Navigating Discrete Difference Equation Governed WMR by Virtual Linear Leader Guided HMPC

Chao Huang, Xin Chen, Enyi Tang, Mengda He, Lei Bu, Shengchao Qin, Yifeng Zeng

Abstract— **In this paper, we revisit model predictive control (MPC) for the classical wheeled mobile robot (WMR) navigation problem. We prove that the reachable set based hierarchical MPC (HMPC), a state-of-the-art MPC, cannot handle WMR navigation in theory due to the non-existence of non-trivial linear system with an under-approximate reachable set of WMR. Nevertheless, we propose a virtual linear leader guided MPC (VLL-MPC) to enable HMPC structure. Different from current HMPCs, we use a virtual linear system with an under-approximate path set rather than the traditional trace set to guide the WMR. We provide a valid construction of the virtual linear leader. We prove the stability of VLL-MPC, and discuss its complexity. In the experiment, we demonstrate the advantage of VLL-MPC empirically by comparing it with NMPC, LMPC and anytime RRT* in several scenarios.**

I. Introduction

The wheeled mobile robot (WMR) navigation problem considered in this paper is to drive a nonlinear dynamical WMR governed by a **discrete difference equation**, from its initial state to the goal state by **on-line real-timely** tuning its control input, in a **continuous** state and input space. When there is *no* available reference path, a successful navigation must ensure the WMR can converge to the goal state, referred to as *stability*. Meanwhile, the real-time nature requires the computation as promptly as possible, referred to as *efficiency*.

An intuitive solution to the problem is to combine the methods of trace planning and trace tracking as a whole. Unfortunately, the distinctive nature of the problem prevents many mutual techniques from being applied. Due to the continuity of state space, traditional heuristic-search based techniques designed for the planning problem of discrete space, such as A* like methods [1], [2] are not applicable. The off-line planning [3], [4] and tracking techniques [5], [6] cannot satisfy the efficiency requirement enforced by the online planning. Besides, the approaches for systems of ordinary differential equations (ODE) [7] are unable to handle a WMR as the property of universal trace following no longer holds for discrete systems.

Current approaches solve this problem by extending the off-line planning and tracking to the online, which mainly fall into two categories: anytime rapidly exploring random tree (anytime RRT) and model predictive control (MPC). Anytime RRT is the online version of RRT [8], [9] for trace planning. Along with the trace techniques [5], [6], it can be used to solve robot navigation problem. However, regardless of the current result that even the off-line RRT cannot ensure the probabilistic stability for nonlinear robots with common settings [10], there is currently no theoretical stability guarantee on anytime RRT.

MPC is one of the main stream methodologies in robotics community for the navigation/control problem of various systems [11], [12], [13], [14], [15], [16]. Its main idea is at each instant t, to predict the behavior of a system, over finite future steps, namely, *prediction horizon*, by solving a programming problem based on the dynamic model and collision avoidance specification, and only apply the first predicted action. This procedure will repeat at the next control instant until a certain convergent criteria is met. Basic MPC approach for WMR is to directly encode all the nonlinear constraints (NMPC) and jointly generate the trace and inputs [17]. Due to the intrinsic complexity of nonlinear programming, NMPC is time consuming.

In recent years, hierarchical MPC (HMPC) as one of the advanced MPC, receives more attention. HMPCs adopt a two-level structure. In the upper level, a simple linear dynamic system is used to help plan the trace by linear MPC, and then the original system tries to follow the trace in the lower level. Such hierarchical treatment has shown a great advantage on efficiency for linear/piecewise affine systems [18], [19], [20]. The key of HMPC is to guarantee the linear system has a "smaller" state reachable set than the original system, such that any trace planned by the linear system is achievable for the original system. It is known as the under-approximation problem [21], [22].

However, in this paper, we show that the traditional *state* reachable-set based HMPCs are inapplicable for WMR navigation due to the nonlinearity introduced by the state variable θ, denoting the direction. On one hand, we prove that there exists one and only one trivial linear system that has an under-approximate reachable set of the WMR, but can only steer in place (see Theorem 1). On the other hand, simply linearizing the WMR dynamic to make classical HMPCs (LMPC) applicable will inevitably introduce approximation error as a disturbance and is unable to ensure stability either.

In order to leverage the high efficiency of HMPC, a novel Virtual Linear Leader guided Model Predictive Control (VLL-MPC) is proposed for WMR navigation. In the problem, the target state consists of both the target position and the target direction. Following the philosophy of successive

Chao Huang, Xin Chen, Enyi Tang and Lei Bu are with the Department of Computer Science and Technology, Nanjing University, Nanjing, Jiangsu, China. chaohuang@seg.nju.edu.cn, {chenxin,eytang,bulei}@nju.edu.cn.

Chao Huang is also with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA.

Mengda He, Shengchao Qin and Yifeng Zeng are with the School of Computing, Teesside University, UK. Shengchao Qin is also with Shenzhen University. {m.he, s.qin, y.zeng}@tees.ac.uk.

Corresponding author: Xin Chen

approaching, it is possible to firstly drive a WMR to a point very near to the target position, and then guide it to the final state by setting the speed and the angular velocity properly (see Proposition 1). As a result, the navigation problem can be safely converted into the problem of finding a **path** convergent to the final position. The advantage coming from such conversion is that the direction variable θ can be safely ignored in the planning stage such that it is easy to build a linear system (VLL) under-approximating the reachable set of the WMR. As the key step of VLL-MPC, the construction of the VLL for the dynamic of WMR is elaborated whose validity is strictly proved. Benefiting from the HMPC structure, VLL-MPC's stability and high efficiency can be theoretically assured.

