Motion Control of the CyberCarpet Platform

Alessandro De Luca*, Raffaella Mattone*
deluca@dis.uniroma1.it, raffaella.mattone@gmail.com
Paolo Robuffo Giordano‡
prg@tuebingen.mpg.de
Heinz Ulbrich#, Martin Schwaiger#
ulbrich@amm.mw.tum.de, dr.schwaiger@gmx.de
Michael Van den Bergh†, Esther Koller-Meier†, Luc Van Gool†
{vandenbergh,vangool}@vision.ee.ethz.ch, esther.koller@satw.ch

***Abstract*— The *CyberCarpet* is an actuated platform that allows unconstrained locomotion of a walking user for Virtual Reality exploration. The platform consists of a linear treadmill covered by a ball-array carpet and mounted on a turntable, and is equipped with two actuating devices for linear and angular motion. The main control objective is to keep the walker close to the platform center in the most natural way, counteracting his/her voluntary motion while satisfying perceptual constraints. The motion control problem for this platform is not trivial since the system kinematics is subject to a nonholonomic constraint. In the first part of the paper we describe the kinematic control design devised within the *CyberWalk* project, where the linear and angular platform velocities are used as input commands and feedback is based only on walker's position measurements obtained by an external visual tracking system. In particular, we present a globally stabilizing control scheme that combines a feedback and a feedforward action, based on a disturbance observer of the walker's intentional velocity. We also discuss possible extensions to acceleration-level control and the related assessment of dynamic issues affecting a walker during his/her motion. The second part of the paper is devoted to the actual implementation of the overall system. As a proof of concept of a final full-scale platform, the mechanical design and realization of a small-scale prototype of the *CyberCarpet* is presented, as well as the visual localization method used to obtain the human walker's position on the platform by an overhead camera. To validate the proposed motion control design, experimental results are reported and discussed for a series of motion tasks performed using a small tracked vehicle representative of a moving user.**

***Index Terms*— Motion control, locomotion platform, nonholonomic systems, input-output feedback linearization, disturbance observer, visual tracking, virtual reality.**

I. Introduction

Exploration of virtual reality (VR) worlds by allowing omni-directional unconstrained locomotion possibilities for a walking user is an active area of research. The final goal is having the user fully immersed in a VR scene viewed by, e.g., a Head Mounted Display (HMD), free to walk in any direction with natural speed, while remaining within the limited physical area of a platform and without the need of wearing any constraining equipment (e.g., for tracking the walker position or for characterizing the gait). To support locomotion in this way, the platform should counteract the intentional motion of the walker in order to keep him/her in place. In doing so, the associated perceptual effects on the walker should be taken into account, in the form of input command constraints, so as to avoid disruptive effects on the user's immersiveness. These have been the main objectives of the European research project *CyberWalk* [1].

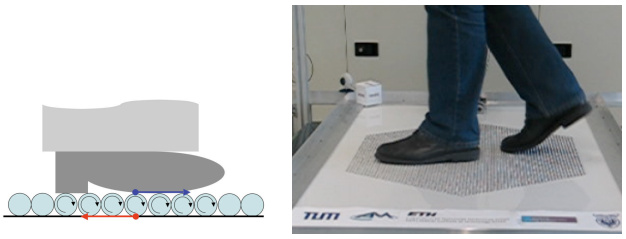
Different locomotion interfaces exist that allow walking in virtual environments (see, e.g., the surveys in [2], [3], [4]). In many of them, locomotion is restricted to a 1D motion on a linear treadmill, like in the Treadport platform [5] with possible slope inclusion [6]. The user is constrained by a harness to apply stabilizing forces and other virtual effects [7]. To allow for small/slow direction changes, the treadmill can be mounted on a turning table [8]. A different approach is taken in the CirculaFloor [9], where active moving tiles follow the feet motion. Again, the walker should avoid sharp turns and high speed. For unconstrained 2D walking, the Omnidirectional Treadmill has been proposed in [10] using two perpendicular belts and a large number of rollers, while a torus-shaped belt arrangement is implemented in the Torus Treadmill [11]. Both systems allow limited speed, mostly due to poor control design. Furthermore, the mechanical implementation is critical due to the large mass of the moving parts (with associated noise). This kind of problems is not present in passive devices like the Cybersphere [12] where, however, the naturalness of walking is limited by the inner curvature of the spherical floor. An alternative principle is used in [13], where a conveyor belt and a turntable transmit motion to a walker through a ball-array board, realizing thus a 2D planar treadmill. In [14], the ball-array lays on a concave surface without actuation, but instrumented with sensors to detect feet contact.

Within the *CyberWalk* project, two different motion concepts have been considered for unconstrained 2D walking on a plane: the omnidirectional belt-array *CyberWalk* platform [15], [16], [17] (similar to [10], [11]), and the ball-array *CyberCarpet* [18], [19], [20] (similar to [13]). In both cases, as project requirements, we wished to eliminate the use of any physical constraints on the feet, body, or legs of the user, as well as to avoid the need of a priori or identified models of the human

* Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Roma, Italy; ‡ MPI for Biological Cybernetics, Tübingen, Germany; # Institute for Applied Mechanics, Technical University of Munich, München, Germany; † Swiss Federal Institute of Technology, Zürich, Switzerland.

gait/walk. The two platform concepts have been analyzed and refined in terms of user mobility, mechanical feasibility, and perceptual effects.

In this paper, we focus on the *CyberCarpet* and its motion control problem. This locomotion platform uses a conveyor belt and a turntable to transmit translational and angular motion to the walker through a ball-array board. Rotating balls are fitted into an array and are in contact with the belt so that a user on the board moves in the opposite direction of the underlying point on the belt, see Fig. 1(a). The walker will be allowed to move in a natural way and indefinitely in any planar direction. In fact, the platform controller will counteract her/his motion by pulling the walker toward the center of the platform, while taking into account physiologically acceptable velocity/acceleration bounds. The body pose on the carpet is acquired through a markerless visual tracking system using an overhead camera.



(a) (b)

Fig. 1. The *CyberCarpet* platform: (a) the motion transmission principle; (b) the final small-scale realization

As a proof of concept of the *CyberCarpet* principle, a small-scale prototype of about 0.8 m of diameter has been designed and built, see Fig. 1(b). The overall system architecture is shown in Fig. 2. While the limited platform dimension is indeed not appropriate for the actual VR exploration by a human user, the whole system has been conceived keeping in mind the requirements and challenges of a full-size realization. In fact, the current mechanical structure can already support the weight of a human user (about 100 kg). Further, our proposed control design allows a straightforward scaling of control gains to the platform size and locomotion speed of the walker, and extensions to acceleration level control make it possible to fully take into account dynamic issues affecting the walker during her/his motion. Finally, the visual tracking algorithm has been tested on human walkers and proved to be robust w.r.t. her/his posture changes (see Fig. 16). To show the effectiveness of the platform control design, we report experimental results in which a top-view human picture has been mounted on a mobile robot and used as a mock-up to emulate the motion behavior of a real user (see Fig. 17).

From the control point of view, one challenging issue is due to the kinematic model of the system, which is highly nonlinear as opposed to the case of other 1D and 2D omnidirectional platforms, and it includes a nonholonomic constraint on the system instantaneous velocities. While the platform cannot provide full local mobility to the user, we will design a control law that achieves smooth and global regulation of the walker position from any initial configuration. Previous

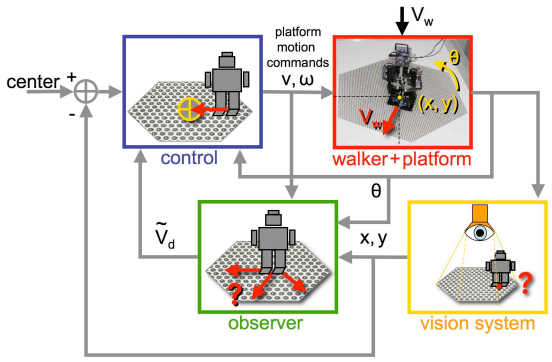


Fig. 2. Control system architecture of the *CyberCarpet*

works on locomotion interfaces have paid little attention to control issues and algorithms, relying mostly on very simple PID laws or heuristic schemes. With the notable exception of [21] for a 1D treadmill, no stability analysis has been considered previously. Moreover, performance in 2D has been predicted only by restricting motion to low and piece-wise constant walker velocity with few directional changes (see, e.g., the paper by [13] on a similar ball-array locomotion device). In contrast, a complete motion control design is presented here for the 2D *CyberCarpet* platform, together with a detailed analysis of the closed-loop performance. A number of experimental results are included to validate our approach.

The paper is organized as follows. In the first part, we present the control approach devised within the *CyberWalk* project[1]. The kinematic model of the *CyberCarpet* platform is given in Sect. II, and its duality with the control problem for nonholonomic wheeled mobile robots is pointed out. In Sect. III, we propose a velocity control scheme based on input-output decoupling and linearization (Sect. III-A), which shows an acceptable overall performance but is still affected by some singularities that prevent global stabilization results. The velocity control scheme is then suitably modified in order to avoid such singularities (Sect. III-B), and an additional feedforward action is used, based on a disturbance observer of the (unavailable) intentional velocity of the walker (Sect. III-C). Performance of this controller is much more satisfactory and is first validated by numerical simulations (Sect. III-D). Finally, a convenient way to extend the design by considering accelerations as control inputs is discussed in Sect. IV. This allows to take explicitly into account acceleration bounds imposed by the actuation system and the perception of the human walker. A simplified analysis of the dynamic effects that a user would experience is also included (Sect. IV-A).

The second part of the paper is devoted to the physical implementation and test of a small-scale but complete prototype of the *CyberCarpet*. Section V illustrates the mechanical design and the hardware realization. Section VI describes the visual tracking algorithm for the localization of the absolute walker position (and orientation) on the platform, based on particle filters. Experimental results are

[1]Preliminary versions of this part have been presented in the conference papers [18], [19] and [20].

reported in Sect. VII, showing the performance of the implemented velocity-level control laws within the integrated system. Videos of all experiments are included in the accompanying material to this paper, as well as available at http://www.dis.uniroma1.it/labrob/research/CW.html together with further illustrating movies. Conclusions and future work are discussed in the final section, including also a comparison with the companion omnidirectional belt-array *CyberWalk* platform.

II. Kinematic Model

Thanks to the ball-array surface of the *CyberCarpet*, any actuated motion of the underlying belt will result in a reverse motion imposed to the walker on the platform, i.e., a forward motion command will move the user backwards and, due to the multiple contact between walker's feet and the ball-array carpet, a clockwise rotation will turn the user counterclockwise —see Fig. 1(a).

With this in mind, it is possible to derive a kinematic model of the *CyberCarpet*. With reference to Fig. 3, let (X_0, Y_0) be the absolute inertial frame (attached to the fixed overlooking camera) and (X_t, Y_t) is the frame attached to the (rotating) treadmill, with the X_t–axis in the direction of the belt, rotated by an angle θ w.r.t. X_0. Both frames have the origin at the center of the *CyberCarpet*. The walker absolute position and orientation are, respectively, (x, y) and θ_w, with the distance from the center $R = \sqrt{x^2 + y^2}$ and the angle $\alpha = \text{ATAN2}(y, x) - \theta$ locating the walker in polar coordinates w.r.t. the rotating frame (X_t, Y_t).

When the walker is *standing still*, we have

$$\begin{aligned} \dot{x} &= -v\cos\theta + y\omega \\ \dot{y} &= -v\sin\theta - x\omega \\ \dot{\theta} &= \omega \\ \dot{\theta}_w &= -\omega. \end{aligned} \tag{1}$$

In eq. (1), v and ω are the commanded linear and angular velocity of the *CyberCarpet*. These are applied on the bottom of the ball-array, which explains the *minus* signs appearing in the first, second and fourth equation of the model.

