Motion Planning with Lattices

Stefania Pancanti, David Salvadorini, Lucia Pallottino, Antonio Bicchi

***Abstract*— In this paper we propose a new approach to motion planning, based on the introduction of a lattice structure in the workspace of the robot, leading to efficient computations of plans for rather complex vehicles, and allowing for the implementation of optimization procedures in a rather straightforward way. The basic idea is the purposeful restriction of the set of possible input functions to the vehicle to a finite set of symbols, or *control quanta*, which, under suitable conditions, generate a regular lattice of reachable points. Once the lattice is generated and a convenient description computed, standard techniques in integer linear programming can be used to find a plan very efficiently. We also provide a correct and complete algorithm to the problem of finding an optimized plan (with respect e.g. to length minimization) consisting in a sequence of graph searches.**

I. Introduction

In this paper, the problem of steering complex systems (such as wheeled vehicles with an arbitrary number of trailers) among obstacles, is approached. The basic idea is to introduce in the robot's workspace a particular structure, consisting of a lattice, on which computations can be very efficient. This can be obtained in some cases by suitably discretizing the space of acceptable commands to the robot, thus reducing it to a finite set of *control quanta* associated to symbols of an alphabet, and describing robot motions through the generated language.

The use of symbolic languages to plan complex motions of large systems capable of complex behaviours, and to hierarchically abstract levels of decision, planning and supervision, is an approach that has been recently advocated. A framework for describing these systems, Motion Description Languages, has been introduced ([**?**], [**?**], [**?**]), while extensions to systems with symmetries have been presented in [1]. Our ideas can be traced back to [2], although the technique there differed substantially from what presented here.

A lattice Λ is an additive group which can be generated by integer combinations of a finite number of linearly independent vectors. If the m generators h_i are rational n-dimensional vectors (which will always be the case for us), and are arranged as the columns of a matrix $H \in \mathbb{Q}^{n\times m}$, then the generated group is always a lattice, denoted as $\Lambda = \{H\lambda | \lambda \in \mathbb{Z}^m\}$.

The crucial observation from which our proposed method departs from is that, under suitable conditions, the set of reachable configurations of a mobile robot under sequences of control quanta, is a lattice. The planning problem is in this case reduced to solving the linear integer equation

$$y = H\lambda \qquad (1)$$

where y represents the desired configuration (or its approximation on the lattice), and $\lambda \in \mathbb{Z}^m$ represents the number of times certain control words, i.e. sequences of control quanta, are to be used. This is a standard problem in linear integer programming, which can be solved very efficiently in polynomial time, by e.g. using the Hermite normal form of H, $H = [B\ 0]\ U$, where $B \in \mathbb{Q}^{n\times n}$ is a nonnegative, lower triangular, nonsingular matrix, and $U \in \mathbb{Q}^{m\times m}$ is unimodular (i.e., obtained from the identity matrix through elementary column operations).

Clearly, once the generating matrix H and its Hermite normal form have been computed (which can be done in polynomial time [3], [4], and off-line), all possible plans to reach any desired configuration y are obtained at once as

$$\lambda = U^{-1}\begin{bmatrix} B^{-1}y \\ \mu \end{bmatrix},\ \forall \mu \in \mathbb{Z}^{m-n}.$$

A lattice structure hence allows to solve different planning instances in free space in practically negligible time. It also proves very useful in planning amid obstacles, and in computing shortest paths, as it will be discussed in this paper.

With such motivations, questions are in order as to which systems can be planned on lattices, and by which means. Although a general answer to this question is not known at present, the theory of quantized control systems (QCS), a topic of recent research, can provide very useful results. It is known, in particular, that the reachable set of nonholonomic systems in chained form ([5]) with piecewise constant controls taking values in a discrete set, is a lattice ([6]). It is also true that, by suitably choosing the control set, the lattice mesh can be made arbitrarily fine.

In this paper we exploit these results and ideas to propose a planner for the n-trailer vehicle model, which is known to be feedback-equivalent to chained form ([7]).

A. Method outline

The basic steps of the proposed method can be summarized as follows (see fig. 1):

1) write the kinematics of the n-trailer system in the usual coordinates and with velocity inputs as $\dot{q} = T(q)v(t)$, $q \in \mathbb{R}^{3+n}$, $v(\cdot) : \mathbb{R}_+ \to \mathbb{R}^2$ (see (11));
2) use a continuous feedback $v(\cdot) = f(q(\cdot), u)$, and a coordinate change $x = \Phi(q)$, as specified in [7], to obtain an equivalent system $\dot{x} = C(x)u(t)$ in chained-form (see (2));
3) restrict the new input $u(t)$ to piecewise constant functions over a sampling time T, and compute the exact discrete–time model $x(k+1) = \bar{C}(x(k), u(k))$ (see (3));
4) choose a finite, symmetric set of input values $U = \{0, \pm u_1, \pm u_2, \ldots, \pm u_m\}$, and impose $u(k) \in U$;

Work partially supported by RECSYS European Project number IST-2001-37170, FIRB contract 2003-111.

Authors are with the Interdepartmental Research Center "E. Piaggio", University of Pisa, Via Diotisalvi 2, Pisa, Italy.

