Piece and Conquer Fireworks 알고리즘을 이용한 자율주행 알고리즘 하이퍼파라미터 최적화 기법

김명준[1], 김건우[1*]
[1]경상국립대학교 컴퓨터과학부
gnu kim98@gnu.ac.kr, gunwoo.kim@gnu.ac.kr

Hyperparameter Optimization of Autonomous Driving exploiting Piece and Conquer Fireworks Algorithm

MyeongJun Kim[1], Gun-Woo Kim[1*]
[1]School of Computer Science, Gyeongsang National University

요 약

본 논문은 F1TENTH 와 같은 자율주행 경주 대회를 위한 고전적인 자율주행 알고리즘의 파라미터 최적화에 관한 연구를 다룬다. 고전적인 자율주행 알고리즘은 하이퍼파라미터의 영향을 크게 받고 더 나아가서 하이퍼파라미터의 설정에 따라서 성능의 차이가 크다. 이 하이퍼파라미터를 빠르게 찾기 위하여 Piece and Conquer Fireworks 방법을 제안한다. 결과적으로 Random search 에 비해서 일반 Fireworks 알고리즘은 약 8.3 배, Piece and Conquer Fireworks 알고리즘은 약 28.5 배 빠른 성능을 보여준다.

1. 서론

최근 테슬라, 구글, GM 등 여러 회사에서 자율주행을 연구하고 있고 시범운행을 하고 있다. 이러한 흐름에 따라서 Indy Autonomous Challenge, Roborace, F1TENTH 등 자율주행 레이싱 대회들도 같이 인기를 끌고 있다.

본 논문에서는 F1TENTH 에서 많이 사용하는 고전적인 자율주행 알고리즘 FGM(Follow Gap Method)을 선택해 lap time 을 줄이려고 한다. FGM 이라는 고전적인 알고리즘의 하이퍼파라미터를 최적화하여 주어진 map 을 빠르게 완주하려고 한다. 이러한 고전적인 알고리즘은 하이퍼파라미터 설정에 따라서 성능의 차이가 크게 난다.[1][2]

FGM 은 핸들을 가장 큰 gap 으로 항하도록 계속 조절하여 자율주행을 하는 알고리즘이다. FGM 의 하이퍼파라미터는 다른 자율주행 알고리즘에 비해서 직관적으로 어떤 알고리즘이 어떤 상황에서 얼마만큼 영향을 끼치는지 알기 어렵고, 하이퍼파라미터간의 상관관계가 적어 사람이 휴리스틱하게 하이퍼파라미터를 찾기 어려워서 본 논문의 실험 알고리즘으로 정하였다. 본 논문에서는 6 개의 하이퍼파라미터를 사용하여 자율주행 알고리즘을 구성하였다.

하이퍼파라미터 탐색을 위한 알고리즘으로는 Fireworks 알고리즘을 사용하였다. Fireworks 알고리즘은 샘플링 기반의 탐색 알고리즘이며 목적함수의 형태가 복잡하여도 좋은 성능을 보여준다. 그리고 Piece and Conquer(PnC) Fireworks 알고리즘을 제안하여 Random search 와 일반 Fireworks 알고리즘, PnC Fireworks 알고리즘을 비교하였다.

2. 관련연구

본 연구에서 사용된 기본 알고리즘은 “Fireworks Algorithm” 이다.[3] 이 알고리즘은 다음과 같은 순서로 진행된다. 첫번째로 n 만큼의 무작위 시작 지점을 선택한다. 두번째로 n 의 각각의 지점 근처를 탐색하여 k 개의 후보를 무작위로 선출한다. 세번째로 n * k 개의 탐색한 후보들 중에서 기대치를 충족하는 값이 있으면 종료한다. 만약 기대치를 충족하는 값이 없으면 n * k 개의 탐색한 후보들 중에서 질이 좋은 n 개를 선택하고 첫번째 과정으로 돌아가게 된다. 이 과정은 기대치를 충족하는 값이 나올 때 까지 반복한다. Fireworks 알고리즘은 목적함수가 불규칙적일 때 좋은 성능을 보인다.

* 교신저자 (Corresponding Author)

3. 데이터셋 및 모델 구조

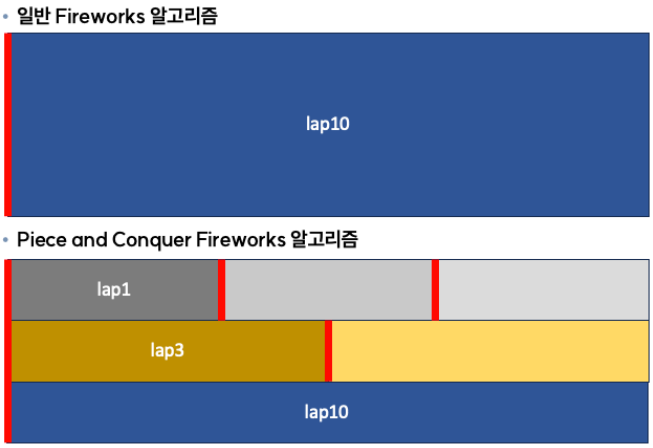
본 논문에서 자율주행 알고리즘의 주행은 python gym 기반의 F1TENTH 시뮬레이터를 이용하였다. 이 시뮬레이터는 각종 map, dynamics 값들을 custom 할 수 있다. map 은 Oschersleben 을 사용하였고 dynamics 값들은 시뮬레이터의 기본값을 사용하였다.

