Received May 16, 2019, accepted June 16, 2019, date of publication July 2, 2019, date of current version July 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2925428

Dynamical Obstacle Avoidance of Task-Constrained Mobile Manipulation Using Model Predictive Control

WEI LI[1] **AND RONG XIONG**[2]
[1]School of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China
[2]State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China

Corresponding author: Rong Xiong (rxiong@zju.edu.cn)

This work was supported in part by the National Science Foundation of China under Grant U1609210, and in part by the Science and Technology Project of Zhejiang Province under Grant 2019C01043.

ABSTRACT Task-constrained motion planning (TCMP) is involved in many practical applications, such as opening the door, opening the drawer, twisting the screw, and so on. Because of the trend of man–machine collaboration and the existence of dynamic in environments, planning the collision-free motions for task-constrained manipulation is a significative problem. This paper explores the TCMP issue for mobile manipulation, which uses a mobile base to enhance the working range and flexibility but simultaneously makes the problem harder than that of single manipulation because of the additional degrees of freedom (dofs). We propose an optimization-based method to plan the obstacle avoidance motion in real-time. First, the global robotic Jacobian matrix that combines the omnidirectional base and the robotic arm is derived. Second, model predictive control is used to plan the control rule in order to maximize the closest distance between the obstacles and the mobile manipulator and minimize the velocities of null space at the same time. We have deduced the model with four differential equations that represent the law of the distance over time. Third, the distances are calculated and sent to the model to calculate the velocity of each joint of the arm and the base using ACADO that is an open-source toolkit. Using the velocities, the mobile manipulator can move away from the approaching people while still fixing the end of the arm to manipulate tool at the same time. Our method is verified on the mobile manipulator that consists of four Mecanum wheels, a base, a UR10 arm, and four kinect cameras in Gazebo simulation with using the ROS operation.

INDEX TERMS Mobile manipulation, motion planning, task constraints, dynamical obstacle avoidance, model predictive control.

I. INTRODUCTION

Task-constrained motion planning (can be called TCMP for short) is a classical problem of robot motion planning which requires manipulators to move their end effectors along pre-defined paths or fix their end effectors. Among them, end fixed constrained motion planning is of great significance because of the space limit of many manipulation tasks. It can be used as twisting the screw, spot welding, drilling holes and so on. The challenge of TCMP is to plan the joint-space paths of manipulators that can avoid the moving and static obstacles and satisfy task constraints at the same time.

The associate editor coordinating the review of this manuscript and approving it for publication was Hui Xie.

Dynamical obstacle avoidance of TCMP is very important in industrial and living application because the robot will not stop working even if people comes up and will avoid him at the same time. It improves man-machine collaboration performance vastly. Similar to the common motion planning, the methods of TCMP can be divided as the following categories: sample-based methods, artificial potential field based methods and optimization-based methods. Because of the exist of constraints, the configuration space of manipulator would form a irregular manifold. So search-based methods (such as A*, Dijkstra and so on) cannot be used in task-constrained manipulation.

Sample-based methods are widely used and matured methods which require the circumstance is known and each

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/

moving obstacle has a predictable trajectory. They utilize the random sampled nodes to build a growing graph. When the target node is added to the graph, the process would be end. Then there should be a path that can connect the initial and target node in this graph. The Constrained Bi-directional Rapidly-exploring Random Tree (CBIRRT) was presented by [1] which can plan paths in manifold of configuration space. After extending a new RRT random tree node in the high dimensional configuration space, CBIRRT uses the pseudo inverse of the Jacobian matrix to do the projection of the node in the manifold space, and add this node to the constrained RRT tree. Repeat this step until the target location is added to the RRT tree so that a path in manifold has been found. The Tangent Bundle RRT algorithm was presented by [2]. This algorithm set the initial node and the target node on the manifold, then construct the tangent spaces of them. Sample a new node on one tangent space and find the nearest neighbor node on the other tangent space. If the distance between the node and the manifold beyond a threshold, then project it to the manifold to create a new tangent space. Repeat these steps then a path which lies on the manifold connect the initial node and the target node would be created. Reference [3] presented the first-order retraction (FR) which gives a rapid project method to sample the node on the manifold. Reference [4] presented an algorithm which based on a struct named foliation. With respect to redundant robots, a point in task space would be mapped to many points in configuration space which can form a leaf. So sampling the next point in task space is sampling the next point in a leaf in configuration space. If no path is available, an arrival node in the same leaf of this point in task space would be find and the next point in configuration space would be searched to connected with the arrival node. An Italy team presented and deepened a random algorithm for planning dynamically feasible motions of robots with task-constraints which also based on foliation [5]–[7]. But each leaf contains not only points but also velocity of points in configuration space. As result, the path and the trajectory on the manifold can be planned at the same time. On average, sample-based methods have good precision but not good real-time, and they can not handle the moving obstacle with unpredictable trajectory.

Artificial potential field based methods are classical methods that are used in motion planning for many years [8]. They utilize an attractive potential field to attract the robot to the target and a repulsive potential field to make the robot avoid obstacles. Reference [9] proposed a new potential field form that consists of a static potential field and a dynamic potential field. The dynamic potential field can make the moving obstacles have larger repulsive forces so that the robot can avoid them in a larger velocity. The velocity of avoidance which in direct proportion to the gradient of potential field is used in the null-space movement so that can satisfy the task constraints. References [10], [11] only considered the distance between the closest point on the obstacle and the critical point in redundant robotic arm. In [10], the velocity of null-space movement is setted inversely proportional to this distance. In [11], the velocity of null-space movement is setted in direct proportion to the gradient of this distance. Using the form of electric field in physics is a natural way to construct the artificial potential field [12], [13]. And adding some terms to avoid joint limits and prevent singular configurations of the manipulator would be mature [12]. In general, artificial potential field based methods are simple in structure and convenient for real-time control. So they have been widely used in real-time obstacle avoidance trajectory control. But the disadvantage is that there is a local optimal solution which is prone to deadlock phenomenon, and thus may cause robot stay at the local best before reaching the target.

The basic thought of optimization-based methods is to construct object functions that involve the distances between robot and obstacles, the distance between robot and target and the velocities and accelerates of each joints, then to optimize this function to get the optimal trajectory of robot. The CHOMP algorithm [14] presented a prior term which include the velocity and accelerate of each joint and an obstacle term, but the obstacles should be static. The ITOMP algorithm [15] divides obstacles to dynamic obstacles and static obstacles, and use an error model to simulate the dynamic obstacles' positions. Convex optimal approach had been used to optimize robot trajectory for many years. Reference [16] presented a way using second-order cone program to optimal the time spent in the robot manipulation. Model predictive control utilizes dynamic models to predict their future behavior, which is becoming more and more popular in motion planning because of its capability to handle various kinds of kinematic and dynamic constraints [17], [18]. Reference [19] presented an automatic C-code generation strategy for real-time nonlinear model predictive control (NMPC), which had been implemented within the software package ACADO Toolkit. ACADO Toolkit has a good real-time performance with a sampling time in the microsecond range. This allows for convenient applys of trajectory planning of different scenes [20], [21]. In general, optimization-based methods have clear goals and would get global optimal solutions. Although they have large time consumption and not good real-time performance, but with the advent of more and more toolkits, their real-time performance will be better and better again.

In this paper, the optimization-based method NMPC is adopted. This is firstly because it is a local path planning algorithm which different from RRT, PRM etc. Local trajectory planning algorithms have forward-looking, which makes them can deal with obstacle avoidance situations outside the plan. And this is secondly because it can take care of more optimization factors (such as velocity) than other methods (such as artificial potential field based methods). We take the base and the robotic arm of the mobile manipulator as a whole, and deduce the global Jacobian matrix of the mobile manipulator with respect to the velocities of all joints of the robotic arm and all dofs of the base. Then we use NMPC to plan the executing velocities of all dofs of the mobile manipulator while people comes to the robot. The kinematic model is deduced with 4 extra equations in order to satisfy the form of the object function of optimization. The equations utilize the global Jacobian matrix.

The main contributions of this paper are as follows:

(i) The Jacobian matrix of arbitrary point in the robotic arm and the base is derived. They are substituted into the kinematic model, and obtain the velocity control rule that covers the whole robot including the robotic arm and the omnidirectional base through ACADO toolkit. The omnidirectional base is used because it can avoid obstacles from left and right side.

(ii) Nonlinear model predictive control is used for dynamic obstacle avoidance of task-constrained mobile manipulation. An object function which is designed to maximize the closest distance between obstacles and the robot and simultaneously minimize the velocities of null-space is proposed. And in order to optimize the function, we have deduced the obstacle avoidance kinematics with 4 differential equations which represent the law of the distance over time. The obstacle avoidance kinematics is different from the traditional kinematic model [7] and the dynamic model [22]. It goes over the velocity and directly takes the reciprocal of the distance as the output. The results of experiments have proved the available of the equations.