In the paper, we make the following contributions:

- We prove that reachable set based HMPCs cannot handle WMR navigation in theory;
- We propose a novel VLL-MPC to enable the HMPC framework for the WMR navigation and give a construction of the virtual linear leader, which is the core of VLL-MPC;
- We give a sufficient condition to theoretically ensure the stability of VLL-MPC and discuss the complexity;
- We demonstrate the performance of VLL-MPC in several scenarios and show its advantages empirically by a comparison with NMPC, LMPC and anytime RRT*.

II. Problem Formulation

A WMR follows the discrete dynamic model [23], [5]:

$$\begin{aligned} x(t+1) &= x(t) + v(t)\cos\theta(t)\varDelta T \\ y(t+1) &= y(t) + v(t)\sin\theta(t)\varDelta T \\ \theta(t+1) &= \theta(t) + \omega(t)\varDelta T \end{aligned}$$

where x and y denote the position under Cartesian coordinate, θ denote the direction, v and ω are the forward and steering velocity respectively. This model can be also represented in a compact form:

$$q(t+1) = f(q(t), u(t)), \tag{1}$$

where $q = [x, y, \theta]$ is the state variable, $u = [v, \omega]$ is the input variable, t denotes the t-th sampling instant and $\varDelta T$ is the sampling period. Due to the limit of the engine power, the velocity is bounded by the input constraint:

$$u \in U \triangleq |v| \leq V_{\text{bound}}. \tag{2}$$

The WMR's behavior is either defined by a *trace* or a *path* [24], [25]. A trace of *W*, denoted as $\mathrm{tr}_W(q)$, is a finite state sequence $\{q(t)\}_{t=0}^{t_f}$ satisfying Constraint (2), derived from f, while $q(0) = q$ and t_f is an arbitrary natural number. A path $\rho_W(q)$ only considers the position part of a trace, i.e. $\{[x(t), y(t)]\}_{t=0}^{t_f}$. $\mathrm{Tr}_W(q)$ and $\mathrm{P}_W(q)$ denote the set of all traces and paths from q, respectively.

Such a robot needs to navigate in a dynamical environment, including both moving and stationary obstacles. We consider the linear convex space, that is, the overall state space of the environment can be described as a polytope Q:

$$q \in Q \triangleq P \times [-\pi, \pi]. \tag{3}$$

where P represents the position space for $[x, y]$.

We assume that there are n obstacles with linear dynamics, and the state q_i of the i-th obstacles include its position $[x_i, y_i]$:

$$q_i(t+1) = f_i(q_i(t)), \quad i = 1, \cdots, n. \tag{4}$$

Let $q_e = [q_1, \cdots, q_n]$. We use the following compact form to represent the environment dynamic:

$$q_e(t+1) = f_e(q_e(t)); \tag{5}$$

The distance between the WMR and any obstacle should not be over a given d_{safe} to avoid collision:

$$\|[x, y] - [x_i, y_i]\|_\infty \geq d_{\text{safe}}, \quad i = 1, \cdots, n. \tag{6}$$

Subsequently we formulate the WMR navigation problem based on the above model of WMR and environment:

Problem 1 (WMR Navigation)**.** *Given a WMR* W*, where*

- *the dynamic model is Equation* (1) *and its input constraint described as Constraint* (2)*;*
- *an initial state is* $q(0) = [x(0), y(0), \theta(t)]$*;*
- *a goal state is* $q' = [x', y', \theta']$*,*

an environment E*, where*

- *the dynamic model is Equation* (5)*;*
- *an initial state is* $q_e(0)$*;*

the collision avoidance specification described as Constraint (6) *and a convergence error bound* ϵ*, the WMR navigation problem is to find a control strategy to determine a path* $\rho_W(q_0)$ *by on-line generating a control sequence* $\{u(t)\}_{t=0}^{\infty}$ *such that* W *can reach* $B_{q'}(\epsilon)$ *without collision, where* $B_x(r)$ *denotes a ball around* x *with radius* r*.*

Two key properties *stability* [26] and *efficiency* are defined as follows.

Definition 1 (Stability)**.** *A control scheme is* stable *for a WMR navigation problem 1,* ***iff*** *starting from the initial state* $q(0)$*, the path* $\rho_W(q_0)$ *of computed by the control scheme reaches* $B_{q'}(\epsilon)$*.*

Definition 2 (Efficiency)**.** *The efficiency of an on-line control scheme is defined as its* average computation time *at each sampling instant.*

III. Revisit MPCs for WMR Navigation

Let us first revisit MPCs, including NMPC and the state-of-the-art HMPC for WMR navigation from an unified perspective of *reachable set* [27], [28] to motivate our idea.

Definition 3 (Reachable set)**.** *The reachable set of* W *from* q *after the next sampling period is defined as* $\mathrm{Reach}_W(q) = \{q' | \exists u \in U, q' = f(q, u)\}$*. Specially, a set* $\underline{\mathrm{Reach}}_W(q) \subseteq \mathrm{Reach}_W(q)$ *is called an under-approximate reachable set.*

In NMPC, at each sampling instant t, an optimization problem needs to be solved online to predict the behavior of the robot, which involves the robot dynamic and all the relevant constraints. Figure 1 shows an example of how NMPC works at each sampling instant t. In NMPC, the original nonlinear dynamic is directly encoded in the optimization problem, which indicates the exact reachable set (irregular black shape) is implicitly considered. The yielded nonlinear programming makes NMPC time consuming.

