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

A PSO method for time-optimal motion planning of a mobile robot

Li-Chun Lai[1], Kuo-Lan Su [2], Chia-Nan Ko[3], Sheau-Wen Lin[4], and Chia-Hung Shih[1]

[1] Bachelor Program in Robotics, National Pingtung University of Education
Pingtung City, Pingtung County 900, Taiwan
[2]Department of Electrical Engineering, National Yunlin University of Science and Technology
Douliou, Yunlin 640, Taiwan
[3]Department of Automation Engineering, Nan Kai University of Technology
Tasotun, Nantou 542, Taiwan
[4] Graduate Institute of Mathematics and Science Education, National Pingtung University of Education
Pingtung City, Pingtung County 900, Taiwan

(email:[1]lclai@mail.npue.edu.tw; [2]sukl@yuntech.edu.tw; [3]t105@nkut.edu.tw; [4]linshewen @mail.npue.edu.tw;
[5]hiroshi.seki@gmail.com)

Abstract: Based on a particle swarm optimization (PSO) algorithm for time-optimal control problem of a two-wheeled mobile robot is addressed in this paper. The PSO algorithm is that it is easier to implement and there are fewer parameters to be adjusted. Different from usual cases, in which the Pontryagin's Minimum Principle (PMP) is used, an iterative procedure is proposed to transform the time-optimal problem into a nonlinear programming (NLP) one. Motion planning of a mobile robot is a NLP problem. In the NLP problem, the count of control steps is fixed initially and the sampling period is treated as a variable in the optimization process. Because the NLP has a initial feasible solutions problem that is not easier to find. In this paper, The PSO algorithm and fitness function to solve initial feasible solutions problem and optimal problem. To show the feasibility of the proposed method, simulation results are included for illustration.

Keywords: Particle Swarm Optimization (PSO), Mobile Robot, Time-optimal Control.

1. INTRODUCTION

Motion planning is an important issue in mobile robotics. In recent years, many research works have been addressed the control problem associated with mobile robots [1,2]. Among these works, a typical mission for a mobile robot can be described as to move from one configuration (position and orientation) to another. This problem has been thoroughly studied by assuming that the motors control the wheel velocities [3]. On the other hand, one also can use the motor to control the accelerations of the wheels [4]. For mobile robots equipped with two independently driven wheels, the time-optimal motion planning problem is that of finding the time-optimal motion in an unobstructed environment between two configurations. Usually, this kind of time-optimal control problem leads to the utilization of the Pontryagin's Maximum Principle (PMP) [5], in which the dynamical equations of the mobile robot are used and one needs to solve a set of differential equations. Since this set of equations is usually highly nonlinear and coupled, it is very difficult to obtain the closed-form solution for the time-optimal trajectory of any mobile robot. Because of the disadvantage in applying the PMP, a NLP method that does not utilize the PMP was developed by one of the authors of this paper to solve the time-optimal control problem of linear systems [6]. The basic idea of this method is that instead of considering a fixed sampling period, the count of control steps is fixed initially and the sampling period is treated as a variable in the optimization process. Extending the concept in [6] to nonlinear systems, this paper shows the generation of time-optimal motion between two configurations for a mobile robot with two independently driven individual wheels. In the beginning of this study, a mobile robot is introduced and its kinematic equations are derived. Then an iterative procedure is proposed to transform the time-optimal problem into a constrained NLP one. However, since the NLP has a initial feasible solutions problem that is not easier to find and two level time-optimal process is very complex. In this paper, The PSO algorithm and fitness function to solve initial feasible solutions problem and time-optimal problem.

Particle Swarm Optimization (PSO) algorithm proposed by Kennedy and Eberhart [7] is another kind of evolutionary computation techniques. This method was based on the simulation of simplified animal social behaviors such fish schooling and bird flocking. In PSO searching space, each single solution acts as a flying bird, which we call it a particle. The PSO algorithm works on a social behavior of particles in the swarm. It finds the global best solution by simply adjusting the trajectory of each particle its own best particle and toward the best particle of the entire swarm at each generation. The PSO algorithm has been used in solving many optimization problems successfully. Compared with GA, the advantage of PSO is that it is easier to implement and there are fewer parameters to be adjusted. In this paper, The PSO algorithm is proposed to determine velocities of two wheels. The path of a mobile robot is time-optimal from a given initial configuration to a desired final configuration.

The remaining sections of this paper are organized as follows. Section 2 shows kinematics equations of a mobile robot. Section 3 presents time-optimal control between two configurations. Section 4 presents time-optimal motion planning with PSO algorithm. Simulations are performed in Section 5 to confirm the feasibility of the proposed algorithm. Section 6 concludes the paper.

