Active and Passive Control of Walk-Assist Robot for Outdoor Guidance

Chun-Hsu Ko, Kuu-Young Young, Yi-Che Huang, and Sunil Kumar Agrawal

***Abstract*—In the currently aging society, walk-assist robots can play an important role in improving the activities of daily living of the elderly. In this paper, we propose a robot walking helper with both passive and active control modes for guidance. From the perspective of human safety, the passive mode adopts a braking control law on the wheels to differentially steer the vehicle. However, if the user walks uphill in the outdoor environment, external forces need to be supplied to the human-walker system. In this paper, we add an active mode to guide the user in situations where the passive control mode alone with user-applied forces is not adequate for guidance. The theory of differential flatness is used to plan the trajectory of control gains within the proposed scheme of the controller. Since the user input force and slope angle of the path are not known *a priori*, the theory of model predictive control is used to periodically compute the trajectory of these control gains. The simulation and experiment results show that the walk-assist robot, along with the structure of this proposed control scheme, can guide the user to a goal on a slope effectively.**

***Index Terms*—Differential flatness, model predictive control (MPC), slope guidance, walk-assist robot.**

I. Introduction

MANY robot walking helpers have been proposed to assist in walking and enhance the user range of activities of daily living [1]–[16]. Machine systems have been proposed that support useful functions of the users such as guidance [1], [9]–[16], obstacle avoidance [2], [10], and health monitoring [1], [7]. To control a robot walking helper, Spenko *et al.* [7] used a variable damping model to increase the walking stability. Chuy *et al.* [2] used passive behavior to enhance the interaction between the user and the active support system. Hirata *et al.* [10] proposed an adaptive motion control algorithm for obstacle/step avoidance and gravity compensation. Agrawal *et al.* [12] proposed a two-phase passive control algorithm for guiding a user to attain desired position and orientation, while allowing for small errors.

In general, the robot walking helpers can be classified as: 1) active and 2) passive. The active robot walking helpers [1]–[5] use servo motors to guide the user, while actively adding energy to the system. The passive walking helpers [10]–[12] move only by user supplied forces. Controlled brakes are used to steer the walker while continuously dissipating energy from the system. With this property of dissipation of energy, they are intrinsically safe when used by humans. However, if a user wants to walk uphill in the outdoor environment, energy must be added to the walker. The passive mode may not be enough to achieve the required control. Hence, it is important to consider an integrated strategy with both active and passive controls in the slope guidance.

In this paper, we propose combining passive and active control modes for effective guidance of the robot walking helper. Passive mode uses a braking control law on the wheels to differentially steer the vehicle. When a user pushes the walker uphill, active mode is used to compensate for gravity. The motor control torques are then obtained by adding brake torque and active torque. Since the robot applies the active torque to help human walking, the required user-applied force for walking uphill can be reduced. Hence, the proposed active–passive integrated method has the advantages of the improvement of the safety of the robot compared to the active-type robot and energy saving of the user on an uphill road compared to the passive control method. Furthermore, the method of differential flatness is used to select time varying trajectories of the control gains so that the vehicle achieves a desired final position and orientation [16]. The differential flatness method [16]–[19] has been proposed to solve both the trajectory planning and robot control problems within a common framework. A system is differentially flat if there exists a set of outputs, called flat outputs, equal in number to the number of inputs, such that all states and inputs can be algebraically expressed in terms of these outputs and their derivatives [17]. For a flat system, the trajectories of flat outputs can be first planned with a set of parameters and mode functions. The system dynamic equations and the input constraints can then be transformed into a set of algebraic equations, which are generally simpler to solve [18]. Previous studies [17], [18] have demonstrated the efficiency of the differential flatness method in controlling wheeled mobile robot in a plane, which motivates us to further explore its possibility in controlling robot walker on a slope.

In this approach, the nonlinear structure of the vehicle dynamic equations on a slope, under the active and passive

Manuscript received December 3, 2011; revised April 9, 2012; accepted May 19, 2012. Date of publication June 11, 2012; date of current version January 18, 2013. Recommended by Technical Editor A. Goswami. This work was supported by the National Science Council of Taiwan under Grant NSC 99-2221-E-009-157. The work of S. K. Agrawal was supported in part by the World Class University program. This paper was presented in part at the IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, October 18–22, 2010.

C.-H. Ko is with the Department of Electrical Engineering, I-Shou University, Kaohsiung 84001, Taiwan (e-mail: chko@isu.edu.tw).

K.-Y. Young and Y.-C. Huang are with the Department of Electrical Engineering, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: kyoung@mail.nctu.edu.tw; ychuang.ece96g@nctu.edu.tw).

S. K. Agrawal is with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716 USA (e-mail: agrawal@udel.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMECH.2012.2201736

control, is explored to seek the property of differential flatness. The states and inputs of the robot walker are first represented as functions of the flat outputs and their derivatives. The trajectories of flat outputs are then planned with three planning methods that include static path planning, constant speed trajectory planning, and acceleration/deceleration trajectory planning. To efficiently select the braking gains during guidance, a passive guidance controller based on user-applied force is proposed. Since the user-applied forces and slope angles are not known *a priori*, model predictive control (MPC) is used to find the solution repetitively. Finally, simulations and experiments are performed with this robot walker on a slope to demonstrate the effectiveness of the proposed approach.

The remainder of this paper is organized as follows: Section II describes the developed robot walking helper. In Section III, the differential flatness approach is addressed. Trajectory planning and controller design are described in Sections IV and V, respectively. Sections VI and VII present a short summary of MPC and the simulation results, respectively. Experimental results are included in Section VIII. Finally, concluding remarks are given in Section IX.

II. Developed Robot Walking Helper

The robot walking helper is shown in Fig. 1. It contains a support frame, two wheels driven by motors with encoders, two passive casters, a force sensor, a slope sensor, and a pc-based controller. The support frame has a semienclosed design that allows the user to walk at the center of the helper and provides good support and stability. The motors provide the torques for controlling the motion of the robot walking helper. The encoders are used to measure the wheel speed. A force sensor detects the user-applied force and the slope sensor detects the slope angle. The controller is used to control the torques of the motors with the measured sensor information. To design the controller, a simplified dynamic model on a slope is described next.

Fig. 2 shows the robot on a slope in the world coordinates *xyz* [16] given by

$$q = [x_c, y_c, z_c, \theta]^T \tag{1}$$

where x_c, y_c, z_c are the coordinates of the center of mass and θ is the heading angle of the robot. The slope angle α is assumed to be a function of y_c, $\alpha = \alpha(y_c)$. When the robot moves on the slope, the body coordinates $x'y'z'$ rotate by θ about z-axis and α about the x-axis. The rotation matrix R is calculated as

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \cos\alpha\sin\theta & \cos\alpha\cos\theta & -\sin\alpha \\ \sin\alpha\sin\theta & \sin\alpha\cos\theta & \cos\alpha \end{bmatrix}. \tag{2}$$

With the assumption of no-slip at the wheel contact points on ground, the velocity of the robot is parallel to the heading direction. Hence, $\dot{q}$ can be expressed as

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{z}_c \end{bmatrix} = v\hat{x}' = v \begin{bmatrix} \cos\theta \\ \sin\theta\cos\alpha \\ \sin\theta\sin\alpha \end{bmatrix} \tag{3}$$

Fig. 1. Robot walking helper.

