韓國 컴퓨터情報學會 論文誌
第10卷 第1號, 2005. 3.

2005-3-1-1-16

레이싱 게임에서 순위 결정을 위한 퍼지 논리 아키텍처

이 세 일*

Fuzzy Logic Architecture for Deciding the Ranking at Racing Games

Se-Il Lee*

요 약

컴퓨터와 레이싱 게임을 하다보면 플레이어 자동차가 처음부터 끝까지 순위를 앞서가거나 퍼지카(FuzzyCar)가 매일 이기게 되면 대부분의 플레이어들은 몇 번만으로 그 게임에 대하여 금방 관심을 멀리 할 것이다. 이것을 해결하고 레이싱 게임에서 많은 재미를 위해서 무엇보다 중요한 것은 순위를 결정하는 일이다. 본 논문에서는 레이싱 게임에 재미를 주기 위해서 퍼지카를 만들어 플레이어와 대결하게 하였다. 선행 퍼지카는 플레이어 자동차보다 앞서 달리고 있기 때문에 뒤에 오는 플레이어 자동차의 속도와 거리의 변화로 인하여 플레이어 자동차와의 대상 행동을 식별하여 퍼지카가 기억하고 있는 기억 내용을 변경하지만 실제의 행동은 실행하지 않는다. 퍼지카가 의사결정을 하려면 타이머가 부여된 상태에서 기억한 내용을 가지고 순위 경쟁을 위한 행동을 한다. 또한 후행 퍼지카도 기억 내용은 다르지만 앞 내용과 같은 방법으로 작동한다. 실험에서는 실제의 값을 테스트 프로그램에 적용하여 순위 경쟁을 위한 결과를 도출하였다. 단순한 if-then 보다는 fuzzy logic을 이용한 방법이 퍼지카의 다양한 행동을 모델링한다는 것을 확인할 수 있었다.

Abstract

If a player car precedes from start to the end or a fuzzy car wins every day during computer or racing games, most players will lose their interest in the games soon after several times. In order to solve this problem and increase amusement at racing games, the more important thing than anything else is decide the ranking. In this thesis, in order to give amusement to racing games, the researcher made a fuzzy car and made it race with player cars. Because the preceding fuzzy car runs ahead of player cars, it can recognize their behaviors according to change of following player cars' speed and distance, and the fuzzy car changes its memory, but doesn't enforce actual behaviors. If the fuzzy car would make decision, it has to do behaviors to compete the ranking on the basis of the contents it has memorized under the situation where a timer is awarded. In addition, although an accompanying fuzzy car has different contents of memory, it is operated in the same way as mentioned above. At the time of experiments, the researcher applied the actual value to the test program and drew result for ranking competition. In conclusion, the researcher could confirm that we can have modeling of various behaviors by means of the method using fuzzy logic rather than simple if-then method.

▸ Keyword : fuzzy logic, racing game, FFLL, FCL

• 제1저자 : 이세일
• 접수일 : 2005.01.28, 심사완료일 : 2005.03.07
* 공주대학교 컴퓨터공학과 박사과정

Ⅰ. 서 론

네트워크 게임의 등장으로 많은 사람들은 지능이 떨어지는 컴퓨터를 상대하는 것보다는 똑똑한 인간과 게임을 즐기는 것이 좋다고 생각하여, 인공지능이 더 이상 게임에 필요없다고 생각하게 되었다. 그러나 게임을 하다보면 항상 사람과 함께 게임을 할 수 있는 조건이 존재하지 않기 때문에 사람을 대신할 똑똑한 인공지능 유닛이 점점 절실하게 필요를 느끼게 되었다[1, 2]. 특히 레이싱 게임에서 플레이어 자동차나 퍼지카가 일반적으로 이기게 되면 무척 재미없어 금방 식상하게 될 것이다.

본 논문에서는 레이싱 게임에서 자동차들의 순위결정을 위하여 퍼지 이론을 적용한 의사결정 아키텍처를 이용하여 좀 더 플레이어들이 재미있게 게임을 할 수 있도록 하였다. 실력을 가지고 있는 플레이어들에게는 속도 경쟁을 하고 운전 솜씨가 없는 플레이어들을 위하여 무한정 퍼지카를 기다리게 하면 현실성이 떨어지므로 퍼지카는 어느 정도 속도를 유지한다.

