Fast Computation of Robot-Obstacle Interactions in Nonholonomic Trajectory Deformation

Olivier Lefebvre, Florent Lamiraux and David Bonnafous
LAAS-CNRS,
7 avenue du Colonel Roche
Toulouse, France
{olefebvr,florent,dbonnafo}@laas.fr

***Abstract*— This paper deals with the optimization of Robot-Obstacle interaction computations, in the context of nonholonomic trajectory deformation for mobile robots. We first recall the principle of the trajectory deformation and the role of the potential field gradient in the configuration space. The contribution of the paper is twofold. First we show that the potential field gradient can be computed without any closed-form expression of the potential function if this latter depends only on the distance between the robot and the obstacles. Then an algorithm to filter obstacles that have no influence in Robot-Obstacle interactions is presented. This algorithm takes advantage of the spatial coherence of the planned trajectory, and has been evaluated by experiments on mobile robot Hilare2 towing a trailer.**

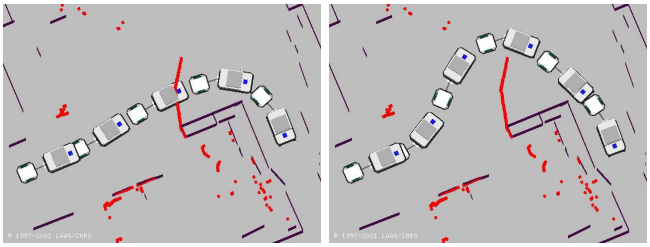

Fig. 1. Application of the trajectory deformation method to the mobile robot Hilare 2 towing a trailer. light dots are obstacles detected by a laser scanner. The robot is at the beginning of the planned trajectory on the left. The trajectory is deformed in such a way that it avoids obstacles, the kinematic constraints keep satisfied and the trajectory starts and end at the same configurations before and after deformation.

I. Introduction

Autonomous navigation for multi-body nonholonomic mobile robots in cluttered environments usually requires two distinct steps. The first step consists in planning a collision-free admissible trajectory given a map of the environment and the second step consists in following this trajectory. Imprecision of the map, localization errors and unexpected obstacles however may make the second step fail. For this reason, the trajectory following task needs to be reactive to these perturbations.

Reactive motion in mobile robots has given rise to a lot of work and a lot of methods have been proposed for simple mobile robots without nonholonomic constraints [12], [4], [2], [10]. Some of these methods have been extended to simple nonholonomic mobile robots like unicycle or car-like systems [6], [11].

Recently, we have proposed a generic method to reactively deform a trajectory for a nonholonomic system in order to avoid obstacles detected by on-board sensors along the motion [8]. The method is based on the minimization of a trajectory potential that increases when the trajectory gets closer to obstacles. The approach is very generic and has been successfully applied to complex truck-trailer systems [7].

The method is based on the perturbation of the input functions of the system along the current trajectory. These functions are iteratively perturbed in such a way that:

1) the deformed trajectory gets away from obstacles,
2) the start and end configurations of the trajectory remain the same and
3) the nonholonomic constraints keep satisfied along the trajectory.

Figure 1 shows the result of the method applied to our mobile robot Hilare 2 towing a trailer. Although points 2 and 3 above are well explained in previous papers [8], the computation of the interactions between the obstacles and the trajectory is the bottleneck of the method and can take up to 80% of the computation time.

This paper deals with the computation of the interactions between the obstacles and the trajectory within the nonholonomic trajectory deformation method. The action of the obstacles over the trajectory is defined by the integral along the trajectory of the gradient of a potential function over the configuration space. As the trajectory is discretized and the potential function is generated by the obstacles, the number of computations without optimization is the product of the number of "*sample configurations*" along the trajectory by the number of "*obstacles*". If the obstacles have a bounded distance of influence however, most of the pairs "*sample configuration - obstacle*" have no effect.

The contribution of our paper is twofold. First we show that the gradient of a configuration space potential field based on the distance between the robot and the obstacles can be computed without the closed-form expression of the potential function. Secondly we propose an algorithm to dramatically reduce the number of computations relative to the interactions between the trajectory and the obstacles by filtering the pairs sample configuration-obstacle that have no effect.

The problem of filtering pairs of objects for distance computation and collision avoidance has given rise to a lot of work in the field of computational geometry [9], [13], [3]. Our algorithm is based on spatial coherence (continuity

of the trajectory) and is significantly different from these previous works. A related work can be found in [1], but it does not deal with multi-body systems trajectories of any shapes.

