A Reactive Path Planning Approach for a Four-Wheel Robot by the Decomposition Coordination Method

Hala El Ouarrak and Mostafa Rachik
Department of mathematics, Computer Sciences
Faculty of Science Ben M'Sik
Av Driss El Harti B.P 7955, Sidi Othmane, Casablanca, Morocco
Telephone:+212619428458
Email: h.elouarrak@gmail.com

Ibrahim Sanou, Fatiha Akef and Mohammed Mestari
Laboratory of Signals Distributed Systems
and Artificial Intelligence (SSDIA)
ENSET Mohammedia
Av Hassan II Mohammedia, Morocco
Email: mestari@enset-media.ac.ma

***Abstract*—In this paper, we discuss the problem of safe navigation by solving a non-linear model for a four-wheel robot while avoiding the upcoming obstacles that may cross its path using the Decomposition Coordination Method (DC). The method consists of first, choosing a non-linear system with the associated objective functions to optimize. Then we carry on the resolution of the model using the Decomposition Coordination Method, which allows the non-linearity of the model to be handled locally and ensures coordination through the use of the Lagrange multipliers. An obstacle-avoidance algorithm is presented thus offering a collision-free solution. A numerical application is given to concert the efficiency of the method employed herein along with the simulation results.**

I. Introduction

The study of autonomous systems and mainly the robot's behaviour has known an increasing amount of research in the recent century focused mainly on autonomous navigation and tracking of nonlinear systems [1], [2]. Many significant results have been adopted concerning nonlinear systems, opening the way to study more complex mobile systems [3], [4].

That being said, methods for solving non-linear systems often require very complex mathematical tools, which makes the convergence very difficult when the size of the problem raises. The search for new approaches and efficient solutions remains an active research venture. Many methods using genetic algorithms have been presented in numerous ways to resolve scalar optimization problem (SOP) [5]-[6]. Nonetheless, the convergence of such methods to optimal Pareto front is very strenuous [7], when the constraints of the problem are not easy to satisfy or when the objective space is non-convex. In addition, these algorithms determine any bound of optimal Pareto front of the problem. On the other hand, the path planning problem [8], [9], [10] and specifically, the collision free path planning problem has known a noticeable improvement and a recurrent interest among the scientific society. The previous work categorizes into two different approaches:

- The reactive approach [12], [13] which offers a fast reactivity to avoid obstacles within a dynamic environment with a low amount of computation but without requiring the knowledge of the robot's surrounding.
- The deliberative approach [14] has been centrally quickened by the industrial use of robot's arm manipulator ever since 1961. The objective is the global planning of the motion strategy based on a model of the world, that will allow the robot to navigate from its current position to a desired final position.

The approach adopted herein is the reactive approach for it doesn't require the full knowledge of the environment's information and for the fast reactivity it provides. Our main objective is to find the optimal control input that would allow the robot to navigate from an initial state q_0 and reach the final desired state q_d in the most optimal way within a dynamic environment. We calculate the control sequence by first providing a model for the robot that integrates the kinematics and dynamics constraints. The model is in most cases a nonlinear system that can't be linearized. We associate several objective functions to optimize. We then proceed to the resolution of the multi-objective optimization problem by the use of the Decomposition Coordination method (DC) [15]. One of the best features of this approach the local treatment of the nonlinearity of the model which reduces the computing time, thus offering a fast reactivity to answer the real-time constraint. The DC Method consists of the conversion of the nonlinear system along with the associated optimization problem to an equivalent scalar optimization problem (SOP) with a single objective (cost) function. This conversion is achieved by the use of the minimax method that we will present later in this paper. To solve such SOP problem, we start by transforming the differential equations into equivalent difference equations that are then computed by discrete-time units

This method can be applied to a various type of optimization

978-1-5386-0549-3/17/$31.00 ©2017 IEEE

problems and facilitate the implementation on an Analog Neural Network (ANN) which one of the propitious applications that it offers [17].

This work is divided as follows: In section II we present the chosen model of the robot. The section III is mainly focused on the formulation of the problem and the process of conversion into SOP problem. In section IV we start by analyzing the problem, and we introduce the DC method as a solution for the nonlinear system. We provide sufficient conditions for the stability and convergence of the algorithm by using two theorems previously presented in [15]. Then in section V, we propose the resolution of a particular case of a four-wheel robot, and we give the simulation results which consolidates the theoretical approach presented herein. The conclusion is contained in section VI.