The remainder of the paper is organized as follows. Section 2 systematically deduced the global Jacobian of the whole robot and the Jacobian of arbitrary point in the robot. Section 3 introduces the model predictive control roughly, and explains how the model is modified to adapt to optimize the object function. Section 4 gives the experimental results of NMPC. As a contrast, we also realize the dynamical obstacle using artificial potential field method [12] and foliation based RRT method [5] and analyze the advantages of NMPC over the last two methods. Finally, Section 5 gives the conclusion.

II. THEORETICAL DERIVATION AND ALGORITHM IMPLEMENTATION

A. DERIVATION OF THE GLOBAL JACOBIAN MATRIX OF ARBITRARY POINT AND ROBOT END IN THE MOBILE MANIPULATOR

In robotic arm, each link can be represented by 4 kinematic parameters $(\alpha_{i-1}, a_{i-1}, d_i, \theta_i)$ which called Denavit-Hartenberg parameters. Figure 1 shows the non-zero D-H parameters of UR10 robotic arm. The D-H parameters can determine the configuration of a robotic arm. The rotation and translation matrix of one link with respect to the forward link can be represented as follows:

$$^{i-1}\boldsymbol{T}_i = \begin{pmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

where s is the mathematic symbol which stands for sin and c is the mathematic symbol which stands for cos.

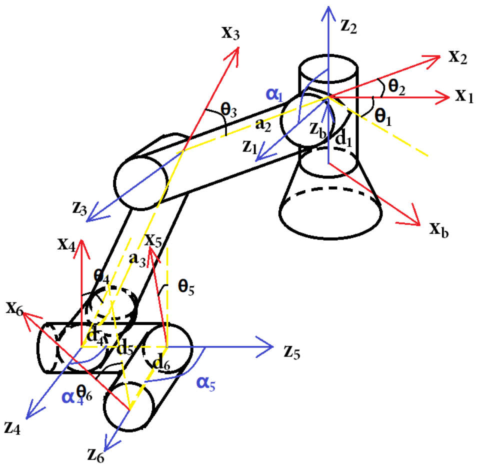

FIGURE 1. The D-H parameters of UR10 robotic arm and the coordinate system of each link.

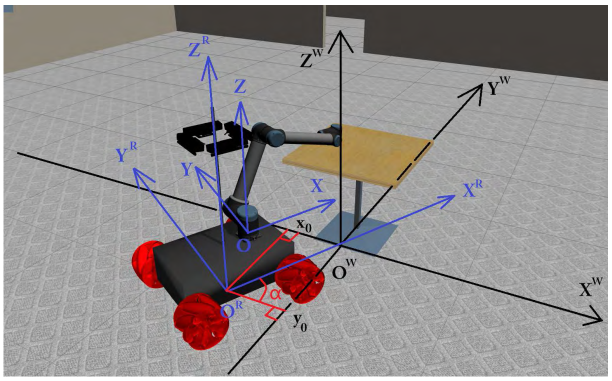

FIGURE 2. The scene of mobile manipulation and the coordinate systems of the mobile manipulator.

As shown in Figure 2, we use (X^W, Y^W, Z^W) to represent the coordinate of the world, use (X^R, Y^R, Z^R) to represent the coordinate of the mobile manipulator, use (X, Y, Z) to represent the coordinate of the robotic arm, and use (X^i, Y^i, Z^i) to represent the coordinate of the i th link of the robotic arm. So in the coordinate of the robotic arm, the rotation and translation matrix of the i-th link can be calculated as follows:

$$\boldsymbol{T}_i = {}^b\boldsymbol{T}_1 \cdot {}^1\boldsymbol{T}_2 \cdot \ldots \cdot {}^{i-1}\boldsymbol{T}_i \tag{2}$$

$$\boldsymbol{T}_i = \begin{pmatrix} \boldsymbol{R}^i & \boldsymbol{t}^i \\ \boldsymbol{0} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_x^i & \boldsymbol{r}_y^i & \boldsymbol{r}_z^i & \boldsymbol{t}^i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

where $\boldsymbol{R}^i$ represents the 3×3 rotation matrix, and $\boldsymbol{t}^i$ represents the 3×1 translation vector. $\boldsymbol{r}_x^i$ represents the projection of the x-axis unit column vector of the i-th link with respect to the coordinate of the robotic arm, and $\boldsymbol{r}_y^i$ and $\boldsymbol{r}_z^i$ respectively represent the y-axis and z-axis unit column vectors with respect to the coordinate of the robotic arm. So the position of an arbitrary point $(x_a^i, y_a^i, z_a^i)^T$ in the i-th link can be represented in the coordinate of the robotic arm as follows:

$$\begin{pmatrix} \boldsymbol{x_a} \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}^i & \boldsymbol{t}^i \\ \boldsymbol{0} & 1 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{x}_a^i \\ 1 \end{pmatrix} \tag{4}$$

where $\boldsymbol{x_a}$ represents the position $(x_a, y_a, z_a)^T$ of the point in the coordinate of the robotic arm, and $\boldsymbol{x_a}$ represents the position $(x_a^i, y_a^i, z_a^i)^T$ of the point in the coordinate of the i-th link.

Within the principle of motion superposition and the differential rule of multielements, the Jacobian matrix of this point with respect to the N joints of the robotic arm can be deduced as follows:

$$\boldsymbol{J_a} = \frac{\partial(x_a, y_a, z_a, \phi_{xa}, \phi_{ya}, \phi_{za})^T}{\partial(\theta_1, \theta_2, \cdots, \theta_N)} \tag{5}$$

$$\boldsymbol{J_a} = \begin{pmatrix} \frac{\partial \boldsymbol{x_a}}{\partial \theta_1} & \cdots & \frac{\partial \boldsymbol{x_a}}{\partial \theta_i} & \boldsymbol{0_{3\times(N-i)}} \\ \boldsymbol{r_z^1} & \cdots & \boldsymbol{r_z^i} & \boldsymbol{0_{3\times(N-i)}} \end{pmatrix} \tag{6}$$

where ϕ_{xa}, ϕ_{ya} and ϕ_{za} are the Eula joints of the i-th link in the coordinate of the robotic arm. So based on the formula above mentioned, the Jacobian matrix of the robotic arm can be written as:

$$\boldsymbol{J_0} = \begin{pmatrix} \frac{\partial \boldsymbol{t^N}}{\partial \theta_1} & \frac{\partial \boldsymbol{t^N}}{\partial \theta_2} & \cdots & \frac{\partial \boldsymbol{t^N}}{\partial \theta_N} \\ \boldsymbol{r_z^1} & \boldsymbol{r_z^2} & \cdots & \boldsymbol{r_z^N} \end{pmatrix} \tag{7}$$

The Jacobian matrixes are all deduced from the base of the robotic arm, but the mobile manipulator has an omnidirectional base which has three additional degrees of freedom: x_0, y_0 and α. So we need to deduce the Jacobian of an arbitrary point in the i-th link from the robot base. The following equations can be derived from the relationship between the position of the point in the robotic arm coordinate system (x_a, y_a, z_a) and the position of the point in the world coordinate system (x_a^w, y_a^w, z_a^w):

$$x_a^w = (x_a + a)\cos(\alpha) - (y_a + b)\sin(\alpha) + x_0 \tag{8}$$

$$y_a^w = (x_a + a)\sin(\alpha) + (y_a + b)\cos(\alpha) + y_0 \tag{9}$$

$$z_a^w = z_a + c \tag{10}$$

Differentiate above three equations with respect to time, the trasformation relationship between the speed of the arbitrary point relative to the world coordinate system and the speed relative to the robotic arm coordinate system will be get as follows:

$$\begin{aligned}\dot{x}_a^w &= \dot{x}_a\cos(\alpha) - (x_a + a)\sin(\alpha)\dot{\alpha} - \dot{y}_a\sin(\alpha) \\ &\quad - (y_a + b)\cos(\alpha)\dot{\alpha} + \dot{x}_0\end{aligned} \tag{11}$$

$$\begin{aligned}\dot{y}_a^w &= \dot{x}_a\sin(\alpha) + (x_a + a)\cos(\alpha)\dot{\alpha} + \dot{y}_a\cos(\alpha) \\ &\quad - (y_a + b)\sin(\alpha)\dot{\alpha} + \dot{y}_0\end{aligned} \tag{12}$$

$$\dot{z}_a^w = \dot{z}_a \tag{13}$$

where the column vector $(a, b, c)^T$ is the translation from the origin of the mobile manipulator coordinate system to the origin of the robotic arm coordinate system. Assume ϕ_{xa}^w, ϕ_{ya}^w and ϕ_{za}^w be the Eula joints of the i-th link in the world coordinate, so the transformation relationship between the angular velocity of the i-th link in the robotic arm coordinate and the angular velocity in the world coordinate system will be calculated as follows:

$$\dot{\phi}_{xa}^w = \dot{\phi}_{xa}\cos(\alpha) - \dot{\phi}_{ya}\sin(\alpha) \tag{14}$$

$$\dot{\phi}_{ya}^w = \dot{\phi}_{xa}\sin(\alpha) + \dot{\phi}_{ya}\cos(\alpha) \tag{15}$$

$$\dot{\phi}_{za}^w = \dot{\phi}_{za} + \dot{\alpha} \tag{16}$$

To sum up, the position and orientation of the arbitrary point and its frame in the i-th link can be represented as:

$$\begin{aligned}\frac{d}{dt}\begin{pmatrix} x_a^w \\ y_a^w \\ z_a^w \\ \phi_{xa}^w \\ \phi_{ya}^w \\ \phi_{za}^w \end{pmatrix} &= \begin{pmatrix} \boldsymbol{R} & \boldsymbol{0_{3\times3}} & \begin{matrix} T_1 & 1 & 0 \\ T_2 & 0 & 1 \\ 0 & 0 & 0 \end{matrix} \\ \boldsymbol{0_{3\times3}} & \boldsymbol{R} & \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{matrix} \end{pmatrix} \cdot \begin{pmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{z}_a \\ \dot{\phi}_{xa} \\ \dot{\phi}_{ya} \\ \dot{\phi}_{za} \\ \dot{\alpha} \\ \dot{x}_0 \\ \dot{y}_0 \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{R} & \boldsymbol{0_{3\times3}} & \begin{matrix} T_1 & 1 & 0 \\ T_2 & 0 & 1 \\ 0 & 0 & 0 \end{matrix} \\ \boldsymbol{0_{3\times3}} & \boldsymbol{R} & \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{matrix} \end{pmatrix} \\ &\quad \cdot \begin{pmatrix} \boldsymbol{J_a} & \boldsymbol{0_{6\times3}} \\ \boldsymbol{0_{3\times N}} & \boldsymbol{I_{3\times3}} \end{pmatrix} \cdot \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_N \\ \dot{\alpha} \\ \dot{x}_0 \\ \dot{y}_0 \end{pmatrix}\end{aligned} \tag{17}$$

$$\boldsymbol{R} = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{18}$$

where $T_1 = -(x_a + a)\sin(\alpha) - (y_a + b)\cos(\alpha)$, $T_2 = (x_a + a)\cos(\alpha) - (y_a + b)\sin(\alpha)$. And let's split the Jacobian matrix $\boldsymbol{J_a}$ to two parts: the translation part $\boldsymbol{J_{at}}$ which comprises the first three rows of $\boldsymbol{J_a}$ and the rotation part $\boldsymbol{J_{ar}}$ which comprises the last three rows of $\boldsymbol{J_a}$. From the equation (17) we can get the form of the global Jacobian matrix of the arbitrary point in the robotic arm:

$$\begin{aligned}\boldsymbol{J_a^w} &= \begin{pmatrix} \boldsymbol{R} & \boldsymbol{0_{3\times3}} & \begin{matrix} T_1 & 1 & 0 \\ T_2 & 0 & 1 \\ 0 & 0 & 0 \end{matrix} \\ \boldsymbol{0_{3\times3}} & \boldsymbol{R} & \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{matrix} \end{pmatrix} \\ &\quad \cdot \begin{pmatrix} \boldsymbol{J_a} & \boldsymbol{0_{6\times3}} \\ \boldsymbol{0_{3\times N}} & \boldsymbol{I_{3\times3}} \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{R\cdot J_{at}} & \begin{matrix} T_1 & 1 & 0 \\ T_2 & 0 & 1 \\ 0 & 0 & 0 \end{matrix} \\ \boldsymbol{R\cdot J_{ar}} & \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{matrix} \end{pmatrix}\end{aligned} \tag{19}$$

And the global Jacobian matrix of the robotic arm end can be written as:

$$J_0^w = \begin{pmatrix} & & T_1 & 1 & 0 \\ & & T_2 & 0 & 1 \\ \boldsymbol{R} & 0_{3\times3} & 0 & 0 & 0 \\ 0_{3\times3} & \boldsymbol{R} & 0 & 0 & 0 \\ & & 0 & 0 & 0 \\ & & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} J_0 & 0_{6\times3} \\ 0_{3\times N} & I_{3\times3} \end{pmatrix} = \begin{pmatrix} & T_1 & 1 & 0 \\ & T_2 & 0 & 1 \\ \boldsymbol{R}\cdot J_{0t} & 0 & 0 & 0 \\ \boldsymbol{R}\cdot J_{0r} & 0 & 0 & 0 \\ & 0 & 0 & 0 \\ & 1 & 0 & 0 \end{pmatrix} \quad (20)$$

If the arbitrary point is on the omnidirectional base, the velocity of which would have nothing to do with the joints of the robotic arm. The following equations can be derived from the relationship between the position of the point in the coordinate system of the mobile manipulator (x_a^r, y_a^r, z_a^r) and the position of the point in the world coordinate system (x_a^w, y_a^w, z_a^w):

$$x_a^w = x_a^r \cos(\alpha) - y_a^r \sin(\alpha) + x_0 \quad (21)$$
$$y_a^w = x_a^r \sin(\alpha) + y_a^r \cos(\alpha) + y_0 \quad (22)$$
$$z_a^w = z_a^r \quad (23)$$

Differentiate above three equations with respect to time:

$$\dot{x}_a^w = (-x_a^r \sin(\alpha) - y_a^r \cos(\alpha))\dot{\alpha} + \dot{x}_0 \quad (24)$$
$$\dot{y}_a^w = (x_a^r \cos(\alpha) - y_a^r \sin(\alpha))\dot{\alpha} + \dot{y}_0 \quad (25)$$
$$\dot{z}_a^w = 0 \quad (26)$$

So the Jacobian matrix of the arbitrary point in the omnidirectional base can be written as follows:

$$J_a^w = \begin{pmatrix} & T_3 & 1 & 0 \\ & T_4 & 0 & 1 \\ 0_{6\times N} & 0 & 0 & 0 \\ & 0 & 0 & 0 \\ & 0 & 0 & 0 \\ & 1 & 0 & 0 \end{pmatrix} \quad (27)$$

where $T_3 = -x_a^r \sin(\alpha) - y_a^r \cos(\alpha)$, $T_4 = x_a^r \cos(\alpha) - y_a^r \sin(\alpha)$.

B. DYNAMICAL OBSTACLE AVOIDANCE WITH USING NONLINEAR MODEL PREDICTIVE CONTROL

As mentioned in [19], the nonlinear model predictive control has the form as follows:

$$\begin{aligned} \min_{\boldsymbol{\xi}(.),\boldsymbol{\zeta}(.)} & \int_0^T (\|\boldsymbol{\xi}(\tau)\|_{\boldsymbol{P}}^2 + \|\boldsymbol{\zeta}(\tau)\|_{\boldsymbol{Q}}^2) d\tau \\ s.t.\ & \dot{\boldsymbol{\xi}}(t) = \boldsymbol{f}(\boldsymbol{\xi}(t), \boldsymbol{\zeta}(t)) \\ & \boldsymbol{\xi}(0) = \boldsymbol{\xi}_0 \\ & \underline{z} \le \boldsymbol{\zeta}(t) \le \bar{z} \quad for\ all\ t \in [0, T] \end{aligned} \quad (28)$$

where $\boldsymbol{\xi}$ denotes the state, $\boldsymbol{\zeta}$ denotes the control input, and $\underline{z}, \bar{z}$ are the control bounds. $\boldsymbol{P}$ and $\boldsymbol{Q}$ are symmetric matrixes. MPC is an optimization-based method for controlling general systems by predicting their future behavior on the basis of dynamic models. NMPC allows one to handle nonlinear dynamics and constraint explicitly [20]. So this method is very suited for the problem of TCMP. For the dynamical obstacle avoidance, the optimization object function must include the closest distance d_{on} between the mobile manipulator and the moving obstacle. So the form of object function can be designed as follows:

$$\min_{\frac{1}{d_{on}}(.),\boldsymbol{\omega}(.)} \int_0^T (|\frac{1}{d_{on}(\tau)}|^2 + \|\boldsymbol{\omega}(\tau)\|_{\boldsymbol{Q}}^2) d\tau \quad (29)$$

where $\boldsymbol{\omega}$ is the general velocity vector $(\dot{\theta}_1, \dot{\theta}_2, \ldots, \dot{\theta}_N, \dot{\alpha}, \dot{x}_0, \dot{y}_0)^T$ of the Jacobian null-space of the mobile manipulator. If the more far away from the moving obstacle the robot is, the bigger the distance d_{on} is, and the smaller the inverse of d_{on} is. But the $1/d_{on}(\tau)$ is not a natural state item in the model of the mobile manipulator. The model of the mobile manipulator is:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_0^{w\dagger}(\dot{\boldsymbol{y}_d} + \boldsymbol{K}\boldsymbol{e_y}) + (\boldsymbol{I} - \boldsymbol{J}_0^{w\dagger}\boldsymbol{J}_0^w)\boldsymbol{\omega} \quad (30)$$

where $\dot{\boldsymbol{q}}$ is the general velocity vector $(\dot{\theta}_1, \dot{\theta}_2, \ldots, \dot{\theta}_N, \dot{\alpha}, \dot{x}_0, \dot{y}_0)^T$ of the mobile manipulator used for executing. $\boldsymbol{J}_0^{w\dagger}$ is the pseudoinverse matrix of the global Jacobian matrix of the robot end. $\boldsymbol{K}$ is a positive definite gain matrix. $\boldsymbol{e_y}$ is the task error stands for $\boldsymbol{y_d} - \boldsymbol{y}$. If the manipulated object is fixed somewhere (as the issue we handle in this paper), such as twisting the screw, $\dot{\boldsymbol{y}_d}$ and $\boldsymbol{e_y}$ should be zeros. Then the model is simplified as follows:

$$\dot{\boldsymbol{q}} = (\boldsymbol{I} - \boldsymbol{J}_0^{w\dagger}\boldsymbol{J}_0^w)\boldsymbol{\omega} \quad (31)$$

