Students-Orientation Software-based Method for Learning Multi-Robot Collaborative Operations

Hsin-Hsiung Huang, Juing-Huei Su, and Chyi-Shyong Lee

***Abstract*—**This study defines a novel and interesting problem for the students of LungHwa University of Science and Technology to learn multi-robot collaborative operations. Besides, the training materials, which contain a series of exercises to learn and simulate that two cars move along their own line tracks.

The proposed study mainly focuses on the three topics. First, the lecture teaches with the ordering of the primary-level, middle-level and advance-level exercises, because this collision problem is very strange to the students of University of Science and Technology. Second, the easy-to-use one-dimension array is provided to design the collaborative operations for two line following robots instead of the complex data structures or theory. Third, both of the stop-then-start method and go-backward method are designed to handle the car collision or the future extensions. In fact, the car with low-priority must wait for the car with high-priority. In order to observe the collaborative operation step-by-step, the programming simulator is easily adjusted the delay time. After the training course, the questionnaires from students are positive that most students can learn materials and programming skills.

***Keywords*—**Student-orientation, software-based simulator, multi-robot, line following robot, collaborative operations, collision.

I. Introduction

Multi-robot collaborative operation is an interesting topic in the recent [1][6][7][8][9]. Many people pay more attentions to the development of the wheeled robots or microcontroller-based applications [2][3][4][5]. Some of them focused on the hardware circuits for educational purposes. Some papers discussed the control theory and implement to increase the performance of the robots. [6] described the complex implementations and several training courses for the multiple robots. KIVA system [7] has been commercially to arrange multiple robots cooperative operations to each others. Most of researches are investigated the hardware circuits, programming and control method. Some of them discussed the teaching materials for multiple robots. Students usually take a long time to learn the multiple robots. Because many physical robots and the control circuits are needed to learn and make the experiments [1][7][8][9].

This work was supported in part by the MOST 103-2511-S-262 -002 from Ministry of Science and Technology.

Hsin-Hsiung Huang, Juing-Huei Su, and Chyi-Shyong Lee are with the Lunghwa University of Science and Technology, Dept. of Electronic Engineering, No.300, Sec.1,Wanshou Rd., Guishan Shiang, Taoyuan County 33306, Taiwan (R.O.C) (phone:+886-2-82093211ext:5632, fax: +886-2-82095165, Corresponding e-mail: pp022@mail.lhu.edu.tw)

It is difficult to design the hardware circuit implementation for multi-robot, the proposed methods or algorithms are also the hard problems to students. Some researches explored the complex theory or algorithms to handle the collision avoidance [10][11][12][13][14]. However, the complex mathematics is not suit to the students of University of Science and Technology. The interesting and contest-based implementation sometimes helps the students a little and draws their attention during their learning procedure [15]. How to consider the students and design the teaching materials to increase the learning effectiveness is the most important event for the students of the University of Science and Technology.

The contributions of this study are as follows. First, the student-orientation training materials, which contain a series of the software-based exercises by C language [16]-[18], are presented to students of University of Science and Technology. Second, the easy-to-use concept, which is designed by using the one-dimension array, is provided to design the programming simulator to easily learn the collaborative operations for two line following robots. Third, not only one stop-then-start method is designed to avoid the car collision, but also the go-backward method is also designed to future extensions. In order to observe the collaborative operations step-by-step, the programming simulator can easily be adjusted the delay time.

The organization of this study is described as follows. Section II describes the preliminary, including the motivation of the software-based training tools, the objectives of training materials and the defined problem to solve by using the training materials. Section III then introduces the proposed method to training the programming skills for the collaborative operation of multiple line following robots without collision. Section IV summarizes questionnaire results from students after the training course. Finally, conclusions are drawn in Section V.

Table 1. Comparisons of the traditional method and the proposed software-based approach.

	Traditional method	Software-based approach
A car with high-priority	YES	NO
A car with low-priority	YES	NO
Sensor-based control circuits	YES	NO
A physical map for robot	YES	NO
A programming simulator	NO	YES

Table 2. Illustration of the topics and descriptions of the programming exercises.

NO.	Topics of exercises	Descriptions of exercises
EX1.1	delay-function	Design the sub-routine to setup the delay time with parameters.
EX1.2	move-by-counter	Handle the car to move forward by using the counting parameters.
EX1.3	array-operation	Use one-dimension array to record the value and reset the values.
EX2.1	one-car-move	Implement the programming codes to guide that one car moves along the line tracks.
EX2.2	detect-collision	Design a set of programming constraints to detect potential collision for two cars.
EX3.1	two-cars-move	Implement the codes to guide that two cars individually move along their own line tracks.

