Xiong, Guangyu; Niu, Lulu; Tian, Yanfu ; Zhou, Jiehan; Mohajeri, Babak; Nyberg, Timo

# Path Planning System for Smart Cars Used in Education

*Published in:*
Proceedings of the 15th IEEE Conference on Industrial Electronics and Applications, ICIEA 2020

*DOI:*
10.1109/ICIEA48937.2020.9248372

Published: 09/11/2020

*Document Version*
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

*Please cite the original version:*
Xiong, G., Niu, L., Tian, Y., Zhou, J., Mohajeri, B., & Nyberg, T. (2020). Path Planning System for Smart Cars Used in Education. In *Proceedings of the 15th IEEE Conference on Industrial Electronics and Applications, ICIEA 2020* (pp. 229-234). Article 9248372 IEEE. https://doi.org/10.1109/ICIEA48937.2020.9248372

# Path Planning System for Smart Cars Used in Education*

Guangyu Xiong, Lulu Niu, Yanfu Tian, Jiehan Zhou, Babak Mohajeri, Timo R. Nyberg

***Abstract*—In this paper, we developed a path planning system for smart cars for teaching electronic engineering or computer science, which consists of the interactive platform for smart cars development and path planning. Designed by Visual C++, the interactive platform can call Matlab engine, allows users to choose path optimization algorithms such as genetic or *A-star(A*)* algorithm for different tasks and control smart cars through serial ports. The simulation and practice demonstrate that our interactive platform can help learners to plan paths and control intelligent vehicles without specially designing a user interface.**

***Index Terms*—Path planning; STEAM; Smart cars; Genetic Algorithm; *A-star(A*)* algorithm**

## I. Introduction

With the deep integration and rapid development of artificial intelligence and education science, reforming the form of education organization, constructing innovative courses of science and technology, cultivating talents with comprehensive science and technology literacy and deep innovative ability, have become a strategy of STEAM education [1]. To promote STEAM education, we usually need to develop related education equipment, design a multi-disciplinary curriculum system. This paper aims at developing a path planning system for smart car used in STEAM education, to develop students' interests and skills in electronics and intelligent technologies.

Smart cars have played a significant role in cosmic exploration, disaster reliefs. Smart cars can also be used for simulations and experiments such as path planning, environmental perception, and automatic driving [2]. The precision control and intelligent level of cars has a significant impact on social systems [3].

*This work was supported in part by the National Key Research and Development Program of China (No. 2019YFB1704102); The National Natural Science Foundation of China under Grants U1909204, 61773381, U1811463, 61773382 & 61872365; Chinese Guangdong's S&T project (2017B090912001); (Corresponding author: Timo R. Nyberg)

Guangyu Xiong is with the Silk Road Business School, Sanya University(USY), Hainan, China. She is also with The Guangdong Engineering Research Center of 3D Printing and Intelligent Manufacturing, The Cloud Computing Center, Chinese Academy of Sciences, Dongguan, China. (e-mail: xiongguangyu@hotmail.com)

Lulu Niu is with the School of Articial Intelligence, University of Chinese Academy of Sciences, Beijing, China. She is also with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. (e-mail: niululu2018@ia.ac.cn)

Yanfu Tian is with the Silk Road Business School, Sanya University (USY), Hainan, China. (e-mail: tianyanfu@126.com)

Jiehan Zhou is with University of Oulu, Oulu, Finland. (e-mail: jiehan.zhou@gmail.com)

Babak Mohajeri and Timo R. Nyberg are with the Department of Industrial Engineering and Management, Aalto University, Espoo, Finland. (e-mail: babak.mohajeri@aalto.fi, timo.nyberg@professori.fi)

Path planning is used to solve the optimal trajectory problem, that is, according to some optimization criteria (such as minimum cost, shortest path or time, etc.) to find the starting point of the workspace, the target route, and the best path to avoid obstacles [4]. Common path planning algorithms can be broadly classified into: sampling-based methods, such as *Vernon, PRM, and RRT, etc.*; node-based methods, such as *A-star(A*), D*, etc.*; mathematical models-based methods [5], [6], such as mixed integer nonlinear programming (*MILP*); biological heuristic algorithm-based, such as genetic algorithm(GA), particle swarm algorithm [7]. Liu *et al.* [8] proposed a super-peer-based coordinated service provision to improve the scalability and robustness of intelligent systems.