For $R \neq 0$, the two Cartesian coordinates (x, y) may also be replaced by the polar coordinates (R, α) (see also [22]), obtaining

$$\begin{aligned} \dot{R} &= -v\cos\alpha \\ \dot{\alpha} &= v\,\frac{\sin\alpha}{R} - 2\omega \end{aligned} \tag{2}$$

A simple analysis of the kinematic equations (1) shows that a holonomic constraint exists,

$$\theta + \theta_w = \theta(0) + \theta_w(0) = \text{constant}, \tag{3}$$

i.e., the change of orientation of the platform under control will be equal to the opposite of the change of orientation of the walker (when standing still). Therefore, only one of these two variables can be independently controlled. However, this is not a limitation for the considered motion control task. When a platform user walks through a virtual world (e.g., wearing a Head Mounted Display), the relevant orientation for a correct VR visualization is only the one intentionally assumed by the walker head, which is fully unrelated from the platform orientation θ and partially independent from the orientation θ_w of the user body.

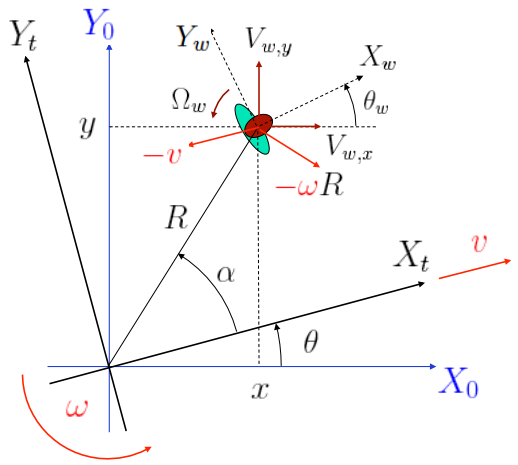


Fig. 3. Frames and variables definition for the *CyberCarpet*

On the other hand, in the three-dimensional configuration space of interest, parametrized by (x, y, θ), the system is subject to the differential constraint

$$\begin{bmatrix} \sin\theta & -\cos\theta & -(x\cos\theta + y\sin\theta) \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = 0. \tag{4}$$

This constraint implies that the platform cannot move the position (x, y) of the walker along the direction of the axis Y_t. As a consequence, the platform will not be able to cancel instantaneously a walker linear velocity in that direction. Nonetheless, equation (4) is nonholonomic (i.e., it cannot be integrated to a positional constraint) and thus the nonlinear system is fully controllable to any desired configuration (x, y, θ) [23]. Therefore, re-centering of a standing user to the platform origin $(0, 0, 0)$ can be in principle achieved by suitable platform *maneuvers* obtained by the control commands v and ω. It is interesting to note that, in view of (4), the motion control problem for the *CyberCarpet* is similar to that of nonholonomic wheeled mobile robots. The analogy of the two problems can be intuitively recognized also by flipping things upside down: the standing user plays the role of the fixed ground, while the nonholonomic platform will act as the controlled wheeled mobile robot.

However, the above duality is lost when the walker starts to move. In fact, when the walker is in motion, the model (1) becomes

$$\begin{aligned} \dot{x} &= -v\cos\theta + y\omega + V_{w,x} \\ \dot{y} &= -v\sin\theta - x\omega + V_{w,y} \\ \dot{\theta} &= \omega \\ \dot{\theta}_w &= -\omega + \Omega_w, \end{aligned} \tag{5}$$

where $V_w = (V_{w,x}, V_{w,y})$ and Ω_w are, respectively, the absolute linear and angular walker *intentional* velocities (see Fig. 3). These are clearly unknown in advance and not directly measurable. In the following, the walker intentional velocities will be considered as *disturbances* in the control design. In

particular, we will not be interested in Ω_w since this exogenous signal will not affect the walker position on the platform. Instead, the walker displacement due to the linear velocity vector V_w needs to be rejected by the control system.

III. Control design

A number of feedback control laws developed for nonholonomic wheeled mobile robots can be adapted to address the regulation problem for the *CyberCarpet*, so as to bring the position (and, if needed, the orientation too) of a standing user to zero. These techniques include Lyapunov design in polar coordinates [22], time-varying nonlinear control [24], control based on the chained-form transformation or on system flatness [25], or recursive control with backstepping [26]. Although successful, these techniques lead to somewhat oscillatory and/or slow transients that are not compliant with user perceptual constraints. Moreover, the extension of such control laws to handle the persistent disturbance due to the motion of the walker has not been considered yet.

The nature of our motivating application (locomotion of a user in a VR environment using a HMD) specifies the control task as regulation of only the position (x, y) of the walker. In this respect, the walker's position can be asymptotically stabilized to the platform origin using a simpler control design based on input-output feedback linearization.

A. Input-output decoupling and linearizing control

Consider first the case of no disturbances, i.e., $V_w = 0$ and $\Omega_w = 0$ (walker standing still in the virtual environment) and define the controlled output as $(x\ y)$. From eq. (1), it is

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -\cos\theta & y \\ -\sin\theta & -x \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = A(x,y,\theta) \begin{bmatrix} v \\ \omega \end{bmatrix}. \quad (6)$$

When $\det A = x\cos\theta + y\sin\theta \neq 0$, we can set

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = A^{-1}(x,y,\theta) \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \quad (7)$$

where v_1 and v_2 are auxiliary velocity inputs to be defined. The resulting input-output behavior is given by simple integrators

$$\dot{x} = v_1, \qquad \dot{y} = v_2,$$

i.e., it has been decoupled and linearized by the feedback law (7). The control design can be completed by the proportional laws

$$v_1 = -k_1 x, \qquad v_2 = -k_2 y, \quad (8)$$

with positive gains k_i, $i = 1, 2$, thus exponentially stabilizing the walker's position to the origin.

An interesting property of the designed controller is the following. With the walker standing at an initial position (x_0, y_0), the controlled time evolution of his/her position will be $x(t) = e^{-k_1 t}x_0$ and $y(t) = e^{-k_2 t}y_0$. Choosing $k_1 = k_2$ leads to

$$\frac{y(t)}{x(t)} = \frac{\dot{y}(t)}{\dot{x}(t)} = \frac{y_0}{x_0},$$

so that the user will be pulled toward the origin along the connecting straight line.

For the purpose of analysis, an equivalent expression for the control law v and ω can be found by selecting (from now on) equal gains $k_1 = k_2 = k > 0$ and replacing eq. (8) into (7). This yields

$$v = \frac{k(x^2+y^2)}{x\cos\theta + y\sin\theta} = \frac{kR^2}{R\cos\alpha} = \frac{kR}{\cos\alpha} \quad (9)$$

and

$$\omega = \frac{k(y\cos\theta - x\sin\theta)}{x\cos\theta + y\sin\theta} = \frac{kR\sin\alpha}{R\cos\alpha} = k\tan\alpha. \quad (10)$$

The control law (9–10) has the singularity $R\cos\alpha = x\cos\theta + y\sin\theta = 0$ (where the determinant of the decoupling matrix is zero), i.e., when the walker is on the Y_t axis (in particular, at the origin). Note that the singularity at the origin is hidden, in polar coordinates, by the fact that the angle α is not defined there. In the following, we will introduce modifications that overcome these control singularities, while preserving the convenient characteristics of the decoupling law (9–10). We still consider for the time being the case of no disturbances, $V_w = 0$ and $\Omega_w = 0$ (walker standing still in the virtual environment), dealing later with the case of walker in motion (Sec. III-C).

B. Handling the singularities at $\cos\alpha = 0,\ R = 0$

When $R \neq 0$, the control singularity at $\alpha = \pm\frac{\pi}{2}$ can be eliminated by taking

$$v = kR\,\mathrm{sgn}(\cos\alpha) \quad (11)$$

and

$$\omega = k\sin\alpha\,\mathrm{sgn}(\cos\alpha), \quad (12)$$

with $\mathrm{sgn}(arg) = 1$ for $arg \geq 0$ and $\mathrm{sgn}(arg) = -1$ otherwise. The control law (11–12) is formally obtained by multiplying eqs. (9–10) by $|\cos\alpha|$. The resulting input-output dynamics is now

$$\dot{x} = -k\,|\cos\alpha|\,x, \quad \dot{y} = -k\,|\cos\alpha|\,y, \quad (13)$$

which is no longer linear nor decoupled, since the angle α depends on both x and y. However, from (13) it follows

$$\frac{y(t)}{x(t)} = \frac{\dot{y}(t)}{\dot{x}(t)} = \frac{y_0}{x_0} \quad (14)$$

as before. Therefore, a standing user will still be driven along the straight line connecting its initial position to the origin ($\theta + \alpha = \theta_0 + \alpha_0 =: \beta_0$ is constant).

In order to show that the control law (11–12) is also asymptotically stabilizing the walker position (x, y) to the origin, consider the positive definite Lyapunov function

$$\begin{aligned} V(x,y,\theta) &= \tfrac{1}{2}(x^2 + y^2 + \sin^2(\beta_0 - \theta)) \\ &= \tfrac{1}{2}(R^2 + \sin^2\alpha) \geq 0, \end{aligned} \quad (15)$$

with $V = 0$ if and only if (x, y, θ) belongs to the set $\mathcal{S} = \{(0, 0, \theta) :\ \sin(\beta_0 - \theta) = 0\}$. Using (2), the time derivative of V along the trajectories of the closed-loop system (1), (11–12) is given by

$$\begin{aligned} \dot{V} &= R\dot{R} + \sin\alpha\,\cos\alpha\,\dot{\alpha} = -k\,|\cos\alpha|\,(R^2 + \sin^2\alpha) \\ &= -2k\,|\cos\alpha|\,V \leq 0, \end{aligned} \quad (16)$$

so that $\dot{V} = 0$ for $(x, y, \theta) \in \mathcal{S}$, as well as for $\cos\alpha = 0$. However, the latter does not correspond to closed-loop system equilibria since then $\omega = \pm k$ and thus θ cannot be constant. Therefore, $\mathcal{S}$ is the (largest) invariant set where $\dot{V} = 0$, and (x, y) will converge to the origin by virtue of LaSalle theorem.

At $R = 0$, the control law (11–12) is clearly discontinuous at the origin $x = y = 0$, due to the discontinuity of the angle α which is not defined there. This causes a chattering of the control input ω when the walker is in a small region around the platform center[2]. In order to avoid this problem, we proposed in [19] the introduction of a dead zone around the origin. However, when the walker is in motion, the chattering of the input commands may appear again at the border of the dead zone.

A more effective solution is obtained by replacing the feedback control law (11–12) with

$$\begin{aligned} v &= kR^2 \,\mathrm{sgn}(\cos\alpha) \\ &= k(x^2 + y^2)\,\mathrm{sgn}(x\cos\theta + y\sin\theta), \end{aligned} \tag{17}$$

and

$$\begin{aligned} \omega &= kR\sin\alpha\,\mathrm{sgn}(\cos\alpha) \\ &= k(y\cos\theta - x\sin\theta)\,\mathrm{sgn}(x\cos\theta + y\sin\theta), \end{aligned} \tag{18}$$

which are formally obtained multiplying eqs. (11–12) by the radial distance R. The control law (17–18) is now well defined and continuous at any system configuration. In fact, the angular velocity command ω converges to zero as (x, y) approaches the origin, so that no chattering occurs in this case. Under the feedback law (17–18) and in the absence of walker motion, the behavior of the controlled outputs becomes

$$\dot{x} = -kR\,|\cos\alpha|\,x, \quad \dot{y} = -kR\,|\cos\alpha|\,y, \tag{19}$$

which can be proved to be asymptotically stable at the origin by the same Lyapunov arguments used for law (11–12). Furthermore, eq. (14) still holds in this case, i.e., the user is pulled toward the origin along the connecting straight line. However, the convergence rate of the x and y variables drops quadratically to zero as the walker approaches the origin.

Remark 1: A Lyapunov-based proof of stability for the control law (17–18) holds also in the case of a varying gain k, provided that $k > 0$. This property can be used to decrease excessively large commands v and/or ω in order to comply with human perceptual constraints. In particular, assume that the control inputs should remain always bounded as

$$|v| \le v_{max}, \qquad |\omega| \le \omega_{max}, \tag{20}$$

and choose a convenient $\bar{k} > 0$ to be used in eqs. (17–18) in the unsaturated case. Then, the following *scaling* law for k

$$k = \frac{\bar{k}}{\max\left\{1, \frac{|v|}{v_{max}}, \frac{|\omega|}{\omega_{max}}\right\}} > 0 \tag{21}$$

will comply with the bounds in (20). ◇

[2]Although a term $\mathrm{sgn}(\cos\alpha)$ appears also in the expression of v, the chattering phenomenon for this control input is overcome by the presence of the factor R vanishing at the origin.