*EX1.1, 1.2, and 1.3 are primary-level exercises; EX2.1 and 2.2 are middle-level exercises; EX3.1 are the advanced-level exercise.

II. Preliminary

In this section, the motivation of this study, the objectives of the training course and problem definition, which is used to train the students about the multi-robot collaborative operation, are presented and discussed in details.

A. Motivation of the Software-based Training Materials

Originally, students designed the line following robots to move along their own line tracks in a MOST project. Collisions of two line following robots may be happen because two robots go to the same line track. Therefore, the additional sensor-based circuits are needed to handle this collision problem. In Fig. 1, two line following robots are abbreviated to cars A and B, respectively. In the study, car A is designed to be high-priority. It means that car A keeps on going and car B with low-priority needs to stop when two cars will be happened to the collision.

According to the experiences, the collision problem is difficult to the new students because the students should design the function of the line following robots. Moreover, the students also design their own control circuits which detect one line following robot and then transmit signals to the other robot.

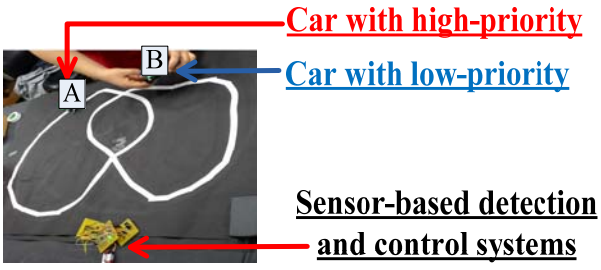


Fig. 1 It is *difficult* to learn by physical robot with *long* time

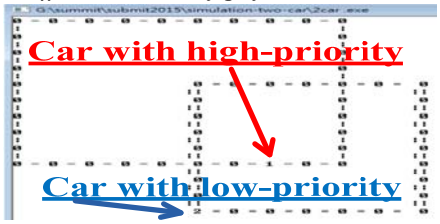


Fig. 2 It is *easy* to learn by software simulator with *short* time

From this experiment, we have an idea of providing software-based approach to help the development of multi-robots collaborative operations. Fig. 2 shows that symbol "1" denotes the car A with high-priority. Symbol "2" represents the car B with the low-priority and car B should stop if there is a car collision. To compare with Fig. 1 and Fig. 2, we know the physical experiments in Fig. 1 can be translated into the software-based simulator in Fig. 2. Similarly, the cars A and B in Fig. 1 are denoted to symbols "1" and "2" in Fig. 2, respectively. Instead of words "line following robot", the symbols "1" and "2" are denoted cars A and B through this whole study. Table 1 shows the comparisons of two methods.

B. Objectives of the Training Course

The objectives and topics of this training course are mainly summarized in Table 2. The exercises which are corresponding to the objectives are described in Table 3. Of course, the trained programming skills which are corresponding to the objectives are shown in Table 4.

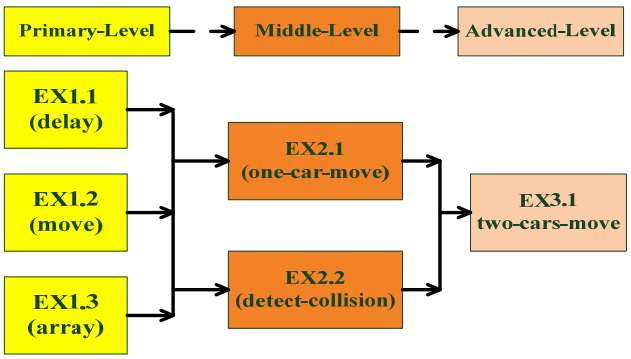


Fig. 3 Framework for the training course

The relationships of six exercises, objectives and skills are discussed in Fig. 3. According to above tables, six objectives of this training course are: 1)Learn to setup delay; 2)Learn to increase the counter; 3)Learn the array operation; 4)Integrate the simulator for one car; 5)Detect the collision for two cars, and 6)Design the simulator for two cars handling the collision.

Table 3. Training objectives and corresponding exercises.