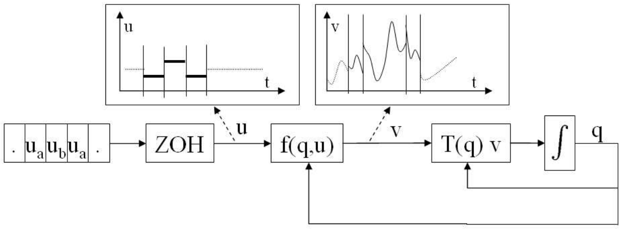

Fig. 1. Symbolic inputs are encoded by feedback into piecewise constant functins, which make the reachable set a lattice.

5) compute the reachable lattice for this system. If the reachable lattice mesh is too rough, change or add elements in U, and recompute;
6) solve the (optimal) planning problem in terms of a finite-length sequence of discrete inputs;
7) apply the computed sequence as a piecewise constant input $u(t)$ to the chained form system, and extract the corresponding path $q(t)$ in the original coordinates, and the piecewise continuos inputs $v(t)$ that solve the steering problem.

II. Reachable Lattices

In this section we briefly report, for reader convenience, some basic definitions and ideas on lattices that will be necessary in the rest of the paper. Details can be found in [6].

In solving the steering problem, a particular class of nonlinear systems, specifically, chained-form systems is considered. Such form has been introduced by Sastry and Murray in [5] as a canonical form for some continuous-time, driftless nonholonomic systems and can be described by ordinary differential equations:

$$\left\{ \begin{array}{lcl} \dot{x}_1 & = & u_1, \\ \dot{x}_2 & = & u_2, \\ \dot{x}_3 & = & x_2 u_1, \\ \vdots & = & \vdots \\ \dot{x}_n & = & x_{n-1} u_1. \end{array} \right. \tag{2}$$

While many steering methods for chained-form systems have been provided in literature, optimal control for these systems is still an open problem.

Consider the case where system inputs, rather than being allowed to change continuously in time, are bound to switch among a finite set of different levels at given switching times, which are multiples of a given time interval. Assuming such sampling interval to be of unit length, a discrete time model of chained-form systems can be easily obtained from (2) by integration as

$$\left\{ \begin{array}{lcl} x_1^+ & = & x_1 + u_1, \\ x_2^+ & = & x_2 + u_2, \\ x_3^+ & = & x_3 + x_2 u_1 + \frac{1}{2} u_1 u_2, \\ \vdots & = & \vdots \\ x_n^+ & = & x_n + \sum_{j=1}^{n-2} x_{n-j} \frac{u_1^j}{j!} + u_1^{n-2} u_2 \frac{1}{(n-1)!}. \end{array} \right. \tag{3}$$

In this paper we assume that inputs $u = (u_1, u_2)$ can take values within a state-independent set of input symbols U, which is symmetric (i.e., if $u \in U$, then also $\bar{u} = -u \in U$). The set Ω of admissible control words (i.e. strings of admissible input symbols) is endowed with a composition law given by concatenation of strings. Because of the symmetry of U, every element $\omega \in \Omega$ has an inverse $\omega^{-1} \in \Omega$, simply defined as $(u_1 u_2 \cdots u_m)^{-1} = -u_m \cdots - u_2 - u_1, \pm u_i \in U, \forall i$.

In the state manifold of chained-form systems (2-3) it is customary to distinguish a *base* subsystem, consisting of the first two state variables (x_1, x_2), and a *fiber* subsystem with coordinates $(x_3, \ldots, x_n)$. Observe that the restriction of chained-form systems to the base variables is linear, and indeed trivial to control. On the other hand, the difficulty in controlling fiber variables increases with the dimension of the state space. A typical example of such situation is in parking maneuvers of tractor-trailer systems, where base variables are associated with the steering tractor, and fiber variables correspond to the configurations of the trailers (see section V).

Accordingly, the reachability problem for discrete-time chained-form systems can be decoupled in the analysis of reachability of the base space, and of the fiber space $\mathbb{R}^{n-2}$ associated with a reachable base point $(\overline{x}_1, \overline{x}_2)$. On the base space system (3) has the simple form

$$x^+ = x + u, \; x \in \mathbb{R}^2, u \in U. \tag{4}$$

For such linear driftless systems, the analysis of the reachable set has been characterized as follows ([6]):

Theorem 1: For the set $R(0, \mathcal{U})$ of configurations reachable from the origin the following holds:

i) A necessary condition for the reachable set from the origin $R(0, \mathcal{U})$ to be dense in $\mathbb{R}^n$ is that U contains $n+1$ controls of which n are linearly independent;
ii) If $\mathcal{U} = \{v_1, \ldots, v_{n+1}\}$, whereof $v_1, \ldots, v_n$ are linearly independent, and ω_i are the components of v_{n+1} w.r.t. the other v_i's, then $R(0, \mathcal{U})$ is dense if and only if ω_i is negative for all i and $1, \omega_1, \ldots, \omega_n$ are linearly independent over $\mathbb{Q}$, that is $a_0 + a_1 \omega_1 + \cdots + a_n \omega_n = 0, a_i \in \mathbb{Q}$, if and only if $a_i = 0$ for all i;
iii) If $u_1, \ldots, u_n \in U$ are linearly independent and there exist n irrational negative numbers $\alpha_1, \ldots, \alpha_n$ such that $v_i = \alpha_i u_i \in U$ for every $i = 1, \ldots, n$ then $R(0, \mathcal{U})$ is dense;
iv) If there exists $m \leq n$ vectors v_i such that $\forall u \in U$, there exists m integers $a_i, \ldots, a_m$ such that $u = a_i v_i$, then $R(0, \mathcal{U})$ is discrete. In particular, it is a lattice.