2장에서는 관련연구에 대하여 기술하고 3장은 퍼지 제어를 이용하여 순위 결정 방법과 의사결정 아키텍처를 기술하며, 4장에서는 실험과 결과를 보이고 5장은 결론과 향후 연구 과제를 제시한다.

Ⅱ. 관련연구

2.1 레이싱 게임의 역사

인공지능 게임의 시작은 1970년대 비디오 게임이 시작되면서 동전을 넣으면 작동되는 아케이드 게임을 위해서, 플레이어가 기계에 계속해서 동전을 넣도록 하는 것이 주 임무였다. 그 이후 인공지능 게임은 꾸준히 발전하기도 하고 때로는 무관심하기도 하였으나 3D 랜더링 하드웨어가 발전하고 그래픽의 질이 매우 발전하게 되자, 모든 관심은 인공지능이 게임의 주요 성공 요인으로 자리 잡게 되었다[3].

레이싱 게임은 스포츠 장르에 포함하기도하고, 하나의 독립된 장르로 드리이빙(Driving)게임이라고도 한다[4]. 레이싱 게임은 시뮬레이션성 레이싱 게임(그림 1)과 아케이드성 레이싱 게임(그림 2) 두 가지로 분류된다. 시뮬레이션성 레이싱 게임은 극도의 사실성과 현실감을 살리며 도로 상태나 타이어상태, 스티어링과 가속도 등 물리공식과 현실적인 인자들을 그대로 반영하기 때문에 현실감이 뛰어나지만 조작이 지나치게 어렵거나 박진감이 떨어진다는 평을 받을 때가 있다. 아케이드성 레이싱 게임은 달리는 재미에 더 초점을 맞추어 달리는 재미는 최고지만 현실과 거리감이 있다[5].

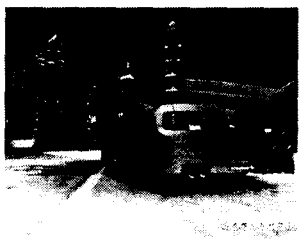

그림 1. 시뮬레이션성 레이싱 게임
Fig 1. Simulation type racing game

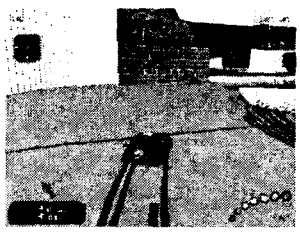

그림 2. 아케이드성 레이싱 게임
Fig 2. Arcade type racing game

2.2 퍼지 이론

컴퓨터가 인공지능을 가지고 인간이 원하는 바를 제대로 하기 위해서는 인간이 사용하는 숫자는 물론이고 애매한 표현도 처리할 수 있어야 하는데 퍼지 이론(Fuzzy Theory)은 이러한 애매한 표현을 처리할 수 있는 이론적인 바탕을 제공하는 것이다[6-8].

퍼지 이론에는 퍼지 집합, 퍼지 논리 제어, 퍼지 관계 등이 있으나 본 논문에서는 퍼지화(Fuzzification)와 역퍼지화(Defuzzification)에 관하여 간략하게 논한다.

2.2.1 퍼지화

입력과 출력값들을 그들의 소속함수(Membership Fuction)로 변경하는 과정을 말하며 퍼지화의 결과는 다른 퍼지 변수들의 다양한 소속도를 표현하는 도표의 집합이다[9-11].

예를 들어 자동차가 경기장 트랙의 Start Line에서 Finish Line까지 달릴 때 들어오는 초를 집합의 수로 나누어 놓는다. <표 1>은 자동차가 도착하는 초를 가지고 입력 범위를 표현했다.

표 1. "시간" 입력 범위
Table 1. Range of inputting "time"

시간(초)	값
빠름	0 ~ 21
보통	17 ~ 30
느림	25 ~ 40

(그림 3)은 퍼지 값들을 그림으로 표현하였다. 값들이 서로 겹쳐있는 것은 사람마다 "빠르다"나 "느리다"라는 생각이 다르기 때문에 범위가 조금은 다르게 표현되는데 이런 것들이 퍼지 이론이다.

