2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Autonomous robot navigation with a global and asymptotic convergence

Hugo Berti, Angel D. Sappa and Osvaldo E. Agamennoni

Abstract— **This paper presents improvements over the Dynamics Window Approach (*I-DWA*), used for computing in real time autonomous robot navigation. A novel objective function that includes Lyapunov stability criteria is proposed. It allows to guarantee a global and asymptotic convergence to the goal, resulting in a more simple and self-contained approach. Experimental results with simulated and real environments are presented to validate the capability of the proposed approach.**

I. INTRODUCTION

Autonomous robot navigation involves the real time achievement of user defined goal/s. The autonomy degree of a given robotic system fix or define both the capability of adaptation to environment changes and the abstraction level in which a given goal can be represented. For example, the achievement of a given goal in a static and known environment can be tackled with a *global planning strategy*. On the contrary, unknown or partially known environments, as well as dynamic environments, should be tackled by means of *reactive navigation strategies*. These reactive strategies allow solving unexpected events in real time, by means of the use of sensors in order to capture the surrounding environment.

Several autonomous robot navigation approaches were proposed during the last decades. Earlier techniques were based on the use of artificial potential field (e.g., [1], [2]). An attractive force produced by the goal drives the robot to the objective, while at the same time, repulsive forces produced by the obstacles keep the robot away from them. Since then, several improvements were introduced giving rise to more evolved techniques such as: *Virtual Field Histogram* (VFH) [3], *Curvature-Velocity Method* (CVM) [4] and *Dynamic Window Approach* (DWA) [5]. The CVM [4] and DWA [5] are the two more widely used approaches since a high speed navigation can be reached. They search for *control commands* (v, w) directly in the velocity space. Similarly, in [6] a trajectory space is used for searching the control commands (*steering angle* and *velocity*). In these cases, control commands are selected by maximizing an objective function, which includes criteria such as: speed, goal-directedness and safety. Constraints from robot and obstacles are incorporated in the velocity space. In spite of these advantages, a hard constraint of these techniques is that they ignore the way in which the robot approaches the goal, so convergence criteria are not considered.

Extensions to the original DWA have been proposed in [7], [8], [9] and [10], to mention a few. Reference [7] presents a Global-DWA to avoid the local minima problems by using connectivity information about the free space. However this global feature is never shown [10]. A Reduced-DWA, to speed up the translational velocity selection, is proposed by [8]. As a result a *dynamic line* is obtained, which requires less processing power. However, this velocity selection is not appropriate when the robot orientation to the goal is high (e.g., > 90 degrees). A more elaborated method, which integrates three different approaches (DWA, elastic band and NF1), is introduced in [9]. This integration mitigates the drawbacks of DWA. Finally, [10] also combines different elements from the original DWA to guarantee convergence. Although [9] and [10] guarantee convergence to the goal, none of them improve the original DWA, they add others approaches for compensating DWA's drawbacks. It gives as a result a more expensive and complex strategy.

Having in mind the aforementioned problems we propose a new compact autonomous navigation strategy as an improvement of the DWA. It is based on the velocity space and proposes an oriented to the goal, safe and efficient navigation. The incorporation of Lyapunov stability criteria, inside the kernel of DWA, is the novelty of our approach. Hence, a simple and self-contained approach is obtained. Stability criteria permit to evaluate the convergence to the goal.

The paper is organized as follow. Next section briefly describes the DWA, as well as its mathematical formalism. Section III presents the proposed navigation technique: I-DWA. Then, experimental results with both simulated and real environments are presented; at the same time, additional constraints required for extending the I-DWA to deal with differential drive robots are also introduced in Section IV. Finally, conclusions and future works are given in Section V.

II. DYNAMIC WINDOW APPROACH

As mentioned above, the proposed technique is based on the *Dynamic Window Approach* (DWA). Therefore, in this section a summary of DWA is given; more details about it can be found in [5].

The main distinctive feature of DWA is based on the fact that control commands (v, w) are directly selected in the velocity space. This space is bounded by constraints directly affecting the robot's behavior; some of those constraints are imposed by the environment (obstacle's configuration), while others are from robot's physical limitations (maximum velocity and acceleration).

This work has been partially supported by the Spanish Ministry of Education and Science under project TRA2004-06702/AUT. The second author was supported by The Ramón y Cajal Program.

