Experience on Running a Small-Size Simulated Car Racing Tournament in an Introductory Programming Course

Kyung-Joong Kim
Dept. of Computer Engineering
Sejong University, Seoul, South Korea
+82-2-3408-3838
kimkj@sejong.ac.kr

Sung-Bae Cho
Dept. of Computer Science
Yonsei University, Seoul, South Korea
+82-2-2123-2720
sbcho@cs.yonsei.ac.kr

ABSTRACT
This paper reports short-term experience on running a small-size car racing tournament in "Programming C Language class" for freshmen. It was based on Computational Intelligence and Games (CIG) 2009 Car Racing Competition software and rules with small modifications. Five students were involved in this small competition and they built their own controllers during this course. Although they don't have much experience on programming and computational intelligence (CI), the small competition makes them enjoy programming and understand the concepts of CI. In this paper, we'd like to introduce the competition, software used, modification to them, controllers developed, and results. This report will be useful to the CIG community which plan to use competition software for their class.

Categories and Subject Descriptors
I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems - Games

General Terms
Algorithms, Measurement, Performance, Human Factors

Keywords
Competition, Game, Car Racing, Education, Rule-based System

1. INTRODUCTION
Recently, there are a lot of game intelligence competitions organized by international conferences (IEEE Conference on Fuzzy Systems (FuzzIEEE), Computational Intelligence and Games (CIG), Genetic and Evolutionary Computation Conference (GECCO), Congress on Evolutionary Computation (CEC), World Congress on Computational Intelligence (WCCI), and Games Innovation Conference (ICE-GIC)) and they cover board games (Othello), car racing, unreal tournament, Super Mario Bros, and PacMan. They provide with library, open source code, and API of games to allow users build their own controllers or strategies. This gives great opportunity to the computational intelligence researchers which plan to apply their idea to the real-world applications.

Recent issue of IEEE Computational Intelligence (CI) Magazine was specialized to "Education" [1]. The main focus of this issue is to share educational ideas for the teaching of difficult concepts like CI. Robot soccer systems can be a good educational tool to teach and share ideas for students [2]. Also, there are several papers introducing case studies of CI courses taught at universities [3]. They assumed that students are undergraduates or graduates with some expert knowledge or experience on engineering, mathematics, or programming.

Is it reasonable to use the competition as a final project for an introductory programming language course? Most students learn a programming language for the first time and they're freshmen in the department of computer engineering. Usually, the attendants to the competitions are researchers or graduates with the expertise about programming language (C, C++, and JAVA) and computational intelligence techniques. For example, the best controller at WCCI 2008 Car Racing competition was built with evolutionary neural networks programmed by C++ [4].

Competitions may be the key to the potential of new knowledge and an attractive way of binding up technology and education [5]. Although the competitions require expert knowledge and programming skills, they can be good platforms for projects in a programming class. It motivates students to get better results from the competition and use their programming skills for practical purpose. The student's work can be evaluated objectively based on the ranks in the competition. It also attracts a lot of students to the computational intelligence society as future members.

In this semester, the second author offered a course entitled as "Software Agent" for graduate students. In the course, students made their own programs for Car Racing, PacMan, and Unreal Tournament. Their score was evaluated based on the results of the three tournaments. In the course, three or four students were grouped into a team and did the projects together. The number of students was 28 and all of them involved in the tournaments.

The first author offered a course entitled as "Programming C" for freshmen in the department of computer engineering. Based on the successful case of the "Software Agent" course, the first author decided to introduce the competition as one of the final projects for the class. Among 80 students from two classes, only five students choose the programming for competitions as a final project. This paper introduces the progress and results of this small competition with the novice computer programmers.

2. BACKGROUNDS

2.1 Artificial Intelligence for Car Racing

There are a lot of papers about computational intelligence for car racing. Most of them use point-to-point car racing simulator developed by Togelius [6]. Recently, several papers are published using The Open Car Simulator (TORCS) platform [4]. Recent competitions (WCCI 2008, CIG 2009, and GECCO 2009) are mainly based on the TORCS which provides realistic car racing simulation (Figure 1).

