FP6-511092

The CyberCarpet - Enabling Omni-directional Walking in Virtual Worlds

Instrument: STREP
Thematic Priority: IST

Deliverable T5.2/D1

Control Design

Due date of deliverable: 30.09.07
Actual submission date: 20.10.07

Start date of project: 01.04.2005 Duration: 36 month

Organisation name of lead contractor for this deliverable: UOR

Revision: 1.0

PROJECT CO-FUNDED BY THE EUROPEAN COMMISSION WITHIN THE 6. FRAMEWORK PROGRAMME (2002–2006)		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	X
CO	Confidential, only for members of the consortium (including the Commission Services)	

Contents page

This document reports about the work performed within Task 5.2 “Control Design” for the ball-array and the linear designs of the *CyberCarpet,* represented in Fig. 2 and Fig. 21, respectively. In particular, the control problem for the ball-array kinematics is considered in Part I of the Control design section, while the linear concept of the platform is examined in Part II. Conclusions are drawn in a final section and references are included.

Indeed, most of this report appears dedicated to the ball-array kinematics, since it presents a larger number of control issues (e.g., the presence of singularities that prevent the direct application of standard decoupling and linearizing schemes, whereas the linear kinematics is already linear and decoupled). Furthermore, many results devised for the ball-array design can be almost directly specialised to the simpler linear case.

The general control problem, including objectives and constraints, and the overall control architecture, are the same for the two platform designs, and are described in the introducing section, while we refer to deliverable T5.1/D1 for details on kinematic modeling and mobility analysis, and to preliminary deliverable T5.3/D1@M24 for the first results of control implementation on the small-scale prototype platform.

Table of contents

Control objectives and architecture

The main motion control objective for the *CyberCarpet* is to keep the walker absolute position within a sufficiently small distance from the center of the platform, despite of the voluntary and unpredictable locomotion of the user. The evaluation of a safe distance from the platform boundaries should be based on the maximum admissible velocity assumed for the walker locomotion and on the actuator capabilities. Although the knowledge of the walker orientation is needed for the correct display of the virtual environment to the user, this information is not relevant for the stated control task. Moreover, in order to achieve a natural and comfortable operation, the linear (and angular) velocities and accelerations (as well as inertial forces/moments) felt by the walker should be kept limited. These limits will reflect into state-dependent upper bounds on the input commands to the platform. Here, these constraints are not explicitly taken into account. However, by tuning the parameters in the control laws it is always possible to comply with physiological constraints in an easy way. Indeed, by lowering the control gains also the transient time needed for recovering the user at the center of the platform will be longer.

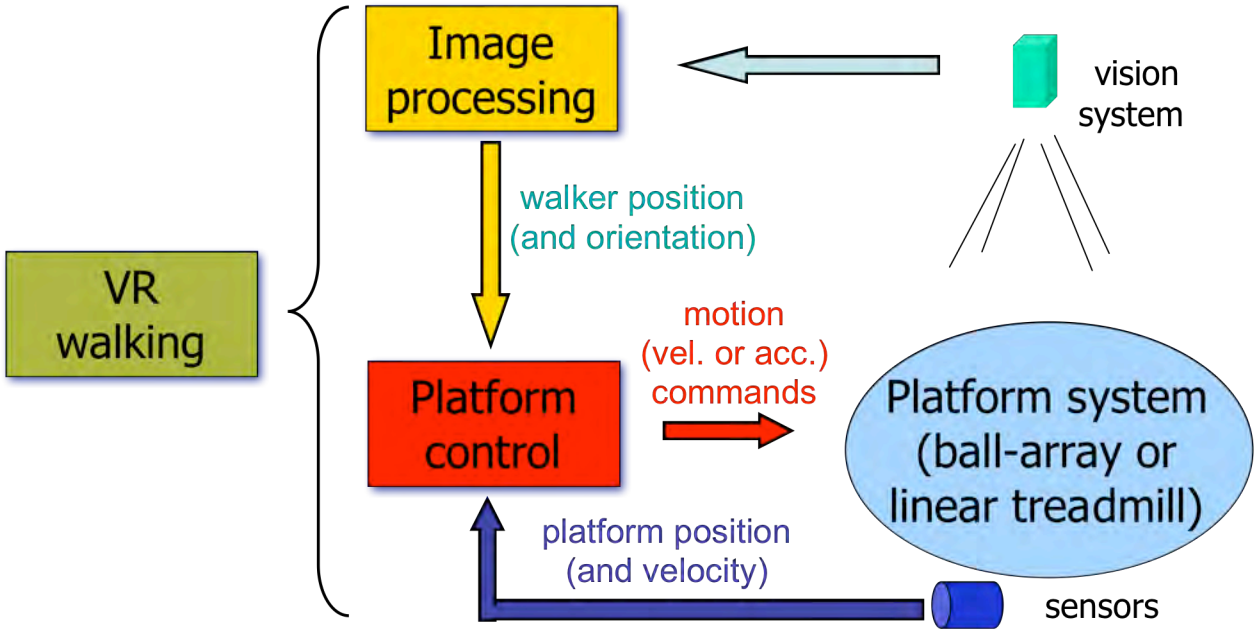

Fig. 1: Control system architecture

The overall control system architecture is shown in Fig. 1. The high-level visual tracker provides the absolute location of the walker on the platform. These measurements, together with the platform state (the angular orientation of the turntable —for the ball-array platform—, and also the platform velocities in the case of acceleration control), are available for the low-level platform motion control. This control output drives the two actuators of the platform. In principle, either kinematic or direct torque controllers can be used: in the first case, which is considered here and is the most common in conventional servo-drives, velocity or acceleration commands are taken as reference for direct-level PIDs.

Control design

Part I —The ball-array platform

This part of the report describes the control design approach pursued for the ball-array design of the *CyberCarpet,* represented in Fig. 2.

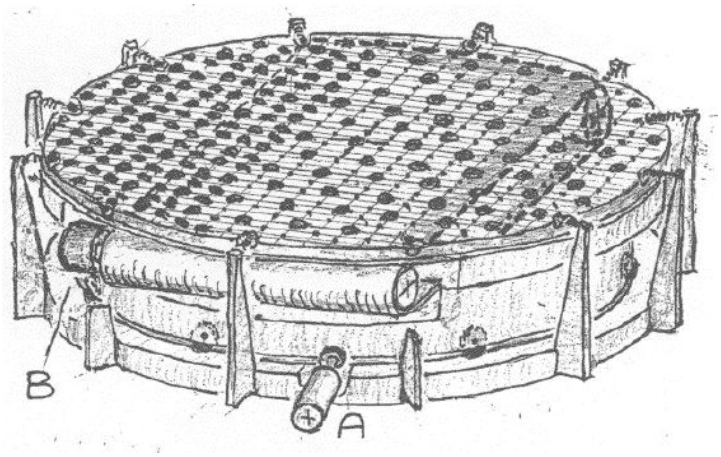

Fig. 2: The ball-array concept for the *CyberCarpet*

For this platform, having a linear and an angular actuating device, the control problem is similar to that of output regulation for nonholonomic wheeled mobile robots in the presence of an unpredictable disturbance due to walker's locomotion. Based on the first-order kinematic model (see Sect. I.1), a velocity control design achieving input-output decoupling and linearization is first proposed in Sect. I.2. In order to address the singularity issues related to this law, two simple but global kinematic control schemes are then presented in Sect. I.4. The feedback stabilizing part, which is based only on the user's pose information, is complemented by a feedforward term derived from a walker's velocity observer (see Sect. I.5). Numerical and graphical simulation results are reported in Sect. I.6. In Sect. I.7, the kinematic model is extended to the second order (acceleration commands are used as control inputs) and, based on this model, two different approaches are proposed in Sects. I.8.1 and I.8.2 for moving the velocity-level control laws of Sects. I.4-I.5 to the acceleration level. Simulation results, including the dynamic effects experienced by the walker (see Sect. I.9 and deliverable T5.1/D1), are reported in Sect. I.10.

I.1 First-order kinematic model

Given the ball-array surface of the *CyberCarpet*, any actuated motion of the belt/turntable will result in a reverse motion imposed to the walker standing on top of the ball array, i.e., a forward motion command will move the user backwards, and a clockwise rotation will turn the user counter-clockwise. Keeping this in mind, a first-order kinematic model of the *CyberCarpet* can be derived with the help of Fig. 3. Therein, (X_0, Y_0) is the absolute frame (also attached to the fixed overlooking camera) and (X_t, Y_t) is the frame rotated by an angle θ and attached to the treadmill, with the X_t-axis in the direction of the belt (along which linear motion is actuated). Both frames have the origin at the center of the *CyberCarpet*. The walker absolute position and orientation are (x, y) and θ_w, respectively, with R being his/her distance from the center. The angle $\alpha = \text{atan2}(y, x) - \theta$ locates the position of the walker in the frame (X_t, Y_t). The pair (R,α) represents the polar coordinates of the walker position and will be often used later in the control design. On the left side of Fig. 3 also the X_w axis (directed as the walker's sight) of the frame attached to the walker is displayed (the Z_w axis is directed as Z_0 and Y_w is defined accordingly). This frame is used to describe the kinematic and dynamic effects on the user due to the platform motion (see Sect. I.6 and deliverable T5.1/D1).

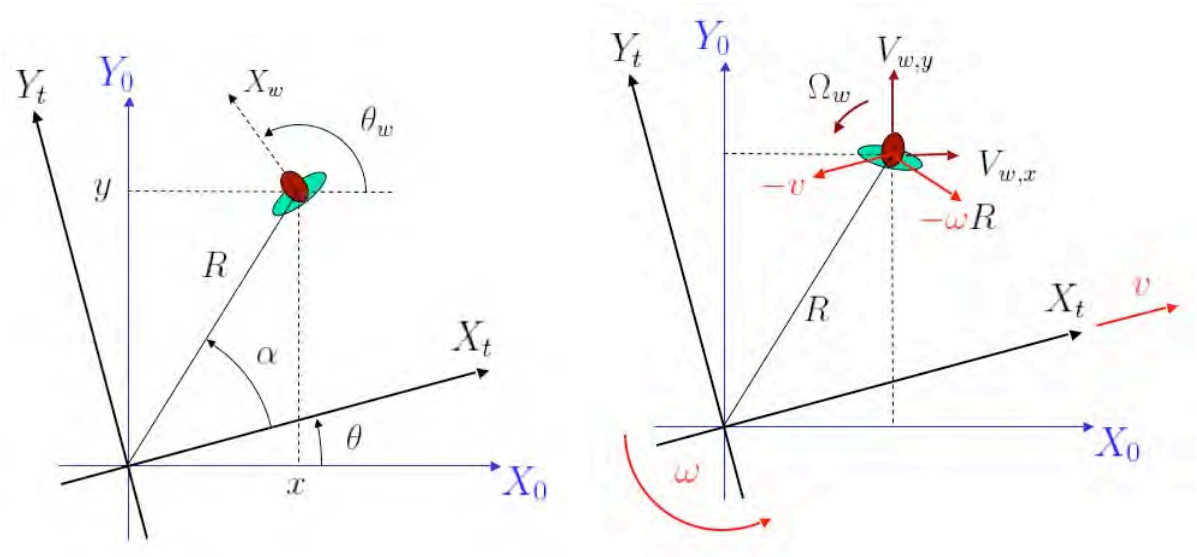

Fig. 3: Frames and variables definition: walker and platform still (left) and in motion (right)

When the walker is in motion the kinematic model is (see deliverable T5.1/D1 for details)

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

where v and ω are the linear and angular commands of the belt and turntable, respectively (under the ball-array). The absolute linear and angular walker velocities are denoted as $V_w = (V_{w,x}, V_{w,y})$ and Ω_w, respectively (see right of Fig. 3). These walker's velocities in eqs. (1) will be treated as "disturbances" acting on the control system and will be always assumed to be not directly measurable.

I.2 Velocity-level control design: Input-output feedback linearization

A number of feedback control laws developed for nonholonomic wheeled mobile robots can be modified to address the regulation problem for the *CyberCarpet*, so as to bring the position (and orientation) of a standing user to zero by suitable maneuvers. However, since only the walker's position has to be asymptotically stabilized to the origin (actually, to an arbitrarily small circle around the origin), we present here a simpler design based on input-output feedback linearization. The control law will use only the instantaneous sensor information on the walker position (the system output to be regulated).