H. Berti is with Facultad de Ingeniería, UNLPam, 6360 Gral. Pico, ARGENTINA hberti@ing.unlpam.edu.ar

A. D. Sappa is with Centro de Visión por Computador, 08193 Bellaterra, Barcelona, ESPAÑA angel.sappa@cvc.uab.es

O. E. Agamennoni is with Dto. de Ing. Eléc. y de Comp, Univ. Nac. del Sur, 8000 B. Blanca, ARGENTINA oagamen@uns.edu.ar

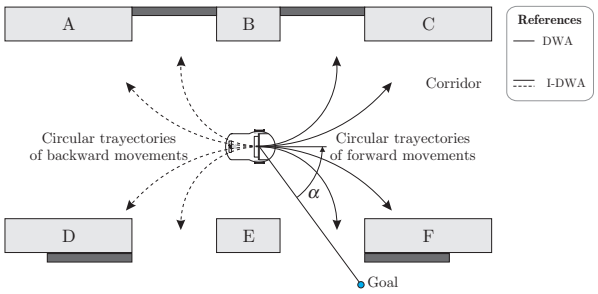

Fig. 1. Illustration of a robot navigation environment.

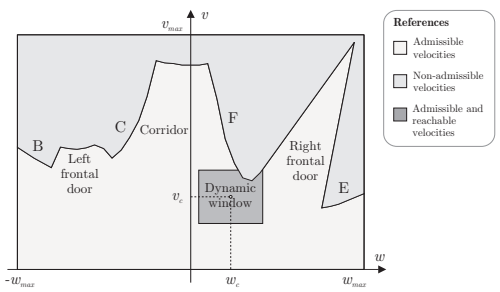

Fig. 2. Velocity map corresponding to the illustration presented in Fig. 1.

It is assumed that the robot moves with a constant velocity (v, w) during each control loop (e.g., [7], [11]). In other words, assuming a null acceleration the robot only moves with *circular trajectories* —with a constant curvature $c = w/v$, see illustration in Fig. 1—.

Obstacles near to the robot impose constraints over translational and rotational velocities —referred to as *admissible velocities*—. The maximum admissible velocity, over a given curvature, depends on the distance to the next obstacle over that curvature. The set of admissible velocities (V_a) is computed by means of a function $Dist$ that evaluates the distance to the nearest obstacle for a given curvature. This can be expressed as:

$$Dist\,(v,w) = \min_{obs \in OBS} dist\,(v,w,obs)\,, \tag{1}$$

where obs is an element from the set of obstacles OBS. The set V_a can be expressed as:

$$V_a = \left\{ (v,w) \middle| \begin{array}{l} v \leq \sqrt{2 \cdot Dist\,(v,w) \cdot \dot{v}_{max},} \\ w \leq \sqrt{2 \cdot Dist\,(v,w) \cdot \frac{\dot{w}_{max}}{c}} \end{array} \right\}, \tag{2}$$

where, $\dot{v}_{max}$ and $\dot{w}_{max}$ are the maximum translational and rotational accelerations respectively.

On the other hand, a set of velocities, referred to as *reachable velocities*, indicates those velocities that the robot can achieve during a control loop —velocities defining a *dynamic window* V_d—. The set of V_d is expressed as:

$$V_d = \left\{ (v,w) \middle| \begin{array}{l} \frac{v - v_c}{\Delta t} \in [-\dot{v}_{max},\, \dot{v}_{max}]\,, \\ \frac{w - w_c}{\Delta t} \in [-\dot{w}_{max},\, \dot{w}_{max}] \end{array} \right\}, \tag{3}$$

where, v_c and w_c are the current translational and rotational velocities, and Δt is the duration of the control loop.

Summarizing, the *search space* of the control commands is reduced to three kinds of constraints: (i) circular trajectories, (ii) admissible velocities, and (iii) reachable velocities. From the constraints imposed over the robot's velocities, a *resulting search space* (V_r) can be defined as:

$$V_r = V_p \cap V_a \cap V_d, \tag{4}$$

where V_p represents the whole space of *possible velocities* for the robot. It is defined by:

$$V_p = \{\,(v,w) |\, v \in [0, v_{max}]\,, w \in [-w_{max}, w_{max}]\}\,, \tag{5}$$

note that v, in the originally proposed DWA, is only defined for positive values, hence the robot cannot move backward.