The recent car racing research was originated from simulated radio-controlled car racing used for CEC 2003, 2004, 2005 competitions. Since 2005, there are a lot of papers based on point-to-point car racing simulator developed by Togelius [7]. In [8], Togelius *et al*. addressed the case of two cars competing against each other on the same track at the same time in the evolution. In [9], Togelius *et al*. investigated the evolution of task-specific (operating well for a specific track) and robust (operating well for a large sets of tracks) neural controllers.

Figure 1. The TORCS game

In 2007, University Essex Group led by S. Lucas published several papers using the point-to-point car racing. In [10][11], genetic programming was used to evolve controllers for car racing. In [12], evolution and temporal difference learning (TDL) was compared in the context of simulated car racing environment. In [13], sensor data was predicted by neural networks trained by evolution and traditional back-propagation algorithm. In [14], tracks for car racing were evolved.

In 2008 and 2009, there are more than ten papers published by different groups (Japan, Singapore, UK and Switzerland). In [15][16], behavior-based approach based on fuzzy logic was proposed and won FuzzIEEE 2007 car racing competition. In [17], they design, build, and tune a fuzzy rule-based car controller for FuzzIEEE 2007 car racing using co-evolution.

In [18], the Ant Colony Optimization algorithm was used to reduce the lap time on a known track. In [19], adaptive controllers were proposed to profile the skill level of the opponent during game. In [20], they proposed a number of partially conflicting objectives in the car racing and used multi-objective optimization algorithm to yield pareto fronts of interesting controllers. In [21][22], fuzzy rule-based systems trained were used for car racing simulation. In [23], the results of CEC 2007 simulated car racing competition were reported in the international journals. In [24], neural network and behavior-based approaches were compared in the context of car racing simulation environments.

In WCCI 2008 competition, TORCS was introduced to the car racing competition and five controllers were submitted [4]. Reynolds *et al*. used the cultural algorithm to train controllers designed from scratch. Lucas modified the supplied sample controller based on the observation of behavior. The parameters were optimized manually because of time limit. Matt Simmerson used evolutionary neural networks called NEAT to drive simulated car and won the competition. Diego Perez *et al*. used rule-based controller [25] and K.C. Tan *et al*. used evolutionary strategies to optimize parameters. Recently, Cardamone *et al*. applied on-line neuro-evolution to the TORCS environments [26].

3. RUNNIG COMPETITION

3.1 Basic Information about This Course

The title of this course is "Programming C" offered to freshmen in the department of computer engineering at Sejong University located in Seoul, South Korea. There are two classes and each has 40 students. The purpose of this class is to teach basic grammar of C language and practice programming.

In the first day of class, students answered to the following questions: "Q1: Did you have any experience on programming?" and "Q2: Have you heard about C language?" Table 1 shows the answers from students. From this survey, 76% of students reported no experience on programming. 28% of students reported that it is the first time to hear the programming language C.

Table 1. Survey from students at the first day of class

	Yes	No
Q1	18 (23.7%)	58 (76.3%)
Q2	55 (72.4%)	21 (27.6%)

In the first half of this semester, they studied about variable, constant, if statement, control statements (for, while, do-while, and switch) and one-dimensional array. After the midterm exam, they studied about multi-dimensional array, function, structure, and pointer. Students have to start a small project after the midterm exam and finish it before the final exam. They have about 2 months for the small projects.

Because they were not yet experienced programmers, it was questionable whether they are able to do a small project related to the car racing. When students feel that the projects are too difficult one, it may result in losing interest in the first introductory programming course.

As a result, instructor decided to provide with three different types of projects and students can choose one of them as a final project. The first option is to solve 50 simple problems using C language (Type 1) provided by instructor. The second option is to make a small program (Type 2) by themselves without any restriction on the topics. The last one is to build a driver program for CIG 2009 car racing competition (Type 3). As expected, most students chose the first option for their project, because they thought the problem solving is an attractable one (Table 2).

Table 2. The choice of project type

	Number	Percentage
Type 1	60	75 %
Type 2	6	8 %
Type 3	5	6 %
None of them	9	11 %

It is interesting to see the profiles of students who choose the car racing as their final projects. Based on their ranks, they're not good at programming C. However, they have great passion on the programming car racing driver using C. All of them have no experience of programming at the first day of this course because they answered No for the Q1.