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 $z = [x\ y\ ]^T$. Differentiating z in time and using eq. (1) gives

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

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

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = A^{-1}(x,y,\theta) \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \tag{2}$$

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

$$\dot{z}_1 = \dot{x} = v_1, \qquad \dot{z}_2 = \dot{y} = v_2,$$

i.e., it is decoupled and linearized by the feedback law (2). The control design can be completed by the proportional laws

$$v_1 = - k_1 x , \quad v_2 = - k_2 y , \tag{3}$$

with positive gains k_i (i =1, 2), thus exponentially stabilizing the walker's position to the origin. The above derivations hold outside the singularities of matrix A, i.e., whenever $x \cos\theta + y \sin\theta = R \cos\alpha \neq 0$ (see Fig. 3). For the purpose of analysis, a more convenient expression for v and ω can be found by choosing the gains $k_1 = k_2 = k >0$ and replacing eq. (3) into (2). This yields

$$v = k \frac{x^2 + y^2}{x\cos\theta + y\sin\theta} = \frac{kR^2}{R\cos\alpha} = \frac{kR}{\cos\alpha}, \tag{4}$$

and

$$\omega = k \frac{y\cos\theta - x\sin\theta}{x\cos\theta + y\sin\theta} = k \frac{R\cos\left(\frac{\pi}{2} - \alpha\right)}{R\cos\alpha} = k\tan\alpha . \tag{5}$$

A limit study for R going to zero and $\cos\alpha \neq 0$ shows that v and ω are continuous, with v vanishing and ω remaining bounded. Moreover, the control singularity at $\alpha = \pm \pi/2$ is relevant only at the initial instant if the walker is located on the Y_t-axis (see Fig. 3), in which case it may be also managed by a simple heuristics applied within a small zone including the Y_t-axis. For all other initial conditions, the angular control law (5) will automatically drive the walker away from this singularity. However, this problem cannot be easily managed when the walker moves, and for this reason we will more carefully deal with the issue of control singularities in Sect. I.4. Another interesting property of the designed controller arises from the linear and decoupled behavior of the closed-loop system. With the walker standing in an initial position (x_0, y_0), the time evolution of his/her position will be forced by eq. (3) to be

$$x(t) = e^{-k_1 t} x_0 \text{ and } y(t) = e^{-k_2 t} y_0 .$$

With $k_1 = k_2$, it will be

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

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

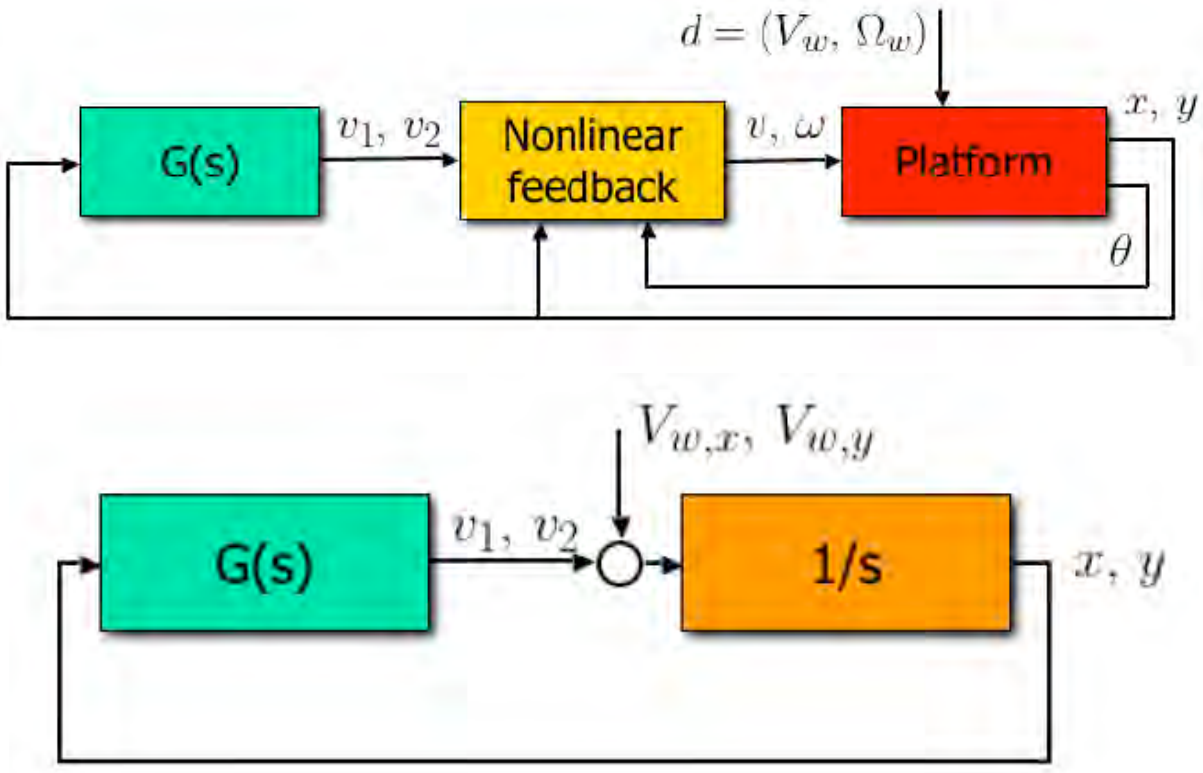

Fig. 4 The nonlinear control scheme (top) and its equivalent input-output linear representation (bottom)

The block diagram of the overall nonlinear control scheme is shown in Fig. 4, with G(s) = - diag (k, k) being a constant (instantaneous) block.

The disturbance signal d represents the walker's motion, i.e., when any or both of V_w and Ω_w are different from zero in eq. (1). A persistent locomotion will in general prevent the convergence of the walker position to the platform center when using the control law (2-3). In fact, the closed-loop input-output equations become in this case

$$\dot{x} = - k\, x + V_{w,x}, \qquad \dot{y} = - k\, y + V_{w,y}.$$

If the user walks indefinitely along a straight line with constant velocity $\overline{V}$, he/she will reach a steady-state position at a distance $\overline{R} = \overline{V} / k$ from the origin. In this case, from standard linear control analysis, the steady-state error can be completely eliminated by adding an integral action in the control loop before the disturbance entry point, i.e., by replacing eq. (3) with

$$v_1 = -k\left(x + a\int x\, dt\right), \qquad v_2 = -k\left(y + a\int y\, dt\right), \tag{6}$$

for suitable $k > 0$ and $a > 0$. These proportional-integral (PI) gains can be chosen so as to assign desired closed-loop poles (two for each input-output channel) in the left-hand side of the complex plane. With reference to the equivalent linear scheme in Fig. 4, this PI action is realized by setting each diagonal component of G(s) equal to - $k(s+a) / s$. The obtained *astatic* behavior copes with an unconstrained infinite walking of the user along a straight line. However, an overshooting of the controlled output is typically associated to the presence of the integral control term. For more general motion patterns, the control law given by eqs. (2) and (6) may not lead to full recovery of the walker position to the origin. Nonetheless, in the absence of an estimate of the walker velocity, not much more can be done (some videos, developed using Simulink and Visual Nastran, can be found at the website http://www.dis.uniroma1.it/labrob/research/CW.html).

I.3 Simulation results

In order to allow a comparison with the performance of the feedback/feedforward schemes presented in Sects. I.4-I.6, we report here some simulation results obtained with the feedback laws (2-3) and (2,6) for the square path[1] with 3 m sides shown in Fig. 5. The walker starts at rest from the 'Init' absolute position (1, 0) and moves along each edge with a trapezoidal velocity profile, having symmetric acceleration/deceleration phases at 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 corner, the walker stops and turns ccw with an angular speed of $\pi/2$ rad/s. Therefore, the total trajectory lasts 16 s. Note that, without motion control of the platform, the walker would exit from the boundary of the circular platform of radius 2.5 m.

The absolute motion of the walker under the platform control law (2-3), with $k_1 = k_2 = 2.5$, and the corresponding commands (v, ω) are shown in Fig. 6 and Fig. 7, respectively. The absolute orientation of the walker is displayed by a segment, while different patterns/colours (in the order, blue, red, green, and violet) are used for the virtual motion along each side of the square. As expected, each time the walker stops at a corner to perform a turn, he/she is pulled towards the origin along a straight line.

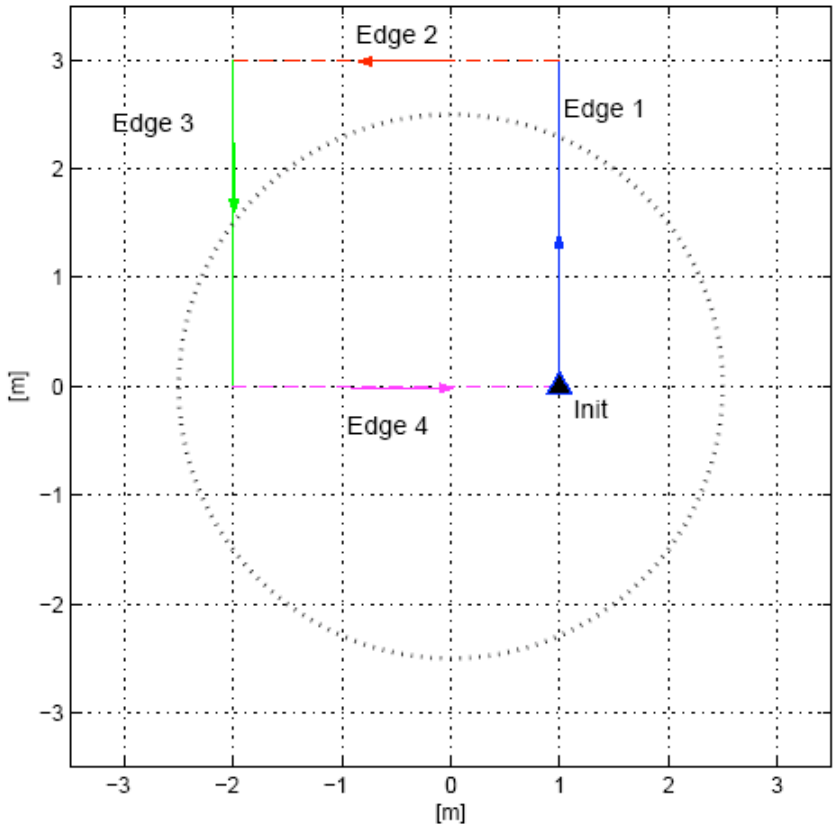

Fig. 5: Walker virtual locomotion: A square path executed counter-clockwise starting from the Init point (the dotted circle represents the platform boundary)

[1] The walker's path and time trajectory have been chosen so as to avoid control singularities for the laws (2-3) and (2,6).

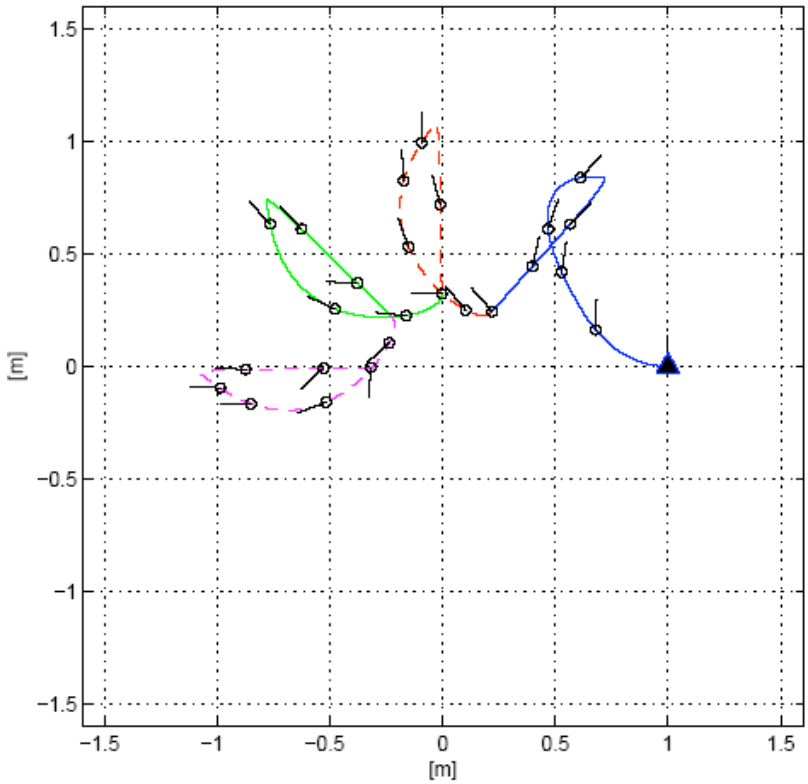

Fig. 6: Walker absolute locomotion under the platform controller (2-3): at start, axis X_t coincides with the horizontal axis X_0

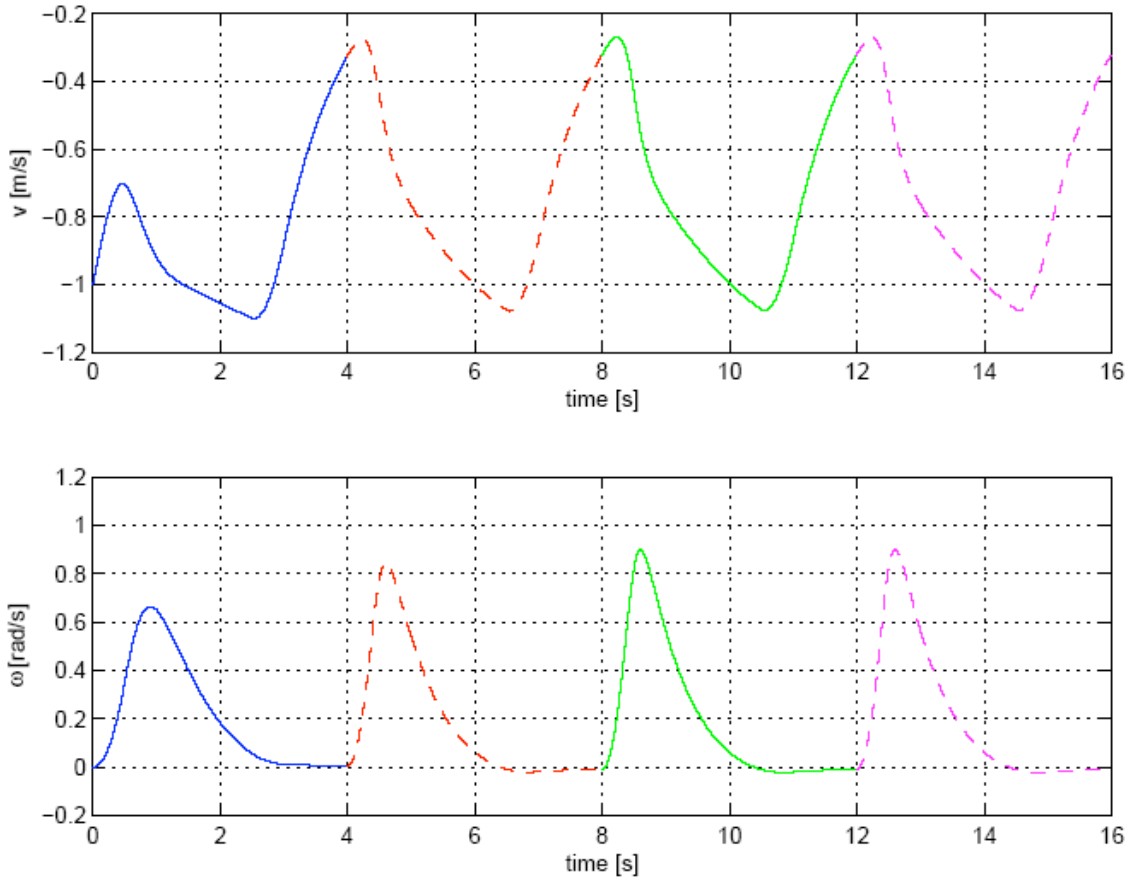

Fig. 7: Linear and angular velocity commands with controller (2-3)

