Journal of Korea Game Society 2015 Oct; 15(5): 131-142
http://dx.doi.org/10.7583/JKGS.2015.15.5.131

게임 프로그래밍

자율운행 자동차의 에이전트 설계 및 프로토타입 개발

임승규, 이재문
한성대학교 멀티미디어공학과
kd0576@naver.com, jmlee@hansung.ac.kr

Design and Prototype Development of An Agent for Self-Driving Car

Seung Kyu Lim, Jae Moon Lee
Dept. of Multimedia Engineering, Hansung University

요 약

자율주행 자동차는 전통적인 차량의 주요 수송 기능을 수행 할 수 있는 자동 운전 차량 말하며, 그것은 주변 환경을 감지하고 인간의 어떠한 입력 없이 이동 가능하여야 한다. 본 논문에서는 이러한 자율주행 자동차를 시뮬레이션 할 수 있는 자율주행 자동차 에이전트를 설계하고 이에 대한 프로토타입을 개발하였다. 이를 위하여 자율주행 자동차에 대한 요구 사항을 분석하고, 전통적인 다중 에이전트 시스템에 적합하도록 에이전트를 설계하였다. 설계의 핵심 은 에이전트들은 오직 조종힘에 따라 이동하도록 하는 것이다. 설계된 에이전트의 프로토타입은 유니티3D를 이용하여 구현되었다. 프로토타입을 이용한 시뮬레이션 결과, 에이전트의 이동은 매우 자연스럽게 나타났다. 그러나 에이전트 수를 증가시키는 경우에 성능이 심각하게 저하되었고, 이에 대한 대안들을 제시하였다.

ABSTRACT

A self-driving car is an autonomous vehicle capable of fulfilling the main transportation capabilities of a traditional car. It must be capable of sensing its environment and navigating without human input. In this paper, we design the agent that can simulate these self-driving cars and develop a prototype for it. To do this, we analyze the requirements for the self-driving car, and then the agent is designed to be suitable for traditional multi-agent system. The key point of the design is that agents move along the steering forces only. The prototype of the designed agent was implemented by using Unity 3D. From simulation results using the prototype, movements of the agents were very realistic. However, in the case of increasing the number of the agent the performance was seriously degraded, and so the alternatives of the problem were suggested.

Keywords : Self-Driving car(자율주행 자동차), Simulation(시뮬레이션), Agent(에이전트), Steering Force(조종힘)

Received: Jul, 05, 2015 Accepted: Aug, 10, 2015
Corresponding Author: Jae Moon Lee(Hansung University)
E-mail: jmlee@hansung.ac.kr

ISSN: 1598-4540 / eISSN: 2287-8211

© The Korea Game Society. All rights reserved. This is an open-access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.otg/licenses/by-nc/3.0), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. 서 론

자율주행 자동차에 대한 연구는 최근 10여년 사이에 많은 연구가 되어 왔다 [1,2]. 자율주행 자동차는 사람의 어떠한 개입 없이 자동차가 스스로 알아서 출발지에서 목적지까지 자동으로 운전해 가는 자동차이다. 이것은 최근 컴퓨터 기술의 발전과 무인 자율주행과 관련된 다양한 센서 기술의 발전이 있었기 가능하다[1]. 자율주행의 장점은 자동차 사고를 획기적으로 줄일 수 있을 뿐만 아니라 출퇴근 시간 절약, 에너지 절감 등의 효과를 얻을 수 있다[1,3]. 특히 노령화 사회에 자율주행 자동차는 필수적인 자동차 모델이 될 것이다.

미국 방위고등연구계획국(Defense Advanced Research Projects Agency)이 주최한 Grand Challenge대회 이후 자율주행 자동차 기술은 미래 자동차 기술로서 세계적 관심을 모으고 있다[4]. 2009년부터 현대자동차/기아자동차 주관으로 매 2년마다 '자율주행자동차 경진대회(AVC: Autonomous Vehicle Contest)'를 개최하여 관련기술 개발과 인력양성을 꾀하고 있다. 자동차 제조회사에서 진행되고 있는 자율주행 자동차 연구는 자율주행 기술을 단계적으로 추가하는 방향으로 이루어지고 있다. 대부분의 자동차 제조사들은 완전한 자율주행 자동차 보다는 수동주행을 보조하는 부분적인 자율주행 자동차를 개발하고 있는 실정이다. 이웃 자동차를 고려한 자율주행은 차간거리, 충동회피 정도의 수준이다. 현대모비스는 2014년 9월 보행자 인식 및 전방 차량 추월, 상황별 자동제동 및 가감속 기능을 구현하는 자율주행시스템과 원하는 장소의 빈 공간을 찾아 스스로 주차하는 자율주차시스템을 시연하는 데 성공했다[1]. 자동차 제조회사들의 연구와 별도로 대학, 연구소 중심으로 이루어지는 자율주행 자동차에 대한 학술 연구도 매우 활발하다. 주요 연구 결과로 자율주행 자동차의 경로 탐색[4], 장애물, 신호등 및 전면 자동차 인식 알고리즘[5], 자율주행 자동차를 위한 프레임워크 ThinkingCap-II 개발[6], 적응형 유한상태기계 기법을 사용하여 자율주행 자동차의 출발지와 목적지에 대한 경로계획을 생성하는 연구[6,7], 경로 계획에 대한 실시간 처리를 위하여 병렬-직렬 구조를 갖는 하이브리드 모델[8] 등이 있다. 이러한 연구 결과로 단일 자동차만 존재하는 실제 도로에서 자율주행이 가능함을 보였다[6,7,8]. 자율주행 자동차에 대한 국내외 연구 동향을 종합하면, 자동차 제조회사 중심의 연구는 완벽한 자율주행 자동차 보다는 수동주행 자동차에 대한 보조 장치 개발하여 상용화하고 있는 것이 특징이고, 대학, 연구소등에서는 전자, 로봇, 자동차 분야 중심으로 자율주행에 필요한 센스 기술들의 개발과 기존 센스들의 정밀도를 높이는 연구를 하고 있으며, 자동차, IT분야 중심으로는 자율주행 자동차의 경로 계획, 장애물 처리 등 인공지능 분야에 집중적으로 연구를 하고 실정이다.