The paper aims to develop a path planing system. It has the following contributions:

- Develop a path planning system, which can reduce the learning burden for students so that they can focus more on optimization algorithm learning and practice.
- Develop an interactive user interface for personalized and automatic operation.

The rest of this paper is organized as follows. Section II proposes a smart car system for education. Section III describes the path planning algorithm and the two optimization methods including the GA and A* algorithms. Section IV designs the interactive user interface. Section V implements the smart car system. Section VI analyzes the experimental results. Section VII concludes this paper.

## II. The smart car system design

The system consists of three layers as shown in Fig.1. The interface layer is programmed based on Microsoft Foundation Classes (MFC), so that the users only need to implement the optimization algorithm such as GA and A* in the Matlab terminal by calling the Matlab engine [9]. It has 3 functions as follows.

- *Func1*: set a path as they like.
- *Func2*: set multiple points randomly.
- *Func3*: generate a grid map and set the obstacle and target points.

The paths are planned for different optimization tasks respectively, then they are transfered through the bluetooth serial port to the communication layer.

The communication layer estimates whether it receives the path data from the interface layer. If no data is received, it will not work.

The control layer receives the data and then controls the car to run along the planned path.

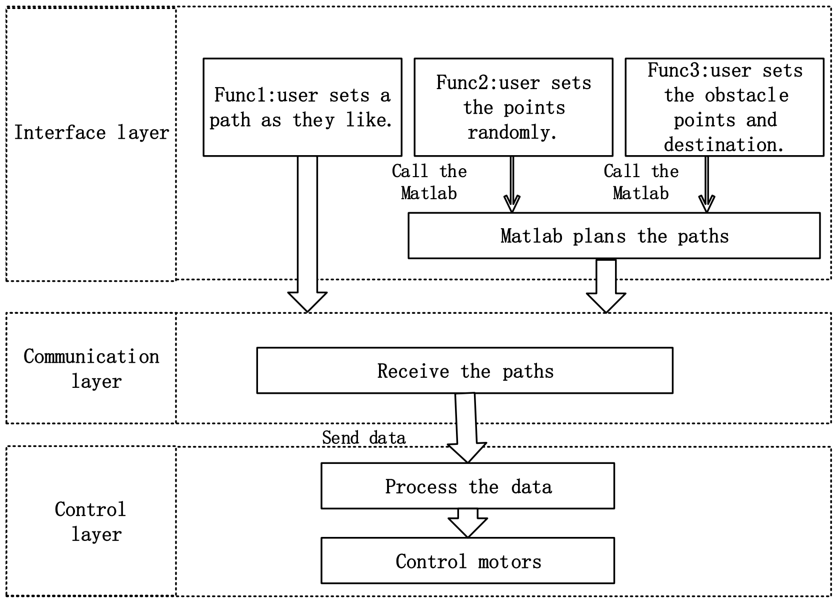


Fig. 1: The modules and interactive process of the system

Taking into account of learning optimization guidelines for middle school students, the GA is used in *Func2* to find the shortest path in our project. The real scene is simulated using the A* algorithm for the path planning in *Func3*. *Func1* allows users to set paths at will. Users can also add their own path planning algorithm as required.

The feasibility of the path planning algorithm using hybrid programming is verified through simulation and practical application. The users only need to implement the optimization algorithm in the Matlab terminal and call the Matlab engine on the interactive system designed in our project [9]. Then the smart car is controlled by the loading optimization algorithm to run, without the need of designing the interactive interface and writing the smart car program and hardware settings. Thus learners can focus more on the optimization algorithm learning and practicing.