Observe that the reachable set R_x from a generic point x is obtained by translation of R_0. Therefore, if the control set U is quantized, symmetric and rational (as it almost always is in cases of interest, and as we assume in the rest of this paper), the reachable set is a lattice.

Fixed a base point $(\overline{x}_1, \overline{x}_2)$, consider the subgroup $\tilde{\Omega} \subset \Omega$ of control words that take the base variables back to their initial configuration.

The effect of such subgroup on the fiber subsystem can be described by [6]

$$z^{+}=z+v, z=(x_3,x_4,\cdots,x_n)\in\mathbb{R}^{n-2}, v\in\tilde{U} \quad (5)$$

where $\tilde{U}=\{\Delta^f(\omega),\omega\in\tilde{\Omega}\}$ and where $\Delta^f(\omega)$ denotes the $(n-2)$-dimensional projection of Δ on the fiber space. Clearly, $\tilde{U}$ is itself symmetric: indeed if $\omega\in\tilde{\Omega}$ then also $\omega^{-1}\in\tilde{\Omega}$ and $\Delta^f(\omega^{-1})=-\Delta^f(\omega)$. The action of the subgroup $\tilde{\Omega}$ on the fiber is additive (namely, $\mathcal{A}(\tilde{\omega}_1,\mathcal{A}(\tilde{\omega}_2,x))=\mathcal{A}(\tilde{\omega}_1,x)+\mathcal{A}(\tilde{\omega}_2,x),\forall\tilde{\omega}_1,\tilde{\omega}_2\in\tilde{\Omega}$), and the structure of the reachable set in the fiber is the same over every (reachable) base point.

For the set $\tilde{U}$ of all control inputs that can be applied to the fiber dynamics (5), corresponding to the set of input words $\tilde{\Omega}$ that drive base variables back to their initial values, the following result holds ([8]):

Theorem 2: Let the control set U be quantized, symmetric and rational. Then, all elements $\Delta^f(\tilde{\omega})\in\tilde{U}$ can be written as integer combinations of a finite set of generators Δ_i^f, uniquely determined from U. Each generator is a rational vector in $\mathbb{Q}^{n-2}$, corresponding to a control word $\tilde{\omega}_i\in\tilde{\Omega}$ in the original alphabet U.

As a consequence, with reference to system (4), we can conclude that if the controls set U is rational and quantized, the reachability structure of a chained-form discrete-time system is completely described by a lattice in the state space (the cartesian product of the base and fiber lattices). Such lattice structure can be described completely by a finite number of generators, whose evaluation can be done in polynomial time with respect to the state space dimension and the number of control symbols in U ([6]).

In relation with the optimal steering problem in next section the computation of optimal generators for the lattice is described in details.

A. Generators and transits

In order to compute generators we need several definitions and lemmas that can be found in details in [6]. First of all, let consider a function Σ defined on the set of input words Ω and that counts the number of symbol that appear in a word taking into account signs, for each positive symbol in the control set $U\in\mathbb{Q}^n$. Since U is symmetric its cardinality is even, for example $2c$, then the function Σ takes value in $\mathbb{Z}^c$. Furthermore, let N_W be an integer value matrix such that $WN_W=0$ and such that G.C.D. of element of each column is 1, for each column.

Let c be the number of positive symbol in U, the subgroup $\tilde{\Omega}$ can be described also through Σ and N_W as follow:

$$\tilde{\Omega}=\{\omega\in\Omega|\Sigma(\omega)=(N_W\alpha),\alpha\in(\mathbb{N}\cup\{0\})^{c-2}\}$$

Furthermore, if we define

$$\mathcal{L}=\{\omega\in\Omega|\Sigma(\omega)=\pm(N_W)_j,\omega \text{ of minimal length}\},$$

where $(N_W)_j$ is the j-th column of N_W, we have that the set $C=\{\omega\tilde{\omega}\omega^{-1};\ \omega\in\Omega,\ \tilde{\omega}\in\mathcal{L}\}$ is a set of generators for $\tilde{\Omega}$ but it is not finite.

By theorems in [8], we have that it is possible to compute a finite set of generators of the form

$$\mathcal{B}_{\text{base}}=\{b_i\in\Omega|\Delta^f(b_i)\in\mathcal{B}\},$$

where b_i are the *generators* and can be written as $b_i=\hat{\omega}_i\tilde{\omega}\hat{\omega}_i^{-1}$ with $\tilde{\omega}\in\mathcal{L}$ and where the control sequences $\hat{\omega}_i$ are called *transits*.

For example, on a two dimensional lattice, the transit u and the word $\omega=v\,u\,-v\,-u\in\mathcal{L}$ (figure 2, left) give the generator $u\,v\,u\,-v\,-u\,-u$ represented in figure 2 (right). With respect to cyclic generators (elements of $\tilde{\Omega}$), transits cause a translation on the lattice structure of cyclic generators (see figure 2).

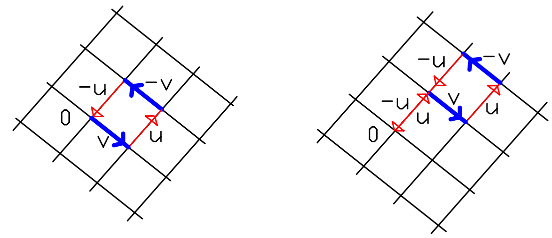

Fig. 2. Left: representation of $\omega=v\,u\,-v\,-u\in\Omega$, it takes back the base variables (form a cycle on the base lattice), right: an example of composition of ω with transit u, it is a non minimal cycle.