2. KINEMATICS EQUATIONS OF A MOBILE ROBOT

In the section, the kinematics of a mobile robot with two independent driven wheels will be described. The configuration of the mobile robot will be denoted by (x, y, θ) as shown in Fig. 1. Where x and y are the coordinates of the midpoint of the wheel axis, and θ is the heading angle of the robot. L is the distance between two wheels and r is the radius of the wheel.

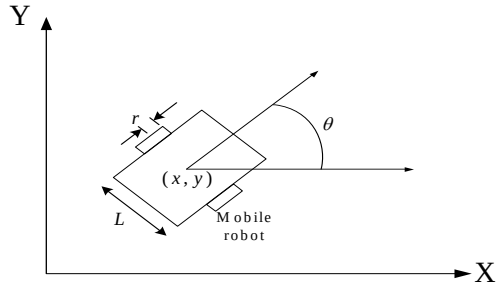

Fig. 1. A diagram of a mobile robot

By assuming that the wheels do not slip, the kinematics of the mobile robot as shown in (1).

$$\begin{bmatrix} \dot{\omega}_r(t) \\ \dot{\omega}_l(t) \\ \dot{\theta}(t) \\ \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} u_r(t) \\ u_l(t) \\ \dfrac{r(\omega_r(t)-\omega_l(t))}{L} \\ \dfrac{r\cos\theta(t)(\omega_r(t)+\omega_l(t))}{2} \\ \dfrac{r\sin\theta(t)(\omega_r(t)+\omega_l(t))}{2} \end{bmatrix} \quad (1)$$

where $\omega_r(t)$ and $\omega_l(t)$ represent the angular velocities of the right and left wheels, respectively. $u_r(t)$ and $u_l(t)$ are the control inputs.

3. TIME-OPTIMAL MOTION PLANNING PROBLEM BETWEEN TWO CONFIGURATIONS

The time-optimal motion planning problem between two configurations is to find the control inputs that will move the initial configuration (x_0, y_0, θ_0) to the final configuration (x_f, y_f, θ_f) while minimizing the traveling time. The control inputs $u_r(t)$ and $u_l(t)$ are assumed to meet the following constraints in (2) and (3).

$$u_{\min} \le u_r(t) \le u_{\max} \quad (2)$$

$$u_{\min} \le u_l(t) \le u_{\max} \quad (3)$$

The mobile robot moves time $[0, t_f]$ is to divide into *N* equal time intervals as shown in (4).

$$\Delta t = \frac{t_f}{N} \quad (4)$$

With (1) through (3), it is turned into (5) through (7).

$$\begin{bmatrix} \omega_r(i\cdot\Delta t) \\ \omega_l(i\cdot\Delta t) \\ \theta(i\cdot\Delta t) \\ x(i\cdot\Delta t) \\ y(i\cdot\Delta t) \end{bmatrix} = \begin{bmatrix} \omega_r((i-1)\cdot\Delta t)+u_r(i\cdot\Delta t)\cdot\Delta t \\ \omega_l((i-1)\cdot\Delta t)+u_l(i\cdot\Delta t)\cdot\Delta t \\ \theta((i-1)\cdot\Delta t)+\dfrac{r(\omega_r(i\cdot\Delta t)-\omega_l(i\cdot\Delta t))}{L}\cdot\Delta t \\ x((i-1)\cdot\Delta t)+\dfrac{r\cos\theta(i\cdot\Delta t)(\omega_r(i\cdot\Delta t)+\omega_l(i\cdot\Delta t))}{2}\cdot\Delta t \\ y((i-1)\cdot\Delta t)+\dfrac{r\sin\theta(i\cdot\Delta t)(\omega_r(i\cdot\Delta t)+\omega_l(i\cdot\Delta t))}{2}\cdot\Delta t \end{bmatrix}$$

$$\text{for } i = 1, 2, \ldots, N \quad (5)$$

$$u_{\min} \le u_r(i\cdot\Delta t) \le u_{\max} \quad \text{for } i = 1, 2, \cdots, N \quad (6)$$

$$u_{\min} \le u_l(i\cdot\Delta t) \le u_{\max} \quad \text{for } i = 1, 2, \cdots, N \quad (7)$$

With (5) through (7), the time-optimal motion planning can be formulated. Assume the following conditions are given:

$$(x(0), y(0), \theta(0)) = (x_0, y_0, \theta_0) \quad (8)$$

$$\omega_r(0) = 0 \quad (9)$$

$$\omega_l(0) = 0 \quad (10)$$

Determine the control inputs $u_r(t)$ and $u_l(t)$ for $[0, t_f]$ to minimize as shown in (11).

$$J = N \cdot \Delta t \tag{11}$$

Subject to

$$(x(t_f), y(t_f), \theta(t_f)) = (x_f, y_f, \theta_f) \tag{12}$$

$$\omega_r(t_f) = 0 \tag{13}$$

$$\omega_l(t_f) = 0 \tag{14}$$

and the constraints in (6) and (7).