본 논문에서는 이러한 자율주행 자동차를 모델링하여 자율주행 자동차를 시뮬레이션 하기 위하여 이에 필요한 에이전트를 설계하고 이에 대한 프로토타입을 개발하는 것이다. 프로토타입 개발의 목적은 향후 자율주행 자동차의 시뮬레이션에 있어서 본 논문에서의 설계와 같이 하는 경우 사실적 시뮬레이션이 가능한 것인지를 사전에 알아보는 것과 기존의 컴퓨팅 파워로 얼마나 대규모의 시뮬레이션이 가능한지를 예측해 보는 것이다. 미래 기술을 시뮬레이션 하는 것이기 때문에 기술적 관점에서 많은 가정을 적용한다. 본 논문에서는 내비게이션의 명령을 인식하여 자율 주행하는 것으로 가정하였다. 출발지와 목적지를 입력하면 내비게이션은 모든 주행 경로를 계산하고, 자동차의 현재의 위치와 도로의 상황을 파악하여 자동차가 목적지에 도달하도록 순차적으로 명령을 내린다. 그러면 자동차는 주변 자동차, 신호등과 같은 도로 환경을 인식하여 내비게이션의 명령을 수행하든지 아니면 이전의 명령을 반복하여 수행하는 것으로 하였다. 내비게이션은 자동차가 명령을 수행하지 못한 것은 인식하면 현 위치에서 목적지 사이의 새로운 주행 경로를 계산하고 그 경로에 따라 새로운 명령을 지시하는 것으로 가정한다. 이러한 가정은 현재 자

동차에 설치된 내비게이션의 기능을 고려할 때 충분히 실현 가능한 기술이다[9].

본 논문에서의 결과는 다양하게 활용될 수 있다. 미래의 자율주행 자동차의 개발에 필요한 다양한 기술들을 자동차와 도로 환경 관점에서 예측하고 시뮬레이션 하는데 활용될 수 있다. 또한 운전 연습용 시뮬레이터로 활용할 수 있다. 새롭게 운전을 배우는 사람들이 이 시뮬레이터로 가상 운전을 하는 경우 주변에 자율적으로 움직이는 자동차들이 있는 경우 훨씬 현실감 있는 운전 연습이 될 것이다. 기존의 운전 연습용 시뮬레이터는 단순히 목적 없이 지나가는 차량이 있는 정도의 수준이다. 더하여 레이싱 게임 등에서도 보다 사실감 있는 게임을 제작하는데 활용될 수 있을 것이다.

논문의 내용을 쉽게 설명하기 위하여 다음 기호를 사용한다. i는 에이전트를 의미하며 m_i, p_i, v_i 및 v_{mi}는 각각 에이전트 i의 중량, 위치, 속도 및 최대속도를 표시한다. $|p-q|$는 벡터 p와 q사의의 크기를 표시한다.

2장에서는 자율주행 자동차의 기능을 분석하며, 3장에서는 자율주행 자동차에 대한 에이전트를 설계한다. 4장에서는 프로토타입 개발 내용을 설명하며, 5장에서는 시뮬레이션 결과를 논한다. 6장에서는 논문의 결론을 설명한다.

2. 자율주행 자동차 기능 분석

2.1 자율주행 차량의 기본 기능

본 논문에서 기본 기능은 도로에 자율주행 자동차 1대만 존재하여 다른 자동차와 상호작용 없이 자율주행하기에 충분한 기능을 의미한다. 다음은 본 논문에서 분석한 기본 기능이다.

- 출발지와 목적지를 입력받아 경로를 결정하는 내비게이션 기능이 있어야 하다. 이 내비게이션은 명령을 내린 후 자동차가 명령을 수행하였는지 여부를 체크할 수 있는 기능도 있어야 한다.
- 자동차의 현재 위치(위도/경도)를 인식할 수 있어야 한다.
- 자동차가 현재 위치하고 있는 도로에서 차선을 인식할 수 있어야 한다.
- 자동차 정면에 있는 신호등의 상태를 인식할 수 있어야 한다.
- 자동차 좌우전후 장애물을 인식할 수 있어야 한다.
- 자동차 조종에 필요한 힘을 가속도로 변환할 수 있어야 한다.