Remark 1. *Notably in practice, rather than computing the reachable set, NMPC only finds a (optimal) feasible point,*

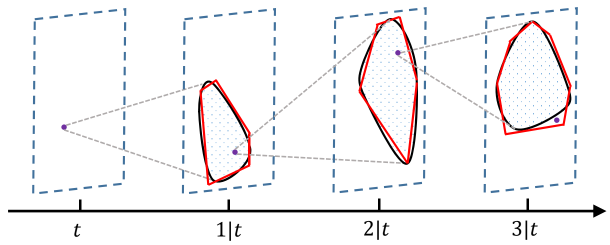


Fig. 1: Methodologies of NMPC and LMPC at sampling instant t: The blue rectangle is the state space. Black and red shape represent the exact and linear-approximate reachable set. The purple points denote the predicted trace in the next three steps. The symbol $i|t, i = 1, 2, 3$ denotes the i-th step predicted at t.

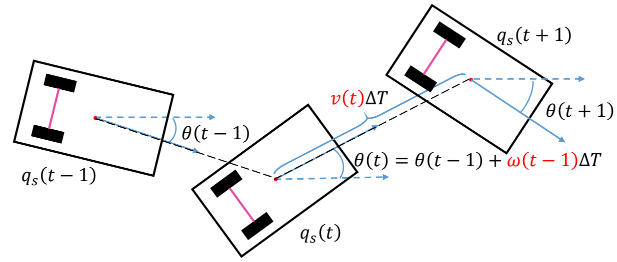


Fig. 2: Relation between the position and input variables: The position of the WMR at $t+1$ is determined by the direction $\theta(t)$ and the linear velocity $v(t)$, while $\theta(t)$ is determined by $\omega(t-1)$. Thus, the position at $t+1$ is determined by $v(t)$ and $\omega(t-1)$.

which has lower computation complexity. Here, we leverage reachable set to provide a mathematical intuition of NMPC. In Section V, we will elaborate the computation complexity.

A naive idea is to explore if we can find a linear system with an under-approximate reachable set of the WMR, such that the WMR can reach every state planned within the under-approximate reachable set. Then such linear system can be used to plan the trace in the upper level by MPC, while the actual input of the WMR is then computed to drive the WMR to follow the trace. This idea is also known as hierarchical MPC (HMPC) and adopted in recent works for linear/piecewise affine systems [18], [19], [20]. Intuitively, the system that only does pivot steering meets such requirement:

$$x(t+1) = x(t), \quad y(t+1) = y(t), \quad \theta(t+1) = A_3\theta(t) + u_3.$$

However, it cannot be used for trace planning and thus we call it *trivial*. Unfortunately, we find that there exists no non-trivial linear under-approximation for WMR:

Theorem 1. *For a WMR with dynamic model* (1) *and input constraint* (2)*, there exists no linear system with an under-approximate reachable set, except for the trivial one.*

Proof. Assume there exists such a system and WLOG, its dynamic is:

$$x(t+1) = A_1x(t) + u_1, \quad y(t+1) = A_2y(t) + u_2,$$
$$\theta(t+1) = A_3\theta(t) + u_3.$$

where A_i and u_i are the coefficient and input for each dimension respectively. Consider the special case $q(t) = [0, 0, 0]$. Recall the dynamic of WMR, we know

$$\text{Reach}_W(q(t)) = [-V_{\text{bound}}\Delta T, -V_{\text{bound}}\Delta T] \times \{0\} \times [-\pi, \pi].$$

To meet the under-approximation requirement, we have

$$0 = y(t+1) = A_2y(t) + u_2 = u_2$$

Similarly, when $q(t) = [0, 0, \pi/2]$, we can obtain that $u_1 = 0$. Thus this system is trivial. ☐

We may simply linearize the nonlinear WMR dynamic and use the linear dynamic for planning to enable HMPC structure, ignoring the requirement of under-approximation. It will lead to the linear approximate reachable set with the linearization error (red polytope). To differ it from the typical HMPC, we name it LMPC. However, such linearization is precise only locally around the given point. Since one cannot "predict" the linearization of the system at future steps, the linearization error will inevitably accumulate over time and makes the robot system unpredictable (See Figure 1).

IV. VLL-MPC

Our idea is based on an interesting observation that once WMR's position is very closed to the goal position, we can always let the overall state also close to the goal state:

Proposition 1. *If $\|[x(t), y(t)] - [x', y']\| \leq \epsilon$, then let $v(t) = 0, \omega(t) = (\theta' - \theta(t))/\Delta T$, we have $\|q(t+1) - q'\| \leq \epsilon$.*

Thus the original problem can be converted into determining a convergent path.

Based on Proposition 1, we propose a virtual linear leader based model predictive control (VLL-MPC) scheme to enable HMPC structure. We construct a virtual robot R with state variable q_R, which has linear dynamic and the smaller **path set** from any state q than the WMR W, that is $\mathrm{P}_R(q) \subseteq \mathrm{P}_W(q)$. We refer to R as a *virtual linear leader*. Then following the HMPC manner, we online plan the path based on the dynamic of R by linear MPC. With a larger path set, the WMR is able to strictly follow the planned path. Thus, we can make W converge to the goal state q' by letting R converge to $q'_R = [x', y']$. In the following, we first introduce the construction of the virtual linear leader. Then the detailed VLL-MPC will be given.