So we need to deduce the model equations to represent the law of the inverse of $d_{on}(\tau)$ over time. If the nearest point in the robot is presented as $(x_n, y_n, z_n)^T$ in the world coordinate, the nearest point in the moving obstacle is presented as $(x_o, y_o, z_o)^T$ in the world coordinate, we know:

$$d_{on} = \sqrt{(x_n - x_o)^2 + (y_n - y_o)^2 + (z_n - z_o)^2} \quad (32)$$

Differential it's reciprocal with respect to time, and suppose the obstacle is instantaneous static :

$$\frac{d}{dt}(\frac{1}{d_{on}}) = -\frac{1}{d_{on}^2} \cdot \begin{pmatrix} \frac{x_n - x_o}{d_{on}} & \frac{y_n - y_o}{d_{on}} & \frac{z_n - z_o}{d_{on}} \end{pmatrix} \cdot \begin{pmatrix} \dot{x}_n \\ \dot{y}_n \\ \dot{z}_n \end{pmatrix} \quad (33)$$

Inspired by the symmetry of equation (33), we use η_1 to represent $1/d_{on}$, use η_2 to represent $(x_n - x_o)/d_{on}$, use η_3 to represent $(y_n - y_o)/d_{on}$, use η_4 to represent $(z_n - z_o)/d_{on}$. Then differential them with respect to time:

$$\dot{\eta}_1 = -\eta_1^2(\eta_2\dot{x}_n + \eta_3\dot{y}_n + \eta_4\dot{z}_n) \quad (34)$$
$$\dot{\eta}_2 = \eta_1\dot{x}_n - \eta_1\eta_2(\eta_2\dot{x}_n + \eta_3\dot{y}_n + \eta_4\dot{z}_n) \quad (35)$$
$$\dot{\eta}_3 = \eta_1\dot{y}_n - \eta_1\eta_3(\eta_2\dot{x}_n + \eta_3\dot{y}_n + \eta_4\dot{z}_n) \quad (36)$$
$$\dot{\eta}_4 = \eta_1\dot{z}_n - \eta_1\eta_4(\eta_2\dot{x}_n + \eta_3\dot{y}_n + \eta_4\dot{z}_n) \quad (37)$$

where $\dot{x}_n$, $\dot{y}_n$ and $\dot{z}_n$ can be deduced by (19) and (30), and suppose the $\dot{\phi}_{xn}$, $\dot{\phi}_{yn}$ and $\dot{\phi}_{zn}$ be the Eula joints of the link which the nearest point in. We have:

$$\begin{pmatrix} \dot{x}_n \\ \dot{y}_n \\ \dot{z}_n \\ \dot{\phi}_{xn} \\ \dot{\phi}_{yn} \\ \dot{\phi}_{zn} \end{pmatrix} = \boldsymbol{J}_n^w \dot{\boldsymbol{q}} = \boldsymbol{J}_n^w [\boldsymbol{J}_0^{w\dagger}(\dot{\boldsymbol{y}_d} + \boldsymbol{K}\boldsymbol{e}_y) + (\boldsymbol{I} - \boldsymbol{J}_0^{w\dagger}\boldsymbol{J}_0^w)\boldsymbol{\omega}] \tag{38}$$

We don't care much about Eula joints, so we can divide the $\boldsymbol{J}_n^w$ to the translation part and the rotation part: a 3 × (N + 3) matrix $\boldsymbol{J}_{tn}^w$ and a 3 × (N + 3) matrix $\boldsymbol{J}_{rn}^w$. So $\dot{x}_n$, $\dot{y}_n$ and $\dot{z}_n$ can be represented as:

$$\begin{pmatrix} \dot{x}_n \\ \dot{y}_n \\ \dot{z}_n \end{pmatrix} = \boldsymbol{J}_{tn}^w [\boldsymbol{J}_0^{w\dagger}(\dot{\boldsymbol{y}_d} + \boldsymbol{K}\boldsymbol{e}_y) + (\boldsymbol{I} - \boldsymbol{J}_0^{w\dagger}\boldsymbol{J}_0^w)\boldsymbol{\omega}] \tag{39}$$

Combine the above equation with equations (34-37), then we get:

$$\begin{pmatrix} \dot{\eta}_1 \\ \dot{\eta}_2 \\ \dot{\eta}_3 \\ \dot{\eta}_4 \end{pmatrix} = \begin{pmatrix} -\eta_1^2\eta_2 & -\eta_1^2\eta_3 & -\eta_1^2\eta_4 \\ \eta_1 - \eta_1\eta_2^2 & -\eta_1\eta_2\eta_3 & -\eta_1\eta_2\eta_4 \\ \eta_1 - \eta_1\eta_2\eta_3 & -\eta_1\eta_3^2 & -\eta_1\eta_3\eta_4 \\ \eta_1 - \eta_1\eta_2\eta_4 & -\eta_1\eta_3\eta_4 & -\eta_1\eta_4^2 \end{pmatrix} \cdot \boldsymbol{J}_{tn}^w \cdot [\boldsymbol{J}_0^{w\dagger}(\dot{\boldsymbol{y}_d} + \boldsymbol{K}\boldsymbol{e}_y) + (\boldsymbol{I} - \boldsymbol{J}_0^{w\dagger}\boldsymbol{J}_0^w)\boldsymbol{\omega}] \tag{40}$$

If the manipulated object is fixed somewhere which we mainly dealed with in this article, the model is simplified as follows:

$$\begin{pmatrix} \dot{\eta}_1 \\ \dot{\eta}_2 \\ \dot{\eta}_3 \\ \dot{\eta}_4 \end{pmatrix} = \begin{pmatrix} -\eta_1^2\eta_2 & -\eta_1^2\eta_3 & -\eta_1^2\eta_4 \\ \eta_1 - \eta_1\eta_2^2 & -\eta_1\eta_2\eta_3 & -\eta_1\eta_2\eta_4 \\ \eta_1 - \eta_1\eta_2\eta_3 & -\eta_1\eta_3^2 & -\eta_1\eta_3\eta_4 \\ \eta_1 - \eta_1\eta_2\eta_4 & -\eta_1\eta_3\eta_4 & -\eta_1\eta_4^2 \end{pmatrix} \cdot \boldsymbol{J}_{tn}^w \cdot (\boldsymbol{I} - \boldsymbol{J}_0^{w\dagger}\boldsymbol{J}_0^w)\boldsymbol{\omega} \tag{41}$$

But $\boldsymbol{J}_{tn}^w$ and $\boldsymbol{J}_0^{w\dagger}$ are functions of configuration vector $\boldsymbol{q}$ which represents $(\theta_1, \theta_2, \ldots, \theta_N, \alpha, x_0, y_0)^T$, that means equation(41) is not complete. A complete model can be built by combining equation (31) with equation(41). And if the manipulated object is on moving, the complete model can be built by combining equation(30) with equation(40). Based on the model, suppose $\boldsymbol{P}$ is the unit matrix, the object function can be written as:

$$\min_{\eta_i(.),\boldsymbol{\omega}(.)} \int_0^T \left(\sum_{i=1}^{4} |\eta_i(\tau)|^2 + \|\boldsymbol{\omega}(\tau)\|_Q^2\right) d\tau \tag{42}$$

and:

$$\eta_2^2 + \eta_3^2 + \eta_4^2 = 1 \tag{43}$$

So the object function (42) is equal to function (29).