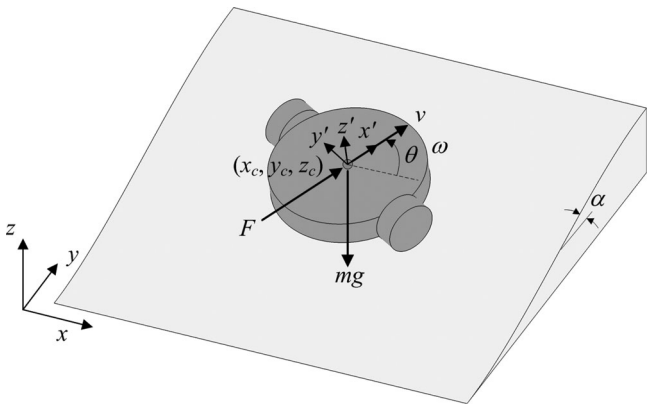


Fig. 2. Configuration of a robot walking helper on a slope.

and

$$\dot{\theta} = \omega \tag{4}$$

where v is the heading speed and ω is the turning speed. The angular velocity can be expressed as

$$\vec{\omega} = \dot{\alpha}\,\hat{x} + \omega\,\hat{z}' = [\,\dot{\alpha} \quad -\omega\sin\alpha \quad \omega\cos\alpha\,]^T \tag{5}$$

where $\dot{\alpha} = \alpha'(y_c)\dot{y}_c$. With the user-applied force F, robot applying torques, the gravity force and the no-slip constraint, the equations of motion are written as

$$m\,[\,\ddot{x}_c \quad \ddot{y}_c \quad \ddot{z}_c\,]^T = \left(F + \frac{\tau_r + \tau_l}{r}\right)\hat{x}' - mg\hat{z} + \lambda\hat{y}' + f_n\hat{z}' \tag{6}$$

and

$$I\dot{\vec{\omega}} + \vec{\omega} \times I\vec{\omega} = \frac{b(\tau_r - \tau_l)}{r}\hat{z}' + M_x\hat{x}' + M_y\hat{y}'. \tag{7}$$

Here, m is the robot mass, I is the moment of inertia of the robot, r is the wheel radius, b is half distance between the two wheels, τ_r and τ_l are the motor torques on the right and left wheels, λ

is the constraint force, f_n is the normal reaction force and M_x, M_y is the constraint moments. The moment of inertia I can be calculated as

$$I = RI_{\text{body}}R^T, I_{\text{body}} = \begin{bmatrix} I_{11} & I_{12} & I_{13} \\ I_{12} & I_{22} & I_{23} \\ I_{13} & I_{23} & I_{33} \end{bmatrix}. \quad (8)$$

With the assumption of symmetry about the plane $x'z'$, the values of I_{12} and I_{23} are zeros. By differentiating (3), $[\ddot{x}_c, \ddot{y}_c, \ddot{z}_c]^T = v\dot{\hat{x}}' + \dot{v}\hat{x}'$. On substituting $[\ddot{x}_c, \ddot{y}_c, \ddot{z}_c]^T$ into (6) and premultiplying by $\hat{x}'^T$, we obtain

$$m\dot{v} = F + \frac{\tau_r + \tau_l}{r} - mg\sin\theta\sin\alpha. \quad (9)$$

Furthermore, by differentiating (5)

$$\dot{\vec{\omega}} = [\,\ddot{\alpha} \quad -\dot{\omega}\sin\alpha - \omega\cos\alpha\dot{\alpha} \quad \dot{\omega}\cos\alpha - \omega\sin\alpha\dot{\alpha}\,]^T. \quad (10)$$

On substituting (10) into (7) and premultiplying by $\hat{z}'^T$, we obtain

$$I_{33}\dot{\omega} = \frac{b(\tau_r - \tau_l)}{r} - \dot{\alpha}^2\cos\theta\sin\theta(I_{11} - I_{22}) - \dot{\alpha}\sin\theta\omega I_{13}. \quad (11)$$

From (3), (4), (9), and (11), the state equations of the robot on a slope are expressed as

$$\begin{aligned} \dot{x}_c &= v\cos\theta \\ \dot{y}_c &= v\sin\theta\cos\alpha \\ \dot{z}_c &= v\sin\theta\sin\alpha \\ \dot{\theta} &= \omega \\ \dot{v} &= \frac{\tau_r + \tau_l}{mr} + \frac{F}{m} - g\sin\theta\sin\alpha \\ \dot{\omega} &= \frac{b(\tau_r - \tau_l)}{I_{33}r} - \dot{\alpha}^2\cos\theta\sin\theta\frac{I_{11} - I_{22}}{I_{33}} \\ &\quad - \dot{\alpha}\sin\theta\omega\frac{I_{13}}{I_{33}}. \end{aligned} \quad (12)$$

Note that the term $-g\sin\theta\sin\alpha$ in the aforementioned equations arises from gravity effect. To compensate for the gravity and control in passive behavior, the control law is chosen as

$$\tau_r = -K_r\dot{\theta}_r + \frac{1}{2}mgr\sin\theta\sin\alpha,$$

$$\tau_l = -K_l\dot{\theta}_l + \frac{1}{2}mgr\sin\theta\sin\alpha \quad (13)$$

where τ_r and τ_l are the control torques of the right and the left wheels, respectively. $\dot{\theta}_r$ and $\dot{\theta}_l$ are the angular speeds of the right and the left wheels, respectively. K_r and K_l are nonnegative parameters. With the no-slip condition, the angular speeds $\dot{\theta}_r$ and $\dot{\theta}_l$ can be calculated as

$$\dot{\theta}_r = (v + b\omega)/r,\ \dot{\theta}_l = (v - b\omega)/r. \quad (14)$$

Substituting (13) and (14) into (12), we can obtain the dynamic equations of the robot given by

$$\begin{aligned} \dot{q} &= SV \\ \dot{V} &= AK + B \end{aligned} \quad (15)$$

where

$$V = \begin{bmatrix} v \\ \omega \end{bmatrix}, S = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta\cos\alpha & 0 \\ \sin\theta\sin\alpha & 0 \\ 0 & 1 \end{bmatrix}$$

$$A = \begin{bmatrix} -\dfrac{v + b\omega}{mr^2} & -\dfrac{v - b\omega}{mr^2} \\ -\dfrac{b(v + b\omega)}{I_{33}r^2} & \dfrac{b(v - b\omega)}{I_{33}r^2} \end{bmatrix}, \quad K = \begin{bmatrix} K_r \\ K_l \end{bmatrix}$$

$$B = \begin{bmatrix} F/m \\ -\dot{\alpha}^2\cos\theta\sin\theta\dfrac{I_{11} - I_{22}}{I_{33}} - \dot{\alpha}\sin\theta\omega\dfrac{I_{13}}{I_{33}} \end{bmatrix}. \quad (16)$$

The nonnegative parameters in K can now be regarded as the new control inputs that need to be planned to steer the vehicle from its current state to the desired goal state. To design the feedback controller, we select the brake gain K by using the differential flatness approach, described in the next section.