Objectives of training course	Corresponding exercises
Learn to setup the delay time with parameters with the sub-routine.	EX1.1
Learn to move forward for a car.	EX1.2
Learn to operate the one-dimension array.	EX1.3
Learn to guide one car along the line tracks.	EX2.1
Learn to detect the collision for two cars at the same line track.	EX2.2
Learn to guide two cars individually along the line tracks.	EX3.1

Table 4. Illustrations of objectives and the corresponding skills.

Objectives	the corresponding skills
Learn to setup the delay time with parameters.	(1)Learn for loop; (2) Design input-output for the sub-routine.
Learn to move forward or go backward for a car.	(1) Increase the counter; (2) decrease the counter.
Learn to operate the one-dimension array.	(1)Initialize the values in the array; (2)Update the values in the array; (3)Reset the values in the array.
Learn to guide one car along the line tracks.	(1)Describe how one car moves along the line tracks; (2)Integrate the counter and array operation into programming simulator.
Learn to detect the collision for two cars at the same track.	(1)Describe the cause for collision of two cars; (2)Learn the if-statement to detect collision with the length of the about *d* grids.
Learn to guide two cars individually along the line tracks.	(1)Describe how two car move along their own line tracks; (2)Integrate the detected collision method, counter and array operation into programming simulator.

C. Problem Definition

Fig. 3 shows the framework for the multi-robot collaborative operation. The line following is denoted as the car in this paper. The programming simulator can guide two cars to move along their line tracks individually without collision. The robot does not explore the unknown environments because the one-dimension array is used to record the line track. Therefore, the problem discussed in this section is described as follows:

Given two cars, which are named as c_1 *and* c_2, *with their individually rectangular line maps* m_1 *and* m_2. *Line maps* m_1 *and* m_2 *contains the numbers of grids* s_1 *and* s_2, *respectively. Two cars are able to move with the speeds of* v_1 *and* v_2, *respectively. When two cars go through the dangerous regions, one car with high-priority keeps on moving and the one with low-priority needs to terminate. To avoid the car collision, the car with low-priority must terminate before the potential collision grid with about* d *grids. In this study, the backward operation* b *is also developed for the future extended purposes. The study attempts to guide two cars move along the line tracks without car collision at the dangerous regions while keeping the one car with high-priority to move on its line tracks.*

III. Method

In this section, all constraints, which described the operations for the car movement without collision (i.e. record array data, move along the line tracks in Sub-section A, and reset the counting for two cars in Sub-section B), car termination (i.e. delay function in Sub-section C), collision detection (i.e. detect before d grids and the car with low-priority terminates in Sub-section D), car backward (i.e. withdraw b grids for the car with low-priority in Sub-section E), are implemented. The corresponding programming constraints are also translated into the C language with the proper drawing pictures. To make the students understand easily, the procedure for cars collaborative operations are summarized and discussed in Sub-section F. The constraints are illustrated as follows.

A. Record Car Movement and Display for the Cars in Screen

The one-dimension array is used to record the current line track for each car. For example, the one-dimension array a[size] is used to record the current line track for the car A, where size is set to be 36 for the size of the rectangular shape. Similarly, the one-dimension array b[size] is used to record the current line track for the car B, where size is set to be 28 for the size of the rectangular shape. The initial values for the cars A and B are both located at line track with the indexes zero. The programming are described as : int a[36]={0}; and int b[28]={0}; , respectively. Moreover, a counter variable is also needed and used to count the line track for each car. For example, two variables cnt1 and cnt2 are used to denote the line tracks of the cars A and B, respectively.

Hence, the line track of car A is denoted by using the a[cnt1]=1, where cnt1 is used to represent the line track of car A and "1" is labeled to the symbol of car A (see Sub-section II.C). Similarly, the line track of car B is denoted by the b[cnt2]=2, where symbol "2" is denoted to car B. The programming codes for cars A and B are: a[cnt1]=1; and b[cnt2] =2; , respectively.

B. Guide Movement and Iteratively Reset the Line Tracks

For the initial condition, there is no car collision and two cars move along the line tracks. The variables counters, cnt1 and cnt2, are increased to one-by-one. The counter constraints for cars A and B are computed by cnt1++ and cnt2++, respectively.

