Optimal motion planning of differential-drive mobile robots based on trapezoidal collocation method

R Mao\textsuperscript{1,2,3}, H Gao\textsuperscript{1,2} and L Guo\textsuperscript{1,2}

\textsuperscript{1}School of Mechanical Engineering, Southwest Jiaotong University, Chengdu, P.R. China
\textsuperscript{2}Engineering Research Center of Advanced Driving Energy-saving Technology, Ministry of Education, Southwest Jiaotong University, Chengdu, P.R. China
\textsuperscript{3}No. 111, North 1st Section of Second Ring Road, Jinniu District, Chengdu City, Sichuan Province, PRC

E-mail: maorun@my.swjtu.edu.cn

Abstract. This study focuses on optimal motion planning for nonholonomic constraints mobile robot. We formulate the dynamics model of a differential-drive mobile robot by using Lagrangian mechanics, where the nonholonomic constraints are accurately described through differential equations. The optimal motion planning of the system is constructed as an optimal control problem which is then converted to a nonlinear programming problem by introducing trapezoidal collocation method, and the formulated nonlinear programming is solved by interior-point method. Compared with the prevailing methods in the field of motion planning, our proposed method can handle different kinds of path constraints, terminal conditions and collision-avoidance requirements. Simulation results indicate that the proposed approach can efficiently deal with various user-specified requirements with advantage of high computing efficiency.

1. Introduction

Wheeled mobile robots (WMR) have received booming interests from the research communities and industrial circles due to their capacity to operate and maneuver in the extended workspaces. Differential drive mobile robot (DDMR) is one of the most widely used WMR. The motion planning for DDMR in cluttered environments is an inherently difficult problem [1]. First, the path has to be physically feasible and meets the nonholonomic constraints. Second, the generated path should avoid the multiple obstacles. Last but not least, the system dynamics of the WMR have to be satisfied.

The biggest difference between the path planning and motion planning is that the motion contains the path, time and control. Therefore, the motion planning is to solve the problem about how to produce a continuous trajectory that connects a start configuration and a goal configuration. Traditional combinatorial approaches for motion planning find paths through the continuous configuration space without resorting to approximations, which have been applied in the motion planning field widely [1]. Virtually all combinatorial motion planning approaches construct a roadmap along the way to solving queries. In visibility graph method, the vertices of polygonal obstacles are corresponding to nodes, and the edges linking the notes are corresponding to straight potential paths. Then the shortest path algorithms such as Dijkstra’s algorithm [2] and genetic heuristics [3] are applied to the generated graph. Likewise, the voronoi diagrams
method [4] and cell decomposition [5] algorithms are the typical combinatorial approaches, which convert the motion planning problem to a graph search problem by transforming the configuration space \( C_{\text{free}} \) into a finite set of regions called cells. However, these techniques show drawbacks of high time complexity in high dimension spaces.

The sampling-based planning is another kind of methods for motion planning, which use collision-detection to probabilistically and incrementally search the configure space for solution [6]. Probabilistic roadmap and rapidly exploring random trees are the leading algorithms of sampling-based planning. The downside is the fact that the solution is suboptimal and the dynamic constraints are hard to be formulated.

Motion planning with differential and dynamic constraints is substantially more difficult to solve due to the dependency between time and the state-space [7]. This kind of problems may be more suitably formulated in the optimal control framework, in which the solution of the problem is a trajectory [8]. In comparison with trajectory optimization, the optimal control provides an excellent ability to address the problems with system nonlinearities of multiple constraints.

The numerical methods to optimal control problems can be divided into two classes: indirect methods and direct methods [9]. In indirect method, the original optimal control problem is first transformed into a multiple-point boundary-value problem (MPBVP) by using calculus of variations. In direct methods, the optimal control problem is discretized and transcribed to a nonlinear optimization problem or nonlinear programming problem (NLP). There are many software programs for solve NLP, such as FMINCON, SNOPT, and IPOPT.

