LLM による実行可能な自動運転評価用シナリオコードの自動生成

EP21086　服部倖也　　指導教授：山下 隆義

1. はじめに

安全な自動運転システムの実現には，あらゆるシーンを対象として，認識アルゴリズムを評価する必要がある．実環境の評価では偶発的に発生する歩行者の飛び出しなど頻度の少ないシーンの評価が困難であるため，シミュレーション環境の利用が検討されている．シミュレーション環境での評価には，大量の評価用シナリオが必要となるが，手作業での作成は人的コストが高い．そこで，本研究では大規模言語モデル（LLM）を用いて効率的にシナリオを再現するコードの自動生成手法を提案する．提案手法では，LLM を用いて生成したシナリオコードはエラーが頻発するため，コード修正に特化した LLM を併用してコード修正を行う．

2. ChatScene

ChatScene[1] は，事故発生リスクの高い交通シナリオのコードをシミュレーション環境向けに生成する手法である．ChatScene が生成するシナリオのコードは，Scenic[2] で記述されており，CARLA シミュレータで実行可能である．LLM を用いてシナリオ文から直接 Scenic コードを生成する手法では，実行不可能なコードが多く生成される．そこで図 1 に示すように，ChatScene では，予め LLM によって生成した大量のコードを役割ごとに Behavior, Geometry, Spawn Position の 3 要素に分けて格納したデータベースを用意する．コード生成時には，データベースから取得したコードを組み合わせることでシナリオのコード生成を行う．このデータベースは各要素の説明文と各要素の Scenic コードの特徴ベクトルが紐づけられており，Sentence Encoder で抽出した特徴ベクトルとの類似度をもとにコードを選択する．得られた要素のコードを結合させることで，実行可能な Scenic コードを組成でき，CARLA で実行が可能である．しかし，生成可能なシナリオはデータベースに依存する．そのため，データベースに存在しないコードの生成は不可能であり，生成可能シナリオの多様性は低い．

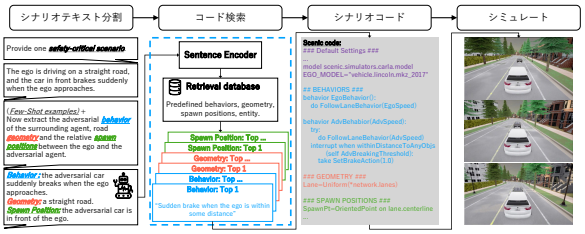

図 1：ChatScene の流れ

3. 提案手法

本研究では，生成可能なシナリオの多様性を向上させることを目的とし，LLM を用いてシナリオのコード生成を行う手法を提案する．提案手法では，シミュレーションでの実行時のエラーに対処するためコード修正 LLM を導入する．提案手法の流れを図 2 に示す．

シナリオテキスト分割

初めに，シナリオとして記述されたテキストを LLM によって Behavior, Geometry, Spawn Position の要素毎に分割する．

コード生成

分割したテキストから要素毎に GPT-4o を用いて Scenic コードの生成を行う．このとき，生成時に Few-shot Prompting を用いることで Scenic コードの生成精度向上を図る．各要素毎に生成したコードを結合することでシナリオ全体の Scenic コードが組成される．

コード修正

生成されたコードは，CARLA で実行時にエラーとなる可能性があるため，コード修正 LLM を導入する．コード修正 LLM では，実行時のエラー文と Scenic コードを LLM に入力することでコード修正を行う．コード修正とシミュレーションでの実行は，実行可能となるまで繰り返し行う．繰り返し修正を行うことで，構文的な誤りが複数あるコードも実行可能になる．コード修正 LLM には，コード修正

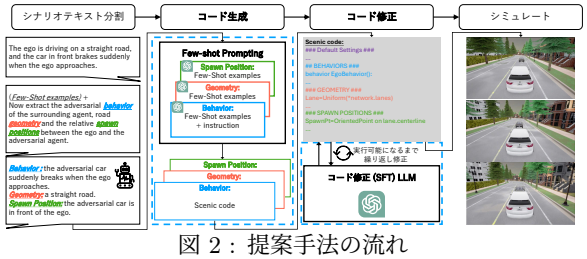

図 2：提案手法の流れ

性能向上のために Supervised Fine-tuning(SFT) を行った GPT-4o を用いる．

4. 評価実験

本実験では，コード修正 LLM の有無による生成したコードの実行可能率を比較する．コード修正 LLM は，SFT 済み GPT-4o と手動でリファレンスを与える GPT-4o の 2 つを比較対象とする．SFT は，ベースモデルが gpt-4o-2024-08-06, エポック数が 10，バッチサイズが 1 で，修正前 Scenic コードと修正済み Scenic コード 10 件のデータセットを学習する．手動でリファレンスを与える GPT-4o は，RAG を人手により模したものでコード修正の際にリファレンスからエラーに関連する部分を手動で与える．

4.1. 実験条件

データセットには，多様なシナリオに対する評価を行うために独自に用意した 20 シーン分の自然言語による説明を用いる．各シーンでのコード生成及びコード修正を実行し，その結果を評価する．評価指標は，「生成した Scenic コードが実行可能か」,「指示に則したシナリオか」,「平均繰り返し修正回数」である．コード修正 LLM を用いる場合の修正回数は 10 回を上限とする．

4.2. 実験結果

コード修正 LLM を用いない手法，コード修正 LLM に GPT-4o を用いて手動でリファレンスを与える手法，コード修正 LLM に SFT 済み GPT-4o を用いる手法について，それぞれの実行可能数，平均修正回数を表 1 に示す．表 1 より，コード修正 LLM を導入することで実行成功数の増加が確認できる．また，コード修正 LLM に SFT 済み GPT-4o を用いることで，実行成功数が増加し平均繰り返し回数が減少した．このことから，コード修正 LLM の SFT が有効であると考えられる．また，実行に失敗したコードのエラーを調べたところ，構文エラーと属性エラーが多いことが分かった．これは，SFT による構文修正能力が不十分であること，Scenic のオブジェクト・クラスに関する情報が足りないことに起因すると考えられる．

表 1：コードの実行成功数と平均繰り返し修正回数

コード修正 LLM	手動リファレンス	SFT	実行成功数（指示通りのシナリオ実行）	平均繰り返し修正回数 [回]
			0 (0) / 20	-
✓	✓		3 (1) / 20	7
✓		✓	5 (2) / 20	6.4

5. おわりに

本研究では，LLM を用いて実行可能なシナリオを自動生成する手法を提案した．評価実験では，コード修正 LLM の SFT を行った後，シナリオ実行結果に対する評価を行うことで実行成功数の増加，平均繰り返し修正回数の減少を確認した．今後は，コード修正 LLM に更なる SFT を行うことやオブジェクトの属性情報を有する RAG を用いることで実行可能数の向上を図る予定である．

参考文献

[1] J. Zhang, *et al.*, “ChatScene: Knowledge-Enabled Safety-Critical Scenario Generation for Autonomous Vehicles”, CVPR, 2024.

[2] D. J. Fremont, *et al.*, “Scenic: A language for scenario specification and data generation”, Machine Learning, pages 1–45, 2022.