The counting variables must reset to zero after the car moves along the line tracks with the size of the rectangles. The constraints for the cars A (counting variable is cnt1) and B (counting variable is cnt2) along the rectangles (size m_1 and m_2) can be described as follows:

if (cnt1>= m_1) cnt1 = 0; and *if* (cnt2>= m_2) cnt2 = 0;

In fact, for the rectangles with m_1 =36 and m_2 =28, the constraints can be formulated as follows:

if (cnt1>=36) cnt1 = 0; and *if* (cnt2>=28) cnt2 = 0;

In order to reset the screen, the values of the one-dimension array must be reset to zero. One *for*-loop can be used to initialize the values of one-dimension array, see Lines 2 and 4. For the cars A and B moving along the rectangles with size of m_1 and m_2 , these codes used to clear the screen are as follows:

```
1 for(int i=0;i< m1 ; i++)
2    {  a[i]=0;  }
3   for (int i=0;i< m2 ; i++)
4    {  b[i]=0; }
```

In fact, for the rectangles with m_1 =36 and m_2 =28, the constraints can be formulated as follows:

```
1 for(int i=0;i<36;i++)
2 {  a[i]=0;  }
3 for(int i=0;i<28;i++)
4 {  b[i]=0;  }
```

C. Design the Function for Controlling Delay

The proper parameters to the delay sub-routine can be used to set the delay time for the simulator. In this study, three-layer *for*-loop is implemented to enlarge the delay time. When the parameter (define x) is large, the delay time is long.

```
1  void set_delay(int x)
2  {
3     for(int i=0; i<x; i++)
4     {
5        for(int j=0; j<x; j++)
6        {
7           for(int k=0; k<x; k++)
8           {
9
10          }
11       }
12    }
13 }
```

D. Detect the Collision at Dangerous Collision Region

When there is car collision because the line tracks of two cars are too close (or the same), the car with low-priority should be terminated. Of course, the car with high-priority keeps going through the dangerous regions and then the car with low-priority restarts. The example with *d*=3 is used to explain. Actually, the random value of *d* can be implemented according the user-defined value. In the condition of *d*=3 for dangerous region 1 (r_1), the following constraints are suggested as follows:

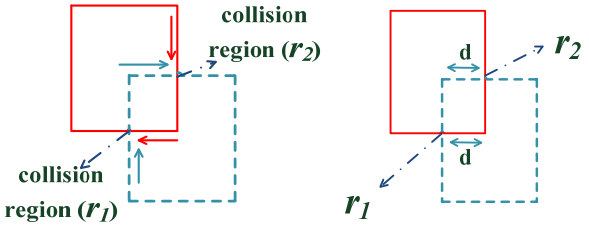
if (((a[10]==1 || a[11]==1 || a[12]==1) && b[1]==2))

In the condition of *d*=3 for dangerous region 2(r_2), the following constraints are suggested as follows:

```
1  if ( ((a[19]==1 || a[20]==1 ||a[21]==1) && b[20]==2 ) )
2  {
3        cnt1++;
4        set_delay(900);
5  }
```

According to above discussions, the car B is terminated and cnt2=0. To observe easily, the delay is set by using the set_delay(900). Lines 1 to 4 are for *if* -statement. Summary, the programming constraints are implemented as follows:

```
1 if (
2     (( a[10]==1 || a[11]==1 || a[12]==1 ) && b[1] ==2)   ||
3     (( a[19]==1 || a[20]==1 || a[21]==1) && b[20]==2)
4  )
5 {
6     cnt1++;
7     set_delay(900);
8 }
```

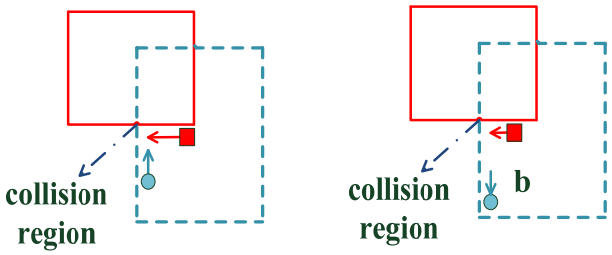


(a) (b)

Fig. 4 Illustration of collision regions and detected grids

E. Move Backward for the Future Extensions

To handle the difficult problems, the backward operation is implemented. In fact, the car sometimes needs more grids (i.e. more physical space). Hence, the car with low-priority goes backward to increase buffer regions, see Fig. 5.