Fig. 1 illustrates an example where the robot first goes through a corridor, then it has to cross a door in order to finally reach a user-defined goal. Fig. 2 presents the velocities involved in the computation of control actions for the situation presented in Fig. 1.

Finally, from the resulting search space (V_r), DWA selects the couple of velocities that maximize an objective function —different functions have been proposed in the literature [8], [11], among others—. The objective function includes terms that trade-off driving the robot at a high speed, oriented to the goal and far away from obstacles: *speed, goal-directedness* and *safety*. Therefore, the objective function is defined as:

$$G\,(v,w) = \mu_1 \cdot Speed\,(v) + \mu_2 \cdot Goal\,(w) + \mu_3 \cdot Dist\,(v,w)\,, \tag{6}$$

where, $\mu_i > 0$, $i = 1,2,3$, and $\sum_i \mu_i = 1$ are weighting factors for each one of those terms. The $Speed$ function is used to enforce a high speed navigation. It is defined as:

$$Speed\,(v) = v / v_{max}. \tag{7}$$

Nevertheless, the use of this function could suppose taking a wrong action under some particular conditions. For example, it happens when the robot's orientation has a high discrepancy with the goal $(\alpha > 90°)$; in this case the robot will move at a high speed away from the goal. On the other hand, the $Speed$ function does not take into account the closeness to the goal, thus when the robot is near the goal this function will promote a wrong action: fast navigation.

The $Goal$ function measures the alignment of the robot orientation with respect to the goal, defined with the parameter (α) (Fig. $3(left)$). It computes an orientation error, assuming that the robot moves with a constant velocity w during the interval of time of the control loop Δt. This function could be defined as (e.g., [4], [7], [8]):

$$Goal\,(w) = 1 - |\alpha - w \cdot \Delta t|\,/\pi. \tag{8}$$

Note that it does not include the angular closeness due to translational velocity; this drawback is emphasized in those cases where the robot is near the goal but with a wrong orientation (high α).

The $Dist$ function, presented in (1), represents the distance to the nearest obstacle over a circular trajectory with a curvature given by the velocities (v, w).

Reference [4] represents each term of the objective function as a piece-wise linear function, where the maximum

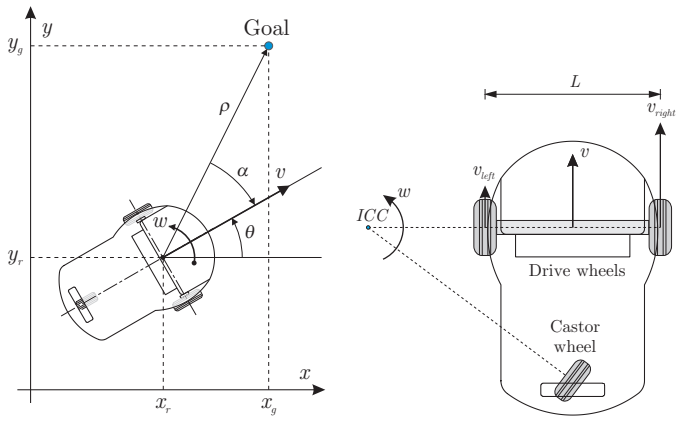

Fig. 3. $(left)$ Robot's representation in Cartesian and Polar coordinate systems. $(right)$ Kinematics characteristics of PIONEER 1 (differential drive robot).

value of (6) is computed by linear programming methods. On the contrary, in [5], [7] and [8], non-linear functions are adopted and the maximum value is computed by looking in a discrete space. Alternatively, [9] proposes the selective use of a precalculated lookup table; this allows to control any-shaped robot contours and a fast search for the maximum.

It has been proved that methods based on control commands space are appropriated to implement navigation strategies (e.g., [6], [7], [8], [9], [10]). They include the environment constraints (2) and the robot dynamics constraints, (3) and (5). Additionally, they state an objective function that imposes suitable behaviors: speed, goal-directedness and safety. However, other convergence criteria need also to be considered in order to evaluate the arrival to the goal.

III. PROPOSED METHOD (I-DWA)

In this section the proposed control strategy, which guarantees and characterizes the arrival to the goal, is presented. In addition, further improvements to the original DWA are introduced to avoid the inconveniences found in the terms of the objective function that impose a fast (7) and oriented to the goal (8) navigation. Next, kinematics equations used to model the robot's motion are introduced. Then, the proposed control law based on Lyapunov stability criteria is presented. Finally, a new objective function is given.