While the control sequences $\tilde{\omega}\in\mathcal{L}$ are obtained at lower cost by construction, for the transits it is necessary the following optimization algorithm (in this formulation the problem is solved in minimal time but more general weights-problems can be solved equivalently).

We consider a function $\sigma:\Omega\mapsto\mathbb{Q}$ defined as

$$\sigma(\omega)=\sum_{i=1}^{c}\alpha_i u_{i,1}$$

where $\alpha_i\in\mathbb{Q}$ and $u_{i,1}$ is the first component of a the control $u_i\in U$. Since the generators are rational and a finite set, it is possible to define a function k from the control sequences Ω to $\mathbb{Z}$ such that , $\sigma(\omega)=\frac{p}{q}k(\omega)$, where k provides the integer part of the value $\sigma(\omega)$.

Suppose that: *the G.C.D. between at least two first components of symbols in U is one*. This condition is strictly related to the existence of $\hat{\omega}_1$ such that $k(\hat{\omega}_1)=1$, and it is sufficient to allow correctness of the following algorithm:

Step i: for i from 1 to $n-3$

Solve

$$\begin{array}{ll}\min & \hat{\Sigma}(\omega)\\ \text{s. t.} & \left\{\begin{array}{ccc} k(\omega) & = & i\\ \omega & \in & \Omega\end{array}\right.\end{array} \quad (6)$$

where the function $\hat{\Sigma}(\omega):\Omega\mapsto\mathbb{N}$ counts the number of symbols in the word ω without taking into account signs. Let $\hat{\omega}_i$ be the optimal solution founded at step i.

Since the optimization problems (6) are linear and have infinite dimension, they are $\mathcal{NP}$-complete. The $\mathcal{NP}$-completeness can be solved rewriting the problem 6 as follow:

$$\begin{array}{ll}\min & \|x\|_1\\ \text{s. t.} & \left\{\begin{array}{ccc} Fx & = & i\frac{p}{q}\\ x_i & \in & \mathbb{N}\end{array}\right.\end{array} \quad (7)$$

where $|U|=m$ and the matrix $F\in\mathbb{Q}^{1\times 2m}$ is composed of the first component of each control input in U and the component x_i of the vector x counts how many times the control to which F_i belongs is considered in the solution.

Such optimization problems can be easly solved by an integer optimization commercial package such as CPLEX [9].

As we have explained at the beginning of this section, the transits are employed to construct cyclic generators in relation to the dimension of the configuration space but it is important to underline that the choice of the transit $\hat{\omega}_i$ is independent of the construction of the increment matrix $\Delta^f(\omega)$ of the steering problem. Indeed, in order to determine the control generators we use only the fact that a transit verifies the condition $k(\omega) = i$, for particular i, without using an explicit word ω, [8].

III. Steering on lattices: the optimal control problem

Consider the system (3) with a quantized, rational and symmetric control set U. Let $H \in \mathbb{Q}^{n-2\times l}$ denote the matrix whose columns are the l generators contribution on the fiber $\Delta_i^f = \Delta(\tilde{\omega}_i), i = 1, \cdots, l$. Without loss of generality, up to rescaling the fiber state space, we may take H to be an integer matrix. The *steering problem* on the fiber without obstacles is the problem of finding a control sequence that takes the system (3) from an initial q_{start} to a desired q_{goal}. Hence, it consists in solving a linear system of the form:

$$Hx = (q_{goal} - q_{start}). \tag{8}$$

Integer solutions $x \in \mathbb{Z}^l$ of this equation exist if an only if the initial and goal points differ by a vector belonging to the fiber lattice, which we will assume henceforth (in other cases, integer truncations of a real solution x will provide approximated steering to the goal, within a tolerance dictated by the lattice mesh).

Any solution $x = (x_1, \cdots, x_l) \in \mathbb{Z}^l$ of system (8) gives a sequence of cyclic control inputs that includes x_i instances of the words $\tilde{\omega}_i$. There are of course infinitely many possible solutions x, each corresponding to a combinatoric number of different possible sequences of control words $\tilde{\omega}_i$.

Optimal steering strategies among solutions of (8) will be considered introducing a cost p_i associated to the control symbol $u_i \in U$. The corresponding cost for a word $\omega = (u_1, u_2, \cdots, u_N), u_i \in U$ is defined as $C(\omega) = \|Px\|$, where x_i stands for the number of appearances of the symbol u_i in ω (with negative sign if $-u_i$ appears), and $P = \mathrm{diag}(p_i)$.

A constrained minimization problem can be considered at this point, i.e.

$$\begin{array}{ll} \min_x & \|Px\| \\ \text{s. t.} & \left\{ \begin{array}{l} Hx = x_{goal} - x_{start} \\ x \in \mathbb{Z}^m \end{array} \right. \end{array} \tag{9}$$

leading to a linear integer program if a one-norm is considered, while using a two-norm would result in an integer quadratic program. Efficient algorithms do exist for both these problems: however, unfortunately, such formulation does not reflect the reality of our optimal control problem.