A. Construct a Virtual Linear Leader

Observing the dynamic of W as Equation (1), we can find that the position at instant $t+1$ is determined by the forward velocity $v(t)$ and steering velocity $\omega(t-1)$ (See Figure 2). It motivates us to construct a virtual linear leader R as follows:

$$x(t+1) = x(t) + v_1(t)\Delta T, \quad y(t+1) = y(t) + v_2(t)\Delta T,$$

or in a compact form:

$$q_R(t+1) = f_R(q_R(t), u_R(t)), \tag{7}$$

where $q_R = [x, y]$ is the state, $u_R = [v_1, v_2]$ is the input. The input constraint is:

$$|v_1|, |v_2| \leq \frac{\sqrt{2}}{2}V_{\text{bound}}, \quad v_1(0) = v_2(0) = 0. \tag{8}$$

Now we point out the validity of our construction R.

Theorem 2. *For a WMR W with the dynamic model* (1) *and the input constraint* (2)*, and the virtual linear leader R with the dynamic model* (7) *and the input constraint* (8)*, $\forall q(0) = [x(0), y(0), \theta(0)] \in Q$, let $q_R(0) = [x(0), y(0)]$, then we have $\mathrm{P}_R(q_R(0)) \subseteq \mathrm{P}_W(q(0))$.*

Proof. Given any path

$$\rho_R(q_R(0)) = \{[x(0), y(0)], \cdots, [x(t'), y(t')]\} \in \mathrm{P}_R(q(0)),$$

where t' is the terminal instant. Let

$$\Delta x(t) \triangleq x(t+1) - x(t), \quad \Delta y(t) \triangleq y(t+1) - y(t),$$
$$\Delta D(t) \triangleq \sqrt{(\Delta x(t))^2 + (\Delta y(t))^2},$$
$$\gamma(t+1) = \arcsin \frac{\Delta y(t+1)}{\Delta D(t+1)}, \quad \text{if } \Delta D(t+1) \neq 0.$$

We construct the input sequence of W as

$$\mu = [v(0), \omega(0)], \cdots, [v(t'-1), \omega(t'-1)],$$

where

$$v(t) = \Delta D(t)/\Delta T, \quad t = 0, \cdots, t'-1$$

$$\omega(t) = \begin{cases} 0, & \Delta D(t+1) = 0, t < t'-1 \\ (\gamma(t+1) - \theta(t))/\Delta T, & \Delta x(t+1) \geq 0, t < t'-1 \\ (\pi - \gamma(t+1) - \theta(t))/\Delta T, & \Delta x(t+1) < 0, t < t'-1 \\ (\theta' - \theta(t'-1))/\Delta T, & t = t'-1 \end{cases} \tag{9}$$

It is easy to check that $v(t)$ is valid in terms of Constraint (2) and the path $\rho_W(q(0))$ of W generated by μ equals $\rho_R(q_R(0))$. ☐

Remark 2. *Note that $\omega(t'-1)$ does not affect the establishment of Theorem 2, thus can be arbitrary. However, when $q_R(t')$ approaches the goal position, according to Proposition 1, our assignment of $\omega(t'-1)$ can let $\theta(t')$ be the goal direction.*

B. Control Scheme

As indicated by Figure 2 and Equation (9), W's input at t is computed based on the states of R at both $t+1$ and $t+2$, thus we need to compute the planned path *two steps* ahead, which is significantly different from the current HMPC.

In detail, at each sampling instant t, the WMR W knows its current state $q(t)$, while the virtual linear robot R knows its state $q_R(t+1)$. the control scheme consists of two successive steps:

Compute the planned state at $t+2$: The controller tries to plan a goal state $q_R(t+2)$ for R at $t+2$ by linear MPC:

$$\min_{\tilde{q}_R(H|t+1)} \sum_{k=0}^{H-1} \|q_R(k|t+1) - q'_R\|_\infty + l_H(q_R(H|t+1))$$
$$\text{s.t.} \begin{cases} q_R(k|t+1) = f_R(q_R(k-1|t+1), u_R(k-1|t+1)), \ 1 \leq k \leq H, \\ |u_R(k-1|t+1)| \leq [\frac{\sqrt{2}}{2} V_{\text{bound}} \Delta T, \frac{\sqrt{2}}{2} V_{\text{bound}} \Delta T], \ 1 \leq k \leq H, \\ q_R(k|t+1) \in Q, \ \ 1 \leq k \leq H, \\ q_e(t+k+1) = f_e(q_e(t+k)), \ \ 1 \leq k \leq H, \\ \|q_R(k|t+1) - q_i(k|t+1)\|_\infty \geq d_{\text{safe}}, \ 1 \leq i \leq n, \ 1 \leq k \leq H, \\ q_R(H|t+1) \in \mathcal{P}_f, \quad q_R(0|t+1) = q_R(t+1). \end{cases} \tag{10}$$

where H is the prediction horizon, the notation $(\cdot)(k|t+1)$ denotes the *predictive* value at the *(t+1+k)*-th collaboration instant computed at time $t+1$, $\tilde{q}(H|t+1) \triangleq q_R(1|t+1), \cdots, q_R(H|t+1)$ denotes the decision variables, and $l_H(q_R(H|t+1)) \triangleq c\,\|q_R(H|t+1) - q'\|_\infty$ is the terminal cost and c is an user-defined coefficient. $\mathcal{P}_f$ is the terminal constraint set. l_H and $\mathcal{P}_f$ are both used to ensure stability. Let $q^*_R(1|t+1), ..., q^*_R(H|t+1)$ denote the optimal solution. The first sample $q^*_R(1|t+1)$ will be used as the desired state $q_R(t+2)$ for R at the instant $t+2$:

$$q_R(t+2) = q^*_R(1|t+1). \tag{11}$$

The VLL R then moves to $q_R(t+2)$.