C. Dealing with walker's velocity

When the walker is in motion, V_w and Ω_w are in general both different from zero and the system kinematics is described by eq. (5). A persistent walker locomotion will typically prevent the convergence of her/his position to the platform center when using the control law (17–18). In particular, when the user walks indefinitely with constant velocity $\bar{V}$ along a straight line in the virtual environment, a steady-state position will be reached under the control law (9–10) at a distance $\bar{R} = \bar{V}/k$ from the origin. Using standard results from linear control theory, we proposed in [18] to add an integral control action so as to completely eliminate the steady-state error in the case of a constant walker velocity. However, use of an integral control action suffers from the typical position overshooting and leads to a poor dynamic performance for more general motion profiles of the walker.

Therefore, we follow a different approach to deal with the walker's intentional motion, based on an estimate $\tilde{V}_w$ of the walker *linear* velocity vector V_w. This allows to add suitable feedforward terms in the control law as

$$\begin{aligned} v &= v_{fb} + v_{ff} = v_{fb} + \begin{bmatrix} \cos\theta & \sin\theta \end{bmatrix} \tilde{V}_w \\ \omega &= \omega_{fb} + \omega_{ff} \\ &= \omega_{fb} + \mathrm{sat}\left(\frac{1}{R}\begin{bmatrix} -\sin\theta & \cos\theta \end{bmatrix} \tilde{V}_w\right), \end{aligned} \tag{22}$$

where the relabeled v_{fb} and ω_{fb} are the feedback contributions given by eqs. (17–18) and $\mathrm{sat}(\cdot)$ is the standard saturation function, with lower/upper saturation limits $\pm\omega_{ff,max}$. It is readily verified that, for $\tilde{V}_w = V_w$, the feedforward term v_{ff} in eq. (22) compensates for the component of the walker velocity along the treadmill direction of the *CyberCarpet*, while ω_{ff} (in the absence of saturation) cancels the component of V_w in the orthogonal direction[3].

Remark 2: When the control scaling strategy (21) is considered (see Remark 1), it should be applied only to the feedback part in (22), i.e., with $v = v_{fb}$ and $\omega = \omega_{fb}$. In fact, the feedforward terms v_{ff} and ω_{ff} in (22) result in a partial or perfect cancelation of the walker intentional velocity. In the absence of a position error, this control action is not 'felt' by the user since he/she would be walking *in place* in the absolute frame. As a result, these feedforward terms should not be subject to perceptual constraints during walking, which instead limit the feedback part of the control law (22). ◇

In order to get an estimate $\tilde{V}_w = (\tilde{V}_{w,x}, \tilde{V}_{w,y})$, consider the two scalar dynamical systems

$$\begin{aligned} \dot{\xi}_x &= -v\cos\theta + y\omega + k_w(x - \xi_x) \\ \tilde{V}_{w,x} &= k_w(x - \xi_x) \end{aligned} \tag{23}$$

and

$$\begin{aligned} \dot{\xi}_y &= -v\sin\theta - x\omega + k_w(y - \xi_y) \\ \tilde{V}_{w,y} &= k_w(y - \xi_y), \end{aligned} \tag{24}$$

[3]The saturation in ω_{ff} is necessary to exclude a possible divergence when R approaches zero. However, for V_w smooth enough, the platform tends to align with V_w so that ω_{ff} remains small enough over time.

where $k_w > 0$ and v and ω are given by (22). Equations (23) and (24) have the structure of *disturbance observers*, respectively with state ξ_x and ξ_y. From these and (5), it follows

$$\begin{aligned} \dot{\tilde{V}}_{w,x} &= k_w \left(V_{w,x} - \tilde{V}_{w,x} \right) \\ \dot{\tilde{V}}_{w,y} &= k_w \left(V_{w,y} - \tilde{V}_{w,y} \right), \end{aligned}$$

i.e., the estimates $\tilde{V}_{w,x}$, $\tilde{V}_{w,y}$ are low-pass filtered versions of the Cartesian components of V_w. In particular, for k_w large enough, they accurately reproduce the two components $V_{w,x}$ and $V_{w,y}$ of the intentional walker velocity, expressed in absolute coordinates.

Note that, even after the feedforward compensation (22), the system is still affected by a residual disturbance

$$V_w - \tilde{V}_w = \frac{s}{s + k_w} V_w.$$

Therefore, while walker's constant velocities are fully compensated at steady state for any positive k_w, for walker's ramp-wise velocities (constant accelerations) the associated steady-state error can only be made arbitrarily small by increasing k_w —an *astatic* velocity disturbance behavior is recovered by the proposed feedback/feedforward controller.

D. Simulation results

We present here two selected results obtained with the singularity-free velocity control law discussed in Sects. III-B, III-C. In both case studies, the walker starts at rest from the initial absolute position $(0, 1)$ m —one that would immediately lead to singularity for the control law (9–10) of Sect. III-A.

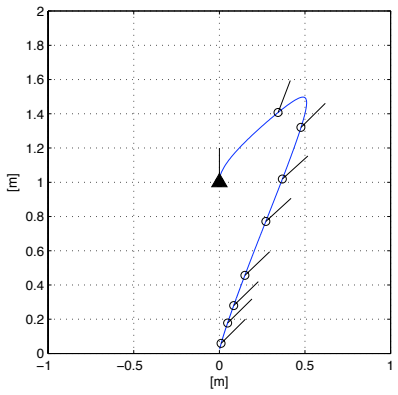


Fig. 4. Virtual straight line: Walker absolute locomotion under the platform controller (22) with the feedback law (17–18) —the initial walker pose is depicted by an oriented triangle

Figures 4–6 refer to the walker moving indefinitely along a straight line directed along the Y_0-axis in the *virtual space*, and with a constant speed of 1 m/s (at time $t = 0$, it is $X_t(0) = X_0$). The platform is controlled by the combined feedback/feedforward scheme (22), wherein the feedback law (17–18) and the disturbance observers (23–24) are used. The associated control parameters are: $\bar{k} = 1$ (i.e., k without the scaling), $k_w = 10$, and $\omega_{ff,max} = 2$ rad/s for the saturation term in (22). Moreover, in order to show the

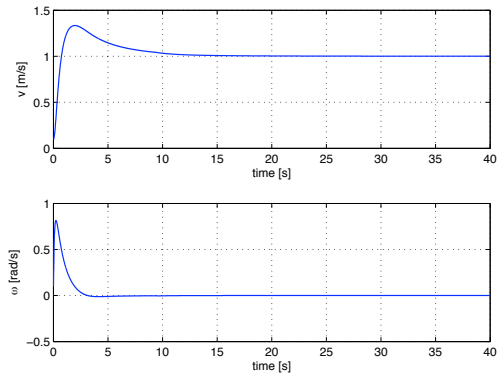


Fig. 5. Linear and angular velocity commands for the trajectory of Fig. 4

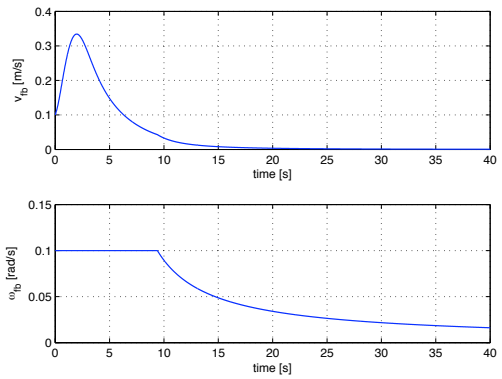


Fig. 6. Feedback part of the linear and angular velocity commands in Fig. 5

effects of a limitation of the velocity commands imposed by perceptual constraints, we have set a maximum value of 0.5 m/s and 0.1 rad/s, respectively, for v_{fb} and ω_{fb} in the feedback part (17–18) of the control law (22). In Fig. 4, the actual motion of the walker in the *absolute space* is shown, with the absolute orientation θ_w of the walker being displayed by a segment. At the start, we have $\theta(0) = 0$ and $\theta_w(0) = \pi/2$. The overall behavior of the linear and angular velocity commands is shown in Fig. 5, while Fig. 6 shows only the feedback part of these velocity inputs. When the bounds set for the feedback part are exceeded, in particular at the beginning of the control interval until $t \approx 10$ s, the gain k is lowered according to eq. (21).

As an example of a more complex motion, we report the results for the *virtual* square path with 3 m sides shown in Fig. 7. The walker starts at rest and moves along each edge with a trapezoidal velocity profile, having symmetric acceleration/deceleration phases with 2.4 m/s^2 for 0.5 s each and a cruise velocity of 1.2 m/s kept for 2 s. At each reached corner, the walker stops and turns counterclockwise with an angular speed of $\pi/2$ rad/s. Thus, the total trajectory of the walker lasts 16 s. Without motion control of the platform, the walker would exit from the platform circular boundary set at a radius of 2.5 m.

The control parameters are chosen as before. In order to evaluate the performance attainable by the proposed control

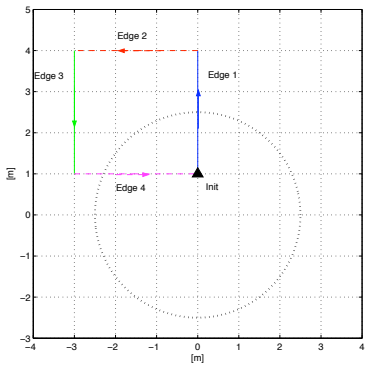


Fig. 7. Virtual square path: Walker moves counterclockwise starting from Init point (a dotted circle represents the platform boundary chosen in simulation)

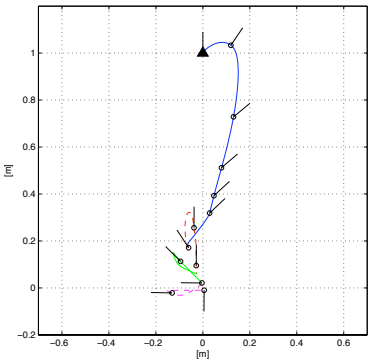


Fig. 8. Virtual square path: Walker absolute locomotion under the platform controller (22) and using the feedback law (17–18)

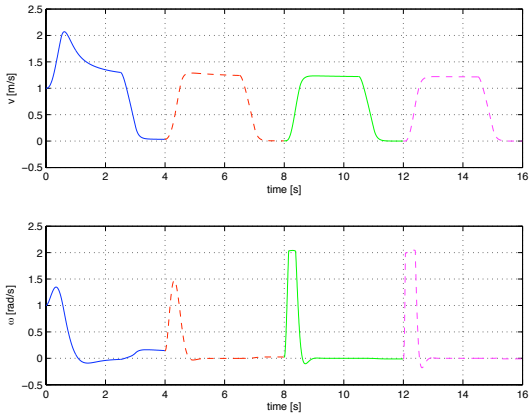


Fig. 9. Linear and angular velocity commands for Fig. 8

scheme, no perceptual constraints on the feedback commands are considered in this second case. The results are shown in Figs. 8–9. Thanks to the combined feedback and feedforward actions, the walker is rapidly brought close to the platform center and then kept there. The linear control input is again smooth and, after an initial transient, does never exceed the walker's voluntary speed. On the other hand, saturation on the angular feedforward term (with $\omega_{ff,max} = 2$ rad/s as before) comes into action when the walker is close to the origin and takes a sharp turn (starting with the corner after Edge 2 of the path). Note that the platform lags behind during each turn performed on place by the walker, since a walker's angular motion without linear displacement triggers no feedforward action.