Indeed, in combining control words by concatenation cancellations of symbols may occur. To obtain the sum of two control actions $\Delta(\tilde{\omega}_i), \Delta(\tilde{\omega}_j)$ on the fiber, corresponding to control words $\tilde{\omega}_i, \tilde{\omega}_j$ whose costs are $C_i = C(\tilde{\omega}_i)$ and $C_j = C(\tilde{\omega}_j)$, respectively, the sum $C_i + C_j$ is only an upper bound to the actual cost of the corresponding control. Indeed, cancellations of one or more trailing symbols in $\tilde{\omega}_i$ with an equal number of symbols leading in $\tilde{\omega}_j$ is possible. We will denote by $\hat{C}(\tilde{\omega}_i, \tilde{\omega}_j)$ the actual cost of the word pair $(\tilde{\omega}_i, \tilde{\omega}_j)$.

For example, if $\tilde{\omega}_i = u_1u_2u_3u_4$ and $\tilde{\omega}_j = -u_4u_5 - u_2 - u_1$, in a minimum time problem we have $C_i = 4$ and $C_j = 4$. However, the concatenation of $\tilde{\omega}_i$ with $\tilde{\omega}_j$ leads, by cancellations, to the control word $u_1u_2u_3u_5 - u_2 - u_1$, so that $\hat{C}(\tilde{\omega}_i, \tilde{\omega}_j) = 6 < 8$. Obviously, cancellations are crucial in minimizing unnecessary maneuvers in the steering problem, and motivate the following reformulation of the optimal control problem.

Consider an oriented graph $G_0 = (N_0, A_0)$ with a set N_0 of $l+2$ nodes, l of which are associated with the contributions $\Delta(\omega_i)$ on the fiber given by generators $b_i = \tilde{\omega}_i$, and where a start node S and a goal node F are additionally considered. In the arc set A_0 of G_0, all arcs connecting the start and goal nodes S, F with all other nodes are included, i.e. $(S, i) \in A_0, i = 1, \ldots, l$ and $(i, F) \in A_0, i = 1, \ldots, l$. An arc (i, j) is included in A_0 only if $\tilde{\omega}_i$ and $\tilde{\omega}_j$ are not the inverse of each other. In particular, for every node $i \neq S, F$, the arc (i, i) is included in A_0.

To the arc $(i, j) \in A_0$ we associate the cost $\hat{C}_{i,j} = \hat{C}(\tilde{\omega}_i, \tilde{\omega}_j) - C(\tilde{\omega}_i) \geq 0$ of the control sequence $(\tilde{\omega}_i, \tilde{\omega}_j)$ (so that the cost of path (S, i), (i, j) on the graph is $C(\tilde{\omega}_i, \tilde{\omega}_j)$), taking into account all possible cancellations. Notice that in general $\hat{C}_{i,j}$ is not equal to $\hat{C}_{j,i}$ (in the example above, for instance, $\hat{C}_{i,j} = 6 - 4 = 2$ while $\hat{C}_{j,i} = 4 - 4 = 0$). In figure 3 an example of graph is represented. A refinement step is necessary to finalize the graph construction for pairs $(\tilde{\omega}_i, \tilde{\omega}_j)$ where the number of cancellations is larger than the half-length of the shortest of the two words. Indeed, in this case it may happen that the cost of a triplet $(\tilde{\omega}_i, \tilde{\omega}_j, \tilde{\omega}_k)$ is underestimated by $\hat{C}_{ij} + \hat{C}_{jk}$. For example, if $\tilde{\omega}_i = u\,v - u\,v\,u$, $\tilde{\omega}_j = -u - v\,u\,v$ and $\tilde{\omega}_k = -v - u - u$, we have $\tilde{\omega}_i\tilde{\omega}_j = u\,v\,v$ ($\hat{C}_{ij} = 3-5 = -2$) and $\tilde{\omega}_j\tilde{\omega}_k = -u - v - u$ ($\hat{C}_{jk} = 3-4 = -1$) while the triplet is $\omega_i\omega_j\omega_k = u\,v - u - u$ whose cost is $4-5 = -1$ whereas on the graph the path $(S,\, i)$, $(i,\, j)$, $(j,\, k)$ would cost $5-2-1 = 2$. To avoid this problem, we remove in the graph the arc (i, j) corresponding to such pairs, and add a new node associated to $\Delta(\tilde{\omega}_i) + \Delta(\tilde{\omega}_j)$ with cost $\hat{C}_{ij}$. These new nodes are connected to all other nodes by arcs whose cost is evaluated as usual, with the exception of arcs corresponding

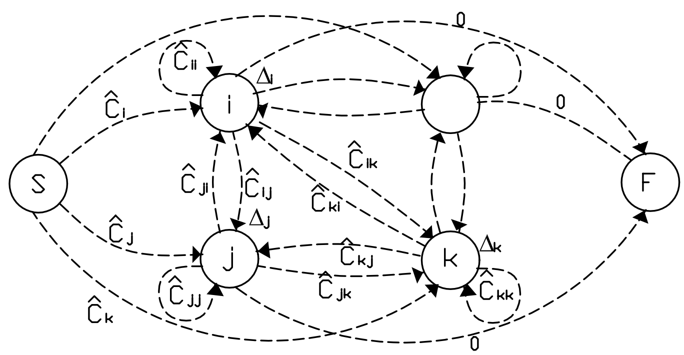

Fig. 3. Graph associated with generators of the fiber displacements.

again to cancellations of more than the half-length of either words, which are not considered in the new graph.

On the graph G_0, all possible combinations of the generating control words $\tilde{\omega}_i$ are represented by connected paths from S to F. The optimal control problem on the fiber space can hence be formulated as follows:

Given the oriented graph G_0, determine the minimum-cost path from S to F with the constraint that the sum of all Δ_i of visited nodes equals the desired fiber displacement $z_{goal} - z_{start}$.