II. MODEL OF THE ROBOT

The aim of this section is the presentation of the robot's model. Many models have been studied in the literature vary from the most simple robot (one directional robot) to the most complicated one (humanoid robots). For a matter of simplicity, we work under the following assumptions: the workspace where the robot navigates is planar and the rolling is slip-free. We also assume the velocity vector to be null when the wheel is in direct contact with the surface at a geometrical point (tire deformation is neglected) [18].

Let us consider O of coordinates (x, y), as the central point of the rear axis, ϕ be the robot heading orientation, φ the steering angle and D the distance between the front wheels and rear axes. Let $q = (x, y, \phi, \varphi)^T$ be the state vector of the robot and $u = (\nu, \eta)^T$ the velocity (See Fig.1). The robot can only navigate in a perpendicular direction to its rotation axis [18] under the Assemptions presented above. One of the most import constraints to consider is the nonholonomic constraints which we can present as follows [19], [20] :

$$\dot{q} = \begin{pmatrix} \cos(\phi) & 0 \\ \sin(\phi) & 0 \\ \frac{1}{D}\tan(\varphi) & 0 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} \nu \\ \eta \end{pmatrix} \tag{1}$$

Another crucial aspect we need to consider is the dynamics of the robot. Many models have been studied in the literature. However, we will use a more simple representation for this case study by adding a parameter γ which would offer better results and the most optimal solution [21]. We have:

$$\dot{q} = \frac{c}{m}u - \gamma q \tag{2}$$

The model we present below shows both kinematics and dynamic constraints which offer a genuine representation of the robot's motion. We present the nonlinear model for the four-wheel robot as follows [22]:

$$\dot{q} = B(q) \times u - \gamma \times q \tag{3}$$

where: $B(q) = \frac{c}{m}\begin{pmatrix} \cos(\phi) & 0 \\ \sin(\phi) & 0 \\ \frac{1}{D}\tan(\varphi) & 0 \\ 0 & 1 \end{pmatrix}$ is the control matrix. $q = (x, y, \phi, \varphi)^T$ and $u = (\nu, \eta)^T$ are respectively the state vector and the control input of the nonlinear system. m and c are respectively the mass and the electro-mechanical transmission coefficient [21]. To compute the discrete-time DC Method, we must convert the nonlinear continuous model (3) into a discrete time model with respect to a specific format of the NECMOP seen in [15], using the forward Euler rule we then obtain:

$$q_{k+1} = B(q_k)u_k + (1 - \gamma\delta t)q_k = f(q_k, u_k) \tag{4}$$

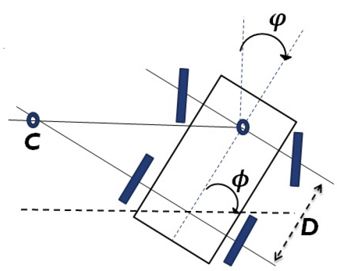

Fig. 1. Four wheel robot

III. STATEMENT OF THE PROBLEM

In this section we examine the nonlinear discrete-time system presented as follows:

$$\begin{cases} q_{k+1} = f(q_k, u_k) \\ q_0 \quad given \end{cases} \tag{5}$$

Where $q_k \in IR^n$ and $u_k \in IR^m$. q_k and u_k are respectively the state and the control input of the system at time k. Our objective is to allow the robot to navigate to the desired state by figuring out the optimal control sequence which enables this outcome at time k while ensuring a collision-free trajectory. To settle the incongruent objectives functions, we associate a weight to each function. These weights are set according to the constraints of the deciding entity. Therefore, solving this problem depends on finding the set of optimal states under the constraint of a single objective function. As mentioned earlier, we make use of the Minimax method [15] to transform the problem from a multi-objectives optimization problem into a SOP problem. This approach gives the smallest value of the maximum values of all the objective functions J_i. We define ω_k as the weight of the k component with $\sum_{k=0}^{p} \omega_i = 1$, and the objective functions $(J_1, J_2, ..., J_p)$ such as:

$$E(q, u) = max_{1 \leq i \leq P}\{w_i J_i(q, u)\} \tag{6}$$

We associate the optimisation problem (6) to the non-linear system (5) thus obtaining the following SOP:

$$\begin{cases} min_{\{U_l^*/0 \leq l \leq N-1\}} E(q, u) \\ s.t \quad q_{k+1} = f(q_k, u_k) \\ with \quad q_0 = q(0) \quad given \end{cases} \tag{7}$$

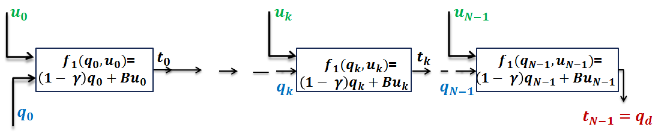

Fig. 2. decomposition principle

where $q_k = (x_k, y_k, \Phi_k, \varphi_k)^T$ and $u_k = (\nu_k, \eta_k)^T$ are respectively the state vector and the control input at time k. Solving such complex system (7) proves difficult due to the great amount of computation it requires which may increase exponentially.