In Section II, we briefly recall the main components of the trajectory deformation method. In Section III, we define the configuration space potential field and we explain how to simply compute the gradient. In Section IV, we describe an algorithm for pruning useless pairs configuration-obstacle in the computation of the interactions between the obstacles and the trajectory. Finally, in Section V we give some experimental results that show the benefit of our algorithm.

II. Nonholonomic trajectory deformation method

A nonholonomic system of dimension n is defined by $k < n$ control vector fields $\mathbf{X}_1$,...,$\mathbf{X}_k$ over the configuration space $\mathcal{C}$ of the system. An admissible trajectory γ is a mapping from an interval $[0, S]$ into the configuration space, the derivative of which is a linear combination of the control vector fields:

$$\begin{array}{rccc} \gamma: & [0,S] & \rightarrow & \mathcal{C} \\ & s & \rightarrow & \gamma(s) \end{array}$$

and there exists k mappings u_1,...,u_k from $[0, S]$ into $\mathbb{R}$ such that:

$$\forall s \in [0,S], \quad \gamma'(s) = \sum_{i=1}^{k} u_i(s)\mathbf{X}_i(\gamma(s))$$

$'$ denotes the derivative w.r.t. s. The u_i's are the input functions relative to trajectory γ.

A. Principle of the trajectory deformation method

In this section, we recall the key points of the trajectory deformation method for nonholonomic systems. We refer the reader to [8] for a precise description.

We define a potential field U over the configuration space, decreasing when the distance between the robot and the obstacles increases. Section III will give more details about this potential field. From this potential field, we define a potential field over the space of trajectories by integration of the configuration space potential value:

$$V(\gamma) = \int_0^S U(\gamma(s))ds \tag{1}$$

The nonholonomic trajectory deformation method is based on the perturbation of the input functions of current trajectory γ. The idea consists in iteratively determining k mappings v_1,...,v_k from $[0, S]$ into $\mathbb{R}$ in such a way that replacing each u_i by $u_i + \tau v_i$, where τ is a small positive real number, yields a new admissible trajectory that:

1) starts and ends at the same configurations $\gamma(0)$ and $\gamma(S)$,
2) has a lower potential value than the initial trajectory.

We denote by γ_τ the new trajectory of input functions $u_i + \tau v_i$. Then, the trajectory deformation is asymptotically defined by a vector valued mapping η from $[0, S]$ into the tangent configuration space ($\mathbb{R}^n$ to make it simple):

$$\gamma_\tau = \gamma + \tau\eta + \varepsilon(\tau)$$

where $\varepsilon(\tau)$ is a negligible term w.r.t. τ when τ tends toward 0.

The relation between input perturbations $\mathbf{v} = (v_1, ..., v_k)$ and trajectory deformation η is given by the linearized system about the initial trajectory.

The key point in the scope of this paper is that the asymptotic variation of the trajectory potential value for a given trajectory deformation η is given by the following expression:

$$< dV(\gamma), \eta >= \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\gamma(s))\eta(s)ds$$

where $\frac{\partial U}{\partial \mathbf{q}}$ is the gradient of the configuration space potential field, that is the derivative of U w.r.t. each configuration variable of the system. Expressed differently:

$$V(\gamma_\tau) = V(\gamma) + \tau \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\gamma(s))\eta(s)ds + \varepsilon(\tau)$$

In the general case, the computations of $\frac{\partial U}{\partial \mathbf{q}}$ may be complicated since we need to have a closed-form expression of the potential function U. In the following section however, we are going to show that if the potential value is a function of the distance between the robot and the obstacles, we do not need the closed-form expression of the configuration space potential function.

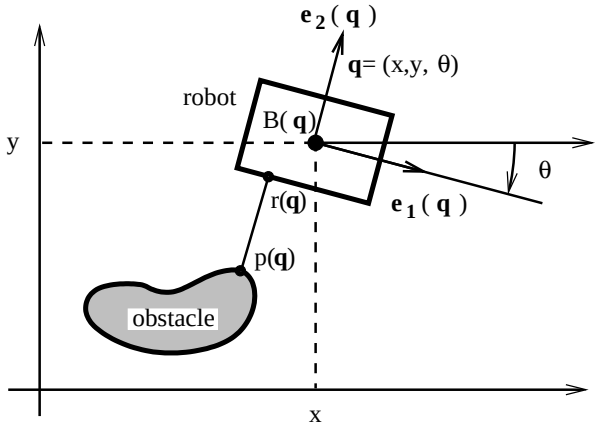

Fig. 2. The gradient of a configuration space potential field depending only on the distance between the robot and the obstacles is a function of the vector $\vec{\mathbf{OR}}$ linking the closest point on the obstacle and on the robot and of the gradient of the position of the point of the robot coinciding with $\mathbf{R}$. The relative motion of $\mathbf{R}$ on the robot can thus be omitted.