IV. Extension to Acceleration-Level Control

In order to take into account limitations on linear and angular accelerations imposed by the actuators and/or by perceptual constraints, an acceleration-level control design is more suitable. To this end, the presented smooth velocity-level control law can be transformed to the acceleration level using the theory of *cascaded systems*, see, e.g., [27], or *backstepping* techniques, see, e.g., [28]. Moreover, the availability of platform accelerations allows the analytical computation of the apparent accelerations felt by a walking user. This possibility is especially relevant for the evaluation of dynamic effects on the walker in a full-scale version of the *CyberCarpet* platform.

To pursue an acceleration-level control design, a second-order version of the kinematic model (5) must be considered. This is obtained by simply extending the first-order kinematic models (1) and (5) with the equations

$$\dot{v} = a, \qquad \dot{\omega} = \eta, \tag{25}$$

where a and η are, respectively, the linear and angular acceleration commands of the platform. When a control law for the regulation objectives stated in Sect. III has to be designed on the extended (second-order) system (5)–(25), the availability of smooth stabilizing laws $v = v_d(x, y, \theta)$ and $\omega = \omega_d(x, y, \theta)$ for the first-order system (5) can be exploited. In the following, we present the design for cascaded systems.

The stability of a cascaded system in the form

$$\begin{aligned} \dot{\zeta}_1 &= f_1(t, \zeta_1) + g_1(t, \zeta_1, \zeta_2)\, \zeta_2 \\ \dot{\zeta}_2 &= f_2(t, \zeta_2), \end{aligned} \tag{26}$$

can be concluded, under mild conditions, from the stability of the two subsystems $\dot{\zeta}_1 = f_1(t, \zeta_1)$ and $\dot{\zeta}_2 = f_2(t, \zeta_2)$ (see [27] for details). In order to use this result, we transform system (5)–(25) in the form (26) by defining

$$\zeta_1 = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \quad \xi = \begin{bmatrix} v \\ \omega \end{bmatrix}, \quad \xi_d = \begin{bmatrix} v_d \\ \omega_d \end{bmatrix},$$

$$\zeta_2 = \xi - \xi_d, \quad u(t) = \begin{bmatrix} a(t) \\ \eta(t) \end{bmatrix}.$$

The system equations become

$$\dot{\zeta}_1 = A(\zeta_1)\xi_d(t, \zeta_1) + A(\zeta_1)\zeta_2 \tag{27}$$

$$\dot{\zeta}_2 = u(t) - \frac{d\xi_d(t)}{dt}, \tag{28}$$

where $A(\zeta_1)$ is defined in (6). Equations (27–28) take the form (26) with $f_1(t, \zeta_1) = A(\zeta_1)\xi_d(t, \zeta_1)$, $g_1(\zeta_1) = A(\zeta_1)$, and $f_2(t) = u(t) - (d\xi_d(t)/dt)$.

From Sect. III, we know already that the "downstream" system (27) is asymptotically stable for $\zeta_2 = 0$, i.e., for $\xi = \xi_d$. In order to stabilize the overall cascaded system, it is then sufficient to stabilize the "upstream" system (28) to the

origin, i.e., to bring ζ_2 to zero. This can be easily obtained by choosing, for any positive definite matrix K,

$$u = \dot{\xi}_d - K\zeta_2 = \dot{\xi}_d - K(\xi - \xi_d). \tag{29}$$

The control law (29) requires the differentiability of the first-order velocity control law ξ_d. Indeed, eqs. (17–18) are not differentiable at configurations where the argument of the sgn function is zero. This problem, however, can be overcome by setting $\dot{\xi}_d = 0$ in eq. (29) at such configurations. Note also that an analytical expression of $\dot{\xi}_d$ can be computed from eqs. (17–18) and the model eqs. (5), (25), where the walker velocity V_w is treated locally as a constant.

The above control design has been compared to a backstepping approach in [20], and their relative performance as well as those of acceleration vs. velocity-level control have been evaluated by simulations on the *CyberCarpet*. The reader is referred to [20] for the numerical results. The general outcome is that a very similar behavior is obtained, with slightly longer regulation transients but smoother resulting velocities when acceleration control is applied.

A. Dynamic effects of platform motion on the walker

Due to the platform motion, the 'virtual world' frame attached to the walker is in general non-inertial. In particular, even when the intentional velocity of the walker is constant, she/he will feel 'apparent' accelerations (and thus forces) due to the rotation and/or not uniform translation of the carpet. One major advantage of moving the control action to the acceleration level is that these accelerations can be reliably computed in closed form.

In particular, when walking at constant velocity ${}^wV_w$ in the non-inertial virtual world, the apparent acceleration felt by the user equals the opposite of her/his absolute acceleration. This is obtained by analytic differentiation of the first two equations in (1), using (25) and the acceleration control law (29). The apparent acceleration $\mathbf{a}$ felt by the user can be decomposed into three different components depending, respectively, on the linear and angular accelerations of the frame (X_t, Y_t, Z_t) (*inertial acceleration*), on the square of the angular velocity of this frame (*centrifugal acceleration*), and on the coupling between its angular velocity and the walker relative velocity (*Coriolis acceleration*). These components should be expressed in the frame (X_w, Y_w, Z_w) attached to the walker, see Fig. 3, in order to evaluate their effects on the user. The results of these computations are reported below, where $\mathrm{Rot}(\beta)$ is the 3×3 orthonormal matrix associated to a rotation by an angle β around the $Z = Z_w$ axis:

- Inertial component

$${}^w\mathbf{a}_{in} = \mathrm{Rot}(-\theta_w)\left(\mathrm{Rot}(\theta)\begin{bmatrix} a \\ 0 \\ 0 \end{bmatrix} + \eta \begin{bmatrix} -y \\ x \\ 0 \end{bmatrix}\right);$$

- Centrifugal component

$${}^w\mathbf{a}_{cen} = \omega^2\,\mathrm{Rot}(-\theta_w)\begin{bmatrix} x & y & 0 \end{bmatrix}^T;$$

- Coriolis component

$${}^w\mathbf{a}_{Cor} = 2\begin{bmatrix} 0 & 0 & \omega \end{bmatrix}^T \times {}^wV_w,$$

where the symbol $\times$ denotes the vector product.

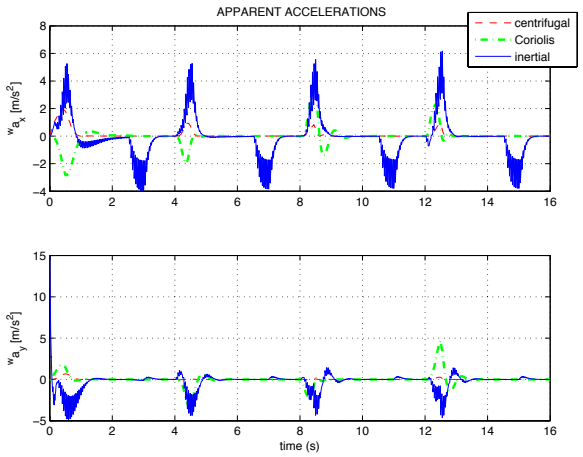


Fig. 10. Inertial, centrifugal and Coriolis components of the apparent acceleration felt by a user in the X_w (top) and Y_w (bottom) directions while walking along a square virtual path with acceleration control commands applied to the platform

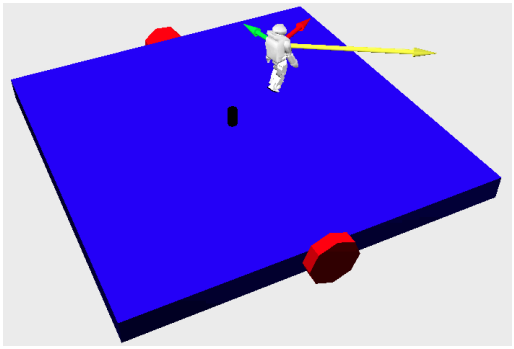


Fig. 11. A sample graphical output of the developed dynamic simulator. The arrows represent (from the right, counterclockwise) the inertial (yellow), centrifugal (red) and Coriolis (green) components of the apparent acceleration felt by the walker on the controlled platform

Figure 10 displays the inertial, Coriolis and centrifugal components of the apparent acceleration felt by the user in the X_w and Y_w directions, when the walker is executing the (virtual) square path considered in Fig. 7 and the platform is controlled by acceleration commands. The control parameters are the same used in Sect. III, whereas $K = \mathrm{diag}(20, 20)$ in eq. (29). Should the total apparent acceleration be too large for the perceptual comfort of the walker, control gains and saturations would need to be adjusted accordingly.

For an extensive evaluation of dynamic effects, we have devised a 3D simulator of a walker moving on the *CyberCarpet*, where the various system and control parameters can be varied at will. A snapshot of the resulting graphical output is shown in Fig. 11, where colored arrows applied to the user body represent the various accelerations felt during a walk on the controlled platform. A video of the square path walk is attached to this paper, while videos of other motion tasks are available at http://www.dis.uniroma1.it/labrob/research /CW.html.

V. DESIGN OF THE SMALL-SCALE PROTOTYPE

This section illustrates the relevant issues in the mechanical design and realization of a small-scale prototype of the *CyberCarpet*. The general guidelines in the design require the platform to be stiff enough and capable of carrying the weight of a human (100 kg), to react very quickly to the control commands, and to be reliable and easy to interface [29]. Figure 12 shows an overview of the whole system, including the used test vehicle $\langle 1 \rangle$. For further details. the reader is referred to [30].

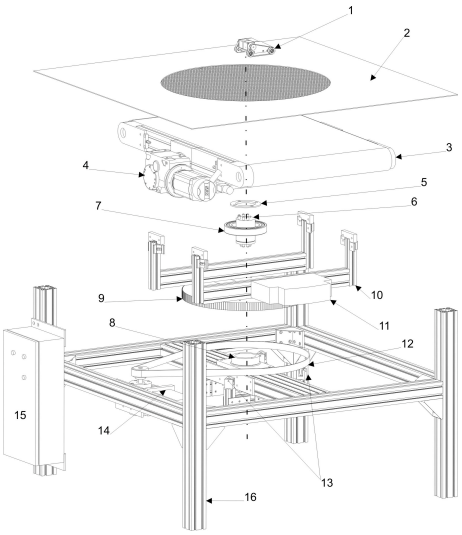


Fig. 12. Overview of the components of the *CyberCarpet* platform (referred to with $\langle\,\rangle$ in the text)

A. Ball-array carpet

The array of balls (see $\langle 2 \rangle$ in Fig. 12) is one of the most crucial elements of the platform. The resulting motion imposed to the walker strongly depends on a well functioning interaction between belt, balls (with their supporting grid plate), and shoe sole. There are also major haptic issues that influence the perception of the walking surface by the subject. After testing different ball sizes with a number of subjects who provided their impressions, a diameter $d_{ball} = 8$ mm has been chosen. The ball array is allocated in a supporting grid that keeps the balls in place. The gap between the balls has been set to $0.5\, d_{ball}$ to provide robustness and uniform feeling to the floor.

For determining appropriate materials of belt, balls, and supporting grid, one major aspect concerns the resulting friction forces at the different contact points and surfaces, see Fig. 13. The ideal combination of materials exhibits maximum friction at the belt-ball and ball-shoe contacts, and minimum friction at the ball-grid and grid-belt interfaces. The ball material was chosen as INOX steel, since it is easily available and proven to be a reliable solution. The chosen belt is a Transilion E8/2 U0/V5, from the Fa. Siegling company, which is covered by a soft PVC layer of 0.5 mm displaying a friction coefficient to inox larger than 0.7.

The grid itself can be realized with different materials, comparing their stability and slip-stick parameters. An acetal POM plate manufactured by CNC die-cutting has been selected, leading to friction coefficients of 0.3 and 0.6 with the balls and belt, respectively. The final array contains 4332 balls and fits into an hexagonal grid. The diameters of the inscribed and circumscribed circles are 693 mm and 800 mm.

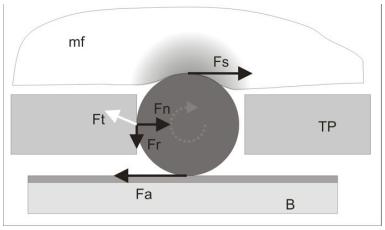


Fig. 13. Diagram of friction forces on a single ball. The ball is in contact with the shoe sole insisting with a mass m_f. When actuated, the belt applies a force F_a to the ball, which gets in point contact with the cylindrical hole of the supporting grid where forces F_n and F_r occur. The resulting force F_s is finally transmitted to the walker on the platform.