Table 3. Profiles of players for the car racing competition

Name (Initial)	Rank in Midterm Exam	Q1	Q2
MJKIM	46/80	No	Yes
CHPARK	69/80	-	-
SHKIM	39/80	No	Yes
SHKANG	61/80	No	No
JWPARK	77/80	No	Yes

3.2 The Preparation of Competition

This project was based on the CIG 2009 car racing competition. The first thing that the instructor did was showing the demonstration of car racing using the sample controller from the CIG 2009 competition site. Although this made student say "Wow," they didn't choose the car racing as their project because they feel it as one of the most difficult projects.

```
float angle = cs.getAngle();
float opponents[36];
for(z=0;z<36;z++) opponents[z]=cs.getOpponents(z);
float track[36];
for(z=0;z<19;z++) track[z]=cs.getTrack(z);
float trackPos = cs.getTrackPos();
```

Figure 2. An excerpt from the modified source code of SimpleDriver.cpp (It allows student do programming without C++ style grammar)

Initially, the competition package was written for C++ or JAVA programmers. Because the students have no knowledge on the object-oriented languages, it was modified to exclude C++ style sentences. Instead, the modified codes allow student do programming with only C style grammars. (Figure 2).

Also, instructor provided with step-by-step instructions on the programming with simulated car racing software. Although CIG 2009 has manuals for the software, it was too technical and also students have difficulty to read documents written in English. You can download the step-by-step guide from [27] (written by Korean).

There are a lot of well-made functions in the sample source code for autonomous driving: automatic transmission (gear change), steering (forcing the car to follow the middle lie of the track), acceleration and brake decision based on track sensors, checking stuck condition and recovery, and anti-lock braking system (ABS). Instead of removing all of them, the instructor removes only the acceleration and brake parts. You can download the modified SimpleDriver.cpp source code from [28].

3.3 The Submitted Controllers

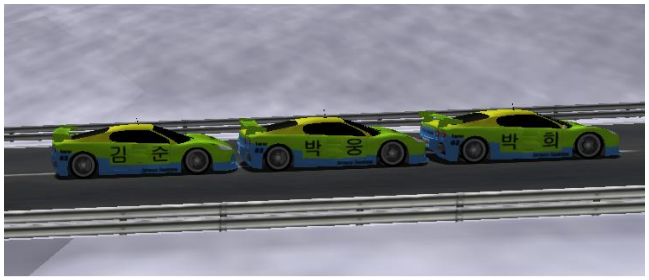


Figure 3. The tail of cars (The text in the side of cars is the name of students. The car's body image can be found in the sub-directory of TORCS. You can edit the image using the software like GIMP (Linux) and ACDSEE PHOTO EDIT (windows))

The instructor recommends students to use the opponent sensors in their final submission. Based on the rules of CIG 2009, if the car's damage is larger than the predefined level, it is removed from the race. It resulted in most of participants fail to complete the first lap. In our competition, old versions of server program (version champ2009win-1-2) which supports a command line option for ignoring the damage restriction (wtorcs.exe –nodamage). Finally, the instructor recommended students to modify their similar steering mechanism based on sample code to avoid the tail of cars (Figure 3). For the competition, a simple batch program was developed to copy each student's car skins and control programs to the TORCS directory and run the competition. It also automatically edits the championship2009server.XML. By doing this, the names in the rank list of TORCS were changed into the player's real name. You can get the software from authors upon request.

In the final competition, the widest one (E-Track 3) was chosen. The track was chosen in the competition day and students had no time to optimize their submission for the track. The track's width is 18 meter. Based on [4], the E-Track 3 has a good mix of shallow curves, tight curves, and straight sections. (Figure 4)

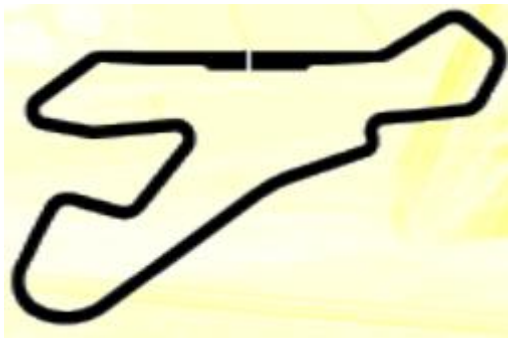


Figure 4. E-Track3

- CHPARK

His controller was composed of five rules based on the speed of cars and distance to track edge sensors. He mentioned that it was quite difficult to add more rules with track sensors and opponent sensors to the controllers initially made by scratch. Adding more rules resulted in low speed and rotation. Finally, he removed all other rules and submitted a controller with simple rules.