III. Potential field over the configuration space

We denote by $\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$ the workspace of the robot. Given two compact subsets $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{W}$, we call *distance between these subsets* the following value:

$$d(\mathcal{A}, \mathcal{B}) = \min_{a \in \mathcal{A}, b \in \mathcal{B}} \|b - a\| \tag{2}$$

Let us consider a multi-body mobile robot composed of n_b bodies $\mathcal{B}_1$,...,$\mathcal{B}_{n_b}$. Let us denote by $\mathcal{B}_i(\mathbf{q}) \subset \mathcal{W}$ the volume occupied by body $\mathcal{B}_i$ in configuration $\mathbf{q}$.

Let the environment contain n_o obstacles $\mathcal{O}_1$,...,$\mathcal{O}_{n_o}$. We consider a configuration space potential function depending on the distances between obstacles and bodies of the robot as follows:

$$U(\mathbf{q}) = \sum_{i=1}^{n_b} \sum_{j=1}^{n_o} f(d_{ij}(\mathbf{q})) \tag{3}$$

where:

$$d_{ij}(\mathbf{q}) = d(\mathcal{B}_i(\mathbf{q}), \mathcal{O}_j)$$

is the Euclidean distance between body i and obstacle j when the robot is in configuration $\mathbf{q}$ and f is a decreasing function. Let us point out that the nonholonomic constraints have no influence on the potential function. The gradient of this expression is thus:

$$\frac{\partial U}{\partial \mathbf{q}}(\mathbf{q}) = \sum_{i=1}^{n_b} \sum_{j=1}^{n_o} f'(d_{ij}(\mathbf{q})) \frac{\partial d_{ij}}{\partial \mathbf{q}}(\mathbf{q})$$

where f' denotes the derivative of f. We thus need to compute the gradient of the distance between each pair body-obstacle. For that, we denote by $\mathbf{R}(\mathbf{q})$ and $\mathbf{O}(\mathbf{q})$ the closest points between body i in configuration $\mathbf{q}$ and obstacle j:

$$d_{ij}(\mathbf{q}) = \|\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q})\|$$

Thus, if ^{T} denotes the transpose of a vector,

$$\frac{\partial d_{ij}}{\partial \mathbf{q}}(\mathbf{q}) = \frac{(\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q}))^T}{\|\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q})\|} \left(\frac{\partial \mathbf{O}}{\partial \mathbf{q}}(\mathbf{q}) - \frac{\partial \mathbf{R}}{\partial \mathbf{q}}(\mathbf{q}) \right) \tag{4}$$

From this expression, it seems that the gradient of the potential field requires the expression of the position of $\mathbf{O}(\mathbf{q})$ and $\mathbf{R}(\mathbf{q})$ w.r.t. $\mathbf{q}$. This expression is obviously difficult to obtain since $\mathbf{O}(\mathbf{q})$ and $\mathbf{R}(\mathbf{q})$ move on the robot and on the obstacle when $\mathbf{q}$ varies. The motion even depends on the shapes of the body and of the obstacle (Figure 2). However, we are going to show that the relative motion of the closest point on the robot and on the obstacle can be omitted.

Let us consider a reference frame $(\mathbf{B}(\mathbf{q}), \mathbf{e}_1(\mathbf{q}), ..., \mathbf{e}_d(\mathbf{q}))$ attached to the body we are considering. $d = 2$ or 3 is the dimension of the workspace. Then $\mathbf{R}(\mathbf{q}) = \sum_{l=1}^{d} \rho_l(\mathbf{q}) \mathbf{e}_l(\mathbf{q})$ where $(\rho_1(\mathbf{q}), ..., \rho_d(\mathbf{q}))$ are the local coordinates of $\mathbf{R}(\mathbf{q})$ on the body and

$$\frac{\partial \mathbf{R}}{\partial \mathbf{q}}(\mathbf{q}) = \sum_{l=1}^{d} \frac{\partial \rho_l}{\partial \mathbf{q}}(\mathbf{q}) \mathbf{e}_l(\mathbf{q}) + \sum_{l=1}^{d} \rho_l(\mathbf{q}) \frac{\partial \mathbf{e}_l(\mathbf{q})}{\partial \mathbf{q}}$$