B. Belt

The actuated belt $\langle 3 \rangle$ in Fig. 12 provides the linear velocity command v to the kinematic system (1–5). The belt, together with its electrical drive $\langle 4 \rangle$ and corresponding servo controller $\langle 11 \rangle$, is rigidly mounted on the supporting beams $\langle 10 \rangle$. In order for the belt to run straight, a proper value for its length-to-width ratio is 1.4. Since the belt width is prescribed by the ball array size, this led to a belt length of 1100 mm. Another issue is the position of the (single) drive actuating the belt, which should always keep the upper strand in a tight span. For this, we placed the drive at the belt center, using an additional pulley placed underneath the belt body: the lower span is thus divided into two small parts and the upper part of the belt is always under direct tension of the actuating force.

C. Turntable

The turntable $\langle 9 \rangle$ has to bear the weight of the belt/ball-array system and of the walker on the platform, and provides the angular velocity command ω to the kinematic system (1–5). The turntable is primarily fixed by the ball bearing $\langle 7 \rangle$ in combination with the mounting plate $\langle 8 \rangle$, and the actuating torque is applied to the turntable by the drive $\langle 14 \rangle$ via the toothed belt $\langle 12 \rangle$. Vertical reaction forces are also provided by the support wheels $\langle 13 \rangle$ that stabilize the turntable. Since the system should be able to rotate endlessly, current and data lines are fed through the rotational feedthrough $\langle 6 \rangle$, which is secured with the ring $\langle 5 \rangle$. The whole system is carried by the framework $\langle 16 \rangle$ of aluminum beams. The moving (rotating) parts of the whole system supported by this framework weigh about 200 kg.

D. Control and system drives

The hardware architecture of the control system is shown in Fig. 14. There are two Lenze (9300 series) low-level servocontrollers for the two identical drives of the belt and the turnable. We used two Lenze three-phase motors, type

MCA 10, providing a power of 0.8 kW with a supply frequency of 140 Hz. The maximum torque is 2.0 Nm at the speed of 3950 min^{-1}. They actuate the belt through a gear with ratio $1{:}19.556$, and the turntable through a gear and the toothed belt $\langle 12 \rangle$, having a ratio of $1{:}8.012$ and $1{:}8.217$, respectively. As a result, the belt can reach for continuous operation a linear speed of 2 m/s, with a maximum acceleration of 5 m/s^2, while the maximum angular velocity and acceleration of the turntable are 2 rad/s, and 4.4 rad/s^2, respectively.

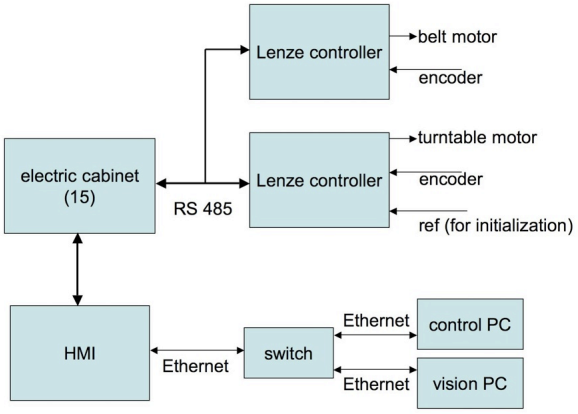


Fig. 14. Control and system drives architecture

Both drives are equipped with integrated incremental encoders, and can be controlled up to a maximum sampling frequency of 100 Hz. The servocontroller for the belt drive is mounted on the rotating part of the machine, and uses a RS485 bus for communication. The turntable drive is equipped with a reference switch used to initialize the encoder count for platform orientation. The PC with the high-level control laws sends the velocity inputs v and ω via Ethernet to a Human Machine Interface (HMI), where the commands are analyzed and suitably translated to be sent to the low-level servocontrollers. Moreover, the HMI checks the parameters of both drives and controllers to detect unnatural behaviors. This structure provides safety functions to the platform.

VI. Visual localization

A visual tracker is used to find the position (x, y), as well as the orientation θ_w of the walker, on the *CyberCarpet*. Our visual tracker is an adaptation of a color-based particle filter [31] and has been tested experimentally at a full scale, with a human user walking (and changing posture) randomly in the field of view of an overlooking camera. The tracker uses a set of *particles* to model the posterior probability distribution of the state of the walker, i.e., of her/his position on the floor and (planar) orientation.

Particle filtering is an estimation approach where multiple hypotheses $\boldsymbol{s}^{(1)}, \ldots, \boldsymbol{s}^{(N)}$ exist at the same time and are kept during the estimation process. In our application, each hypothesis (or particle) $\boldsymbol{s}^{(j)}$ represents one possible state of the walker, with a corresponding discrete sampling probability $\pi^{(j)}$, $j = 1, \ldots, N$. The particle type considered in [31] consisted of an elliptical blob with a position and a varying size, describing the boundary of the object being tracked. Since the size of the walker is considered to be constant in our case, each particle (the index j is dropped for compactness) has been initially specified as

$$\boldsymbol{s} = \{x, y, \theta_w\}, \tag{30}$$

where x and y represent the position of the center of the ellipse and θ_w is the orientation angle of the minor axis of the ellipse, which corresponds to the walker forward direction. However, due to perspective changes and the head bobbing about, the appearance of the user changes continuously during the walk, and the localization algorithm was not fully able to track the walker position in a reliable way. Therefore, the model (30) was extended from a simple ellipse to the combination of an elliptical shoulder region and a circular head region, as seen in a top-view of a person and illustrated in Fig. 15. The description of the particle is

$$\boldsymbol{s} = \{x, y, \theta_w, c_x, c_y\}, \tag{31}$$

where c_x and c_y are the position of the head circle relative to the ellipse center.

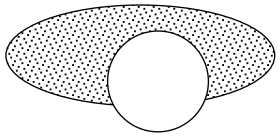


Fig. 15. Each particle is modeled by an ellipse and a circle, representing the top-view of the shoulders and head, parametrized as in eq. (31)

The N particles are initialized using for all of them the same set of values entered manually (by mouse clicks) on the user interface. At each iteration t of the visual localization algorithm, *three steps* are performed to compute the updated set of hypotheses from the one available at the previous iteration $t-1$ and the corresponding estimation of the walker position and orientation.

A. Particle filtering

In the first step, each particle $\boldsymbol{s}_{t-1}$ of the previous iteration is propagated according to a simple dynamic model. In particular, the evolution $\boldsymbol{s}_t$ is computed as

$$\boldsymbol{s}_t = \boldsymbol{s}_{t-1} + \boldsymbol{w}_{t-1}, \tag{32}$$

where $\boldsymbol{w}_{t-1}$ is a multivariate Gaussian random variable that models the walker motion in the time interval between two iterations. This generates the prior distribution of the next state.

B. Color-based observation

To test the probability of the evolved particle $\boldsymbol{s}_t$ being a good hypothesis, the current image captured by the overhead camera is observed. A color histogram p is computed for the shoulder region and another histogram p' for the head region. The shoulder region is defined by all the pixels inside the ellipse, excluding those inside the head circle. The head region is defined by all the pixels inside the head circle. The resulting histogram p is normalized and compared to

a stored histogram or target shoulder model q (acquired at the tracker initialization) using the *Bhattacharyya coefficient* $\rho[p,q]$, which is explained in more detail in [31]. The same is done for p' and target head model q', resulting in $\rho[p',q']$. The distance between the particle and the target model is defined by the *Bhattacharyya distance* [32]

$$d = \sqrt{1 - \frac{\rho[p,q] + \rho[p',q']}{2}}. \tag{33}$$

This similarity measure provides a likelihood of each particle that will be used to update the particle set in the next step. In fact, each element $\boldsymbol{s}^{(j)}$ of the particle set can be assigned a probability $\pi^{(j)}$ in terms of the observations (color histogram), namely

$$\pi^{(j)} = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{d^2}{2\sigma^2}\right), \tag{34}$$

where σ is the standard deviation of the Gaussian distribution (a constant that can be fine tuned). The $\pi^{(j)}$, $j = 1,\ldots,N$, are further normalized so as to define the discrete posterior probability distribution of the walker state.

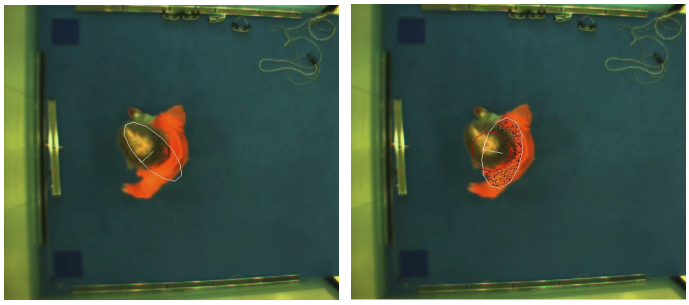
C. Updating the particle set and tracker state

The third step of each iteration generates the new set of hypotheses used for updating the tracker state (the new estimation of the walker position and orientation). The new set of N particles is drawn from the current set by choosing the generic particle $\boldsymbol{s}^{(j)}$ with probability $\pi^{(j)}$. Particles are drawn with replacement, so that those with higher probability are in general selected several times. Finally, the updated tracker state is computed as a weighted mean over all the current particles, using their Bhattacharyya distance (33) to the target model as weights.

D. Final considerations

Given the increased number of degrees of freedom in the adapted model (31), our experiments have shown that a minimum of $N = 500$ particles are needed to correctly track the walker. This, however, slowed down the visual tracking rate to 3 Hz. Considering the high resolution of the input image and the relative size of the ellipse, it appeared that more pixels than necessary were evaluated for building the histograms. Thus, we introduced a *random sampling*, picking 500 random points within the ellipse region. These points were stored as a vector of (p_x, p_y) positions relative to the ellipse center (x, y) and angle of orientation θ_w. The same has been done for the head region. The histograms could then be computed faster by evaluating only the pixels corresponding to these points, and the final visual localization algorithm could run up to 17 Hz.

Typical snapshots of the tracker state for a human walker in a room are shown in Fig. 16, where the 3-dimensional particle (30) and the 5-dimensional particle (31) have been used, respectively in (a) and in (b). It is apparent that the correct posture is lost (at times) in the first case, motivating the use of the extended model. Complete videos of these two human tracking experiments are included in the material accompanying the paper. Note that for the experiments with the small-scale *CyberCarpet* platform in Sect. VII, the same visual algorithm working for human walkers was used. Since the walker was replaced by a mobile robot mock-up, the rate of the tracker was limited to 10 Hz without performance loss.



(a) (b)

Fig. 16. Snapshots of the visual localization algorithm tracking a walking person; the elliptical model without head region (30) is used in (a), while the model with head region (31) is used in (b), resulting in a more accurate tracking

VII. Experimental results

The velocity-level control algorithms described in Sect. III have been implemented and validated on the small-scale *CyberCarpet* prototype described in Sect. V. The walker was emulated by a differentially-driven tracked mobile robot carrying on top a picture of a human body (see Fig. 17, including at typical camera view with superimposed localization ellipse/circle). Its position on the platform was localized with the technique of Sect. VI, using a Sony DFW-VL500 Color/VGA camera, with 640×480 pixels at 30 fps, placed at about 131 cm over the surface of the carpet.