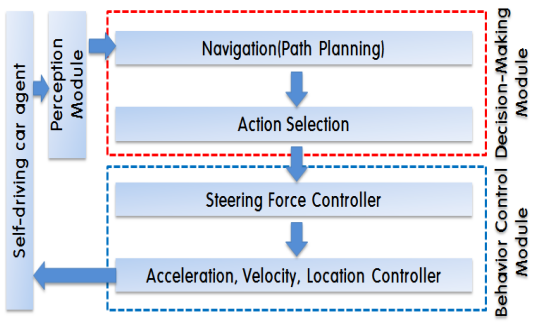
첫 번째 기능은 매우 복잡한 기능이지만 실제 현재 상용화되어 있는 내비게이션의 기능을 보면 충분히 실현 가능한 기능이다[9]. 두 번째 기능은 자동차의 위치를 인식해야 내비게이션에서 이를 기반으로 자동차에 명령을 내릴 수 있으므로 반드시 필요한 기능이다. 현재에도 상용 내비게이션 자체에 GPS를 내장하고 있어 이의 기능은 큰 문제가 없다고 판단한다. 세 번째 기능인 차선인식 기능은 비교적 어려운 기능이다. GPS 기능이 매우 우수하다면 도로맵 상에 위도, 경도 좌표를 저장함으로써 쉽게 해결할 수 있으나 GPS의 오류 범위가 차선의 넓이보다 큰 경우 해결하기 매우 어려운 문제이다. 한편 카메라를 이용하여 차선을 인식하는 것도 부분적으로 가능하나 현재의 도로사정, GPS의 성능으로 차선을 인식하는 기능은 쉬운 문제가 아니다[10]. 네 번째 신호등의 상태를 인식하는 것은 무엇보다 중요한 문제이다. 이 문제 역시 현재의 도로 사정이나 영상 인식기술로 쉽게 해결할 수 있는 문제가 아니다. 다섯 번째 좌우전후 장애물의 인식은 비교적 현재의 기술로 많이 해결된 문제이다. 마지막으로 조종에 필요한 힘을 가속도로 변환하는 기술은 현재 자동차의 크루즈 시스템[1,2]이나 미래의 전기 자동차 기술을 고려하면 큰 문제가 아니다.

본 논문에서는 상기 기본 기능은 미래의 기술이

나 도로 환경으로 충분히 극복될 수 있기 때문에 자율주행 자동차가 상기 5가지 기능을 완벽히 가졌다고 가정한다.

2.2 자동 에이전트 시스템

자율주행 자동차를 시뮬레이션 하는 다양한 방법이 존재 한다[3,11]. 가능한 하나의 방법이 자율주행 자동차를 자율적으로 움직이는 하나의 에이전트로 고려하는 다중 에이전트 시스템이다. 다중 에이전트 시스템은 에이전트들을 중앙에서 제어하는 시스템과 각 에이전트들이 상황에 판단하여 자율적으로 조종하는 시스템으로 나뉘어진다[12,13,14,15,16]. 자율주행 자동차 시뮬레이션은 후자가 더 적합하다.



[Fig. 1] System architecture for simulating self-driving cars

자율주행 에이전트는 [Fig. 1]에서와 같이 의사결정 모듈, 행동제어 모듈 및 인식 모듈로 나뉜다. 의사결정 모듈은 인식 모듈로부터 주변 환경정보와 내부 정보를 사용하여 자율주행에 대한 경로를 결정하고 액션 선택에 대한 의사결정을 한다. 이 때 주변 환경정보는 에이전트 주변의 도로 상황, 장애물 상황 등이고 내부 정보는 에이전트의 현재 상태, 마지막 명령의 성공여부 등이다. 에이전트의 행동제어 모듈에서는 의사결정을 바탕으로 필요한 조종 힘을 발생시켜 에이전트를 이동하게 한다. 인식 모듈은 에이전트의 현재의 위치, 차선, 장애물, 신호등 유무 및 상태 등을 장착된 센서들을 이용하여 인식한다.

자율주행 자동차의 액션들은 '출발하기', '직진하기', '차선 바꾸기', '좌.우 회전하기', 'U턴하기' 및 '정지하기'로 정의할 수 있다. '출발하기'는 정지 상태에서 시작된다. 정지 상태는 목적지를 향하여 처음으로 출발 하거나 신호 등으로 인하여 정지 상태에 있다가 출발하는 경우이다. '출발하기'는 의사결정 모듈의 내비게이션에 의하여 '출발하기'에 적절한 좌표가 주어지고 그 좌표를 향하여 찾아가기를 한다. 자동차는 달리고 있는 상태에서 내비게이션이 특정 길이 이상 직진하라는 명령을 받으면 직진한다. 이 '직진하기'의 경우는 외부의 영향이 없는 경우 내비게이션의 다음 명령이 있을 때 까지 주어진 차선을 따라서 직진한다. '직진하기'에서는 도로의 최대 허용 속도와 앞차의 속도 중 낮은 속도록 주행한다.

3. 무인 운전 자동차 설계

3.1 행동 제어의 설계

자율주행 자동차 시스템에서는 다수의 자동차 에이전트가 존재하며 각각의 자동차 에이전트는 목적 지점까지 가기 위하여 다수의 경유 지점을 갖고 있다. 기본적으로 경유 지점을 갈 때는 '찾아가기' 힘[12,13]을 발생시키며 신호등이 빨간불인 상황이어서 정지해야 하거나 목적지점을 도달할 때는 '도착하기' 힘[12,13]을 발생시킨다. 충돌을 피하기 위해 각각의 자동차 에이전트는 현재 속도를 고려하여 주변 에이전트를 탐색한다. 충돌 영역은 정면, 우측, 좌측에 대해서 고려하며 충돌이 예측되면 충돌을 피하는 조종힘 발생시킨다.

자동차 에이전트의 조종힘에 대한 가장 기본적인 힘은 '찾아가기'와 '도착하기'이다[12,13]. '찾아가기'는 목표물을 향해서 최대 속도로 전진하는 힘으로 목표물을 향한 최대의 속도를 요망 속도라고 할 경우[13]에서는 이 힘을 요망속도와 현재 속도의 차이로 표현하였다.

$$f_s(i,p) = c_s m_i \frac{(p-p_i).n() \times v_{mi} - v_i}{\Delta t} \quad (1)$$

여기서 p는 목표물의 위치이며, $(p-p_i).n()$은 벡터 $(p-p_i)$의 단위 벡터를 의미하며, Δt는 힘 $f_s(i,p)$가 적용되는 시간이다. c_s는 다른 조종힘에 대한 '찾아가기' 조종힘의 가중치 계수이다.