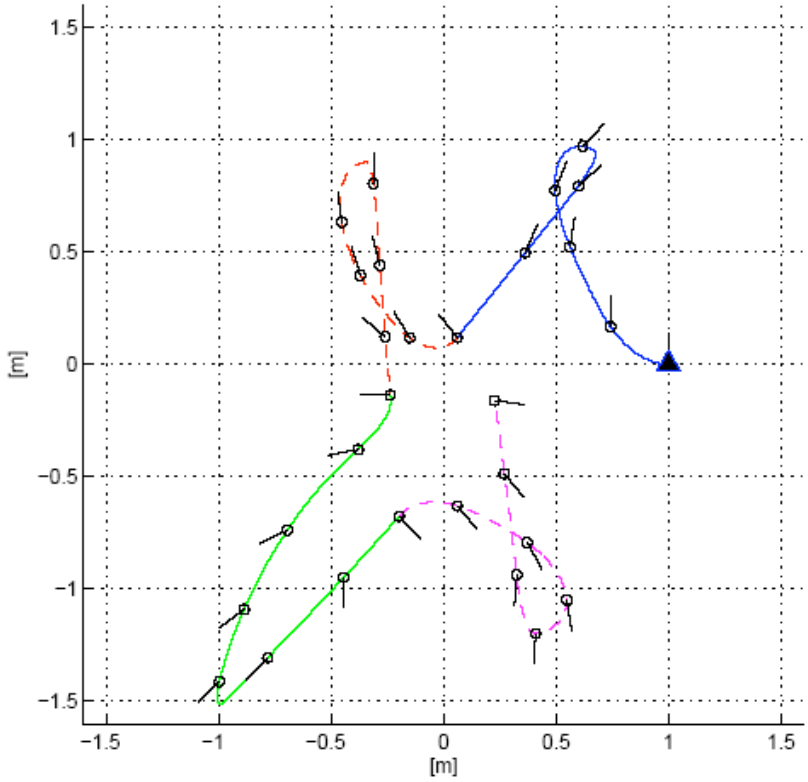

Fig. 8: Walker absolute locomotion under the platform controller (2, 6): at start, axis X_t coincides with the horizontal axis X_0

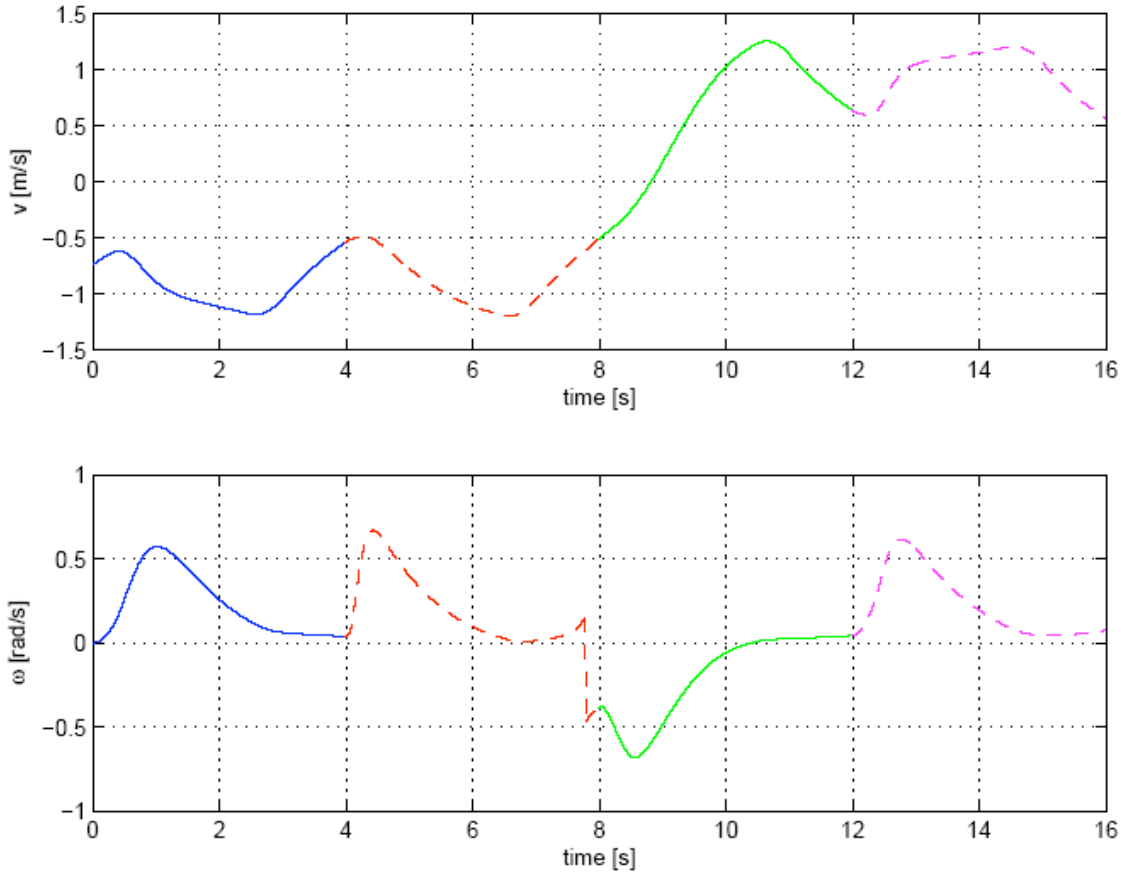

Fig. 9: Linear and angular velocity commands with controller (2, 6)

I.4 Velocity-level control design: Handling of singularities

The Cartesian expression of the decoupling control law (2-3) is singular when $x\cos\theta + y\sin\theta = R\cos\alpha = 0$ (where the determinant of the decoupling matrix is zero), i.e., either when the walker stands on the Y_t-axis or is at the origin. On the other hand, the expression of the control law in polar coordinates given in eqs. (4-5) clearly shows the control singularity at $\alpha = \pm\pi/2$, while the singularity at the origin of the (x,y) space is hidden by the fact that the angle α is not defined there. However, a step of $\pm\pi$ rad in the value of α is experienced when crossing the origin, which makes the control input ω suddenly change sign. As a consequence, even measurement noise on the x and y variables can cause an undesired chattering in the value of ω when the walker approaches the platform center.

In the following we shall introduce modifications that deal with these control singularities, while trying to preserve some convenient characteristics of the decoupling law (4-5). We consider first the case of absence of disturbances, $V_w = 0$ and $\Omega_w = 0$ (walker standing still in the virtual environment), dealing later with the motion of the walker (Sect. I.5).

I.4.1 Singularity at $\cos\alpha = 0$

The singularity at $\alpha = \pm\pi/2$ can be eliminated by simply taking

$$v = k\,R\,\mathrm{sgn}(\cos\alpha), \tag{7}$$

and

$$\omega = k\sin\theta\,\mathrm{sgn}(\cos\alpha), \tag{8}$$

with $\mathrm{sgn}\,x = 1$ for $x \neq 0$ and $\mathrm{sgn}\,x = -1$ otherwise. The control law (7-8) is formally obtained by multiplying eqs. (4-5) by $|\cos\alpha|$. The resulting closed-loop input-output dynamics is now

$$\dot{x} = -k\,|\cos\alpha|\,x, \qquad \dot{y} = -k\,|\cos\alpha|\,y, \tag{9}$$

that is not anymore linear nor decoupled, since the angle θ depends on both x and y. However, note that from (9) it follows

$$\frac{y(t)}{x(t)} = \frac{\dot{y}(t)}{\dot{x}(t)} = \frac{y_0}{x_0}, \tag{10}$$

just as with the decoupling and linearizing law (4-5). Therefore, a user standing still will be driven along the straight line connecting its initial position to the origin.

In order to show that the control law (7-8) asymptotically stabilizes the walker position (x, y) to the origin, consider the positive definite Lyapunov function

$$\tilde{V}(x,y,\alpha) = \frac{1}{2}(x^2 + y^2 + \sin^2\alpha),$$

for which it is $\tilde{V} = 0$ if and only if

$$(x,y,\alpha) \in S = \left\{\left(0,0,\bar{\alpha}\right),\ \sin\bar{\alpha} = 0\right\}.$$

Taking into account eqs. (1), the derivative of $\tilde{V}$ along the system trajectories is given by

$$\dot{\tilde{V}} = x\dot{x} + y\dot{y} + \sin\alpha\cos\alpha\,\dot{\alpha} = -k\left|\cos\alpha\right|\left(x^2 + y^2 + \sin^2\alpha\right) \leq 0$$

so that $\dot{\tilde{V}} = 0$ for $(x, y, \alpha) \in S$ as well as for $\cos\alpha = 0$. However, the latter does not correspond to closed-loop system equilibria, since $\omega = \pm k$ at these configurations. Therefore, S is the largest invariant set for the system, and (x,y) converges to the origin by virtue of LaSalle's theorem.

I.4.2 Singularity at $R = 0$

The control law (7-8) is clearly not continuous at the origin, due to the discontinuity of the angle α which is not defined at $x = y = 0$. 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 propose next two different strategies.

I.4.2.1 Dead zone around the origin

A simple way to overcome input chattering when the walker approaches the platform center is to introduce a small dead zone for the input commands v and ω around the origin, i.e., setting $v = \omega = 0$ for $R < R_d$. At the cost of a small (namely, of amplitude R_d) error in the final configuration, which is consistent with the task specification, this strategy is sufficient to solve the problem in the absence of walker's motion: the platform stops as soon as the walker is brought inside the dead zone. However, when the walker moves, the chattering of the input commands may appear again at the border of the dead zone. In order to at least reduce the frequency of this chattering, some *hysteresis* can be introduced at the border of the dead zone by defining two different thresholds, $R_{d,\text{off}}$ and $R_{d,\text{on}}$, for triggering the switch-off and switch-on of the control law (7-8). This heuristic approach usually shows a satisfactory performance both in terms of input commands behavior and of walker's executed path (see the simulation results in Sect. I.6).

I.4.2.2 A smooth nonsingular control law

A more elegant solution to the chattering problem is obtained by replacing the feedback control law (7-8) with

$$v = k\,R^2\,\text{sgn}(\cos\alpha) = k\,(x^2 + y^2)\,\text{sgn}(x\cos\theta + y\sin\theta), \tag{11}$$

and

$$\omega = k\,R\sin\alpha\,\text{sgn}(\cos\alpha) = k\,(y\cos\theta - x\sin\theta)\,\text{sgn}(x\cos\theta + y\sin\theta), \tag{12}$$

which are formally obtained by multiplying eqs. (7-8) by the radial distance R. For the sake of clarity, both the expressions in polar coordinates and in Cartesian coordinates are given. The control law (11-12) is defined at any system configuration. Furthermore, the angular velocity command ω converges to zero as (x,y) approaches the origin, so that no chattering problem exists in this case. Under the feedback law (11-12), the closed-loop behavior becomes

$$\dot{x} = -k\,R\,|\cos\alpha|\,x, \qquad \dot{y} = -k\,R\,|\cos\alpha|\,y, \tag{13}$$

that can be proved to be asymptotically stable at the origin by the same Lyapunov arguments used for the control law (7-8). Furthermore, note that eq. (10) still holds for the solutions of (13), i.e., the user is pulled toward the origin along the connecting straight line.

Finally, in comparison with the discontinuous control modification of Sect. I.4.2.1, the smooth and nonsingular feedback law (11-12) does not exhibit any steady-state error. However, the convergence rate of x and y drops quadratically to zero as the walker approaches the origin. In view of our main control objective, this should not be seen as a negative feature.

I.5 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. (1). A persistent walker locomotion will in general prevent the convergence of her/his position to the platform center when using the control laws (7-8) or (11-12). For example, it can be shown that,

---

[2] Although a term $\text{sgn}(\cos\alpha)$ 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.

when the user walks indefinitely in the virtual environment along a straight line with constant velocity $\overline{V}$, a steady-state position at a distance $\overline{R} = \overline{V}/k$ from the origin will be reached, just as in the case of the decoupling and linearizing law (2-3). In Sect. I.2, using standard results from linear control theory, we suggested to add in eq. (6) an integral action in order to completely eliminate this steady-state error. However, a poor dynamic performance results, due to the typical output overshooting associated to the presence of an integral control term. We propose here a different approach to deal with walker's locomotion. Based on an estimate $\hat{V}_w$ of the walker linear velocity V_w, we shall include a suitable feedforward term in eqs. (7-8) or in eqs. (11-12) as follows

$$\begin{aligned} v_c &= v + v_f = v + [\cos\theta \quad \sin\theta\,]\cdot \hat{V}_w\,, \\ \omega_c &= \omega + \omega_f = \omega + \mathrm{sat}(1/R\,[\,-\sin\theta \quad \cos\theta\,]\cdot \hat{V}_w\,). \end{aligned} \tag{14}$$

Here, sat(·) is the standard saturation function, with lower/upper saturation limits to be defined according to given (or desired) constraints on the input velocities. It is readily verified that, for $\hat{V}_w = V_w$, the feedforward term v_f in eqs. (14) compensates for the component of the walker velocity along the direction of the *Cyber-Carpet* linear motion, while ω_f (when the saturation is not active) cancels the component of V_w in the orthogonal direction[3].