A. Kinematics Equation

Assuming the robot is represented by a point, its kinematics equations, in a Cartesian space, can be expressed as:

$$\begin{aligned} \dot{x} &= v \cdot \cos(\theta), \\ \dot{y} &= v \cdot \sin(\theta), \\ \dot{\theta} &= w, \end{aligned} \tag{9}$$

where θ defines the robot's orientation according to a global coordinate system(Fig. $3(left)$). These equations can be expressed in a polar coordinate system associated with the goal:

$$\begin{aligned} \dot{\rho} &= -v \cdot \cos(\alpha), \\ \dot{\alpha} &= -w + v \cdot \sin(\alpha)/\rho, \\ \dot{\theta} &= -\dot{\alpha}. \end{aligned} \tag{10}$$

Although the robot has been represented as a point, this model can be extended to different kinds of robots, for instance synchronous drive robots or differential drive (see Section IV). In these cases, a direct implementation could be to represent the robot rotation center as the reference together with its minimum robot's bounding circle.

B. Convergence analysis

This section presents an *ideal control law* (v_i, w_i) that allows driving the robot to the goal guaranteeing a global convergence. In order to do that we propose a candidate Lyapunov law involving two state variables (ρ, α) in a polar coordinate system:

$$V(\rho, \alpha) = V_1 + V_2 = \rho^2/2 + \alpha^2/2. \tag{11}$$

The time derivation of (11), over the trajectories defined by the set of kinematics equations (10), is expressed as:

$$\begin{aligned} \dot{V}(\rho, \alpha) &= \dot{V}_1 + \dot{V}_2 = \dot{\rho} \cdot \rho + \dot{\alpha} \cdot \alpha \\ &= -v_i \cdot \cos(\alpha) \cdot \rho + (-w_i + v_i \cdot \sin(\alpha)/\rho) \cdot \alpha. \end{aligned} \tag{12}$$

The sought convergence is reached by using a control law where the terms of $\dot{V}(\rho, \alpha)$ are always negative defined. Additionally, both velocity values should not be bigger than the maximum values. Thus, we propose a modification in the term related to a fast navigation (7) of the objective function:

$$v_i := k_v \cdot v_{max} \cdot \cos(\alpha) \cdot \tanh(\rho/k_\rho), \tag{13}$$

where, the function $\tanh(\rho/k_\rho) \to 1$ if $\rho \to \infty$, therefore a limit for the translation velocity is defined by v_{max}; k_ρ is a weighting factor that works when the robot approaches the goal, smoothing its speed reduction. Consequently, the selected velocity will increase according to the value of ρ, but it will be asymptotically bounded. An interesting point of the proposed function is that $\cos(\alpha)$ permits to consider the robot orientation according to the goal. Thus, the selected speed will be high when the robot orients to goal. At the same time, $\cos(\alpha)$ allows even backward movements ($\alpha > 90$).

Otherwise, the rotation velocity selection must evaluate the relative closeness originated by the translation velocity. In that sense, the following law is defined for selecting the correct rotation velocity:

$$\begin{aligned} w_i &:= k_\alpha \cdot \alpha + v_i \cdot \sin(\alpha)/\rho = \\ &= k_\alpha \cdot \alpha + k_v \cdot v_{max} \cdot \tanh(\rho/k_\rho) \cdot \sin(2 \cdot \alpha)/(2 \cdot \rho), \end{aligned} \tag{14}$$

where, $\frac{\tanh(\rho/k_\rho)}{\rho} \to 1/k_\rho$ if $\rho \to 0$, therefore the rotation velocity is bounded; k_α and k_v are positive weighting factors intended for obtaining the required robot behavior: k_α works over the angular error whereas k_v works over the distance error. From (14), the following relationship between these factors and maximum robot's velocities is obtained: $k_\alpha \leq (|w_{max}| - |k_v \cdot v_{max}/(2 \cdot k_\rho)|)/\pi$.