(1)에 의하면 $f_s(i,p)$는 목표물 근처에서도 최대 속력을 유지하려는 하는 힘을 나타내므로 정작 목표지점에 도착하게 되면 관성력에 의하여 목표물을 지나치게 된다[12,13]. [13]에서는 목표 지점 근처에서 속도를 줄여 목표 지점에서 정지하도록 하는 조종힘을 '도착하기'라는 이름으로 다음과 같이 정의하였다.

$$f_a(i,p,\alpha) = c_a m_i \frac{\alpha(p-p_i).n() \times v_{mi} - v_i}{\Delta t} \quad (2)$$

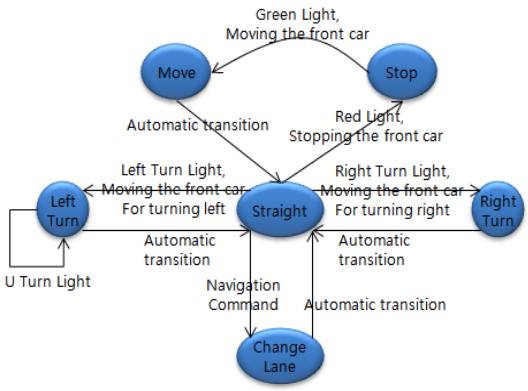
(2)에서 α는 $\alpha \le 1$으로 대부분의 경우 1의 값을 유지하나 특정 거리 이하의 경우 1보다 작은 값을 지정함으로써 자연스럽게 적은 힘으로 목표물에 접근하도록 한다. c_a는 다른 조종힘에 대한 '도착하기' 조종힘의 가중치 계수이다.

충돌을 회피하기 위한 힘은 [12,13,14]에서 제시한 분리 힘을 변경하여 사용한다. [12,13,14]에서는 모든 에이전트들의 크기가 일정하다고 가정하였다. 그러나 자율주행에 있어서 자동차들은 그 크기가 다를 수 있다. 따라서 충돌 회피의 조종힘을 (3)과 같이 정의 한다. (3)에서 음수 부호는 이 조종힘이 척력임을 의미한다. 여기서 p_j는 에이전트 j의 위치이고, c_c는 다른 조종힘에 대한 '충돌 회피' 조종힘의 가중치 계수이다.

$$f_c(i,j) = -c_c \frac{m_i m_j}{|p_i - p_j|^2} \quad (3)$$

3.2 액션 모듈의 설계

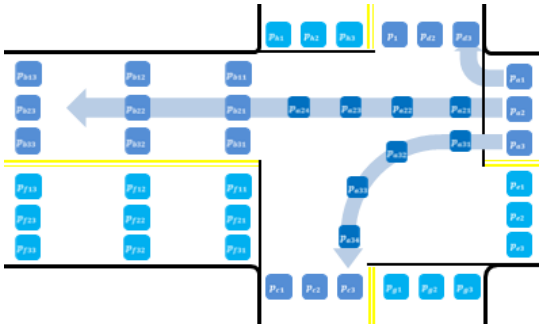
자율주행 자동차를 설계하기 위한 다양한 방법이 있겠으나 본 논문에서는 자율주행 자동차가 가질 수 있는 액션 모델을 유한 상태 기계로 설계한다. 자동차가 가질 수 있는 액션들을 하나의 상태로 설계하고 각 상태에서 다른 상태로 전이하는 조건을 정의하는 것이다. 자동차의 일반적인 행동을 고려할 때 자동차의 행동 상태는 '출발하기', '직진하기', '좌회전하기', '우회전하기', '차선변경하기', 'U턴하기'로 구분할 수 있다. 이들을 각각 '출발', '직진', '좌회전', '우회전', '정지'로 상태로 표시한다. 'U턴하기'는 좌회전 두 번하는 대체한다. 그림 2는 상태 다이어그램이다.



[Fig. 2] State diagram of the agent for the self-driving car

[Fig. 2]에서 하나의 상태에서 다른 상태로의 전이를 위한 조건은 중요하다. 먼저 '출발' 상태에서 '직진' 상태로의 전이는 내비게이션의 명령에 의한 '출발'이나 신호등의 감지에 의하여 출발이 일어난다. '출발' 상태에 도달하자마자 자동차는 항상 '직진' 상태로 자동으로 전이한다. '직진' 상태에서는 '좌회전', '우회전', '차선변경' 및 '정지' 상태로 전이할 수 있다. 먼저 좌회전인 경우 내비게이션의 명령을 받고 신호등이 좌회전 신호이면 좌회전을 수행한다. '좌회전' 상태에서 연속적으로 좌회전을 내비게이션이 명령하는 경우 이것은 U턴이 된다. 좌회전을 완료하면 '좌회전' 상태에서 '직진' 상태로 자동으로 전이한다. 우회전은 좌회전과 동일하다. '직진' 상태에서 '차선변경' 상태로 전이할 수 있다.

차선 변경 후 자동으로 '직진' 상태로 전이한다. 마지막으로 '직진' 상태에서 '정지' 상태로 전이할 수 있다. '정지' 상태로의 전이는 최종 목적지에 도착하여 내비게이션의 정지 명령에 의하여 정지하는 경우와 신호등에 의하여 정지하는 경우 그리고 앞차의 정지에 의하여 정지하는 경우이다. '정지' 상태에서는 '출발' 상태로만 전이할 수 있는데 내비게이션에 의한 명령이나 신호등의 인식에 의하여 '출발' 상태로 전이한다.