- MJKIM

He used more than 20 rules with opponent sensors, track position sensors, and distance to track edge sensors. For each track position (middle, right and left), he made multiple rules conditioned only by opponents sensors. He used opponent sensor #2, #9, #27, and #34. Based on the values of the opponent sensors, he changed the brake and steering angle. The parameters are manually tuned. The following is the one of rules used.

```
if (opponents[27]>1 && opponents[34]>3.5
   && opponents[2]>3.5 && opponents[9]>1){
    accel = 1.0f;  brake = 0.0f;
    targetAngle=(angle-trackPos*0.5);
    steer = (targetAngle)/0.785398;
}
```

- SHKIM

His controller is relatively simple. Although his controller is not too fast, his one is robust. Unlike other players, he changed the steering mechanism provided in the sample code (following middle of the track). This makes his controller avoid the tail of cars in the middle of track and pass other cars. The following shows the excerpt from his control codes.

```
if(steer >0 && t >= 9) ch=1;
if(steer <0 && t >= 9) ch=2;
if (t> 0.1 && t< 8.0 ) ch=4;
if (speedx <= 50) ch=3 ;
if(track[9] >50) accel = 0.8f;

switch(ch){
  case 1 : accel = 0.45f,brake= 0.1f;
  case 2 : accel = 0.45f,brake= 0.1f;
  case 3 : accel = 1.0f;
  case 4 : accel=1.0f;
}
float targetAngle=(angle-trackPos*0.5-0.3);
steer = (targetAngle)/0.785398;
```

- JWPARK

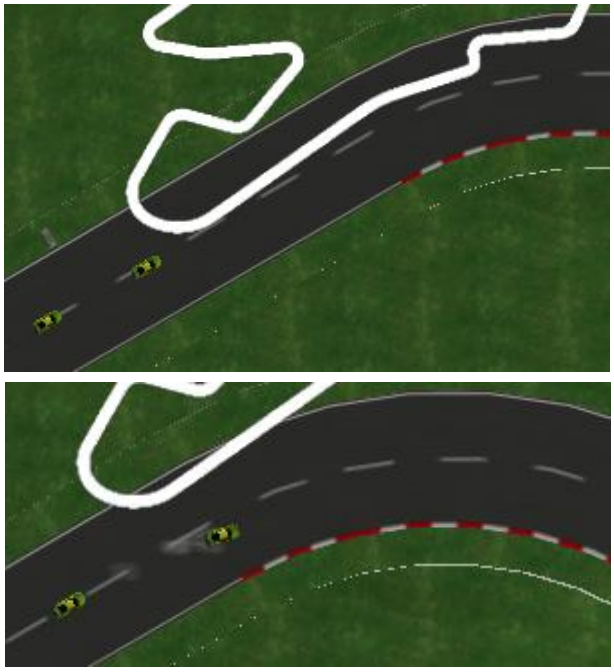
If the speed is below than 300km/h, the acceleration is 1.0. He used multiple rules based on track sensors and opponent sensors to change steering angle, and brake (Figure 5). His car showed very nice cornering and high speed driving. However, sometimes, the car stopped in the corner and moved very slowly. It seems that this is due to conflicts of multiple rules.

