Time-Optimal Trajectory Planning Along Predefined Path for Mobile Robots with Velocity and Acceleration Constraints

Mišel Brezak and Ivan Petrović

***Abstract*— This paper is concerned with the problem of finding an optimal velocity profile along the predefined path in order to traverse the path in shortest time. A dynamic model of the mobile robot that accounts for robot actuator constraints is derived. It is shown how to use this dynamic model to express acceleration limits and velocity limit curve required by the optimal time-scaling algorithm. The developed trajectory planning algorithm is demonstrated on the Pioneer 3DX mobile robot. A method is very flexible and can be extended to account for other constraints, such as limited grip between the robot wheels and the ground.**

I. Introduction

The field of mobile robotics is an active research area with promising new application domains in industrial as well as in service sectors. Mobile robots are especially appropriate in applications where flexible motion planning is required. There are many robot navigation strategies, and in cases where operating environment map is known, trajectory planning is commonly used. The term trajectory denotes the path that robot should traverse as a function of time. Trajectory planner generates the appropriate trajectory with goal of arriving at a particular location, patrolling trough specified area etc, and at the same time avoiding collisions with different kinds of obstacles. To obtain feasible trajectory, that is, the trajectory that is robot actually able to track, the planner also needs to consider various robot physical and dynamic limitations such as its velocity and acceleration limits. A trajectory can be generated in real-time on the basis of current sensor readings or generated in advance on the basis of operating environment map. A priori trajectory planning, unlike reactive approaches, results in deterministic and predictable motion, which is desired for automated guided vehicles (AGVs), fork lifts, intelligent transportation systems and many other robotic applications.

Trajectory planning problem can be solved using two approaches: (1) the direct approach, where the search is performed directly in the system's state space or (2) the decoupled approach, where first a path in the configuration space is found and then a velocity profile subject to the actuator constraints is computed. In this paper the decoupled approach is used, mainly due to its computational efficiency. Decoupled approach can also be referred as trajectory planning problem constrained to a precomputed path.

This research was supported by the Ministry of Science, Education and Sports of the Republic of Croatia (grant No. 036-0363078-3018).

Mišel Brezak and Ivan Petrović are with University of Zagreb, Faculty of Electrical Engineering and Computing, 10000 Zagreb, Croatia {misel.brezak, ivan.petrovic}@fer.hr

An algorithm that solves problem of moving a manipulator in minimum time along a specified geometric path subject to input torque/force constraints is described in [1], [2]. The algorithm is primarily used for offline trajectory planning for static manipulators. However, in this paper we show that on modern hardware it is also possible to use it for online motion planning of a mobile robot. In [3] a simple velocity planning for autonomous fork lift trucks is used, however it is not capable of accounting for more complex actuator constraints, such as wheel slipping prevention. Another example is trajectory planning which is specifically designed to deal with motor current and battery voltage constraints [4].

A contribution of this paper is development of the dynamic model of mobile robot that accounts for velocity and acceleration limits and enables expression of acceleration limits and velocity limit curve. Using this model, the algorithm for time-optimal velocity planning along predefined path subject to given constraints is successfully implemented and used in real time.

This paper is organized as follows. In section 2 a time-optimal velocity planning algorithm is introduced. In section 3 the time-optimal velocity planning algorithm is applied for the middle size mobile robot and the corresponding dynamic model is derived. In section 4 experimental results are presented. The paper ends with conclusions and ideas for future work.

II. Time-Optimal Velocity Planning

A path $q(s)$ can be defined as a twice-differentiable curve in the configuration space $q(s) : [0, s_g] \rightarrow \mathcal{C}$, that maps some path parameter s to a curve in configuration space $\mathcal{C}$, where s_g is path parameter at goal configuration. Path may be found by e.g. roadmap or sampling based methods (see [5] for path planning methods overview). To specify a trajectory, we first introduce a *time scaling* function $s(t)$ as $s(t) : [0, t_g] \rightarrow [0, s_g]$, which assigns a value s to each time $t \in [0, t_g]$. The time scaling $s(t)$ is assumed to be twice-differentiable and monotonic, i.e. $\dot{s}(t) > 0,\ t \in (0, t_g)$, where $\dot{s}(t)$ denotes time derivative of $s(t)$. The twice-differentiability of $s(t)$ ensures that the acceleration is well defined and bounded. Using both path and time scaling, a trajectory can now be defined as

$$q(s(t)) : [0, t_g] \rightarrow \mathcal{C}, \tag{1}$$

which is short-written as $q(t)$.