## III. Preliminaries

### A. Genetic Algorithm

The GA accomplishes the optimization search by simulating the biological evolution process. Its main feature is the population search strategy and the information exchange between individuals in the population [10]. The search is not based on gradient information. It is suitable for dealing with complex and non-linear problems. It is widely used in combinatorial optimization, machine learning, adaptive control, planning and design, artificial life and other fields [11].

The roulette selection [12] method is used in our system, the individual selection probability is proportional to its fitness value. Assume the population size is $n$, where the fitness value of individual $i$ is $f_i$. The probability of $i$ being selected is as(1):

$$p_i = \frac{f_i}{\sum_{j=1}^{n} f_j} \tag{1}$$

Obviously, the probability $p_i$ reflects the proportion of individual $i$'s fitness value in the sum of individual fitness values. The greater the individual's fitness, the higher the probability of being selected, and vice versa.

### B. A-Star(A*) Algorithm

The A-Star(A*) algorithm is the most effective direct search method for solving the shortest path in a static road network [13]. The closer the heuristic function $h(n)$ distance between the estimated value and the actual value $g(n)$ in the algorithm, the faster the final search speed. The formula is as (2):

$$f(n) = g(n) + h(n) \tag{2}$$

Where $f(n)$ is the estimated cost from the initial state via state $n$ to the target state, $g(n)$ is the actual cost in the state space from the initial state to state $n$, and $h(n)$ is the estimated cost of the best path from state $n$ to the target state.

The key to finding the shortest path is the selection of the evaluation function $f(n)$ and the heuristic function $h(n)$. We express the true distance dissipation from state $n$ to the target state $d(n)$. The selection of $h(n)$ can be divided into the following three cases. In the case of $h(n) < d(n)$, the search process is heavy, and many candidate regions are invalid. But optimal solutions can be obtained. When $h(n) = d(n)$, the path search process will be performed strictly along the shortest path, and the search efficiency at this time is the highest. If $h(n) > d(n)$, the search process is quick and with low complexity, but it may omit the optimal solution.

## IV. The software realization of the system

### A. The Application of GA

The GA solution process parameters are set as follows:

*1) Population initialization:* Each person is individually coded for the set target point, and real numbers from 1 to $n$ are randomly arranged in a real number coding manner. Initialization parameters include the number of population $M$, the number of chromosomal genes $T_f$ (that is, the number of target points), and the number of iterations $C$, crossover probability $p_c$, mutation probability $p_m$ and so on.

*2) Fitness function setting:* Since the distance $D(i,j)$ between any two target points is known, the total distance can be calculated for each chromosome (the random arrangement of $n$ target points), so the reciprocal of a total random distance can be calculated. Fitness function $F(x)$ can be defined as (3):

$$F(x) = \frac{1}{\sum_{i=1}^{n-1} d(C_i, C_{i-1}) + d(C_1 + C_n)} \tag{3}$$

That is, the shorter the distance, the better the fitness function.

*3) Select operation:* In our paper, the roulette method is used to generate the initial population. According to the calculated individual fitness degree, the entire population is ranked from the smallest to the largest, and the selection percentage $p$ is set. In order to maintain the diversity of the population, the choice $p$ should be selected. Good individuals of $p/2$ should be chose, and then we choose $p/2$ individuals into the mating pool for crossover, mutation and other genetic operations.

*4) Crossover operation:* Two individuals are randomly selected and several gene segments are exchanged at the corresponding positions, and at the same time, each individual is still kept in a random arrangement from 1 to $n$ to prevent local convergence.

*5) Mutation operation:* Individuals are selected at random and the gene of them are randomly selected for exchange to achieve mutation operations.

After the parameters of chromosome code and crossover mutation operator are designed, the specific implementation steps of the GA are considered [15]. The pseudocode used in our project is shown in Algorithm 1.