Compute the input: The controller then derives the input of the WMR W based on the state $q_R(t+1)$ and $q_R(t+2)$ of R. That is, compute $u(t) = [v(t), \omega(t)]$ according to the state $q_R(t+1)$ and $q_R(t+2)$ by (9) and apply it on W by (1) such that W reaches $q(t+1)$, where $[x(t+1), y(t+1)] = q_R(t+1)$.

This procedure will be repeated at the next sampling instant $t+1$ based on until the $q(t)$ meets the convergence criteria ϵ.

Remark 3. *Note that an obstacle here can be either uncontrollable (e.g. a building) or controllable (e.g. a robot with linear dynamics). Specially, when handling a controllable robot obstacle, our approach can be naturally integrated with existing distributed MPC schemes [18], [33].*

V. Algorithm Analysis

[Stability] Theorem 2 ensures that the path planned based on R can be achieved by W. Thus the stability of VLL-MPC is determined by the linear MPC law (10), (11).

Lemma 1. *VLL-MPC scheme stabilizes the WMR W* ***iff*** *the MPC law* (10), (11) *stabilizes the virtual linear leader R.*

In the theory of linear MPC, the terminal constraint set and cost function methods are widely used to ensure the stability by properly choosing its *stable parameters*, a terminal cost l_H, a terminal constraint set $\mathcal{P}_f$ and a controller function $\kappa(\cdot)$ [26], [29], [30]. We adopt the strategy of parameter configuration in [29] and give the following theorem:

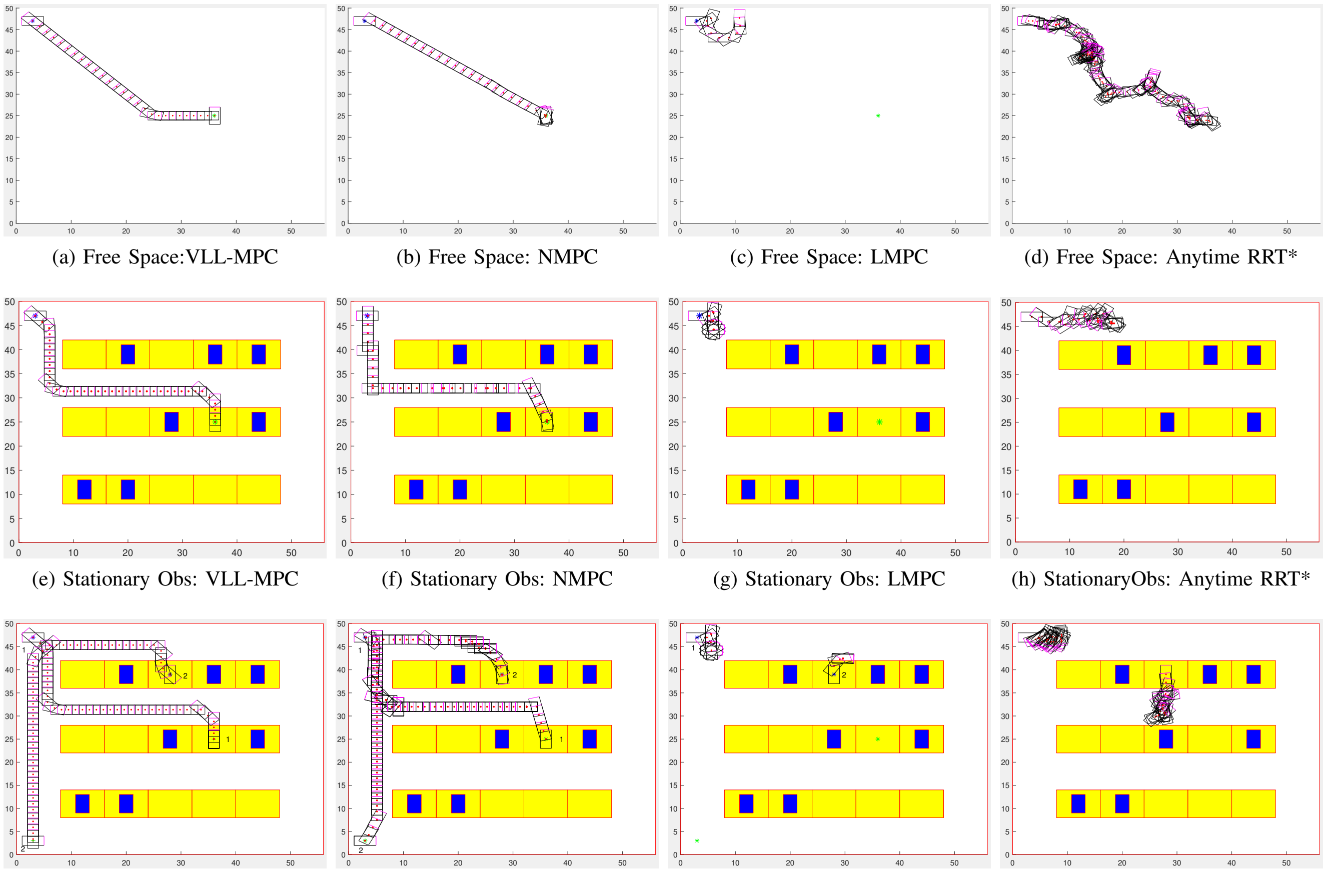
Theorem 3. *$\forall H \in \mathbb{N}^+$, $c \in \mathbb{R}^+$, such that $\mathcal{P}_f = \{q'_R\}$, then the VLL-MPC scheme stabilizes the WMR W.*