We assume that a mobile robot is operating in a planar workspace, so that its configuration $q(t)$ is given by position coordinates $(x(t), y(t))$ and orientation $\theta(t)$, i.e. $q(t) = [x(t)\ y(t)\ \theta(t)]^T$. In this case the requirement for

twice differentiable path translates to the requirement for continuous path curvature. Additionally, if the motion is to be planned for a nonholonomic mobile robot, the planned path must also satisfy nonholonomic constraints of the particular robot.

We choose that the path parameter s denotes the distance traveled along the path in the case that robot's motion has translation component. In this case longitudinal velocity $v(t)$ is non-zero (i.e. $v(t) \neq 0$) and longitudinal and angular velocities can be expressed as functions of s

$$v(t) = \dot{s}(t), \quad \omega(t) = \kappa(s(t))\dot{s}(t), \tag{2}$$

where $\omega(t)$ is angular velocity and $\kappa(s)$ is signed path curvature. By differentiating (2), we obtain accelerations

$$\dot{v}(t) = \ddot{s}(t), \quad \dot{\omega}(t) = \frac{d\kappa(s)}{ds}\dot{s}(t)^2 + \kappa(s(t))\ddot{s}(t). \tag{3}$$

In case of pure rotation, i.e. $v(t) = 0$ and $\omega(t) \neq 0$, s will denote the traversed angle. Then the velocities are

$$v(t) = 0, \quad \omega(t) = \dot{s}(t), \tag{4}$$

and accelerations

$$\dot{v}(t) = 0, \quad \dot{\omega}(t) = \ddot{s}(t). \tag{5}$$

The motion planning problem is solved using the decoupled approach. Therefore, the trajectory planning module takes a smooth path $q(s)$ as an input, and tries to find the fastest feasible trajectory that follows this path. The term "fastest" actually denotes the time-optimal scaling $s(t)$, whereas the term "feasible" refers to the actuator limits and possibly other external constraints.

The appropriate optimal time-scaling algorithm for our purpose is developed by Dobrow et al. [2] and Shin and McKay [1], see also [5]. To make the paper mostly self-contained we give a short description of the algorithm. Let a general dynamic model of the robot be given by

$$u = M(q)\ddot{q} + \dot{q}^T\Gamma(q)\dot{q} + g(q), \tag{6}$$

where u is the vector of generalized forces acting on the generalized coordinates q, $M(q)$ is an $n_\mathcal{C} \times n_\mathcal{C}$ symmetric, positive definite mass or inertia matrix, $\Gamma(q) \in \mathbb{R}^{n_\mathcal{C} \times n_\mathcal{C} \times n_\mathcal{C}}$ can be viewed as an $n_\mathcal{C}$-dimensional column vector, where each element is a matrix whose elements are Christoffel symbols of the inertia matrix $M(q)$, and $g(q) \in \mathbb{R}^{n_\mathcal{C}}$ is a vector of gravitational forces. The developed forces are subject to the actuator limits

$$u_i^{\min}(q, \dot{q}) \leq u_i \leq u_i^{\max}(q, \dot{q}). \tag{7}$$

In the most general form the actuator limits are expressed as functions of the robot configuration and velocity. An example may be the torque available to accelerate a DC motor, which decreases as its angular velocity increases.

By expressing a path q as a function of parameter s, (6) can be written more compactly as the vector equation

$$a(s)\ddot{s} + b(s)\dot{s}^2 + c(s) = u, \tag{8}$$

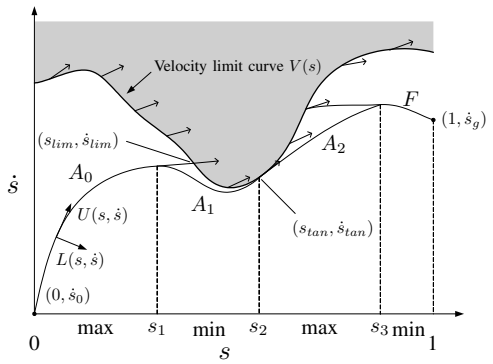


Fig. 1. An illustration of the optimal time-scaling algorithm. The feasible accelerations at points on the velocity limit curve are shown as arrows.

which defines robot dynamic model constrained to the path $q(s)$. The vector functions $a(s)$, $b(s)$, and $c(s)$ are inertial, velocity product, and gravitational terms in terms of s, respectively.

