scenario_simulator_v2
完全自動運転のための
シナリオテストフレームワーク

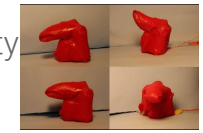
SWEST 23

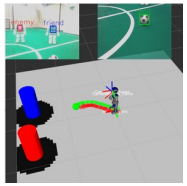
Tier IV.inc Software Engineer 片岡　大哉

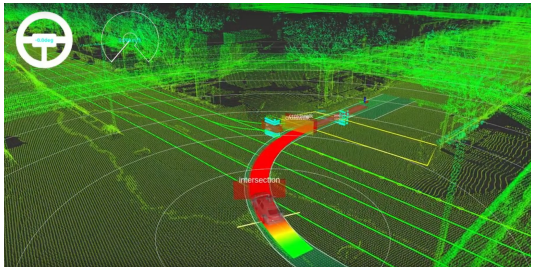
Tier IV

自己紹介

- 2010
  - プログラミングをやろうとして「猫でもわかるC言語プログラミング」を購入、何もわからなくて挫折
- 2014
  - 大学時代にVRがやりたくてOculus DK2を購入してUnityでゲームプログラミングを始める
- 2016-2018
  - 研究室でロボットを始める
    - 大学時代の専門はSoft Robotics
    - ロボカップでロボットサッカーにハマる
  - 海洋ロボットの開発に着手
    - 今も趣味で自律航行システムを開発中
- 2018-
  - 株式会社ティアフォー入社
    - 現在はSimulationチーム所属
    - AutowareのSimulationやCI/CD周りを担当







Autoware

オープンソースな自動運転システム

自己位置推定、認識、計画、制御、ハードウェアドライバーまで**すべてのソフトウェアスタック**をgithub上で公開

AWFによる世界連合

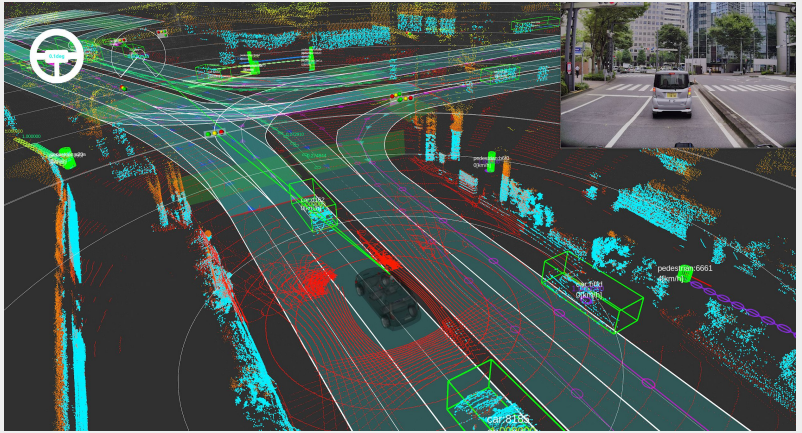
Tier IVー社で開発、運用しているわけではなく**AWF***加盟企業による世界連合によってエコシステムを形成