III. Differential Flatness Approach

In the differential flatness approach [16]–[19], we first define the flat outputs and then express all state variables and input in terms of the flat outputs and their derivatives. The wheel mobile robot has been shown to be differentially flat with the flat outputs $(y_1, y_2) = (x_c, y_c)$ [17]. We, thus, use these flat outputs to express the robot states and input gains on a slope. With a given slope height $z(y_2)$, the slope angle and its derivatives can be expressed as

$$\begin{aligned} \alpha &= \tan^{-1} z'(y_2) \\ \dot{\alpha} &= \frac{z''}{1 + z'^2}\dot{y}_2 \\ \ddot{\alpha} &= \frac{z'''\dot{y}_2^2 + z''\ddot{y}_2}{1 + z'^2} - \frac{2z'z''^2\dot{y}_2^2}{(1 + z'^2)^2}. \end{aligned} \quad (17)$$

Define s as the length of the slope, then

$$\begin{aligned} \dot{s} &= \dot{y}_2/\cos\alpha \\ \ddot{s} &= \frac{\ddot{y}_2 + \dot{y}_2 z'\dot{\alpha}}{\cos\alpha} \\ \dddot{s} &= \frac{\dddot{y}_2 + 2\ddot{y}_2 z'\dot{\alpha} + 2\dot{y}_2 z'^2\dot{\alpha}^2 + \dot{y}_2\dot{\alpha}^2 + \dot{y}_2 z'\ddot{\alpha}}{\cos\alpha} \end{aligned} \quad (18)$$

and

$$\begin{aligned} \theta &= \arctan\left(\frac{\dot{s}}{\dot{y}_1}\right) \\ v &= \sqrt{\dot{y}_1^2 + \dot{s}^2} \\ \omega = \dot{\theta} &= \frac{\dot{y}_1\ddot{s} - \ddot{y}_1\dot{s}}{\dot{y}_1^2 + \dot{s}^2} \\ \dot{v} &= \frac{\dot{y}_1\ddot{y}_1 + \dot{s}\ddot{s}}{\sqrt{\dot{y}_1^2 + \dot{s}^2}} \\ \dot{\omega} &= (-\dddot{y}_1\sin\theta + \dddot{s}\cos\theta - 2\dot{v}\omega)/v. \end{aligned} \quad (19)$$

With (16)–(19), the control gain K can be expressed with the flat outputs and their derivatives, given by

$$K(\dot{y}_1, \ddot{y}_1, \dddot{y}_1, y_2, \dot{y}_2, \ddot{y}_2, \dddot{y}_2) = A^{-1}\left(\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} - B\right). \quad (20)$$

Note that K should be nonnegative in passive control. When the angular speed $\dot{\theta}_r$ or $\dot{\theta}_l$ is zero, A^{-1} does not exist due to singularity and K cannot be obtained. Under this condition, the walker speed should be increased with the brake gain K set to zero. The aforementioned equation can be used in planning a desired trajectory with the passive mode.

IV. Trajectory Planning for Slope Guidance

In trajectory planning of this walk-assist robot for slope guidance, we intend to find a smooth trajectory ($y_1(t)$, $y_2(t)$) that passes from the initial point to target point during a period of t_f. Here, t_f is derived via an optimization process described shortly. Three planning methods, that include static path planning, constant speed trajectory planning, and acceleration/deceleration trajectory planning, are used to determine the brake gain for the proposed controller in real time, which will be discussed later in Section IV for controller design. Note that the process of trajectory planning is executed each time the current speed of the walker is updated. These trajectories planning methods are briefly described as follows.

A. Static Path Planning

In static path planning [20], we first select a set of waypoints that can be connected into a smooth path. As shown in Fig. 3(a) and (b), there are two starting circles with radius c tangent to the starting position and direction. On the other hand, two ending circles with radius c are tangent to the end position and direction. There are four smooth paths that consist of a starting arc, a straight line, and an ending arc, to be selected. Since an arc path is more difficult to follow than a straight line, the path combination with the minimum sum of starting arc angle θ_s and ending arc angle θ_e is preferred. Fig. 3(a) shows that a minimum arc-angle path with three waypoints is selected. Furthermore, if starting arc angle θ_s or ending arc angle θ_e is larger than a predefined arc angle θ_m, the arc is split into small arcs that have smaller arc angle than θ_m. As shown in Fig. 3(b), five waypoints are added to the path. With this waypoint selection, we can quickly obtain a smooth path that consists of subpaths of small angle arcs ($\leq \theta_m$) or straight lines. The robot can, thus, guide the user to pass through these waypoints consecutively.

B. Constant Speed Trajectory Planning

The stable walking trajectory with a small variance of walking speed is generated by using this constant speed trajectory planning. It takes the current speed of the walker as a constant speed to plan a reference trajectory for guidance. Because each subpath is a line or an arc, the trajectory can be fitted with the following polynomial function of time [21]:

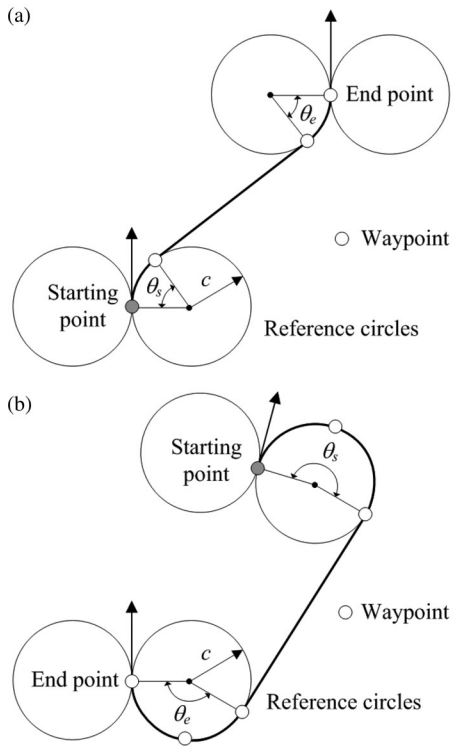


Fig. 3. Path planning: (a) three waypoints path planning and (b) five waypoints path planning.

$$y_j(t) = c_{j0} + c_{j1}t + c_{j2}t^2 + c_{j3}t^3 + c_{j4}t^3(t - t_f) + c_{j5}t^3(t - t_f)^2,\ j = 1, 2. \quad (21)$$