IV. ANALYSIS OF THE PROBLEM

In this section we propose a solution to the problem (7) by using the DC method. We proceed to the decomposition of the system into a group of N interconnected subsystems organised into an easy serial configuration(See Fig.2), where t_k is the output for the subsystem k.

$$\begin{cases} t_k = f(q_k, u_k), \quad k = 0, \dots, N-2 \\ q_k = t_{k-1}, \quad k = 0, \dots, N-1 \end{cases} \tag{8}$$

Therefore, the problem (7) can be written as follows:

$$\begin{cases} min_{\{u_k^* | 0 \le k \le N-1\}} E(q,u) \\ s.t \quad t_k = f(q_k, u_k) \\ with \quad q_0 = q(0) \quad given \end{cases} \tag{9}$$

With $q_k = (x_k, y_k, \Phi_k, \varphi_k)^T$ and $u_k = (\nu_k, \eta_k)^T$ are respectively the state vector and the control input at time k. We build the ordinary Lagrange function [15], to solve the derivation problem for the cost function (9):

$$\begin{cases} L_0 = \frac{1}{N} E(q,u) + \mu_0^T (f(q_0, u_0) - t_0) \\ L_k = \frac{1}{N} E(q,u) + \mu_k^T (f(q_k, u_k) - t_k) + \beta_k^t (q_k - t_{k-1}) \\ L_{N-1} = \frac{1}{N} E(q,u) + \mu_{N-1}^T (f(q_{N-1}, u_{N-1}) - q_d) \\ +\beta_{N-1}^t (q_{N-1} - t_{N-2}) \end{cases} \tag{10}$$

Where μ_k(n components) and β_k (n components) are the Lagrange multiplier vectors presented to take into consideration the equality constraints (8). By derivating the ordinary Lagrange function (10), we can transpose the equality-constrained minimization problem (9) into a set of differential equations. According to the KKT conditions [15], an equilibrium point (q_k^*,u_k^*,μ_k^*,β_k^*,t_k^*) , must satisfy the following equations:

$$\nabla_{q_k} L = \frac{1}{N} \frac{\partial E}{\partial q_k} + \mu_k^{*T} (1-\gamma) + \beta_k^{*T} = 0 \tag{11}$$

$$\nabla_{u_k} L = \frac{1}{N} \frac{\partial E}{\partial q_k} + \mu_k^* B = 0 \tag{12}$$

$$\nabla_{\mu_k} L = B u_k^* + (1-\gamma) q_k^* - t_k^* = 0 \tag{13}$$

$$\nabla_{t_k} L = -\mu_k^* + \beta_{k+1}^* = 0 \tag{14}$$

$$\nabla_{\beta_k} L = q_k^* + t_{k-1}^* = 0 \tag{15}$$

Thus to elucidate the equality constrained minimization problem (9) we need to resolve the associated system of differential equations (11) - (15).

A. *Decomposition-Coordination algorithm*

This method was introduced by [15], [16] it relies on a decomposition procedure for the treatment of the associated system of differential equations (11)-(15) into two levels (see Fig.3). The upper level utilizes equations (14) and (15) and fixes t_k and $_k$, which in turn are proposed to the lower level that runs equations (11)-(14). The resolution of equations (11)-(14) is performed locally to solve the wholeness of the problem (9). We then need to proceed to the discretization of the system of differential equations (11)-(15) using the forward Euler rule. The system of differential equations (11)-(15) can be turned into the system of difference equations:

$$\begin{cases} q_k^{(j+1)} = q_k^{(j)} + \lambda_q \left(\frac{1}{N} \frac{\partial E}{\partial q_k} + \mu_k^{(j)^T} (1-\gamma) + \beta_k^{(j)^T} \right) \\ u_k^{(j+1)} = u_k^{(j)} + \lambda_u \left(\frac{1}{N} \frac{\partial E}{\partial q_k} + \mu_k^{(j)} B \right) \\ \mu_k^{(j+1)} = \mu_k^{(j)} + \lambda_\mu \left(B u_k^{(j)} + (1-\gamma) q_k^{(j)} - t_k^{(j)} \right) \\ \quad for \quad k = 1, ..., N-1 \end{cases} \tag{16}$$

