2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Feedback stabilizer-based trajectory planning of mobile robots with kinematic constraints

Xuebo Zhang, *Member, IEEE*, Yuan Li and Yongchun Fang, *Senior Member, IEEE,* Baoquan Li

***Abstract*— Many theoretic approaches for feedback stabilization control of nonholonomic mobile robots cannot be directly applied to practical robots since various kinematic constraints such as the velocity and acceleration limits are not considered in existing methods. To deal with this issue, we aim to propose a generic approach which first uses an (arbitrary) feedback stabilizer to generate the 'path' and then rebuilt the corresponding 'trajectory' along this 'path' to meet various kinematic constraints, which ultimately gives a practical satisfactory solution for local trajectory planning. Specifically, a general framework is established to transform feedback stabilizers into a feasible and highly efficient trajectory planner by using path generation and optimal velocity planning techniques, considering both kinematic and differential constraints. Extensive simulation results are provided to validate the proposed approach.**

I. Introduction

One of the most fundamental problem for nonholonomic mobile robots (WMR) is the closed-loop stabilization control due to the well-known Brockette's necessary condition, which states that nonholonomic systems cannot be asymptotically stabilized by any time-invariant continuous state feedback control law [1]. As a consequence, many researchers devotes themselves to this challenging problem and a lot of controllers are proposed in the literature [2]–[8].

Though great theoretical achievements have been made, careful review of existing literatures and patents reveals that few stabilization controllers are ultimately applied to the practice or integrated on the commercial mobile robots. In contrast, it is interesting that, in the robotics and control applications community, motion planning and trajectory tracking control are more focused than stabilization control especially faced with practical applications [10]–[14]. In this situation, one may ask: what are the key obstacles in the way of existing stabilization controllers to real applications?

One reason is that the velocity and acceleration limits cannot be guaranteed to be satisfied theoretically for most existing stabilization controllers. Some of remarkable achievements in [15]–[17] only considers a part of these constraints, which largely limit its application for true mobile robots to conduct realizable tasks. Another reason is that stabilization controllers are not appropriate to be utilized in the scenario with large pose errors. Since a general idea of the feedback control is that "greater error, greater control", when the control gains are tuned to generate appropriate control inputs for the case of small pose errors, it is prone to generate too large control signals to violate kinematic constraints. Alternatively, if we decrease the control gains to generate realizable motions in the presence of large pose errors, the servoing efficiency will become lower and lower as the pose error decreases.

With the aforementioned two reasons in mind, a question naturally emerges: when facing various state and control constraints, are the existing luxuriant stabilizers almost useless in practice? In this paper, we try to give the answer "No!" and we aim to propose an approach which first uses an (arbitrary) feedback stabilizer to generate the 'path' and then rebuilt the corresponding 'trajectory' along this 'path' to meet various kinematic constraints, which ultimately gives a practical satisfactory solution for local trajectory planning.

One of our essential idea in this paper is to regard the feedback stabilization system as a "path generator" rather than a "trajectory generator". By numerically solving the closed-loop differential system equations, a discrete trajectory can be represented by a time-related sequence of discrete points in the configuration space. By discarding the time label, the satisfaction of kinematic constraints can be rebuilt. Another idea is to convert the implicit discrete path into an explicit analytical expression. In this way, we can obtain a more compact representation of the path, which is appropriate for teleoperation or remote control with low-bandwidth communications with lower-level controllers. Moreover, this analytical expression facilitates the subsequent replanning process since the computation of the derivative and curvature becomes more simple.

By combining the aforementioned two ideas together, we propose a framework of transforming practical infeasible feedback stabilization controllers into a feasible and highly efficient trajectory planner by using path generation and optimal velocity planning techniques, despite the presence of both kinematic and differential constraints. Various comparative simulation results are conducted to verify the effectiveness of the proposed approach.

The main contribution of this paper is summarized as: 1) a bridge is built between the feedback stabilization control and highly efficient local trajectory planning, which extends the function of existing stabilizers to act as new types of trajectory planners; 2) The established framework is stabilizer-free, which means that the stabilizer can be chosen arbitrarily from existing methods.

This work is supported in part by National Natural Science Foundation of China under Grant 61203333, in part by Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant 20120031120040, in part by Tianjin Natural Science Foundation under Grant 13JCQNJC03200, and in part by the Open Project of Chongqing Key Laboratory of Computational Intelligence under Grant CQ-LCI-2013-03.

X. Zhang, Y. Li, Y. Fang (corresponding author) and B. Li are with the Institute of Robotics and Automatic Information System(IRAIS) and Tianjin Key Laboratory of Intelligent Robotics, Nankai University, Tianjin 300071, China. Email: {zhangxb, yfang}@robot.nankai.edu.cn

Note that we only consider local trajectory planning problem in this work, while the upper level motion planning with obstacles are beyond the scope of this paper.