4. TIME-OPTIMAL MOTION PLANNING PROBLEM WITH PSO ALGORITHM

The mathematic description of PSO is as the follow:

Suppose the dimension of the searching space is D, the number of particle is n. Vector $\vec{x}_j = (x_{j1}, x_{j2}, \ldots, x_{jD})$ represents the position of the i-th particle and $P_j = (p_{j1}, p_{j2}, \ldots, p_{jD})$ is the best position searched by now, and the whole particle swarm's best position is represented as $P_g = (p_{g1}, p_{g2}, \ldots, p_{gD})$ Vector $\vec{v}_j = (v_{j1}, v_{j2}, \ldots, v_{jD})$ is the position change rate of the i-th particle. Each particle updates its position according to the following formulas:

$$\begin{aligned} v_{jd}(k+1) &= w \times v_{jd}(k) + c_1 \times rand(\,) \times (p_{jd}(k) - x_{jd}(k)) \\ &\quad + c_2 \times rand(\,) \times (p_{gd}(k) - x_{jd}(k)) \end{aligned} \tag{15}$$

$$x_{jd}(k+1) = x_{jd}(k) + v_{jd}(k) \tag{16}$$

Where c_1 and c_2 are positive constant parameters called acceleration coefficients. The $rand(\,)$ is a random function with the range [0,1]. w is called the inertia weight and is less than 1.

The fitness function of PSO algorithm is defined as shown in

$$fitness = \frac{1}{\lambda e e^T + 1} \tag{17}$$

where e is a error between the final states of the mobile robot and final configuration. The λ is a penalty value that the error e is convergence to zero fastly. In order to reach final configuration in optimal time, the paper is used PSO algorithm repeatedly. The method is used the optimal time of the last to replace the next initial time until the maximum iteration.

The details of the determination of the attractive factor can be summarized as follows:

Algorithm:

Step 1: Define the problem space.

Step 2: Initialize an array of particles with random positions and velocities.

Step 3: Evaluate the desired fitness function of the particles using equation (17).

Step 4: Determine the best personal position visited so far by each particle.

Step 5: Determine the best global position visited so far by all the particles.

Step 6: Update velocities using Eq. (15).

Step 7: Update particles' positions using Eq. (16).

Step 8: Repeat the procedure in Step 3 through Step 7 until all the particles have attained their desired fitness.

5. SIMULATION RESULTS

Example 1:

In this simulation example, the mobile robot will be moved from $(x_0, y_0, \theta_0) = (0,0,0)$ to $(x_f, y_f, \theta_f) = (1\text{m}, 1\text{m}, \frac{\pi}{3})$. The initial time of Δt, the initial value of *N*, the radius of the wheel r, and the distance between two wheels L are chosen to be 5 (sec.), 10, 5 cm, and 30 cm, respectively. Meanwhile, the bounds on the control inputs, $u_{\min}$ and $u_{\max}$ are chosen to be –0.5 rad/s^2 and 0.5 rad/s^2. In applying PSO algorithm, the population size and maximal generation number are chosen to be 1000 and 1000, respectively. The value c_1, c_2, the inertia weight, and λ are chosen to be 2, 2, 0.8, and 10000, respectively. The error e will be defined as in (18).

$$e = [\Delta x_f \;\; \Delta y_f \;\; \Delta\theta_f \;\; \Delta\omega_{rf}(t_f) \;\; \Delta\omega_{lf}(t_f)] \tag{18}$$

$$\Delta x_f = x(t_f) - x_f \tag{19}$$

$$\Delta y_f = y(t_f) - y_f \tag{20}$$

$$\Delta\theta_f = \theta(t_f) - \theta_f \tag{21}$$

$$\Delta\omega_{rf} = \omega_r(t_f) - 0 \tag{22}$$

$$\Delta\omega_{lf} = \omega_l(t_f) - 0 \tag{23}$$

Applying PSO algorithm, the optimal time of Δt is found to be $\Delta t = 1.6073$ (sec.) and the plots of $x(t)$, $y(t)$, $\theta(t)$, $\omega_r(t)$, $\omega_l(t)$, $u_r(t)$, and $u_l(t)$ are shown in Fig. 2 through 8, respectively.