To guarantee the transmission of the necessary information to the operative of the lower level with a global outlook to the optimization process (i.e., satisfaction of all equations (11)-(15)), it is crucial to coordinate the two levels. This coordination lay foundation on the simultaneous use of $\beta_k^{(j)} (k = 1, ..., N-1)$ and $t_k^{(j)} (k = 0, ..., N-2)$ by the upper level. These parameters which constitutes the coordination parameters are thought-about as known within the lower level, thus enabling the local resolution of the system of difference equations (16) and the determination of the variables $q_k^*(t_k^{(j)}, \beta_k^{(j)})$, $u_k^*(t_k^{(j)}, \beta_k^{(j)})$ and $\mu_k^*(t_k^{(j)}, \beta_k^{(j)})$ which respectively satisfy equations (16). The

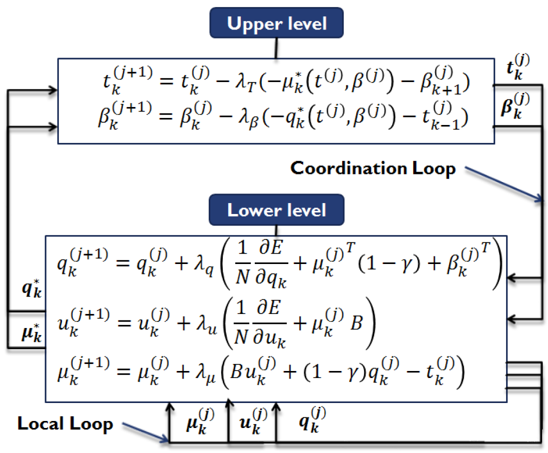

Fig. 3. Information transfer between the upper level and the lower level

results $q_k^*(t_k^{(j)}, \beta_k^{(j)})$ and $\mu_k^*(t_k^{(j)}, \beta_k^{(j)})$ are given to the upper level which verifies the correctness of the previously supplied information and corrects it if necessary. The coordination parameters $t_k^{(j)}$ and $\beta_k^{(j)}$ are given by the upper level resulting from the relations:

$$\begin{cases} t_k^{(j+1)} = t_k^{(j)} + \lambda_t \Big(\mu_k^*(t_k^{(j)}, \beta_k^{(j)}) + \beta_{k+1}^{(j)} \Big) \\ \quad k = 0, 1, ..., N-2 \\ \beta_k^{(j+1)} = \beta_k^{(j)} + \lambda_\beta \Big(q_k^*(t_k^{(j)}, \beta_k^{(j)}) + t_{k+1}^{(j)} \Big) \\ \quad k = 0, 1, ..., N-2 \end{cases} \tag{17}$$

We keep running the solution of the equations (16) until we achieve satisfactory coordination, i.e. fulfillment of the coordination equations (17). The algorithm thus presented is shown in Fig.3.

B. Stability analysis

To study the convergence of the DC method we prove that the resolution can be shortened to the solving of the coordinating level. We introduce the following notations to simplify the study of stability and convergence. Let us posit $v_k = \begin{pmatrix} q_k \\ u_k \end{pmatrix}$, and call $v_k^*(t_k^*, \beta_k^*)$,$\mu_k^*(t_k^*, \beta_k^*)$,t_k^* and β_k^* the solution sought. Also let us call $v_k^*(t_k^{(j)}, \beta_k^{(j)})$ and $\mu_k^*(t_k^{(j)}, \beta_k^{(j)})$ are the variables calculated at the lower level through the set of equations (16), in order to establish the local satisfaction of equations (11)-(13). we also define t_k^* and β_k^* as the coordination variables to be processed at the upper level through the algorithms of coordination (17). Let us also posit: $G_k = \begin{pmatrix} \nabla_{q_k} L \\ \nabla_{u_k} L \end{pmatrix}$, $P_k = \nabla_{\mu_k} L$, $R_k = \nabla_{\beta_k} L$, $H_k = \nabla_{t_k} L$ Let us posit:

$$\begin{cases} e_{v_k}^{(j)} = v_k^*(t_k^{(j)}, \beta_k^{(j)}) - v_k^*(t_k^*, \beta_k^*) \\ e_{\mu_k}^{(j)} = \mu_k^*(t_k^{(j)}, \beta_k^{(j)}) - \mu_k^*(t_k^*, \beta_k^*) \\ e_{t_k}^{(j)} = t_k^{(j)} - t^* \\ e_{\beta_k}^{(j)} = \beta_k^{(j)} - \beta^* \end{cases} \tag{18}$$

which illustrates the errors computed at iteration j of the coordination loop. Let us posit the following Lyapunov function:

$$\Theta(j) = \frac{1}{2} \sum_{k=0}^{N-1} {e_{t_k}^{(j)}}^T e_{t_k}^{(j)} + {e_{\beta_k}^{(j)}}^T e_{\beta_k}^{(j)} \tag{19}$$