II. General Framework

A. Basic Framework

There are three basic procedures in the general framework:

1) Feedback stabilizer-based path generation by numerical solvers. The kinematic/dynamic model of mobile robots, together with the stabilizers, constitutes a closed-loop system represented by differential equations, which can be solved using numerical methods such as the Euler method or Runge-Kutta methods.Subsequently, appropriate discrete trajectory points in the configuration space are carefully selected to be utilized in the next procedure to generate explicit paths.
2) Explicit analytical path derivation. we discard the time label of the discrete trajectory points to yield a series of discrete path points. To deal with the possible cusp points in the trajectorythe discrete path is segmented into several path segments, and the derivation of the analytical form of every path segment becomes a constrained quadratic programming problem considering the boundary conditions.
3) Optimal velocity allocation along the analytical path, wherein the linear and angular velocity/acceleration limits are guaranteed to be satisfied. Therefore, a practically feasible and efficient trajectory is obtained.

B. Problem statement

In this paper, we consider a unicycle mobile robot:

$$\dot{x} = v\cos\theta \tag{1}$$
$$\dot{y} = v\sin\theta \tag{2}$$
$$\dot{\theta} = w \tag{3}$$

where the system state is $\boldsymbol{x} = [x \quad y \quad \theta]^{\mathrm{T}}$ representing the robot position $x, y \in \mathbb{R}$ and orientation $\theta \in \mathbb{R}$, and the control input is $\boldsymbol{u} = [v \quad w]^{\mathrm{T}}$ with $v \in \mathbb{R}$ and $w \in \mathbb{R}$ being the linear and angular velocities of the mobile robot.

To derive a practically feasible trajectory, the velocity and acceleration should satisfy the following conditions:

$$\|v\| \leq v_{\max}, \quad \|a_v\| \leq a_{v\max} \tag{4}$$
$$\|w\| \leq w_{\max}, \quad \|a_w\| \leq a_{w\max} \tag{5}$$

where $a_v,\ a_w \in \mathbb{R}$ denotes the linear acceleration and angular acceleration, respectively. $v_{\max},\ a_{v\max},\ w_{\max},\ a_{w\max} \in \mathbb{R}$ represent the corresponding maximum value of the velocities and accelerations $v,\ a_v,\ w,\ a_w$.

Objective: for a given stabilization controller $v(\boldsymbol{x}, t)$ and $w(\boldsymbol{x}, t)$, the objective is to transform this feedback stabilizer, which is probably infeasible due to velocity and acceleration constraints, into a practically feasible and high efficient trajectory planner as described in section II-A.

For simplicity, we chose the reference world coordinate system to be coincident with the target robot frame. In this case, the target pose of the mobile robot becomes $\boldsymbol{x}_g = [0,\ 0,\ 0]^{\mathrm{T}}$, and thus the stabilization task is to design $w(\cdot)$ and $v(\cdot)$ to drive the robot from its initial pose to the origin.

III. Feedback stabilization-based path generation

Given the kinematic model of mobile robots such as (1)–(3), many existing feedback stabilizers can be chosen to drive the mobile robot from its initial pose to the origin. Without loss of generality, we choose the polar coordinate-based stabilizer as a typical example due to its natural driving behavior [9], [20]. We introduce the polar coordinates as

$$\rho(x,\ y) = \sqrt{x^2 + y^2}, \tag{6}$$
$$\gamma(x,\ y,\ \theta) = \operatorname{atan2}(y, x) - \theta + \pi, \tag{7}$$
$$\delta(x,\ y) = \gamma(x,\ y,\ \theta) + \theta = \operatorname{atan2}(y, x) + \pi, \tag{8}$$

where $\rho,\ \gamma,\ \delta \in \mathbb{R}$ are the polar coordinate-related parameters. Subsequently, the feedback stabilizer is then designed as [9], [20]

$$v(\rho,\ \gamma) = k_1\rho\cos\gamma, \tag{9}$$
$$w(\rho,\ \gamma,\ \delta) = k_2\gamma + k_1\frac{\sin\gamma\cos\gamma}{\gamma}(\gamma + k_3\delta). \tag{10}$$

with $k_1,\ k_2,\ k_3 \in \mathbb{R}$ being constant control gains.

By substituting (6)–(8) into (9) and (10), we know that the designed linear velocity and angular velocity are both functions of the system state $x,\ y,\ \theta$, namely, they can be written in a form of

$$v(x,\ y,\ \theta) = v\big(\rho(x,\ y),\ \gamma(x,\ y,\ \theta)\big) \tag{11}$$
$$w(x,\ y,\ \theta) = w\big(\rho(x,\ y),\ \gamma(x,\ y,\ \theta),\ \delta(x,\ y)\big) \tag{12}$$

Subsequently, substituting the resultant linear velocity and angular velocity into the open-loop error dynamics (1)–(3), we can obtain the closed-loop error dynamics as

$$\dot{x} = v(x,\ y,\ \theta)\cos\theta; \tag{13}$$
$$\dot{y} = v(x,\ y,\ \theta)\sin\theta; \tag{14}$$
$$\dot{\theta} = w(x,\ y,\ \theta). \tag{15}$$

Additionally, we augment the system with an equation on the path length $s \in \mathbb{R}$ as

$$\dot{s} = v(x,\ y,\ \theta). \tag{16}$$

So far, it is seen that the closed-loop system dynamics can be described by a group of nonlinear differential equations (13)–(16) with respect to the system state variables. Hence, given the initial state, these differential equations can be solved using standard numerical methods such as Euler method or Runge-Kutta methods. Let the sampling period be T, then we can numerically obtain the path point sequences $x(kT), y(kT), \theta(kT)$ and the corresponding path length $s(kT)$ at the sampling instants.