Indirect methods can achieve more accurate solutions than direct methods, at the cost of more difficulty to construct and solve [10]. Because the former one need to analytically construct the necessary and sufficient conditions of optimality for the original problem, while the latter methods only solve a discrete approximation of the original problem [11]. The direct shooting method and the direct collocation method are the most important direct methods. Direct shooting methods are suitable for the applications with simple control and few path constraints, since the intermediate state variables are not decision variables of the NLP. The low-order direct collocation method is speedy but poor in precision, while the high-order method is slow but precise [11]. With additional segments, the precision of the low-order method can be improved. These methods are widely used in the field of robot motion planning. However, most of these only considered the kinematics constraint and ignored the dynamics constraint.

It is impossible to judge whether one approach is better than another, since different methods are applied in certain scenarios. In this research, we regard motion planning as an optimal control problem, in which the nonholonomic constraints, dynamics constraints, collision-avoidance requirement, and additional mechanical constraints are strictly described. Considering the trade-offs between computational efficiency and solution accuracy, the trapezoidal collocation method may be the most efficient numerical approach to solve this optimal control problem. With the trapezoidal collocation method, the optimal control problem is converted to a large scale NLP. The interior-point method (IPM) is an effective solver to handle NLPs. The resulted NLP is solved by a IPM which is a standard algorithm of MATLAB optimization toolbox solver. Finally, the effectiveness and capability of the proposed approach are demonstrated through simulation studies.

2. Kinematic and dynamic model
A differential drive vehicle has two independently actuated wheels on a common axis at a distance \( 2b \), as described in Figure 1. These wheels are generally subjected to nonholonomic constraints, in other words, the number of the controllable DOF (2) is smaller than the number of DOF (3) of the robot. In this section, the theoretical background of nonholonomic constraints is discussed, and the kinematic and dynamic equations of DDMR are presented. The symbols in Figure 1 are explained in Table 1.
Table 1. The symbols and parameters of the DDMR.

| Symbols | Description |
|---------|-------------|
| C       | the center of mass point |
| A       | the mid-point of the axis |
| b       | the length between two driven wheels |
| d       | the half length between point C and A |
| r       | the radius the of driven wheel |
| φ       | the orientation in the inertial frame |
| θ₁r     | the angular position of the right wheel |
| θ₁l     | the angular position of the left wheel |

2.1. Kinematics for the vehicle

As shown in Figure 1, two coordinate frames are defined as: the world-fixed inertial reference frame $O_I$ and the robot-fixed reference frame with origin at the center of axis $O_R$. Due to the fact that the trajectory of the vehicle is constrained to the horizontal plane, the position and orientation in these two frames can be defined as $\xi_I = [x_I, y_I, \phi_I]^T$ and $\xi_R = [x_R, y_R, \phi_R]^T$, respectively. A clear mapping between two reference frame is required.

$$\xi_I = R \xi_R$$

where

$$R = \begin{bmatrix} \cos \phi & \sin \phi & 0 \\ -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(2)

The motion of the vehicle is characterized by two nonholonomic constraints, which are obtained by two assumption: no lateral slip constraint and pure rolling constraint. Therefore, the three constraint equations are denoted as:

$$\Lambda(q) \dot{q} = 0$$

(3)

where

$$\Lambda(q) = \begin{bmatrix} -\sin \phi & \cos \phi & -d & 0 & 0 \\ \cos \phi & \sin \phi & b & -r & 0 \\ \cos \phi & \sin \phi & -b & 0 & -r \end{bmatrix}$$

(4)

$$q = [x, y, \phi, \theta_r, \theta_l]^T$$

(5)
The linear velocity of the DDMR in the robot frame is the average linear velocity of the two wheels
\[ v = \frac{r}{2}(\dot{\theta}_r + \dot{\theta}_l) \] (6)
and the angular velocity equals
\[ \omega = \frac{r}{2b}(\dot{\theta}_r - \dot{\theta}_l) \] (7)