And let us define:

$$\begin{cases} \Delta e_{t_k}^{(j)} = e_{t_k}^{(j+1)} - e_{t_k}^{(j)} = -\lambda H_k^{(j)} \\ \Delta e_{\beta_k}^{(j)} = e_{\beta_k}^{(j+1)} - e_{\beta_k}^{(j)} = -\lambda R_k^{(j)} \end{cases} \tag{20}$$

with $\lambda_t = \lambda_\beta = \lambda$. We calculate the variation of the Lyaponov function as follows:

$$\Delta\Theta = \Theta(j+1) - \Theta(j) = A(j)\lambda^2 + B(j)\lambda \tag{21}$$

where $A(j) = \sum_{k=0}^{N-1} \Delta{e_{t_k}^{(j)}}^T \Delta e_{t_k}^{(j)} + \Delta{e_{\beta_k}^{(j)}}^T \Delta e_{\beta_k}^{(j)}$ and $B(j) = \sum_{k=0}^{N-1} {e_{t_k}^{(j)}}^T \Delta e_{t_k}^{(j)} + {e_{\beta_k}^{(j)}}^T \Delta e_{\beta_k}^{(j)}$ We employ the fallowing theorems presented in [15]:

Theorem IV.1. *Let $e_{v_k}^{(j)}$, $e_{\mu_k}^{(j)}$, $e_{t_k}^{(j)}$ and $e_{\beta_k}^{(j)}$ be the errors calculated at the iteration j of the coordination loop. Then: $e_{v_k}^{(j)} \to 0$ and $e_{\mu_k}^{(j)} \to 0$ if $e_{t_k}^{(j)} \to 0$ and $e_{\beta_k}^{(j)} \to 0$*

Theorem IV.2. *The convergence is satisfied with sufficient conditions if one of the matrices $\frac{\partial G_k^*}{\partial v_k}$ $(k = 0, 1, \ldots, N-1)$ is positive definite and the others are only positive semi-definite and if $A(j) \neq 0$, λ should be chosen as: $0 \leq \lambda \leq |\frac{B(j)}{A(j)}|$*

C. Obstacle avoidance

In this section, we aim to calculate an optimal collision-free trajectory. The main purpose of this embedded solution is the automatic execution of the optimal control sequence that would allow the robot to navigate safely while avoiding the eventual obstacles it may collides. The algorithm that would provide such a trajectory can be expressed as follows:

- m: Number of obstacles.
- T_m: The optimal trajectory computed from the m^{th} obstacle.
- $q_0^{(m)}(x_0^{(m)}, y_0^{(m)}, \phi_0^{(m)}, \varphi_0^{(m)})$: Coordinates of the initial starting from the m^{th} obstacle.
- $q_{obs}(x_{obs}^{(m)}, y_{obs}^{(m)}, \phi_{obs}^{(m)}, \varphi_{obs}^{(m)})$: Coordinates of the m^{th} obstacle.

The algorithm runs as follows:

- Compute the optimal trajectory T_0 without any obstacle from the initial state $q_0^{(0)}$ to the final state q_F, with $m = 0$.
- As q_F is not reached yet :
  - if an obstacle appears:
    - Then $m = m + 1$
    - Correction of the trajectory, we put:
      - $x_0^{(m)} = x_{obs}^{(m)} + \delta x$
      - $y_0^{(m)} = y_{obs}^{(m)} + \delta y$
      - $\theta_0^{(m)} = \theta_{obs}^{(m)} + \delta\theta$
      - $\psi_0^{(m)} = \psi_{obs}^{(m)} + \delta\psi$
    - and then compute the optimal trajectory T_m from $q_0^{(m)}$ to q_F.
  - Otherwise continue to execute the previous trajectory.
- Optimal safe trajectory reached.

The information gathered would allow us to foresee the crash and identify its likely-hood. Since the external environment regroups many parameters namely: position, speed, acceleration etc... those parameters are crucial to the selection of a suitable control sequence for a collision-free path.