Because the robot motion is constrained to the path $q(s)$, its state at any time is determined by $(s, \dot{s})$. Now the actuator limits can be expressed as a function of $(s, \dot{s})$, yielding lower limit $u^{\min}(s, \dot{s})$ and upper limit $u^{\max}(s, \dot{s})$. From equation (8), we conclude that the system must satisfy the constraints

$$u^{\min}(s, \dot{s}) \leq a(s)\ddot{s} + b(s)\dot{s}^2 + c(s) \leq u^{\max}(s, \dot{s}). \tag{9}$$

This equation enables us to express the minimum and maximum accelerations $\ddot{s}$ as functions of parameters $(s, \dot{s})$, which are required to obtain time-optimal scaling function. We denote the minimum and maximum accelerations $\ddot{s}$ satisfying the i-th component of equation (9) by $L_i(s, \dot{s})$ and $U_i(s, \dot{s})$, respectively. We additionally define overall acceleration limits as

$$L(s, \dot{s}) = \max_{i \in 1 \ldots n_\mathcal{C}} L_i(s, \dot{s}), \quad U(s, \dot{s}) = \min_{i \in 1 \ldots n_\mathcal{C}} U_i(s, \dot{s}), \tag{10}$$

the actuator limits (9) can now simply be expressed as

$$L(s, \dot{s}) \leq \ddot{s} \leq U(s, \dot{s}). \tag{11}$$

The problem of finding the time-optimal trajectory constrained to a path is defined as: given a path $q(s) : [0, s_g] \to \mathcal{C}$, an initial state $(0, \dot{s}_0)$, and a final state $(s_g, \dot{s}_g)$, find a monotonically increasing twice-differentiable time scaling $s(t) : [0, t_g] \to [0, s_g]$ that (i) satisfies $s(0) = 0$, $\dot{s}(0) = \dot{s}_0$, $s(t_g) = s_g$, $\dot{s}(t_g) = \dot{s}_g$, and (ii) minimizes the total travel time t_g along the path while respecting the actuator constraints (11) for all time $t \in [0, t_g]$.

The problem is best visualized in the $(s, \dot{s})$ state space. The feasible acceleration constraint (11) can be illustrated as a cone of tangent vectors defined at any state $(s, \dot{s})$. Then the lower edge of the cone corresponds to the minimum acceleration $L(s, \dot{s})$, while upper edge corresponds to the maximum acceleration $U(s, \dot{s})$. The interior of the cone corresponds to a range of feasible accelerations $\ddot{s}$ at the state $(s, \dot{s})$ along the path so that $L(s, \dot{s}) \leq \ddot{s} \leq U(s, \dot{s})$. One such cone is illustrated in Fig. 1.

The problem is now to find a curve from $(0, \dot{s}_0)$ to $(s_g, \dot{s}_g)$ such that $\dot{s} \geq 0$ and the curve tangent is inside the cone at each state. Further, to minimize the travel time, the velocity $\dot{s}$ along the path should be maximized. This can be obtained by maximizing the area beneath the curve. A consequence of this is that the curve always follows the upper or lower bound of the cone at each state, i.e. the system always operates at minimum or maximum acceleration. The problem is now to find an alternating sequence, i.e. the switching points between maximum and minimum accelerations.

Actuation constraints (11) impose that there could be some states at which there is no feasible acceleration that is required for the system to continue to follow the path. This region is depicted in gray in Fig. 1 and is called inadmissible region. If the robot state is in this region, it will leave the path immediately. But even if robot state is in admissible region, robot may still be doomed to leave the path, no matter of the selected command acceleration. This occurs if the range of admissible accelerations is directed to the inadmissible region.

Here it is assumed that, for any s, the robot is strong enough to maintain its configuration statically, so all $\dot{s} = 0$ states are admissible and the path can be executed arbitrarily slow. It is also assumed that as $\dot{s}$ increases from zero for a given s, there will be at most one switch from admissible to inadmissible, which occurs at the *velocity limit curve* $V(s)$, consisting of states $(s, \dot{s})$ satisfying

$$L(s, \dot{s}) = U(s, \dot{s}). \tag{12}$$

Because of $\min(\cdot)$ and $\max(\cdot)$ function used in calculus of $L(s, \dot{s})$ and $U(s, \dot{s})$, these functions, and velocity limit curve are generally not smooth. However, in some special cases the equation (12) may have multiple solutions $\dot{s}$ for a single value of s. This may be due to friction in the system or weak actuators. In this case, there may be inadmissible "islands" in the phase plane. This complicates the problem of finding an optimal time scaling, and the solution is given in [1].