The next expression is obtained after including the control laws (13) and (14) in (12):

$$\begin{aligned} &\dot{V}(\rho, \alpha) = -\rho \cdot v_{max} \cdot \tanh(k_1 \cdot \rho) \cdot \cos^2(\alpha) - k_\alpha \cdot \alpha^2, \\ &\dot{V}(\rho, \alpha) \leq 0. \end{aligned} \tag{15}$$

Hence, the proposed Lyapunov function (11) is always non-incremental in time, as we were looking for. Additionally, this kind of function guarantees a global and asymptotical convergence to the goal (see [12]); numerically it can be expressed as:

$$\dot{V}(\rho,\alpha) = \dot{V}_1 + \dot{V}_2 \Rightarrow \left\{ \begin{array}{l} \rho(t) \\ \alpha(t) \end{array} \right. \rightarrow 0, \text{ if } t \rightarrow \infty . \quad (16)$$

This function is only valid when there are not obstacles in the environment and when the robot does not present acceleration limitations. The latter condition appears in the initial point when the robot needs to start the motion and reach the velocities stated in (13) and (14). These velocities take into account only the robot's position and orientation discrepancies with the given goal (ρ, α). Therefore, additional constraints, to avoid choosing unreachable velocities, should be imposed. Unreachable velocities are due to physical limitations in the robot's accelerations; in addition velocities that involve some collision risk should be also avoided. Next, both drawbacks are considered.

C. Navigation function

The proposed navigation function used to drive the robot to the goal, by avoiding collisions and taking into account robot dynamics constraints (maximum accelerations and velocities), is presented in this section.

Dynamics constraints imposed by the robot and obstacles are considered by using the search space defined in (4). As in previous works, a dynamic window, which contains the current set of reachable and admissible velocities, is computed. Then, an improved objective function $G^*(v,w)$ is proposed to replace (6). The domain of that objective function is defined by a dynamic window and contains the following terms:

$$\begin{array}{rl} & \lambda_1 \cdot (1 - |v - v_i| / (2 \cdot v_{max})) + \\ G^*(v,w) = & \lambda_2 \cdot (1 - |w - w_i| / (2 \cdot w_{max})) + \\ & \lambda_3 \cdot Dist(v,w), \end{array} \quad (17)$$

where, $\lambda_i > 0$, $i = 1,2,3$, and $\sum_i \lambda_i = 1$, represent weighting factors. The variables v_i and w_i are defined by the proposed (13) and (14) and they are the responsible for driving efficiently the robot to the goal. The chosen set of velocities are those that maximize the objective function (17).

The first two terms of (17) favor choosing velocities that drive the robot to the goal, the third term implements the collision avoidance strategy. Therefore, the values adopted by the weighting factors will be reflected in the robot behavior. High λ_1 and λ_2 values will result in a goal oriented behavior, while a high λ_3 value will result in a highly reactive behavior that favor the collision avoidance part. Actually, the tuning of these parameters is defined by characteristics such as: the environment's structure (number and distribution of obstacles), the required robot behavior (level of reaction), and the degree of knowledge of the environment.

IV. EXPERIMENTAL RESULTS

Experimental results obtained with both simulated and real environments are presented. In both cases a PIONEER 1 robot was used (see Fig. 3$(right)$). It presents a maximum translation velocity of 600 mm/s, a maximum rotation velocity of about $2.5\,rad/s$, and a distance between drive wheels of 325 mm. Before going into details about the obtained experimental results the required extension of I-DWA to tackle differential drive robots is presented.

A. Differential drive robots

This section presents an extension of the I-DWA algorithm in order to be able to tackle differential drive locomotion problems. Kinematics equations, together with the corresponding transformations in the velocity space, to handle this kind of robots are also introduced.

A differential drive robot uses a simple locomotion system composed of two drive wheels and a passive rear wheel. Drive wheels are independently controlled while the passive rear wheel is only used as an additional leaning point to keep the robot's balance. The rear wheel is automatically oriented according to the robot motion. The robot's displacement is achieved by means of a separated control of each drive wheel. An instantaneous center of curvature (ICC), defined by the intersection of the drive wheels' axis with the passive wheel's axis, is automatically defined according to the robot's displacement (see Fig. 3$(right)$).