Then, the velocity of point C can be represented as
\[ \dot{\xi}_R = \begin{bmatrix} \frac{r}{2}(\dot{\theta}_r + \dot{\theta}_r) \\ \frac{dr}{2b}(\dot{\theta}_r - \dot{\theta}_r) \\ \frac{r}{2b}(\dot{\theta}_R - \dot{\theta}_r) \end{bmatrix} \] (8)

The forward kinematic equation of the DDMR can be expressed as:
\[ \dot{q} = S(q)\beta \] (9)

where
\[ S(q) = \begin{bmatrix} \frac{r}{2} \cos \phi - \frac{dr}{2b} \sin \phi & \frac{r}{2} \cos \phi + \frac{dr}{2b} \sin \phi \\ \frac{r}{2} \sin \phi + \frac{dr}{2b} \cos \phi & \frac{r}{2} \sin \phi - \frac{dr}{2b} \cos \phi \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \beta = \begin{bmatrix} \dot{\theta}_r \\ \dot{\theta}_l \end{bmatrix} \] (10)

2.2. Dynamics for the vehicle
A mobile robot subjected to \( m \) constraints, which has a \( n \)-dimensional configuration space \( C \) with generalized coordinates \( (q_1, q_2, \cdots, q_n) \), can be described by
\[ M(q)\ddot{q} + V(q, \dot{q})\dot{q} + F(q) + G(q) = B(q)\tau + \Lambda^T(q)\lambda \] (11)

where \( M(q) \in \mathbb{R}^{n \times n} \) is a symmetric, positive definite inertia matrix, \( V(q, \dot{q}) \in \mathbb{R}^{n \times n} \) represents the centripetal and coriolis matrix, \( F(q) \in \mathbb{R}^{n \times 1} \) denotes the surface friction, \( G(q) \in \mathbb{R}^{n \times 1} \) is the gravitational vector, \( B(q) \in \mathbb{R}^{n \times r} \) is the input transformation matrix, \( \tau \in \mathbb{R}^{n \times 1} \) means the input vector, \( \Lambda \in \mathbb{R}^{m \times n} \) is the matrix associated with the constraints, and \( \lambda \in \mathbb{R}^{m \times 1} \) is the vector of constraint forces.

For systems with holonomic constraints, all constraints are integrated into geometrical constraints. If the constraints are nonholonomic, this approach does not work. There is no general method to handle the nonholonomic problems. The dependent equations can be eliminated by the method of Lagrange multipliers only when special nonholonomic constraints are given in differential form. For constraints in the form of equalities, the Lagrange equation of the first kind can be written in the following form:
\[ \frac{d}{dt}(\frac{\partial L}{\partial \dot{q}}) - \frac{\partial L}{\partial q} = B(q)\tau + \Lambda^T(q)\lambda \] (12)

We know that \( L = T - U \). However, the mobile robot has a horizontal restricted dynamics that yields \( U = 0 \). The total kinetic energy is composed of the kinetic energy of the robot base (\( T_b \))
and driven wheels ($T_r$ and $T_l$).

\[
T_p = \frac{1}{2} m_b (\dot{x}_b^2 + \dot{y}_b^2) + \frac{1}{2} I_b \dot{\phi}^2 \\
T_r = \frac{1}{2} m_w (\dot{x}_{wr}^2 + \dot{y}_{wr}^2) + \frac{1}{2} I_m \dot{\phi}_r^2 + \frac{1}{2} I_w \dot{\theta}_r^2 \\
T_l = \frac{1}{2} m_w (\dot{x}_{wl}^2 + \dot{y}_{wl}^2) + \frac{1}{2} I_m \dot{\phi}_l^2 + \frac{1}{2} I_w \dot{\theta}_l^2
\]

(13)

where $m_b$ is the mass of the robot platform without wheels and actuators, $m_w$ denotes the mass of each driving wheel with actuator, $I_b$ equals the moment of inertia of the DDMR (without wheels and actuators) about the vertical axis through the point $C$, $I_w$ shows the moment of inertia of each driving wheel with a motor about the wheel axis, and $I_m$ represents the moment of inertia of each driving wheel with an actuator about the wheel diameter.