### B. Implementation Effect of GA

We set those parameters in the Matlab program: the population size is 10, the maximum evolutionary algebra is 1000, the crossover probability is 0.4, and the mutation probability is 0.2.

**Algorithm 1** GA

```
Input: The record of coordinate position when the mouse clicks interactive interface. Initialize $p_c, p_m, T_f, G, etc.$ parameters and random generated population $P = s_0, s_1 ..., s_n$.
output: The optimal path between each point.
1: while "Any chromosome's score $> T_f$, or reproductive generations $> G$" do
2:     Use $F_i = 1/\sum_{j=1}^{n-1} d(C_j, C_(j-1)) + d(C_1 + C_n)$ to calculate individuals' fitness in the initial parameters.
3:     repeat
4:         Two individuals were selected from population $P$ according to fitness and proportional selection algorithm;
5:         The crossover and mutation operations were performed for two individuals by $p_c$ and $p_m$ respectively;
6:         Add two new individuals to $newP$
7:         Replace $P$ with the $newP$ and update;
8:     until "The number of created children$< M$"
9: end while
10: return $PathResult$
```

### C. Application of A* Algorithm

We applies the A* algorithm to the *Func3:* grid map maze. Before the application, we first need determine the position of the grid, that is, to identify the grid. Generally, the following methods are used [16].The same position can be expressed by above methods, respectively:

*Rectangular coordinate method:* It uses the Cartesian coordinate system, which sets the coordinate origin as the upper left and sets the $X$ axis as the right horizontally, and the $Y$ axis as vertically downwards, counts the number of $x$ and $y$ axes and uses the $(x, y)$ coordinates to set the grid. The position is marked as shown in (4), $p$ means the position of the grid:

$$p = x + 10 * y \tag{4}$$

*Serial number method:* Directly count the number of the grid marks, from the upper left corner to 1, in order from top to bottom, from left to right, the position $p$ can also use (5) to convert into the Cartesian coordinate$(x, y)$:

$$\begin{cases} x = mod(p, 10) \\ y = int(p, 10) \end{cases} \tag{5}$$

In (5), $mod$ represents the remainder of $p/10$, and $int$ represents the integer of $p/10$ [17].

This paper uses the serial number method and the process of algorithm's application is as follows:Take a $10 * 10$ grid diagram as an example. Assume that there is a directed right graph. Each point is in a state, and the state can only reach eight states adjacent to it. The difference between the positions is represented by the serial number method, so the relative position of the center point and the surrounding neighborhood points are $[-11, -10, -9, -1, 0, 1, 9, 10, 11]$ and the real distance $g(n)$ of them are $[1.4, 1, 1.4, 1, 0, 1, 1.4, 1, 1.4]$.

The real distance function $g(n)$ has been obtained. Then, the setting of the heuristic function $h(n)$ is performed. In general,

$h(n)$ selects the Manhattan distance. And $h(n)$ can also be set to 0, at this time, the A* algorithm will degenerate into *Dijkstra* algorithm. Due to the characteristics of the grid map, the distance $d$ from one state node to another state node always has $d(n)1$, and the heuristic function can be set to $h(n) = 1$ to make sure $0 \leqslant h(n) \leqslant g(n)$. By comparing the total time spent using different heuristic algorithms, it can be found that although the final planned path is consistent, the running time of the algorithm when $h(n) = 1$ is far less than the running time of $h(n) = 0$, as well as, the closer the value of heuristic function is to the target real distance, the shorter total running time of the algorithm is.

After the obstacle point is set, the distance function $g(n)$ from all points to this obstacle point need be set to infinity. The A* algorithm design procedure in our project is shown in Algorithm 2.