To obtain higher precision, it is generally required that the sampling frequency is sufficiently large, which leads to a small sampling period. In this case, the discrete points are very dense and the volume of the data might be huge. Hence, we introduce a point selection mechanism to just save representative points in the computer memory for the subsequent analytical path derivation. In this way, not only the numerical precision is guaranteed, but also the efficiency of the subsequent algorithms are improved and the usage of the memory is decreased.

Let the initial pose of the robot be $x(0) = x_0,\ y(0) = y_0,\ \theta(0) = \theta_0$ with $x_0,\ y_0,\ \theta_0 \in \mathbb{R}$ being arbitrarily known constants, with the target pose being $x(t_f) = 0,\ y(t_f) = 0,\ \theta(t_f) = 0$ (since the reference world coordinate system is chosen as the target robot frame as stated in section II-B). The feedback stabilization-based path generation and point selection algorithm is described in Algorithm 1.

In Algorithm 1, the sign function $sgn(\cdot)$, which is utilized to find the cusp points, is defined as

$$sgn(\xi) = \begin{cases} 1; & if\ \xi \geq 0 \\ -1; & if\ \xi < 0 \end{cases} \tag{17}$$

with $\xi \in \mathbb{R}$ being an arbitrary variable.

Several key points should be remarked in Algorithm 1:

1) For selection of path points, a threshold $\epsilon \in \mathbb{R}$, which denotes the path length between the current point and the last selected point, is utilized to select path points in a regular way. Line 7 means that the current pose is set as the target pose when they are sufficiently close.
2) Cusp points are detected for the subsequent path segmentation: the index of cusp points among the selected points are recorded in $ptInd(i)\ (i = 1,\ 2,\ \cdots,\ cnt)$ with $cnt \in \mathbb{R}$ being the total number of the cusp points. These points are detected in Line 14 when the velocity direction are changed.
3) As for trajectory generation by numerically solving the forward kinematics, only eight temporary variables $preX,\ preY,\ pre,\ preS,\ curX,\ curY,\ cur\theta,\ curS$, rather than all the trajectory points in every sampling instant, are utilized to decrease the usage of the computer memory and improve the computing efficiency.

The output of this module includes a sequence of selected path points and the corresponding path length $x(i),\ y(i),\ s(i)\ (i = 1,\ 2,\ \cdots,\ N)$ with N being the total number of the selected points. Additionally, the number of the cusp points cnt, their index among the selected points $ptInd(i)\ (i = 1,\ 2,\ \cdots,\ cnt)$, and the corresponding robot orientation $\theta_{cp}(i)$ at these cusp points are recorded.

Algorithm 1 Feedback stabilization-based path generation

Input: An initial pose $(x_0,\ y_0,\ \theta_0)$, the target pose $(0,\ 0,\ 0)$, a feedback stabilizer $v(x,\ y,\ \theta)$, $w(x,\ y,\ \theta)$, sampling period T, a positive threshold $\epsilon \in \mathbb{R}$.

Output: Path points $x_s(i),\ y_s(i)$ selected from the numerically generated trajectory, the corresponding path length $s_s(i),\ (i = 1,\ 2,\ \cdots,\ N)$ with N being the number of the selected points, the number of cusp points cnt and their index $ptInd(i),\ (i = 1,\ 2,\ \cdots,\ cnt)$ within the array of the selected points, the robot orientation $\theta_{cp}(i)$ at the cusp points.

```
1:  Initialization: $k = 0,\ i = 1,\ FLAG = 0,\ N = 1,\ cnt = 0,\ preX = x_s(1) = x_0,\ preY = y_s(1) = y_0,\ pre\theta = \theta_0,\ preS = s_s(1) = 0,\ preV = v(preX,\ preY,\ pre\theta),\ curX = curY = cur\theta = curS = curV = 0$.
2:  Employ the chosen controller $v(x,\ y,\ \theta)$, $w(x,\ y,\ \theta)$ to obtain closed-loop system differential equations (13), (14), (15), and the path evolution equation (16).
3:  repeat
4:      set $k = k + 1$
5:      On the basis of the robot pose and path length at time $(k-1)T$, namely, $preX,\ preY,\ pre\theta,\ preS$, Euler or Runge-Kutta methods are utilized to compute those variables at the time $kT$ from (13)–(16), namely $curX,\ curY,\ cur\theta,\ curS$.
6:      Update the current velocity from (11) as $curV = v(curX,\ curY,\ cur\theta)$ .
7:      if $curX^2 + curY^2 + cur\theta^2 \leq \epsilon^2$ then
8:          $N = i + 1, FLAG = 1$
9:          $s_s(N) = curS,\ x(N) = 0,\ y(N) = 0$
10:     else
11:         if $curS - s_s(i) \geq \epsilon$ then
12:             $i = i + 1$
13:             $s_s(i) = curS,\ x(i) = curX,\ y(i) = curY$
14:         else $\{sgn(preV)! = sgn(curV)\}$
15:             $cnt = cnt + 1,\ ptInd(cnt) = i + 1$
16:             $\theta_{cp}(cnt) = cur\theta,\ i = i + 1$
17:             $s_s(i) = curS,\ x(i) = curX,\ y(i) = curY$
18:         end if
19:     end if
20:     set $preX = curX,\ preY = curY,\ pre\theta = cur\theta,\ preS = curS,\ preV = curV$
21: until $FLAG \neq 0$
```