Thus, the total kinetic energy is

\[ L = T = T_b + T_r + T_l \]  

(14)

Using (12-14) along with (11), the motion equation of the system are given by

\[ M(q) \ddot{q} + V(q, \dot{q}) \dot{q} = B(q) \tau + \Lambda^T(q) \lambda \]  

(15)

where

\[
M(q) = \begin{bmatrix}
  m & 0 & 2m_w d \sin \phi & 0 & 0 \\
 0 & m & -2m_w d \cos \phi & 0 & 0 \\
2m_w d \sin \phi & -2m_w d \cos \phi & I & 0 & 0 \\
0 & 0 & 0 & I_w & 0 \\
0 & 0 & 0 & 0 & I_w
\end{bmatrix}
\]

(16)

\[
V(q, \dot{q}) = \begin{bmatrix}
0 & 0 & 2m_w d \dot{\phi} \cos \phi & 0 & 0 \\
0 & 0 & 2m_w d \dot{\phi} \sin \phi & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

(17)

and $m = m_b + 2m_w$, $I = I_b + 2I_m + 2m_w d^2 + 2m_w b^2$.

Next, the Equation 9 can be rearranged as

\[ \dot{q} = S(q) \beta \]  

(18)

It can be verified that the transformation matrix $S$ is in the null space of the constraint matrix $\Lambda$. Then we have

\[ S^T(q) \Lambda^T(q) = 0 \]  

(19)

Next, taking the time derivative of (18) gives

\[ \ddot{q} = S(q) \ddot{\beta} + \dot{S}(q) \dot{\beta} \]  

(20)

Substituting (18) and (20) into (15) and multiplying both sides by $S^T$ leads to

\[ \tilde{M}(q) \ddot{\beta} + \tilde{V}(q, \dot{q}) \dot{\beta} = \tilde{B}(q) \tau \]  

(21)

Using the relationship between $v = [\dot{v}, \omega]^T$ and $\beta = [\dot{\theta}_r, \dot{\theta}_l]^T$, (21) can be rewritten as

\[ \tilde{M}(q) \ddot{v} + \tilde{V}(q, \dot{q}) v = \tilde{B}(q) \tau \]  

(22)
where

\[
\begin{align*}
\hat{M}(q) &= M(q)T \\
\hat{V}(q, \dot{q}) &= V(q, \dot{q})T \\
\hat{B}(q) &= B(q)
\end{align*}
\] (23)

\[
\begin{align*}
\hat{S} &= \begin{bmatrix}
\frac{1}{r} & b \\
-\frac{1}{r} & -b \\
\end{bmatrix}
\end{align*}
\] (26)

Considering the generalized coordinate vector as \( \dot{x} = [x, y, \phi, v, \omega]^T \), the dynamic equation of the system can be rearranged as the form of first-order differential equation:

\[
\dot{x} = \begin{bmatrix}
\hat{S}v \\
-\hat{M}(q)^{-1}\hat{V}(q, \dot{q})v
\end{bmatrix} + \begin{bmatrix}
0 \\
\hat{M}(q)^{-1}\hat{B}
\end{bmatrix} \tau
\] (27)

where

\[
\hat{S}(q) = \begin{bmatrix}
\cos \phi & -d \sin \phi \\
\sin \phi & d \cos \phi \\
0 & 1
\end{bmatrix}
\] (28)

3. Optimal control
In this section, the optimal motion planning of the DDMR in the presence of multiple obstacles is formulated as an optimal control problem. Optimal control is to determine the inputs of a dynamical system that optimize (i.e., minimize or maximize) a specified performance index while satisfying some constraints on the motion of the system [9]. The cost function for an optimal control problem can be laid out as:

\[
J = \phi(t_0, x_0, t_f, x_f) + \int_{t_0}^{t_f} w(t, x, u)dt
\] (29)