(a) general case (b)go backward

Fig. 5 Illustration of movement backward for car B

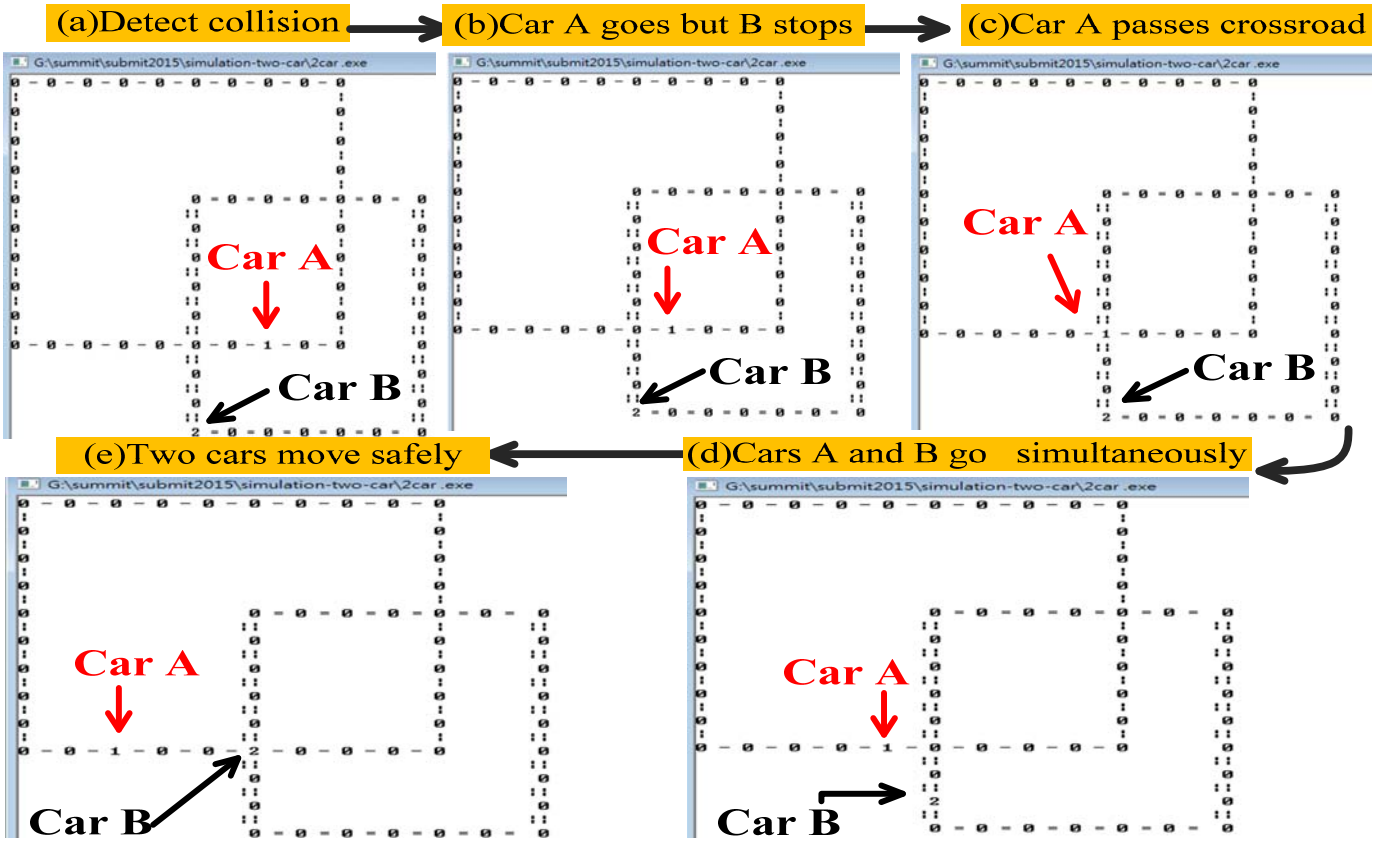


Fig. 6 Illustration of collision detection and the proposed method solves the collision problem