IV. Path segmentation and explicit path derivation

On the basis of the derived cusp points, we segment the previous discrete path into several path segments, which will be approximated using analytical polynomials with respect to the path length parameter s.

A. Path segmentation

In this subsection, we will introduce a path segmentation mechanism to segment the discrete path into several smooth segments that can be approximated by a polynomial for each segment. Let the whole path denoted by $L : x(i),\ y(i),\ s(i)\ (i = 1,\ 2,\ \cdots,\ N)$, which is divided into $p = (cnt+1)$ path segments $L_k\ (k = 1,\ 2,\ \cdots,\ p)$ by the previously recorded cnt cusp points. The path segmentation process is described in Algorithm 2.

After using the Algorithm 2, the whole path $L :\ x(i),\ y(i),\ s(i)\ (i = 1,\ 2,\ \cdots,\ N)$ is divided into p path segments $L_k :\ \big(x_s(i),\ y_s(i),\ s_s(i)\big)\ (i = N_k,\ N_k + 1,\ \cdots,\ N_{k+1})$, with N_k and N_{k+1} being the starting and ending point index within the whole path.

B. Explicit path derivation

For every path segment $L_k : \big(x_s(i),\ y_s(i),\ s_s(i)\big)\ (i = N_k,\ N_k + 1,\ \cdots,\ N_{k+1})$, we can approximate it with

Algorithm 2 Path segmentation algorithm

```
Input: The whole path $L: (x(i),\ y(i),\ s(i))$ $(i = 1,\ 2,\ \cdots,\ N)$ and the cusp point index $ptInd(i)$ $(i = 1,\ 2,\ \cdots,\ cnt)$.
Output: Starting and ending point index: $N_1,\ N_2,\ \cdots,\ N_{p+1}$ such that a total of $p = (cnt + 1)$ path segments $L_k : (x(i),\ y(i),\ s(i))$ $(i = N_k,\ N_k + 1,\ \cdots,\ N_{k+1})$ are obtained, with $k = 1,\ 2,\ \cdots,\ p$.
1: Initialization: $k = 1,\ N_1 = 1$.
2: repeat
3:     $N_k = ptInd(k)$
4:     set $k = k + 1$
5: until $k > p$
6: $N_{p+1} = N$.
```

polynomials to yield an analytical expression.

Since the robot orientations $\theta_{cp}(k)$ are recorded for common positions $\left(x_s(N_{k+1}),\ y_s(N_{k+1}),\ s_s(N_{k+1})\right)$ connecting the path segment L_k and L_{k+1}, one novel point in this subsection is that we propose to formulate these common robot poses as linear equality constraints that must be satisfied. Consequently, the path approximation for every path segment becomes independent and they can be conducted simultaneously in a parallel way to increase the whole efficiency of the algorithm.

With the selected points for the path segment L_k : $\left(x_s(i),\ y_s(i),\ s_s(i)\right)$ $(i = N_k,\ N_k{+}1,\ \cdots,\ N_{k+1})$, We can parameterize the segment L_k using polynomials as follows:

$$x\left(s\right) = a_{k0} + a_{k1}s + a_{k2}s^2 \cdots + a_{kn}s^n \tag{18}$$

$$y\left(s\right) = b_{k0} + b_{k1}s + b_{k2}s^2 \cdots + b_{kn}s^n \tag{19}$$

where $n \in \mathbb{R}$ is a carefully selected polynomial order, with $a_{ki},\ b_{ki} \in \mathbb{R}$ $(i = 0,\ 1,\ \cdots,\ n)$ being the coefficients to be determined.