In order to get an accurate estimate $\hat{V}_w$ of the walker velocity, consider the two scalar dynamic systems (these are velocity observers)

$$\begin{aligned} \dot{\xi}_x &= -v\cos\theta + y\,\omega + k_w\,(x - \xi_x) \\ \hat{V}_{w,x} &= k_w\,(x - \xi_x), \end{aligned} \tag{15}$$

and

$$\begin{aligned} \dot{\xi}_y &= -v\sin\theta - x\,\omega + k_w\,(y - \xi_y) \\ \hat{V}_{w,y} &= k_w\,(y - \xi_y), \end{aligned} \tag{16}$$

where $k_w > 0$. From eqs. (15-16) and (1), it follows

$$\begin{aligned} \dot{\hat{V}}_{w,x} &= -k_w \hat{V}_{w,x} + k_w V_{w,x}\,, \\ \dot{\hat{V}}_{w,y} &= -k_w \hat{V}_{w,y} + k_w V_{w,y}\,, \end{aligned}$$

i.e., the estimates are low-pass filtered versions of absolute Cartesian components of of the actual absolute walker velocity V_w. In particular, for k_w large enough, they accurately reproduce the two components $V_{w,x}$ and $V_{w,y}$. Note that, even after this feedforward compensation, the system is still affected by a residual disturbance

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

Therefore, a constant velocity of the walker in any direction is fully compensated at steady state for any positive k_w, while for walker's ramp-wise velocities (i.e., when moving with constant acceleration) the associated steady-state error can be made in principle arbitrarily small by increasing k_w — an *astatic* disturbance rejec-

[3] The saturation in ω_f is formally necessary to exclude a possible divergence when (x, y) approaches the origin. However, for V_w smooth enough, the platform tends to align with V_w so that ω_f is always close to zero at steady state.

tion behavior is recovered by using the feedback/feedforward control scheme (14-16). Note that the overall controller is no longer instantaneous, as it includes the dynamics of the walker's velocity observer.

I.6 Velocity-level control: Simulation results

We present some selected simulation results obtained with the motion control laws discussed in Sects. I.4 and I.5 (see deliverable T5.3/D1@M24 for experimental results on the small-scale platform). In all case studies, the walker starts at rest from the initial absolute position (0, 1) m —one that would immediately lead to control problems for the laws (2-3) and (2,6). Fig. 10 to Fig. 13 refer to the walker moving indefinitely along a straight line directed along the Y_t-axis in the virtual space (i.e., for a fixed platform), with a constant velocity of 1 m/s. Initially, we have $\theta = 0$ (X_t is aligned with X_0) and $\theta_w = \pi/2$.

In Fig. 10, the actual motion of the walker in the absolute space is shown. The absolute orientation θ_w of the walker is displayed by a segment. The platform is controlled here by the combined feedback/feedforward scheme (14), wherein the feedback law (7-8) is used together with a dead zone/hysteresis as described in Sect. I.4.2.1. The control parameters are: $k = 1$, $k_w = 10$, $R_{d,off} = 0.05$ m, $R_{d,on} = 0.1$ m, and $|\omega_f| \leq 2$ rad/s. We note that there is a small error at steady state, just within the distance $R_{d,off}$, where the input commands are simply $v_C = v_f$ and $\omega_C = \omega_f$ (note that the dead zone applies only to the feedback terms). At the end, the platform is oriented in the same direction of the walker's motion, at an angle $\theta = \theta_w = \pi/4$. The linear and angular velocity commands shown in Fig. 11 confirm that the feedforward terms in the control law converge to the actual velocity of the walker ($v_f = 1$ m/s and $\omega_f = 0$), thus achieving an exact motion compensation.

The same task is performed again with the feedback/feedforward scheme (14), but using next the smooth feedback law (11-12) and the same relevant control parameters as above. A zero final position error is obtained (Fig. 12). The control behavior shown in Fig. 13 clearly indicates the absence of discontinuities, but transients of the input commands v and ω are now somewhat longer.

As a more complex motion, we report the results for the virtual square path with 3 m sides shown in Fig. 14 (the same as in Fig. 5 but with a different starting point). We report only the results for the smooth feedback law (11-12) used within the scheme (14), see Fig. 15. The control parameters are chosen as before. Thanks to the combined feedback and feedforward actions, the walker is rapidly brought close to the platform center and then kept there (compare the absolute walker motion of Fig. 15 with that of Fig. 6 and Fig. 8). The linear control input is smooth (see Fig. 16) and, after an initial transient, never exceeds the walker's voluntary speed. On the other hand, the saturation on the angular feedforward term ($|\omega_f| \leq 2$ rad/s) becomes relevant when the walker is close to the origin and takes a sharp turn (starting with the corner after the red edge 2). Note that the platform lags behind any turn performed on place by the walker, since there is no feedforward action triggered by a walker's angular motion without linear displacement. Finally, it can be recognized that a periodic behavior is eventually reached, starting with the green edge 3 of the square path.

In order to fully appreciate the overall motion of the platform/walker system, a 3D graphical simulation environment has also been developed, using Simulink and Visual Nastran. Videos of the presented motion tasks are available at http://www.dis.uniroma1.it/~labrob/research/CW.html.

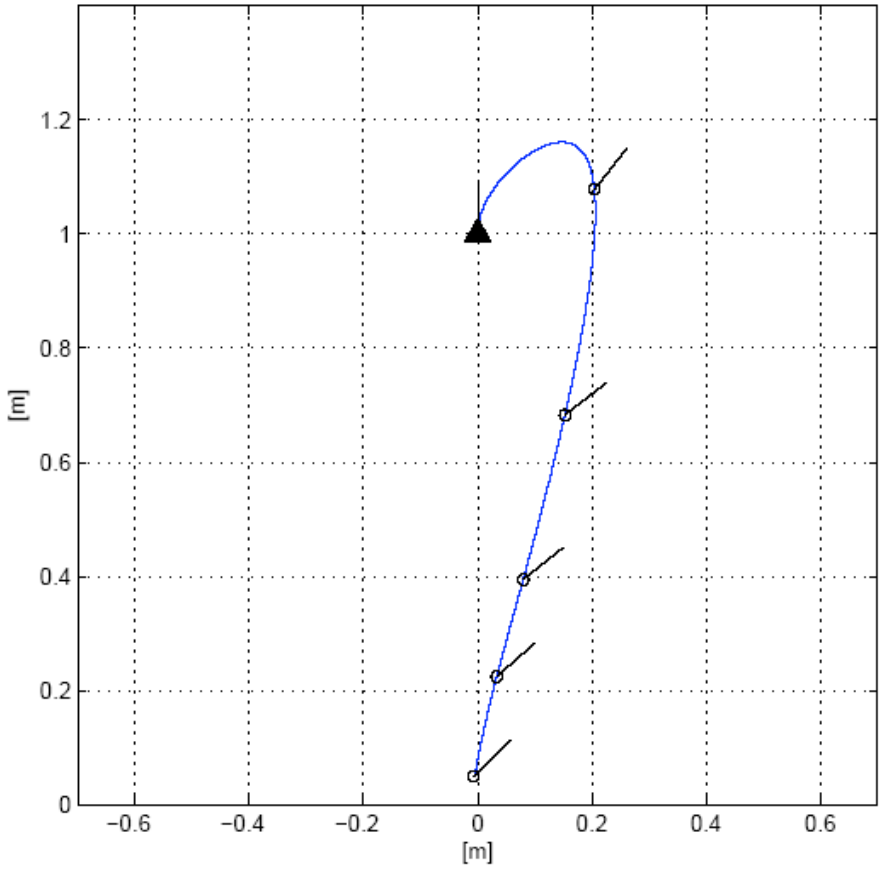

Fig. 10: Virtual straight line: Walker absolute locomotion under the platform controller (14), using the feedback law (7-8) and a dead zone with hysteresis around the origin

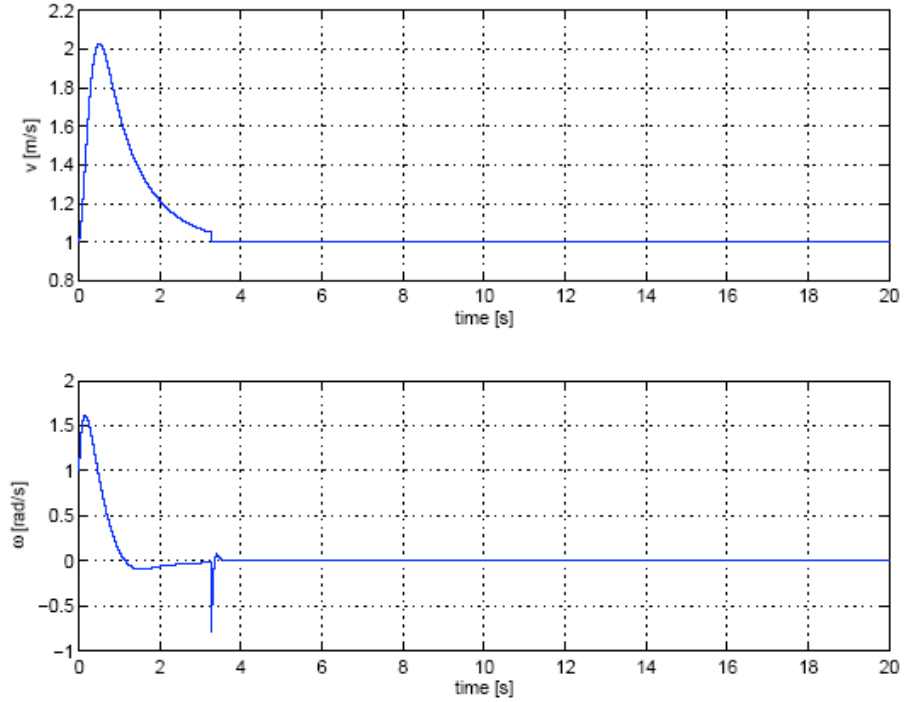

Fig. 11: Linear and angular velocity commands for Fig. 10

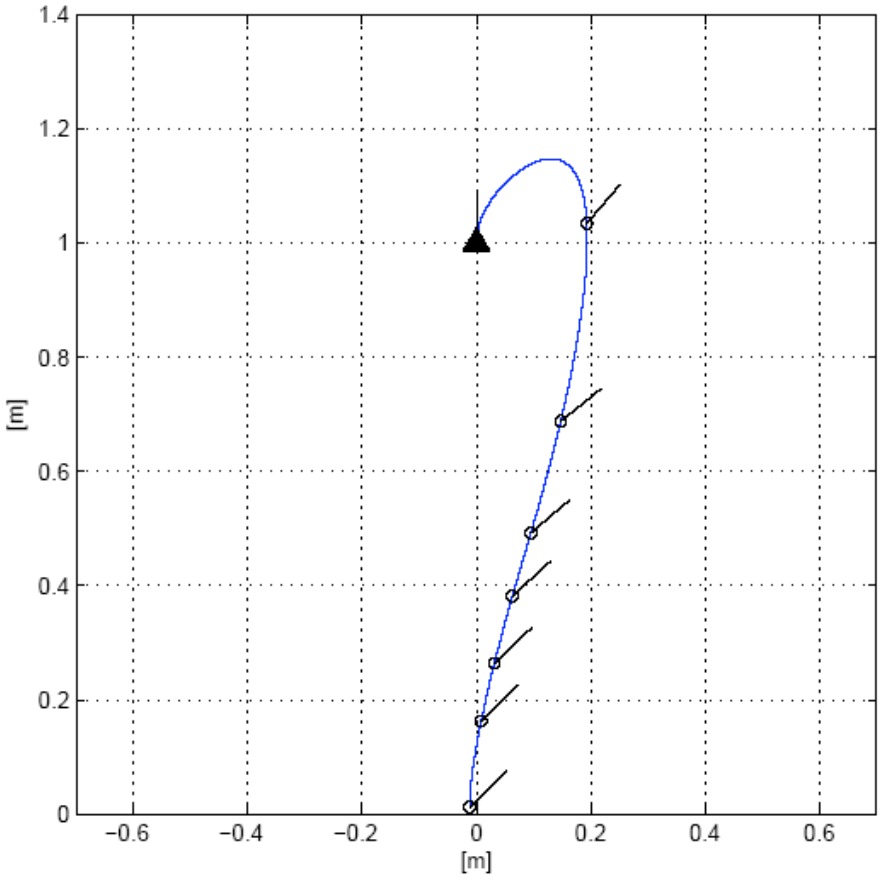

Fig. 12: Virtual straight line: Walker absolute locomotion under the platform controller (14), using the smooth feedback law (11-12)