The algorithm that gives optimal time scaling, i.e. the optimal sequence of s values where the switching between maximum and minimum acceleration should occur, is given by the following algorithm [5]:

1) Initialize an empty list of switches $\mathcal{S} = \{\}$ and a switch counter $i = 0$. Set $(s_i, \dot{s}_i) = (0, \dot{s}_0)$.
2) Integrate the equation $\ddot{s} = L(s, \dot{s})$ backward in time from $(s_g, \dot{s}_g)$ until the velocity limit curve is penetrated (reached transversally, not tangentially), $s = 0$, or $\dot{s} = 0$ at $s < s_g$. There is no solution if $\dot{s} = 0$ is reached. Otherwise, call this phase plane curve F and proceed to the next step.
3) Integrate the equation $\ddot{s} = U(s, \dot{s})$ forward in time from $(s_i, \dot{s}_i)$. Call this curve A_i. Continue integrating until either A_i crosses F or A_i penetrates the velocity limit curve. (If A_i crosses $s = s_g$ or $\dot{s} = 0$ before either of these two cases occurs, there is no solution.) If A_i crosses F, then increment i, let s_i be the s value at which the crossing occurs, and append s_i to the list of switches $\mathcal{S}$. The problem is solved and $\mathcal{S}$ is the solution. If instead the velocity limit curve is penetrated, let $(s_{lim}, \dot{s}_{lim})$ be the point of penetration and proceed to the next step.
4) Search the velocity limit curve $V(s)$ forward in s from $(s_{lim}, \dot{s}_{lim})$ until finding the first point where the feasible acceleration ($L = U$ on the velocity limit curve) is tangent to the velocity limit curve. If the velocity limit is $V(s)$, then a point $(s_0, v(s_0))$ satisfies the tangency condition if $\frac{dv}{ds}|_{s=s_0} = U(s_0, v(s_0))/v(s_0)$. Call the first tangent point reached $(s_{tan}, \dot{s}_{tan})$. From $(s_{tan}, \dot{s}_{tan})$, integrate the curve $\ddot{s} = L(s, \dot{s})$ backward in time until it intersects A_i. Increment i and call this new curve A_i. Let s_i be the s value of the intersection point. This is a switch point from maximum to minimum acceleration. Append s_i to the list $\mathcal{S}$.
5) Increment i and set $(s_i, \dot{s}_i) = (s_{tan}, \dot{s}_{tan})$. This is a switch point from minimum to maximum acceleration. Append s_i to the list $\mathcal{S}$. Go to step 3.

An illustration of the time-scaling algorithm is shown in Fig. 1. Although the described optimal time-scaling algorithm is general, to use it we have to model actuator limits (11), which are specific for each particular robot type.

One drawback of planning at maximum accelerations is that it does not leave any maneuver space to recover from eventual trajectory tracking error. Such an error can be caused by many factors, e.g. a non-flat floor, which can be characterized as an unpredictable external disturbance. A possible solution is to to leave some acceleration reserves and work with lower accelerations than the actual maximums.

III. Trajectory Planning for Mobile Robot with Velocity and Acceleration Constraints

Here we derive a time-optimal scaling algorithm specifically for robots that have fixed limits on maximum longitudinal and angular velocities and accelerations. It is thus assumed that a robot has wheel-driving motors that are strong enough to attain both maximum longitudinal and angular acceleration simultaneously. A typical example of such robots is the Pioneer 3DX mobile robot shown in Fig. 2, whose maximum allowed accelerations can be customized in robot firmware. The Pioneer 3DX robot will therefore be used thorough this paper for algorithm derivation and experiments. This is a nonholonomic mobile robot with the differential drive that has two driving wheels and a caster (passive) wheel. An assumption of independent maximum longitudinal and angular accelerations may not be fulfilled if the user chooses too high acceleration and velocity limits, however, here we do not study such cases.

The command inputs to the Pioneer 3DX robot are referent longitudinal and angular velocities. However, dynamic model given by equation (6) requires use of generalized forces as the command input. Therefore longitudinal acceleration a and angular acceleration α (which are proportional to the corresponding forces) will be used as components of generalized forces vector u, i.e. $u = [a\ \alpha]^T$. Final output of

Fig. 2. Pioneer 3DX mobile robot.

the trajectory planning algorithm will be the velocity profile as the function of time.