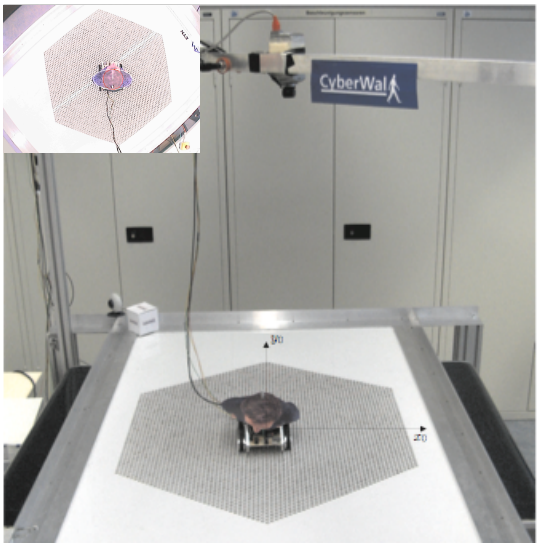


Fig. 17. The experimental set up with the *CyberCarpet*, the mobile robot carrying a picture of a human body, and the overhead camera for visual tracking; in the top-left box, a view from the overhead camera with superimposed localization

The testing campaign involved four different motions for the vehicle. In particular, the following scenarios were chosen:

A. Standing still off the origin;

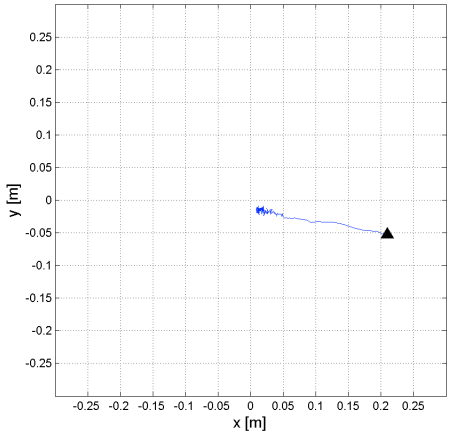


Fig. 18. Absolute trajectory in the experiment of Sect. VII-A (standing still)

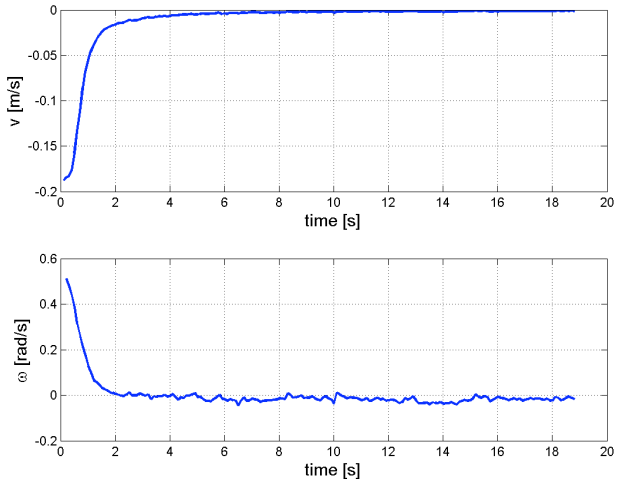


Fig. 19. Linear and angular velocity commands for the trajectory of Fig. 18

B. Starting at the origin and moving along a straight line with a constant speed of about 0.22 m/s;

C. Starting at the origin and moving along a circular path of radius 0.35 m with a constant speed of about 0.14 m/s;

D. Starting at the origin and moving along a square path of side 0.4 m with a constant linear velocity of about 0.1 m/s, and turning at the corners with an angular velocity of about $\pi/4$ rad/s.

As for the control laws, for the three scenarios *B* to *D* where the user is in motion, we comparatively tested the pure static feedback law $v = v_{fb}$, $\omega = \omega_{fb}$ given in (17–18) alone against the full feedback/feedforward strategy $v = v_{fb} + v_{ff}$, $\omega = \omega_{fb} + \omega_{ff}$ of (22), (23–24). The parameters of the control laws were set to $k = 4$ in eqs. (17–18), $k_w = 0.3$ in eqs. (23–24), and $\omega_{ff,max} = 0.05$ rad/s as saturation level in eq. (22). Videos of all experiments accompany the paper (also available at http://www.dis.uniroma1.it/labrob/research/CW.html).

A. Standing still

In the first experiment, the vehicle is placed in the absolute position $(0.2, -0.05)$ m and keeps a zero intentional velocity during the whole experiment. In Fig. 18 the planar absolute trajectory executed under the control law (17–18) is shown, with a black triangle marker representing the starting position. The corresponding behavior of the linear and angular velocity commands is displayed in Fig. 19. Note that the recovering path in absolute space is close to the straight line predicted in the control analysis of Sect. III. The discrepancies are due to the presence of noise in the image processing step, to the discrete sampling of the measurements (10 Hz on the position (x, y)), and to the discrete sampling of the control output (10 Hz on the commanded platform velocities).

B. Moving at constant velocity

In the second experiment, we tested separately the control laws (17–18) and (22), (23–24), i.e., without or with the on-line velocity estimation and compensation. The robot moves along a straight line with a constant speed of about 0.22 m/s.

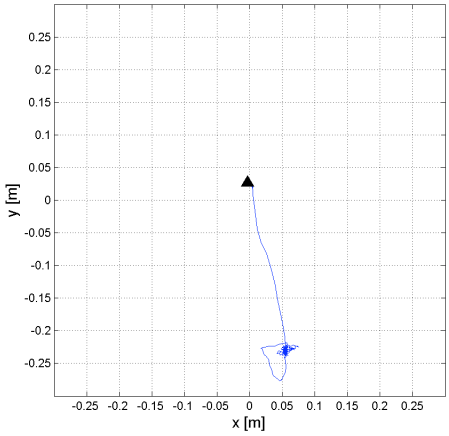


Fig. 20. Absolute trajectory in the experiment of Sect. VII-B.1 (moving at constant velocity, with pure feedback control)

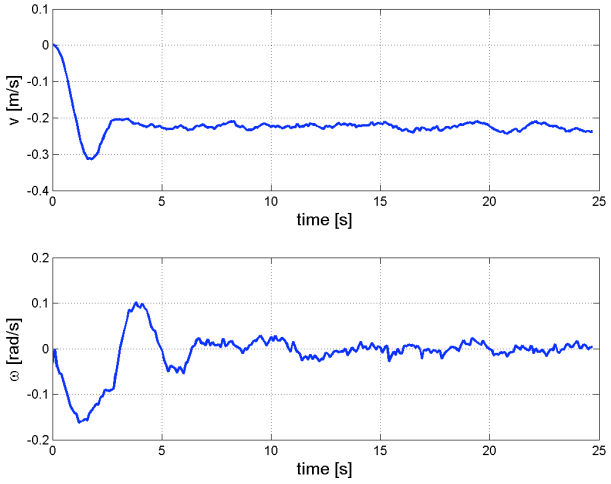


Fig. 21. Linear and angular velocity commands for the trajectory of Fig. 20

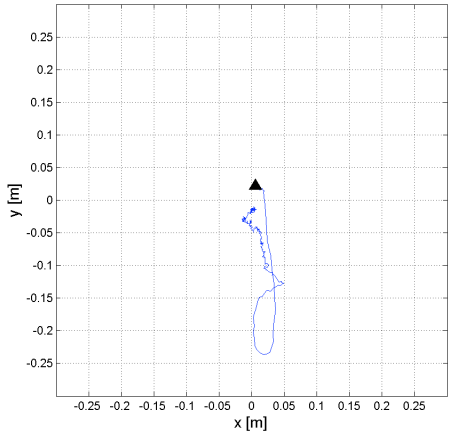


Fig. 22. Absolute trajectory in the experiment of Sect. VII-B.2 (moving at constant velocity, with feedforward control action)

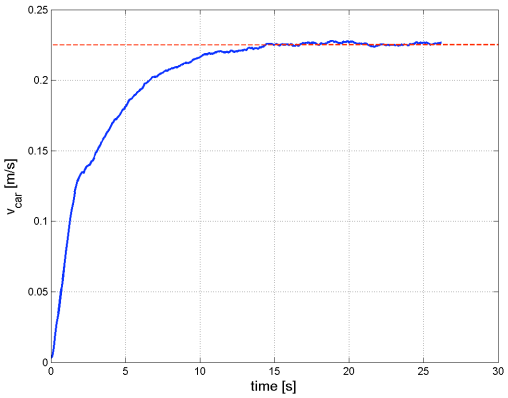


Fig. 23. Estimation of the intentional speed for the trajectory of Fig. 22

1) Pure feedback law (17–18): The absolute trajectory of the vehicle is reported in Fig. 20. The starting point is at $(0, 0.02)$ m (black triangle) and motion proceeds in the negative y-direction ($-Y_0$). As expected, the static feedback (17–18) is not able to fully compensate the persistent intentional motion, but a steady state is obtained after about 6 s with a non-zero constant position error. The reached equilibrium position is at about $(0.05, -0.23)$ m (this value depends on the chosen control gains) and the control is not able to recenter the robot. The linear and angular velocity commands are shown in Fig. 21. Note that, after an initial transient, the linear velocity command matches the user intentional velocity, while the angular command is close to zero.

2) Complete feedback/feedforward law (22), (23–24): In this case, the additional presence of the feedforward action based on the observer (23–24) is able to fully recover the platform center despite the persistent intentional motion, similar to what could be obtained by an integral control action. The vehicle starts at $(0.006, 0.022)$ m (black triangle) and moves in the negative y-direction as before. After a transient phase, the estimate of the intentional speed converges to the actual value (see Fig. 23) and the control law (22) brings back the robot to the origin (see Fig. 22). The commanded linear and angular platform velocities are shown in Fig. 24. In comparison with Fig. 21, the angular velocity appears now more erratic, especially toward the end of the motion (i.e., close to the platform center). This effect is due to the presence of the term $1/R$ in the feedforward term ω_{ff} of eq. (22), which grows unbounded as R goes to zero. The saturation introduced in eq. (22) helps in softening this effect, but cannot avoid some chattering around the origin (see Fig. 25 for the time behavior of ω_{ff}).

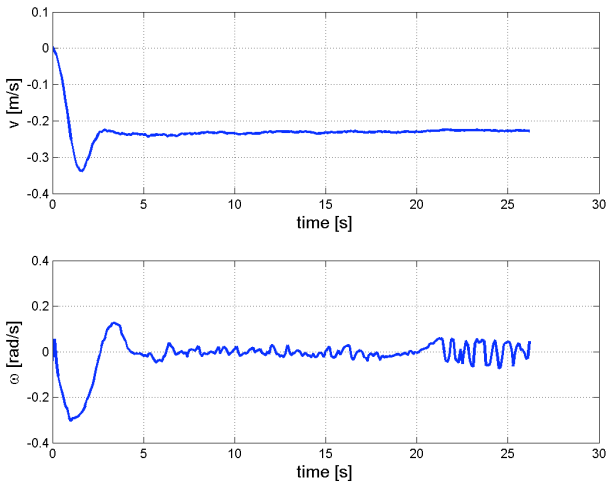


Fig. 24. Linear and angular velocity commands for the trajectory of Fig. 22

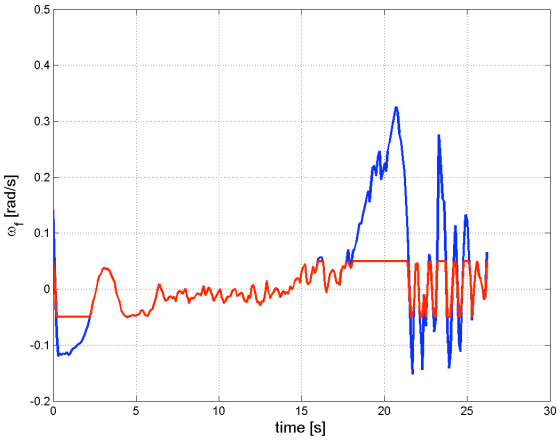


Fig. 25. The angular feedforward term ω_{ff} without (blue line) and with (red line) saturation for the trajectory of Fig. 22

C. Moving along a circular path

In this experiment, the vehicle is moving along a circular path of radius 0.35 m with a constant speed of about 0.14 m/s, and, as a result, with a constant angular velocity of about 0.4 rad/s. This test case is significantly different from the pre-