V. NUMERICAL APPLICATION

Consider the four wheel car model described by the non-linear discrete-time equation:

$$\begin{cases} q_{k+1} = f(q_k, u_k) = 0.99q_k + 0.1B(q_k)u_k \\ q_0 \quad is\ given \end{cases} \tag{22}$$

$$where \quad B(q_k) = \frac{c}{m}\begin{pmatrix} \cos(\phi_k) & 0 \\ \sin(\phi_k) & 0 \\ \frac{1}{D}\tan(\varphi_k) & 0 \\ 0 & 1 \end{pmatrix}$$

with $q_k = (x_k, y_k, \phi_k, \varphi_k)^T$ being the state vector and $u_k = (\nu_k, \eta_k)^T$ the control input at time k. We associate the following optimization problem:

$$\begin{cases} min\frac{1}{2}[\sum_{k=0}^{8} u_k^2] \\ s.t \quad q_{k+1} = f(q_k, u_k) \\ and \quad t_k = q_{k+1} \\ with \quad q_0 = q(0) \quad k = 0, ..., 8 \end{cases} \tag{23}$$

where q_0 is the given initial condition. The corresponding Lagrangian is:

$$L = \sum_{k=0}^{2} L_k \tag{24}$$

$$L_k = \frac{1}{2}\sum_{k=0}^{8} u_k^2 + \sum_{k=0}^{8}\mu_k^T(0.99q_k + 0.1Bu_k - t_k)$$

$$+\sum_{k=0}^{8}\beta^T(0.99q_k - t_{k-1}) \tag{25}$$

where $\mu_k(k = 0, ..., 8)$ and $\beta_k(k = 1, ..., 8)$ are the Lagrange multiplier vectors presented to take into consideration the equality constraints (8). The optimal set q_k^*, u_k^*, μ_k^*, $t_k^*(k = 0, ..., 8)$ and $\beta_k^*(k = 1, ..., 8)$ should fulfill these Lagrangian conditions of stationeries in relation with the variables and the Lagrange multipliers:

$$\left.\begin{aligned} \nabla_{\mu_k} L = \delta Bu_k^* + 0.99q_k^* - t_k^* = 0 \\ \nabla_{q_0} L = 0.99q_0^* + \mu_0^* = 0 \\ \nabla_{q_k} L = 0.99q_k^* + \mu_k^* + \beta_k^* = 0 \\ \nabla_{u_k} L = 0.1u_k^* + B^T\mu_k^* = 0 \end{aligned}\right\} \text{Lower Level} \tag{26}$$

$$\left.\begin{aligned} \nabla_{\beta_k} L = 0.99q_k^* - t_{k-1}^* = 0 \\ \nabla_{t_k} L = -\mu_k^* - \beta_{k+1}^* = 0 \\ \nabla_{t_2} L = \mu_2^* = 0 \end{aligned}\right\} \text{Upper Level} \tag{27}$$

As it is depicted in Fig3, the calculation of the above equations is distributed into two levels. Firstly, the upper level treats equations (27) and computes $t_0^{(j)}$ and $\beta_1^{(j)}$. Then, the lower level utilizes the equations (26) to solve the problem in a global way. The coordination parameters $t_0^{(j)}$ and $\beta_1^{(j)}$ are adjusted by the upper level abiding these equations:

$$t_0^{(j+1)} = t_0^{(j)} + \lambda_t\left(\mu_0^*(t_0^{(j)}) + \beta_1^{(j)}\right) \tag{28}$$

$$\beta_1^{(j+1)} = \beta_1^{(j)} + \lambda_\beta\left(q_1^*(\beta_1^{(j)}) + t_2^{(j)}\right) \tag{29}$$

where $t_0^{(j)}$ and $\beta_1^{(j)}$ are given. The parameters $u_0^*(t_0^{(j)})$, $\mu_k^*(t_0^{(j)})$and $q_k^*(\beta_k^{(j)})$ are calculated as follows, in accordance with (26)-(27):

$$\begin{cases} \nabla_{\mu_0} L = 0 \\ \nabla_{u_0} L = 0 \\ \nabla_{t_0} L = 0 \\ \nabla_{\beta_1} L = 0 \end{cases} \Rightarrow \begin{cases} q_0^* t \frac{1}{0.99}(t_0^{*(j)} 0.1Bu_0^*) \\ u_0^* = \frac{1}{0.1}B\mu_0^* \\ \mu_0^* = \beta_1^{*(j)} \\ q_1^* = \frac{1}{0.99}t_0^{*(j)} \end{cases} \tag{30}$$