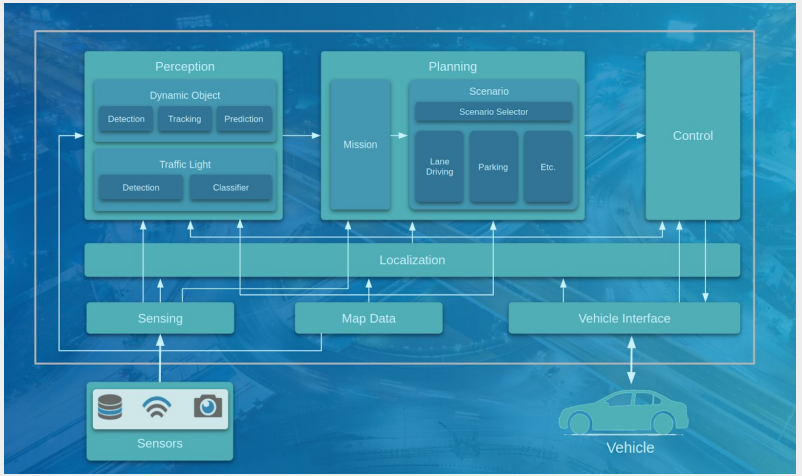
開発、運用を支えるプラットフォーム

クラウドやIoTを活用したプラットフォームである**Web.Auto**による強力なサポート

Point

OSSであるAutowareを使えば誰でも自動運転サービスに参入できるようになり自動運転の民主化が実現





* AWF=The Autoware Foundation

Autoware開発における Simulationの役割

デグレの防止

自動運転システムは非常に複雑でありデグレが起きやすい。Simulatorで多数のテストケースを事前に確認してから公道に行かないと事故の原因に

実車試験をするコストの低減

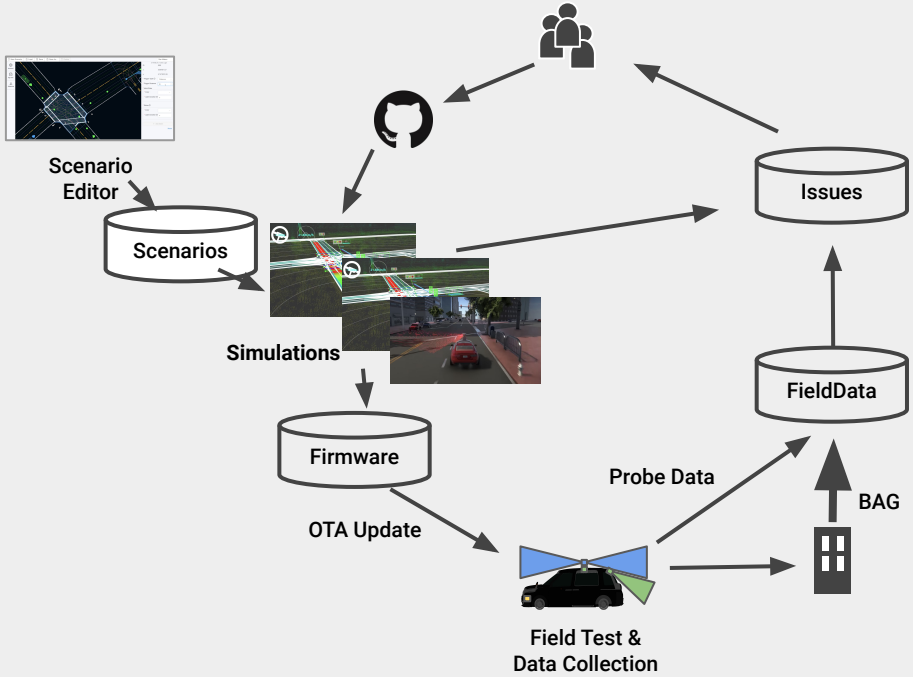
実車試験ですべてのテストケースを実施するには大きなコストがかかりくまなくテストすることは非現実的

事故リスクの最小化

自動運転システムが事故を起こしてしまった場合公道であれば人命に関わることも起こりうるリスクが存在

Point

仮想環境で実環境走行前のテストを実施することで公道実験に伴うリスクとコストを最小化することが可能



Point

Simulation環境をWeb.Autoと呼ばれるAutowareを支えるクラウドシステムとに組み込むことで開発効率をさらに高める

Simulationの実施方法

ODD類型別ユースケース検証

所謂、リグレッションテスト.
自車両以外の障害物や信号などの動きを定義し、事前に定義されたシナリオを走りきれるかテストする.
変数が多くテストケースが膨大に

ランダム検証

ある簡単なルールを条件に自車両以外を発生させて、事故を発生させることなく長時間走れるかの検証.
新たなユースケースの発見などに利用.