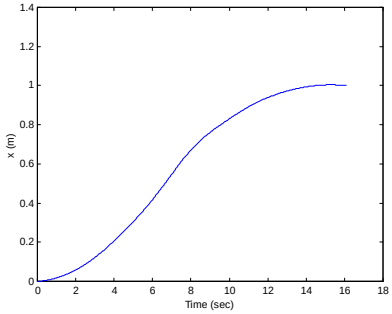

Fig. 2. Plot of $x(t)$

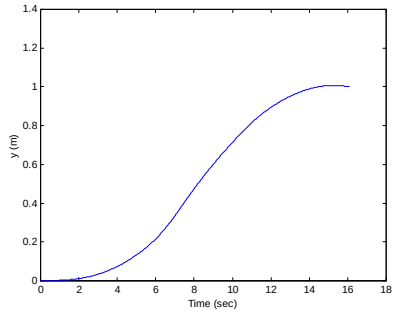

Fig. 3. Plot of $y(t)$

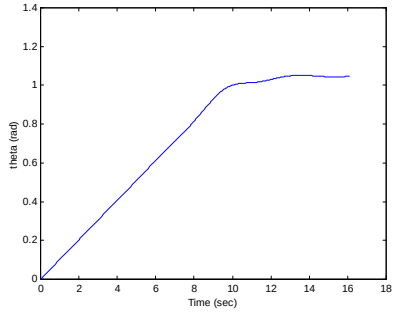

Fig. 4. Plot of $\theta(t)$

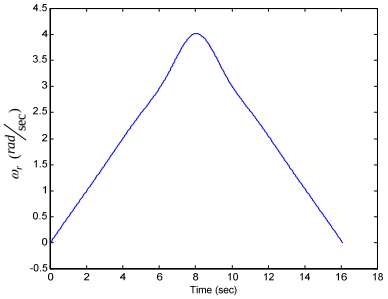

Fig. 5. Plot of $\omega_r(t)$

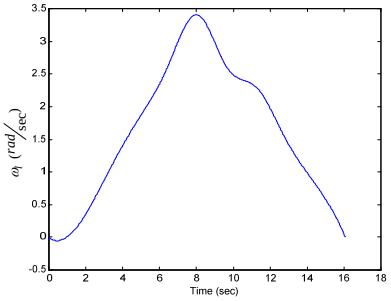

Fig. 6. Plot of $\omega_l(t)$

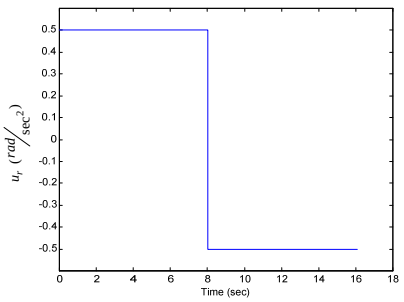

Fig. 7. Plot of $u_r(t)$

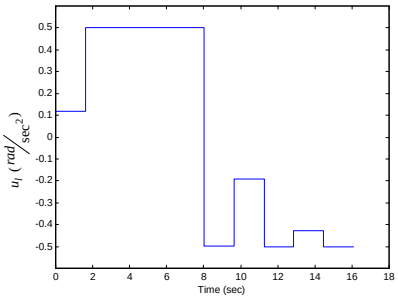

Fig. 8. Plot of $u_l(t)$

6. CONCLUSIONS

The paper uses PSO algorithm to solve time-optimal motion planning problem of a mobile robot with two independently driven wheels. The PSO algorithm and fitness function to solve initial feasible solutions problem and optimal problem. Compared with GA, the advantage of PSO is that it is easier to implement and there are fewer parameters to be adjusted. The simulation results can show the feasibility of the proposed method.

ACKNOWLEDGMENTS

This work was supported in part by the National Pingtung University of Education, Taiwan, R.O.C.

REFERENCES

[1] Wang SM, Lai LC, Wu CJ Shiue YL (2007), Kinematic control of omni-directional robots for time-optimal movement between two configurations. Journal of Intelligent and Robotic Systems 49(4):397-410

[2] Lai LC, Wu CJ, Shiue YL (2007), A potential field method for robot motion planning in unknown environments. Journal of the Chinese Institute of Engineers 30(3):369-377

[3] Soueres P, Laumond JP (1996), Shortest paths synthesis for a car-like robot. IEEE trans. on Automatic Control 41(5):672-688

[4] Reister DB, Pin FG (1994), Time–optimal trajectories for mobile robots with two independently driven wheels. The International Journal of Robotics Research 13(1): 38-54

[5] Pontryagin LS, Boltyanskii VG, Gamkrelidze RV, Mishchenko EF (1986), The Mathematical Theory of Optimal Processes Gordon and Breach Science Publishers. New York

[6] Chung TS, Wu CJ (1992), A computationally efficient numerical algorithm for the minimum-time control problem of continuous systems. Automatica 28:841-847

[7] Kennedy J, Eberhart R (1995), Particle swarm optimization. IEEE International Conference on Neural Network, pp. 1942-1948