This expression is in fact the law of composition of motions. The first term of the sum is the relative motion of $\mathbf{R}(\mathbf{q})$ on the body while the second term is the absolute motion of the point of the body coinciding with $\mathbf{R}(\mathbf{q})$. The relative motion of $\mathbf{R}(\mathbf{q})$ stays on the boundary of the body and is therefore orthogonal to vector $\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q})$:

$$(\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q}))^T \sum_{l=1}^{d} \frac{\partial \rho_l}{\partial \mathbf{q}}(\mathbf{q}) \mathbf{e}_l(\mathbf{q}) = 0$$

Applying the same reasoning to $\frac{\partial \mathbf{O}}{\partial \mathbf{q}}(\mathbf{q})$, we notice that:

$$(\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q}))^T \frac{\partial \mathbf{O}}{\partial \mathbf{q}}(\mathbf{q}) = 0$$

since the obstacles are fixed and thus the motion of the point coinciding with $\mathbf{O}$ on the obstacle is zero and the relative velocity of $\mathbf{O}$ on the boundary of the obstacle is orthogonal to vector $\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q})$.

Expression (4) thus becomes:

$$\frac{\partial d_{ij}}{\partial \mathbf{q}}(\mathbf{q}) = \frac{(\mathbf{R}(\mathbf{q}) - \mathbf{O}(\mathbf{q}))^T}{\|\mathbf{O}(\mathbf{q}) - \mathbf{R}(\mathbf{q})\|} \frac{\partial \mathbf{R}_{\in body}}{\partial \mathbf{q}}$$

where $\frac{\partial \mathbf{R}_{\in body}}{\partial \mathbf{q}} = \sum_{l=1}^{d} \rho_l(\mathbf{q}) \frac{\partial \mathbf{e}_l(\mathbf{q})}{\partial \mathbf{q}}$ is the absolute velocity induced by variations of $\mathbf{q}$, of the point of the body coinciding with $\mathbf{R}(\mathbf{q})$.

Therefore, computing the gradient of the configuration space potential field does not require the closed-form expression of the potential function. We only need to know the closest points between each body of the robot and each obstacle, the variation of the potential value w.r.t. this distance and the velocities of a point of a body implied by variations of the configuration variables. These latter velocities are easy to compute as illustrated by the following example.

Example: a robot in the plane

Let us consider a mobile robot in the plane composed of one body and subject to the action of an obstacle, represented in Figure 2. The configuration of the robot is denoted by $\mathbf{q} = (x, y, \theta)$. The potential generated by the obstacle is given by the expression: $U(\mathbf{q}) = f(d(\mathcal{B}(\mathbf{q}), \mathcal{O}))$ where $\mathcal{B}(\mathbf{q})$ and $\mathcal{O}$ are the volumes in the workspace occupied respectively by the robot in configuration $\mathbf{q}$ and the obstacle.

If ρ_1, ρ_2 are the local coordinates of $\mathbf{R}$ in the local reference frame $(\mathbf{B}(\mathbf{q}), \mathbf{e}_1(\mathbf{q}), \mathbf{e}_2(\mathbf{q}))$ of the robot, the position of the point coinciding with $\mathbf{R}$ on the robot is the following:

$$\mathbf{R}_{\in body} = \begin{pmatrix} x + \rho_1 \cos\theta - \rho_2 \sin\theta \\ y + \rho_1 \sin\theta + \rho_2 \cos\theta \end{pmatrix}$$

hence,

$$\frac{\partial U}{\partial \mathbf{q}}(\mathbf{q}) = \frac{f'(d)}{d} \begin{pmatrix} \delta_1, & \delta_2, & \delta_2(\rho_1 \cos\theta - \rho_2 \sin\theta) \\ & & -\delta_1(\rho_1 \sin\theta + \rho_2 \cos\theta)) \end{pmatrix}$$

with $\mathbf{O} = (O_1, O_2)^T$, $\mathbf{R} = (R_1, R_2)^T$, $\delta_1 = R_1 - O_1$, $\delta_2 = R_2 - O_2$ and $d = d(\mathcal{B}(\mathbf{q}), \mathcal{O})$.