Normally, robot configuration is described by the vector $[x\ y\ \theta]^T$, where (x, y) denotes robot position, and θ its orientation. However, with this choice it is not possible to directly use described time-scaling algorithm since we have to control three variables while there are only two components of the command input vector—this is a consequence of the nonholonomic constraint. Nevertheless, the velocity pair $[v\ \omega]^T$ can be used to describe robot state, so that dynamic model of the robot can be written as

$$\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = u. \tag{13}$$

Although the state vector $[v\ \omega]^T$ does not answer us about the robot configuration directly, this information can be obtained implicitly through the time history of the velocity vector. In this way the configuration can be computed if an initial configuration $[x_0\ y_0\ \theta_0]^T$, time history of the velocity vector and kinematic model of the robot are known. Kinematic model of the differential drive robot is given by

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{14}$$

Additionally, the nonholonomic constraint that comes from the fact that the driving wheels can only roll, but not slip, must be fulfilled:

$$-\dot{x}\sin\theta + \dot{y}\cos\theta = 0. \tag{15}$$

For $v \neq 0$ we can substitute (2) and (3) into dynamic model (13), obtaining

$$\begin{bmatrix} 1 \\ \kappa(s) \end{bmatrix} \ddot{s} + \begin{bmatrix} 0 \\ \frac{d\kappa(s)}{ds} \end{bmatrix} \dot{s}^2 = \begin{bmatrix} a \\ \alpha \end{bmatrix}. \tag{16}$$

From this model, and using analogy with (8) we have the following limits due to limited longitudinal acceleration

$$L_1(s, \dot{s}) = a_{min}(s, \dot{s}), \quad U_1(s, \dot{s}) = a_{max}(s, \dot{s}), \tag{17}$$

where the minimum longitudinal acceleration is nonlinear function given by

$$a_{min}(s, \dot{s}) = \begin{cases} \bar{a}_{min}, & \dot{s} \in (-\bar{v}_{max}, \bar{v}_{max}) \\ 0, & \text{otherwise} \end{cases}, \tag{18}$$

where $\bar{a}_{min}$ is a negative constant that denotes the minimum longitudinal acceleration, while $\bar{v}_{max}$ is the maximum absolute longitudinal velocity of the robot. The maximum longitudinal acceleration is given similarly by

$$a_{max}(s, \dot{s}) = \begin{cases} \bar{a}_{max}, & \dot{s} \in (-\bar{v}_{max}, \bar{v}_{max}) \\ 0, & \text{otherwise} \end{cases}, \tag{19}$$

where $\bar{a}_{max}$ is a positive constant denoting the maximum longitudinal acceleration. It is usually $\bar{a}_{max} = |\bar{a}_{min}|$, although some electronic motor drivers achieve $|\bar{a}_{min}| > \bar{a}_{max}$, i.e. they enable faster braking than accelerating.

From robot dynamic model (16), the acceleration constraints due to angular acceleration limits are

$$L_2(s, \dot{s}) = \begin{cases} \dfrac{\alpha_{min}(s, \dot{s}) - \frac{d\kappa(s)}{ds}\dot{s}^2}{\kappa(s)}, & \kappa(s) > 0 \\ \text{undefined}, & \kappa(s) = 0 \\ \dfrac{\alpha_{max}(s, \dot{s}) - \frac{d\kappa(s)}{ds}\dot{s}^2}{\kappa(s)}, & \kappa(s) < 0 \end{cases}$$
$$U_2(s, \dot{s}) = \begin{cases} \dfrac{\alpha_{max}(s, \dot{s}) - \frac{d\kappa(s)}{ds}\dot{s}^2}{\kappa(s)}, & \kappa(s) > 0 \\ \text{undefined}, & \kappa(s) = 0 \\ \dfrac{\alpha_{min}(s, \dot{s}) - \frac{d\kappa(s)}{ds}\dot{s}^2}{\kappa(s)}, & \kappa(s) < 0 \end{cases}, \tag{20}$$

where the minimum angular acceleration is nonlinear function given by

$$\alpha_{min}(s, \dot{s}) = \begin{cases} \bar{\alpha}_{min}, & \kappa(s) = 0 \vee \dot{s} \in \left[\frac{-\bar{\omega}_{max}}{|\kappa(s)|}, \frac{\bar{\omega}_{max}}{|\kappa(s)|}\right] \\ 0, & \text{otherwise} \end{cases}, \tag{21}$$

where $\bar{\alpha}_{min}$ is a negative constant denoting minimum angular acceleration, while $\bar{\omega}_{max}$ is the maximum absolute angular velocity of the robot. The maximum angular acceleration is given similarly by

$$\alpha_{max}(s, \dot{s}) = \begin{cases} \bar{\alpha}_{max}, & \kappa(s) = 0 \vee \dot{s} \in \left[\frac{-\bar{\omega}_{max}}{|\kappa(s)|}, \frac{\bar{\omega}_{max}}{|\kappa(s)|}\right] \\ 0, & \text{otherwise} \end{cases}, \tag{22}$$

where $\bar{\alpha}_{max}$ is the a positive constant denoting maximum angular acceleration. For $\kappa(s) = 0$ we see that eq. (20) is undefined, because pure translational motion is not constrained by angular acceleration limit.