ODD (Operational Design Domain)

引用: Tier IV Safety Report 2020

シナリオテストのpain point

膨大なテストケース

現在Tier IVでは何百、何千といったテストケースが毎日自動検証されており、も含めると今後テストケースの数は指数関数的に増加

完璧なシミュレータは存在しない

全てのセンサ、ECU、車両の運動を完璧にsimulationできるシミュレータは存在しない。全てのテストケースを満たすには複数のシミュレータを組み合わせた検証パイプラインを組み立てが必要

シナリオ言語はResearch Area

どのようにシナリオを記述するかは自動運転システムの評価方法やプログラミング言語開発的視点から研究が進められている分野であり現状最終的な解決策が不明瞭

要求仕様

決定性

あまりにたくさんのテストケースがあるため、人力で一つ一つ検証するのは合理的ではなく、いつからこのテストケースがこけるようになったかを効率的に検出するためにはシナリオ実行処理には決定性が必要

Co-Simulation Modelの採用

自動運転シミュレータをNPCの挙動を制御する部分とそれ以外の部分に分割しそれぞれ協調シミュレーションすることでシミュレータが変わってもシナリオを書き直す必要がない構造を実現

複数シナリオフォーマットサポート

交通流シミュレータを共有ライブラリとして実装、それをラップする形でシナリオ解釈器、言語を実装することで同時に複数のシナリオフォーマットのサポートを実現

シナリオテストのpain point

シナリオが冗長

NPCはシナリオの指示に従って動く必要があるが、かなり細かく挙動を指示する必要があり、シナリオが長く、冗長に

テストツールの信頼性

NPCロジックに機能追加を行ったりバグ修正を行った際に挙動が変化して既存シナリオの実行結果が変わる可能性がある

要求仕様

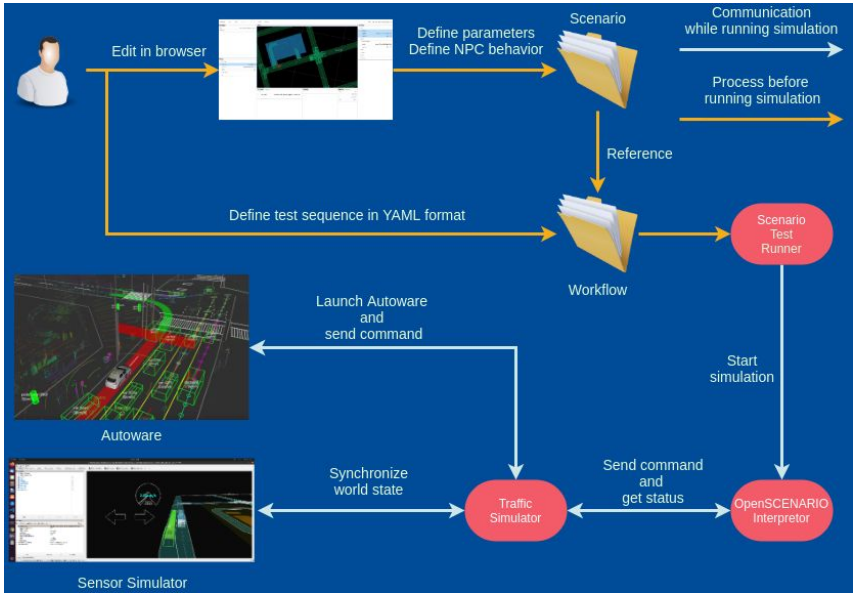
ロボット技術の導入

自動運転システム技術をNPCロジックに対して適用、最低限の指示で道路交通法を守りながらシナリオの指示を受けて動くNPCを実現

ツールに対する徹底したCI/CD

本フレームワークへのソースコード変更を含む変更に対しては単体テストだけではなく結合テストも実施。結合テスト時の実行シナリオは分岐網羅率を確認しながら漏れをできるだけ防ぐ工夫を導入