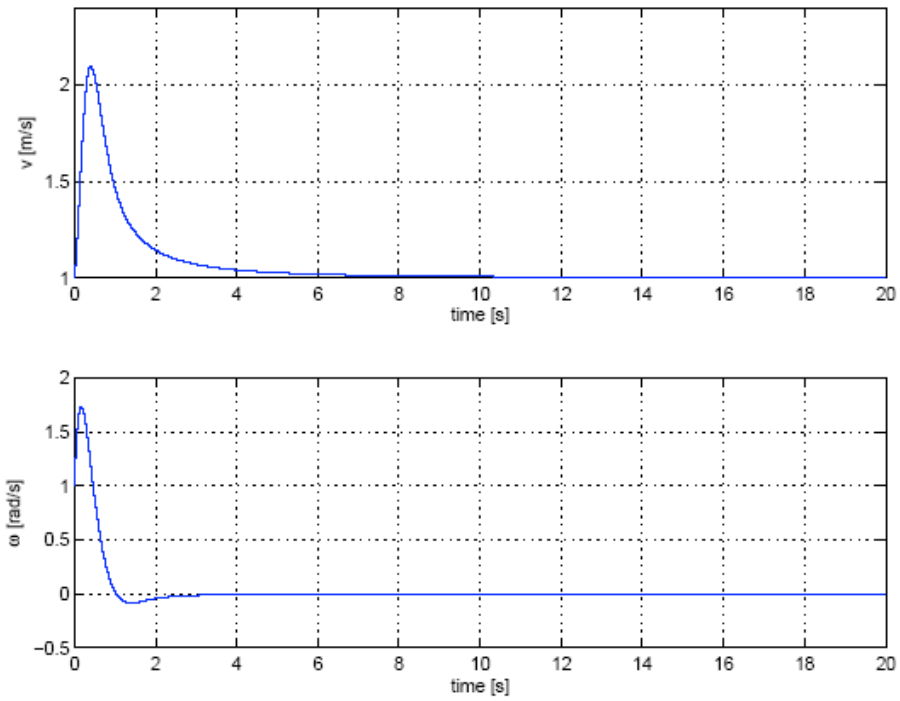

Fig. 13: Linear and angular velocity commands for Fig. 12

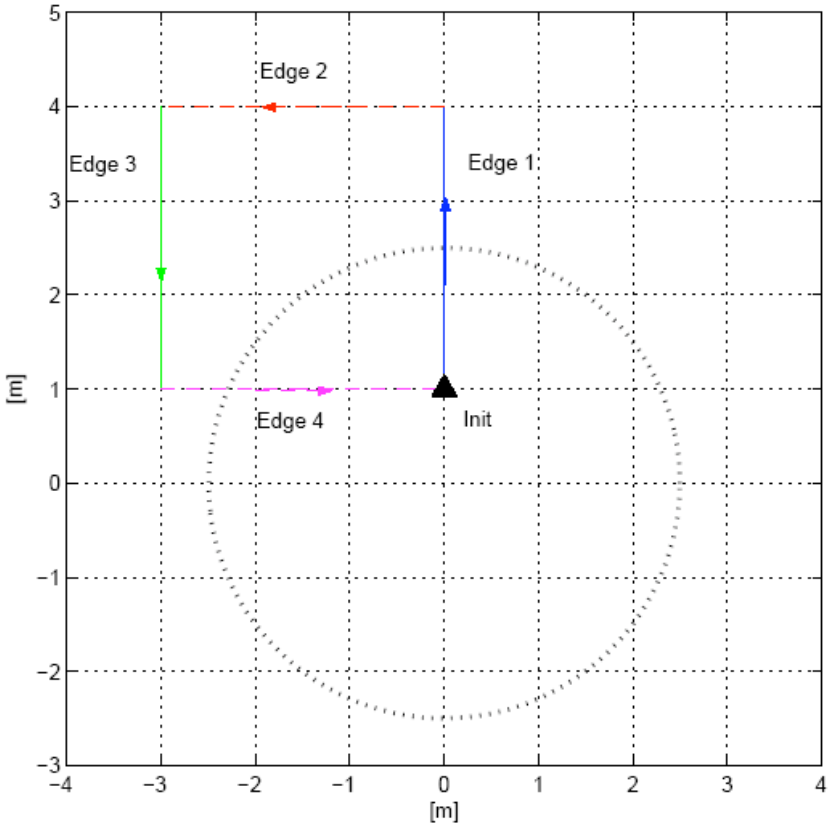

Fig. 14: Walker virtual locomotion: A square path executed counterclockwise starting from the Init point (the dotted circle represents the platform boundary)

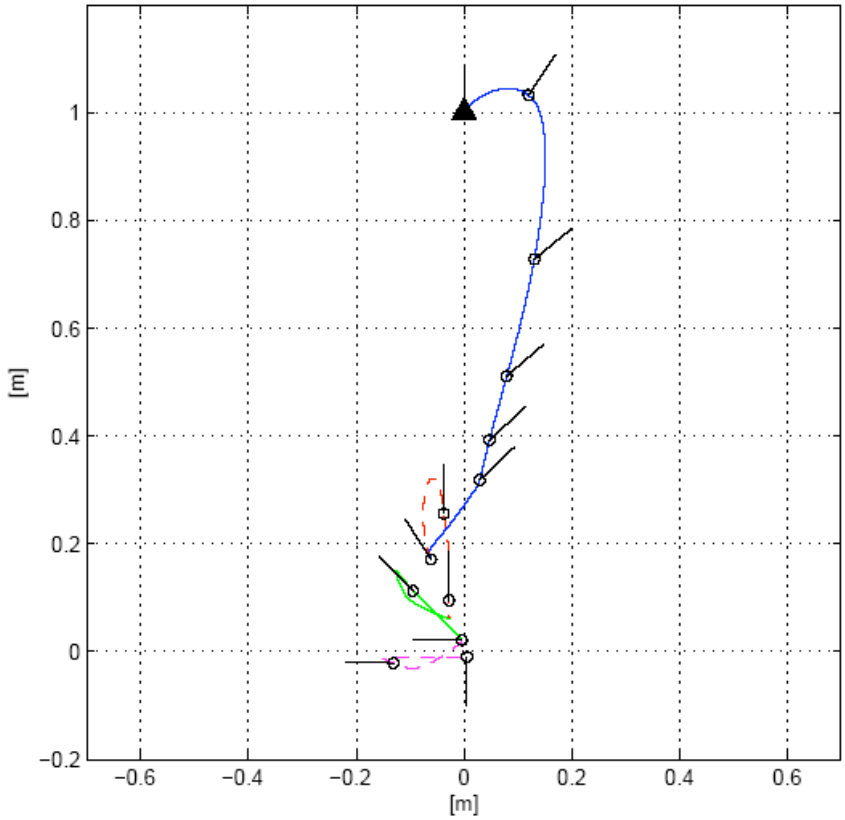

Fig. 15: Virtual square path: Walker absolute locomotion under the platform controller (14), using the smooth feedback law (7-8)

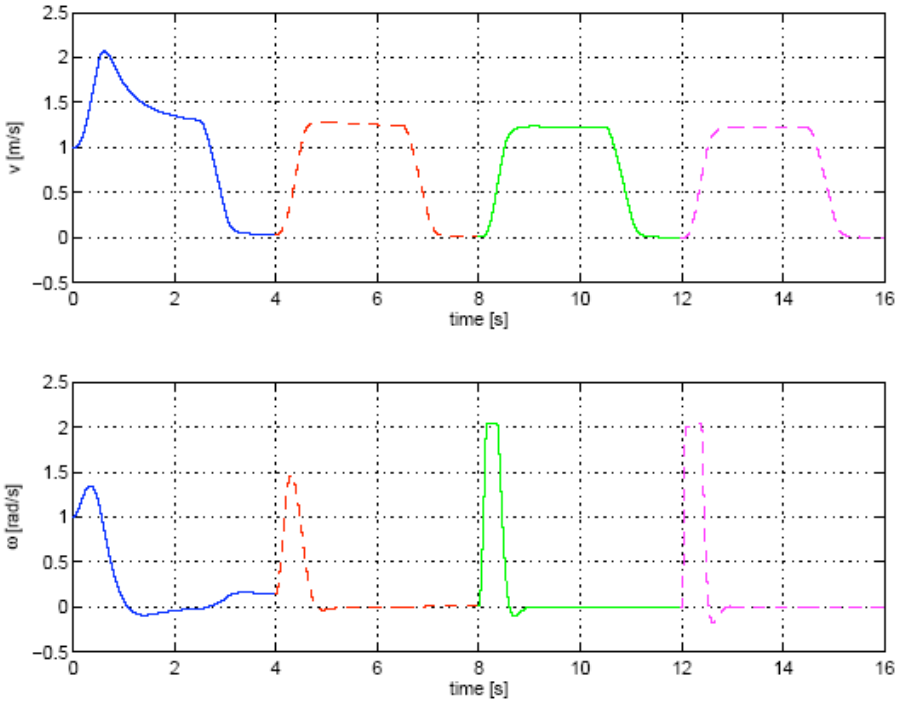

Fig. 16: Linear and angular velocity commands for Fig. 15

I.7 Second-order kinematic model

If we assume that the platform motion is commanded through the linear and angular acceleration inputs a and η, the first-order kinematic model (1) is extended as

$$\begin{aligned}
\dot{x} &= -v\cos\theta + y\omega + V_{w,x} \\
\dot{y} &= -v\sin\theta - x\omega + V_{w,y} \\
\dot{\theta} &= \omega \\
\dot{\theta}_w &= -\omega + \Omega_w \\
\dot{v} &= a \\
\dot{\omega} &= \eta
\end{aligned} \tag{17}$$

that corresponds to the addition of two integrators on the velocity inputs of model (1). The linear and angular velocities v and ω of the platform become now two further states of the system. Thus, in order for the whole system state to be still available, we must assume that both these linear and angular platform velocities are measurable. Note that the second-order kinematic model (17) is characterized by the same motion singularities of the first-order model (1).

I.8 Acceleration-level control

When a control law, with the same objectives stated in the introducing section of this report, has to be devised for the extended (second-order) system (17), the availability of some stabilizing laws $v = v_d(x,y,\theta)$ and $\omega = \omega_d(x,y,\theta)$ for the first-order system (1) can be exploited in different ways. We describe in the following two different approaches, one based on *backstepping* (see, e.g., [9]) and the other on the theory of *cascaded systems* (see. e.g., [10]).

We note preliminarily that both approaches apply independently from the particular platform kinematics ("ball-array" or "linear"). The specific platform will affect only the particular expression of the first-order

feedback stabilizing law v_d and ω_d on which the second-order control design relies. In Part II of this document we shall, however, address a more direct solution for the acceleration control of the linear platform.

I.8.1 Backstepping

This technique provides a constructive systematic method to devise globally stabilizing control laws for the general class of triangular nonlinear systems of the form

$$\begin{aligned}
\dot{\xi} &= f(\xi) + g(\xi)\xi_1 \\
\dot{\xi}_1 &= f_1(\xi,\xi_1) + g_1(\xi,\xi_1)\xi_2 \\
\dot{\xi}_2 &= f_2(\xi,\xi_1;\xi_2) + g_2(\xi,\xi_1;\xi_2)\xi_3 \\
&\vdots \\
\dot{\xi}_n &= f_n(\xi,\xi_1,\ldots,\xi_n) + g_n(\xi,\xi_1,\ldots,\xi_n)u,
\end{aligned} \tag{18}$$

where u is the actual control input to the system. This is obtained in the following way. First consider ξ_1 as a virtual input to stabilize the first subsystem with state ξ. Let ξ_1^d be the control law that solves this first subproblem and $V_1(\xi)$ a corresponding Lyapunov function (used to show asymptotic stability). Next, define z_1 to be the difference between ξ_1 and its desired value ξ_1^d, and consider ξ_2 as a virtual input to stabilize the $[\xi\, z_1]^T$ subsystem. In order to find a desired behavior for ξ_2 that stabilizes this second subsystem, we build a Lyapunov function candidate by simply augmenting $V_1(\xi)$ with a quadratic term in z_1, i.e.,

$$V_2(\xi,\xi_1) = V_1(\xi) + \frac{1}{2} z_1^T z_1, \qquad \text{with } z_1 = \xi_1 - \xi_1^d. \tag{19}$$

We choose then ξ_2 so as to make $\dot{V}_2 < 0$. Proceeding step by step along these lines, one finally arrives at defining a stabilizing law for the control u.

For the second-order system (17), in the absence of walker's intentional motion, we have $n = 1$ in (18) and

$$\xi = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \qquad \xi_1 = \begin{bmatrix} v \\ \omega \end{bmatrix}, \qquad u\,(=\xi_2) = \begin{bmatrix} a \\ \eta \end{bmatrix}, \tag{20}$$

$$f(\xi) = f_1(\xi,\xi_1) = 0, \quad g(\xi) = A(x,y,\theta), \qquad g_1(\xi,\xi_1) = I_2,$$

being I_2 the 2×2 identity matrix. In the first step of the backstepping procedure we may select $\xi_1^d = [v_d\ \omega_d]^T$ as the stabilizing control law (14), (11–12). Correspondingly, we can use the Lyapunov function $V_1(\xi)$ as in Sect. I.4. In the second (and last) step, we build the Lyapunov function V_2 as in eq. (19). Along the system trajectories it holds

$$\dot{V}_2 = \frac{\partial V_1}{\partial \xi} A(\xi)\xi_1 + \left(\xi_1 - \xi_1^d\right)^T \left(u - \dot{\xi}_1^d\right). \tag{21}$$

From eq. (21), it is clear that a control law making the overall system (17) asymptotically stable is

$$u = \dot{\xi}_1^d - K\left(\xi_1 - \xi_1^d\right) - A^T(\xi)\frac{\partial V_1}{\partial \xi}^T \tag{22}$$

for any positive definite, typically diagonal matrix K. Correspondingly, it is

$$\dot{V}_2 = \dot{V}_1^d(t) - \left(\xi_1 - \xi_1^d\right)^T K\left(\xi_1 - \xi_1^d\right)$$

where $\dot{V}_1^d(t)$ is the time derivative of the Lyapunov function $V_1(\xi)$ evaluated along the solution trajectories $\xi(t)$ of subsystem (1), when $\xi_1 = \xi_1^d$. Note that the control law (22) guarantees also the convergence of ξ_1 to ξ_1^d. However, the computation of $\dfrac{\partial V_1}{\partial \xi}$ is required for its implementation.