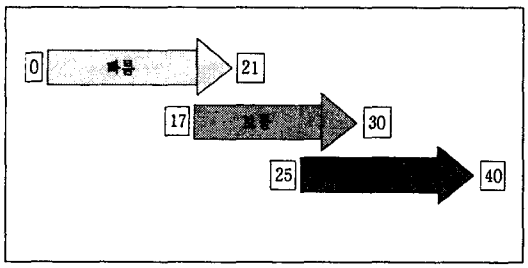

그림 3. 초를 나타내는 퍼지 값
Fig 3. Fuzzy value indicating second

각 값들은 달리기의 빠르기를 가지고 있고 소속도를 가지며 모델링된 소속함수를 수식으로 표현하려면 직선 위의 두 점을 표현하면 된다.

빠름(Fast) : (0, 1) (3, 1) (21, 0)
보통(Normal) : (17, 0) (23, 1) (30, 0)
느림(Slow) : (25, 0) (35, 1) (40, 1)

0~40까지 빠르기를 퍼지집합으로 나타내기 위하여 그에 맞는 정의를 삼각형, S자 모양, 사다리꼴, 직선 등의 다양한 모양으로 소속함수를 사용할 수 있으나 (그림 4)에서는 삼각형과 사다리꼴 모양으로 표현하고 있다. 그림 4에서 어떤 자동차가 경기장 크랙을 25.8초로 들어왔다면 '보통'에는 0.60 소속값을 가지며, '느림'에는 0.08의 소속값을 갖는 것을 보여 주고 있다.

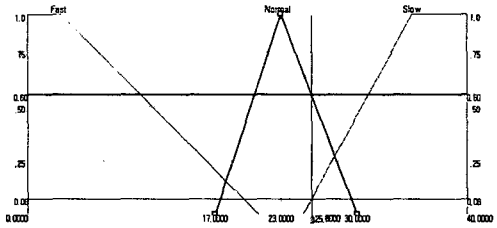

그림 4. 빠르기에 대한 퍼지 집합
Fig 4. Fuzzy sets for speed

2.2.2 역퍼지화

만들어진 퍼지 제어 시스템에서 퍼지 출력값을 실제로 사용하기 위해서 정확한 값으로 변경하는 것을 말한다. 역퍼지화 방법에는 무게 중심법(Center of Graity Method), 맘다니 추론법(Mamdani's Inference Method), 최대값 평균(Mean of Maximum) 방법이 존재한다[12].

2.3 FFLL

FFLL(Free Fuzzy Logic Library)는 오픈소스 퍼지 논리 클래스 라이브러리이며, 비디오 게임과 같이 속도가 중요한 응용 프로그램을 위해 최적화된 API이다. 속도를 위해서는 약간의 메모리를 희생하며 Lookup 테이블을 사용하고 있다. 또한 멀티쓰레드와 유니코드를 지원하며 전체 퍼지 논리 모델을 FFLL의 익스포트된 클래스를 만들 수도 있지만, FCL을 사용하여 퍼지 논리 모델을 생성하는 것이 가장 좋다. FFLL은 누구나 기능을 확장시킬 수 있으며 AI 엔진이나 프로토타입에 사용하게 되면 상당한 시간과 경제적인 도움을 받을 수 있다[13-15].

III. 순위 결정을 위한 퍼지 제어

레이싱 경기에서 플레이어가 운전하는 자동차는 퍼지카 앞에서 달리기도하고 뒤에서 따라 올수도 있다. 퍼지카가

항상 일등을 한다거나 플레이어가 운전하는 자동차가 항상 일등을 한다면 정말 재미없는 게임이 되기 때문에 게임의 재미를 위해서는 퍼지카의 속도를 적절히 조정해야 한다. 그런 속도 조절을 위하여 퍼지카에 퍼지 논리를 적용하였다. 선행하는 퍼지카는 언제 순위를 내주어야 하는지를 조절하여야 할 것이며 후행하는 퍼지카는 앞에 가는 차를 어떻게 앞서야 할 것인가를 조절해야 게임에 긴박감과 재미가 어우러질 것이다.

3.1 선행하는 퍼지카의 레이싱 퍼지 논리 집합

퍼지카가 플레이어 자동차를 앞서 가는 경우 여러 가지를 생각할 수 있으나 본 논문에서는 뒤차의 속도와 뒤차와의 거리를 생각하여 퍼지카의 속도를 어떻게 변경하여야 하는가를 구현한다.

뒤에 오는 플레이어 자동차의 속도를 〈표 2〉와 (그림 5)로 퍼지 집합을 나타내고 있다.