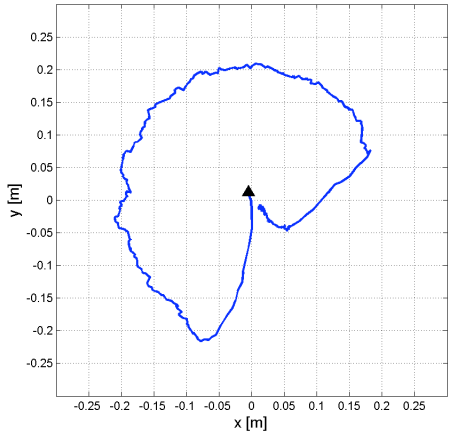


Fig. 26. Absolute trajectory in the experiment of Sect. VII-C.1 (circular path, with pure feedback control)

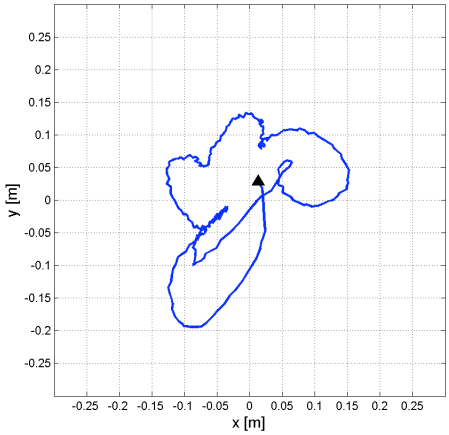


Fig. 28. Absolute trajectory of the car in the experiment of Sect. VII-C.2 (circular path, with feedforward control action)

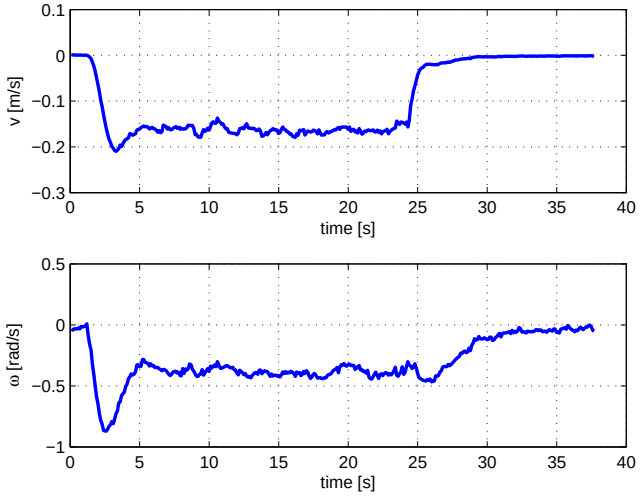


Fig. 27. Linear and angular velocity commands for the trajectory of Fig. 26

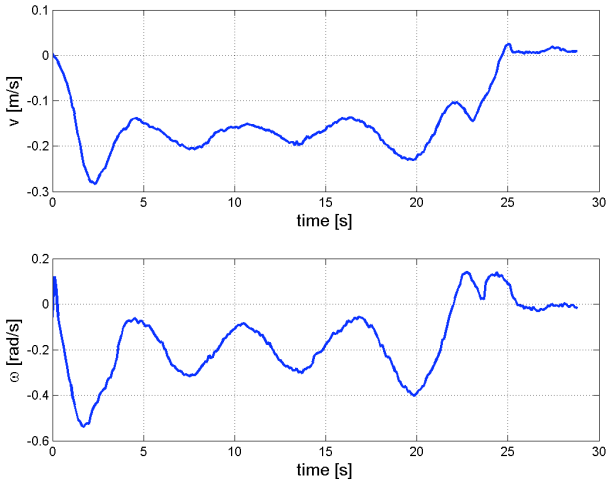


Fig. 29. Linear and angular velocity commands for the trajectory of Fig. 28

vious one, since the intentional velocity vector is continuously changing direction during motion. This is a more demanding task for the disturbance observer, which needs to track a highly time-varying signal.

1) Pure feedback law (17–18): The absolute trajectory of the vehicle is shown in Fig. 26. As expected, the control law is only able to partially compensate for the intentional motion: the robot is kept within a distance of 0.2 m from the platform, which is anyway smaller than the original dimension of the circular path (0.35 m). The corresponding platform velocity commands are shown in Fig. 27. It is interesting to note that, after the initial transient, the platform linear and angular velocities match the actual linear (0.14 m/s) and angular (0.4 rad/s) velocity of the vehicle, confirming again that a steady-state condition has been reached. At about $t = 23.5$ s, the robot stops its motion and is thus brought back to the center of the platform.

2) Complete feedback/feedforward law (22), (23–24): Despite the more challenging task for the observer, the complete feedback/feedforward control law is able to keep the user closer to the platform center than in the previous case, as shown by the absolute trajectories in Figs. 26 and 28 for the two experiments. The velocity commands sent to the platform are shown in Fig. 29.

We note that the convergence of the intentional speed estimate is not perfect (see Fig. 30), so that a residual motion around the platform center is still present in Fig. 28. The reason of this behavior is intrinsic to the structure of the proposed observer. Indeed, the estimated velocity is a low-pass filtered version of the actual velocity, with cut-off frequency given by the observer gain k_w. From standard linear analysis, when the input signal (the actual velocity of the vehicle) has a sinusoidal behavior, the observer output will result in a sinusoid with the same frequency, but different amplitude and phase. In particular, since we have set $k_w = 0.3$, and the frequency of the input signal is 0.4 rad/s (the angular velocity of the robot), the resulting estimated velocity will be attenuated by -4.5 dB and the phase shift will be 0.92 rad, equivalent

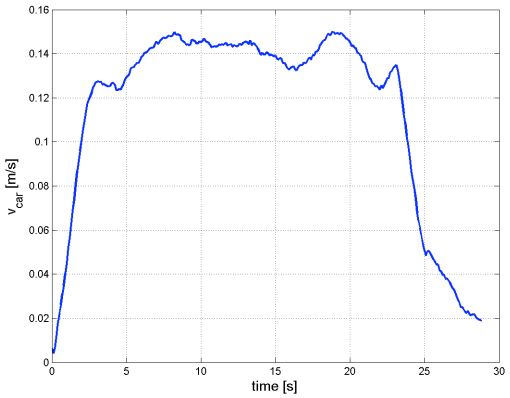


Fig. 30. Estimation of the intentional speed for the trajectory of Fig. 28

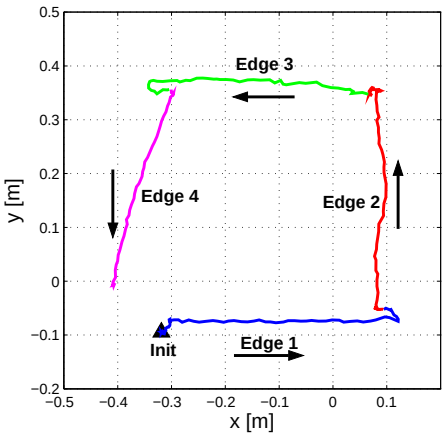


Fig. 31. Trajectory of the user in the virtual world while executing a square path

to a time delay of about 2.3 s. Therefore, even at steady state, the estimated velocity lags behind the actual one, thus preventing the controller to complete a perfect recover of the position to the origin. Accordingly, the estimated intentional speed oscillates around its nominal value of 0.14 m/s, before dropping to zero starting from $t = 23.5$ s when the robot suddenly stops.

D. Moving along a square path

In this last experiment, the vehicle travels along a square path of 0.4 m side with constant linear velocity of about 0.1 m/s. The absolute trajectory in the virtual world during the execution of the square path is shown in Fig. 31 (different colors are used for each side). The robot starts from the *Init* position (black triangle) and moves to the left along Edge 1 of the square. Then it stops, turns 90° counterclockwise, and starts travelling along Edge 2, repeating the same sequence until tracing the complete square. The total motion time is thus approximately 24 s. The mismatch between the actual trajectory in Fig. 31 and the ideal commanded square path is mainly due to slippage of the tracks of the mobile vehicle during motion and, to a less extent, to the inaccurate execution of its commanded velocities.

1) Pure feedback law (17–18): The static feedback law is able to partially compensate for the intentional motion, keeping the vehicle within a circle centered at the origin and having a radius of about 0.2 m (Fig. 32). The velocity commands sent to the platform are shown in Fig. 33.

2) Complete feedback/feedforward law (22), (23–24): The absolute motion for this case is shown in Fig. 34, while the velocity commands sent to the platform are given in Fig. 35. The benefits of the estimation of the intentional velocity are not so evident as in the previous cases. In particular, the controlled motion of the vehicle remains in an area around the platform center that is as wide as when using a pure feedback law. This is mainly due to the slow convergence of the velocity observer w.r.t. the duration of the motion along each side of the square. By looking closer at the behavior of the estimated speed in Fig. 36, it is clear that 4 s of linear motion along each side are not sufficient for convergence with

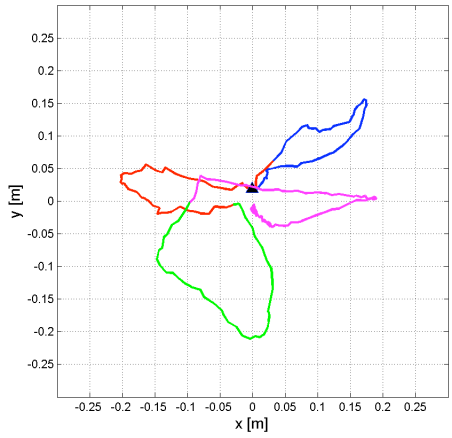


Fig. 32. Absolute trajectory in the experiment of Sect. VII-D.1 (square path, with pure feedback control)

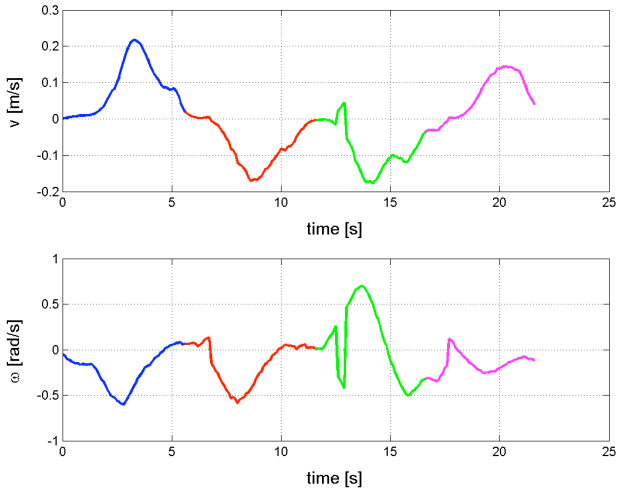


Fig. 33. Linear and angular velocity commands for the trajectory of Fig. 32

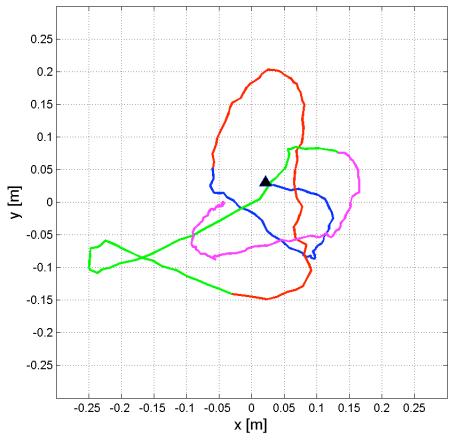


Fig. 34. Absolute trajectory in the experiment of Sect. VII-D.2 (square path, with feedforward control action)

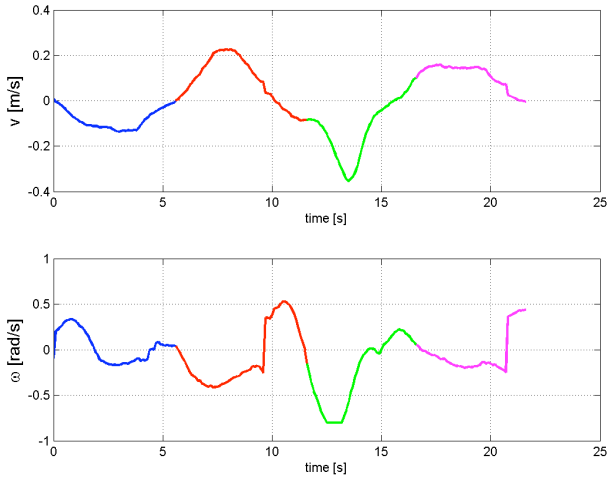


Fig. 35. Linear and angular velocity commands for the trajectory of Fig. 34

the chosen observer gains. The estimate always lags behind the actual speed: when the vehicle is moving at 0.1 m/s, increasing maximum values of 0.08, 0.094, and 0.098 are reached along the first, second, and third side. Similarly, when the vehicle stops and turns in place at the corners, the estimate drops toward zero though not reaching this final value. Indeed, higher observer gains would largely improve this behavior, but these were not allowed by measurement noise and feasible sampling times.