With the boundary conditions $y_j(0)$, $\dot{y}_j(0)$, $\ddot{y}_j(0)$, $y_j(t_f)$, $\dot{y}_j(t_f)$, $\ddot{y}_j(t_f)$, the coefficients c_{jk} can be easily calculated as

$$\begin{aligned} c_{j0} &= y_j(0), c_{j1} = \dot{y}_j(0), c_{j2} = 0.5\ddot{y}_j(0) \\ c_{j3} &= (y_j(t_f) - c_{j0} - t_f c_{j1} - t_f^2 c_{j2})/t_f^3 \\ c_{j4} &= (\dot{y}_j(t_f) - c_{j1} - 2t_f c_{j2} - 3t_f^2 c_{j3})/t_f^3 \\ c_{j5} &= (0.5\ddot{y}_j(t_f) - c_{j2} - 3t_f c_{j3} - 3t_f^2 c_{j4})/t_f^3, j = 1, 2. \end{aligned} \quad (22)$$

Here, the initial conditions $y_1(0)$, $\dot{y}_1(0)$, $\ddot{y}_1(0)$, $y_2(0)$, $\dot{y}_2(0)$, $\ddot{y}_2(0)$ and the final conditions $y_1(t_f)$, $\dot{y}_1(t_f)$, $\ddot{y}_1(t_f)$, $y_2(t_f)$, $\dot{y}_2(t_f)$, $\ddot{y}_2(t_f)$ of the trajectory can be calculated from the initial state $x_c(0)$, $y_c(0)$, $\theta(0), v(0)$, $\omega(0)$, the final state $x_c(t_f)$, $y_c(t_f)$, $\theta(t_f), v(t_f)$, $\omega(t_f)$, and the given slope height $z(y_2)$, with those parameters expressed as

$$\begin{aligned} y_1 &= x_c,\ \dot{y}_1 = v\cos\theta,\ \ddot{y}_1 = \dot{v}\cos\theta - v\omega\sin\theta \\ y_2 &= y_c,\ \dot{y}_2 = v\sin\theta\cos\alpha \\ \ddot{y}_2 &= \dot{v}\sin\theta\cos\alpha + v\omega\cos\theta\cos\alpha - v^2\alpha'\sin^2\theta\sin\alpha\cos\alpha. \end{aligned} \quad (23)$$

Note that these expressions in (23) demand the values of the initial and final accelerations $\dot{v}(0)$, $\dot{v}(t_f)$. For guiding the user into stable walking, the constant speed trajectory is preferred and the values of $\dot{v}(0)$, $\dot{v}(t_f)$ are set to be zeros. Furthermore, the final speed $v(t_f)$ is set to be the initial speed $v(0)$ and the final angular speed $\omega(t_f)$ set to be zero.

Once the flat outputs are obtained using (21)–(23), the robot states and inputs can be represented as functions of t_f. The constant speed trajectory $(y_1^c(t), y_2^c(t))$ and the value of the final time t_f are then obtained by solving the following nonlinear optimization problem:

$$\min_{t_f} J = \frac{1}{t_f}\int_0^{t_f} (v(t) - v(0))^2 dt \tag{24}$$

where J is the difference between the walking speed and initial speed. A smaller J implies a more stable walking trajectory. Note that only the parameter t_f is required in solving the optimization problem so that constant speed trajectory planning can be fast executed.

C. Acceleration/Deceleration Trajectory Planning

The acceleration/deceleration trajectory planning is used to change the robot speed during guidance. In this planning, we propose using the time transform technique on the constant speed trajectory $(y_1^c(t),\ y_2^c(t))$ already obtained. With a given time transform function $u(t)$, the acceleration/deceleration trajectory $(y_1^a(t),\ y_2^a(t))$ is expressed as

$$y_1^a(t) = y_1^c(u(t)),\ y_2^a(t) = y_2^c(u(t)). \tag{25}$$

The acceleration/deceleration speed $v^a(t)$ is calculated as $\dot{u}(t)v^c(t)$, where $v^c(t)$ is the speed of the constant-speed trajectory. The derivative $\dot{u}(t)$ of time transform function can be regarded as the speed scaling function, set as

$$\dot{u}(t) = (1-a)e^{-t/\gamma} + a \tag{26}$$

where a is the speed scaling parameter and γ time constant. If $a > 1$, the velocity $v^a(t)$ is increased; if $a < 1$, $v^a(t)$ is decreased. By integrating $\dot{u}(t)$ with $u(0) = 0$, the time transform function $u(t)$ can be obtained as

$$u(t) = \gamma(1-a)(1-e^{-t/\gamma}) + at. \tag{27}$$

Furthermore, the derivatives of the acceleration/deceleration trajectory are calculated as

$$\begin{aligned}
\dot{y}_j^a &= y_j^{c'}\dot{u} \\
\ddot{y}_j^a &= y_j^{c''}\dot{u}^2 + y_j^{c'}\ddot{u} \\
\dddot{y}_j^a &= y_j^{c'''}\dot{u}^3 + 3y_j^{c''}\dot{u}\ddot{u} + y_j^{c'}\dddot{u},\ j = 1, 2.
\end{aligned} \tag{28}$$

Equation (28), along with (19) and (20), can be used in planning an acceleration/deceleration trajectory with passive control gain K.

V. Controller Design

As the robot should provide proper passive control gain K according to the user's walking intention, detected with the force sensor, the ratio of the control force $K|\dot{\theta}|/r$ to user-applied force F is an important factor in controller design. For the two wheel torques to be checked for passivity, the smaller one is selected for calculating the passive parameter. We, thus, define a dimensionless passive control index p as

$$p = \min\left(\frac{K_r|\dot{\theta}_r|}{Fr}, \frac{K_l|\dot{\theta}_l|}{Fr}\right). \tag{29}$$

The index p is calculated when the user-applied force F is positive. The larger the p is, the higher brake torque is used in passive mode. When p is positive but small, the control gain K is small. Under this situation, the robot is suggested to increase the brake torques and the deceleration trajectory planning is used. On the other hand, when p is large and the speed is small, too much energy is dissipated. The brake torque should be decreased and the acceleration trajectory planning is used. Furthermore, if p and the walking speed are appropriate, the robot can guide the user for stable walking with the constant speed trajectory. Based on the aforementioned discussion, an algorithm for determining proper brake gain is developed. Given the current robot state, user-applied force F, walking speed $\bar{v}$, angular speed $\bar{\omega}$, tolerance force F_{tol}, tolerance speed v_{tol}, the minimum passive index value $p_{\min}$, and the maximum passive index value $p_{\max}$, the brake gain K can be efficiently selected using the following algorithm.