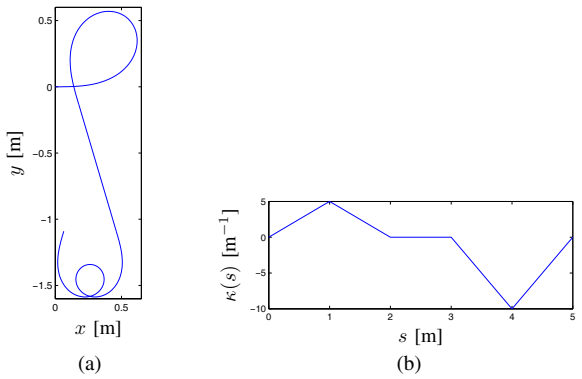
In case when $v = 0$, we have a pure rotation. Then the dynamic model reduces to a single equation $\ddot{s} = \alpha$, from which we derive acceleration limits as

$$L_{1,rot}(s, \dot{s}) = \alpha_{min,rot}, \quad U_{1,rot}(s, \dot{s}) = \alpha_{max,rot}, \tag{23}$$

where

$$\alpha_{min,rot}(s, \dot{s}) = \begin{cases} \bar{\alpha}_{min}, & \dot{s} \in [-\omega_{max}, \omega_{max}] \\ 0, & \text{otherwise} \end{cases}$$
$$\alpha_{max,rot}(s, \dot{s}) = \begin{cases} \bar{\alpha}_{max}, & \dot{s} \in [-\omega_{max}, \omega_{max}] \\ 0, & \text{otherwise} \end{cases}. \tag{24}$$

Computation of the velocity limit curve is complicated by the fact that $\min(\cdot)$ and $\max(\cdot)$ functions are used in equation (10). Thus we do not know in advance from



(a) (b)

Fig. 3. *(a)* The experimental path. *(b)* Curvature profile of the path.

which constraints to compute the curve for particular s. In other words, we must check all possible combinations of i and j in equation $L_i(s,\dot{s}) = U_j(s,\dot{s})$, which could be computationally expensive for large number of constraints.

However, if the values of robot parameters are known in advance, we can substitute them into corresponding equations to predict which i and j combinations are relevant. If we do that for the Pioneer 3DX robot, it becomes evident (as will be demonstrated by the experimental results) that for straight path segments (i.e. $\kappa(s) = 0$), only the condition $L_1(s,\dot{s}) = U_1(s,\dot{s})$ is relevant for computing the velocity limit curve. Thus, from $L_1(s,\dot{s}) = U_1(s,\dot{s})$ and using (17) the velocity limit curve is $V_{11}(s) = \bar{v}_{max}$.

For $\kappa(s) \neq 0$ three conditions can be relevant, namely $L_2(s,\dot{s}) = U_2(s,\dot{s})$, $L_1(s,\dot{s}) = U_2(s,\dot{s})$, and $L_2(s,\dot{s}) = U_1(s,\dot{s})$. All of these equations have closed form solution that is easy to find. It must be emphasized that these results are not general—for different robots or parameter values the involved conditions might change.