표 2. 플레이어 자동차의 속도
Table 2. Player cars' speed

속도	퍼지 집합
매우느림(Very_Slow)	0 ~ 50km
느림(Slow)	40 ~ 90km
보통(Normal)	80 ~ 120km
빠름(Fast)	110 ~ 160km
매우빠름(Very_Fast)	150 ~ 200km

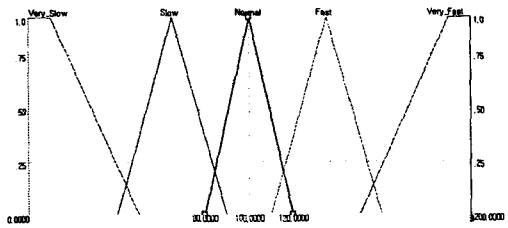

그림 5. 플레이어 자동차 속도의 소속도 집합들
Fig 5. Sets to which player cars' speed belongs

매우 빨리 달리는 자동차는 200km 이상달릴수도 있으며 그 값들은 모두 '매우빠름'에 속하게 된다. 마지막 집합에서 속도를 200까지 제한한 이유는 유한 집합을 구성하기 위해서이다.

〈표 3〉과 (그림 6)은 플레이어 자동차가 퍼지카를 어느 정도의 거리를 주는가에 따라 퍼지카의 속도 변화를 주고 있다.

표 3. 플레이어 자동차의 거리 변화
Table 3. Change of player cars' distance

거리	퍼지집합
매우멂(Very_Far)	퍼지카와의 거리 반(-)
멂(Far)	0보다작음(<0)
거리유지(D_M)	변화 없음(0)
가까움(Near)	0보다큼(>0)
매우가까움(Very_Near)	퍼지카와의 속도 반(+)

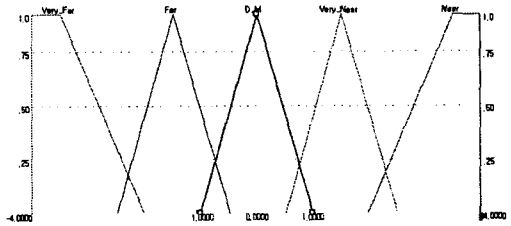

그림 6. 플레이어 자동차와 퍼지카의 거리 소속도 집합들
Fig 6. Sets to which distance of player cars and fuzzy cars belongs

〈표 2〉와 〈표 3〉을 조합하여 퍼지카는 표 4에 있는 퍼지 집합 중 하나를 선택하여 행동하게 되고 그 값을 그림(그림 7)으로 나타내고 있다.

표 4. 퍼지카의 속도 변화
Table 4. Change of fuzzy cars' speed

속도	퍼지 집합
매우저속(V_L_Slow)	현재속도의 1/4 수준으로
저속(L_Slow)	현재속도의 1/2 수준으로
평범(Common)	변함없음
고속(H_Slow)	현재속도의 1.25배 수준으로
매우고속(V_H_Slow)	현재속도의 1.5배 수준으로

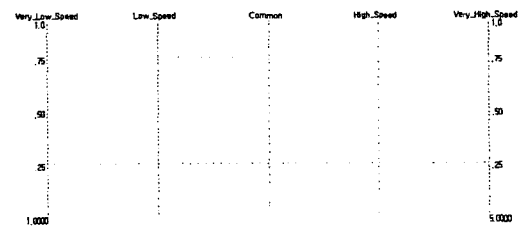

그림 7. 퍼지카의 값
Fig 7. Fuzzy cars' value

〈표 2〉와 〈표 3〉을 이용하여 표 4의 결과 중 하나를 선택하지만 그 결과를 얻기 위해서는 표 5와 같은 규칙 테이블을 가지고 있다. 규칙 테이블에서 플레이어 자동차의 속

도가 너무 느리면 퍼지카의 속도를 무한정 느리게 하여 플레이어 자동차를 항상 앞서게 한다면 게임의 재미를 반감시키게 되므로 퍼지카의 속도를 적정하게 유지한다. 그리고 플레이어 자동차가 매우 빠르게 이동할 경우 퍼지카의 속도도 매우 고속으로 증가하였지만 퍼지카의 속도를 1.5배 정도로 한정지어서 갑작스럽게 속도가 증가하는 것을 제안하여 재미를 추가 하였다. (그림 8)은 퍼지카의 속도를 어떻게 변경할 것인가를 규칙 맵으로 모여 주고 있다.