I.8.2 Cascaded approach

An alternative method for deriving a second-order feedback law based on the available first-order control laws of Sect. I.4 is to use a result on the stability of cascaded systems (see [10] for details). We can roughly summarize it by saying that, under certain conditions, the stability of the following autonomous system (obtained once the control input u has been chosen)

$$\begin{aligned} \dot{\zeta}_1 &= f_1(t,\zeta_1) + g_1(t,\zeta_1,\zeta_2)\zeta_2 \\ \dot{\zeta}_2 &= f_2(t,\zeta_2) \end{aligned} \tag{23}$$

can be concluded from the stability of the two subsystems $\dot{\zeta}_1 = f_1(t,\zeta_1)$ and $\dot{\zeta}_2 = f_2(t,\zeta_2)$. In order to be able to use this result for the stabilization of system (17), we have to put it in the form (23). To this purpose, using the same notation of Sect. I.8.1, it is sufficient to perform the change of coordinates

$$\zeta_1 = \xi = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \qquad \zeta_2 = \xi_1 - \xi_1^d,$$

so that the system equations become

$$\begin{aligned} \dot{\zeta}_1 &= A(\zeta_1)\xi_1^d(\zeta_1) + A(\zeta_1)\zeta_2, \\ \dot{\zeta}_2 &= u(t) - \frac{d\xi_1^d(t)}{dt}. \end{aligned} \tag{24}$$

From the results of Sect. I.4, we know already that the "downstream" system (described by the first set of eqs. (24)) is asymptotically stable for $\zeta_2 = 0$, i.e., for $\xi_1 = \xi_1^d$. In order to stabilize the overall cascaded system it is thus sufficient to stabilize the "upstream" system (described by the second set of eqs. (24)) to the origin, i.e., to bring ζ_2 to zero. This can be obtained by the simple control law

$$u = \dot{\xi}_1^d - K\zeta_2, \tag{25}$$

for any positive definite, typically diagonal matrix K. Compared with the backstepping design (22), the control law (25) is certainly simpler to compute, and therefore it has been selected for implementation.

The acceleration control law (25), as well as (22), requires the differentiability of the velocity control law ξ_1^d. When choosing this as in eqs. (11-12), we easily see that differentiability is missing at configurations where the argument of the sgn function is zero. This problem, however, can be simply solved by setting $\dot{\xi}_1^d = 0$ in eq. (25) at such configurations (i.e., by setting $d/dz(\operatorname{sgn} z) \equiv 0$, for all z). Note also that an analytical expression of $\dot{\xi}_1^d$ can be computed from eqs. (11-12) and the model (17), assuming that the walker velocity V_w can be treated as locally constant.

I.9 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 walker moves with constant velocity in the virtual world, she/he will feel "apparent" accelerations due to the rotation and/or not uniform translation of the carpet. These accelerations must be evaluated in order to verify that they do not exceed the limits of physiological comfort. Note that, having moved the control action to the acceleration level, these computations can be more reliably performed in analytical form, since platform acceleration commands are available and it is not necessary to resort to numerical differentiation. In particular, when the user walks at constant velocity ${}^{w}V_w$ in the non-inertial virtual

world[4], the total apparent acceleration being felt equals her/his absolute acceleration (computable by differentiation of the first two equations in (17)), changed in sign. This acceleration is usually decomposed into three different components depending, respectively, on the linear and angular accelerations of the reference frame (*inertial* acceleration), on the square of the frame angular velocity (*centrifugal* acceleration), and on the coupling between the frame angular velocity and the walker *relative* velocity (*Coriolis* acceleration). All these components should be expressed in the frame (X_w, Y_w, Z_w) attached to the walker, in order to evaluate the physiological effects on the user. The results of the described computational procedure are reported below (see also deliverable T5.1/D1).

- Inertial acceleration (due to the linear and angular accelerations of the reference frame):

$$\vec{a}_{in} = \dot{v}\begin{bmatrix}\cos(\theta-\theta_w)\\ \sin(\theta-\theta_w)\\ 0\end{bmatrix} + \dot{\omega}R\begin{bmatrix}-\sin(\theta+\alpha-\theta_w)\\ \cos(\theta+\alpha-\theta_w)\\ 0\end{bmatrix} = a\begin{bmatrix}\cos(\theta-\theta_w)\\ \sin(\theta-\theta_w)\\ 0\end{bmatrix} + \eta R\begin{bmatrix}-\sin(\theta+\alpha-\theta_w)\\ \cos(\theta+\alpha-\theta_w)\\ 0\end{bmatrix}$$
$$= Rot(-\theta_w)\left(Rot(\theta)\begin{bmatrix}a\\0\end{bmatrix} + \eta\begin{bmatrix}-y\\x\end{bmatrix}\right);$$

- Centrifugal component (due to the frame rotation)

$$\vec{a}_{cen} = \omega^2 R\begin{bmatrix}\cos(\theta+\alpha-\theta_w)\\ \sin(\theta+\alpha-\theta_w)\\ 0\end{bmatrix} = \omega^2 Rot(-\theta_w)\begin{bmatrix}x\\y\end{bmatrix}, \quad \text{with } \|\vec{a}_{cen}\| = \omega^2 R;$$

- Coriolis component (due to the coupling between the walker relative velocity and the rotation of the non-inertial frame)

$$\vec{a}_{Cor} = 2\vec{\omega} \times {}^w\vec{V}_w,$$

where ${}^w\vec{V}_w$ is the walker intended velocity, expressed in her/his frame.

Note that, for the inertial and centrifugal components, we have also given an expression that is computable and does not depend on the angle α (which would not be defined at the origin).

I.10 Acceleration-level control: Simulation results

In order to evaluate the performance of the proposed acceleration control scheme (25), we report here the results for the *virtual* square path of Fig. 14. In

Fig. 17, the absolute motion of the walker (as seen by the overlooking camera) is shown. The absolute orientation θ_w of the walker is displayed by a segment. Initially, we have $\theta = 0$ and $\theta_w = \pi/2$. In law (25), it is K = diag(20,20), while the "reference" behavior ξ_1^d for the linear and angular carpet velocities is given by the smooth feedback law (11-12) used in scheme (14), with associated control parameters $k = 1$, $k_w = 10$, and $|\omega_f| \leq 2$ rad/s. The corresponding linear and angular acceleration commands are shown in Fig. 18. Furthermore, in Fig. 19, the resulting linear and angular velocities of the carpet (blue, solid) are compared with the velocity commands (red, dashed) that would be generated by the first-order feedback controller (11-12) and (14) under the same walker path. This comparison confirms that the second-order controller achieves the same performance of the first-order law, while allowing a direct monitoring of system accelerations. In particular, thanks to the combined feedback and feedforward actions, the walker is rapidly brought close to the platform center and then kept there. As in the case of velocity control, note that the platform lags behind any turn per-

[4] Note that a constant velocity ${}^wV_w$ in the virtual world does not correspond to constant absolute velocities ($V_{w,x}$, $V_{w,y}$) in eq. (17).

formed on place by the walker, since there is no feedforward action triggered by a walker's angular motion without linear displacement. Finally, Fig. 20 displays the inertial, centrifugal and Coriolis components of the apparent acceleration felt by the user in the X_w and Y_w directions, due to the platform accelerations of Fig. 18, and computed as illustrated in Sect. I.9. Videos of this and other motion tasks are available at http://www.dis.uniroma1.it/~labrob/research/CW.html.

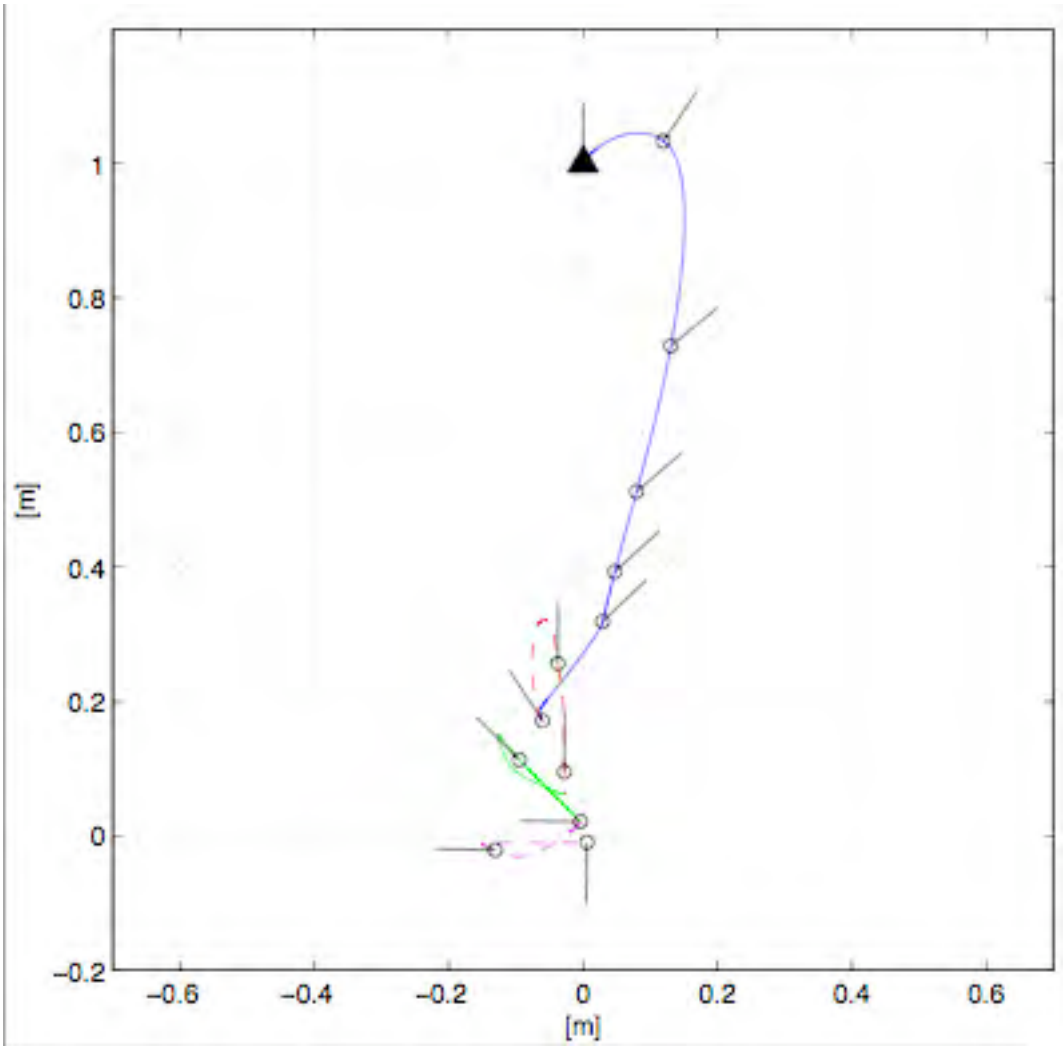

Fig. 17: Virtual square path: Walker absolute locomotion under the second-order platform controller (25), where the reference velocity behaviors are given by the smooth feedback law (11-12) used in scheme (14)