Kinematics equations define the interaction between control commands and the corresponding space state. Thus, in a differential drive locomotion robot, these equations will reflect the robot's position (x, y, θ) when the velocity of each drive wheels is controlled (v_{right}, v_{left}). Therefore, from (9), translational and rotational velocities can be expressed by means of the drive wheel's velocities:

$$v = (v_{right} + v_{left})/2, \quad w = (v_{right} - v_{left})/L, \quad (18)$$

where L is the length of the drive wheels' axis (Fig. 3$(right)$). A clockwise displacement $(w > 0)$ is performed when $v_{right} > v_{left}$, otherwise a counter-clockwise displacement will be executed. From (18) the kinematics equations of a differential drive robot can be expressed as:

$$\begin{array}{rcl} \dot{x} & = & ((v_{right} + v_{left})/2) \cdot \cos(\theta), \\ \dot{y} & = & ((v_{right} + v_{left})/2) \cdot \sin(\theta), \\ \dot{\theta} & = & ((v_{right} - v_{left})/L). \end{array} \quad (19)$$

Equation (18) can be depicted by means of a matrix as:

$$\left[\begin{array}{c} v \\ w \end{array} \right] = \left[\begin{array}{cc} 1/2 & 1/2 \\ 1/L & -1/L \end{array} \right] \cdot \left[\begin{array}{c} v_{right} \\ v_{left} \end{array} \right] \quad (20)$$

hence, its inverse representation can be easily expressed as:

$$\left[\begin{array}{c} v_{right} \\ v_{left} \end{array} \right] = \left[\begin{array}{cc} 1 & L/2 \\ 1 & -L/2 \end{array} \right] \cdot \left[\begin{array}{c} v \\ w \end{array} \right] \quad (21)$$

The (20) and (21) equations allow the representation of the robot's velocities (assuming the robot is represented by a point) together with the drive wheels' velocities. After adding

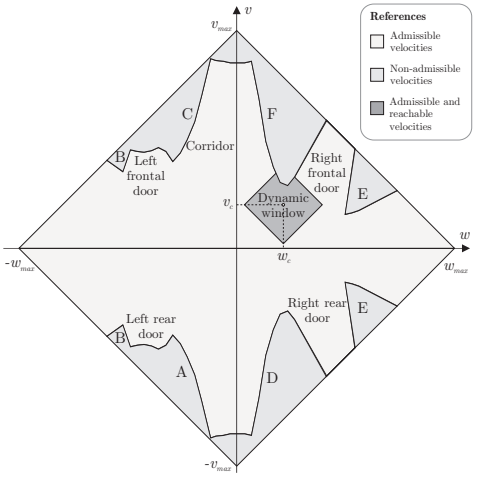

Fig. 4. Velocity map corresponding to the illustration presented in Fig. 1.

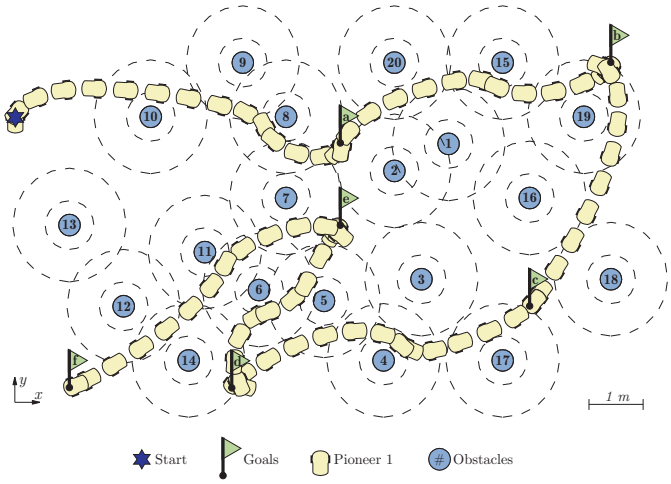

Fig. 5. Trajectories in a simulated environment performed during the test of the proposed approach.

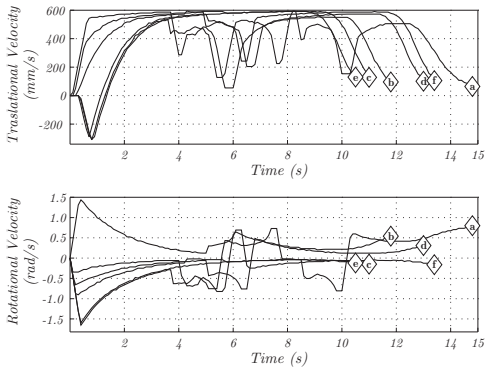

Fig. 6. Velocities corresponding to the trajectories presented in Fig. 5.

admissible and reachable velocities a representation such as the one presented in Fig. 4 is obtained. This representation is used to define the corresponding dynamic window and select the appropriate control action for the next control loop (17). Experimental results with a differential drive robot are presented in the next section.