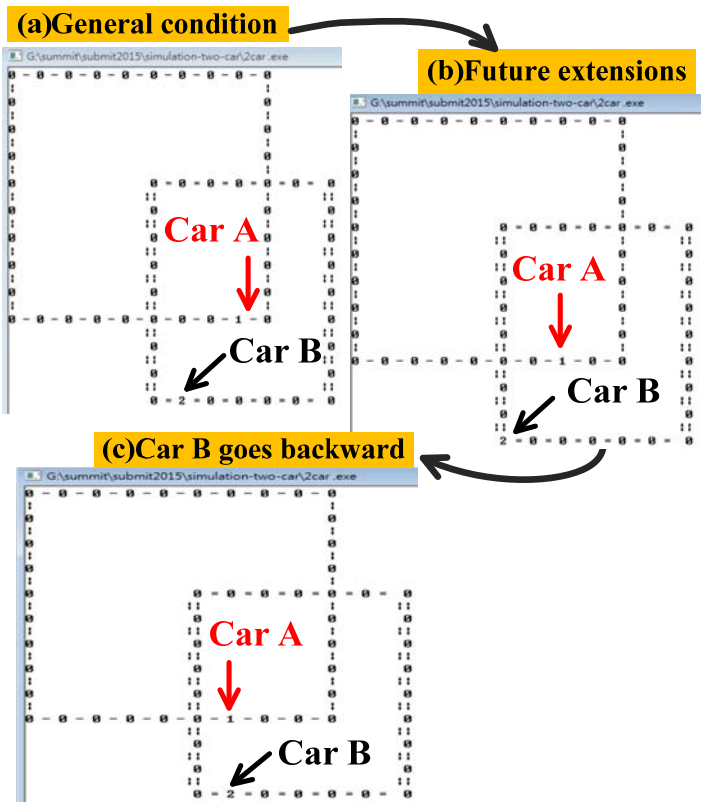


Fig. 7 Illustration of going backward for future extensions

F. *The Results of the Proposed Procedure*

Figs. 6 and 7 show the proposed method to handle the collision for two cars. The general case is shown in Fig. 7. When two cars will happen the car collision, then the car with low-priority stops before the potential collision point with d grids which is discussed in Sub-section II.C, Figs. 4(a) and 4(b). Fig. 6 shows that the procedure to handle two cars without car collision before the potential collision point with 3 grids (d =3). The additional consideration for the future extensions, students should make car B go backward with b grids which is also discussed in Sub-section II.C and shown in Fig. 7. It shows that when the collision is detected (Figs. 7(a) and (b)), and car B goes backward with 1 grid (b =1) to take more space to the future extension in Fig. 7(c).

IV. Experimental Results

In this work, the language is implemented by using the C language and the experiments are performed on an Intel i7-2600 3.40GHz machine with 8GB memory. The students are of the electronic engineering of University of Science and Technology. The series of training exercises in Fig. 3 are used to train the concept and implementation for the collaborative operations. Most of them have the experiences with implementations of the wheeled robots or the skills to design one microcontroller. Based on their background from wheeled robots, they are able to think the collaborative operations.

The questionnaires surveyed are from 8 students to collect their opinions about this training course. The teaching style is the lecture describes the concept and implementation for collaborative operation by using the slides containing the programming codes and the assigned simulators. Then the student can go to setup the delay to simulate. TABLE 5 lists the questionnaire from the 8 students. The results of the questionnaire are quite positive. It means that the course assignment, the teaching materials, slides and programming exercise are interesting and effective to learn the operations. Fig. 8 shows the distribution of points 1,2,3,4 and 5 from the questionnaire. Item 3 "I learn to how to setup the delay to guide the car to move along the line tracks by using programming." obtained the highest points from the questionnaire and it means that the students learn how to observe the simulator by adjusting the delay. Fig. 8 shows that the percentage of points 4 and 5 plays the major role for this questionnaire. Most students give the 5 and 4 points for the Items 1 to 5. It means that most students give the training course the positive suggestions and opinions.

V. CONCLUSIONS

This study explores the interesting learning framework to increase the learning results for the students of University of Science and Technology to learn multi-robot collaborative operations. The interesting problem is defined and the lecture designs the series of exercises to form the necessary teaching materials for learning the multi-robot collaborative operation without collision. The easy-to-use one-dimension array is used to solve this difficult problem, instead of the complex data structures or algorithms. Both of the traditional stop-then-start method and the extended go-backward method are designed to handle the car collision or future extensions. To make students observe the simulator easily, the delay for each step are adjusted. From the questionnaire, most students agreed that they learn the skills to setup the proper delay to observe the simulator.

References

[1] J. McLurkin, J. Rykowski, M. John, Q. Kaseman, A. Lynch, "Using Multi-Robot Systems for Engineering Education: Teaching and Outreach with Large Numbers of an Advanced, Low-Cost Robot, " *IEEE Transactions on Education*, 2013.