**Algorithm 2** A* searching

```
Input: Enter target point as the final node and obstacle
    position as unreachable nodes in interactive interface.
output: Optimal path to reach the target point.
1:  Put the start node on the OPEN list;
2:  while "the OPEN list is not empty" do
3:      Get the node off the open list with the lowest $f(n)$
        and call it $nodeCurrent$;
4:      if "$nodeCurrent$ is the same state as $nodeGoal$" then
5:          Break while loop;
6:      end if
7:      Generate each state $nodeSuccessor$ that can come
        after $nodeCurrent$;
8:      for each $nodeSuccessor$ in $nodeCurrent$ do
9:          Set the cost of $nodeSuccessor$ to be the cost of
            $nodeCurren$ plus the cost to get to $nodeSuccessor$ from
            $nodeCurrent$;
10:         Find $nodeSuccessor$ on the OPEN list;
11:         if "$nodeSuccessor$ is on the OPEN list or CLOSE
            list but the existing one is as good or better" then
12:             Continue;
13:         end if
14:         Remove occurences of $nodeSuccessor$ from
            OPEN and CLOSE list;
15:         Set the parent of $nodeSuccessor$ to $nodeCurrent$
            and heuristic function $h(n)$ to be the estimated distance
            to $nodeGoal$;
16:         Add $nodeSuccessor$ to the OPEN list;
17:     end for
18:     Add $nodeCurrent$ to the CLOSE list;
19: end while
20: return $PathResult$
```

## D. Implementation Effect of the A* Algorithm

To draw a grid map in Matlab, use the *setgca* command to change the axes so that the ordinate is in the vertical direction; use the *colormap* command to draw a grid map. Draw a $10*10$ raster map by using command $colormap([0\ 0\ 0; 1\ 1\ 1])$, Draw a black and white map by using command $pcolor(0.5 : size(D, 2) + 0.5, 0.5 : size(D, 1) + 0.5, b)$. Fig 2 shows four situations where obstacles are in different locations with a map size of $10*10$ and the path planning result of the A* algorithm by using $h(n) = 1$.

## E. Effect of the Algorithm Combined with Interactive Interface

The algorithm and related interactive interface were implemented on a PC with 4GB of memory, Intel core i3 CPU, and win10 system. The development environment was Visual Studio 2013 and Matlab2015b(32bit). The system of interactive interface was written by using MFC. After the mouse clicked on the target point, the Matlab engine was called for path planning. The system of interactive interface is implemented by MFC in Visual Stdio 2013. It calls *ActiveX MSCOMM* widget to send and receive serial data, and then communicate with smart car. In the system, we also design relative program to call the engine of Matlab for path planning, and then sends the information of planning result through the serial port to the smart car. The realization effect of each function in the system is shown in Fig 3.

The overall design interface is shown in Fig 3(a). After the user uses the *Func1* to set a path in the interactive interface at will in Fig.3(b), click the *Send* button, and send the path information from com6. The smart car serial module receives this data, and then the single chip microcomputer(SCM) controls the motor to run following the information. Fig.3(c) shows the realization of *Func2*. In the blank map, 10 artificial points are randomly set. After the GA is implemented in Matlab, the Matlab engine is called on the interactive interface to find the shortest path through the 10 points, and then the information of planed optimal path is returned to the upper computer. And when the *Send* button is clicked, the route information will be sent to serial data receiver module of the smart car. Fig 3(d) shows the implementation process of *Func3*. In the grid map, we can set obstacle points and destination node at will

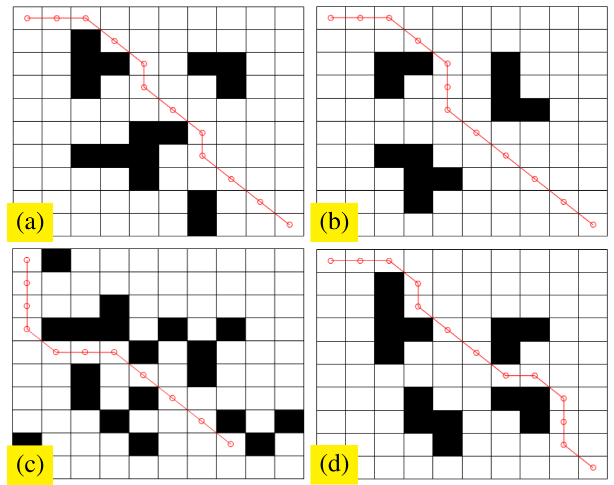


Fig. 2: The planning results by using A* algorithm in different obstacle positions, such as (a),(b),(c),(d).