B. Simulated environments

Simulated environments were considered to test the proposed technique. They are useful to study the robustness of the technique and the performance of the robot when different values are used for tuning the parameters. A simulated environment consists of known obstacles and a set of known goals. The model defined in (9) is used to estimate the robot trajectory, the distance to the obstacles and the distance to the next goal. The use of these predefined scenarios allows to do a fair evaluation of the performance of the proposed technique, avoiding problems related to the perception of a real environment (e.g., obstacle and goal recognition, poor perception, robot position errors).

Fig. 5 shows one of the proposed scenarios to evaluate the robot's behavior. Goals are indicated with flags while the robot' trajectories through that scenario are illustrated by means of small icons. Note that the robot is plotted by regular intervals of one second, which allows to infer the robot velocities. Each obstacle has associated two concentric circles. The inner circle corresponds to an obstacle's enlargement according to the robot's radius (robot's minimum bounding circle). The outer circle represents the influence limit; in other words, when the robot reaches the area defined by that outer circle it should start with the collision avoidance strategy. The set of trajectories performed by the robot to reach each one of the proposed goals shows the efficiency of the proposed approach. Note that I-DWA correctly drives the robot along narrow passages, such as the $<e>$ trajectory that goes through the 5 and 6 obstacles. High speeds were reached even in presence of obstacles.

Table I shows the robot average speed in every trajectory. In all the cases the average translation velocity is higher than half the maximum speed. Moreover, that maximum speed is also reached in every trajectory (see Fig. 6).

Fig. 6 shows a plot of the velocities reached by the robot in its way to the goal. This figure demonstrates that the movements are smooth with few oscillations on the robot's orientation. It is also observable that the robot first tries to orientate towards the goal, in some cases by moving backward, and then, it advances trying to reach the highest speed in absence of obstacles. This behavior is regulated with the factors k_ρ, k_α and k_v from ideal control law, (13) and (14); as well as the weighting factors of the objective function $(\lambda_1, \lambda_2, \lambda_3)$, (17). In the current implementation they were defined as: $k_\rho = r_r^{-1} \simeq 3\,[m^{-1}]$; $k_\alpha = 0.59\,[s^{-1}]$; $k_v = 1$; $\lambda_1 = 3/13$; $\lambda_2 = 3/13$; and $\lambda_3 = 7/13$.

These values were experimentally selected by using testbed environments. As mentioned above, since a simulated environment is used, several experiments can be performed keeping away from risky situations. Simulated environments make it easier and faster the parameter's tuning. The used strategy is as follow. Firstly, a free-obstacles environment is considered until the sought robot behavior is obtained —i.e., efficient and successful movements to the goal—. Finally, a fine tuning is preformed by using a highly populated

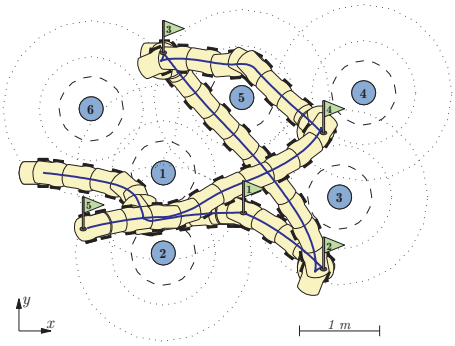

Fig. 7. Real navigation environment used for testing I-DWA with the PIONEER 1.

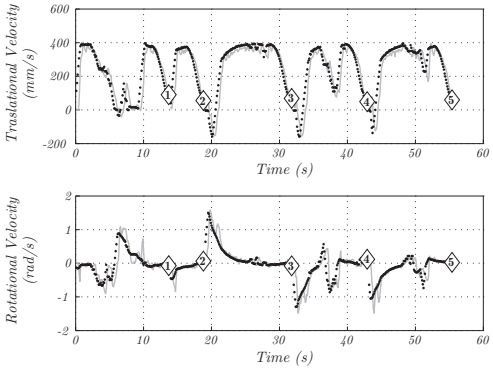

Fig. 8. Velocities corresponding to the trajectories presented in Fig. 7.

environment, where a safe collision avoidance is reached.