Proof. The MPC law (10), (11) stabilizes the virtual linear leader R with this parameter configuration [29]. According to Lemma 1, it is equivalent to that VLL-MPC scheme stabilizes the WMR W. ☐

[Complexity] Observe the control scheme of VLL-MPC. The overall computation consists of solving the optimization problem (10) and computing the input (9). It is obvious that Equation (9) is an elementary function with fixed number of operations, which means that computing Equation (9) takes constant time. Thus, the time complexity of overall VLL-MPC scheme is determined by the optimization (10). Based on the current works, we show the advantage of VLL-MPC on efficiency compared with NMPC in theory by the following conclusion on complexity.

Theorem 4. *NMPC is NP-hard. VLL-MPC is NP-complete in general. Specially, VLL-MPC is of polynomial time if no obstacle exists.*

Proof. NMPC needs to solve a nonlinear programming (NLP). It has been shown in [31] that solving a NLP with quadratic objective and non-convex constraints is NP-hard. In general, the optimization problem (10) is a mixed integer programming (MILP), which is NP-complete. If there is no obstacle, (10) degenerates to a linear programming, which is of polynomial time. ☐



(a) Free Space:VLL-MPC (b) Free Space: NMPC (c) Free Space: LMPC (d) Free Space: Anytime RRT*

(e) Stationary Obs: VLL-MPC (f) Stationary Obs: NMPC (g) Stationary Obs: LMPC (h) StationaryObs: Anytime RRT*

(i) Moving Obs: VLL-MPC (j) Moving Obs: NMPC (k) Moving Obs: LMPC (l) Moving Obs: Anytime RRT*

Fig. 3: Robot traces in different scenarios

[Sub-optimality] Note that in Equation (10), we restrict the behavior of W to a linear dynamic system with under-approximate path set. Thus, the obtained control scheme is only optimal with respect to the linear model, but sub-optimal for the original nonlinear model. We believe it is a fair cost of the efficiency improvement and maintaining the stability.

VI. Simulation

We compare our VLL-MPC method with state-of-the-art navigation control algorithms, including NMPC [32], LMPC [23] and anytime RRT* [8]. As mentioned previously, NMPC has theoretical guarantee on the stability under certain pre-conditions [26], LMPC cannot ensure stability due to the introduction of additional disturbance (approximation error), while anytime RRT* no longer maintains the probabilistic stability. We empirically demonstrate it by simulation.

A. Experimental Setup

We consider a practical scenario, autonomous parking scenario, where the parameters of WMR are set as $V_{\text{bound}} = 2, \Delta T = 1$. We carefully choose three typical scenarios, which are named *Free Space* scenario, *Stationary Obstacle* scenario, and *Moving Obstacle* scenario, to demonstrate the pros and cons of each approach. Details on scenario setting are described as follows:

[Free Space Scenario]: Free Space scenario is the simplest case, where there is no constraint on robot position and no obstacle in the environment. The WMR can move freely on the plane. The initial state of the robot is $[3, 47, 0]$ and the goal state is $[36, 25, 1.5\pi]$. We set the prediction horizon H=30, c=1. With respect to anytime RRT*, we allow it to execute no longer than 10 seconds at each sampling instant.

[Stationary Obstacle Scenario]: Stationary Obstacle scenario is more close to a real parking lot than Free Space scenario, with constraints on robot positions and stationary obstacles in the environment. The whole parking lot is a $[0, 56] \times [0, 50]$ rectangle area. There are 15 parking spaces in the parking lot, which arrange into three rows. Some of the parking spaces have already been occupied by WMRs. The concerned WMR needs to reach the goal parking space from the entrance without entering any other parking space or going out of the parking lot area. The initial state and the goal state are the same as in Free Space Scenario. We set the prediction horizon H=30, c=1. Considering the all the possible situations, we conservatively set $d_{\text{safe}} = 6.25$. The maximal execution time for anytime RRT* is 20 seconds.

[Moving Obstacle Scenario]: Moving Obstacle scenario is further close to a real parking lot than Stationary Obstacle one. There are still position constraints and stationary obstacles. However, except for the WMR 1 we concern, there is another WMR, WMR 2, in the environment. WMR 1 needs to reach the goal parking space as in Stationary Obstacle scenario, while WMR 2 wants to achieve that state $[3, 3, \pi]$, denoting the exit, from the current parking space $[28, 39, 1.5\pi]$. One WMR should avoid to collide with another during the process. In this case, a robot needs to act by guessing another robot's behavior. As mentioned in Remark 3, we therefore integrate each MPC with a standard distributed control method proposed in [33] to tackle it. We set H=60, c=1, $d_{\text{safe}} = 6.25$. The maximal execution time for anytime RRT* in this scenario is 60 seconds.