표 5. 선행 퍼지카의 규칙 테이블
Table 5. Table of rules for preceding fuzzy car

속도 \ 거리	매우멈	멈	거리 유지	가까움	매우 가까움
매우 느림	저속	평범	평범	평범	고속
느림	저속	평범	평범	고속	고속
보통	매우 저속	저속	평범	평범	고속
빠름	매우 저속	저속	평범	고속	매우 고속
매우 빠름	저속	평범	고속	매우 고속	매우 고속

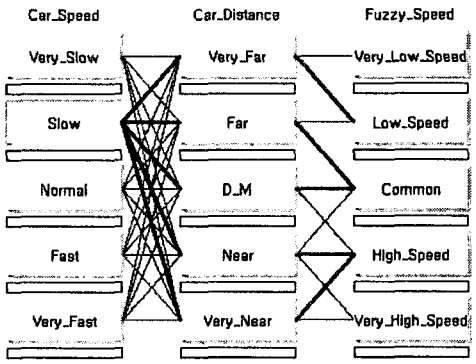

그림 8. 선행 퍼지카의 속도 규칙 맵
Fig 8. Map of speed rules for preceding fuzzy car

3.2 후행 퍼지카 레이싱 퍼지 논리 집합

퍼지카가 플레이어 자동차의 뒤를 쫓아갈 때는 앞서가는 플레이어 자동차와의 속도와 거리를 계산하여 퍼지카의 속도를 조절하여 플레이어 자동차와 적절한 레이싱을 하여야 좀 더 재미있게 된다.

퍼지카가 플레이어 자동차를 따라갈 때는 앞에서 사용한 속도와 거리는 같이 사용할 수 있지만 퍼지카의 속도 규칙 맵이 많이 틀려진다. (그림 9)는 퍼지카가 플레이어 자동차를 따라잡기 위한 속도 규칙 맵이다.

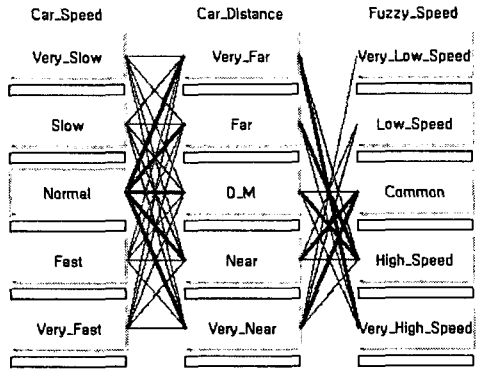

그림 9. 후행 퍼지카의 속도 규칙 맵
Fig 9. Map of speed rules for accompanying fuzzy car

후행 퍼지카는 어느 적절한 시기에 선행하는 플레이어 자동차의 속도가 떨어질 때 따라 붙고 고속이 되면 순위를 내주게 되어 좀 더 재미있는 게임을 만들 수 있다.

3.3 퍼지카의 의사결정 엔진 아키텍처

퍼지카가 의사결정을 하기 전에 현재의 환경을 인식하고 그에 대응하는 행동을 선택하는 아키텍처를 정의하여야 한다. 퍼지카는 다른 자동차들의 정보를 관리하는 자료구조를 가지며 다른 자동차의 행동변화에 따라 자료구조의 내용도 변하게 된다. 레이싱 중에 플레이어 자동차가 그의 환경과 상호작용하는 경우 그 결과를 퍼지카나 다른 플레이어 자동차들에게 전파되는 것을 퍼지카는 이벤트로 인식하고 있다. 퍼지카는 내부 기억에 들어있는 내용을 조회하여 이벤트출처 자동차와 대상 행위를 식별하여 해당 이벤트에 맞게 갱신한다. 퍼지카가 이벤트가 일어나는 것을 알게 되면 기억이 변경될 뿐 어떠한 행동이 일어나지 않는데 이것이 일반적인 이벤트 처리 기반 시스템과 다른 중요한 점이다. 퍼지카가 의사결정을 하려면 타이머를 두어야 한다[16]. 타이머는 이벤트와 독립적으로 발생하며 타이머의 제어점이 퍼지카에 돌아올 때 표 5와 같은 규칙 중 하나를 실행하게 된다. (그림 10)에서 ①에서 퍼지카는 현재 환경 변화를 이벤트로 받아들인다. ②는 이벤트의 발생으로 인한 환경에 대한 퍼지카의 기억을 변경한다. ③은 타이머에 의해 의사결정을 수행하고 ④에서 퍼지카는 의사 결정은 가지고 있는 기억에 접근한다. ⑤는 퍼지카가 실제로 의사결정을 할 행동을 결정하고 ⑥은 의사결정에 의해 실제의 속도 증가와 같은 환경이 변하게 된다.