IV. Optimizing Robot-Obstacle Interaction Computations

In the previous section, we have explained how to compute the gradient of the potential field generated by an obstacle on a body of the robot when this potential field depends on the distance between the body and the obstacle. The reactive trajectory deformation approach we have developed successively applies the four following steps to the current discretized admissible trajectory:

1) given a set of obstacles detected by on-board sensors, find the first collision on the trajectory,

2) choose an interval centered on the first collision and not containing the current position of the robot,
3) compute the gradient of the configuration space potential field along this interval,
4) apply the deformation process to get a new trajectory.

During steps 1 and 3, numerous computations need to be performed: in the worst case, for each body of the robot, each sample configuration along the discretized trajectory needs to be checked for collision (step 1) or taken into account for potential gradient calculation (step 3) with each obstacle. We denote by n_s the number of sample configurations, n_o the number of obstacles detected and n_b the number of bodies of the robot. The complexity without optimization is therefore equal to $n_s \times n_o \times n_b$.

In step 1 n_s is proportional to the distance on which collisions are checked on the current trajectory, starting from the current robot position. In step 3, n_s is proportional to the length of the deformation interval. This number can be very large if the discretization step of the trajectory is very small as it is the case when navigating in very cluttered environments.

In reactive obstacle avoidance, we usually define two distances:

- a desired clearance to obstacles involved in the collision checking step,
- a distance of influence above which obstacles have no influence on the trajectory, involved in the potential gradient computation step.

To simplify notations, we will denote by ρ_{infl} the distance involved in each step, even though it can have different values. This corresponds to:

- considering that the robot is in collision if the distance between a body and an obstacle is lower than ρ_{infl}
- imposing that function f in Equation (3) satisfies: for any $d \geq \rho_{infl}, \quad f(d) = 0$

With this reasonable constraint, each configuration along the discretized trajectory is at a distance lower than ρ_{infl} to few obstacles. Therefore most pairs *sample configuration - obstacle* give rise to no interaction.

In this section, we are going to present an algorithm that enables us to prune most useless *sample configuration - obstacle* pairs both in collision checking and potential field gradient computation.

A. Principle of the Robot-Obstacle interaction filtering algorithm

The principle of this algorithm consists in taking advantage of the spatial coherence along the trajectory. Between two successive sample configurations of the robot, the distance between a body and an obstacle changes by less than the maximum distance traveled by each point of the body between both configurations.

This property enables us to update a lower bound of the distance between a body and an obstacle without recomputing the exact distance. Only obstacles the distance lower bound of which is lower than ρ_{infl} are taken into account. We will show how to manage this list of distances between obstacles and a body.

B. Lower bound of the distance between a body and an obstacle

Let us consider two configurations $\mathbf{q}_1$ and $\mathbf{q}_2$ of the robot. Let us define φ_i as the rigid-body transformation moving any point of body $\mathcal{B}_i$ in configuration $\mathbf{q}_1$ to the same point in configuration $\mathbf{q}_2$:

$$\mathcal{B}_i(\mathbf{q}_2) = \varphi_i(\mathcal{B}_i(\mathbf{q}_1))$$

and let us denote by Δ an upper bound of the distance traveled by the points of body $\mathcal{B}_i$ between configurations $\mathbf{q}_1$ and $\mathbf{q}_2$:

$$\text{for any } \mathbf{R} \in \mathcal{B}_i(\mathbf{q}_1) \quad \|\varphi_i(\mathbf{R}) - \mathbf{R}\| \leq \Delta$$

Then, we have the following property.

Property 1: For any compact subset $\mathcal{O} \subset \mathcal{W}$ and for any d_1,
if

$$d(\mathcal{B}_i(\mathbf{q}_1), \mathcal{O}) \geq d_1$$

then

$$d(\mathcal{B}_i(\mathbf{q}_2), \mathcal{O}) \geq \max(d_1 - \Delta, 0)$$

Proof: If $d_1 - \Delta \leq 0$, the conclusion is obvious. If $d_1 - \Delta > 0$, for any $\mathbf{O} \in \mathcal{O}$ and any $\mathbf{R}_1 \in \mathcal{B}_i(\mathbf{q}_1)$,

$$d_1 \leq \|\mathbf{R}_1 - \mathbf{O}\|$$

For any $\mathbf{O} \in \mathcal{O}$ and any $\mathbf{R}_2 \in \mathcal{B}_i(\mathbf{q}_2)$,