In general, the cost function includes two terms: a boundary objective \( \phi(\cdot) \) and a integral objective \( w(\cdot) \) [11]. The decision variables in optimization problem is the variables that minimize the cost function (29). For the DDMR motion planning problem, the initial and final time \( (t_0, t_f) \), the state and control trajectories \( (x(t), u(t)) \) are considered as the decision variables. Thus, the objective is to minimize the cost function (29)

\[
\min_{t_0, t_f, x, u} J
\] (30)

As mentioned before, the DDMR are subjected to nonholonomic and dynamic constraints which should be considered by the optimization problem. It can be formulated as the first-order differential equation as

\[
\dot{x} = f(t, x, u)
\] (31)

In addition, the path constraints usually arises in the motion planning problem. The path constraints in this paper is formulated to avoid the obstacles in the workspace. The general form of the path constraint can be described as

\[
h(t, x, u) \leq 0
\] (32)

For the DDMR motion planning problem, the boundary constraint means that the robot has to satisfy the requirements (i.e., position and velocity) of start and final configurations. It can be formulated as inequality constraints

\[
g(t_0, x_0, t_f, x_f) \leq 0
\] (33)
Mechanical systems are usually accompanied by physical limits. The DDMR might have limits on the workplace and the motors have limits on torque and speed. These limits also can be formulated as inequality constraints

\[
\begin{align*}
    x_{\text{low}} & \leq x \leq x_{\text{upp}} \\
    u_{\text{low}} & \leq u \leq u_{\text{upp}}
\end{align*}
\] (34)

Because of the nonlinearity of system dynamics, it is formidable to solve the optimal solution analytically. Therefore, numerical methods are the realistic approaches for solving optimal control problems. Dynamic programming is an excellent approach for the optimal control problem of unconstrained low-dimensional systems, but it does not scale well for high-dimensional systems, since it requires a discretization of the full state space. Direct methods yield a single trajectory through state and control space rather than a global policy like dynamic programming. In this paper, the trapezoidal collocation method is used by converting a continuous problem into a NLP.

3.1. Trapezoidal collocation

Trapezoidal collocation is a widely used direct method in the field of trajectory optimization. Using trapezoidal quadrature, the continuous function, such as dynamic function and objective function, can be converted into a discrete approximation.

(1) Trapezoidal collocation: objective function

The objective functions are usually continuous and integral expressions in trajectory optimization. The continuous integral \( \int w(\cdot) dt \) can be approximated to a summation \( \sum c_k w_k \) by trapezoidal collocation. The summation only depends on the value of the integrand \( w(t_k) = w_k \) at the discrete-time nodes \( t_k \) along the trajectory. With the trapezoid rule, the integration between collocation points can be denoted as

\[
\int_{t_0}^{t_f} w(t, x, u) dt \approx \sum_{k=1}^{N-1} \frac{1}{2} h_k (w_k + w_{k+1})
\] (35)

where \( h_k = t_{k+1} - t_k \).

(2) Trapezoidal collocation: dynamic function

Using direct collocation method, the system dynamics can be converted to a set of collocation constraints, which have an impact on the trajectory optimization problem. By integrating both sides of the dynamic constraints (31) and approximating the integral using trapezoidal quadrature [12], the first-order differential constraints can be converted as equality constraints

\[
\begin{align*}
    \dot{x} &= f \\
    \int_{t_k}^{t_{k+1}} \dot{x} dt &= \int_{t_k}^{t_{k+1}} f dt \\
    x_{k+1} - x_k &\approx \frac{1}{2} h_k (f_k + f_{k+1})
\end{align*}
\] (36)

The collocation constraints are made in the collocation points:

\[
\begin{align*}
    x_{k+1} - x_k &\approx \frac{1}{2} h_k (f_k + f_{k+1})
\end{align*}
\] (37)

where \( k \in 0, \ldots, (N - 1) \).
3. Trapezoidal collocation: constraint functions

