The Eighteenth International Symposium on Artificial Life and Robotics 2013 (AROB 18th '13),
Daejeon Convention Center, Daejeon, Korea, January 30-February 1, 2013

Motion Planning of the Multi-robot Based Chess Game

Yung-Chin Lin[1,3], Yi-Lin Liao[1], Cheng-Yun Chung[1], Kuo-Lan Su[2]
[1]Graduate school Engineering Science and technology, National Yunlin University of Science & Technology
Douliou, Yunlin, Taiwan,
[2]Department of Electrical Engineering, National Yunlin University of Science & Technology, Taiwan
(Tel: +886-5-5342601, Fax: +886-5-5312065)
[3] Department of Electrical Engineering, WuFeng University of Science of Technology, Taiwan
E-mail: yclin@wfu.edu.tw, sukl@yuntech.edu.tw, g9910801@yuntech.edu.tw

Abstract: The article uses mobile robots to present movement scenario of each chess for Chinese chess game, and programs the shortest motion path using enhance A* searching algorithm. We play the Chinese chess game using the mouse according to the evaluation algorithm on the user interface of the supervised computer. The supervised computer controls mobile robots according to the programmed motion paths of the designed chesses moving on the platform via wireless RF interface. We develop the user interface using Borland C++ Builder language. The user interface displays the positions of the mobile robot based chesses, and plots the motion paths of the assigned chesses. The proposed algorithm can programs the shortest motion trajectories of the chesses that move to target positions from start positions, and avoids the collision points of the motion trajectories for the assigned chesses simultaneously, and can re-programs the new motion path to avoid collision. Finally, we implement the movement scenario on the chessboard platform using mobile robots. Mobile robots move on the platform according to the programmed motion paths, and uses IR sensor modules to avoid the obstacles (mobile robots) and detect the cross points of the platform, and calculate numbers of the cross point to decide the final locations.

Keywords: mobile robots, Chinese chess game, enhance A* searching algorithm, wireless RF module, Borland C++ Builder

I. INTRODUCTION

Chinese chess [1] game is one of the most popular games. A two-player game with a complexity level is similar to Western chess. The Chinese chess game has gradually attracted many researchers' attention, and many evolutionary algorithms to be proposed in the recent. In the paper, we use thirty-two mobile robots to present the movement scenario of each chess, and uses enhance A* searching algorithm to program the shortest paths for mobile robots moving to the target positions. In general, users move chess (robot) to the target position according to the game rules. In some conditions, user moves the chess to take the chess of the other side. There are two chesses (robots) moving in the platform simultaneously. The assigned two robots may collide on the programmed motion paths. We want to solve the collision condition to improve the shortest motion paths of the two robots using enhance A* searching algorithm.

There are many algorithms to be proposed in Chinese chess game. Darwen and Yao proposed the co-evolutionary algorithm to solve problems where an object measure to guide the search process is extremely difficult to device [2]. Wang used adaptive genetic algorithm (AGA) to solve the problems of computer Chinese chess [3]. Su developed smart mobile robot using voice module, and programmed the motion trajectories for multiple mobile robots based Chinese chess game [4]. Fu used the position evaluation function to play an important role for building an intelligent Chinese chess computer game [5]. Zhou constructed the computer Chinese chess game platform that created favorable conditions for the development of discrete event dynamic system theory and provided a good research platform for solving dynamic counter measure problems [6].

We use multiple mobile robots to present the movement scenario of the chesses. In some condition, mobile robots must program the shortest path and avoid the obstacles moving to the next positions. The obstacles may be static or dynamic obstacles. We used A* searching algorithm to program the shortest path on the Chinese chess game. A^* heuristic function is introduced to improve local searching ability and to estimate the forgotten value. Kong used adaptive harmony search algorithm to solve optimization problems [7]. Flavio presented a multi-robot exploration algorithm that aims at reducing the exploration time and to minimize the overall traverse distance of the robots by coordinating the movement of the robots performing the exploration [8].