As to the point in obstacle, it's of heavy computation to compute the nearest point in the robot with irregular shape in real time. So the wise strategy is using a set of points to represent the robot. Then the distances are calculated between the obstacle's point and these characteristic points of the robot. As shown in Figure 3, the characteristic points are depicted all around the robot uniformly. The nearest point in the obstacle (the back people) is the point B, with the coordinate (x_o, y_o, z_o). Assume the nearest m points in the mobile manipulator are $A_1, A_2, \ldots, A_m$ (m = 3 in our experiment), with the coordinate (x_{ni}, y_{ni}, z_{ni}), (i = 1, . . . , m). We use d_{oni} to represent the distance between the i-th nearest characteristic point and the point in obstacle, use η_{1i} to represent $1/d_{oni}$, use η_{2i} to represent $(x_{ni} - x_o)/d_{oni}$, use η_{3i} to represent $(y_{ni} - y_o)/d_{oni}$, use η_{4i} to represent $(z_{ni} - z_o)/d_{oni}$. Then the obstacle avoidance model of end fixed constrained mobile manipulation which we mainly dealed with in this article can be represented as follows:

$$\begin{pmatrix} \dot{\eta}_{11} \\ \dot{\eta}_{21} \\ \dot{\eta}_{31} \\ \dot{\eta}_{41} \\ \dot{\eta}_{12} \\ \dot{\eta}_{22} \\ \dot{\eta}_{32} \\ \dot{\eta}_{42} \\ \vdots \\ \dot{\eta}_{1m} \\ \dot{\eta}_{2m} \\ \dot{\eta}_{3m} \\ \dot{\eta}_{4m} \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} -\eta_{11}^2\eta_{21} & -\eta_{11}^2\eta_{31} & -\eta_{11}^2\eta_{41} \\ \eta_{11} - \eta_{11}\eta_{21}^2 & -\eta_{11}\eta_{21}\eta_{31} & -\eta_{11}\eta_{21}\eta_{41} \\ \eta_{11} - \eta_{11}\eta_{21}\eta_{31} & -\eta_{11}\eta_{31}^2 & -\eta_{11}\eta_{31}\eta_{41} \\ \eta_{11} - \eta_{11}\eta_{21}\eta_{41} & -\eta_{11}\eta_{31}\eta_{41} & -\eta_{11}\eta_{41}^2 \end{pmatrix} \cdot \boldsymbol{J}_{tn1}^w \\ \begin{pmatrix} -\eta_{12}^2\eta_{22} & -\eta_{12}^2\eta_{32} & -\eta_{12}^2\eta_{42} \\ \eta_{12} - \eta_{12}\eta_{22}^2 & -\eta_{12}\eta_{22}\eta_{32} & -\eta_{12}\eta_{22}\eta_{42} \\ \eta_{12} - \eta_{12}\eta_{22}\eta_{32} & -\eta_{12}\eta_{32}^2 & -\eta_{12}\eta_{32}\eta_{42} \\ \eta_{12} - \eta_{12}\eta_{22}\eta_{42} & -\eta_{12}\eta_{32}\eta_{42} & -\eta_{12}\eta_{42}^2 \end{pmatrix} \cdot \boldsymbol{J}_{tn2}^w \\ \vdots \\ \begin{pmatrix} -\eta_{1m}^2\eta_{2m} & -\eta_{1m}^2\eta_{3m} & -\eta_{1m}^2\eta_{4m} \\ \eta_{1m} - \eta_{1m}\eta_{2m}^2 & -\eta_{1m}\eta_{2m}\eta_{3m} & -\eta_{1m}\eta_{2m}\eta_{4m} \\ \eta_{1m} - \eta_{1m}\eta_{2m}\eta_{3m} & -\eta_{1m}\eta_{3m}^2 & -\eta_{1m}\eta_{3m}\eta_{4m} \\ \eta_{1m} - \eta_{1m}\eta_{2m}\eta_{4m} & -\eta_{1m}\eta_{3m}\eta_{4m} & -\eta_{1m}\eta_{4m}^2 \end{pmatrix} \cdot \boldsymbol{J}_{tnm}^w \end{pmatrix} \cdot (\boldsymbol{I} - \boldsymbol{J}_0^{w\dagger}\boldsymbol{J}_0^w)\boldsymbol{\omega} \tag{44}$$

Based on the model, the object function can be written as:

$$\min_{\eta_{ij}(.),\boldsymbol{\omega}(.)} \int_0^T \left(\sum_{j=1}^{m}\sum_{i=1}^{4} |\eta_{ij}(\tau)|^2 + \|\boldsymbol{\omega}(\tau)\|_Q^2\right) d\tau \tag{45}$$

and:

$$\eta_{2i}^2 + \eta_{3i}^2 + \eta_{4i}^2 = 1, \quad for\ i = 1 \sim m \tag{46}$$

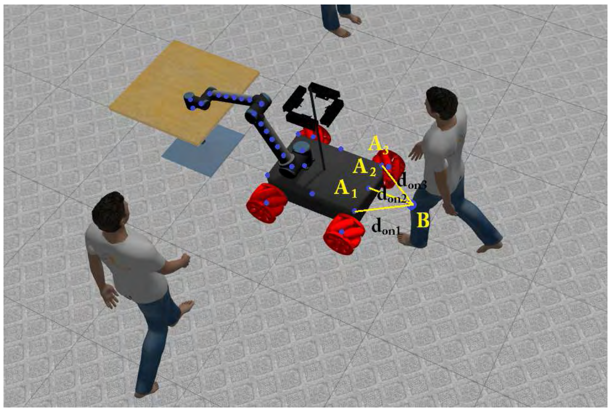

FIGURE 3. The characteristic points in the mobile manipulator.

III. EXPERIMENTAL RESULTS

Our planner was implemented on a 64-bit Intel Core i5-7300HQ CPU running at 2.50 GHz. And the simulation experiments are performed on Gazebo 2.0 and Ros Indigo which installed on Ubuntu 14.04 operating system. The mobile manipulator which used in simulation comprises an UR10 robotic arm, an omnidirectional base and 4 kinect cameras.

In order to verify the performance of the robot's obstacle avoidance algorithm, we design a set of experiments. The first experiment is moving people from back to approach the mobile manipulator, and see the performance of NMPC method. The people is about 0.91 m away from the robot at the beginning, and move toward the robot at a speed of 0.7 m/s. The second experiment is moving people from left to approach the mobile manipulator, and see the performance of NMPC method. The people is about 1.35 m away from the robot at the beginning, and move toward the robot at a speed of 0.7 m/s. For comparison, we perform the same two sets of experiments using the artificial potential field based method (APF) and the foliation based RRT method (FRRT).

Figure 4 shows the NMPC solution how the mobile manipulator avoids the moving people from it's back and maintain it's robotic arm end fixed at the working position. And Figure 5 shows the NMPC solution how the mobile

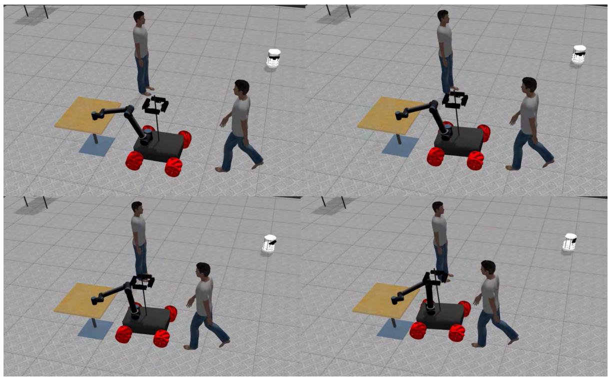

FIGURE 4. The solution how the working mobile manipulator avoids the moving people from it's back using NMPC method.

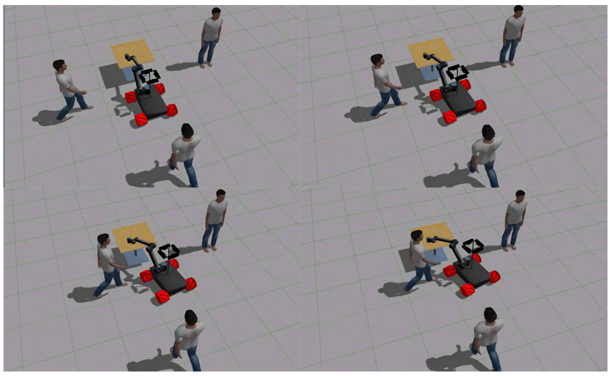

FIGURE 5. The solution how the working mobile manipulator avoids the moving people from it's left using NMPC method.

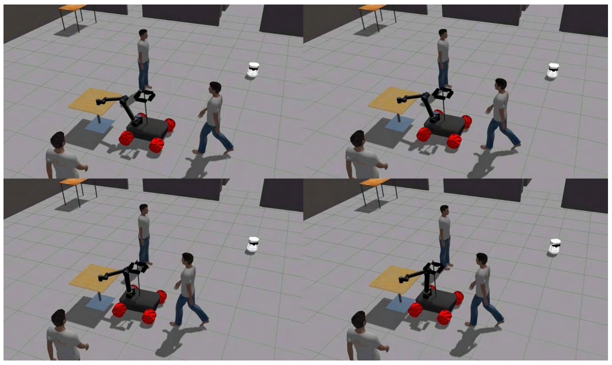

FIGURE 6. The solution how the working mobile manipulator avoids the moving people from it's back using artificial potential field based method.

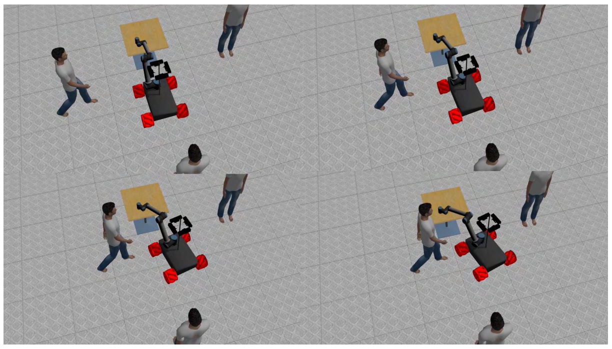

FIGURE 7. The solution how the working mobile manipulator avoids the moving people from it's left using artificial potential field based method.