IV. Experimental Results

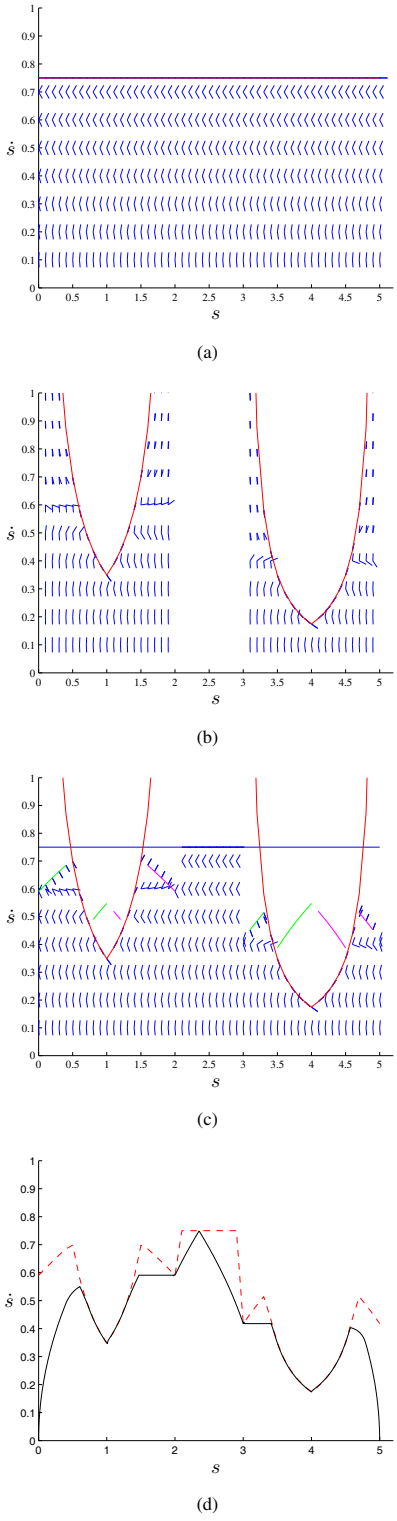
The values of model parameters for the Pioneer 3DX robot that are used for trajectory planning experiments are shown in Table I. The parameters that are marked by a star can be adjusted by user in robot firmware.

TABLE I

Parameters of the Pioneer 3DX robot.

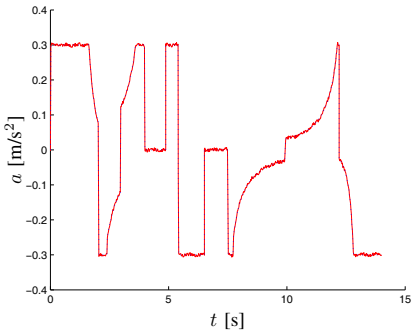
Parameter	*Symbol*	*Value*	*Unit*
Mass	m	28.05	kg
Inertia moment	J	0.175	kg m^2
Min. long. acceleration*	$\bar{a}_{min}$	-0.3	m/s^2
Max. long. acceleration*	$\bar{a}_{max}$	0.3	m/s^2
Min. ang. acceleration*	$\bar{\alpha}_{min}$	-1.745	rad/s^2
Max. ang. acceleration*	$\bar{\alpha}_{max}$	1.745	rad/s^2
Max. long. velocity*	$\bar{v}_{max}$	0.75	m/s
Max. ang. velocity*	$\bar{\omega}_{max}$	1.745	rad/s

An experimental path and its curvature profile are shown in Fig. 3. It is generated using algorithms described in [6] and consists of four clothoids and a straight line segment. It has parts with high curvature in order to better demonstrate

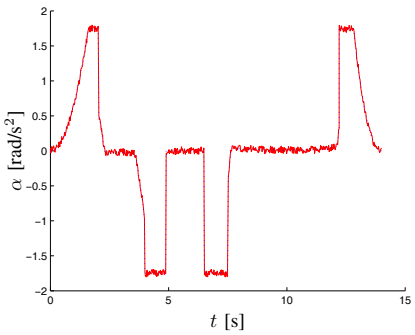


(a) (b) (c) (d)

Fig. 4. Acceleration limits and velocity limit curves computed for the Pioneer 3DX robot with path in Fig. 3. *(a)* Velocity limit curve $V_{11}(s)$. *(b)* Velocity limit curve $V_{22}(s)$. *(c)* Overall velocity limit curve $V(s)$. Acceleration limits $L_i(s,\dot{s})$, $U_j(s,\dot{s})$, and velocity limit curves $V_{ij}(s)$ originating from different constraints are plotted with the following colors: $i=1, j=1$ (blue), $i=2, j=2$ (red), $i=1, j=2$ (green), $i=2, j=1$ (magenta). *(d)* Time-optimal velocity along the experimental path (black-solid) and overall velocity limit curve $V(s)$ (red-dashed). Note: The slopes of acceleration cones in figures (a)-(c) are obtained as $\frac{d\dot{s}}{ds} = \frac{d\dot{s}}{dt}\frac{dt}{ds} = \frac{\ddot{s}}{\dot{s}}$, therefore e.g. in figure (a) we see that slope of the cones decreases as $\dot{s}$ increases.