II. SYSTEM ARCHITECTURE

The system architecture of the multiple robots based Chinese chess game system is shown in Fig. 1. The system contains a supervised computer, a chessboard platform, some wireless RF modules and thirty-two mobile robots. The players of the chess game are classified red side and black side. Each side includes sixteen chesses. Each player only moves chess on one step, and waits to play the next step until the other side moving chess on the user interface. These chesses stay at fixed positions to start the chess game. The movement rules of the chess game have two cases. Player moves chess to free position of the chessboard platform in the case one. The user interface programs the shortest path for the assigned robot. In the case two, player moves the chess to take the chess of the other side. The user interface programs two motion paths for the assigned chesses, and avoids collision points on the programmed motion paths. The two mobile robots move in the chessboard platform using the programmed motion paths.

Players move chesses using the mouse on the user interface of the supervised computer to be shown in upper side of Fig. 1. The supervised computer transmits the ID code and motion command to the assigned mobile robots, and receives the ID code and position from the mobile robots via wireless RF interface. Users can set game time until one player wins the game, and programs the using time for player moving chess on one step. The mobile robot has the shape of cylinder, and it's equipped with a microchip (MCS-51) as the main controller [9].

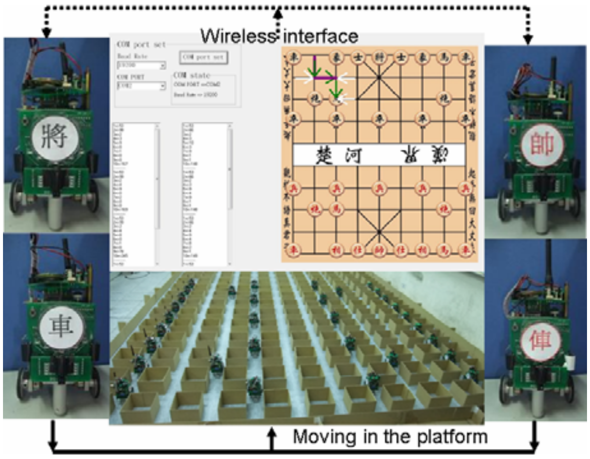


Fig. 1. System architecture

We implement the Chinese chess game using mobile robots on the grid based chessboard platform to be shown in the bottom of Fig. 1. The arrangement of the chessboard is 11 grids on the horizontal direction (X axis), and is 12 grids on the vertical direction (Y axis). The game only uses 9×10 grids based platform to arrange all chesses. The taken chesses move to the assigned positions around the chessboard. The distance is 30cm between the center of aisle on the X direction and Y direction of the chess board, and the width of the corridor is 12cm. The mobile robot uses the IR sensor modules to detect static and dynamic obstacles, and stays at the cross points of the chessboard platform.

The communication protocol of the system is 10 bytes. There are start byte (1 byte), data byte (8 bytes) and check byte (1 byte). We set the communicate protocol from the supervised computer to the mobile robot to be listed in Table 1. The start byte trigs the wireless RF module to receive the signals from the transmitter. The data bytes contain ID code (one byte), robot number (one byte), and movement direction and distance of the mobile robot. The definition of the movement direction is listed in right side of Fig. 2 for mobile robots. There are four various directions to match the grid based platform. The communication protocol is listed in Table 2 from the mobile robot to the supervised computer. The data bytes contain ID code, robot number, and position and orientation of robots and obstacles and finished command feedback data.

Table 1. Control command

Byte	0	1	2	3	4
Definition	Start	ID	R_CH	Work[1]	Work[2]

5	6	7	8	9
Work[3]	Work[4]	R_D	GO_flag	Check

Table 2. Feedback command

Byte	0	1	2	3	4
Definition	Start	ID	R_CH	Work[1]	Work[2]

5	6	7	8	9
Work[3]	Work[4]	No use	Text	Check

III. Motion Planning

We use A^* searching algorithm to solve the shortest path problem of multiple nodes travel system. The formula of A^* searching algorithm is following

$$f(n) = g(n) + h(n) \tag{1}$$

The core part of an intelligent searching algorithm is the definition of a proper heuristic function $f(n)$. $g(n)$ is the exact cost at sample time n from start point to the target point. $h(n)$ is the minimum cost. In this study, n is reschedules as n' to generate an approximate minimum cost schedule for the next point. The equation (1) can be rewritten as follows:

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

The programmed motion path only computes the movement distance, and can't considers the total displacement using A* searching algorithm. We use pulse numbers of the encoder to define the total movement cost $f(n)$. The pulse numbers are 355 to move one grid on the platform, and the pulse numbers are 92 to rotate 90^0 for the mobile robot. The total displacement contains movement distance and turning numbers. The paper uses enhance A* searching algorithm to programs the shortest motion path, and considers the turning numbers. We can program seven steps to compute the shortest motion path using the proposed algorithm to be following:

Step 1: We construct two labels to be Open list and Close list.