Thus, the optimal control problem can be regarded as a minimum-cost path search on a graph, with a constraint on the sum of "tokens" collected at each visited node. Notice that G_0 contains cyclic arcs of type (i, i), allowing to collect an arbitrary integer number of the corresponding token $\Delta(\tilde{\omega}_i)$. The search problem is a $\mathcal{N}P$-complete linear integer programming problem ([3],[4]), and differs substantially from standard shortest path searches on a graph because of the constraint and of the presence of cycles (cyclic paths are obviously never considered in unconstrained path searches). The following section proposes a correct and complete algorithm to solve this optimal control problem.

IV. A SOLUTION ALGORITHM

The non-standard nature of the optimization problem described above is such that even rather general solution techniques, as e.g. branch and bound, and commercial software tools for integer programming, cannot be used directly to solve the problem. We propose a procedure for the solution of this problem which basically consists of solving a sequence of problems of increasing complexity.

Consider first that an upper limit U on the optimal control cost can be easily obtained by evaluating the cost U_0 of *any* solution of the integer linear system (8) – for instance, a solution to problem (9), in the following will be referred to as *starting solution*.

At the first stage of the proposed algorithm, a new graph $G_1 = (N_1, A_1)$ is built by setting $N_1 = N_0$ and by removing all cyclic arcs from A_0, namely $A_1 = A_0 \setminus \{(i, i), \forall i\}$. Let now formalize the optimization problem obtained with the formulation given in previous section. Consider the incidence matrix $E \in \mathbb{R}^{s \times t}$ associated with the graph G_1: given an order to the elements of set A_1 (cardinality t) and of set N_1 (cardinality s), the element $E_{ij} = -1$ if the $i-th$ node is the first node of arc j, $E_{ij} = 1$ if the $i-th$ node is the second node of arc j, $E_{ij} = 0$ otherwise. Let $x \in \mathbb{R}^t$ be the vector variables taking values in $\{0, 1\}^t$ and representing the ordered arcs of the graph. Let $q \in \mathbb{R}^s$ such that $q_S = -1$, $q_F = 1$ and $q_i = 0$ for $i \neq S, F$. Finally, let $C^T \in \mathbb{R}^t$ be the vector in which the cost of the arcs are reported, the optimization problem is then

$$\begin{array}{ll} \min & Cx \\ s.t. & \left\{ \begin{array}{l} Ex = q \\ \tilde{H}x = d \\ x \in \{0, 1\}^t \end{array} \right. \end{array} \tag{10}$$

where the set of constraints $\tilde{H}x = d$ (in the following will be referred to as set of *token constraints*) represents the constraints given in (8) where $\tilde{H} \in \mathbb{R}^{n-2 \times t}$ and the column $\tilde{H}_j$ is associated to the arc $j = (i, k)$ and represent the "token" payed at node k that is $\Delta(b_k)$ (where b_k is the generator associated with node k). The vector D represent the total displacement we intend to achieve on the fiber.

A branch-and-bound algorithm is applied to search minimum cost, token-constrained paths on G_1. Within such branch-and-bound subprocedure, the token constraint is relaxed, hence a number of classical minimum cost path search problems are obtained (solvable by the Dijkstra algorithm [10]) in each of which an arc is forced to be ($x_i = 1$) or not ($x_i = 0$) in the optimal solution. If the forced condition $x_i = 1$ or $x_i = 0$ brings to a shortest path of cost larger that U then the relative branch is cut and not further explored. Otherwise, another arc is forced to be or not in the optimal solution. If all branch are cut then no solution with cost less than U has been found. Otherwise, an optimal solution is found with cost $U_1 < U$. This solution is the shortest path from node S to F but in order to be an admissible solution of problem (10) it has to verify the token constraint. In this case the upper bound U on the optimal cost is updated, $U = U_1$.

At the $i+1$–th step of the algorithm, a graph $G_{i+1} = (N_{i+1}, A_{i+1})$ is built such that $N_{i+1} = N_i + N_0 \setminus \{S, F\}$, and A_{i+1} contains all connecting arcs between different nodes in N_{i+1} (without cyclic arcs). In other words, each node j with a cycle arc is split into two nodes (see figure 4) so that at step i, path with i cycles can be considered. A branch-and-bound algorithm is used again to find the constrained minimum cost U_{i+1}, and the upper bound is updated if $U_{i+1} < U$ and if the solution verifies the token constraint.

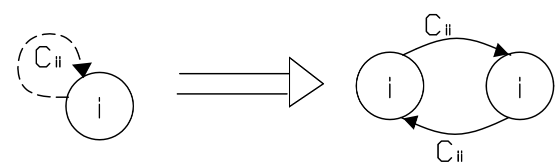

Fig. 4. The node i with a cycle arc is split into two nodes and two arcs.

A stopping condition for the procedure can be provided as follows. A lower bound on the optimal control cost solution L is initially set equal to the cheapest cost $L_0 = C_i$ of arcs of type (S, i) in the G_1 graph, since the cost of arc (i, F) is zero. At each step, the lower bound is updated as $L = L_{i+1} = L_i + \hat{C}_c$, where $\hat{C}_c$ denotes the minimum cost of a closed cycle in the graph G_1. The value of $\hat{C}_c$ is determined once and for all at the beginning of the procedure, by solving a standard (unconstrained) minimum-cost path problem on G_1.