By substituting the path points $\left(x_s(i),\ y_s(i),\ s_s(i)\right)$ $(i = N_k,\ N_k + 1,\ \cdots,\ N_{k+1})$ into the (18) and (19), the following relationship is obtained:

$$A\boldsymbol{X} = \boldsymbol{B} \tag{20}$$

with the matrix $A \in \mathbb{R}^{2(N_{k+1}-N_k+1)\times 2(n+1)}$ and vectors $\boldsymbol{X},\ \boldsymbol{B} \in \mathbb{R}^{2(N_{k+1}-N_k+1)}$ being

$$A = \begin{bmatrix} Q & 0 \\ 0 & Q \end{bmatrix} \tag{21}$$

$$\boldsymbol{X} = [\boldsymbol{X}_1^T \quad \boldsymbol{X}_2^T]^T \tag{22}$$

$$\boldsymbol{B} = [\boldsymbol{B}_1^T \quad \boldsymbol{B}_2^T]^T \tag{23}$$

where the matrix $Q \in \mathbb{R}^{(N_{k+1}-N_k+1)\times(n+1)}$ and the vectors $\boldsymbol{X}_1,\ \boldsymbol{X}_2,\ \boldsymbol{B}_1,\ \boldsymbol{B}_2 \in \mathbb{R}^{N_{k+1}-N_k+1}$ are of the following forms:

$$Q = \begin{bmatrix} 1 & s(N_k) & s^2(N_k) & ... & s^n(N_k) \\ 1 & s(N_k+1) & s^2(N_k+1) & ... & s^n(N_k+1) \\ ... & ... & ... & ... & ... \\ 1 & s(N_{k+1}) & s^2(N_{k+1}) & ... & s^n(N_{k+1}) \end{bmatrix} \tag{24}$$

$$\boldsymbol{X}_1 = \left[\ a_{k0} \quad a_{k1} \quad a_{k2} \quad \cdots \quad a_{kn}\ \right]^T, \tag{25}$$

$$\boldsymbol{X}_2 = \left[\ b_{k0} \quad b_{k1} \quad b_{k2} \quad \cdots \quad b_{kn}\ \right]^T, \tag{26}$$

$$\boldsymbol{B}_1 = \left[\ x(N_k) \quad x(N_k+1) \quad \cdots \quad x(N_{k+1})\ \right]^T, \tag{27}$$

$$\boldsymbol{B}_2 = \left[\ y(N_k) \quad y(N_k+1) \quad \cdots \quad y(N_{k+1})\ \right]^T. \tag{28}$$

To determine the coefficients of the polynomials $\boldsymbol{X}$, it is generally to minimize the following cost function

$$\min \|A\boldsymbol{X} - \boldsymbol{B}\|^2 = \min\left(A\boldsymbol{X} - \boldsymbol{B}\right)^{\mathrm{T}}\left(A\boldsymbol{X} - \boldsymbol{B}\right) \tag{29}$$

Furthermore, to guarantee the motion feasibility and the smooth connection of successive path segments, the initial robot pose and the final one for each path segment L_k are formulated as equality constraints. Specifically, the initial and final position constraints are described as

$$\begin{bmatrix} \boldsymbol{C}_1^{\mathrm{T}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_1^{\mathrm{T}} \\ \boldsymbol{C}_2^{\mathrm{T}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_2^{\mathrm{T}} \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} x(N_k) \\ y(N_k) \\ x(N_{k+1}) \\ y(N_{k+1}) \end{bmatrix} \tag{30}$$

with vectors $\boldsymbol{C}_1,\ \boldsymbol{C}_2 \in \mathbb{R}^{n+1}$ being

$$\boldsymbol{C}_1 = \left[\ 1 \quad s(N_k) \quad s^2(N_k) \quad \cdots \quad s^n(N_k)\ \right]^{\mathrm{T}} \tag{31}$$

$$\boldsymbol{C}_2 = \left[\ 1 \quad s(N_{k+1}) \quad s^2(N_{k+1}) \quad \cdots \quad s^n(N_{k+1})\ \right]^{\mathrm{T}} \tag{32}$$

To depict the equality constraints for the initial robot orientation, we have

$$\frac{\partial x(s)}{\partial s}|_{s=s(N_k)} \sin\theta_{cp}(k) - \frac{\partial y(s)}{\partial s}|_{s=s(N_k)} \cos\theta_{cp}(k) = 0 \tag{33}$$

It follows from the polynomial expressions (18) and (19) that $\frac{\partial x(s)}{\partial s}|_{s=s(N_k)}$ and $\frac{\partial y(s)}{\partial s}|_{s=s(N_k)}$ can be written in a linear form with respect to the coefficients $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ as

$$\frac{\partial x(s)}{\partial s}|_{s=s(N_k)} = \left[\boldsymbol{C}_{d1}^{\mathrm{T}} \quad \boldsymbol{0}\right]\boldsymbol{X}$$

$$\frac{\partial y(s)}{\partial s}|_{s=s(N_k)} = \left[\boldsymbol{0} \quad \boldsymbol{C}_{d1}^{\mathrm{T}}\right]\boldsymbol{X}.$$

with $\boldsymbol{C}_{d1} \in \mathbb{R}^{n+1}$ being

$$\boldsymbol{C}_{d1} = \left[0 \quad 1 \quad 2s(N_k) \quad \cdots \quad ns^{n-1}(N_k)\right]^{\mathrm{T}} \tag{34}$$