The continuous constraints in (32-34) are discretized into a set of constraints in the collocation points.

\[ h(t, x, u) \leq 0 \rightarrow h(t_k, x_k, u_k) \leq 0 \]
\[ g(t_0, x_0, t_f, x_f) \leq 0 \rightarrow g(t_0, x_0, t_k, x_k) \leq 0 \]
\[ x_{low} \leq x \leq x_{upp} \rightarrow x_{low} \leq x_k \leq x_{upp} \]
\[ u_{low} \leq u \leq u_{upp} \rightarrow u_{low} \leq u_k \leq u_{upp} \]  

(38)

(4) Trapezoidal collocation: interpolation

To obtain a smooth trajectory, the interpolation on collocation points should be applied. Normally the control input between in a step can be considered as a constant, so it can be formulated as linear splines. Based on the value of control input on the collocation points (knot points), the value \( u \) on the interval \( t \in [t_k, t_{k+1}] \) can be represented as

\[ u(t) \approx u_k + \frac{\tau}{h_k}(u_{k+1} - u_k) \]  

(39)

where \( \tau = t - t_k \).

The same procedure can be easily adapted to obtain the expression for \( f \) on the interval \( t \in [t_k, t_{k+1}] \).

\[ f(t) = \dot{x}(t) \approx \dot{x}_k + \frac{\tau}{h_k}(\dot{x}_{k+1} - \dot{x}_k) \]  

(40)

By integrating both sides of equation (40), the state \( x(t) \) is obtained as

\[ x(t) = \int \dot{x}(t) dt \approx c + f_k \tau + \frac{\tau^2}{2h_k}(f_{k+1} - f_k) \]  

(41)

By substituting the value of the state at the boundary \( \tau = 0 \) into equation (41), the constant for integration \( c \) can be solved. Thus, the final expression for the state is denoted as

\[ x(t) \approx x_k + f_k \tau + \frac{\tau^2}{2h_k}(f_{k+1} - f_k) \]  

(42)

3.2. Obstacle avoidance and objective function

There are two types of path constraints in trajectory optimization problem: inequality constraint and equality constraint. For the obstacle avoidance problem, the path constraints are generally constructed as inequality. To simplify the math, the obstacle in this paper is represented as a circle. The structure of the obstacle avoidance problem is clarified in Figure 2.

To be specific, the \( i \)th obstacle is coordinated in the position \( (x_{obi}, y_{obi}) \) with radius \( r_i \) which can get a inequality constraint

\[ r_{obi}^2 - ((x - x_{obi})^2 + (y - y_{obi})^2) \leq 0 \]  

(43)

where the radius \( r_i \) can be expressed as

\[ r_{obi} = r_i + r_m \]  

(44)

and the \( r_m \) is the radius of the enclosed circle of the mobile robot.

Using Equation (44), the mobile robot can be treated as a point robot in the procedure of solving trajectory optimization problem.
The key idea behind the trajectory optimization is to minimize the objective function, also known as cost function. In this study, for the optimal motion planning of the mobile robot, the cost function is to minimize the actuating inputs. In order to keep the mobile robot from these obstacles as far as possible, a quadratic form of distance potential function, as a supplement of path constraints, is applied to the cost function. Therefore, using equation (29), the objective function can be rewritten as

\[
J = \int_{t_0}^{t_f} (\|u\|^2 + \sum_{i=1}^{N_{ob}} \|P_i^2\|)dt
\]  

(45)

where \(\|u\|^2\) is the generalized squared norm of the control vector, and \(\|P_i^2\|\) is the supplement to the collision avoidance between the mobile robot and the \(i^{th}\) obstacle. As shown in Figure 2, the distance between the \(i^{th}\) obstacle and the center point of mobile robot \((x, y)\) is obtained by

\[
d_i = \sqrt{(x - x_{obi})^2 + (y - y_{obi})^2}
\]  

(46)