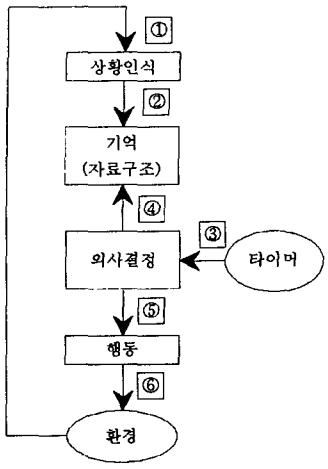

그림 10. 퍼지카의 의사결정 시스템
Fig 10. Fuzzy car's decision making system

Ⅳ. 실험 및 결과

본 논문의 실험은 Pentium 4, 2.4G, 256M에 Windows XP 버전의 운영체제에서 Visual C++ 6.0으로 실습하였다. 퍼지 논리 모델을 만들기 위해서 FCL을 사용하고 라이브러리는 FFLL를 이용하였다.

```
FUNCTION_BLOCK
VAR_INPUT
  Car_Speed          REAL; (* RANGE(0 .. 200) *)
  Car_Distance  REAL; (* RANGE(-4 .. 4) *)
END_VAR
VAR_OUTPUT
  Fuzzy_Speed  REAL; (* RANGE(1 .. 5) *)
END_VAR

FUZZIFY Car_Speed
  TERM Very_Slow := (0, 1) (10, 1) (50, 0) ;
  TERM Slow      := (40, 0) (65, 1) (90, 0) ;
  TERM Normal    := (80, 0) (100, 1) (120, 0) ;
  TERM Fast      := (110, 0) (135, 1) (160, 0) ;
  TERM Very_Fast := (150, 0) (190, 1) (200, 1) ;
END_FUZZIFY

FUZZIFY Car_Distance
  .
  .
  TERM Very_Near := (2.0, 0) (3.5, 1) (4, 1) ;
END_FUZZIFY
  .
  .
RULEBLOCK first
  .
  .
  RULE 0: IF Very_Slow AND Very_Far THEN Low_Speed;
  RULE 1: IF Very_Slow AND Far THEN Common;
  .
  .
  RULE 10: IF Normal AND Very_Far THEN
Very_Low_Speed;
  RULE 11: IF Normal AND Far THEN Low_Speed;
  .
  .
  RULE 24: IF Very_Fast AND Very_Near THEN
Very_High_Speed;
END_RULEBLOCK
END_FUNCTION_BLOCK
```

그림 11. 퍼지카의 행동 생성 FCL
Fig 11. Fuzzy car's behavior forming FCL

선행 퍼지카의 퍼지 집합을 이용하여 플레이어 자동차의 속도가 125km로 주행 중이고 퍼지카와 거리는 1을 유지하고 있을 때 결과를 출력하기 위한 (그림 11)의 FCL과 (그림 12)의 Visual C++ 6.0 프로그램의 일부분이다.

```
#include "FFLLAPI.h" // FFLL API
    .
    .
int main(int argc, char* argv[])
{
    .
    .
    int model = ffll_new_model();
    int ret_val = (int)ffll_load_fcl_file(model,

"\\ffll\\src\\after_fuzzycar_1.fcl");
    .
    .
    int child = ffll_new_child(model);
    .
    .
    ffll_set_value(model, child, CAR_SPEED, car_speed);
    ffll_set_value(model, child, CAR_DISTANCE,
car_distance);
    int output = (int)ffll_get_output_value(model, child);
    .
    .
    .
}
```

그림 12. Visual C++ 테스트 프로그램
Figure 12. Visual C++ test program

FCL에서 입력으로 "Car_Speed"와 "Car_Distance"를 사용하고 출력으로는 "Fuzzy_Speed"를 사용하고 있다. 또한 초기화 작업과 퍼지 집합을 생성하기 위한 초기값을 보여주고 있다. Visual C++ 테스트 프로그램에서 사용하고 있는 API 함수들의 목적은 다음과 같다.