VIII. Conclusions

We have presented the design and implementation of motion control laws for the *CyberCarpet*, a novel concept of platform that re-centers a user in unlimited locomotion by combining the linear and angular mobility of a treadmill mounted on a turntable with the presence of a ball-array carpet. Despite of the restricted local mobility, due to the presence of a nonholonomic constraint on the instantaneous velocities, the controller is able to keep a freely walking user close to the

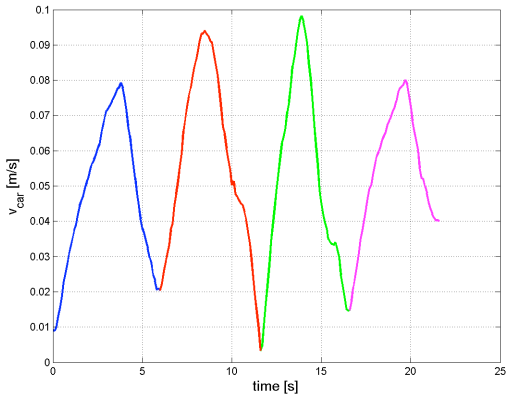


Fig. 36. Estimation of the intentional speed for the trajectory of Fig. 34

platform center in a natural way.

The control law is designed at the velocity level as the composition of a feedback and a feedforward action. The nonlinear feedback is a singularity-free version of an input-output linearization law, while the feedforward term is based on an estimation of the unknown walker intentional velocity through a disturbance observer. The feedback gains can be conveniently scaled so as to decrease the control velocities to perception levels that are acceptable by the user. The smoothness of the velocity control law allows a direct extension to the design of an acceleration-level controller, which was performed using the theory of cascaded systems. This enables to take into account also acceleration bounds due to user's perception constraints, as well as to perform a dynamic analysis of inertial forces acting on the walker.

To validate the *CyberCarpet* principle and to test the actual performance of the motion control laws, a small-scale platform has been designed and built having in mind scalability properties. In particular, from the mechanical and actuation points of view, the prototype is already capable of sustaining and moving the full weight of a person. Moreover, the method based on particles filter for visual localization by an overhead camera has been already tested on human walkers in arbitrary motion, proving to be quite robust. Using this setup, a series of velocity-level control experiments have been conducted with a mobile robot vehicle as a mock-up of the walking user. Comparison between pure feedback control and the complete feedback/feedforward scheme has shown the benefits of compensating for the unknown intentional velocity, especially for smooth vehicle motion. In particular, the absence of sharp turns allows a faster convergence of the observer to the intentional linear velocity, and thus its full cancellation.

The main limitations that we found were related to the high level of noise in the measurement of the vehicle position, and to the relatively low sampling frequency of the visual localization. The walker's orientation, though available from visual estimation, has not been used in the control law, thus preventing the prediction of intentional turns and delaying the re-centering. Nonetheless, the robot vehicle never approached dangerously the platform border, proving the effectiveness of velocity-level control up to intentional velocities of 0.25 m/s,

comparable to the diameter of 0.8 m of the prototype. At this scale, resort to acceleration control was found not necessary.

Having proven the feasibility of the *CyberCarpet* concept in all its components, the next step would be the construction of a full-scale device to have a human user walk at normal speed while being immersed in a Virtual Reality environment. Our parallel experience with the other omni-directional belt-array platform developed within the *CyberWalk* project [16], [17] suggests that the compact mechanical principle underlying the ball-array platform may still be a convenient choice in terms of weight, needed power, ease of maintenance, and reduced part wearing. The total weight of the moving parts of the 5×5 m omni-directional *CyberWalk* platform is over 7500 kg, with the need of a combined electrical and hydraulic actuation. A full scale ball-array platform with similar performance could be of smaller size/weight and more efficient due to its actuation mechanism. In fact, the *CyberCarpet* tends always to align its main belt with the motion direction of a persistent walking user. Thus, in a steady-state condition, only one dof needs to be actuated, considerably reducing the energy consumption. Conversely, the belt-array *CyberWalk* platform needs in general full 2D actuation for compensating a persistent walk. Additional investigations are required to establish the perceptual effects of the combined linear and angular motion imposed by the *CyberCarpet* on the walker especially during transients, so as to set accordingly control design constraints. The introduced transposition of the smooth control design to the acceleration level represents a needed and useful step.

ACKNOWLEDGMENTS

This work has been supported by the European Commission as part of the STREP project FP6–511092 *CyberWalk*.

REFERENCES

[1] *CyberWalk*, "EU STREP Project FP6–511092, http://www.cyberwalk-project.org," 2005.
[2] H. Iwata, "Locomotion interface for virtual environments," in *Proc. 9th Int. Symp. on Robotics Research*, pp. 275–282, 2000.
[3] J. M. Hollerbach, "Locomotion interfaces," in *Handbook of Virtual Environments Technology* (K. M. Stanney, ed.), pp. 239–254, Lawrence Erlbaum Associates, 2002.
[4] J. M. Hollerbach, "Locomotion interfaces and rendering," in *Haptic Rendering: Foundations, Algorithms, and Applications* (M. Lin and M. Otaduy, eds.), A. K. Peters, 2008.
[5] J. M. Hollerbach, Y. Xu, R. R. Christensen, and S. C. Jacobsen, "Design specifications for the second generation Sarcos Treadport locomotion interface," in *Haptic Symposium, Proc. of ASME Dynamic Systems and Control Division*, pp. 1293–1298, 2000.
[6] R. C. Hayward and J. M. Hollerbach, "Implementing virtual stairs on treadmills using torso force feedback," in *Proc. IEEE Int. Conf. on Robotics and Automation*, (Washington, DC), pp. 586–591, 2002.
[7] D. Checcacci, J. M. Hollerbach, R. Hayward, and M. Bergamasco, "Design and analysis of a harness for torso force applications in locomotion interfaces," in *Proc. EuroHaptics Conf.*, (Dublin, IR), pp. 53–67, 2003.
[8] H. Noma, T. Sugihara, and T. Miyasato, "Development of Ground Surface Simulator for Tel-E-Merge System," in *Proc. IEEE Virtual Reality Conf.*, pp. 217–224, 2000.
[9] H. Iwata, H. Yano, H. Fukushima, and H. Noma, "CirculaFloor," *IEEE Computer Graphics and Applications*, vol. 25, no. 1, pp. 64–67, 2005.
[10] R. Darken, W. Cockayne, and D. Carmein, "The Omnidirectional Treadmill: A locomotion device for virtual worlds," in *Proc. Symp. User Interface Software and Technology*, pp. 213–221, 1997.
[11] H. Iwata, "The Torus Treadmill: Realizing locomotion in VEs," *IEEE Computer Graphics and Applications*, vol. 9, no. 6, pp. 30–35, 1999.
[12] K. J. Fernandes, V. Raja, and J. Eyre, "Cybersphere: The fully immersive spherical projection system," *Communications of the ACM*, vol. 46, no. 9, pp. 141–146, 2003.
[13] A. Nagamori, K. Wakabayashi, and M. Ito, "The Ball Array Treadmill: A locomotion interface for virtual worlds," in *Work. on New Directions in 3D User Interfaces (at VR 2005)*, (Bonn, D), 2005.
[14] J.-Y. Huang, "An omnidirectional stroll-based virtual reality interface and its application on overhead crane training," *IEEE Trans. on Multimedia*, vol. 5, no. 1, pp. 39–51, 2003.
[15] A. De Luca, R. Mattone, P. Robuffo Giordano, and H. H. Bülthoff, "Control design and experimental evaluation of the 2D CyberWalk platform," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, (St. Louis, MO), pp. 5051–5058, 2009.
[16] J. Souman, P. Robuffo Giordano, I. Frissen, A. De Luca, and M. Ernst, "Making virtual walking real: Perceptual evaluation of a new treadmill control algorithm," *ACM Trans. on Applied Perception*, vol. 7, no. 2, pp. 11:1–11:14, 2010.
[17] J. Souman, P. Robuffo Giordano, M. Schwaiger, I. Frissen, T. Thümmel, H. Ulbrich, A. De Luca, H. H. Bülthoff, and M. Ernst, "CyberWalk: Enabling unconstrained omnidirectional walking through virtual environments," *ACM Trans. on Applied Perception*, vol. 8, no. 4, 2011.
[18] A. De Luca, R. Mattone, and P. Robuffo Giordano, "The motion control problem for the *CyberCarpet*," in *Proc. IEEE Int. Conf. on Robotics and Automation*, (Orlando, FL), pp. 3532–3537, 2006.
[19] A. De Luca, R. Mattone, and P. Robuffo Giordano, "Feedback/feedforward schemes for motion control of the *CyberCarpet*," in *Proc. 8th IFAC Symp. on Robot Control*, (Bologna, I), 2006.
[20] A. De Luca, R. Mattone, and P. Robuffo Giordano, "Acceleration-level control of the *CyberCarpet*," in *Proc. IEEE Int. Conf. on Robotics and Automation*, (Roma, I), pp. 2330–2335, 2007.
[21] H. Noma and T. Miyasato, "Design for Locomotion Interface in a Large Scale Virtual Environment – ATLAS: ATR Locomotion Interface for Active Self Motion," in *Proc. 7th Annual Symp. on Haptic Interface for Virtual Environments and Teleoperated System (with ASME Winter Meeting*, pp. 111–118, 1998.
[22] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino, "Closed loop steering of unicycle-like vehicles via Lyapunov techniques," *IEEE Robotics & Automation Mag.*, vol. 2, no. 1, pp. 27–35, 1995.
[23] A. Isidori, *Nonlinear Control Systems*. Springer-Verlag, 3rd ed., 1995.
[24] C. Samson, "Time-varying feedback stabilization of car-like wheeled mobile robots," *Int. J. of Robotics Research*, vol. 12, no. 1, pp. 55–64, 1993.
[25] M. K. Bennani and P. Rouchon, "Robust stabilization of flat and chained systems," in *Proc. 3rd European Control Conf.*, (Roma, I), pp. 2642–2646, 1995.
[26] O. J. Sørdalen and O. Egeland, "Exponential stabilization of nonholonomic chained systems," *IEEE Trans. on Automatic Control*, vol. 40, no. 1, pp. 35–49, 1995.
[27] E. Panteley and A. Lorýa, "On global uniform asymptotic stability of nonlinear time-varying systems in cascade," *Systems and Control Letters*, vol. 33, no. 2, pp. 131–138, 1998.
[28] M. Krstić, I. Kanellakopoulos, and P. Kokotović, *Nonlinear and Adaptive Control Design*. Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control, John Wiley and Sons, 1995.
[29] M. Schwaiger, T. Thümmel, and H. Ulbrich, "A 2D-motion platform: The Cybercarpet," in *2nd Joint EuroHaptics Conf. and Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, (Washington, DC), pp. 415–420, 2007.
[30] M. Schwaiger, *Konstruktion und Entwicklung omnidirektionaler Laufplattformen*. PhD thesis, Technische Universität München, Munich, Germany, 2008 *(in German)*.
[31] K. Nummiaro, E. Koller-Meier, and L. V. Gool, "An adaptive color-based particle filter," *Image Vision Computing*, vol. 21, no. 1, pp. 99–110, 2003.
[32] F. Aherne, N. Thacker, and P. Rockett, "The Bhattacharyya metric as an absolute similarity measure for frequency coded data," *Kybernetika*, vol. 32, no. 4, pp. 1–7, 1997.