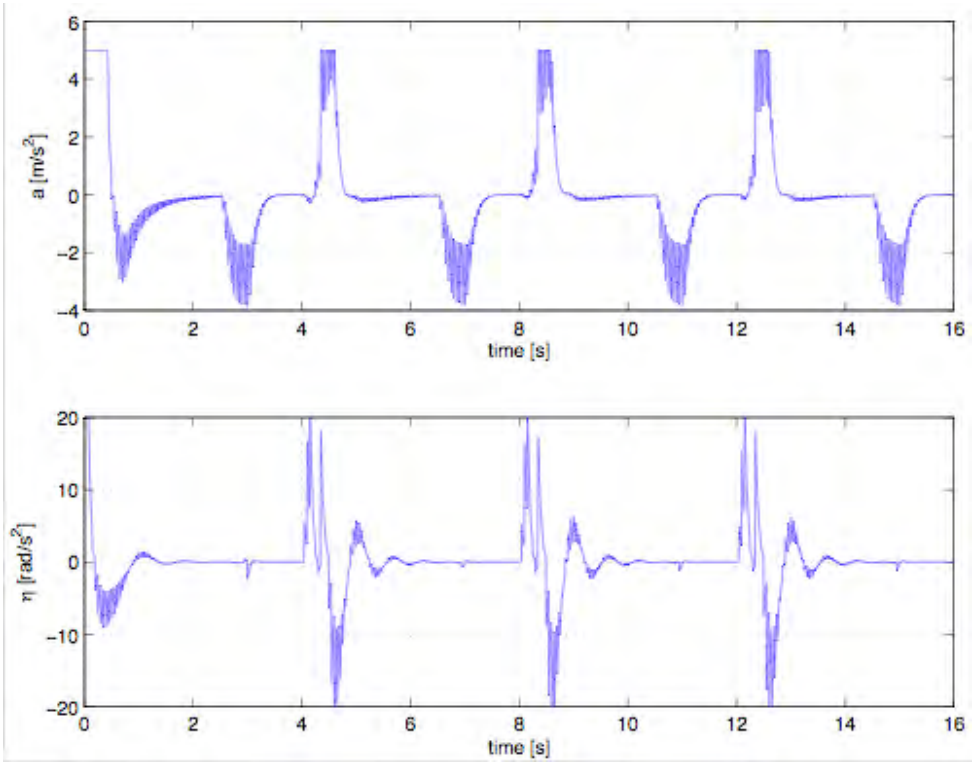

Fig. 18: Linear (top) and angular (bottom) acceleration commands for Fig. 17

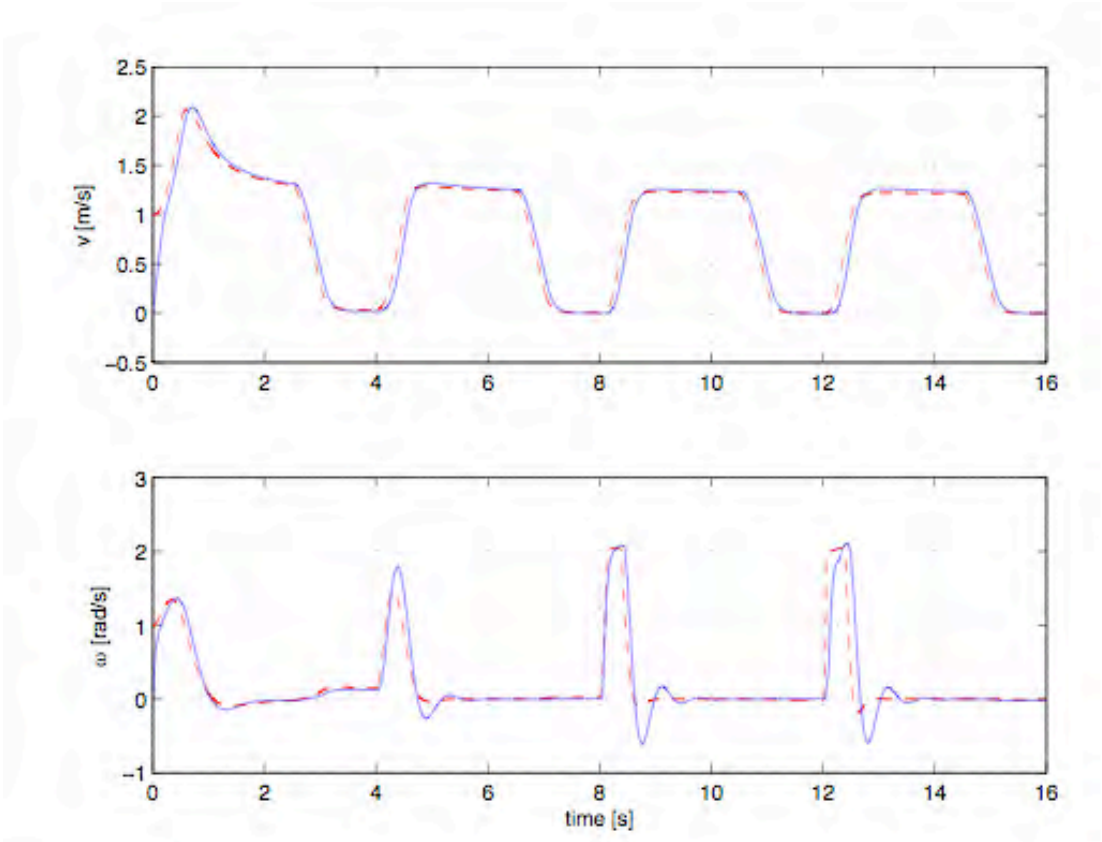

Fig. 19: Carpet linear (top) and angular (bottom) velocities (blue, solid) corresponding to the acceleration commands of Fig. 18, compared with the velocity commands (red, dashed) generated by the first-order feedback controller (11-12) and (14) for the same walker path of Fig. 14

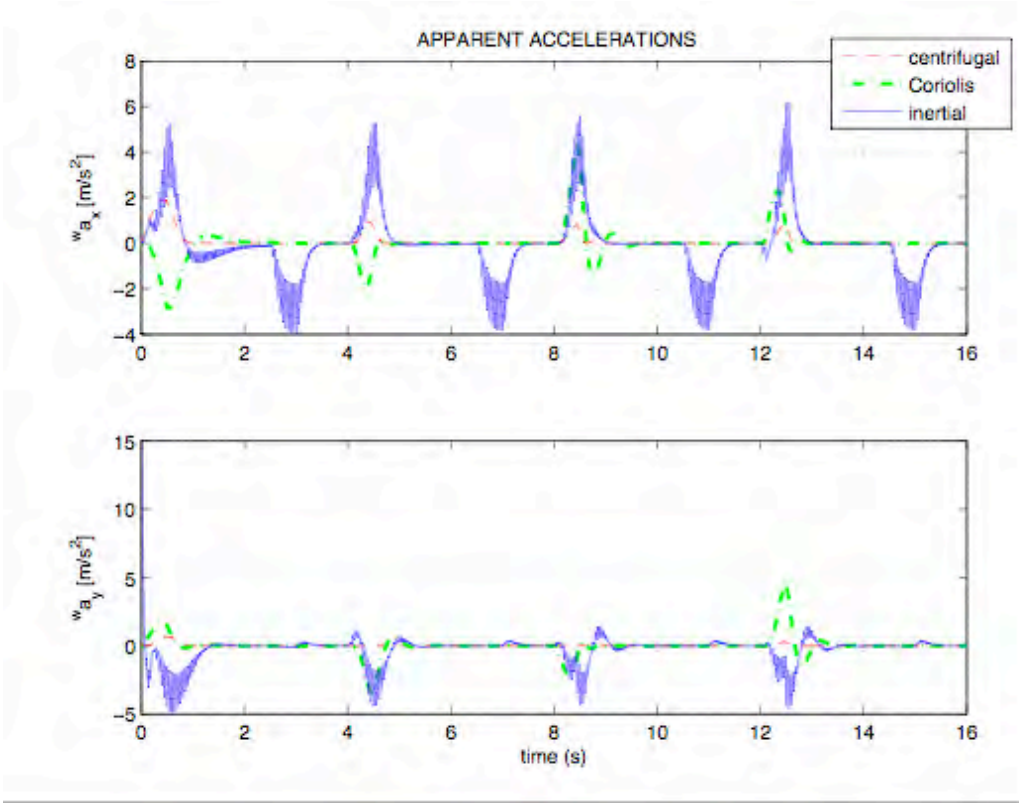

Fig. 20: Inertial, centrifugal, and Coriolis components of the apparent acceleration felt by the user in the X_w (top) and Y_w (bottom) directions, due to the carpet accelerations of Fig. 18

Part II —The linear platform

In this second part of the report, the control problem for the two-dimensional (2D) linear platform shown in Fig. 21 is considered (see deliverable T5.1/D1 for details on the kinematic design).

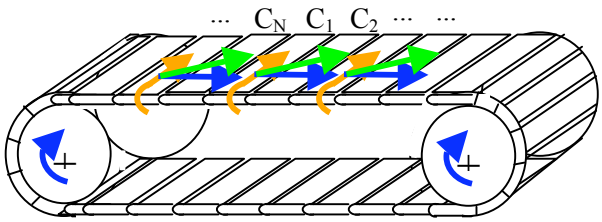

Fig. 21: The linear concept for the CyberCarpet

In principle, the design of a control law at the acceleration level may be performed in this case as already done for the ball-array kinematics, i.e., starting by tackling the (simpler) problem at the velocity level, and then *moving* the control law to the acceleration level by one of the methods of Sect. I.8. However, due to the much simpler (in particular, linear and decoupled) structure of the system model, we decided in this case to face the control problem directly at the acceleration level based on the following second-order kinematic model of the platform

$$\begin{cases} \dot{x}_i = v_i \\ \dot{v}_i = a_{ci} + a_{wi} \end{cases} \qquad i = 1,2. \tag{26}$$

In eqs. (26), for each of the two controlled directions (labeled i = 1,2, with $x_1 = x$ and $x_2 = y$) on the planar platform surface, x_i is the (measurable) absolute position of the walker, v_i is the (not measurable) absolute velocity of the walker, a_{ci} is the carpet acceleration (our control input), and a_{wi} is the walker voluntary acceleration (not measurable and regarded as a disturbance).

Since the system model and the corresponding control problem is exactly the same in the two orthogonal directions of platform motion, we will drop the index i in the following and consider from now on the control problem as one-dimensional (1D).

II.1 Acceleration-level control

The simplest way for stabilizing the linear system (26) to a desired position x_{ref} would be to use a control law of the form

$$a_c = -a_w - k_v v + k_x (x_{ref} - x), \tag{27}$$

with $k_v > 0$ and $k_x > 0$. However, this would require the availability of the unmeasurable quantities v and a_w. This problem can be solved by replacing v and a_w in eq. (27) with proper estimates $\hat{v}$ and $\hat{a}_w$, i.e., by taking

$$a_c = -\hat{a}_w - k_v \hat{v} + k_x (x_{ref} - x). \tag{28}$$

The problem of obtaining the estimates $\hat{v}$ and $\hat{a}_w$ is considered in the next two subsections.

II.1.1 Estimation of the walker voluntary acceleration

The walker voluntary acceleration a_w is estimated by the linear observer

$$\begin{cases} \dot{\xi}_1 = \xi_2 + k_1(x - \xi_1) \\ \dot{\xi}_2 = a_c + k_2(x - \xi_1) \\ \hat{a}_w = k_2(x - \xi_1), \end{cases} \tag{29}$$

with $k_1 > 0$ and $k_2 > 0$. In fact, in the Laplace domain, it results

$$\hat{A}_w(s) = \frac{k_2}{s^2 + k_1 s + k_2} A_w(s) = F_1(s) A_w(s),$$

showing that the estimation $\hat{a}_w$ is a stable, low-pass filtered version of the unknown quantity a_w.

II.1.2 Estimation of the walker absolute velocity

Similarly, an estimation of the walker absolute velocity v is provided by the linear observer

$$\begin{cases} \dot{\xi}_3 = k_3(x - \xi_3) \\ \hat{v} = k_3(x - \xi_3), \end{cases} \tag{30}$$

with $k_3 > 0$, yielding

$$\hat{V}(s) = \frac{k_3}{s + k_3} V(s) = F_2(s) V(s).$$

Note that the dynamics of the two estimations $\hat{v}$ and $\hat{a}_w$ are completely independent. Furthermore, having available a good estimation $\hat{v}$ allows to estimate also the walker voluntary velocity v_w as

$$\hat{v}_w = \hat{v} - v_c,$$

where v_c is the carpet velocity (assumed measurable).

The block diagram of the overall control law (28), where eqs. (29) and (30) are used for the estimations $\hat{a}_w$ and $\hat{v}$, is represented in Fig. 22. By standard manipulation of the block diagram, the equivalent closed-loop system is shown in Fig. 23, where

$$P_1(s) = \frac{s + k_3}{s^3 + k_3 s^2 + (k_v k_3 + k_x) s + k_x k_3}$$

and

$$P_2(s) = \frac{s^2 + k_1 s}{s^2 + k_1 s + k_2}.$$

There are five control parameters to choose: k_1, k_2, k_3, k_v, k_x. Using Routh's criterion, we get that both $P_1(s)$ and $P_2(s)$ are asymptotically stable if and only if all these parameters are positive. Note also that $P_2(s)$ is a high-pass filter; therefore, it allows recovering to zero the position error for any constant disturbance a_w.

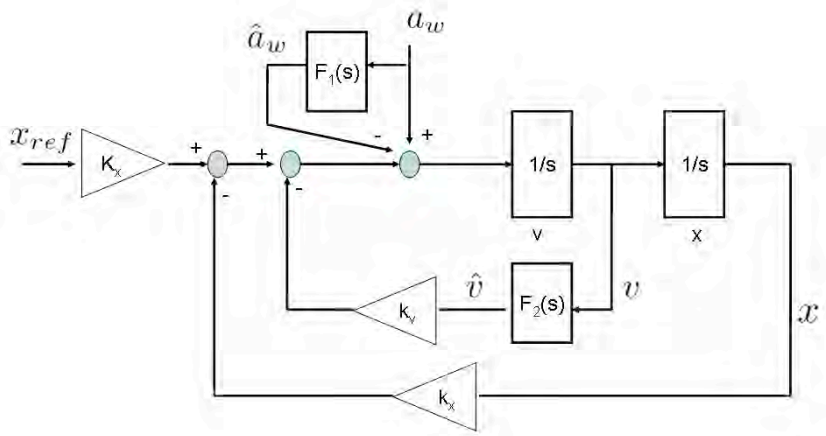

Fig. 22: The block diagram of control law (28) with the estimations $\hat{a}_w$ and $\hat{v}$ given by eqs. (29) and (30)

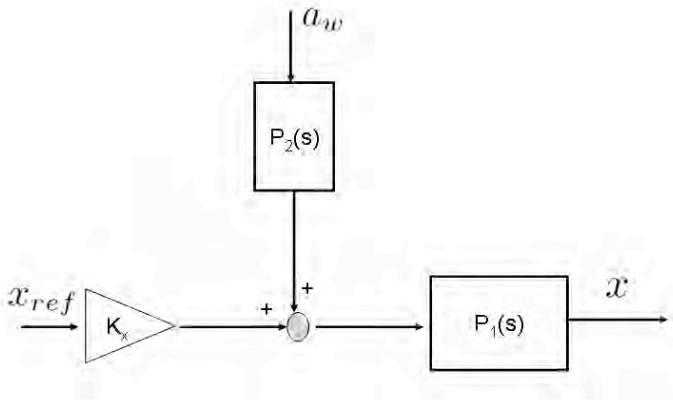

Fig. 23: An equivalent scheme for the closed-loop system of Fig. 22

II.2 Simulation results

We report here the results of a simulation, where the walker moves with a bang-coast-bang acceleration (0.5 m/s^2 for 2 *s*, 1 m/s for 8 *s*, and -0.5 m/s^2 for 2 *s*), starting from the platform center with zero velocity and stopping the voluntary motion after 12 *s*. The five control parameters were chosen as follows:

$$\begin{cases} k_1 = 14 \\ k_2 = 49 \\ k_3 = 7 \\ k_v = 5 \\ k_x = 6 \end{cases}$$

Note that these values result into real and coincident poles at -7 for both observers (29) and (30), thus preventing overshooting of the estimates, and that the observation transients are more than critically damped and slightly faster than the position error recovery.