[Fig. 3] Location information stored in the road

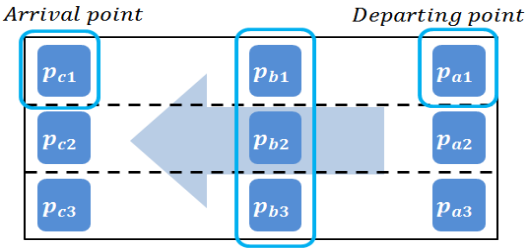
3.3 기본 맵의 설계

맵은 도로와 건물로 구성되며 또한 가로, 세로 일정 간격의 그리드로 나누어진다. 이것은 실제 지구의 위도, 경도와 같은 의미이다. 그림 3에서처럼 맵 상에서 도로들은 모든 차선에 대하여 일정 간격으로 위치정보가 저장되어 있다. 특히 교차로에서는 좌회전, 우회전을 '찾아가기'로 이동 가능하도록 차선별로 곡률 반경에 따라 위치정보가 저장되도록 설계하였다. 따라서 임의의 지점에서 좌회전을 하는 경우 미리 저장된 좌표를 따라 '찾아가기' 조종힘을 발생하면 좌회전이 되는 것이다.

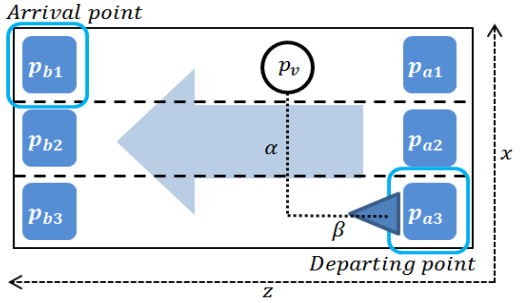
4. 프로토타입의 개발

4.1 기본 동작

'출발' 상태로 전이는 가장 단순하다. '정지' 상태에서 자동차를 특정 방향으로 $f_s(i,p)$를 적용한다. 여기서 p가 자동차가 출발하고자 하는 방향이다. '직진' 상태 전이도 '출발' 상태 전이와 동일하다. '정지' 상태로 전이는 단순히 주어진 위치로 향하여 $f_a(i,p,\alpha)$만 적용하면 된다. 여기서 p는 도착 지점을 의미하며 α는 얼마나 급하게 정지하여야 하는 가를 나타낸다.



[Fig. 4] An example for the car to go to straight



[Fig. 5] An example of a lane-changing car

[Fig. 4]와 같이 p_{a1}에서 p_{b1}를 거쳐서 p_{c1}를 가야 하는 경우를 고려해 보자. 이 경우 '직진' 상태로 전이하는 것은 $f_s(i,p_{b1})$의 힘을 적용하면 되고, p_{b1}을 통과하는 지점에서는 '직진' 상태를 유지하는 것도 $f_s(i,p_{c1})$의 힘을 적용하면 된다. 따라서 자율주행 자동차 에이전트의 의사결정 모듈은 인식모듈을 이용하여 매순간 GPS로부터 현재의 좌표를 인식하고, 맵에서의 좌표를 비교함으로써 현재 어느 지점을 통과하고 있는지를 판단할 뿐만 아니라, 지시한 명령이 제대로 수행되었는지를 판단한다.

'차선변경' 상태로의 전이는 차선변경이 필요할

때만 발생한다. [Fig. 5]에서와 같이 현재 p_{a3}에 있는 에이전트가 p_{b1}이 있는 지점까지 가면서 차선을 이동해야 한다고 가정하자. 가장 단순한 방법은 $f_s(i, p_{b1})$을 적용하면 된다. 그런데 $|p_{b1} - p_{a3}|$이 차선 변경에 적절한 거리이고, 주변의 장애물이 없다면 이 방법도 가능하나 $|p_{b1} - p_{a3}|$가 매우 크다면 매우 긴 거리를 차선을 가로질러 이동하는 것이므로 비현실적이다. 자연스러운 차선변경을 위해 자동차 에이전트는 p_{b1}로 '찾아가기'를 하는 것이 아니라 그림 5에서와 같이 일종의 가상 지점(p_v)을 생성하여 '찾아가기'를 한다. 일관성 있는 계산을 위하여 자동차가 지나가는 차선의 방향 z으로 변환하여 가상 지점을 계산한다. 이때 $p_v.x$는 차선폭의 배수가 되며, $p_v.z$는 현재 이동 중인 에이전트의 속도가 반영되어야 한다. 즉, 매우 빠른 속도로 진행하는 자동차와 느린 속도로 진행하는 자동차에 대해서 $p_v.z$을 다르게 주어야 할 것이다. 본 논문에서는 자동차의 속도에 비례하는 값으로 주었다.

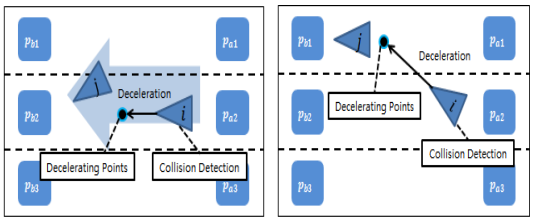
$$(p_v.x,\ p_v.z) = (p_{as}.x + c_1|p_{bt}.x - p_{as}.x|,\ p_{as}.z + c_2|v_i|) \quad (4)$$

(4)에서 p_{as}는 현재 에이전트의 위치이고, p_{bt}는 차선변경을 하고자하는 차선의 임의의 지점이다. 그림 5에서 이들은 각각 p_{a3}와 p_{b1}에 해당된다.