Figure 6 shows the cornering of JWPARK's controller. Before the corner, his controller follows the middle of the track. When the corner is close, his car changes the direction into the inside of corner minimizing the cornering distance. After finishing the cornering, the car returns to the middle of the track with small steering change.

```
if(speedx <300) accel = 1.0f; else accel=0.0f;
float targetAngle=(angle-trackPos*0.5);
steer = (targetAngle)/0.785398;
if (trackPos == 0 || -0.5< trackPos <0.5) steer = 0.0f;
if (trackPos<-0.5) steer = 0.25f;
else if (trackPos>0.5) steer = -0.25f;
if (track[4]>50) {brake = 0.1f; steer = 1.0f;}
 if (track[5]>50) {brake = 0.1f; steer = 0.88f;}
if (track[9]<78 && track[6]>75) {brake = 0.1f; steer = 1.0f;}
 if (track[9]<=72 && track[7]>80)  {brake = 0.1f; steer = 0.65f;}
 if (track[9]<=57 && track[8]>70)  {brake = 0.1f; steer = 0.55f;}
 if (track[9]<30 && track[8]>15 && track[10]>60) {brake = 0.3f;
steer = 0.8f;}
```

```
 if (track[9]<30 && track[8]>60 && track[10]>15) {brake = 0.3f;
steer = -0.8f;}
 if (track[8]<10) brake = 0.7f;
 if (track[9]<10) brake = 0.7f;
 if (track[10]<10) brake = 0.7f;
 if (track[9]<=50 && track[9]>20) {brake =0.3f; steer = 1.0f;}
 if (track[9]<=57 && track[10]>50) {brake = 0.1f; steer = -0.55f;}
 if (track[9]<=72 && track[11]>67)  {brake = 0.1f; steer = -
0.65f;}
 if (track[9]<78 && track[12]>62) {brake = 0.1f; steer = -1.0f;}
 if (track[13]>50) {brake = 0.1f; steer = -0.88f;}
 if (track[14]>50) {brake = 0.1f; steer = -1.0f;}
if (opponents[18]<20 && trackPos >0.5) steer = -0.1f;
 if (opponents[18]<20 && trackPos <-0.5) steer = 0.1f;
 if (opponents[9]<10) steer = -0.05f;
 if (opponents[27]<10) steer = 0.05f;
 if (opponents[0]<10) accel = 0.3f;
```

Figure 5. A controller source code of JWPARK



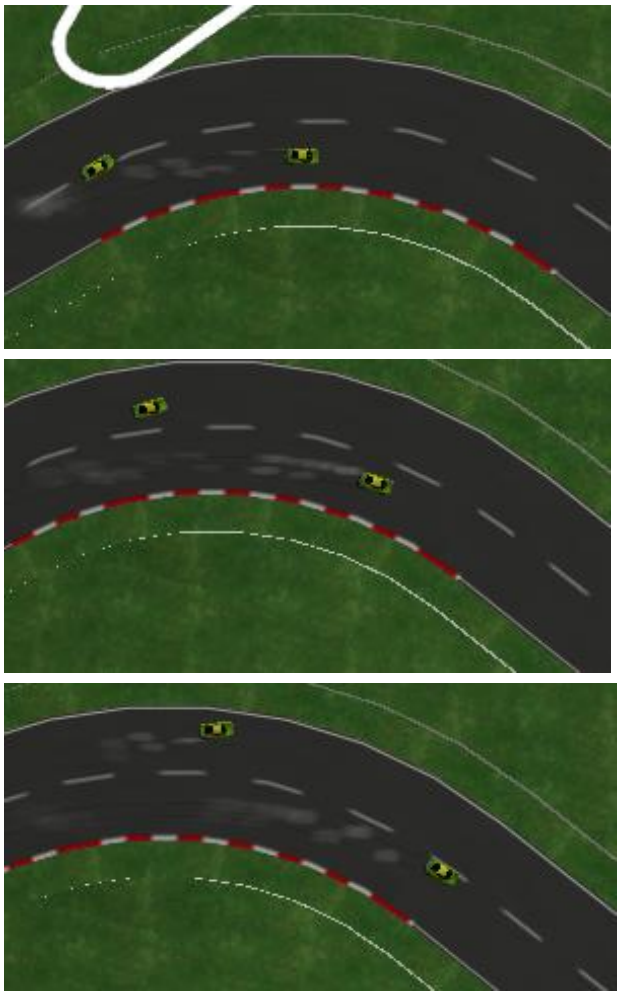


Figure 6. The cornering of JWPARK's controller (The front car is controlled by JWPARK' program)

- SHKANG

He tried to replace the original steering mechanism as a new one but failed to do that. As a result, his controller didn't work well.

3.4 Results and Discussions

The final racing was consisted of two stages. In the first stage, the participants race alone. In the second stage, all drivers race together. For each run, the starting position was determined randomly. Table 4 shows the results of the first stage. In this stage, it is important to drive the track as quickly as possible without considering other opponents. It is interesting that three of five drivers outperform the example program provided by the instructor.