Step 2: The "Close list" fills the start point and evaluation points, and the neighbour points of the start point fill in the "Open list".

Step 3: We construct label on the first searching result, and calculate the values of $f(n)$, $g(n)$ and $h(n)$ function, and compares the values of the function. Then we select the minimum value of the function $f(n)$ to be stored in "Close list".

Step 4: Repeat the processing from step 2 to step 4.

Step 5: We can find the target point on the final searching.

Step 6: We compare the motion paths on the neighbour points of each turning points, and we can decide a shortest path to control the mobile robot moving to the target point.

Step 7: The "Open list" can't include the target point. We decide the path does not exist.

We make an example to explain the motion planning using enhance A* searching algorithm. Player moves the chess "red cannon" to take the chess "black horse". The user interface programs the motion path using A* searching algorithm to be shown in Fig. 2, and avoid the obstacles (chesses). The programmed motion path turns right four times and turns left four times. We can find out the four points to make the landmarks "B" "C" "D" and "4", and select the new motion path from the landmarks "4" "C" and "B" to reduce two turning numbers. The experimental results are shown in Fig. 3(a) and (b).

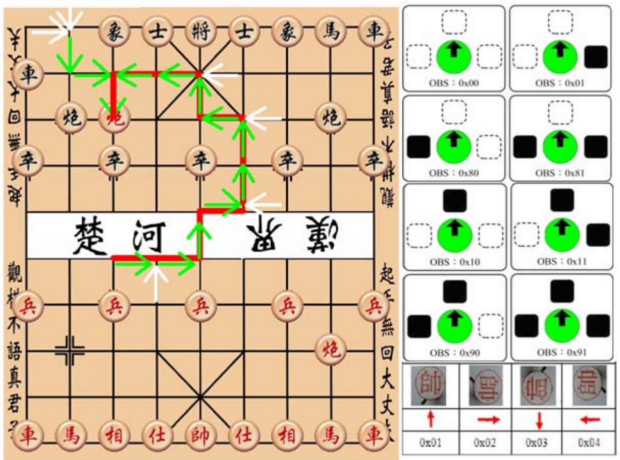


Fig. 2. Path planning

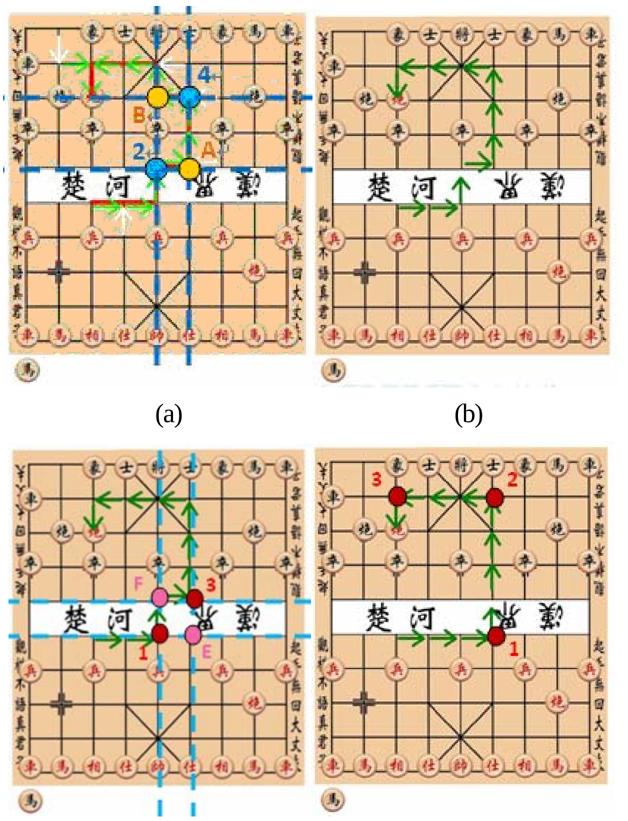


Fig. 3. Re-programmed motion path

Then we can find out new programming motion path to make four landmarks "1" "E" "3" and "F". We can select the new motion path through three points "1" "E" and "3" to instead of the old motion path through three points "1" :F" and "3". The new path can reduces two turning numbers to be shown in Fig. 3(c). Finally, we can search the shortest motion path to be shown in Fig. 3(d). The motion path turns left two times and turns right two times.