All the simulations were performed on a computer equipped with 8 GB RAM and an Intel Core i5 4570 CPU under Windows. The LP and MILP problems were solved by the Gurobi solver [34]. NLP problems encountered in the NMPC were solved by Tomlab toolbox [35].

B. Result Analysis

The traces computed by various algorithms are shown in Fig 3. In all the sub-figures, a WMR is drawn as a rectangle, where the short purple edge denotes the back and the red point denotes the center. The blue and green *'s denote the initial position and the goal positions respectively. The white space denotes the permitted moving area for WMRs. In the scenarios with obstacles (Fig 3e-3l), the parking lot is drawn as a large rectangle with red edges, the yellow areas denote the parking spaces and the blue rectangles denote the stationary WMRs. Observing Fig 3a, 3e, 3i, VLL-MPC completed the navigation task in all the scenarios.

NMPC also finished the simulation (see Fig 3b, 3f, 3j). It is worthy noting robots moved slowly at some points in Moving Obstacle scenario. The reason is that the performance of non-linear solvers relies heavily on the initial guess of decision variables while solving NLP problems. In complicated cases, it is difficult to find an appropriate initial value. When the default initial value cannot lead to a solution satisfying the sufficient condition of stability, we will randomly pick an initial value in the neighborhood of the current state.

LMPC did not finish any scenario (See Fig 3c, 3g, 3k). As mentioned in Section III, linearization is precise only locally around the given equilibrium point. Since one cannot "predict" the linearization of the system at future steps, the linearization error will will inevitably accumulate over time and make the robot dynamic uncontrollable. As a result, the robot was going to leave the default observable area in Figure 3c, while in Figure 3g and 3k, the WMR just moved chaotically around the initial position.

Notably there exists randomness in RRT-based methods, therefore we ran Anytime RRT* 10 times for each scenario, where Fig 3d, 3h, 3l show the best results. Anytime RRT* succeeded in the Free Space scenario but failed in other two. More specifically, the robot stuck for a long time from a certain instant in the Stationary and Moving Obstacle scenarios. In fact, there is currently no explicit theoretical result on the stability of anytime version of RRT or RRT*.

TABLE I: Comparison on Effiency (Average Computation Time at Each Instant in Seconds)

	Free	StationaryObs	MovingObs
VLL-MPC	0.165	0.768	2.310
NMPC	1.463	36.64	131.5
LMPC	0.129	0.671	2.563
Anytime RRT*[1]	10.00	20.00	60.00

[1] Considering the characteristics of sampling based algorithms, we give a computation time bound for the anytime RRT* in each scenario. Such a setting allows the anytime RRT* to sample at least 800, 1500, 4200 points each time on our platform in Free space, Stationary Obstacle and Moving Obstacle scenarios respectively.

In [8], only the situations with the pre-specified reference trace were discussed in the experiments. A possible reason is that at each instant, the number of sampling is small due to the limited computation time and results at previous instants are dropped, which makes the sampling set not dense in the state space and probabilistic stability no longer hold. Even though RRT based approaches show the great advantage in off-line motion planning, it still needs more effort to ensure the stability for on-line planning.

Table I shows the efficiency, i.e. the average computation time at each instant, of all the algorithms in terms of different scenarios. In the following, we explain the table in detail.

- Compare with NMPC: VLL-MPC achieved at least 88.7% improvement, and the computation time of NMPC grew rapidly with the complexity of scenarios. The result fitted well our theoretical conclusion in Section V.
- Compare with LMPC: Due to that the optimization problems involved in our VLL-MPC and LMPC are both linear programming and of the same scale, it is not surprised that they share the similar efficiency.
- Compare with Anytime RRT*: Anytime RRT* has a high chance to fail the experiment, even though it is 1 order-of-magnitude longer than the time required by VLL-MPC. It is worthy noting that we may let it stable by specifying a sufficient large time bound. However, such a setting makes anytime RRT* degenerate to the traditional off-line version.

VII. Conclusion

In this paper, we show reachable set based HMPC, one of the advanced MPC techniques for robot control, cannot handle WMR navigation. Then we propose VLL-MPC to enable HMPC structure to improve the efficiency compared with NMPC while maintaining stability. Future work includes the integrating VLL-MPC with RRT for off-line path planning.

Acknowledgement

This research was supported by the National Key Research and Development Program of China (No.2017YFA0700604), and the National Natural Science Foundation of China (No.61690204, No.61772347, No.61836005, No.61572249, No.61772260). Dr. Yifeng Zeng is also supported by the EPSRC Grant (EP/S011609/1).

REFERENCES

[1] G. Wagner and H. Choset, "M*: A complete multirobot path planning algorithm with performance bounds," in *IROS*. IEEE, 2011, Conference Proceedings, pp. 3260–3267.

[2] W. Zeng and R. L. Church, "Finding shortest paths on real road networks: the case for a*," *International journal of geographical information science*, vol. 23, no. 4, pp. 531–543, 2009.

[3] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[4] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Rob. Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[5] Z. Li, J. Deng, R. Lu, Y. Xu, J. Bai, and C. Y. Su, "Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach," *IEEE Transactions on SMC: Systems*, vol. 46, no. 6, pp. 740–749, 2016.