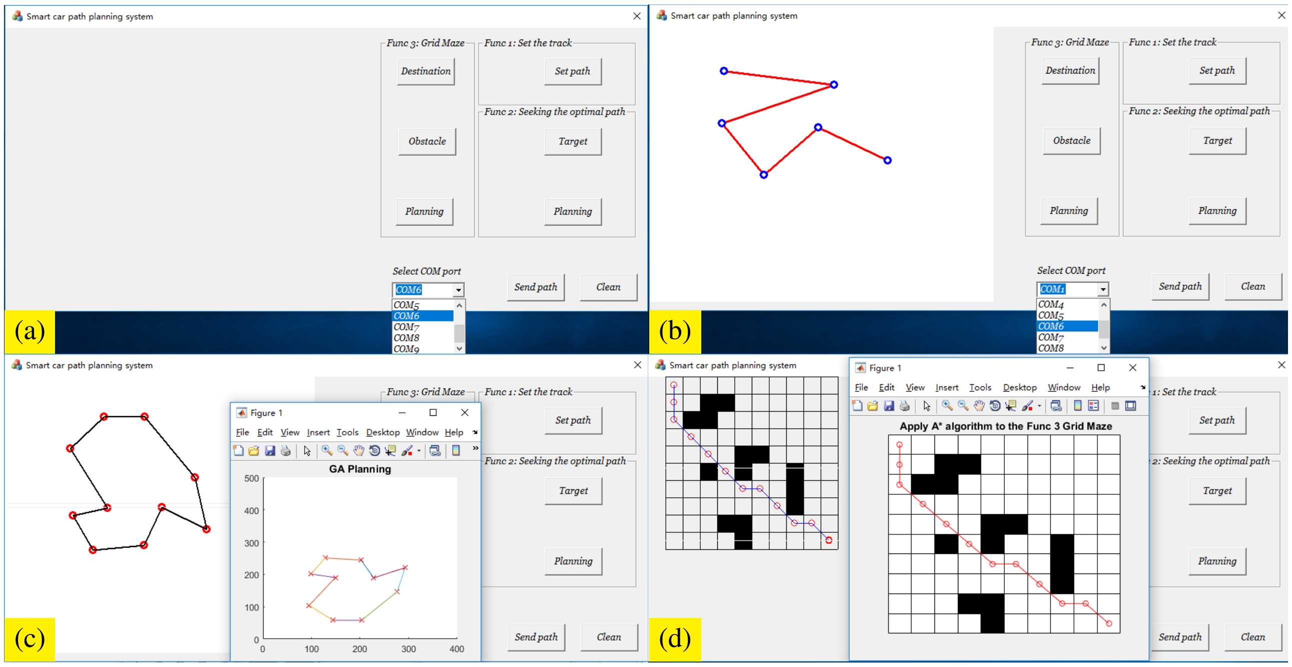


Fig. 3: (a)The overall designing interface programmed by using MFC. (b)The interface of *Func3*. Users are allowed to set the path as they like. (c)The realization of *Func2*. Ten artificial points are randomly set. The Matlab engine is called to implement GA, then the planned optimal path is transported form Matlab to the VC interface. (d)The implementation of *Func3*. We can set obstacle points and destination at will in grid map. And it will find the optimal path automatically.

to simulate a maze. Then these information we set of the path is sent to the Matlab engine, and the $.m$ files written in Matlab starts execution. Using path information about obstacle and destination and implementing the A* algorithm, we can find the optimal path through the maze and return the path information to the interactive interface. The solid line in the red dot is the planned path. From the results, it can be seen the obtained path is optimal.