Therefore, the initial orientation constraint (33) can be written in a linear form with respect to $\boldsymbol{X}$ as

$$\left[\ \sin\theta_{cp}(k)\boldsymbol{C}_{d1}^{\mathrm{T}} \quad -\cos\theta_{cp}(k)\boldsymbol{C}_{d1}^{\mathrm{T}}\ \right]\boldsymbol{X} = 0 \tag{35}$$

In a similarly way, we can obtain the linear equality constraint for the final orientation of the path segment L_k as:

$$\left[\ \sin\theta_{cp}(k+1)\boldsymbol{C}_{d2}^{\mathrm{T}} \quad -\cos\theta_{cp}(k+1)\boldsymbol{C}_{d2}^{\mathrm{T}}\ \right]\boldsymbol{X} = 0 \tag{36}$$

with $\boldsymbol{C}_{d2} \in \mathbb{R}^{n+1}$ being

$$\boldsymbol{C}_{d2} = \left[0 \quad 1 \quad 2s(N_{k+1}) \quad \cdots \quad ns^{n-1}(N_{k+1})\right]^{\mathrm{T}} \tag{37}$$

To summarize, the objective function (29), together with the linear equality constraints for the initial/final position (30) and orientation (35), (36), constitute a constrained linear least-square problem for approximation of the path segment L_k as

$$\min\left(A\boldsymbol{X}-\boldsymbol{B}\right)^{\mathrm{T}}\left(A\boldsymbol{X}-\boldsymbol{B}\right) \tag{38}$$

subject to

$$\begin{bmatrix} \boldsymbol{C}_1^{\mathrm{T}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_1^{\mathrm{T}} \\ \boldsymbol{C}_2^{\mathrm{T}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_2^{\mathrm{T}} \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} x(N_k) \\ y(N_k) \\ x(N_{k+1}) \\ y(N_{k+1}) \end{bmatrix} \tag{39}$$

$$\begin{bmatrix} \sin\theta_{cp}(k)\boldsymbol{C}_{d1}^{\mathrm{T}} & -\cos\theta_{cp}(k)\boldsymbol{C}_{d1}^{\mathrm{T}} \end{bmatrix}\boldsymbol{X}=0 \tag{40}$$

$$\begin{bmatrix} \sin\theta_{cp}(k+1)\boldsymbol{C}_{d2}^{\mathrm{T}} & -\cos\theta_{cp}(k+1)\boldsymbol{C}_{d2}^{\mathrm{T}} \end{bmatrix}\boldsymbol{X}=0 \tag{41}$$

For this constrained linear least-square problem, standard equality constrained quadratic programming can be utilized to obtain the optimal solution of the polynomial coefficients $\boldsymbol{X}$ by transforming the original objective function into an equivalent standard quadratic form as

$$\min\left(\boldsymbol{X}^T A^T A\boldsymbol{X} - 2\boldsymbol{B}^T A\boldsymbol{X}\right) \tag{42}$$

with the equality constraints (39), (40) and (41).

V. Optimal Velocity Planning Along the Pre-Computed Explicit Path

Along the computed explicit analytical paths, many velocity scheduling algorithms, usually termed as "path-constrained trajectory planning", have been reported in the literature [18], [19]. Note that for the unicycle mobile robots, the time-optimal solution under kinematic constraints has already been obtained by Mišel Brezak and Ivan Petrović in [18], which is directly applied in our framework.

VI. Simulation Results

To verify the effectiveness of the proposed approach, we present some comparative simulation results.

In the simulation, the feedback stabilizer is chosen as the polar coordinate-based one as described in section III, the control gains are selected as $k_1 = 3.1,\ k_2 = 6.9,\ k_3 = 1.1$, while the velocity and acceleration limits are set as $v_{\max} = 1$ m/s, $w_{\max} = 1$ rad/s, $a_{v\max} = 0.5$ m/s^2, $a_{wmax} = 1$ rad/s^2. Since the robot should stop at the cusp points leading to possible low efficiency, thus we utilize the path generated from the initial pose symmetric to the origin if the symmetric path presents no cusp points.