The overall procedure is stopped whenever $L \geq U$.

Theorem 3: The solution algorithm is correct and complete.

Proof: Because initial and goal configurations are assumed to belong to the lattice, the optimum exists. Also, because the action on the fiber of the whole group $\tilde{\Omega}$ of control inputs that correspond to the desired final value of the base variables, is generated by the finite set of generators $\Delta(\tilde{\omega}_i), i = 1, \ldots, m$, and this set is (implicitly, but completely) searched by the branch-and-bound algorithm at successive stages of the algorithm, the algorithm is correct. On the other hand, the two sequences $\{L_i\}_{i \geq 0}$ and $\{U_i\}_{i \geq 0}$ are strictly uniformly increasing and non-increasing, respectively, and at any stage it holds $L_i \leq U_i$. Hence the algorithm stops in a finite number of stages, all of which consist of an implicit search on a finite graph, i.e. of a finite number of operations. ■

The proposed algorithm has exponentially increasing complexity with the number of generators, as it uses a number of instances of a branch and bound procedure: this is hardly a surprise, as we are after all dealing with a nontrivial optimal control problem. However, performance can be improved by providing good initial estimates of the upper bound U_0. Some preprocessing of generators to facilitate the algorithm convergence can also help, and work is currently ongoing in this direction. The next section will provide some numerical examples of application of the proposed algorithm.

V. n-TRAILER STEERING WITH OBSTACLES

As mentioned in the introduction, among the nonlinear systems which can be converted in chained-form (2), wheeled vehicles represent a particularly interesting class.

The kinematic model of a tractor with n trailers is given by

$$\left\{\begin{array}{lcl} \dot{x} & = & \cos\theta_n v_n \\ \dot{y} & = & \sin\theta_n v_n \\ \dot{\theta}_n & = & \frac{1}{d_n}\sin(\theta_{n-1}-\theta_n)v_{n-1} \\ & \vdots & \\ \dot{\theta}_i & = & \frac{1}{d_i}\sin(\theta_{i-1}-\theta_i)v_{i-1} \quad i=1,\ldots,n \\ & \vdots & \\ \dot{\theta}_1 & = & \frac{1}{d_1}\sin(\theta_0-\theta_1)v_0 \\ \dot{\theta}_0 & = & \omega \end{array}\right. \tag{11}$$

where (x, y) is the absolute position of the center of the axle between the two wheels of the rear-most trailer; θ_i is the orientation angle of trailer i with respect to the x-axis, with $i \in \{1,\ldots,n\}$; θ_0 is the orientation angle of the tractor axle with respect to the x-axis; d_i is the distance from the center of trailer i to the center of trailer $i-1$, $i \in \{2,\ldots,n\}$; d_1 is the the distance from the wheels of trailer 1 to the wheels of the tractor. The two inputs of the systems are v_0 and ω, the tangential velocity of the car and the angular velocity of the tractor respectively. The tangential velocity of a trailer i, v_i, is given by

$$v_i = \cos(\theta_{i-1}-\theta_i)v_{i-1} = \prod_{j=1}^{i}\cos(\theta_{j-1}-\theta_j)v_0,$$

where $i \in \{1,\ldots,n\}$. Incidentally, this model is identical to the model of a four-wheeled car pulling $n-1$ trailers, provided $\theta_0-\theta_1$ denotes the angle of the front wheels relative to the orientation θ_1 of the rear axle of the four-wheeled car.

Sørdalen in [7] has shown, by a constructive method, that system (11) can be converted in chained-form. We consider here the application of the lattices steering algorithms to the general steering problem with obstacles and to the optimal steering problem for wheeled vehicles with trailers.

This implies introducing time and control quantizations in (11) that by conversion and feedback has a lattice as reachable set. Solving the linear system (8), a solution to the steering problem in an unconstrained environment is computed. In particular, from a solution of (8), a controls collection $\{\overline{u}_i\}_{i\in J}$, with $J \subset \mathbb{N}, |J| < \infty$, is provided to solve the specific steering task in polynomial time.

By conversion, this control sequence yields a sequence of piece-wise continuous controls for the original system (11).

A. *Collision free trajectory planning for n-trailers*

Once controls are obtained as solution of the steering problem (8) the continuous time trajectory of system (11) is computed through the integration of such controls. Let consider an environment with obstacles that can be approximated with polyhedral O_i of the form

$$O_i = \{x \in \mathbb{R}^n | A_i x \leq b_i\}.$$

A *Collision Test function* can be introduced, as a function from the configuration space $\mathcal{X}$ to binary values $\{0, 1\}$, as following

$$\mathcal{CT} : \mathcal{X} \mapsto \{0, 1\}$$

where $\mathcal{CT}(x) = 0$ if no collision is detected and $\mathcal{CT}(x) = 1$ otherwise. In checking collisions, a security distance d from the side of the polyhedral obstacles and a security radius r for the vehicles to steer are considered.

A conflict free trajectory can be easily and quickly computed as follows. Given initial and final configurations (named c_i and c_f respectively) a solution of the steering problem is obtained by solving the linear system (8). Each solution provide the number of time x_i each generator ω_i must be applied in order to reach the desired configuration. The order of application of control words is arbitrary. Let choose a sequence $\{\overline{u}_i\}_{i=1,\ldots,m}$ of generator such that each generator ω_i appears exactly x_i times in the sequences, where $m = \sum_{i=1}^{l} x_i$ represents the total number of times generators are applied. The chosen control sequence is then integrated and the collision test is applied to the obtained continuous time trajectory. If a collision is detected, the control $\overline{u_j}$ which causes the collision can be computed and then removed from the solution control sequence $\{\overline{u}_i\}_{i=1,\ldots,m}$. Let now consider another order of the cutted sequence $\{\overline{u}_i\}_{i=j,\ldots,m}$ so that the first generator is different from $\overline{u_j}$ otherwise the same conflict is detected.