Next, the quadratic potential function between the mobile robot and the \(i^{th}\) obstacle can be expressed as

\[
\|P_i\|^2 = \frac{1}{(d_i - r_{obi})^2}
\]  

(47)

4. Simulation and results

In the previous section, the optimal motion planning problem in presence of multiple obstacles is converted into a general constrained optimization form, i.e., NLP. Once in this form, the problem can be solved by a commercial solver, such as SNOPT, IPOPT, or FMINCON. In this paper, the FMINCON solver based on MATLAB is used to solve the NLP which is constructed in previous. FMINCON requires the objective function and constraint function to be separate functions, which results in twice work of evaluating the dynamics at every grid-point. To improve computing efficiency, it is necessary to create a single function that computes the integration of the dynamics, and store the last input states and output solution.

The effectiveness of the proposed method is demonstrated through simulation studies in MATLAB R2016b on a personal computer equipped with an Intel Core i7 processor (2.2GHz CPU and 16GB RAM) in the environment of macOS. The relevant physical parameters of mobile and design parameters are shown in Table 2.

| Table 2. The value of parameters. |
|-----------------------------------|
| Parameter | Value | Unit  |
|---------- |------- |-------|
| \(m\)    | 11 kg  |        |
| \(m_b\)  | 10 kg  |        |
| \(m_w\)  | 0.5 kg |        |
| \(I\)    | 0.4136 \(kgm^2\) | |
| \(I_m\)  | 0.001 \(kgm^2\) | |
| \(I_w\)  | 0.001 \(kgm^2\) | |
| \(I_b\)  | 0.4 \(kgm^2\)  |     |
| \(b\)    | 0.1 m  |       |
| \(d\)    | 0.04 m |       |
| \(r\)    | 0.025 m|       |
| \(r_m\)  | 0.4 m  |       |
The state limits, control limits and boundary constraints used in the simulations are presented as

\[
\begin{align*}
\mathbf{x}_{\text{upp}} &= \begin{bmatrix} x_{\text{upp}} \\ y_{\text{upp}} \\ \phi_{\text{upp}} \\ v_{\text{upp}} \\ \omega_{\text{upp}} \end{bmatrix} = \begin{bmatrix} 6 \\ 6 \\ 2\pi \\ 6 \\ 10 \end{bmatrix}, \\
\mathbf{x}_{\text{low}} &= \begin{bmatrix} x_{\text{low}} \\ y_{\text{low}} \\ \phi_{\text{low}} \\ v_{\text{low}} \\ \omega_{\text{low}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -2\pi \\ 0 \\ -10 \end{bmatrix} \\
\mathbf{u}_{\text{upp}} &= \begin{bmatrix} u_{1\text{upp}} \\ u_{2\text{upp}} \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}, \\
\mathbf{u}_{\text{low}} &= \begin{bmatrix} u_{1\text{low}} \\ u_{2\text{low}} \end{bmatrix} = \begin{bmatrix} -3 \\ -3 \end{bmatrix}
\end{align*}
\]  

(48)

For most trajectory optimization methods, a good initial guess is necessary. The initializations for trajectory optimization which consider the problem-specific knowledge are beneficial for the solver to rapidly arrive at the globally optimal solution. In this way, the initial guess of state of the simulations in this section is a straight line between initial and final states, and the initial guess of time is \( t_{\text{guess}} = \|\mathbf{p}_0 - \mathbf{p}_f \| \) based on the guess that the velocity of mobile robot is 1 m/s. Also, we use a uniform time grid with \( N = 15 \) points. To be on the safe side, the previous strategy can plus a random number to obtain several different initialization strategies that are used to try and check if they all converge to the same solution.