Table 4. The result of the final racing alone (E-Track 3, 1 lap) (The bold one is the best)

	Time	Top Speed	Min Speed	Damages
JWPARK	**2:09:11**	224	-71	7485
CHPARK	3:12:03	150	-81	9254
MJKIM	3:13:16	220	-93	7486
EXAMPLE	3:13:19	136	-64	2053
SHKIM	3:13:54	153	-67	2771
SHKANG	OUT	OUT	OUT	10000

We used F1 scoring system: 1st rank = 10 points, 2nd rank = 8 points, 3rd rank = 6 points, 4th rank = 4 points, 5th rank = 2 points and 6th rank = 0 point. Although JWPARK did very well for 1st, 2nd and 3rd runs, he failed to move in corners in the last two runs. SHKIM always got high ranks for all runs.

Table 5. The final result of racing together (The bold one is the best in the run) (E-Track 3, 1 lap, 5 runs)

	1st run	2nd run	3rd run	4th run	5th run
JWPARK	**2:42**	**2:34**	**2:05**	+1 lap	+1 lap
SHKIM	+0:25	+0:45	+1:06	**3:06**	**3:09**
EXAMPLE	+0:26	+0:31	+1:33	+0:17	+0:18
CHPARK	+1:35	+1:03	+1:41	+0:45	+0:18
MJKIM	+l lap	+1:18	+1:13	+0:10	+2:13
SHKANG	+1 lap	+1 lap	+1 lap	+1 lap	+1 lap

	Score
JWPARK	34
SHKIM	**42**
CHPARK	20
MJKIM	22
SHKANG	0

After the final racing, JWPARK expressed his thought on this project.

Dear Instructor,

In this semester, the project is the most memorable thing. After choosing the racing as my project, I always think about the car racing. I did the project although my classmates told me "Idiot! Why do you choose such a difficult thing? Type I project is much easier than this car racing." Sometimes, there was no progress in my project although I did everything that I can do. At that time, I wish that I can solve the problem clearly. From this project, I learnt that the programming task is not a trivial one and needed professional skills. Although I repeated a number of tedious rebuilding, execution, and debugging cycles, I learnt a lot of things from this project. This was very exciting experience and finally I can laugh.

4. CONCLUSIONS

As mentioned before, the five participants were not familiar to programming but they did a great job in this car racing project. To realize this, we need some modification in the original competition software and additional small software written by authors. The provided software, manuals and the controllers by students can be good sources for the researchers plan to offer competition for students with less experienced programming skills.

Although students used only rules, they can build interesting and successful controller by themselves. From this project, they learned the difficulty of rule-based controls and the parameter turning. This motivates students to study more advanced computational intelligence techniques like fuzzy rule-based system, learning, search and adaptation algorithms.

In the next semester, the first author has a plan to teach C++ programming to the freshmen and intelligent systems to the senior. Although only the car racing is used for the project of this semester's course, other games (PacMan, and Unreal Tournament) can be good alternatives to attract more students into this kind of projects.

5. ACKNOWLEDGMENTS

This research was supported by the Original Technology Research Program for Brain Science through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0018948)

6. REFERENCES

[1] *IEEE Computational Intelligence Magazine*, February 2009.

[2] J.-H. Kim, Y.-H. Kim, S.-H. Choi, and I.-W. Park, "Evolutionary multi-objective optimization in robot soccer system for education," *IEEE Computational Intelligence Magazine*, pp. 31-41, February 2009.

[3] A. E. Smith, "Evolving an adaptive optimization course," pp. 52-54, *IEEE Computational Intelligence Magazine*, February 2009.

[4] D. Loiacono *et al*., "The WCCI 2008 simulated car racing competition," *IEEE Symposium on Computational Intelligence in Games*, pp. 119-126, 2008.

[5] V. Dagiene, "Information technology contests – Introduction to computer science in an attractive way," *Informatics in Education*, vol. 5, no. 1, pp. 37-46, 2006.

[6] J. Togelius, *Optimization, Imitation, and Innovation: Computational Intelligence and Games*, Ph.D. Thesis, Department of Computer Science, University of Essex, Sep 2007.

[7] J. Togelius, and S. Lucas, "Evolving controllers for simulated car racing," *IEEE Congress on Evolutionary Computation*, vol. 2, pp. 1906-1913, 2005.