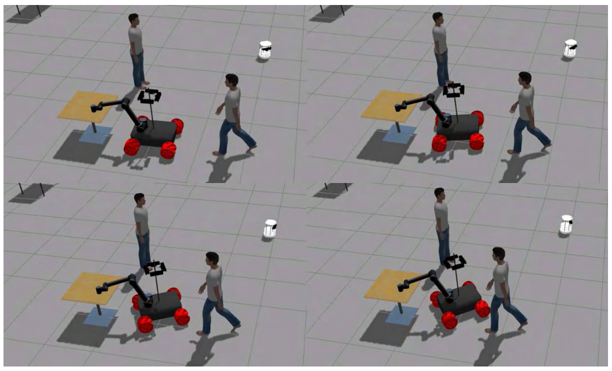

FIGURE 8. The solution how the working mobile manipulator avoids the moving people from it's back using foliation based RRT method.

manipulator avoids the moving people from it's left and maintain it's robotic arm end fixed at the working position too. Both of two experiments demonstrate the effectiveness of our algorithm. For comparison, the solutions of the APF are shown in Figure 6 and Figure 7 and the solutions of the FRRT are shown in Figure 8 and Figure 9.

When people approaches the robot, the minimum distance between them falls down whether using NMPC method, APF method or FRRT method. In the scene that people approaches the robot from back, at the time 0.9 s, the distance of NMPC method is 0.63 m, the distance of APF method is 0.66 m and the distance of FRRT method is 0.59 m. While in the scene that people approaches the robot from left, at the time 1.6 s, the distance of NMPC method is 0.71 m, the distance of APF method is 0.69 m, and the distance of FRRT method is 0.65 m. The different between them is very little. The results can be seen from Figure 10(a, c, e) and 11(a, c, e). In the scene that people approaches the robot from back, at the time 0.9 s, the robotic end maintained fixed, the displacement (δx, δy, $\delta\alpha$) of the robot base computed by NMPC is (0.19, 0.32, −0.05), the displacement computed by APF method is (0.13, 0.38, 0.04), and the displacement computed by FRRT is (0.38, 0.15, 0.57). The displacement of NMPC is very close to APF. But the x displacement of FRRT is bigger than NMPC while the y displacement of FRRT is smaller than NMPC, that because the robot using FRRT avoids the people to the right rather than forward like NMPC and APF. In the scene that people approaches the robot from left, at the time 1.6 s, the robotic end maintained fixed, the displacement (δx, δy, $\delta\alpha$) of the robot base computed by NMPC is (0.62, 0.27, 0.22), the displacement computed by APF method is (0.60, 0.29, 0.26), and the displacement computed by FRRT is (0.53, 0.42, 0.19). The different between them is small. results can be seen from Figure 10(b, d, f) and 11(b, d, f).

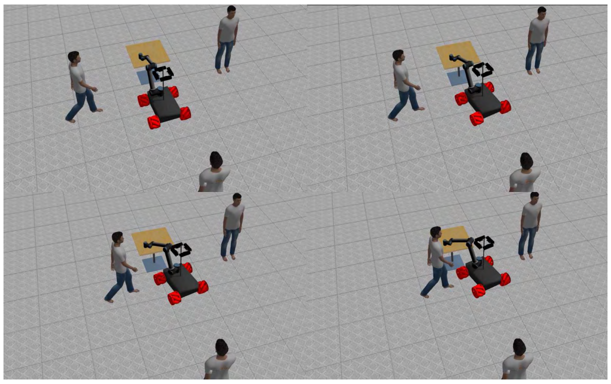

FIGURE 9. The solution how the working mobile manipulator avoids the moving people from it's left using foliation based RRT method.

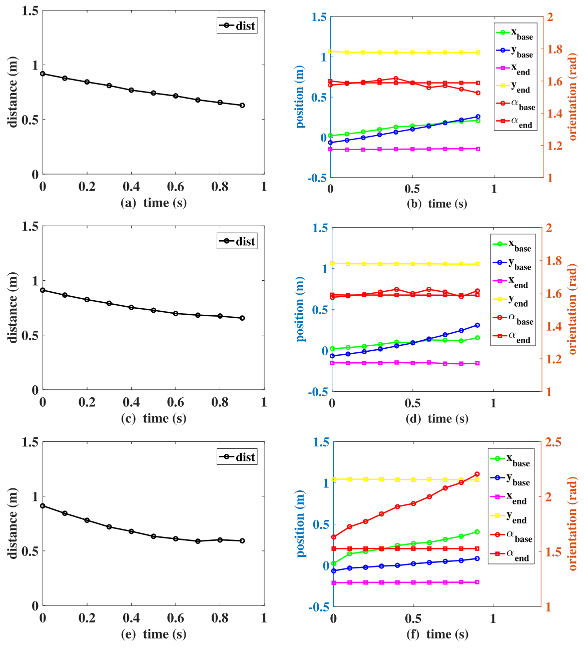

FIGURE 10. People approaches the robot from back. (a) The variation of minimum distance between robot and people with time using NMPC method. (b) The variation of the positions and orientations of robot base and robotic arm end with time using NMPC method. (c) The variation of minimum distance between robot and people with time using artificial potential field based method. (d) The variation of the positions and orientations of robot base and robotic arm end with time using artificial potential field based method. (e) The variation of minimum distance between robot and people with time using foliation based RRT method. (f) The variation of the positions and orientations of robot base and robotic arm end with time using foliation based RRT method.

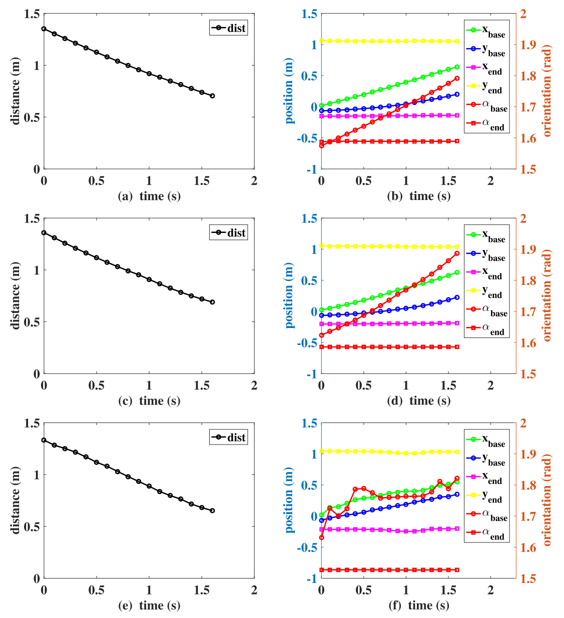

FIGURE 11. People approaches the robot from left. (a) The variation of minimum distance between robot and people with time using NMPC method. (b) The variation of the positions and orientations of robot base and robotic arm end with time using NMPC method. (c) The variation of minimum distance between robot and people with time using artificial potential field based method. (d) The variation of the positions and orientations of robot base and robotic arm end with time using artificial potential field based method. (e) The variation of minimum distance between robot and people with time using foliation based RRT method. (f) The variation of the positions and orientations of robot base and robotic arm end with time using foliation based RRT method.

As shown in Function (29), the NMPC optimized object function includes the modulus of the null-space velocities, so the null-space velocities calculated by NMPC are confined with in a certain range (−0.5 rad/s to 0.5 rad/s, −0.5 m/s to 0.5 m/s). But the null-space velocities of APF method

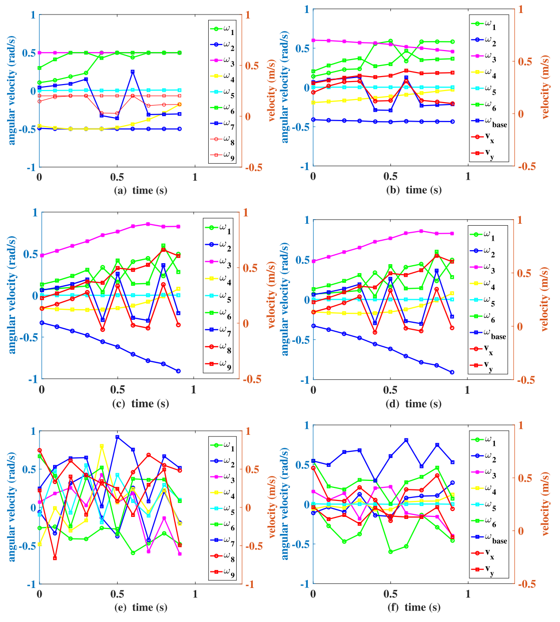

FIGURE 12. People approaches the robot from back. (a) The variation of the velocities and angular velocities of the robot's dofs in null-space with time using NMPC method. (b) The variation of the real executed velocities and angular velocities of the robot's dofs with time using NMPC method. (c) The variation of the velocities and angular velocities of the robot's dofs in null-space with time using artificial potential field based method. (d) The variation of the real executed velocities and angular velocities of the robot's dofs with time using artificial potential field based method. (e) The variation of the velocities and angular velocities of the robot's dofs in null-space with time using foliation based RRT method. (f) The variation of the real executed velocities and angular velocities of the robot's dofs with time using foliation based RRT method.