교차로에서는 우회전 또는 좌회전이 필요하다. 의사결정 모듈의 내비게이션은 적절한 지점에서 우회전 또는 좌회전을 할 수 있도록 이미 차선 변경 명령을 하달하여 에이전트는 좌회전 또는 우회전 가능한 차선에 진입하였다고 가정한다. 교차로에는 그림 3과 같이 회전에 적합하도록 차선별 좌표가 저장되어 있도록 하였다. 따라서 내비게이션은 회전을 위하여 순차적으로 저장된 좌표에 따라 '찾아가기' 힘을 발생하도록 한다.

4.2 충돌 예측 및 처리

자율주행 자동차 에이전트의 충돌 검출은 매우 중요하다. 인식 모듈은 레이저, 카메라 등 다양한 장치를 사용하여 실시간으로 장애물을 인식하여 의사결정 모듈에 전달한다. 의사결정 모듈은 이를 바탕으로 충돌을 예측하여 속도를 제어하여 충돌을 방지한다. 모든 충돌 검출은 자동차 에이전트의 전방만 검출을 시도한다. 충돌검출 대상은 타겟 에이전트로부터 반경 $v_i + dist(safe)$의 원 안에 있는 전방 에이전트들이다. 기본적으로 속도에 비례함으로써 속도가 빠른 에이전트는 큰 범위에서, 속도가 느린 에이전트는 작은 범위에서 대상 에이전트들을 찾도록 하였다. 여기서 $dist(safe)$를 더한 이유는 속도가 0인 상황에서 충돌검출 범위가 0이 되버리기 경우를 막기 위함이다.



(a) (b)

[Fig. 6] Two cases of a side collision of the agent

정면충돌의 경우는 매우 간단하다. 동일 차선에서 앞서가는 에이전트와 충돌이 예측되는 경우 이 에이전트가 충돌범위를 벗어날 때까지 $f_c(i, j)$을 적용한다. 여기서 j는 앞서가는 에이전트이다.

측면 충돌은 [Fig. 6]에서와 같이 자동차 에이전트가 차선변경을 시도하거나 다른 에이전트가 차선변경을 시도할 때 발생한다. 먼저 그림 6-(a)와 같이 다른 에이전트가 차선 변경의 목적으로 에이전트의 진행 차선으로 접근하는 경우이다. 그림 6-(a)에서 볼 수 있듯이 에이전트 i는 차선을 유지하며 주행하고 있으며 에이전트 j는 차선을 바꾸는 중이다. 이 때 자동차 i는 j에 대해 충돌검출을 인식하여 부딪치지 않도록 $f_c(i, j)$를 적용하여 감속한다. 그림 6-(b)의 경우 에이전트 i가 차선을 바꿔 주행하고 j는 차선을 유지하며 주행하고 있다. 이 때 에이전트 i는 전방에 있는 에이전트 j에

대해 충돌검출을 인식하여 차선 변경을 포기하고 에이전트 j가 충돌범위에서 벗어날 수 있도록 $f_c(i,j)$를 적용하여 감속한다.

4.3 내비게이션

자율주행 자동차에 대한 완벽한 내비게이션을 개발한다는 것은 자율주행의 모든 경우를 고려하여야 하므로 매우 복잡한 것이다[3]. 이러한 이유로 프로토타입의 개발에서 내비게이션은 기능을 단순하게 개발되었다.

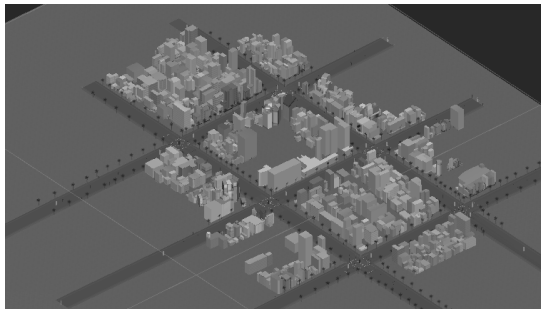
개발된 내비게이션은 출발지와 목적지가 정해지면 먼저 A^*알고리즘[16]을 적용하여 경로를 찾는다. 이 경로는 에이전트가 지나갈 좌표들의 목록이다. 기본적으로 내비게이션은 인식 모듈로부터 입력받는 현재의 위치와 탐색한 경로를 비교하여 순차적으로 행동제어 모듈에게 명령을 하도록 개발하였다. 차선 변경은 오직 좌회전, 우회전을 위한 사전 액션으로만 제한하였고, 좌회전, 우회전 명령 시점도 주변 상황의 고려 없이 교차로로부터 일정한 거리에서 일어나도록 개발하였다. 또한 예외적인 상황으로 주변 에이전트들이나 신호등에 대한 정보를 입력받아 충돌 예측 및 처리를 하거는 '정지' 상태 및 '출발' 상태의 전이를 하도록 간략화하여 개발하였다.

5. 시뮬레이션 및 성능 측정

5.1 시뮬레이션 환경

앞장에서 개발된 프로토타입은 그림 1의 구조하에서 유니티3D를 사용하여 구현되었다. 시뮬레이션을 위하여 Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz 64bit 컴퓨터를 사용하였다.

시뮬레이션을 위한 도로 맵은 그림 7과 같이 구현하였다. 그림에서와 같이 5개의 3차원 도로와 6개의 교차로를 구성하였다. 또한 약 440개의 빌딩을 포함하였으며 빌딩으로 인한 폴리곤 수는 약 693.8k 정도이다. 도로 및 빌딩은 많은 데이터를 포함하고 있기에 속도 문제가 있어서 무작정 크게 만들 수가 없다. 그림 7에서 구현된 도로 맵은 시뮬레이션을 위한 최소한의 크기이다. 구현된 맵은 1000×1000의 격자로 나누어져 위치 좌표를 나타낸다.