We repeat the relationships (30) at each development of coordination parameters and until the equations (27)-(26) are fulfilled. To study the convergence, we must respect the completion of the two sufficient conditions for stability described in section IV. We then can write:

$$G_k = \begin{pmatrix} \nabla_{q_k} L \\ \nabla_{u_k} L \end{pmatrix} = \begin{pmatrix} (1 - \gamma\delta t)q_k + \mu_k + \beta_k \\ 0.1u_k + B\mu_k \end{pmatrix}, \quad V_k = \begin{pmatrix} q_k \\ u_k \end{pmatrix} \tag{31}$$

$$\text{for} \quad k = 0, ..., 8$$

The first condition is satisfied because all the matrices $\dfrac{\partial G_k^{*T}}{\partial v_k}$ $(k = 0, 1, \ldots, N-1)$ are positive definite, so they are also semi-definite, thereby satisfying the convergence conditions presented in section IV:

$$\frac{\partial G_k^{*T}}{\partial v_k} = \begin{pmatrix} 0.99 & 0 \\ 0 & 0.1 \end{pmatrix} \tag{32}$$

with $\gamma = 0.1$ then the matrices $\dfrac{\partial G_k^*}{\partial v_k}$ are positive semi definite. To validate the second condition we choose:

$$0 < \lambda < -\frac{B(j)}{A(j)} = \left|\frac{B(j)}{A(j)}\right| \tag{33}$$

We proceed to the simulation using the following numerical example: the initial state is set to $q_0 = (0, 0, 0, 0)$ and the final desired state to reach is set to $q_d = (5, 5, 0, 0)$. The results clearly shows a series of straight lines formed between q_0 and each $q_0^{(m)}$, which is the most optimal trajectory possible while

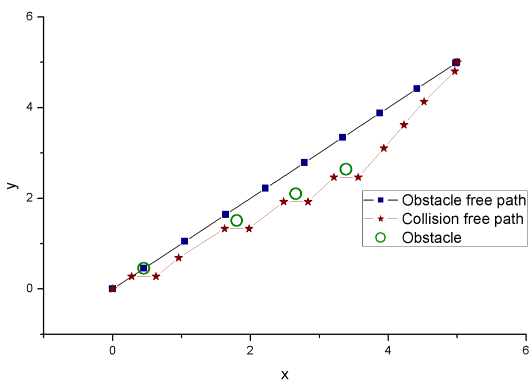

Fig. 4. Optimal trajectory obstacle free and the optimal collision free path from the initial state q_0 to the final state q_F

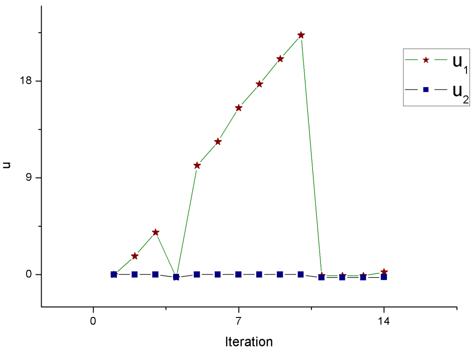

Fig. 5. Optimal control parameters $u_1 = \nu$ and $u_2 = \eta$ for the obstacle avoidance path

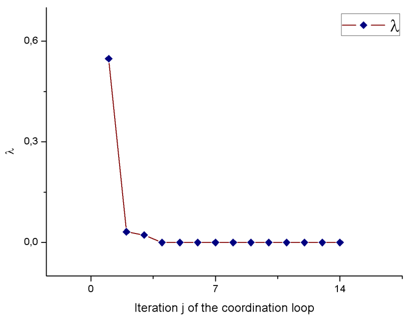

Fig. 6. Evolution of the adaptability coefficient λ adjusted according to the condition $0 < \lambda < -\frac{B(j)}{A(j)} = \left|\frac{B(j)}{A(j)}\right|$ in case (a) with four obstacles and case (b) with three obstacles

avoiding the obstacles. The variation of the control sequence u_k for each state q_i is illustrated in Fig.5, where $u_1 = v_k$ and $u_2 = \eta_k$. In Fig.5, we observe the change in the control occurs every time the robot is about to enter in collision with an obstacle. Finally, in Fig.6 we can easily see the convergence of the method after the third iteration, which guarantees a rapid responsiveness with a low computing time regardless of the complexity of the system.

VI. Conclusion

In this paper, we propose a new way for solving a non-linear discrete-time model for a four-wheeled robot with the Decomposition Coordination Method within a dynamic environment to ensure a collision free path. One of the best features of the DC method is the local treatment of the non-linearity of the system which is very useful for the case study of this article. The approach consists primarily, of the concepts parallel treatment, while providing effective coordination. We also proved the stability and convergence of the algorithm by satisfying the conditions presented in section IV. Moreover, we presented an efficient solution to the safe navigation problem using an obstacle avoidance algorithm. The Numerical application shown in this paper proves that the method employed here can be advantageously applied to SOP problem, especially for resolving problems that would require an amount of calculations that are hard to conduct through an overall approach. Another interesting features of the DC method is that it is easily adaptable to parallel computation and can, therefore, be implemented on an Analogue Neural Network. The future work will mainly consist of the implementation on a neural network and its applications.