[8] J. Togelius, and S. Lucas, "Arms races and car races," *9th International Conference on Parallel Problem Solving from Nature*, pp. 613-622, 2006.

[9] J. Togelius and S. Lucas, "Evolving robust and specialized car racing skills," *IEEE Congress on Evolutionary Computation*, pp. 1187-1194, 2006.

[10] A. Agapitos, J. Togelius, and S. Lucas, "Evolving controllers for simulated car racing using object oriented genetic

programming," *9th Annual Conference on Genetic and Evolutionary Computation*, pp. 1543-1550, 2007.

[11] A. Agapitos, J. Togelius and S. Lucas, "Multiobjective techniques for the use of state in genetic programming applied to simulated car racing," *IEEE Congress on Evolutionary Computation*, pp. 1562-1569, 2007.

[12] S. Lucas, and J. Togelius, "Point-to-Point car racing: An initial study of evolution versus temporal difference learning," *IEEE Symposium on Computational Intelligence in Games*, pp. 260-267, 2007.

[13] H. Marques, J. Togelius, M. Kogutowska, O. Holland and S. M. Lucas, "Sensorless but not senseless: Prediction in evolutionary car racing," *IEEE Symposium on Artificial Life*, pp. 370-377, 2007.

[14] J. Togelius, R. Nardi, and S. Lucas, "Towards automatic personalized content creation for racing games," *IEEE Symposium on Computational Intelligence and Games*, pp. 252-259, 2007.

[15] D. T. Ho, and J. M. Garibaldi, "A novel fuzzy inferencing methodology for simulated car racing," *IEEE International Conference on Fuzzy Systems*, pp. 1907-1914, 2008.

[16] D. T. Ho, and J. M. Garibaldi, "A fuzzy approach for the 2007 CIG simulated car racing competition," *IEEE Symposium on Computational Intelligence and Games*, pp. 127-134, 2008.

[17] S. Guadarrama, and R. Vazquez, "Tuning a fuzzy racing car by coevolution," *3rd International Workshop on Genetic and Evolving Fuzzy Systems*, pp. 59-64, 2008.

[18] L. delaOssa, J. A. Gamez, and V. Lopez, "Improvement of a car racing controller by means of ant colony optimization algorithms," *IEEE Symposium on Computational Intelligence and Games*, pp. 365-371, 2008.

[19] C. H. Tan, J. H. Ang, K. C. Tan, and A. Tay, "Online adaptive controller for simulated car racing," *IEEE Congress on Evolutionary Computation*, pp. 2239-2245, 2008.

[20] A. Agapitos, J. Togelius, S. Lucas, J. Schmidhuber and A. Konstantinidis, "Generating diverse opponents with multiobjective evolution," *IEEE Symposium on Computational Intelligence in Games*, pp. 135-142, 2008.

[21] S. Fujii, T. Nakashima, and H. Ishibuchi, "A study on constructing fuzzy systems for high-level decision making in a car racing game," *IEEE International Conference on Fuzzy Systems*, pp. 2299-2306, 2008.

[22] T. Nakashima, and S. Fujii, "Effect of opponent players on the performance of fuzzy decision making systems for car racing," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 1800-1805, 2008.

[23] J. Togelius et al. "The 2007 IEEE CEC simulated car racing competition," *Genetic Programming and Evolvable Machines*, vol. 9, pp. 295-329, 2008.

[24] H. Tang, C. H. Tan, K. C. Tan and A. Tay, "Neural network versus behavior based approach in simulated car racing game," *IEEE Workshop on Evolving and Self-Developing Intelligent Systems*, pp. 58-65, 2009.

[25] D. Perez, Y. Saez, G. Recio, and P. Isasi, "Evolving a rule system controller for automatic driving in a car racing competition," *IEEE Symposium on Computational Intelligence and Games*, pp. 336-342, 2008.

[26] L. Cardamone, D. Loiacono, and P. L. Lanzi, "On-line neuroevolution applied to the open car racing car simulator," *IEEE Congress on Evolutionary Computation*, pp. 2622-2629, 2009.

[27] http://dasan.sejong.ac.kr/~kimkj/C_2009/car_racing.pdf (Koran Document)

[28] http://dasan.sejong.ac.kr/~kimkj/C_2009/SimpleDriver.cpp