We compare the pulse numbers of the two motion paths. The total pulse numbers of the old shortest path $f(n)$ can be calculated as

$$f(n) = 11 \times 355 + 8 \times 92 = 4694 \qquad (3)$$

In the other condition, we rebuild enhance A* searching algorithm on the step 6. We can calculate the pulse numbers of the new motion path moving to the target position. The total pulse numbers $f(n)$ as

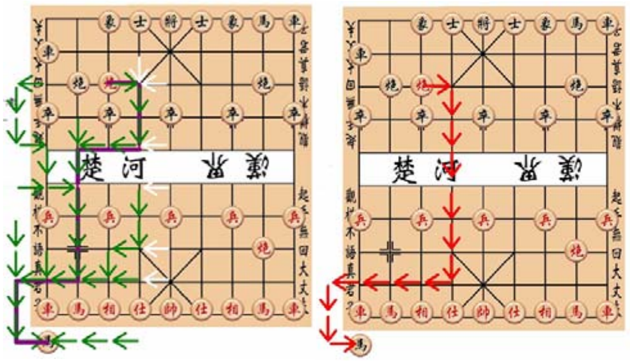
$$f(n) = 11 \times 355 + 4 \times 92 = 4218 \qquad (4)$$

IV. EXPERIMENTAL RESULTS

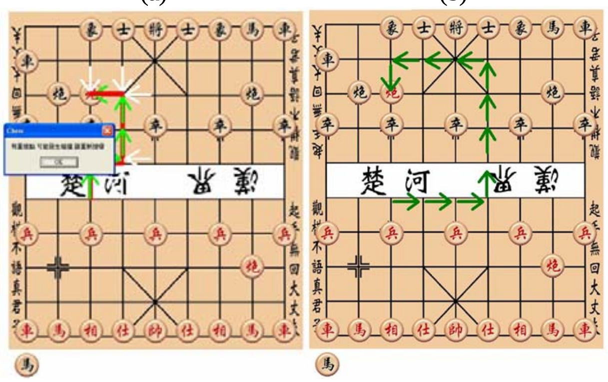
Users move the chess to play Chinese chess game, and present movement scenario of mobile robots using enhance A* searching algorithm. The experimental scenario is the chess "red cannon" moving to take the chess "black horse" on the user interface. The user interface programs two motion paths using A* searching algorithm for the assigned two robots. The first motion path is the chess "black horse" moving to the back side of the red side. The programmed motion path is shown in Fig. 4(a). The motion paths have eight times to turn right or left, and must spend more time moving to the target position for the mobile robot. The user interface programs the new motion path using enhance A* searching algorithm. The new motion path reduces turning right one time and turning left one time. The supervised computer orders the command to the mobile robot "black horse" via wireless RF interface. The mobile robot moves one grid, and turns right 90^0 moving six grids. Then it turns right 90^0 moving four grids. The chess moves to the final position to be shown in Fig. 4(b).

Then the user interface programs the motion path for the chess "red cannon" that moves to the position of the chess "black horse". The assigned motion path has the collision points with the pre-programmed motion path of the chess "black horse". The user interface displays the alarm signal to re-program the new motion path. The experimental result is shown in Fig. 4(c). The programming processing is shown in Fig. 3. We can find out the final motion path of the chess "red cannon" to be shown in Fig. 4(d).

The user interface has been programmed the two motion paths for two chesses "red cannon" and "black horse". Then the supervised computer orders command to the two mobile robots (red cannon and black horse) moving on the chessboard platform according to the programmed motion paths. The two mobile robots move on the platform simultaneously, and speak the movement status using Chinese language. The movement scenario is shown in Fig. 5(a). We plot the motion trajectories of the two mobile robots in Fig. 5. Finally, we can see the chess "black horse" moving to the back side of the chess "red rook", and see the robot "red cannon" moving to the target position.