## V. Smart Car System Implementation

### A. The Design of Smart Car Hardware Structure

The smart car's hardware is designed by *RP5* tracked vehicle and it uses *L298n* module to drive motor. The control core board is *STM32f103c8t6*. It communicates with the upper computer through a bluetooth serial port as Fig 4(a) shows. It uses the photoelectric encoder as the feedback for the angle of rotation and forward distance, and uses the *J-LINK* emulator's *JTAG interface* mode to download the program and debug.

### B. Lower Computer Smart Car Control Program Design

*1) Control smart car design:* The Keil $\mu$version 5 development environment is used to compile, download and debug the SCM program on PC that runs in 4GB memory, Intel core i3 CPU, and win10 system.

*2) Use PID controller to eliminate errors:* Accumulation of errors during straight-line progress results in deviations from straight-line trajectories. Therefore, the PID controller is used in the SCM to eliminate the error. The two-wheel encoder difference $e$ is input to the controller, and the output PWM wave is corrected to eliminate the error $e$ [18].

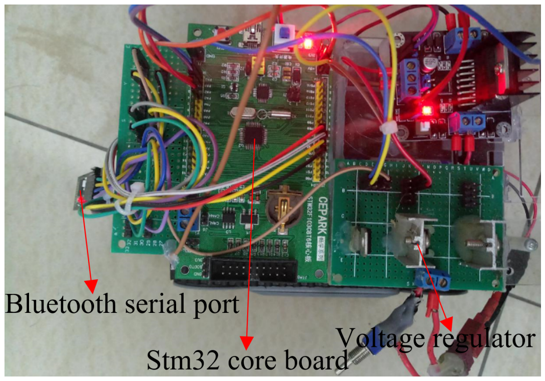
### C. Smart Car Design Results

Fig 4(a) is the practicality picture of the smart car. The hardware components have been indicated in the figure. After receiving the serial port data form the system, the STM32 core board begins to control the smart car to follow the planned trajectory. When driving straight, the same PWM wave will be given to both wheels. When the vehicle needs to turn, it will use the method of turning in place. One motor will rotate in the forward direction and the other will rotate in the reverse direction. Fig 4(b) is the USB to serial port module of the upper computer and the bluetooth sending module, which is responsible for sending path information planned by the upper computer to the smart car.

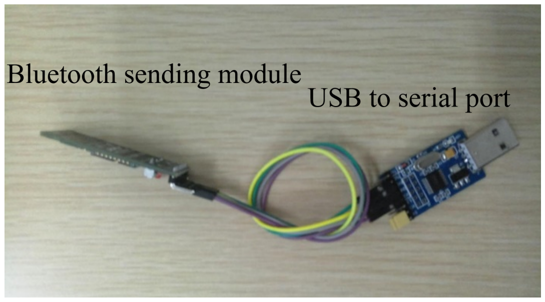
## VI. Result Analysis and Discussion

The scale between the interface layer path planning and reality is set to $1:45$. That is, $1cm$ in the interactive interface equals $0.45$ meters in reality. Obstacles are set in reality and obstacle points are entered on the upper computer. Then path planning is performed after the target point is set. The designed system not only shows the path in the upper computer, but also calls Matlab to draw the action trajectory. The smart car runs following the planned path of the upper computer. Obviously, the paths showed in Fig 3(d) is optimal.

In general, our system design is ideal. After completing the route following planned route, the optimized path information



(a) The hardware of smart car



(b) USB to serial port module and bluetooth sending module

Fig. 4: The profile of overall system

can be successfully loaded and the obstacle avoidance in reality can be achieved.