[Fig. 7] An implemented map for simulation

자동차 에이전트는 구현의 단순함을 위하여 모두 동일한 성능을 가진 단일 모델을 적용하였다. 렌더링 비용을 최소화하기 위하여 자동차 모델은 63개의 폴리곤으로 구성하였다. 기본적인 시뮬레이션에서는 10대의 자동차 에이전트를 사용하였다.

5.2 시뮬레이션 결과 및 논의

시뮬레이션에서는 [Fig. 7]의 도로 환경에서 입력키를 누르면 출발지와 목적지를 임의적으로 생성한 후 하나의 에이전트를 생성하여 자율운행 하도록 하였다. 하나의 에이전트는 출발지에서 생성하여 목적지에 도착하면 자동적으로 다시 출발지에서 출발하도록 하였다. 시뮬레이션은 이렇게 각각 다른 시점에 생성된 10대의 에이전트를 자율주행 시키면서 기본 동작, 충돌 예측 및 처리, 내비게이션의 동작을 관측하는 것이다. 시뮬레이션 결과 매우 사실적으로 에이전트가 이동하는 것을 관찰 할 수 있었다. [Fig. 8]-(a)는 자율 주행 자동차의 1인칭 관점에서 본 스크린 샷이다. 운전자 관점에서 바라보는 시각이 이와 같게 된다. 향후 본 논문을 확장하여 운전 연습용 시뮬레이터나 게임으로 확장할

경우 사용자는 이와 같은 화면에서 운전하게 될 것이다. [Fig. 8]-(b)는 자율 주행 자동차의 3인칭 관점에서 본 스크린 샷이다. [Fig. 8]-(b)에서 하얀색 박스는 [Fig. 3]에서 보인 교차로에서의 경유 지점을 표시하는 것이다. 실제 시뮬레이션에서는 이러한 경유 지점은 안보이게 된다.

(a) (b)

[Fig. 8] Screen shots of simulation

프로토타입 개발의 중요 목적은 향후 자율주행 자동차의 시뮬레이션에 있어서 본 논문에서의 설계와 같은 경우 사실적 시뮬레이션이 가능한 것인지를 사전에 알아보는 것과 기존의 컴퓨팅 파워로 얼마나 대규모의 시뮬레이션이 가능한지를 예측해 보는 것이다.

먼저 첫 번째 목적을 위하여 프로토타입에서 직전, 좌우회전, 차선변경 및 충돌검출을 구현하였고 시뮬레이션 하였다. 본 논문에서는 에이전트의 이동을 위하여 오직 조종힘 만을 적용한 시뮬레이션 결과는 기본적인 기능에 있어서는 사실적 상황과 매우 유사하였다. 또한 '찾아가기', '도착하기', '충돌 피하기' 등 각 조종힘별로 적절한 가중치(c_s, c_a, c_c)를 적용하는 것이 사실감 있는 시뮬레이션 결과에 많은 영향을 미치는 것을 알 수 있었다. 이러한 적절한 가중치를 찾는 것이 단일 모델의 자동차만 존재하는 경우에는 비교적 어렵지 않으나 다양한 자동차가 혼재되어 있어 에이전트의 최대 속도, 중량 등이 다른 경우 최적의 가중치는 많은 시행착오를 하면서 찾아야 할 것이다. 또한 프로토타입에서는 매우 단순한 내비게이션 기능만 구현하였다. 그러나 내비게이션은 의사 결정 모듈의 핵심 기능이고, 에이전트의 모든 행동을 제어하기 때문에 보다 돌발 상황 등 다양한 경우를 포함하여 개발되어야 함을 알 수 있었다.

[Table 1] FPS vs Number of Agents

에이전트 수	1	2	4	8	16
FPS	109.2	108.8	99.7	48.2	6.7

다음은 두 번째 목적을 위하여 기본적인 성능을 측정하였다. [Table 1]은 도로 맵을 변경하지 않는 상태에서 자동차 수를 증가시키면서 FPS(Frames per second)를 측정한 결과이다. [Table 1]에서 볼 수 있듯이 에이전트 수가 증가함에 따라 급속히 FPS가 저하되는 것을 알 수 있었다. 특히 에이전트의 수가 16개 이상으로 증가하면 FPS가 10이하로 급격히 감소함을 측정할 수 있었다. 이 시뮬레이션 결과를 고려한다면 16대 이하의 에이전트로 시뮬레이션 하여야 적절한 FPS를 기대할 수 있다는 것으로 이것은 사실적 시뮬레이션에 크게 미치지 못하는 결과이다. 이것은 더 큰 성능의 컴퓨터를 사용하든지 아니면 보다 성능이 좋은 엔진을 사용하여 개발하여야 한다는 것을 의미한다. 유니티3D 프로파일러를 사용하여 CPU 시간을 측정한 결과 자동차 수가 5일 때 연산 시간과 렌더링 시간이 약 1.5:1 비율로 나타났으나 에이전트 수가 증가함에 따라 연산 시간이 급격히 증가하였다. 이것은 의사 결정 모듈의 연산 량과 자동차간 충돌 검사 량이 급격히 증가하기 때문이다. 이 문제를 해소하기 위하여 보다 효율적인 알고리즘을 개발해야할 뿐만 아니라 연산 부분을 병렬로 처리하는 것도 고려할 수 있을 것이다.

6. 결 론

자동차 산업에서는 미래 자동차의 결정판이라 할 수 있는 자율주행 자동차에 대한 연구가 활발하다. 본 논문에서는 이러한 자율주행 자동차를 시뮬레이션 할 수 있는 자율주행 자동차 에이전트를 설계하고 이에 대한 프로토타입을 개발하였다.