[6] C. J. Ostafew, A. P. Schoellig, T. D. Barfoot, and J. Collier, "Learning-based nonlinear model predictive control to improve vision-based mobile robot path tracking," *Journal of Field Robotics*, vol. 33, no. 1, pp. 133–152, 2016. [Online]. Available: http://dx.doi.org/10.1002/rob.21587

[7] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[8] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the rrt," in *ICRA*. IEEE, 2011, Conference Proceedings, pp. 1478–1483.

[9] J. D. Hernández, E. Vidal, G. Vallicrosa, E. Galceran, and M. Carreras, "Online path planning for autonomous underwater vehicles in unknown environments," in *ICRA*. IEEE, 2015, Conference Proceedings, pp. 1152–1157.

[10] T. Kunz and M. Stilman, "Kinodynamic rrts with fixed time step and best-input extension are not probabilistically complete," in *Algorithmic foundations of robotics XI*. Springer, 2015, pp. 233–244.

[11] M. Ramirez, M. Papasimeon, L. Benke, N. Lipovetzky, T. Miller, and A. R. Pearce, "Real–time uav maneuvering via automated planning in simulations," in *IJCAI*. AAAI Press, 2017, Conference Proceedings, pp. 5243–5245.

[12] E. Fernández-González, E. Karpas, and B. C. Williams, "Mixed discrete-continuous heuristic generative planning based on flow tubes," in *IJCAI*, 2015, Conference Proceedings, pp. 1565–1572.

[13] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1714–1721.

[14] F. R. Hogan, E. R. Grau, and A. Rodriguez, "Reactive planar manipulation with convex hybrid mpc," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 247–253.

[15] A. Tanguy, D. De Simone, A. I. Comport, G. Oriolo, and A. Kheddar, "Closed-loop mpc with dense visual slam-stability through reactive stepping," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 1397–1403.

[16] I. B. Hagen, D. K. M. Kufoalor, E. F. Brekke, and T. A. Johansen, "Mpc-based collision avoidance strategy for existing marine vessel guidance systems," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7618–7623.

[22] E. Goubault, O. Mullier, S. Putot, and M. Kieffer, "Inner approximated reachability analysis," in *hscc*. ACM, 2014, Conference Proceedings, pp. 163–172.

[17] H. A. v. Essen and H. Nijmeijer, "Non-linear model predictive control for constrained mobile robots," in *ECC*, 2001, Conference Proceedings, pp. 1157–1162.

[18] C. Huang, X. Chen, Y. Zhang, S. Qin, Y. Zeng, and X. Li, "Hierarchical model predictive control for multi-robot navigation," in *IJCAI*. AAAI Press, 2016, Conference Proceedings, pp. 3140–3146.

[19] B. Picasso, X. Zhang, and R. Scattolini, "Hierarchical model predictive control of independent systems with joint constraints," *Automatica*, vol. 74, pp. 99–106, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109816302989

[20] C. Huang, X. Chen, Y. Zhang, S. Qin, Y. Zeng, and X. Li, "Switched linear multi-robot navigation using hierarchical model predictive control," in *IJCAI*, 2017, Conference Proceedings, pp. 4331–4337. [Online]. Available: 10.24963/ijcai.2017/605

[21] E. Asarin, O. Bournez, T. Dang, and O. Maler, "Approximate reachability analysis of piecewise-linear dynamical systems," in *HSCC*. Springer, 2000, Conference Proceedings, pp. 20–31.

[23] F. Künhe, J. Gomes, and W. Fetter, "Mobile robot trajectory tracking using model predictive control," in *LARS*, 2005, Conference Proceedings.

[24] S. Mitra, "A verification framework for hybrid systems," Thesis, 2007.

[25] R. Alur, R. Grosu, I. Lee, and O. Sokolsky, "Compositional refinement for hierarchical hybrid systems," in *HSCC*. Springer, 2001, Conference Proceedings, pp. 33–48.

[26] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[27] C. Huang, X. Chen, W. Lin, Z. Yang, and X. Li, "Probabilistic safety verification of stochastic hybrid systems using barrier certificates," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 5s, pp. 1–19, 2017.

[28] Z. Yang, C. Huang, X. Chen, W. Lin, and Z. Liu, "A linear programming relaxation based approach for generating barrier certificates of hybrid systems," in *International Symposium on Formal Methods*. Springer, 2016, pp. 721–738.

[29] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[30] Y. Kuwata and J. P. How, "Stable trajectory design for highly constrained environments using receding horizon control," in *ACC*, vol. 1. IEEE, 2004, Conference Proceedings, pp. 902–907.

[31] R. Hemmecke, M. Köppe, J. Lee, and R. Weismantel, "Nonlinear integer programming," in *50 Years of Integer Programming 1958-2008*. Springer, 2010, pp. 561–618.

[32] A. Boccia, L. Grüne, and K. Worthmann, "Stability and feasibility of state constrained mpc without stabilizing terminal constraints," *Systems & Control Letters*, vol. 72, pp. 14–21, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167691114001595

[33] T. Keviczky, F. Borrelli, and G. J. Balas, "A study on decentralized receding horizon control for decoupled systems," in *ACC*, vol. 6. IEEE, 2004, Conference Proceedings, pp. 4921–4926.

[34] I. Gurobi Optimization, "Gurobi optimizer reference manual," 2016. [Online]. Available: http://www.gurobi.com

[35] K. Holmström and M. M. Edvall, *The tomlab optimization environment*. Springer, 2004, pp. 369–376.