C. Real environments

Finally, the proposed technique was validated in real environments (about 4x5 m) with a PIONEER 1 robot. The I-DWA parameters were tuned with the same values than the selected in the simulated environments. I-DWA was implemented off-board, in the framework Aria/Saphira, using a master/slave architecture. Obstacles in the scene were detected by the PIONEER 1 robot ultrasonic sensors. Robot odometer was used to estimate the current position and the distance to the next goal.

Fig. 7 presents a scene with six unknown obstacles and a sequence of five user-defined static goals. Obstacles and robot are represented as in the simulated environment (Fig. 5). As it was expected, in all the cases the robot reaches the goals with smooth and safe trajectories. Fig. 8 depicts the velocities corresponding to every trajectory, solid line shows the sent velocities from off-board computer and dot line shows the velocities reached by the robot. Equally than in the simulated scenario, efficient movements, with few oscillations on the robot's orientation, are performed. Note that the robot stops when reaches a goal; in some cases it goes backward in order to quickly orientate to the next goal. The maximum speed has been reduced up to 400 mm/s to avoid localization problems related to the odometry.

TABLE I

SPEED AND TIME FOR EACH TRAJECTORY.

Trajectory	Average Speed	Time
$< a >$	442.95 $[mm/s]$	14.8 $[s]$
$< b >$	466.30 $[mm/s]$	11.8 $[s]$
$< c >$	450.85 $[mm/s]$	11.0 $[s]$
$< d >$	463.00 $[mm/s]$	13.0 $[s]$
$< e >$	370.60 $[mm/s]$	10.5 $[s]$
$< f >$	453.75 $[mm/s]$	13.4 $[s]$

V. CONCLUSIONS

This paper presents a novel and compact approach (I-DWA) for autonomous robot navigation. It improves the original technique by incorporating Lyapunov stability criteria inside the kernel of DWA. Therefore, an arrival to the goal with a global and asymptotic convergence is guaranteed. As a result, a more simple and self-contained approach is obtained. Different robot behavior can be reached by tuning a set of control law parameters.

Simulated and real experimental results validate the proposed technique when different environments' configuration are used. It should be noticed that with the proposed approach the convergence drawbacks of DWA are solved. At the same time, the capability of I-DWA to regulate different behaviors can be easily appreciated. Further work will address the current odometry problems; hence larger real environments could be considered.

REFERENCES

[1] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The Int. Journal of Robotics Research*, 5(1):90–98, 1986.

[2] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1179–1187, Sep/Oct 1989.

[3] J. Borenstein and Y. Koren. The vector field histogram - fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, June 1991.

[4] R. Simmons. The curvature-velocity method for local obstacle avoidance. In *IEEE International Conference on Robotics and Automation (ICRA'96)*, pages 3375–3382, Los Alamitos, CA, USA, April 1996.

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

[6] S. Shimoda, Y. Kuroda, and K. Iagnemma. Potential field navigation of high speed unmanned ground vehicles on uneven terrain. In *IEEE International Conference on Robotics and Automation (ICRA'05)*, pages 2839–2844, Barcelona, Spain, April 2005.

[7] O. Brock and O. Khatib. High-speed navigation using the global dynamic window approach. In *IEEE International Conference on Robotics and Automation (ICRA'99)*, pages 341–346, 1999.

[8] K. O. Arras, J. Persson, N. Tomatis, and R. Siegwart. Real-time obstacle avoidance for polygonal robots with a reduced dynamic window. In *IEEE International Conference on Robotics and Automation (ICRA'02)*, pages 3050–3055, May 2002.

[9] R. Philippsen and R. Siegwart. Smooth and efficient obstacle avoidance for a tour guide robot. In *IEEE International Conference on Robotics and Automation (ICRA'03)*, pages 446–451, 2003.

[10] P. Ögren and N. E. Leonard. A convergent dynamic window approach to obstacle avoidance. *IEEE Transactions on Robotics and Automation*, 21(2):188–195, April 2005.

[11] O. A. A. Orqueda, H. Berti, and O. E. Agamennoni. A strategy for safe goal-directed autonomous navigation. In *International Symposium on Intelligent Components and Instruments for Control Applications (SICICA 2000)*, pages 135–140, Buenos Aires, Argentina, 2000.

[12] H. Secchi, R. Carelli, and V. Mut. Design of stable algorithms for mobile robot control with obstacle avoidance. In *14th IFAC World Congress on Automatic Control (IFAC'99)*, pages 185–190, July 1999.