Algorithm for Brake Gain *K* Selection

Choose constants $\beta \in (0.5, 1)$, $\rho_d \in (0, 1)$, and $\rho_i > 1$;
If $F < F_{tol}$ or $v < v_{tol}$, then select K=0;
Else
  Calculate p, v, ω using constant speed trajectory $(y_1^c(t), y_2^c(t))$;
  If $p < p_{min}$ or $v > \bar{v}$ or $\omega > \bar{\omega}$, then robot decreases speed;
    set n=0;
    while $p < p_{min}$ or $v > \bar{v}$ or $\omega > \bar{\omega}$
      n=n+1;
      calculate p, v, ω using deceleration trajectory $(y_1^a(t), y_2^a(t))$ with $a = \rho_d^n$;
    end
    calculate K using deceleration trajectory $(y_1^a(t), y_2^a(t))$ with $a = \rho_d^n$;
  Else if $p > p_{max}$ and $v < \beta\bar{v}$ and $\omega < \beta\bar{\omega}$, then robot increases speed;
    set n=0;
    while $p > p_{\max}$ and $v < \beta\bar{v}$ and $\omega < \beta\bar{\omega}$,
      n=n+1;
      calculate p, v, ω using acceleration trajectory $(y_1^a(t), y_2^a(t))$ with $a = \rho_i^n$;
    end
    calculate K using acceleration trajectory $(y_1^a(t), y_2^a(t))$ with $a = \rho_i^n$;
  Else robot keeps speed using constant speed trajectory $(y_1^c(t), y_2^c(t))$ and calculate K;
End

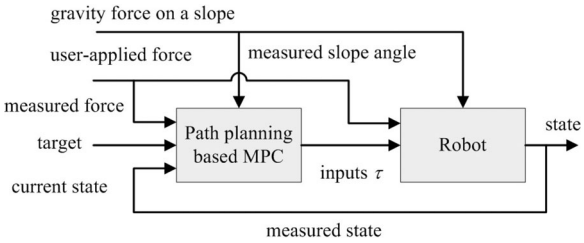


Fig. 4. Proposed model predictive control scheme for robot walking helper.

In the algorithm, the control gain K is determined according to the value of the force sensor. The force sensor noise affects this value, and consequently the accuracy of the guidance. To enhance the robustness, the proposed MPC, discussed in next section, updates the control gain K iteratively according to the current sensor information. Therefore, unless there is persistent large noise, the MPC should be able to deal with sensor noise. Later, the simulation results also verify this.

VI. Model Predictive Control for Slope Guidance

Fig. 4 shows the proposed MPC scheme [22]–[24] for robot walking helper, with which a feasible trajectory and the torques are predicted for the period t_f, given the current waypoint, current measured states, slope angle, and user-applied forces. To achieve real-time control, we do not use a cost function to search for the optimal solution, but only set a constraint of nonnegative K to find a feasible solution. Predictive inputs are then used for controlling the robot when pushed by the user. Note that the slope model is required in MPC, and a fitted slope model $z(y)$ is used. Given the current position (y_{c1}, z_{c1}), the target position (y_{c2}, z_{c2}), the measured current slope angle α_1, and the target slope angle α_2, a predictive slope function $z(y)$ are then fitted with the following form:

$$z(y) = d_0 + d_1(y - y_{c1}) + d_2(y - y_{c1})^2 + d_3(y - y_{c1})^3. \tag{30}$$

With the boundary conditions $z(y_{c1}) = z_{c1}$, $z'(y_{c1}) = \tan(\alpha_1)$, $z(y_{c2}) = z_{c2}$, and $z'(y_{c2}) = \tan(\alpha_2)$, the coefficients d_i ($i = 0,\ldots,3$) are solved as

$$\begin{aligned} d_0 &= z_1, d_1 = \tan\alpha_1 \\ d_2 &= 3(z_{c2} - z_{c1})(y_{c2} - y_{c1})^{-2} \\ &\quad - (2\tan\alpha_1 + \tan\alpha_2)(y_{c2} - y_{c1})^{-1} \\ d_3 &= -2(z_{c2} - z_{c1})(y_{c2} - y_{c1})^{-3} \\ &\quad + (\tan\alpha_1 + \tan\alpha_2)(y_{c2} - y_{c1})^{-2}. \end{aligned} \tag{31}$$

The MPC is based on a constant value of current measured slope angle and user-applied force, while in reality the slope angle and user-applied force are time varying and may be with noise. Therefore, the MPC only uses the predictive inputs to control the robot over a period of time Δt. At the next time step, MPC needs to read the states, slope angle, and user-applied force, and generate a new trajectory. The procedure will be repeated until the stop criterion is satisfied. Meanwhile, the robot guides the user walking through waypoints consecutively. The slope guidance algorithm with MPC is developed and stated as follows.

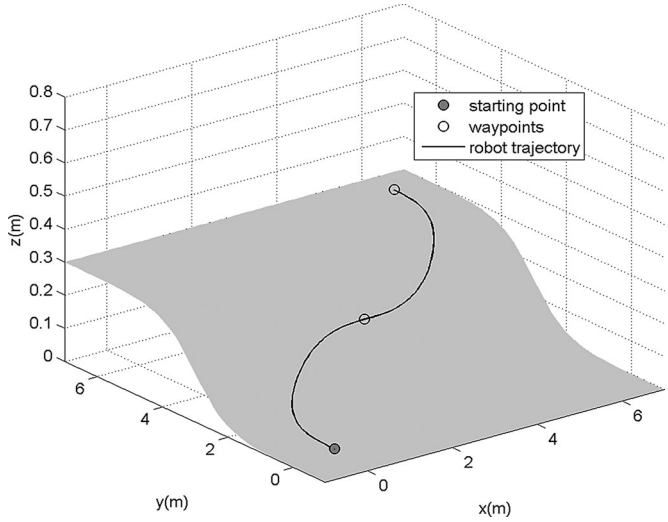


Fig. 5. Robot trajectory on a slope in S-shaped simulation.

Slope Guidance Algorithm With MPC

Step 1: Specify the initial and target points.

Step 2: Generate a set of sequential waypoints (N) with static path planning.

Step 3: Execute model predictive control and set the first waypoint as current waypoint, $n = 1$.

Step 4: Perform robot control algorithm to obtain a brake gain K.

Step 5: Use the predictive gain K to control the robot over a period of time Δt.

Step 6: Measure the robot state and check the stop criterion.

If the robot does not reach the current waypoint, go to step 4;

else if $n < N$, set the next waypoint as the current waypoint, $n = n + 1$, go to step 4;

otherwise, stop.

VII. Simulations

To evaluate the performance of the proposed approach, it was applied to a robot walker for guidance. The parameter values of the robot were set to be $m = 30$ kg, $I_{11} = 2.61$ kg·m^2, $I_{22} = 2.61$ kg·m^2, $I_{33} = 5.22$ kg·m^2, $I_{13} = 1.74$ kg·m^2, $r = 0.0605$ m, and $b = 0.295$ m. The user-applied force F was assumed to follow the function $10(1 + 0.1\sin(\pi t/2))$ N and the time step Δt was set as 0.25 s. The parameter values for the passive controller were set to be $\bar{v} = 0.3$ m/s, $\bar{\omega} = 0.3$ rd/s, $F_{\text{tol}} = 0.1$ N, $v_{\text{tol}} = 0.03$ m/s, $p_{\min} = 0.1$, $p_{\max} = 0.3$, $\beta = 0.95$, $\rho_d = 0.98$, and $\rho_i = 1.02$.