표 6. API 함수들의 목적
Table 6. Purpose of API functions

API 함수	목적
ffll_new_model	퍼지 논리 모델을 포함하는 모델 객체를 생성
ffll_load_fcl_file	FCL 포맷으로된 파일로부터 퍼지 모델을 생성
ffll_new_child	모델을 위해 자식을 생성
ffll_set_value	자식의 입력 변수 값을 설정
ffll_get_output_value	자식을 위해 비퍼지화된 출력 변수를 가져옴

플레이어 자동차가 125km로 속도를 유지하면 퍼지카는 (그림 13)을 보고 퍼지 값을 얻어 기억내용을 변경하는 이벤트가 일어난다. 거리의 변화도 또한 (그림 14)에서 퍼지 값을 얻고 기억내용을 변경하는 이벤트가 발생한다. 두 가지 이벤트가 동시에 일어나거나 아니면 약간의 시간 차이를 두고 발생하였다 하더라도 타이머가 퍼지카에 부여되지 않으면 아무 일도 일어나지 않기 때문에 0.1초 간격으로 타이머를 조정하여 타이머를 부여하였다.

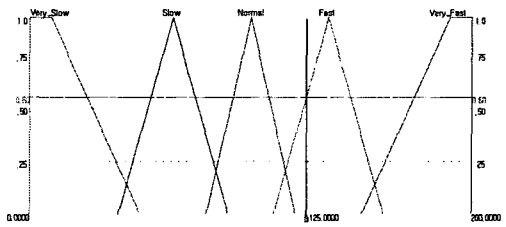

그림 13. 플레이어 자동차 속도의 예
Fig 13. Example of player cars' speed

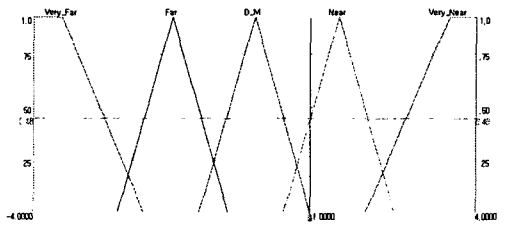

그림 14. 플레이어 자동차 거리의 예
Fig 14. Example of player cars' distance

타이머에 의해 의사결정 과정을 호출하여 수행할 행동을 〈표 5〉를 참고하여 퍼지카가 기억된 것이 선택되면 그림 15와 같이 속도를 "고속"으로 변경했다.

(그림 15)의 출력 변수는 "Fussy_Speed"이며 단일 선 모양 출력 집합을 가지고 있다. 단일 선 모양 출력 집합을 나타내기 위해 최대값 평균 방법을 사용하고 있다.

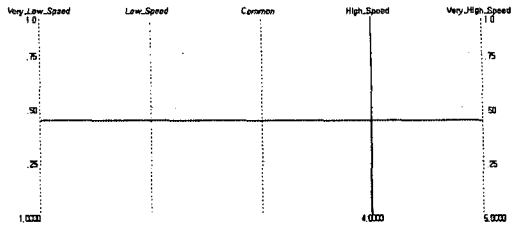

그림 15. 퍼지 집합의 출력 값
Fig 15. Output value of fuzzy sets

(그림 16)의 결과는 플레이어 자동차의 속도(125)가 빠른 속도를 유지하면서 퍼지카와의 거리(1)가 가깝게 변화게 되면 퍼지카의 행동은 지금의 속도보다 1.25배 빠른 속도를 증가하게 된다.

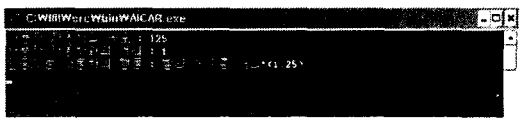

그림 16. 프로그램 테스트 결과
Fig 16. Test result of the program

(그림 17)은 플레이어 자동차의 속도와 거리 변화에 대한 비퍼지화 한 퍼지카의 속도 변화를 보여주고 있다. 플레이어 자동차와의 거리가 가깝고 속도도 증가하면 퍼지카도 같이 순위 경쟁을 하지만 멀리 있고 느리면 지루함을 없애기 위하여 플레이어 자동차보다는 조금 빠르게 움직이게 하였다.