(a)



(b)

Fig. 5. Accelerations computed by time-optimal scaling algorithm (blue-dotted) and reference accelerations generated by nonlinear controller (red-solid). *(a)* Longitudinal acceleration. *(b)* Angular acceleration.

effects of angular velocity and acceleration limits. Acceleration limits L_i and U_i and velocity limit curves V_{ii} obtained from constraints $L_i(s, \dot{s}) = U_i(s, \dot{s})$ are shown in Fig. 4 (a) and (b) for $i = 1$ and $i = 2$, respectively. Velocity limit curves V_{12} and V_{21} are not displayed in a separate figure because of lack of space, however, in Fig. 4 (c) all velocity limit curves V_{ij} are shown, and the overall velocity limit curve $V(s)$ is their minimum. It can be noticed that in zero-curvature segment only the velocity limit curve $V_{11}(s)$ is relevant.

The optimal time-scaling algorithm is implemented in C++ programming language. One has to be extremely careful in algorithm implementation, as many involved functions have discontinuities that make numerical integration prone to errors. To accelerate algorithm execution, the values of acceleration limits and velocity limit curve are stored into a lookup table so that most of the computational burden stems from numerical integration. With the first order integration and step size of 20 ms a reasonable precision is achieved with planning time of less than 10 ms on 3 GHz PC, which is fast enough for real-time application.

The trajectory tracking experiment is performed in simulator based on ODE library that simulates rigid body dynamics considering variables such as mass, inertia, velocity, friction, etc. [7]. The robot commands are generated using feedback control with nonlinear trajectory tracking controller [8], which enables compensation of errors generated by parameter and measurement uncertainties and other disturbances. For feedback control to be effective, in experiments we used acceleration limits somewhat higher than those used for trajectory planning.

The final velocity profile obtained by the time-optimal scaling algorithm is shown in Fig. 4 (d). It can be seen that it consists of four acceleration segments, four deceleration segments and two constant velocity segments. Based on the computed velocity profile and using numerical integration, one can compute velocity and acceleration profile in dependency of time. For the velocity profile from Fig. 4 (d) the correspondent longitudinal and angular acceleration profiles are displayed in Fig. 5. It can be seen that both accelerations never exceed the limits. The total travel time of the obtained trajectory is approximately 14 s, while the average longitudinal velocity is 0.36 m/s.

V. Conclusion

In this work the optimal time-scaling algorithm is applied for trajectory planning along the predefined path while respecting the actuator constraints. The dynamic model of the mobile robot that accounts for velocity and acceleration limits is developed and utilized to express acceleration limits and velocity limit curve for the optimal time-scaling algorithm, which are further used for algorithm implementation in case of Pioneer 3DX differential-drive robot.

This paper covers only simple velocity and acceleration constraints, but it establishes a basis for design of more complex robot models that account for e.g. case with dependent maximum longitudinal and angular accelerations, limited grip between the wheels and the ground (this is very important for reliable operation of lightweight robots at high velocities), or tip over prevention of tall mobile robots. Such more complex models will be developed in future works. It is also planned to test the planner on the real robot and to further optimize numerical integration by using more advanced integration techniques.

References

[1] K. G. Shin and N. D. McKay, "Minimum-time control of robot manipulators with geometric path constraints," *IEEE Transactions on Automatic Control*, vol. 30, no. 6, pp. 531–541, 1985.

[2] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *International Journal of Robotics Research*, vol. 4, no. 4, pp. 3–17, 1985.

[3] M. Hentschel, D. Lecking, and B. Wagner, "Deterministic path planning and navigation for an autonomous fork lift truck," in *6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*, 2007.

[4] J. Choi and B. Kim, "Near-time-optimal trajectory planning for wheeled mobile robots with translational and rotational sections," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 85–90, 2001.

[5] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion*. MIT Press, 2005.

[6] M. Brezak and I. Petrović, "Path smoothing using clothoids for differential drive mobile robots," 2011, accepted for publication for World Congress of the International Federation of Automatic Control (IFAC).

[7] G. Klančar, M. Brezak, I. Petrović, and D. Matko, "Two approaches to mobile robots simulator design," in *Proceedings of the 6th EUROSIM congress on Modelling and Simulation*, 2007.

[8] M. Brezak, I. Petrović, and N. Perić, "Experimental comparison of trajectory tracking algorithms for nonholonomic mobile robots," in *Proc. of 35th Annual Conference of the IEEE Industrial Electronics Society*, Porto, Portugal, 2009.