Two sets of simulations, S-shaped guidance and turn-around guidance on a slope, were performed with slope height z assumed to be a function of y as $z(y) = 0.3/(1 + e^{-2(y-3)})$ m. Figs. 5–7 show the simulation results of S-shaped guidance with the start and end points (x_c, y_c, θ) set to (0 m, 0 m, π/2 rd) and (6 m, 6 m, π/2 rd), respectively. The turn-around guidance is

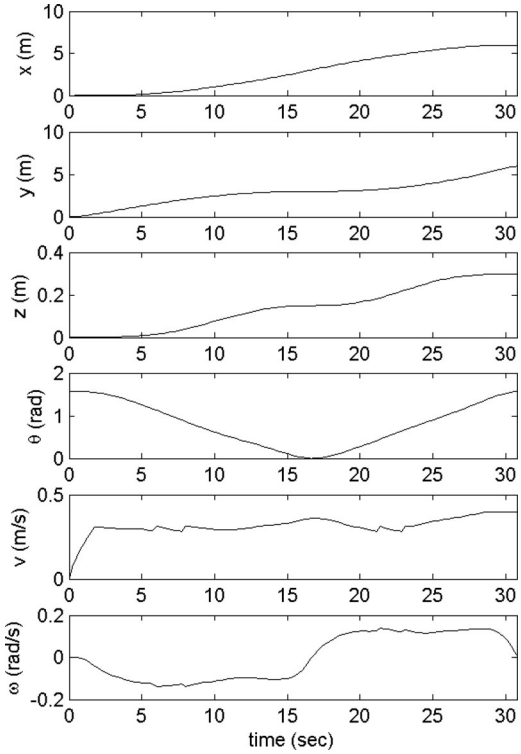


Fig. 6. State variables x, y, z, θ, v, and ω of the robot in S-shaped simulation.

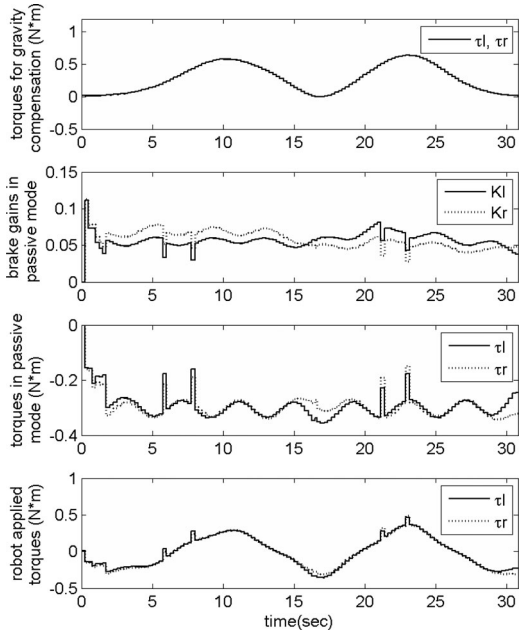


Fig. 7. Torques and brake gains of the robot in S-shaped simulation.

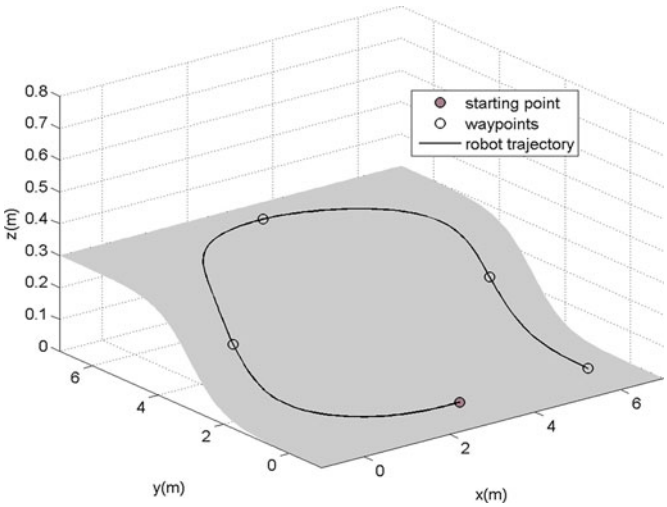


Fig. 8. Robot trajectory on a slope in turn-around simulation.

with the start and end points (3 m, 0 m, π rd) and (6 m, 0 m, $-\pi/2$ rd), and the simulation results are shown in Figs. 8–10. Fig. 5 shows the robot trajectory on the slope for the S-shaped guidance. It was observed that two waypoints were generated. The robot trajectory accurately passed through the waypoints, while approaching the target gradually. Fig. 8 shows the robot trajectory on the slope for the turn-around guidance. The simulation result shows that the proposed scheme generated four waypoints and smoothly guided the user to turn around on the slope. The trajectories of x, y, z, θ, v, ω of two guidance simulations are shown in Figs. 6 and 9, respectively. The (x_c, y_c, θ) errors of S-shaped guidance and turn-around guidance between the desired and actual end points are (9.22e-6 m, 8.68e-5 m, 5.15e-4 rd) and (1.72e-6 m, 3.83e-4 m, 9.6e-5 rd), respectively. We also observed that the mean walking speed and maximum turning speed of S-shape were about 0.315 m/s and 0.139 rd/s, respectively. Meanwhile, the user-supplied energy was computed to be 185.4 J when the walking helper was under the passive mode, and it became 97.59 J when the active control was added, leading to effective energy saving. In turn-around guidance, the mean walking speed and maximum turning speed were about 0.375 m/s and 0.223 rd/s, respectively. Since the turn-around guidance includes downhill path, larger speed was obtained. Figs. 7 and 10 show the values of the robot-applied torques τ_r, τ_l for two kinds of guidance, respectively, which included active torques for gravity compensation and passive torques for walking guidance. The active torques varied with slope angles. When the user walks uphill (downhill), the active torques were all positive (negative). The passive torques were all negative and also showed oscillations to compensate for the oscillating user-applied force for stable walking. The mean time needed for trajectory planning in MPC was only about 0.2 ms with 1.66 GHz CPU and 1 GB memory. The simulation results show that the robot walker with the proposed guidance scheme helped the user to walk on the slope in real time, and guided the user to the target stably and accurately.

To demonstrate the robustness of the MPC, the S-shaped simulations with force sensor noise were performed. The force noise was randomly generated in each time step, with its amount set to be within 25% user-applied force. For several simulations, we observed that the trajectories with noise were slightly different from the trajectory without noise (see Fig. 5), but they still passed through the waypoints with small deviations, demonstrating the robustness of the MPC.