設計コンセプト



可能な限り決定性を担保

ROS2に依存しない部分はすべてフルスクラッチで実装、決定性の担保を実現

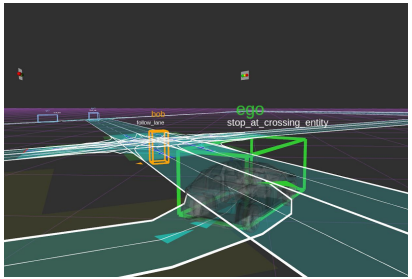
全てのパーツをプラガブルに

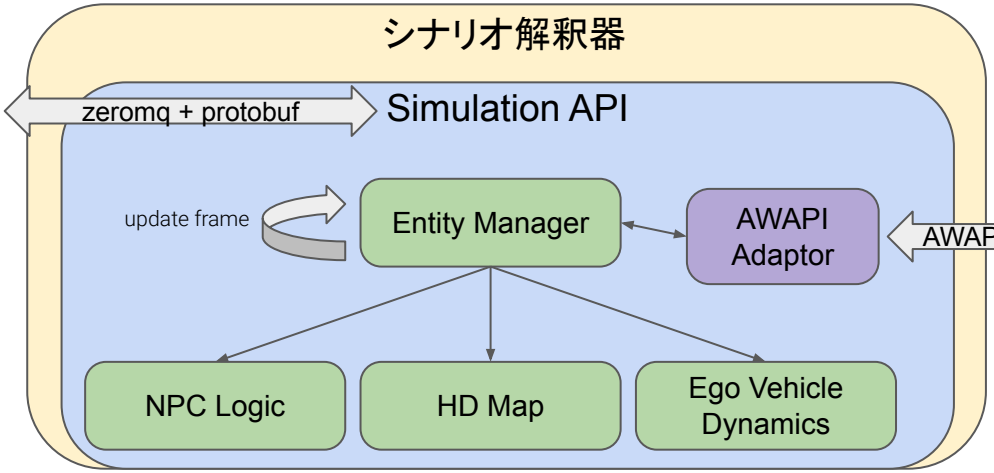
Simulator、シナリオフォーマットは後から変更、追加が可能

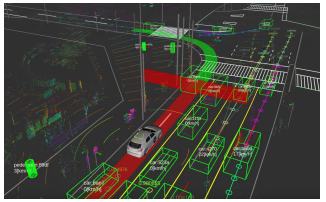
オープンソース

全てのソースコードをapache 2.0ライセンスでgithub上で公開、世界中誰でもcontributionが可能

全体アーキテクチャ







- インタプリタ全体を**モノリシックなバイナリ**で構成、**決定性を担保**
  - Simulation APIにsimulatorのコア機能（NPCロジックやフレーム管理）を詰め込む
  - シナリオ解釈機はシナリオを解釈し、その結果に従って simulatorのC++APIを叩く
  - Simulation APIは任意のROS2 componentに紐付けられるので新しいシナリオフォーマットを構築するのも簡単

Point

・NPCロジック、シナリオフォーマット、simulator、どのAutowareを使うか等を**すべてプラガブル**に
・同期的なAPI構成を取ることによりSimulatorは入力に対して**決定性**を持つ

まとめ

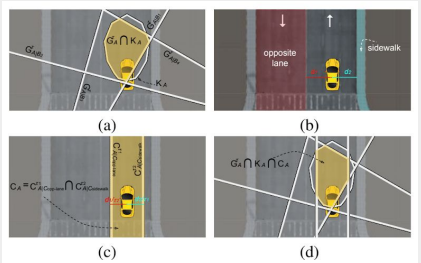
- オープンなシナリオテストフレームワークを紹介
  - 複数のシミュレータをサポート
  - 複数のシナリオ言語をサポート
  - 自動運転システムを応用したNPCロジック
  - 全体アーキテクチャの紹介

Please Check !

https://github.com/tier4/scenario_simulator_v2

今後の展望

SUMMIT: A Simulator for Urban Driving in Massive Mixed Traffic [panpan et al. 2020]



SUMMIT: A Simulator for Urban Driving in Massive Mixed Traffic [panpan et al. 2020]

NPCの更なる高度化

SUMMIT [panpan et al. 2020] simulatorの手法ベースに開発

NPCがさらに複雑な状況下でも適切に行動可能に

将来的には左折時に割り込んでくるバイクや自転車も実現