[2] Hsin-Hsiung Huang, Chyi-Shyong Lee, Juing-Huei Su, Chia-Lung Yang and Tsai-Ming Hsieh, "Industry-Orientation Training Course by Line Following Maze Robot," *International Journal of Education and Information Technologies*, pp. 105-112, 2011.

[3] C.R. Osorio, J.A. Romero, C.M. Peña, and I.L. Juárez, "Intelligent Line Follower Mini-Robot System," *International Journal of Computers, Communications & Control*, pp73-83, 2006.

[4] Cristina Turcu, Cornel Turcu, Vasile Gaitan, "Integrating Robots into the Internet of Things," *International Journal of Circuits, Systems and Signal Processing*, pp. 430-437, 2015.

[5] Marko Odak, Ţeljko Marušić, Nikolaj Lazić, Mladen Bruck, "Selecting the Best Robotic Kits in Term of Increasing Creativity," *International Journal of Education and Information Technologies*, pp.179-186, 2014.

[6] J. McLurkin, "Experiment Design for Large Multi-Robot Systems," in *Proc. Robot., Sci. Syst., Workshop Good Exp. Method. Robot*, Seattle, WA, Jun. 2009.

[7] R. D'Andrea, "Guest Editorial: A Revolution in the Warehouse: A Retrospective on Kiva Systems and the Grand Challenges Ahead," *IEEE Transactions on Automation Science and Engineering*, 2012.

[8] N. Smolic-Rocak, S. Bogdan, Z. Kovacic, T.Petrovic, "Time Windows Based Dynamic Routing in Multi-AGV Systems," *IEEE Transactions on Automation Science and Engineering*, 2010.

[9] T. Nishi, M. Ando, M. Konishi, "Distributed Route Planning for Multiple Mobile Robots using an Augmented Lagrangian Decomposition and Coordination Technique," *IEEE Transactions on Robotics*, pp. 1191-1200, 2012.

[10] C.T. Cai, C.S. Yang, Q.D. Zhu, and Y.H. Liang, "Collision Avoidance in Multi-Robot Systems," in *Proc. of International Conference on Mechatronics and Automation*, pp. 2795- 2800, 2007.

[11] Yan Yongjie and Zhang Yan, "Collision Avoidance Planning in Multi-Robot based on Improved Artificial Potential Field and Rules," in *Proc. of IEEE International Conference on Robotics and Biomimetics*, pp. 1026-1031, 2008.

[12] D. Sun, A. Kleiner and B. Nebel, "Behavior-based Multi-Robot Collision Avoidance," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*,pp. 1668-1673, 2014.

[13] Daniel E. Soltero, S.L. Smith and D. Rus, "Collision Avoidance for Persistent Monitoring in Multi-Robot Systems with Intersecting Trajectories," in *Proc. of 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.3645-3652, 2011.

[14] F. Belkhouche, and T. Jin, "An Approach for Collaborative Path Planning in Multi-robot Systems," in *Proc. of IEEE International Conference on Systems, Man and Cybernetics*, pp. 2356 - 2361, 2009.

[15] Hsin-Hsiung Huang, Juing-Huei Su, and Chyi-Shyong Lee, "A Contest-Oriented Project for Learning Intelligent Mobile Robots," *IEEE Transactions on Education*, 2013.

[16] http://www.math.ncu.edu.tw/~jovice/c++/boards/devcpp.htm

[17] http://www.bloodshed.net/devcpp.html

[18] http://bytes.com/topic/c/answers/545639-what-does-system-cls-do

Hsin-Hsiung Huang becomes an IEEE member since 2010 and was born in Taiwan in Oct. 1974. He received the M.S. and Ph.D degrees in the Department of Information Computer Engineering and Institute of Electronic Engineering from Chung Yuan Christian University, Taoyuan, Taiwan, in 2000 and 2008, respectively. He is working toward the algorithm–related fields, such the applications of line-following maze robot for the shortest path problems and EDA algorithms for the VLSI, the floorplanner and performance-driven routing with the obstacles.

He is currently an Associate Professor in Department of Electronic Engineering, Lunghwa University of Science and Technology in Taoyuan in Taiwan. From 2000 to 2002, He is a hardware engineering to design the Ethernet product at Accton Corporation, Hsin-Chu, Taiwan. From 2002 to 200, He serves as a research and development engineering and focus on the chip design for the 10/100/1000 Mbps Ethernet MAC at TM-Technology Corporation, Hsin-Chu, Taiwan.