References

[1] O. Derbali, *"Etude et simulation de l'apprentissage d'un robot mobile sur une tâche de grazing "*. Ph.D. dissertation, University dAngers, 2005.

[2] L. Huang, Y. S Lim, D. Li, C. E. L. Toeh, *"Design and Analysis of a Four-wheel Omnidirectional Mobile Robot"*. in 2nd Int. Conf. Auton. Robot. Agents, vol. 5, New zealand, 2004, pp. 2013.

[3] P. C. Müller, *"Robot dynamics and control"*. Automatica, vol. 28, no. 3, pp. 655656, 1992.

[4] K. Kozowski, D. Pazderski, I. Rudas, and J. Tar, *Modeling and control of a 4-wheel skid-steering mobile robot : From theory to practice,*. Poznan University of Technology, 2004.

[5] D. The Luc, *"Multi-Objective Linear Programming Problem"*. Springer, 2012, vol. 3, no. 1.

[6] G. Eichfelder, *Adaptive scalarization methods in multiobjective optimization*. SIAM J. OPTIM., vol. 19, no. 4, pp. 16941718, 2008.

[7] A. Abraham, L. Jain, and G. E. Robert, *"Evolutionary multiobjective optimization"*. Springer, 2005.

[8] L. Caeacciolo, A. D. Luca, and S. Iannitti, *Trajectory Tracking Control of a Four-wheel Differentially Driven Mobile Robot*. in 1999 IEEE Int. Conf. Robot. Autom. Detroit, Michigan: 1999 IEEE, 1999, pp. 26322638.

[9] G. Mester, *Motion Control of Wheeled Mobile Robots Modeling of the Wheeled Mobile Robots*. in 4th Serbian-Hungarian Jt. Symp. Intell. Syst. SISY, 2006, pp. 119130.

[10] B. D. Argall, *Learning Mobile Robot Motion Control From Demonstration and Corrective Feedback*. Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, 2009.

[11] T. Lozano-Perez, *Spatial planning: a configuration space approach*. AI Memo 605. Cambridge, Mass. MIT Artificial Intelligence Laboratory, 1980.

[12] M. Khatib, *"Sensor-based motion control for mobile robots"*. Ph.D. dissertation, LAAS-CNRS, 1996.

[13] P. Fiorini and Z. Shiller, *"Motion planning in dynamic environments using velocity obstacles"*. Int. J. Rob. Res., vol. 17, no. 7, pp. 760772, 1998.

[14] T. Lozano-Perez, *"Automatic planning of manipulator transfer movements"*. IEEE Trans. Syst. Man Cybern., vol. 11, no. 10, pp. 681698, 1981.

[15] M. Mestari, M. Benzirar, N. Saber, and M. Khouil, *"Solving nonlinear equality constrained multiobjective optimization problems using neural networks"*. IEEE Trans. neural networks Learn. Syst., vol. 26, no. 10, pp. 250020, oct 2015.

[16] M. Mestari, A. Namir, and J. Abouir, *"Switched capacitor neural networks for optimal control of nonlinear dynamic systems: Design and stability analysis"*. SAMS Journal, vol. 41, no. 3, pp. 559-591, 2001.

[17] M. Mestari, *"An Analog Neural Network Implementation in Fixed Time of Adjustable Order Statistic Filters and Applications"*. IEEE Transactions on Neural Networks, vol. 15, no. 3, May 2004, pp. 766-785.

[18] S. R. PETTI, *"Safe navigation within dynamic environments: a prtial motion planning approach"*. Ph.D. dissertation, 2007.

[19] Z. Song, D. Zhao, J. Yi, and X. Li, *"Robust motion control for nonholonomic constrained mechanical systems: sliding mode approach"*. in Proceeding. 2005, Am. Control Conf. 2005., 2005, pp. 28832888.

[20] K. Kozlowski and D. Pazderski, *"Modeling and control of a 4-wheel skid-steering mobile robot"*. Int. J. Appl. Math. Comput. Sci., vol. 14, no. 4, pp. 477496, 2004.

[21] M. Egerstedt, *"Control of Mobile Robots"*. [Online]. Available: https://class.coursera.org/conrob-002

[22] H. E. Ouarrak, A. Bouaine, M. Rachik and M. Mestari, *"Trajectory planning for a four-wheel robot using decomposition-coordination principle"*. in 2015 Third World Conference on Complex Systems (WCCS), Marrakech, Morocco, Nov 2015, pp. 1-6. doi: 10.1109/ICoCS.2015.7483265.