, needed cases.

# PPLs are used to train ML with more systematical input and it also takes care corner cases.

1. What are the features of Scenic?
2. factoring the process into syntactically independent specifiers which can be combined in arbitrary ways.
3. Providing constructs to generalize simple scenarios by adding noise or by composing multiple scenarios together.
4. Reduced sample space 🡪 Random generation of concrete scenarios as scenario improvisation, as it is inspired by and closely related to a class of problems known as control improvisation. ( spatiotemporal pruning technique)
5. Scenic enforces several default requirements: all objects must be contained in the workspace, must not intersect each other, and must be visible from the ego object.
6. What is the purpose to use environment modeling language, like VerifAI?

To intelligently select an unexplored concrete scenario that is likely to induce a violation of a specification.

1. Flow:

一張含有 文字, 螢幕擷取畫面, 字型, 圖表 的圖片

自動產生的描述

1. Some Basic Commands in Scenic:

(a) Using “offset by” as above overrides the default position of the Car.

(b) Scenic’s general operator “X relative to Y”.

(c) “left of X by D and with model M” are specifiers 🡺specifiers can specify multiple properties simultaneously.

(d) “X can see Y” predicate.

(e) “mutate taxi by 2” (which adds twice as much Gaussian noise)

1. Some Advanced Commands in Scenic:
2. We call Scenic objects which take actions (behaviors) over time dynamic agents, or simply agents.
3. This “try-interrupt” statement has the following semantics: at first, the code block after the try (the body) is executed.
4. Successive interrupt clauses take precedence over those which precede them.
5. The “require always” statement enforces that the given condition must hold at every time step of the scenario.
6. a condition hold at some time during the scenario using the “require eventually” statement.
7. Behavior 🡪 Run automatically repeatedly/ Monitor🡪 “While” and “wait” /Scenario 🡪 reusable blocks. (setup and compose)
8. the “initial scenario”, i.e., the very first scenario to run.
9. Definitions in experiments: Precision = tp/(tp+f p )and Recall = tp/(tp +f n ), where true positives tp is the number of correct detections, false positives f p is the number of predicted boxes that do not match any ground truth box, and false negatives f n is the number of ground truth boxes that are not detected.

Ideas:

This paper gives me a starting point to realize that the scenario could be decomposed into many parts with parallel and serial behaviors at the same time while following the priority like collision avoidance should be a priority for most cases.

Also, I see the usage of probabilistic programming languages (PPLs) can heavily lower the time and cost to effectively train a model.

Next Step:

1. Scenario test in Autonomous Vehicles: <https://ieeexplore.ieee.org/abstract/document/9294368>

(From the usage of Scenic to the real-life test. 2020)

<https://ieeexplore.ieee.org/abstract/document/9936045>

(the case study for the “merging” scenario, 2022)

<https://ieeexplore.ieee.org/abstract/document/10107450>

(the idea of “Advanced Scenario Generation” to have less gap between the scenarios tests and the reality, 2022)

1. Scenario and Prediction:

<https://ieeexplore.ieee.org/abstract/document/9733973>

(Compare the different models for motion predictions, 2019)

1. Generation for Scenarios:

<https://www.sciencedirect.com/science/article/pii/S2046043022000867>

( using data-mining method to generate the scenarios for the “traffic accidents” ,2023 International Journal of Transportation Science and Technology)