Planning, trajectory integration and collisions checking are repeated until a free control sequence is obtained as solution of the steering problem. Assume that after a sequence $\{\overline{u}_i\}_{i=1,\ldots,k-1}$ the state $\tilde{c}_1$ is reached and no conflict free control sequences can be found by permutations of $\{\overline{u}_i\}_{i=k,\ldots,m}$. In this case also the control $\overline{u}_{j-1}$ is removed by the control sequence and the procedure continues as described above.

If during the described procedure, it is necessary to remove all controls from the initially computed sequences the algorithm is stopped and it is not enable to provide a solution. Otherwise, a collision free path has been obtained through a permutation of the sequence $\{\overline{u}_i\}_{i=1,\ldots,m}$.

In figure 5 a trajectory computed with the described procedure is reported for a car-like system with a polyhedral obstacle, the trajectory cost is equal to 12.

Fig. 5. Trajectory of cost 12 for the car-like system with polyhedral obstacle.

Fig. 6. Optimal trajectory of cost 8 for the car-like system in an environment with no obstacle.

B. Optimal collision free trajectory planning for n-trailers

Let now consider the problem of optimally steering n-trailers in an environment with polyhedral obstacles. First, the system (11) is converted in chained-form, and the algorithm described in section III and IV is applied. The collision test function has been integrated in the algorithm as follows.

At each step of the algorithm a solution is provided, this solution consists in a sequence of control that steer the system as desired and has a minimum cost for the current step. The continuous time trajectory is then computed and tested for collisions through the collision function $\mathcal{CT}$. If collisions are detected the solution is not admissible and the optimal solution of the algorithm it is not updated. Notice that the collision test does not modify neither the structure of the algorithm reported in IV and nor its completeness and correctness properties.

In figure 6 an optimal trajectory, computed with the algorithm described in section IV is reported for a car-like system, the trajectory cost is equal to 8.

Referring to figure 7 the same problem reported in figure 6 is exploited, in particular same initial and final configurations are considered. Let consider an obstacle in front of the car-like vehicle, the optimal conflict trajectory computed with the algorithm described in section IV is not collision free. The optimal collision free trajectory tested with the collision test function has always cost 8 and is reported in figure 7.

Fig. 7. Optimal trajectory of cost 8 for the car-like system with polyhedral obstacle.

VI. CONCLUSIONS

In this paper, the steering problem with obstacles for wheeled vehicles with trailers, has been studied by introducing inputs quantization and converting the continuous-time kinematic model of these systems in chained-form.

This systems class, that represents a so called “canonical form” for a wide range of mechanic systems, has the important property to have a lattice as reachable set under quantized rational inputs. This structure plays a central role in solving the steering problem for this systems class, for which a polynomial algorithm has been developed.

The lattice structure can be used to solve the optimal steering problem. In particular the optimal control problem on reachable lattices is formalized as an integer linear programming problem that cannot be solved directly by standard integer programming techniques and therefore a correct and complete solution algorithm has been proposed.

With relation to wheeled vehicles with trailers, by inputs quantization and conversion in chained-form, the steering problem has been solved on lattices and our optimization algorithm yields sub-optimal solutions for the optimal control.

Applying the previous results, the steering problem has been solved also in presence of obstacles with a minimal computational additional cost and the experiments give satisfactory results.

Since in this paper the steering problem has been considered, open–loop controls are computed. Our future work consists in solving the optimal control in close–loop, applying the lattice structure of the reachable sets and classical control technics such us Dynamic Programming.

REFERENCES

[1] E. Frazzoli, M. A. Dahleh, and E. Feron, “Real-time motion planning for agile autonomous vehicles,” in *Proc. AIAA Guidance, Navigation, and Control Conference and Exhibit*, August 2000, pp. 1–11.

[2] A. Marigo and A. Bicchi, “Steering driftless nonholonomic systems by control quanta,” in *Proc. IEEE Int. Conf. on Decision and Control*, 1998.

[3] A. Schrijver, *Theory of Linear and Integer Programming*. Wiley Interscience Publ., 1986.

[4] L. A. Wolsey, *Integer Programming*. Wiley Interscience Publ., 1998.

[5] S. S. S. R. M. Murray, "Nonholonomic motion planning: Steering using sinusoids," *IEEE Trans. on Automatic Control*, vol. 38, pp. 700–716, 1993.

[6] A. Bicchi, A. Marigo, and B. Piccoli, "On the recahability of quantized control systems," *IEEE Trans. on Automatic Control*, May 2002, in press.

[7] O. Sordalen, "Conversion of the kinematics of a car with n trailers into a chained form," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1993, pp. 382–387.

[8] A. Marigo, B. Piccoli, and A. Bicchi, "Reachability analysis for a class of quantized control systems," in *Proc. IEEE Int. Conf. on Decision and Control*, 2000, pp. 3963–3968.

[9] "Ilog cplex user-s guide," Tech. Rep., 1999.

[10] E. V. Denardo, *Dynamic Programming: Models and Applications*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey,, 1982.