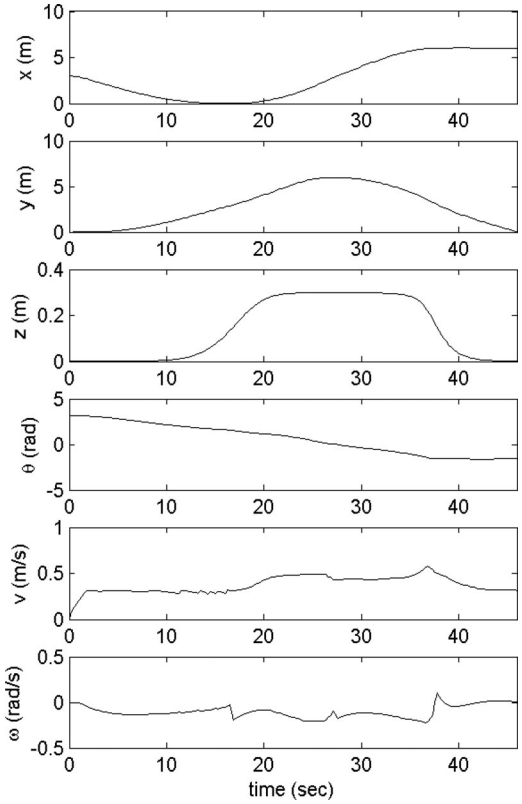


Fig. 9. State variables x, y, z, θ, v, and ω of the robot in turn-around simulation.

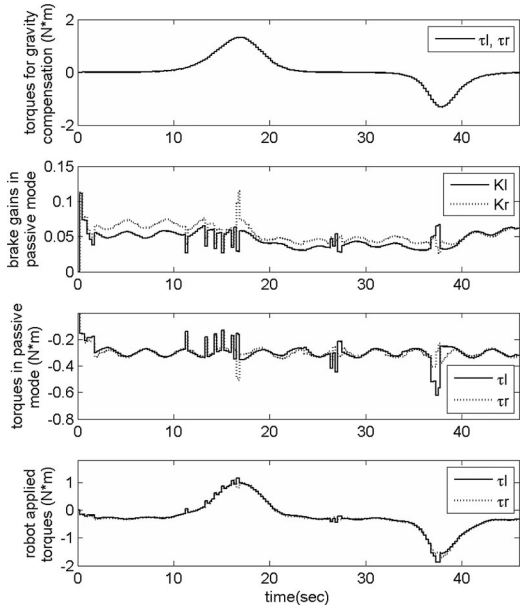


Fig. 10. Torques and brake gains of the robot in turn-around simulation.

VIII. Experiments

For demonstration, we performed the straight-line and S-shaped experiments on a slope. Fig. 11 shows the experimental scenario. In the straight-line experiment, the user repeated twice walking straight uphill. One was with robot control torques and the other without. Fig. 12 shows that the user-applied forces with robot control were smaller than without robot control. This demonstrates that the proposed control method can effectively help the user walking uphill. For S-shaped experiment, the start and end points (x_c, y_c, θ) were set to be (0 m, 0 m, π/2 rd) and (4 m, 5 m, π/2 rd), respectively. Fig. 13 shows the robot trajectory on the slope during S-shaped guidance, which indicates that the robot guided the user to be near the waypoints with accurate orientations. The position error between the final robot position and the target is about 0.45 m. Fig. 14 shows the trajectories of x, y, z, θ, v, ω. The human walking speed was with a mean value of 0.19 m/s and maintained within 0.36 m/s. The turning speed was within 0.34 rd/s. Fig. 15 shows the human-applied force (with a mean at about 7.068 N), the brake gains K, and the robot-applied torques τ_r, τ_l. Furthermore, the right wheel torques were initially smaller than left wheel torques for right turn. When the robot was approaching the end point, the right wheel torque was larger than left wheel torque for left turn. We also invited additional users to conduct the S-shaped experiments. The resultant trajectories were slightly different from that in Fig. 13, while the robot guided the users to be near the

Fig. 11. Experimental scenario for guidance on a slope.

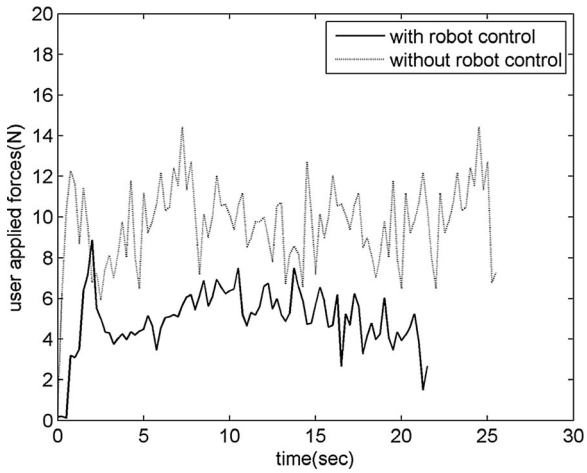


Fig. 12. User-applied forces in walking uphill experiment.

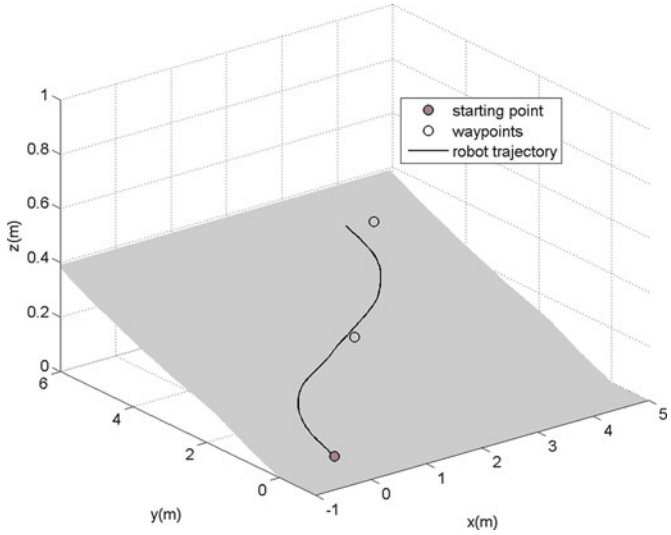


Fig. 13. Robot trajectory on a slope in S-shaped experiment.

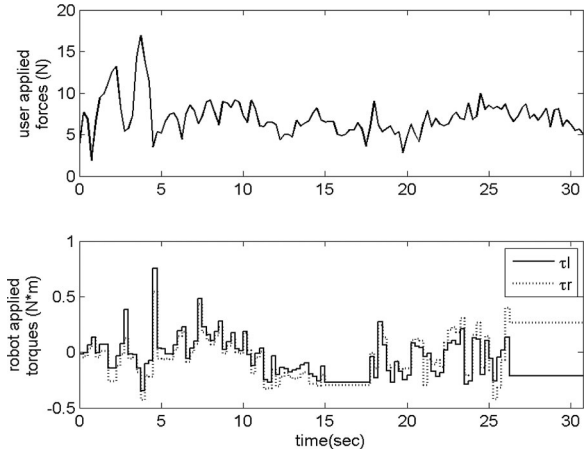


Fig. 15. Human-applied forces and robot torques in S-shaped experiment.