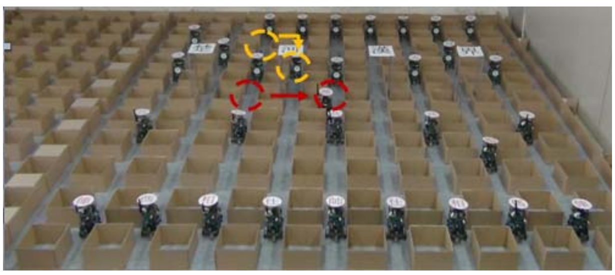


(a) (b)

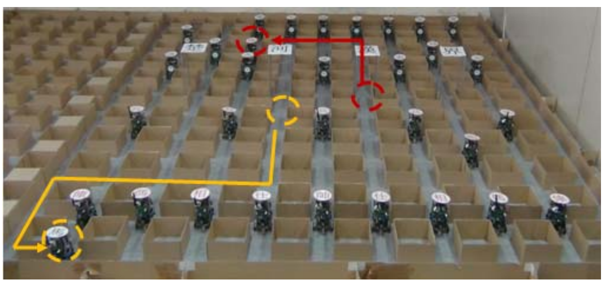


(c) (d)

Fig. 4. Motion paths of two chesses



(a)



(b)

Fig. 5. Movement scenario of two robots

V. CONCLUSION

We have presented a Chinese chess game system using multiple mobile robots. Mobile robots execute the chess attribute using two interfaces. One is wireless RF interface, and the other is voice interface. The user interface displays the total time to play the game, and computes using time to move chess by player on one step, and receive the status of mobile robots via wireless RF interface. The user interface programs the shortest path using evaluation algorithm and A* searching algorithm. Then the user interface improves the motion paths to reduce the turning numbers, and avoid static and dynamic obstacles (chesses) using enhance A* searching algorithm. Users can move the chess using the mouse on the supervised computer, and moves the chess in the assigned time.

ACKNOWLEDGMENT

This work was supported by the National Science Council of Taiwan, (NSC 101- 2221- E-224-007)

REFERENCES

[1] Yen SJ, Chen JC, Yang TN and Hsu SC (2004), Computer Chinese Chess, ICGA Journal, Vol.27, No. 1, pp.3-18.

[2] Darwen P and Yao X (2002), Coevolution in Iterated Prisoner's Dilemma with Intermediate Levels of Cooperation: Application to Missile Defense, International Journal of Computational Intelligence Applications, Vol. 2, No. 1, pp.83-107.

[3] Wang J, Luo YH, Qiu DN and Xu XH (2005), Adaptive Genetic Algorithm's Implement on Evaluation Function in Computer Chinese Chess, Proceeding of ISCIT, pp.1206-1209.

[4] Su KL, Shiau SV, Guo JH and Shiau CW (2009), Mobile Robot Based Onlin Chinese Chess Game, The Fourth International Conference on Innovative Computing, Information and Control, pp.63.

[5] Fu T and Yin H (2012), Designing a hybrid position evaluation function for Chinese-chess computer game, International Conference on Software Engineering and Service Science, Beijing, China, pp.75-78.

[6] Zhou W, Liu JC and Zhao YH (2011), The construction of Chinese chess computer game platform, International Conference on Computer Science & Education, SuperStar Virgo, Singapore, pp.126-128,.

[7] Kong Z, Gao LQ, Wang LF and Ge YF, Onan adaptive harmony search algorithm, International journal of Innovative Computing, Information and Control, Vol. 5, No. 9, pp.2551-2560.

[8] Flavio CM and Xiao JH (2012), Aflood algorithm for multirobot exploration, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, Vol. 42, No. 3, pp.850-863.

[9] Su KL, Chung CY, Liao YL and Guo JH (2011), A* Searching Algorithm Based Path Planning of Mobile Robots, The Innovative Computing, Information and Control – Express Letters, Part B: Applications (ICIC-ELB),Vol. 2, No. 1, pp.273-278.