In this case, the mobile robot is considered to move from the initial state \( \mathbf{x}_0 = [1.5, 1.5, \pi/4, 0, 0]^T \) to the final state \( \mathbf{x}_f = [5.5, 5.5, 0, 0, 0]^T \) within 100 seconds. The solution took 4.9 seconds to compute and 156 iterations. As shown in Figure 3, there are two circle obstacles in the test environment: the first one locates at point \( \mathbf{P}_1 = [2.5, 5] \) with radius \( r_1 = 0.1 \) and the other one locates at point \( \mathbf{P}_2 = [4, 4] \) with radius \( r_2 = 0.1 \). Using the Equation (44), the mobile robot can be considered as a point robot. In this way, the inflated radius of the two circle obstacles are \( r_{\text{ob1}} = 0.5 \) and \( r_{\text{ob2}} = 0.5 \), respectively.

As it can be seen in Figure 3, the solution of the point-to-point motion planning is a collision free optimal path. To be specific, the path has a short length and stays away from the obstacles simultaneously, which means that the objective function (45) takes effect.

The optimal coordinates \((x, y)\) and orientation of each grid point are depicted in Figure 4. Meanwhile, the number of grid points is then extended from 15 to 150 with the quadratic interpolation method (42). It also can be seen that the time duration of the trajectory is 3.13 seconds. The optimal linear and angular velocity of mobile robot are shown in Figure 5. In addition, the optimal torques exerted to the actuators of right wheel \((u_1)\) and left wheel \((u_2)\) are demonstrated in Figure 6. Furthermore, as shown in Figure 4, 5 and 6, the greatest values of state \( \mathbf{x}_{\text{max}} = [5.5, 5.508, 1.14, 4.097, 0.407]^T \) and torque \( \mathbf{u}_{\text{max}} = [0.573, 0.702]^T \), and the minimum value of state \( \mathbf{x}_{\text{min}} = [1.5, 1.5, -0.039, 0, -1.702]^T \) and torques \( \mathbf{u}_{\text{min}} = [-0.748, -0.876]^T \) do not exceed the boundary of \( \mathbf{x}_{\text{upp}}, \mathbf{u}_{\text{upp}}, \mathbf{x}_{\text{low}} \) and \( \mathbf{u}_{\text{low}} \) defined in (48) and (49). Figure 7 and 8 illustrate the integral of absolute error in collocation constraint along each segment of the trajectory, and the maximum values of errors are \([0.015, 0.021, 0.016, 0.001, 0.061]\).

5. Conclusions

In this paper, optimal motion planning of the nonholonomic mobile robot based on the transcription method and non-linear programming has been presented. The mobile robot suffers from not only the nonlinear states and control constraints, but also multiply obstacles restriction. Then the motion planning problem is discretized by the trapezoidal collocation method and converted into nonlinear programming problem. The proposed objective function consists of two part: force squared function and potential function. By minimizing this objective, the generated trajectory can keep the mobile robot away from the obstacles and make the length of path as short as possible. Finally, the capability of the presented method has been demonstrated by simulation experiments.
5.1. Acknowledgments
This research received the financial support from the Fundamental Research Funds for Central Universities(2682017ZDPY09).
References

[1] LaValle S M 2006 Planning algorithms (United Kingdom: Cambridge university press)
[2] Bang-Jensen J and Gutin G Z 2008 Digraphs: theory, algorithms and applications (Germany: Springer Science & Business Media)
[3] Nearchou and C A 1998 Robotica 16 575
[4] Takahashi O and Schilling R J 1989 IEEE Transactions on robotics and automation 5 143
[5] Cai C and Ferrari S 2009 IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 672
[6] Janson L, Schmerling E, Clark A and Pavone M 2015 The International journal of robotics research 34 883
[7] Goerzen C, Kong Z and Mettler B 2010 Journal of Intelligent and Robotic Systems 57 65
[8] Korayem M H, Nazemizadeh M and Nohooji H R 2014 Journal of the Brazilian Society of Mechanical Sciences and Engineering 36 221
[9] Rao A V 2009 Advances in the Astronautical Sciences 135 497
[10] Betts J T 1998 Journal of guidance, control, and dynamics 21 193
[11] Kelly M 2017 SIAM Review 59 849
[12] Li B and Shao Z 2015 Advances in Engineering Software 87 30