A. Polynomial order selection & fitting precision

Though the fitting precision has no impact on the feasibility of the fitting path due to the linear equality constraints, it is generally preferred to achieve relative good precision, which can depict the original feedback stabilizer-based path behavior in a better way. In general, higher order polynomial can achieve higher fitting precision of the discrete path generated by the feedback-stabilizer, however, it is undesirable to use a very high order for the sake of simplicity and numerical stability. To chose a proper order of the polynomial, we can use the residual to evaluate the fitting precision. Hence, we

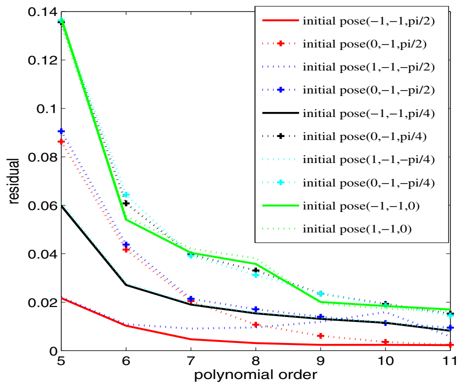

Fig. 1. Fitting precision with respect to polynomial order

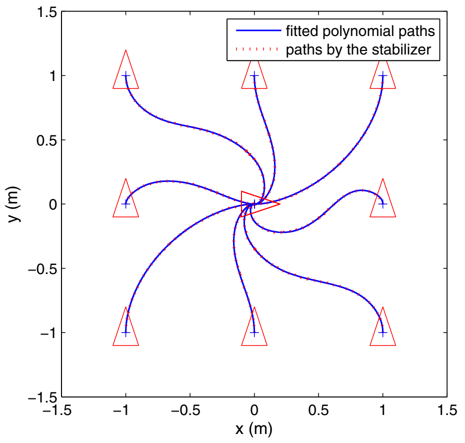

Fig. 2. The feedback stabilizer-based path (solid lines) and the fitted path using polynomials (dotted lines) with different initial configurations

choose a lot of different initial poses, and then we compute the residuals for different orders of polynomials to find a tradeoff. In Fig. 1, we display the results for 10 initial poses, and it is shown that after an order of 10, the decreasing of the residual becomes not evident. Therefore, we choose the used polynomial order be 10. In this case, the fitting results for different initial configurations are shown in Figure 2, from which we can see that the fitting precision is high and thus the analytical 10-order polynomials are acceptable for path representation.

B. Comparative simulation results

1) Comparison between the original feedback stabilizer and the proposed approach: Figure 3 and Figure 4 display the practical velocity limit curves (dashed lines) computed based on the linear/angular velocity/acceleration constraints, the velocity by the original feedback stabilizer (dotted-dashed lines) and the velocity by the proposed approach (solid lines). The initial pose is selected as $(1\text{ m}, -1\text{ m}, \pi/2\text{ rad})$ in this example. It is evident that the original feedback stabilizer lead to too large velocity controls to be realizable since it is greater than the allowable computed velocity limits, while the proposed approach generates realizable velocity signals that are very close to the computed velocity curve to simultaneously achieve high efficiency.

2) Comparison between a classical path planning approach and the proposed approach: a classical local path

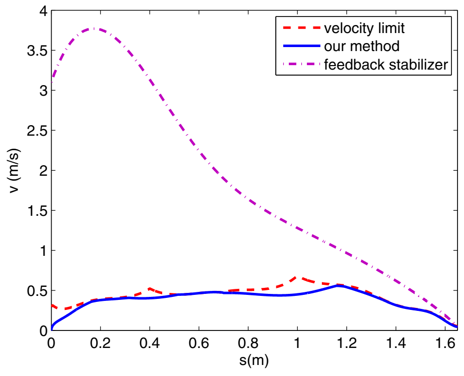

Fig. 3. Comparative results for linear velocity

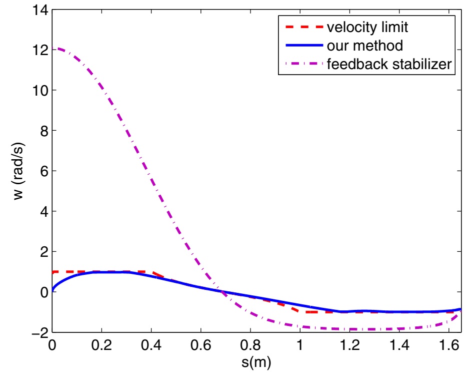

Fig. 4. Comparative results for angular velocity

TABLE I

COMPARISON OF THE ARRIVAL TIME

Initial poses (m, m, rad)	proposed approach (sec)	3-order polynomial (sec)
(1, 0, $\pi/2$)	8.63	6.68
(1, 1, $\pi/2$)	3.31	9.92
(0, 1, $\pi/2$)	4.03	6.67
(-1, 1, $\pi/2$)	6.09	6.71
(1, 0, 0)	2.71	3.25
(1, 1, 0)	4.03	6.77
(0, 1, 0)	6.03	6.64
(-1, 1, 0)	4.01	6.55

planning method for mobile robot is to use the three-order interpolation polynomial as stated in section 11.5.3 of the book [9], and the path parameters are set as $k_i = k_f = 1.5$. For an equal comparison, the subsequent path-constrained trajectory planning for both approaches are conducted with the same velocity planning method as stated in [18]. Table I presents the comparison results for the arrival time, and it is seen that the proposed approach is generally more efficient than the traditional one in most cases.