Dr. Huang is also an oversea member of Institute of Electronics, Information and Communication Engineers (IEICE for short) since 2009. Some publications are listed in IEEE conferences, including ISCAS, MWSCAS and ISIC, respectively.

Juing-Huei Su (S'87–M'93–SM'08) was born in Tainan, Taiwan, in 1965. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1987, 1989, 1993, respectively. From 1993 to 1995, he served as a military officer in the army. In 1995, he became a Senior Engineer with the Taian Electric Co., Ltd.

Since 2007, he has been a Professor with the Department of Electronic Engineering, Lunghwa University of Science and Technology, Taoyuan, Taiwan. He is now interested in developing his own learning platforms for automatic control and robotic education.

His research interests also include robust control theory, power electronic systems, and embedded control system implementations.

Chyi-Shyong Lee received the B.S. from National Taipei University of Technology, Taiwan, in 1979, and the master degree from the National Tsing Hua University, Taiwan, in 1985, both in electrical engineering. From 1985 to 1988, he served as a Lecturer with the Hwa Hsia Institute of Technology, Taiwan. He was a Lecturer with the Department of Electronic Engineering, Lunghwa University of Science and Technology, Taiwan, from 1989 to 2007, and is now an Associate Professor. Currently, his research interests include digital control of power electronic systems and the applications of microcontrollers and embedded systems.

ACKNOWLEDGEMENTS

The authors would like to thank the financial supports of the grant MOST 103-2511-S-262 -002 from Ministry of Science and Technology.

Table 5. Questionnaire from the students after the training course.

Item	Question for students	Points (1~5)
1	**I am satisfied with the arrangement of training materials and programming simulators.**	**3.9**
2	**I learn how to implement the collaborative operation for two cars by using one-dimension array.**	**3.8**
3	**I learn to how to setup the delay to guide the car to move along the line tracks by using programming.**	**4.3**
4	**I would like to encourage the other students to learn the training materials and simulator.**	**4.1**
5	**I learn to use one-dimension array to display the line tracks of the car on the screen.**	**4.1**

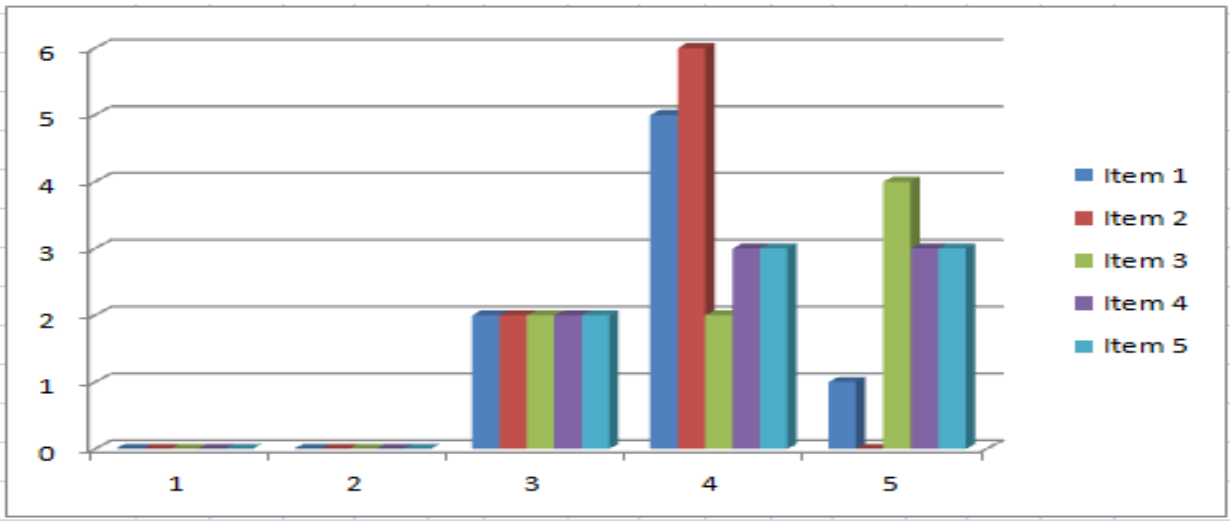


Fig. 8 Illustration of questionnaires results of five items from 5 point to 1 point