프로토타입 개발의 중요 목적은 향후 자율주행 자동차의 시뮬레이션에 있어서 본 논문에서의 설계로 사실적 시뮬레이션이 가능한 것인지를 사전에 알아보는 것과 기존의 컴퓨팅 파워로 얼마나 대규모의 시뮬레이션이 가능한지를 예측해보는 것이다. 프로토타입에 대한 개발이기 때문에 자율주행 자동차의 핵심 기능만 개발하였고 보행자, 장애물 등에 의한 돌발 상황에 대한 고려는 제외하였다. 프로토타입의 구현은 유니티3D로 구현하였다. 개발된 프로토타입을 이용한 시뮬레이션 결과 조종힘만을 적용한 자동차 에이전트에 대한 설계는 타당하나 에이전트의 특성이 다양해 질 경우 각 조종힘에 대한 최적의 가중치를 찾는 어려움이 있을 것으로 나타났으며, 대규모 에이전트를 사용하는 경우 성능 문제가 심각해질 것으로 나타기에 기존의 범용 엔진보다는 전용 엔진 등을 개발하여야 함을 알 수 있었다.

향후 연구 내용으로는 대규모 자동차 에이전트를 대상으로 하는 시뮬레이션에서 성능 향상을 위한 연구가 필요하며, 또한 현실처럼 다양한 종류의 자동차 에이전트로 확장하는 연구가 필요하다. 이러한 문제를 해결하는 하나의 방법으로 병렬 처리에 대한 연구도 필요하다. 또한 본 논문에서 개발한 자율운행 자동차에 수동운행 자동차를 더함으로써 운전 연습용 기능성 게임으로 쉽게 확장될 수 있고, 자동차 운행이 포함된 게임에 본 논문의 기술을 적용하는 경우 훨씬 사실감 있는 게임이 될 것이다.

ACKNOWLEDGMENTS

본 연구는 한성대학교 교내연구비 지원과제임.

REFERENCES

[1] Jae Kwan Lee and In-sik Lee, “Intelligent Advanced Safety Vehicle Technology Development”, Auto Journal, 28(4), pp. 22-27, 2015.

[2] http://www.google.com/about/careers/lifeatgoogle/self-driving-cartest-steve-mahan.html

[3] WEI, Junqing, et al. A behavioral planning framework for autonomous driving. In: Intelligent Vehicles Symposium Proceedings, 2014 IEEE. IEEE, pp. 458-464, 2014.

[4] Kim, Jung-Min, et al. “Path-planning using Modified Genetic Algorithm and SLAM based on Feature Map for Autonomous Vehicle.” Journal of Korean Institute of Intelligent Systems 19.3, pp. 381-387, 2009.

[5] D. I. F. Ferguson and D. A. Dolgov, “Modifying behavior of autonomous vehicle based on predicted behavior of other vehicles”, Pat. no. 20130261872A1, United States, 2013.

[6] Martínez-Barberá, H., and D. Herrero-Pérez. “Programming multirobot applications using the ThinkingCap-II Java framework.” Advanced Engineering Informatics 24.1, pp. 62-75, 2010.

[7] Sales, Daniel O., et al. “Adaptive finite state machine based visual autonomous navigation system.” Engineering Applications of Artificial Intelligence 29, pp. 152-162, 2014.

[8] Wei, Junqing, et al. “A behavioral planning framework for autonomous driving.” Intelligent Vehicles Symposium Proceedings, 2014 IEEE. IEEE, 2014.

[9] I-navi navigation, http://www.inavi.com/Products.

[10] S. Habenicht, H. Winner, S. Bone, F. Sasse, and P. Korzeniet, “A maneuver based lane change assistance system”, IEEE Intelligent Vehicles Symposium, Germany, pp. 375-380, 2011.

[11] AZIMI, Reza, et al. STIP: Spatio-temporal intersection protocols for autonomous vehicles. In: ICCPS'14: ACM/IEEE 5th International Conference on Cyber-Physical Systems (with CPS Week 2014). IEEE Computer Society, pp. 1-12, 2014

[12] Reynolds, C. W., “Flocks, Herds, and Schools: A Distributed Behavioral Model”, SIGGRAPH, 21(4), pp. 25-34, 1987.

[13] Mat Buckland, “Programming Game AI by Example”, ISBN 1556220782, Wordware

Publications, 2005.
[14] Jae Moon Lee, “An Efficient Algorithm to Find k-Nearest Neighbors in Flocking Behavior”, Information Processing Letters, pp. 576-579, 2010.
[15] Kang, Shin-Jin, Jung Lee, and Soo-Kyun Kim. “High Density Crowd Simulation based on SPH.” Journal of Korea Game Society 11.6, pp. 193-199, 2011.
[16] Lee, Jae Moon, and Young Mee Kwon. “A Model of Pursuing Energy of Predator in Single Predator-Prey Environment.” Journal of Korea Game Society 13.1, pp. 41-48, 2013.
[17] SEET, Boon-Chong, et al. A-STAR: A mobile ad hoc routing strategy for metropolis vehicular communications. In: Networking 2004. Springer Berlin Heidelberg, pp. 989-999, 2004.

임 승 규(Lim, Seong Kyu)

2015년 한성대학교 멀티미디어공학과 재학

관심분야 : 게임프로그래밍, 인공지능

이 재 문(Lee, Jae Moon)

1986년 한양대학교 전자공학과(학사)
1988년 한국과학기술원 전기및전자공학과(석사)
1992년 한국과학기술원 전기및전자공학과(박사)
1994년-현재 한성대학교 멀티미디어공학과 교수

관심분야 : 기계학습, 게임프로그래밍, 감성컴퓨팅