and FRRT method are not confined, the plan results can be seen from Figure 12(a, c, e) and Figure 14(a, c, e). From the figures we can see that FRRT has the largest volatility of null-space velocity as the curves are very chaos. That due to the random sampling of position in FRRT. The APF has the second large volatility of null-space velocity, and the NMPC has the smallest volatility of null-space velocity. The real-executed velocities are calculated by Equation(31), which means they are projections from the null-space velocities to the constrained manifold. So the range of the null-space velocities would influence the range of the real-executed velocities directly, which can be seen in Figure 12(b, d, f) and Figure 14(b, d, f). Just take the velocities of the base as an example. In the scene that people approaches the robot from back, the x direction velocity of NMPC changes from 0.10 m/s to 0.31 m/s, the x direction velocity of APF changes from −0.06 m/s to 0.35 m/s, the x direction velocity of FRRT changes from 0.09 m/s to 0.59 m/s, the y direction velocity of NMPC changes from 0.29 m/s to 0.46 m/s, the y direction velocity of APF changes from 0.23 m/s to 0.66 m/s, the y direction velocity of FRRT changes from −0.06 m/s to 0.22m/s, the angular velocity of NMPC changes from −0.29 rad/s to 0.13 rad/s, the angular velocity of APF changes from −0.30 rad/s to 0.36 rad/s, the angular velocity of FRRT changes from 0.30 rad/s to 0.80 rad/s. In the scene that people approaches the robot from left, the x direction velocity of

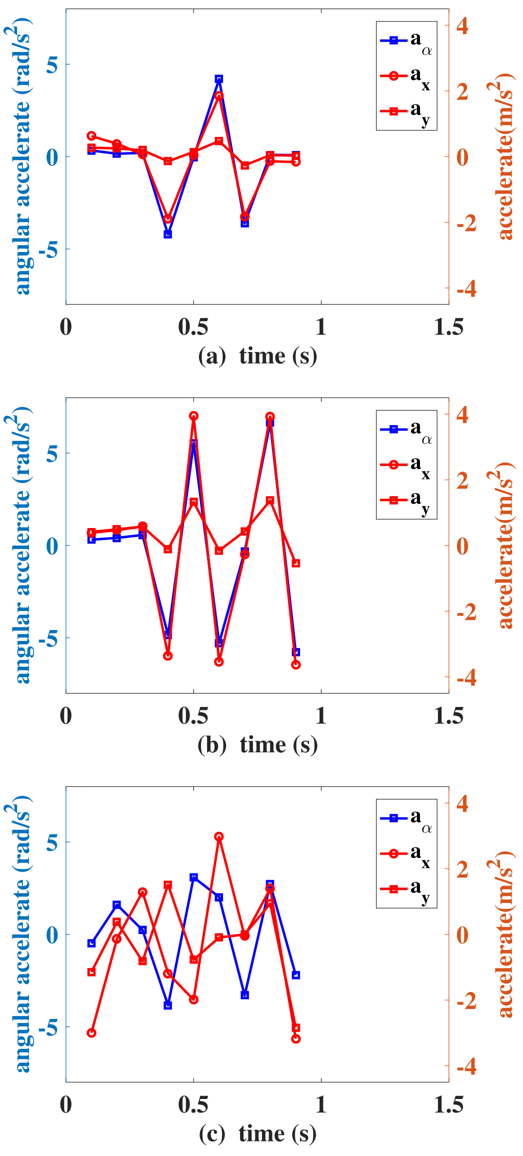

FIGURE 13. People approaches the robot from back. (a) The variation of the real executed accelerates and angular accelerate of the robot base with time using NMPC method. (b) The variation of the real executed accelerates and angular accelerate of the robot base with time using artificial potential field based method. (c) The variation of the real executed accelerates and angular accelerate of the robot base with time using foliation based RRT method.

NMPC changes from 0.34 m/s to 0.43 m/s, the x direction velocity of APF changes from 0.27 m/s to 0.47 m/s, the x direction velocity of FRRT changes from 0.00 m/s to 0.59 m/s, the y direction velocity of NMPC changes from 0.05 m/s to 0.32 m/s, the y direction velocity of APF changes from 0.05 m/s to 0.45 m/s, the y direction velocity of FRRT changes from 0.12 m/s to 0.39 m/s, the angular velocity of NMPC changes from 0.13 rad/s to 0.20 rad/s, the angular velocity of APF changes from 0.11 rad/s to 0.25 rad/s, the angular velocity of FRRT changes from −0.26 rad/s to 0.63 rad/s. So in summary, the FRRT method and the APF method have a larger volatility of real executed velocity than the NMPC method.

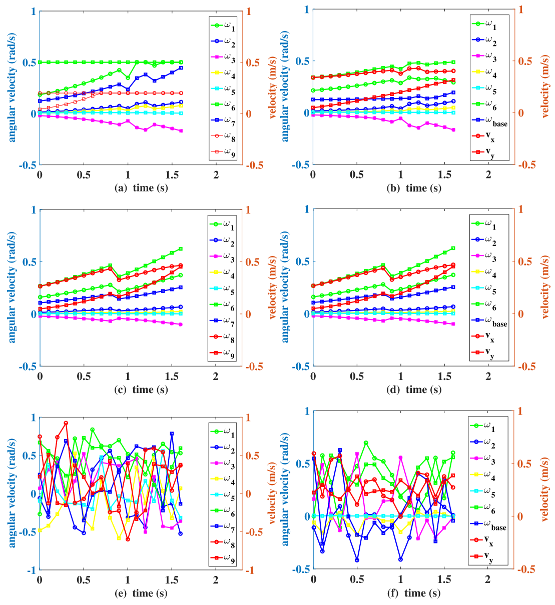

FIGURE 14. People approaches the robot from left. (a) The variation of the velocities and angular velocities of the robot's dofs in null-space with time using NMPC method. (b) The variation of the real executed velocities and angular velocities of the robot's dofs with time using NMPC method. (c) The variation of the velocities and angular velocities of the robot's dofs in null-space with time using artificial potential field based method. (d) The variation of the real executed velocities and angular velocities of the robot's dofs with time using artificial potential field based method. (e) The variation of the velocities and angular velocities of the robot's dofs in null-space with time using foliation based RRT method. (f) The variation of the real executed velocities and angular velocities of the robot's dofs with time using foliation based RRT method.

The volatilities of velocities can be represented as accelerates and angular accelerates, and the larger the accelerates and angular accelerates are, the bigger forces would be given. Figure 13 and 15 show the accelerates and angular accelerates of the robot base while people approaching them from back and left. In Figure 13, in the scene that people approaches the robot from back, the largest absolute value of accelerate a_x of NMPC is 1.90 m/s^2 in 0.4 s, the largest absolute value of accelerate a_x of APF is 3.94 m/s^2 in 0.5 s, and the largest absolute value of accelerate a_x of FRRT is 3.18 m/s^2 in 0.9 s, the largest absolute value of accelerate a_y of NMPC is 0.47 m/s^2 in 0.6 s, the largest absolute value of accelerate a_y of APF is 1.37 m/s^2 in 0.8 s, and the largest absolute value of accelerate a_y of FRRT is 2.83 m/s^2 in 0.9 s, the largest absolute value of angular accelerate a_α of NMPC is 4.22 rad/s^2 in 0.4 s, the largest absolute value of angular accelerate a_α of APF is 6.64 rad/s^2 in 0.8 s, and the largest absolute value of angular accelerate a_α of FRRT is 3.84 rad/s^2 in 0.4 s. In Figure 15, in the scene that people approaches the robot from left, the largest absolute value of accelerate a_x of NMPC is 0.47 m/s^2 in 1.1 s, the largest absolute value of accelerate a_x of APF is 1.07 m/s^2 in 0.9 s, and the largest absolute value of accelerate a_x of FRRT is 4.04 m/s^2 in 0.1 s, the largest absolute value of accelerate a_y of NMPC is 0.26 m/s^2 in 1.3 s, the largest absolute value of accelerate a_y of APF is 0.55 m/s^2 in 1.6 s, and the largest absolute value of accelerate a_y of FRRT is 2.20 m/s^2 in 1.4 s,