## VII. Conclusion

This paper uses the interactive system designed by MFC to facilitate the education and teaching process for learners. The Matlab is used to run the GA and A* optimization algorithm on the interface layer to perform path planning for different optimization tasks. Through simulation and practical application, we have verified the feasibility of using a hybrid programming method with VC and Matlab to control smart cars using different path optimization algorithms. Learners who're in learning and practicing smart car path planning, only need to learn and design a path planning algorithm by Matlab. They needn't design the interactive interfaces and write programs of smart car controlling. In addition, it reduce the difficulty of programming for middle school and college students greatly. Generally speaking, the control effect is relatively ideal. The smart cars can follow the planned path, automatically search the optimal path and achieve obstacle avoidance in reality. The smart car developed in this paper can be used for electronics and computer science technology teaching or related interdisciplinary education innovation, and it not only cultivates the electronic design ability of the students, but also trains them to learn optimization algorithms. Although substantial progress has been made, there is plenty of room for improving. For example the design of the interactive interface need continue to be improved to achieve convenience, aesthetics, and meet the different needs of educational researchers.

## References

[1] J. Wang, Y. H. Wu, The Reflection and Innovation Path of STEAM Education Application in the Era of Internet+[J]. Journal of Distance Education, 2016.

[2] A. Mammela, J. Riekki, A. Kotelba and A. Anttonen, Multidisciplinary and Historical Perspectives for Developing Intelligent and Resource-Efficient Systems. IEEE Access, vol. 6, pp. 17464-17499, 2018.

[3] F. Y. Wang, Computational Dissemination: Toward Precision and Smart Impacts for Computational Social Systems. IEEE Transactions on Computational Social Systems, vol. 4, no. 4, pp. 193-195, Dec. 2017.

[4] B. Dai, X. M. Xiao, Z. X. Cai, Current Status and Future Development of MobiIe Robot Path PIanning TechnoIogy. ControI Engineering of China, 2005, 12(3):198-202.

[5] S. P. Chin, J. Cohen, A. Albin, et al., A Mathematical Analysis of Network Controllability Through Driver Nodes, IEEE Transactions on Computational Social Systems, vol. 4, no. 2, pp. 40-51, June 2017.

[6] F. Y. Wang, Computational Social Systems in a New Period: A Fast Transition Into the Third Axial Age, IEEE Transactions on Computational Social Systems, vol. 4, no. 3, pp. 52-53, Sept. 2017.

[7] Y. J. Jia, Research on Biologic Heuristic Algorithm and its Improvement. University of Science and Technology of China, 2010.

[8] M. Liu, T. Koskela, Z. Ou, J. Zhou, J. Riekki, M. Ylianttila, Super-peer-based Coordinated Service Provision. Journal of Network and Computer Applications, 34 (4), 1210-1224, 2011

[9] sina blog.(2014) C and MATLAB Hybrid Programming. [Online]. Available: http://blog.sina.com.cn/s/blog_7e2e98ad0101hef2.html

[10] K. Zhu, H. Wang, Y. Kong, et al., Optimization of the Replenishment Strategy for the Supplier Based on Genetic Algorithm. International Journal of Business & Management, 2011, 6(1).

[11] S. S. Ma, Research on Multi-Objective Genetic Algorithm And Non-dominated Sort. Shandong University of Science and Technology, 2010.

[12] H. Y. Liang, X. Zhang, An Improved Method for Roulette Wheel Selection of Genetic Glgorithm. Information Technology, 2009, 33(12):127-129.

[13] R. Huang, M. liu, The Principle of Realizing the Shortest Path based on A* Algorithm. The world of entrepreneurs, 2009(7):124-125.

[14] F. Yi, Topological Method of Hierarchical Routing based on Primary Routing Computation: CN, CN 1816000 A[P]. 2006.

[15] C. Jacob, Stochastic Search Methods. Intelligent Data Analysis, 2007:303-356.

[16] S. J. Ren, Bingrong Hong, Dehai Huang, A Robot Path Planning Algorithm based on Grid Expansion. Journal of Harbin Institute of Technology, 2001, 33(1):68-72.

[17] Y. L. Zhao, Data Structures and Algorithms. Tsinghua University press, 2008.

[18] J. K. Liu, Advanced PID Control and MATLAB Simulation. Publishing House of Electronics Industry, 2004.