$$\begin{aligned} \|\mathbf{R}_2 - \mathbf{O}\| &= \|R_2 - \varphi_i^{-1}(R_2) + \varphi_i^{-1}(R_2) - \mathbf{O}\| \\ &\geq \big|\|R_2 - \varphi_i^{-1}(R_2)\| - \|\varphi_i^{-1}(R_2) - \mathbf{O}\|\big| \end{aligned}$$

by triangular inequality. As $\varphi_i^{-1}(R_2) \in \mathcal{B}_i(\mathbf{q}_1)$,

$$d_1 \leq \|\varphi_i^{-1}(R_2) - \mathbf{O}\|$$

and by definition of Δ,

$$\|R_2 - \varphi_i^{-1}(R_2)\| \leq \Delta$$

Thus, as $d_1 > \Delta$,

$$\|R_2 - \varphi_i^{-1}(R_2)\| \leq \|\varphi_i^{-1}(R_2) - \mathbf{O}\|$$

and

$$\begin{aligned} \|\mathbf{R}_2 - \mathbf{O}\| &\geq \|\varphi_i^{-1}(R_2) - \mathbf{O}\| - \|R_2 - \varphi_i^{-1}(R_2)\| \\ &\geq d_1 - \Delta \end{aligned}$$

Therefore, $d(\mathcal{B}_i(\mathbf{q}_2), \mathcal{O}) \geq d_1 - \Delta$.

■

This property is the core of our filtering algorithm. The idea is to maintain for each body, a sorted list of lower bounds of distances between obstacles and the body. When the list is initialized, it contains exact distances to obstacles. Then at each sample configuration $\mathbf{q}_s$, the list is updated by subtracting the maximal traveled distance Δ between $\mathbf{q}_s$ and $\mathbf{q}_{s+1}$ to each of its elements. That is, obstacles getting further from the trajectory are treated as if they were getting closer.

Lower bounds that reach ρ_{infl} (and only these) have a twofold consequence:

- they correspond to obstacles that might have an influence and as such, these obstacles are treated in Robot-Obstacle interaction computations.
- their exact distance to the body is recomputed, and they are re-inserted within the partially sorted list, in order to maintain it sorted.

Algorithm : Robot obstacle interaction filtering

```
/* Initialization of lists */
s ← 1
/* loop over bodies of the robot */
for i in {1, ..., n_b} {
    for j in {1, ..., n_o} {
        /* compute distance to obstacle */
        compute d(B_i(q_s), O_j)
        /* one array per body */
        dist[j, i] ← d(B_i(q_s), O_j)
    }
    /* sort array using quick-sort algorithm */
    quicksort(dist[., i])
}
/* loop over trajectory interval */
while (s ≤ s_max /* last sample config */) {
    /* loop over bodies of the robot */
    for i in {1, ..., n_b} {
        l ← 1;
        while (dist[l, i] ≤ ρ_infl) {
            j ← index of l-th obstacle in list dist[l, i]
            compute interaction between O_j and B_i
            /* recompute exact distance to this obstacle*/
            dist[l, i] ← d(B_i, O_j)
            l ← l + 1
        }
        l_infl ← l
        /* sort first l_infl elements of array of distances */
        insertion_sort(dist[., i], l_infl)
        Δ[i] ← max dist traveled by B_i between q_s and q_{s+1}
        for j in {1, ..., n_o} {
            /* Subtract upper bound of traveled distance */
            dist[j, i] ← max(0, dist[j, i] − Δ[i])
        }
    }
    s ← s + 1
}
```

TABLE I

ALGORITHM TO FILTER OBSTACLES THAT HAVE NO INTERACTION WITH THE ROBOT, ALONG A TRAJECTORY.

The algorithm for a multi-body robot is described in details in Table I. Figure 3 illustrates its principle for a robot with a trailer. It presents the interactions between obstacles and the trailer between two successive sample configurations. The evolution of the list of distances highlights the following steps of the algorithm:

- Initialization of the list
- Subtraction of the upper bound of the traveled distance Δ
- Computation of exact distances to obstacles for lower bounds lower than ρ_{infl}
- Sort of this exact distances within the sorted list

V. EXPERIMENTAL RESULTS

We have implemented the algorithm presented in Section IV to mobile robot Hilare2 towing a trailer. This nonholonomic system of degree two benefits from the nonholonomic trajectory deformation method presented