In Fig. 24 we report the behaviors of the walker voluntary acceleration a_w (blue) and of its estimation $\hat{a}_w$ (red), while Fig. 25 shows the walker absolute velocity v (blue) and its estimation $\hat{v}$ (red). In particular, the estimation $\hat{v}$ accurately reproduces v, so that a good estimate of the walker voluntary velocity $\hat{v}_w = \hat{v} - v_c$ can be also obtained as a byproduct (see Fig. 26). In the same Figure, the behavior of the alternative estimation $\hat{v}_w^{\text{int}} = \int \hat{a}_w dt$ (open-loop integration of the estimated walker voluntary acceleration) is also displayed (in green). This second approach would avoid the need of the observer (30) for $\hat{v}$, replaced in this case by $\hat{v} = \hat{v}_w^{\text{int}} + v_c$, but, of course, it would not allow recovering from any error on the initial state, noise, numerical drift, and so on. Finally, the (smooth) acceleration command a_c and the resulting walker absolute position x are reported in Fig. 27 and Fig. 28, respectively.

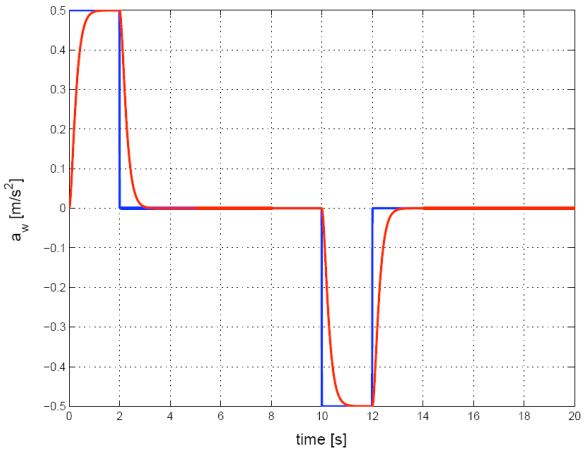

Fig. 24: Walker voluntary acceleration a_w (blue) and its estimation $\hat{a}_w$ (red) with observer (28)

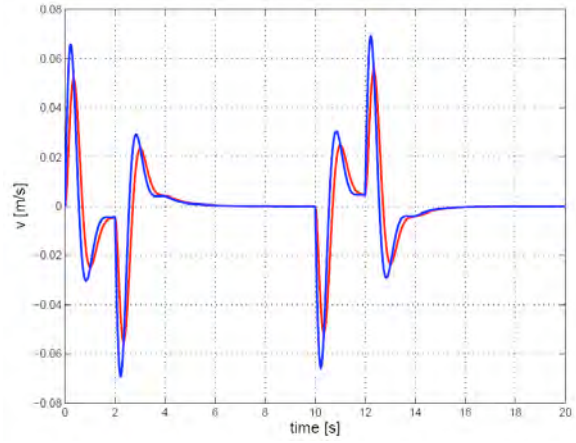

Fig. 25: Walker absolute velocity v (blue) and its estimation $\hat{v}$ (red) with observer (30)

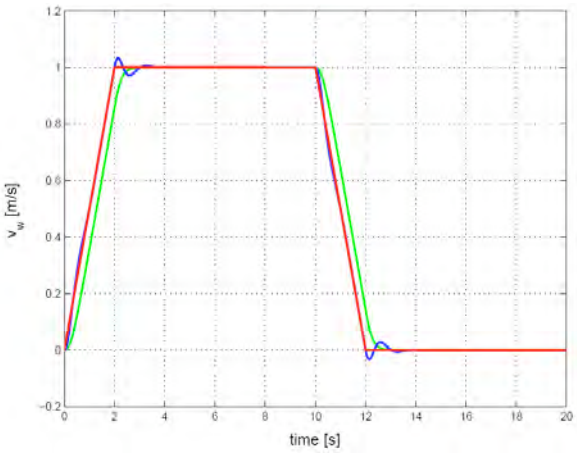

Fig. 26: Walker voluntary velocity v_w (red), and its estimations $\hat{v}_w = \hat{v} - v_c$ (blue) and $\hat{v}_w^{\text{int}} = \int \hat{a}_w dt$ (green)

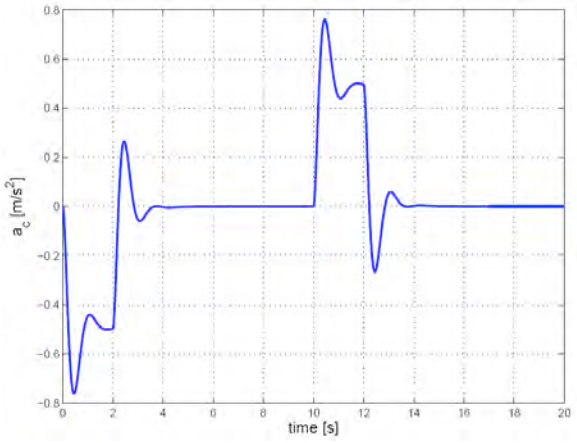

Fig. 27: The acceleration command a_c

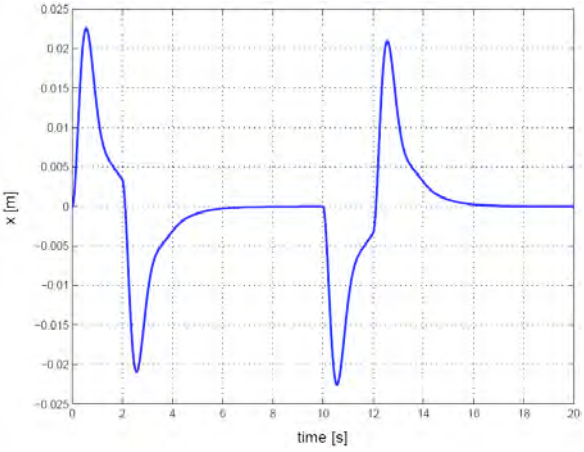

Fig. 28: Walker absolute position x

Conclusions

In the two parts of this report, motion control laws for the two design concepts of the *CyberWalk* omnidirectional platform have been presented. We have considered both velocity-level and acceleration-level control designs, combining a position feedback action with a feedforward action at the differential level. In this framework, the use of dynamic observers allows to fully compensate for the intentional unknown motion of the walker and/or to avoid the use of quantities that are unavailable for direct measure (e.g., the velocity and acceleration of the walker).

For the "ball-array" platform, special care has been devoted to eliminate the problems occurring from the nonholonomic nature of the system (which prevents, in certain configurations, actuating instantaneous directions of walker's recovery toward the platform center). In this case, first a smooth nonlinear and singularity-free velocity-level controller has been incrementally designed and then completed with an estimator of the walker's intentional velocity. Next, the controller has been moved to the acceleration-level by using a cascaded approach that relies on the already developed velocity control laws.

For the "linear" platform, by taking advantage of the linearity and decoupled characteristics of its kinematic model, a control solution has been derived directly in terms of acceleration inputs to the system. The complete design for the 2D platform is the simple composition of the two independent controllers in the two orthogonal directions of actuation of the platform.

For ease of reference, the best performing control laws presented in this report (with reference to the equations needed) and the number of control parameters involved are summarized in the following Table.

	Ball-Array platform	**Linear platform**
Velocity control	Eq. (14) with (11)-(12) and observers (15)-(16) **2 parameters**	None (but possible time integration of acceleration law below)
Acceleration control	Eq. (25) using all 5 equations above **4 parameters**	Eq. (28) with observers (29)-(30) **5 parameters**

Assuming velocities or accelerations (translational and rotational for the ball-array platform, and both translational for the full-size 2D *CyberWalk* linear platform) as control commands complies with the available inputs to servo-drives commonly used for the direct actuation of such mechanical systems. It should be stressed that the acceleration-level design allows to evaluate more accurately dynamic effects acting on the walker because of the controlled motion of the platform and to generate smoother and constrained commands that satisfy physiological upper bounds. Last but not least, even if the actuation hardware of the platform accepts only velocity inputs, these can be generated by a simple integration of the acceleration commands.

Extensive numerical tests of the proposed controllers have been performed by simulation, including a graphical environment to visualize the overall motion of the walker/platform system. Videos are available at the web page http://www.dis.uniroma1.it/~labrob/research/CW.html of the UOR partner, which is accessible also from the project web site http://www.cyberwalk-project.org/.

The velocity control laws have been already successfully implemented on the small-size ball-array platform developed by the partner TUM-U (see Fig. 31), including also the 2D visual tracking system developed by the partner ETHZ. Similarly, the acceleration control laws have been already tested with satisfactory performance on the 1D linear treadmill available at the site of the partner MPS (see Fig. 32), where also preliminary test on virtual immersion have been conducted. At the time of deliver of this report, work is already under way for the final implementation on the 2D large-scale linear platform designed at TUM-U (see Fig. 33). The full details on the control implementation, including software integration and hardware interface, sampling rates of the visual and communication systems, fine tuning of control parameters for taking into account the additional perceptual constraints, and so on will be described in the deliverable T5.3 due @ M36. A preliminary version @M24 is already available.

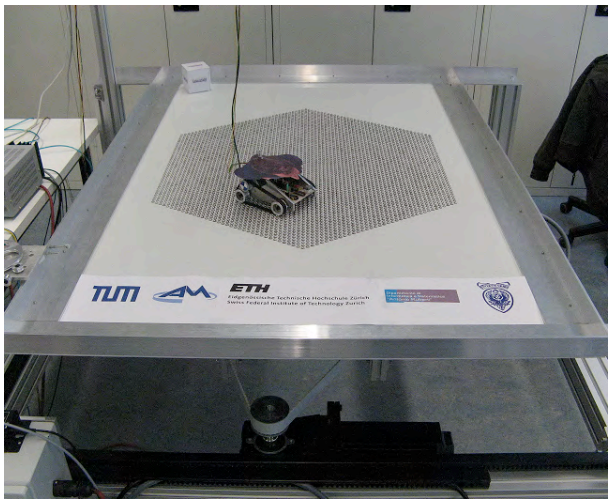

Fig. 29: The ball-array platform in its small-size version at the partner site TUM-U

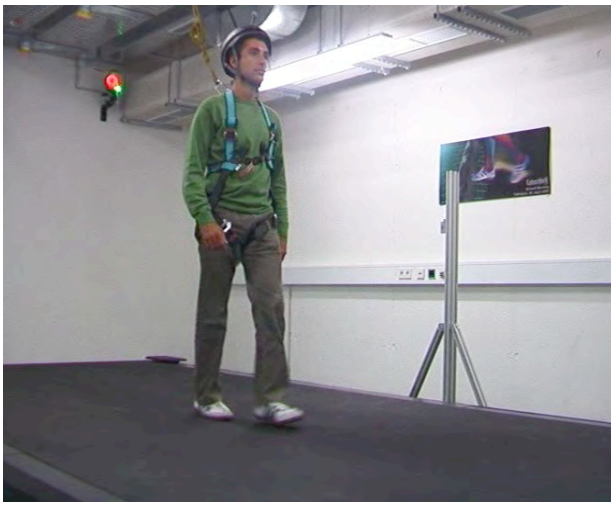

Fig. 30: The 1D linear treadmill at the partner site MPS

Fig. 31: The current status of the 2D linear platform at the partner site TUM-U

References

[1] H. Iwata, “Locomotion interface for virtual environments” in Proc. 9th Int. Symp. on Robotics Research, pp. 275–282, 2000.

[2] J. M. Hollerbach, “Locomotion interfaces” in Handbook of Virtual Environments Technology (K. M. Stanney, Ed.), pp. 239–254, 2002.

[3] 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.

[4] H. Iwata, “The Torus Treadmill: Realizing locomotion in VEs” IEEE Computer Graphics and Applications, vol. 9, pp. 30–35, 1999.

[5] 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 VR2005), (Bonn, D), 2005.

[6] A. De Luca, R. Mattone, and P. Robuffo Giordano, “The motion control problem for the CyberCarpet” in Proc. 2006 IEEE Int. Conf. on Robotics and Automation, pp. 3532–3537, (Orlando, FL), 2006.

[7] A. De Luca, R. Mattone, and P. Robuffo Giordano, “Feedback/feedforward schemes for motion control of the CyberCarpet” in Proc. 2006 IFAC Symp. on Robot Control, (Bologna, I), 2006.

[8] 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, pp. 27–35, 1995.

[9] 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.

[10] E. Panteley and A. Lorýa, “On global uniform asymptotic stability of nonlinear time-varying systems in cascade” Systems and Control Lett., vol. 33, no. 2, pp. 131–138, 1998.

[11] A. De Luca, R. Mattone, and P. Robuffo Giordano, “Acceleration-level control of the CyberCarpet” in Proc. 2007 IEEE Int. Conf. on Robotics and Automation, pp. 2330–2335, (Roma, I), 2007.