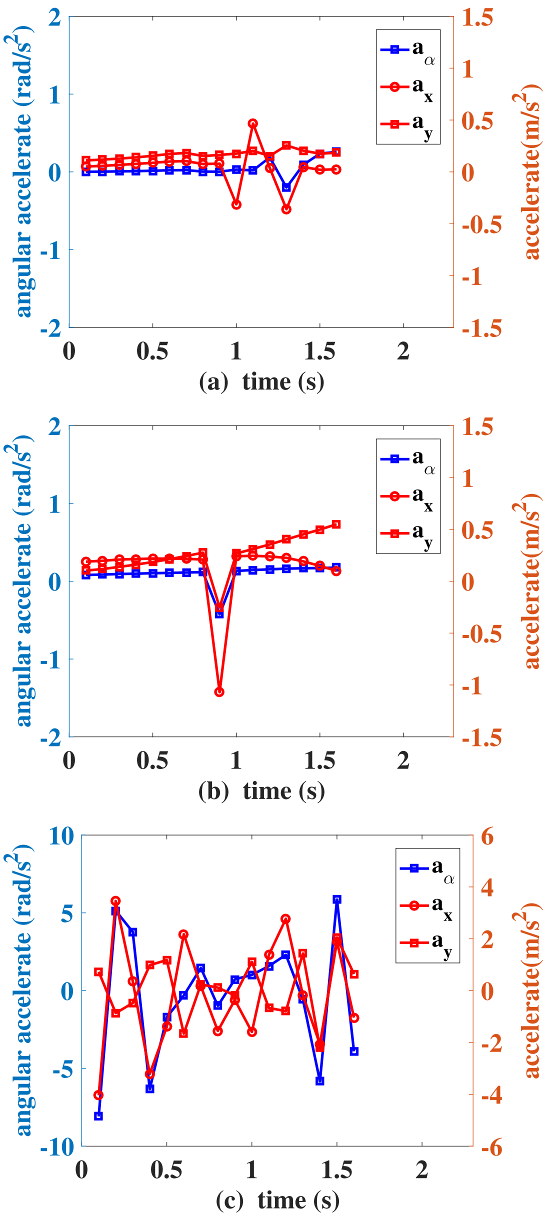

FIGURE 15. People approaches the robot from left. (a) The variation of the real executed accelerates and angular accelerate of the robot base with time using NMPC method. (b) The variation of the real executed accelerates and angular accelerate of the robot base with time using artificial potential field based method. (c) The variation of the real executed accelerates and angular accelerate of the robot base with time using foliation based RRT method.

the largest absolute value of angular accelerate a_α of NMPC is 0.26 rad/s^2 in 1.6 s, the largest absolute value of angular accelerate a_α of APF is 0.43 rad/s^2 in 0.9 s, and the largest absolute value of angular accelerate a_α of FRRT is 8.09 rad/s^2 in 0.1 s. So in summary, the FRRT method and the APF method have a larger accelerate and angular accelerate of robot base than the NMPC method. As the mass of robot base is very large, the difference of forces and torques would be very large too. So choosing NMPC method would reduce the wear and prolong the service life of the robot's electrical machines.

IV. CONCLUSIONS AND FUTURE WORK

This paper proposes a Nonlinear Model Predictive Control method for dynamical obstacle avoidance of task-constrained mobile manipulation. This method constructs a new model which combines the change law of inverse of distance between the robot and the obstacles with the original kinematical equation of the robot. This model can fit the object optimization object function of NMPC, and can be optimized by ACADO tool kit. Using the optimized result, the velocity control rule can drive the robot to avoid moving people while maintain the robot end fixed on the working position at the same time. For comparison, the artificial potential field based method and the roliation based RRT method are used to realize the same tasks. But the results shows that the NMPC method has a better dynamical stability than the artificial potential field based method, because using the NMPC method, the accelerates and angular accelerate of the robot base are smaller than the APF method and the FRRT method. In the scene that people approaches the robot from back, the maximum acceleration of base in the x direction is only 1.90 m/s^2, the maximum acceleration of base in the y direction is only 0.47 m/s^2, the maximum angular acceleration of base is only 4.22 rad/s^2. And in the scene that people approaches the robot from left, the corresponding values are 0.47 m/s^2, 0.26 m/s^2 and 0.26 rad/s^2. Choosing NMPC method would reduce the wear and prolong the service life of the robot's electrical machines. So the NMPC method will be a good method to improve the collaboration performance of mobile manipulator.

The task scene in this paper is end fixed constrained motion planning. In the future, the method can be extended to the scene that the robotic arm end move on an arbitrary specified trajectory. And the NMPC method would be verified in more scenes.

REFERENCES

[1] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds," in *Proc. 26th IEEE Int. Conf. Robot. Autom.*, Kobe, Japan, May 2009, pp. 625–632.

[2] B. Kim, T. T. Um, C. Suh, and F. C. Park, "Tangent bundle RRT: A randomized algorithm for constrained motion planning," *Robotica*, vol. 34, no. 1, pp. 202–225, Jan. 2016.

[3] M. Stilman, "Global manipulation planning in robot joint space with task constraints," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 576–584, Jun. 2010.

[4] J. Kim, I. Ko, and F. C. Park, "Randomized path planning on foliated configuration spaces," in *Proc. 11th Int. Conf. Ubiquitous Robots Ambient Intell.*, Kuala Lumpur, Malaysia, Nov. 2014, pp. 209–214.

[5] M. Cefalo, G. Oriolo, and M. Vendittelli, "Task-constrained motion planning with moving obstacles," in *Proc. 26th Int. Conf. Intell. Robots Syst.*, Tokyo, Japan, Nov. 2013, pp. 5758–5763.

[6] M. Cefalo and G. Oriolo, "Dynamically feasible task-constrained motion planning with moving obstacles," in *Proc. 31th IEEE Int. Conf. Robot. Autom.*, Hong Kong, China, Jun. 2014, pp. 2045–2050.

[7] M. Cefalo and G. Oriolo, "A general framework for task-constrained motion planning with moving obstacles," *Robotica*, vol. 37, pp. 575–598, Mar. 2019.

[8] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proc. 2nd IEEE Int. Conf. Robot. Autom.*, 1985, pp. 500–505.

[9] D. H. Park, H. Hoffmann, P. Pastor, and S. Schaal, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *Proc. 8th IEEE-RAS Int. Conf. Humanoid Robots*, Dec. 2008, pp. 91–98.

[10] H. Shen, H. Wu, B. Chen, Y. Jiang, and C. Yan, "Obstacle avoidance algorithm for 7-DOF redundant anthropomorphic arm," *J. Control Sci. Eng.*, vol. 7, pp. 1–9, Jan. 2015.

[11] M. Cefalo, E. Magrini, and G. Oriolo, "Parallel collision check for sensor based real-time motion planning," in *Proc. 34th IEEE Int. Conf. Robot. Autom.*, Singapore, Jun. 2017, pp. 1936–1943.

[12] N. A. Scott and C. R. Carignan, "A line-based obstacle avoidance technique for dexterous manipulator operations," in *Proc. 25th IEEE Int. Conf. Robot. Automat.*, May 2008, pp. 3353–3358.

[13] A. Winkler and J. Suchý, "Dynamic collision avoidance of industrial cooperating robots using virtual force fields," *IFAC Proc.*, vol.45, no. 22, pp. 265–270, Sep. 2012.

[14] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Siddhartha, "CHOMP: Gradient optimization techniques for efficient motion planning," in *Proc. 26th IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 489–494.

[15] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments," in *Proc. 22nd Int. Conf. Automated Planning Scheduling*, May 2012, pp. 207–215.

[16] D. Verscheure, B. Demeulenaere, J. Swevers, J. D. Schutter, and M. Diehl, "Practical time-optimal trajectory planning for robots: A convex optimization approach," *IEEE Trans. Autom. Control*, to be published.

[17] S. Ide, T. Takubo, K. Ohara, Y. Mae, and T. Arai, "Real-time trajectory planning for mobile manipulator using model predictive control with constraints," in *Proc. 8th Int. Conf. Ubiquitous Robots Ambient Intell.*, Nov. 2011, pp. 244–249.

[18] M. Salaj, M. Gulan, and B. Rohal'-Ilkiv, "Pendubot control scheme based on nonlinear MPC and MHE exploiting parallelization," in *Proc. 19th Int. Conf. Intell. Eng. Syst.*, Sep. 2015, pp. 353–358.

[19] B. Houska, H. J. Ferreau, and M. Diehl, "An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range," *Automatica*, vol. 47, no. 10, pp. 2279–2285, Oct. 2011.

[20] R. Quirynen, M. Vukov, M. Zanon, and M. Diehl, "Autogenerating microsecond solvers for nonlinear MPC: A tutorial using ACADO integrators," *Optim. Control Appl. Methods*, vol. 36, pp. 685–704, Nov. 2014.

[21] M. Cefalo, E. Magrini, and G. Oriolo, "Sensor-based task-constrained motion planning using model predictive control," *IFAC-PapersOnLine*, vol. 51, no. 22, pp. 220–225, 2018.

[22] M. H. Korayem and S. F. Dehkordi, "Dynamic modeling of flexible cooperative mobile manipulator with revolute-prismatic joints for the purpose of moving common object with closed kinematic chain using the recursive Gibbs-Appell formulation," *Mechanism Mach. Theory*, vol. 137, pp. 254–279, Jul. 2019.

WEI LI received the M.S. degree from the University of Science and Technology of China, in 2013. He is currently pursuing the Ph.D. degree with Zhejiang University. His research interests include mobile manipulation and biped robot.

RONG XIONG is currently a Professor with the School of Control Science and Engineering, Zhejiang University. She has independently developed a humanoid robot system that can quickly and continuously play dynamic table tennis, a four-legged robot that can drive and jump, and a small soccer robot that has won international championships. Her research interests include robotic intellisense, cognition, decision and control, machine vision, machine learning, SLAM, and humanoid robot.