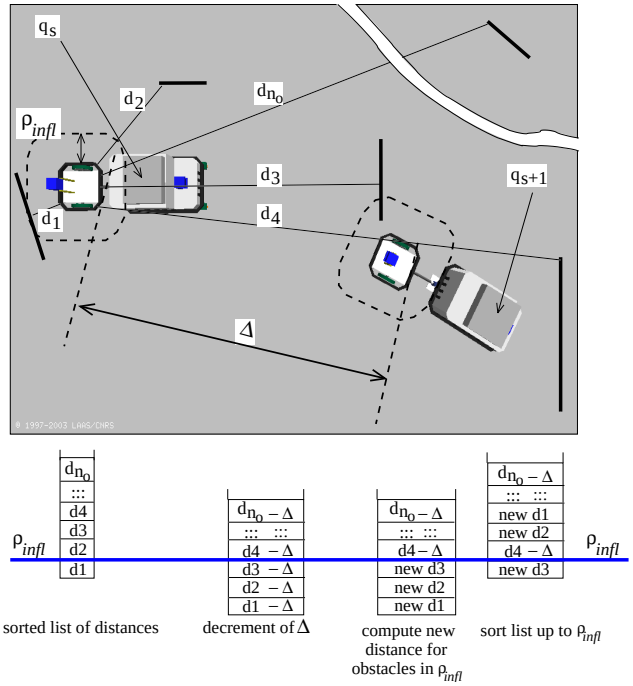

Fig. 3. **Illustration of the filtering algorithm.** Only interactions between obstacles and the trailer are presented. Δ is an upper bound of the distance traveled by the points of the trailer. Distances to obstacles are computed when the robot is at $\mathbf{q}_s$ and stored in a sorted list. Each element of the list is decreased by Δ. Exact distances to obstacles are then recomputed for elements lower than ρ_{infl}. New distances are eventually inserted in the list in order to keep it sorted.

in Section II. However, since this robot evolves in very cluttered environments, the time of computation of Robot-Obstacle interactions can be very important in the absence of optimization.

To evaluate the benefits of the optimization algorithm, we have run several iterations of the trajectory deformation process, with and without optimization, in the same environment and on the same trajectory. To make sure that the computations are exactly the same in both cases, we have run the experiments in our simulation environment. The robot is equipped with two laser scanners, the data of which are simulated given the map of the environment and the position of the robot. The trajectory discretized into 2500 sample configurations is represented in Figure 4. Table II reports the average computation times for 15 iterations in both cases. The unit of time is the millisecond. However, times are over-estimated since simulation requires more computations than real experiments. These figures have value for comparison only.

From these experiments, we can notice that although the optimization algorithm leads to new computations such as:

- distance between robot and obstacles,
- sorting an array in disorder: we use the quick sort algorithm for this part, the complexity of which is $O(n \log(n))$,
- sorting a partially ordered array: we use the insertion sort algorithm, the complexity of which is $O(n^2)$ on

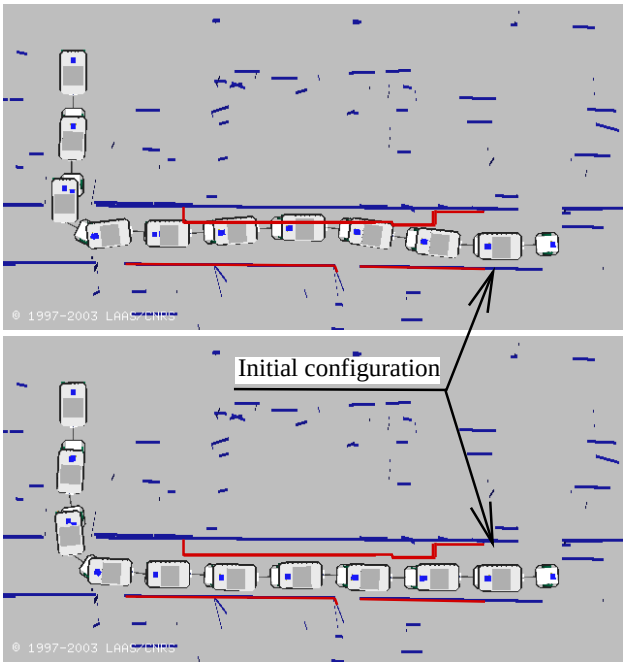

Fig. 4. The trajectory before and after applying the nonholonomic trajectory deformation process. 15 iterations have been applied with and without optimization of the Robot-Obstacle interaction computations. The robot is at the beginning of the trajectory on the right.

random lists, but which is very efficient when a part of the list is already sorted,

- upper bound of the traveled distance between two configurations,

the overall time of computation is considerably reduced with the filtering algorithm.