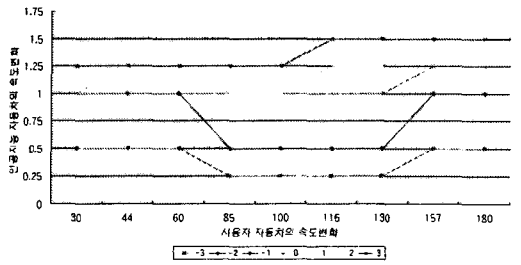

그림 17 플레이어 자동차의 속도와 거리 변화에 대한 퍼지카의 속도 변화
Figure 17. Fuzzy car's speed change as for player cars' change of speed and distance

V. 결론 및 향후 연구

앞에서 살펴본 것과 같이 레이싱 게임에서 퍼지카의 움직임에서 대하여 알아보았다. 선행 퍼지카는 뒤에 달리고 있는 플레이어 자동차의 속도와 거리를 살펴 퍼지카 자신의 기억 내용을 변경하게 된다. 또한 후행 퍼지카도 앞에 달리고 있는 플레이어 자동차 환경의 변화에 따라 기억 내용을 달리한다. 환경의 변화 하나하나를 이벤트라고 하였는데 이 벤트의 변화가 반드시 퍼지카를 움직이는 것이 아니라 타이머가 부여되어야만 퍼지카는 의사결정을 하여 기억에 접근하고 자신이 수행할 행동을 하게 된다.

일반 자동차에 퍼지 논리를 부여하니 다음과 같은 결과를 얻게 되었다.

우선, 퍼지카가 앞에서 레이싱하고 플레이어 자동차가 추월하기 위하여 속도를 증가하면 퍼지카도 속도를 증가하게 되지만 결국에는 플레이어 자동차가 승리하는 것을 볼 수 있다. 또한 레이싱이 서투른 플레이어 자동차에게는 조금씩 거리 차이를 늘리게 되는 것을 볼 수 있다. 퍼지카가 플레이어 자동차를 추격할 때도 비슷한 결과가 나오는 것을 알 수 있다. 앞의 내용으로 보아 플레이어 자동차가 레이싱을 잘하면 퍼지카의 순위는 뒤지게 되고 플레이어 자동차가 레이싱을 못하면 퍼지카의 순위가 앞서는 것을 알 수 있다.

앞에서 살펴본 것과 같이 퍼지카를 레이싱 게임에 접목시키면 좀 더 재미있는 게임이 될 것으로 생각한다.

향후 연구 과제는 현재 도로의 상태나 finish line과의 거리, 코너링, 장애물에 대한 관계 등을 추가한 퍼지카를 만들고자 한다.

참고문헌

[1] http://www.g-matrix.pe.kr

[2] 서정호, 정광호, 게임 인공지능의 형태와 앞으로의 발전 방향, 한국게임학회, 2002

[3] Steve Rabin, AI GAME PROGRAMMING WISDOM, 정보문화사, 류광 역, pp.61-67, 2003

[4] 김경석, 게임제작개론, 형설, pp.53-54, 2003

[5] http://www.khgames.co.kr

[6] 이상용, 인공지능, 상조사, pp.158-189, 2003

[7] 김연숙, 김창완, 퍼지 이론을 이용한 한국어 및 일어 화자 인식에 관한 연구, 한국컴퓨터정보학회, 2000

[8] 김도윤, 서재현, Neuro-Fuzzy를 이용한 이상 침입 탐지, 2004

[9] http://www.aistudy.co.kr

[10] Mark Deloura, GAME PROGRAMMING GEMS, 정보문화사, 류광 역, pp.416-428, 2001

[11] Mark Deloura, GAME PROGRAMMING GEMS 2, 정보문화사, 류광 역, pp.430-446, 2002

[12] 권일경, 이상용, 퍼지 플로킹 기반의 보이드 행동 모델링, 퍼지 및 지능시스템학회, 2004

[13] Steve Rabin, "AI Game Programming Wisdom", Chales River Media, 90-102

[14] O'Brien, Larry, "Fuzzy Logic in Games", Game Developer Magazine, April/May, pp. 53, 1996

[15] http://ffll.sourceforge.net

[16] Steve Rabin, AI GAME PROGRAMMING WISDOM, 정보문화사, 류광 역, pp.513-520, 2003

저자소개

이 세 일

1993년 대전공업대학교 전자계산학과 졸업(공학사)

2001년 청운대학교 대학원 전산전자정보공학과(공학석사)

2004~현재 공주대학교 대학원 컴퓨터공학과(박사과정)

〈관심분야〉 Ubiquitous Computing, Context Awareness, Collaborative Filtering, 게임 알고리즘