본 논문에서 FGM 의 파라미터는 6 개로 설정하였다. 각 하이퍼파라미터별 범위는 <표 1>과 같이 설정하였다.

<표 1> FGM 의 하이퍼파라미터와 설정 범위

하이퍼파라미터 이름	하이퍼파라미터 설정 범위
BUBBLE_RADIUS	30 단위로 90 ~ 240(6 개)
PREPROCESS_CONV_SIZE	1 단위로 2~5(4 개)
BEST_POINT_CONV_SIZE	10 단위로 70~120(6 개)
MAX_LIDAR_DIST	1000000 단위로 2000000~5000000(4 개)
STRAIGHTS_SPEED	0.2 단위로 7.0~11.2(15 개)
CORNERS_SPEED	0.2 단위로 5.0~9.2(15 개)

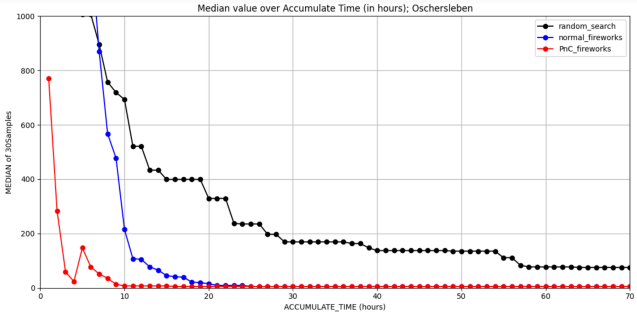
본 논문에서 제안하는 PnC 구조는 직역하자면 조각별로 정복한다는 뜻이다. 일반적인 Fireworks 알고리즘은 목표 lap 이 10 바퀴인 경우 10 바퀴를 완주한 데이터만을 이용하여 최적의 하이퍼파라미터를 찾는다. PnC Fireworks 알고리즘은 lap1 완주 조각의 최적결과 n 개 종료 지점이 lap3 완주 조각의 n 개 시작 지점이 되고 lap3 완주 조각의 최적결과 n 개 종료 지점이 lap10 완주 조각의 n 개 시작 지점이 된다. 즉, lap1 의 데이터조각과 lap3 의 데이터조각을 목표인 lap10 의 탐색에 이용한다.



(그림 1) 일반 Fireworks 알고리즘(위)과 PnC 알고리즘(아래)

PnC Fireworks 알고리즘은 같은 시간이 주어질 경우 더 많은 하이퍼파라미터에 대해서 탐색할 수 있다. (그림 1)의 사각형 너비는 하나의 하이퍼파라미터 탐색에 걸리는 시간이고 사각형 높이는 search 알고리즘이 탐색한 하이퍼파라미터들의 갯수이고 사각형 넓이는 search 알고리즘이 하이퍼파라미터를 탐색하는데 필요한 총 시간이다. (그림 1)을 보면 일반적인 Fireworks 알고리즘은 사각형의 높이만큼만 탐색 가능한데 반해 PnC Fireworks 알고리즘은 사각형 높이보다 더 많은 탐색을 할 수 있다. 이것은 이해를 돕기 위해 간단하게 도식화를 한 것이며 실제로는 약 4.6 배 더 많은 탐색을 할 수 있다.

(그림 2) Random, normal Fireworks, PnC Fireworks 결과



4. 결과 및 결론

(그림 2)의 X 축은 FGM 알고리즘 주행 시간이고 Y 축은 해당 시간까지 탐색한 하이퍼파라미터들 중 최적 하이퍼파라미터의 순위를 의미한다. 각각의 알고리즘마다 30 번씩 실행한 값의 중앙값을 사용하였다.

Random Search, 일반 Fireworks 알고리즘, PnC Fireworks 알고리즘을 비교해본 결과 (그림 2)과 같이 Leaky PnC Fireworks 알고리즘이 가장 우수한 성능을 보였다. 최적의 값을 찾기까지 Random Search 는 400 시간, 일반 Fireworks 알고리즘은 48 시간, PnC 알고리즘은 14 시간이 걸렸다. 만약, 시뮬레이션 가속 기능을 이용할 경우 약 8 배에서 10 배정도 빠르게 탐색이 가능하다.

본 논문을 통해서 PnC Fireworks 알고리즘이 고전적인 자율주행 알고리즘의 하이퍼파라미터를 찾는데 도움을 준다는 것을 알 수 있었다.

Acknowledgement

본 논문은 2023 년도 정부(교육부)의 제원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (NRF-2021R1G1A1006381)

참고문헌

[1] Badue, Claudine, et al. "Self-driving cars: A survey." Expert Systems with Applications 165 (2021)

[2] Moll, Mark, et al. "HyperPlan: A framework for motion planning algorithm selection and parameter optimization." 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2021.

[3] Tan, Ying, and Yuanchun Zhu. "Fireworks algorithm for optimization." Advances in Swarm Intelligence: First International Conference, ICSI 2010, Beijing, China, June 12-15, 2010, Proceedings, Part I 1. Springer Berlin Heidelberg, 2010.