Robot	"Collision Checking" Time		"Potential Computation" Time	
	no-optim	optim	no-optim	optim
Hilare2 w/ trailer	1697	119	2552	413
Hilare2	636	20	1220	348
	Deformation Computation Time			
Hilare2 w/ trailer	2960			
Hilare2	2000			

TABLE II

COMPARISON OF TIME OF COMPUTATION OF ROBOT-OBSTACLES INTERACTIONS, WITH AND WITHOUT THE OPTIMIZATION ALGORITHM.

The numerous experiments we have carried out on our real robot since we have integrated the filtering algorithm into the trajectory deformation scheme confirm the huge gain in performance, in various type of environments and for several deformation interval lengths.

VI. CONCLUSION

This paper deals with the computation of Robot-Obstacle interactions in the context of nonholonomic trajectory deformation for mobile robots. It is shown that the potential field of the trajectory in the configuration space can be computed without any closed-form expression.

An algorithm to optimize the computation of Robot-Obstacles interactions is then presented. It takes advantage of spatial coherence to filter obstacles outside an influence zone around the robot. This algorithm can be applied to multi-body systems and to obstacles of any shape.

We have carried out experiments with robot Hilare 2 towing a trailer to evaluate the performance of this algorithm. Since the robot evolves in very cluttered environments, both the number of sample configurations and the number of obstacles are very large. Results show that the Robot-Obstacles interaction processes are not the bottleneck of the trajectory deformation method anymore.

Acknowledgment: This work has been partially supported by the European Project MOVIE (IST-2001-39250) and by the CNRS interdisciplinary program ROBEA.

REFERENCES

[1] D. Baraff. *Dynamic Simulation of Non-Penetrating Rigid Bodies*. PhD thesis, CUCS, Ithaca, 1992. Cornell Computer Science Technical Report 92-1275.

[2] O. Brock and O. Khatib. Real-time replanning in high dimensional configuration spaces using sets of homotopic paths. In *International Conference on Robotics and Automation*, pages 550–555, San Francisco, CA, April 2000. IEEE.

[3] E. Ferré and J.-P. Laumond. An iterative diffusion algorithm for part disassembly. In *IEEE International Conference on Robotics and Automation*, pages 3149–3154, New Orleans, Louisiana, April 2004.

[4] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33, March 1997.

[5] M. Kallmann and M. Mataric. Motion planning using dynamic roadmaps. In *IEEE International Conference on Robotics and Automation*, New Orleans, Louisiana, April 2004.

[6] M. Khatib, H. Jaouni, R. Chatila, and J.-P. Laumond. Dynamic path modification for car-like nonholonomic mobile robots. In *International Conference on Robotics and Automation*, pages 2920–2925, Albuquerque, NM, April 1997. IEEE.

[7] F. Lamiraux, D. Bonnafous, and C. Van Geem. *Control Problems in Robotics*, chapter Path Optimization for Nonholonomic Systems: Application to Reactive Obstacle Avoidance and Path Planning, pages 1–18. Springer, 2002.

[8] F. Lamiraux, D. Bonnafous, and O. Lefebvre. Reactive path deformation for nonholonomic mobile robots. to appear in the IEEE Transactions on Robotics.

[9] E. Larsen, S. Gottschalk, M. Lin, and D. Manocha. Fast distance queries with rectangular swept sphere volumes. In *International Conference on Robotics and Automation*, pages 3719–3726, San Francisco, CA, April 2000. IEEE.

[10] J. Minguez and L. Montano. Nearness diagram (nd) navigation: collision avoidance in troublesome scenarios. *IEEE Transactions on Robotics and Automation*, 20(1):45–59, Feb 2004.

[11] J. Minguez, L. Montano, and J. Santos-Victor. Reactive navigation for non-holonomic robots using the ego-kinematic space. In *International Conference on Robotics and Automation*, pages 3074–3080, Washington D.C., May 2002. IEEE.

[12] S. Quinlan and O. Khatib. Elastic bands: Connecting path planning and control. In *International Conference on Robotics and Automation*, pages 802–807, Atlanta, GA, May 1993. IEEE.

[13] S. Redon, Y. Kim, M. Lin, and D. Manocha. Fast continuous collision detection for articulated models. In G. Elber, N. Patrikalakis, and P. Brunet, editors, *Symposium on Solid Modelling and Applications*, 2004.