VII. CONCLUSION

This paper propose a general approach to transform feedback stabilizers into feasible and highly efficient trajectory planners by using numerical path generation, quadratic programming-based polynomial fitting and optimal velocity planning techniques. By using the proposed method, practical velocity/acceleration constraints can be taken into account to yield feasible trajectories. Extensive simulation results verify the effectiveness of the presented method.

REFERENCES

[1] R. W. Brockett, “Asymptotic stability and feedback stabilization,” in Differential Geometric Control Theory, R. W. Brockett, R. S. Millman, and H. J. Sussmann, Eds. Boston: Birkhauser, 1983, pp. 181-191.

[2] A. De Luca, G. Oriolo, C. Samson, “Feedback control of a nonholonomic car-like robot,” in Robot Motion Planning and Control, J.-P. Laumond, Ed., LNCIS, vol. 229, pp. 171-253, Springer, 1998

[3] G. Oriolo, A. De Luca, M. Vendittelli, “WMR control via dynamic feedback linearization: Design, implementation and experimental validation,” *IEEE Trans. on Control Systems Technology*, vol. 10, no. 6, pp. 835-852, 2002

[4] A. Astolfi, “Discontinuous control of nonholonomic systems,” *Systems & Control Letters*, vol. 27, no. 1, pp. 37-45, 1996.

[5] Y. P. Tian, S. Li, “Exponential stabilization of nonholonomic dynamic systems by smooth time-varying control,” *Automatica*, vol. 38, no. 7, pp. 1139-1146, 2002.

[6] Z. P. Jiang, “Iterative design of time varying stabilizers for multi-input systems in chained form,” *Systems & Control Letters*, vol. 28, no. 5, pp. 255-262, 1996.

[7] X. Zhang, Y. Fang, Y. Zhang. “Discrete-time control of chained nonholonomic systems,” *IET Control Theory and Applications*, vol. 5, no. 4, pp. 640-646, 2011.

[8] X. Zhang, Y. Fang, X. Liu, “Motion-estimation-based visual servoing of nonholonomic mobile robots,” *IEEE Trans. on Robotics*, vol. 27, no. 6, pp. 1167-1175, 2011.

[9] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modelling, Planning and Control, Springer, 2009.

[10] P. Ogren, and N. Leonard, “A convergent dynamic window approach to obstacle avoidance,” *IEEE Trans. on Robotics*, vol. 21, no. 2, pp. 188-195, 2005.

[11] M. Seder, I. Petrovic, “Dynamic Window Based Approach to Mobile robot Motion Control in the Presence of Moving Obstacles,” *Proc. of IEEE International Conference on Mobile robotics and Automation*, pp. 1986-1992, April 2007.

[12] K. Goto, K. Kon, F. Matsuno, “Motion Planning of an Autonomous Mobile Robot Considering Regions with Velocity Constraint,” *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, pp. 3269-3274, 2010.

[13] D. Kiss, G. Tevesz, “A receding horizon control approach to obstacle avoidance,” *IEEE International Symposium on Applied Computational Intelligence and Informatics*, pp. 397-402, 2011.

[14] N. Faiz, S. Agrawal, R. Murray. “Trajectory planning of differentially flat systems with dynamics and inequalities,” *AIAA Journal of Guidance, Control and Dynamics*, 2001, 24(2): 219-227.

[15] J. Huang, C. Wen, W. Wang and Z. P. Jiang, “Adaptive stabilization and tracking control of a nonholonomic mobile robot with input saturation and disturbance,” *Systems & Control Letters*, vol. 62, no. 3, pp. 234-241, 2013.

[16] Z. P. Jiang, E. Lefeber and H. Nijmeijer,“Saturated stabilization and tracking of a nonholonomic mobile robot,” *System & Control Letters*, vol.42, no.5, pp.327-332, 2001.

[17] V. Sankaranarayanan, A. D. Mahindrakar, “Configuration constrained stabilization of a wheeled mobile robot-theory and experiment,” *IEEE Trans. Control System Technology*, vol. 21, no. 1, pp. 275-280, 2013.

[18] M. Brezak and I. Petrović, “Time-optimal trajectory planning along predefined path for mobile robots with velocity and acceleration constraints,” *Proc. of IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Hungary, 2011, pp. 942-947.

[19] S. M. LaValle, Planning Algorithms. Cambridge, U.K.: Cambridge University Press, 2006, also available at http://planning.cs.uiuc.edu/

[20] M. Aicardi, G. Casalino, A. Bicchi, A. Balestrino. “Closed loop steering of unicycle like vehicles via lyapunov techniques,” *IEEE Robotics and Automation Magazine*, vol. 2, no. 1, 1995, pp. 27-35.

[21] X. Zhang, Y. Fang, N. Sun, “Visual servoing of mobile robots for posture stabilization: from theory to experiments,” *Int. J. Robust Nonlinear Control,* 2013, online, doi: 10.1002/rnc.3067.