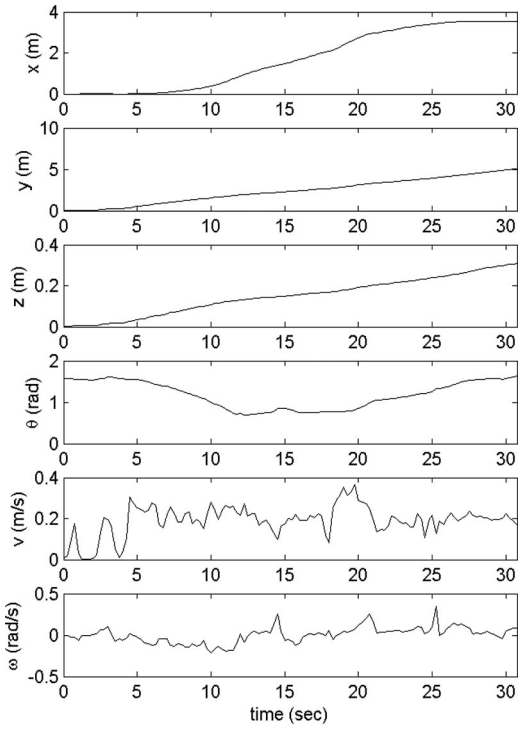


Fig. 14. State variables x, y, z, θ, v, and ω of the robot in S-shaped experiment.

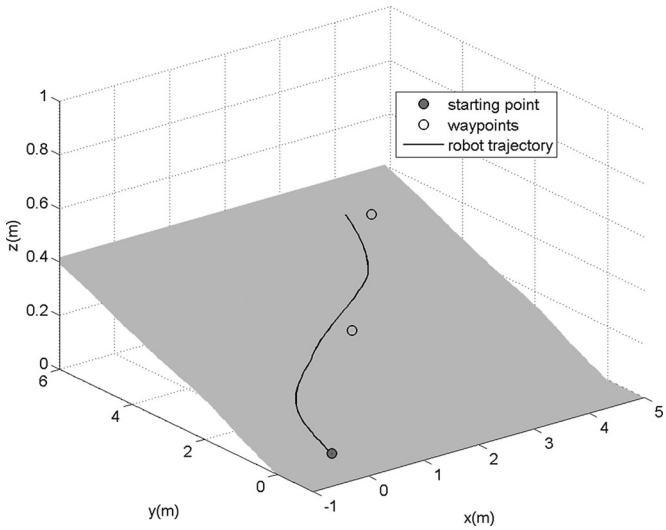


Fig. 16. Robot trajectory generated by another user in S-shaped experiment.

waypoints with accurate orientations successfully, as Fig. 16 shows one of the trajectories, which is with 0.4 m position error between the final robot position and the target. For the errors present between the actual and desired robot positions in Figs. 13 and 16, we consider they might result from wheel slipping. In summary, the experiment results demonstrate that the proposed scheme can help the user to walk stably on slope and guide the user to reach the target.

IX. Conclusion

In this paper, we have presented a novel method to choose time-varying control gains for a robot walking helper. The dynamic model of the vehicle on a slope with passive/active control law was first established. The trajectories of the gains were developed by using approaches from the theory of differential flatness. Accordingly, feasible trajectories were found by using the proposed passive/active method. Furthermore, MPC was also implemented for the robot walking helper since user-applied forces and slope angles are not known *a priori*. The simulation and experimental results show that the robot walking helper with MPC can accurately guide the user to a goal on a slope, demonstrating the effectiveness of the proposed control scheme. In future works, we will continue working on the reduction of power consumption for the robot walker helper and also improve its guiding performance.

REFERENCES

[1] H. Yu, M. Spenko, and S. Dubowsky, "An adaptive shared control system for an intelligent mobility aid for the elderly," *Auton. Robots*, vol. 15, no. 1, pp. 53–66, 2003.

[2] O. Chuy, Y. Hirata, Z. Wang, and K. Kosuge, "A control approach based on passive behavior to enhance user interaction," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 899–908, Oct. 2007.

[3] O. Chuy, Y. Hirata, and K. Kosuge, "A new control approach for a robotic walking support system in adapting user characteristics," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 36, no. 6, pp. 725–733, Nov. 2006.

[4] A. J. Rentschler, R. A. Cooper, B. Blaschm, and M. L. Boninger, "Intelligent walkers for the elderly: Performance and safety testing of VA-PAMAID robotic walker," *J. Rehab. Res. Dev.*, vol. 40, no. 5, pp. 423–432, 2003.

[5] G. Wasson, P. Sheth, M. Alwan, K. Granata, A. Ledoux, and C. Huang, "User intent in a shared control framework for pedestrian mobility aids," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2003, pp. 2962–2967.

[6] A. M. Sabatini, V. Genovese, and E. Pacchierotti, "A mobility aid for the support to walking and object transportation of people with motor impairments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2002, pp. 1349–1354.

[7] M. Spenko, H. Yu, and S. Dubowsky, "Robotic personal aids for mobility and monitoring for the elderly," *IEEE Trans. Neural Syst. Rehab. Eng.*, vol. 14, no. 3, pp. 344–351, Sep. 2006.

[8] A. Morris, R. Donamukkala, A. Kapuria, A. Steinfeld, J. T. Matthews, J. Dunbar-Jacob, and S. Thrun, "A robotic walker that provides guidance," in *Proc. IEEE Int. Conf. Robot. Autom.*, Sep. 2003, pp. 25–30.

[9] B. Graf, "An adaptive guidance system for robotic walking aids," *J. Comput. Inf. Technol.*, vol. 17, no. 1, pp. 109–120, 2009.

[10] Y. Hirata, A. Hara, and K. Kosuge, "Motion control of passive intelligent walker using servo brakes," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 981–990, Oct. 2007.

[11] N. Nejatbakhsh and K. Kosuge, "Optimal guidance by omnidirectional passive mobility aid system," *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 5786–5791.

[12] J. C. Ryu, K. Pathak, and S. K. Agrawal, "Control of a passive mobility assist robot," *J. Med. Devices*, vol. 2, no. 1, p. 011002, 2008.

[13] K. Wakita, J. Huang, P. Di, K. Sekiyama, and T. Fukuda, "Human-walking-intention-based motion control of an omnidirectional-type cane robot," *IEEE/ASME Trans. Mechatronics*, to be published.

[14] Y. C. Huang, H. P. Young, C. H. Ko, and K. Y. Young, "Human intention recognition for robot walking helper using ANFIS," in *Proc. Asian Cont. Conf.*, May 2011, pp. 311–316.

[15] C. H. Ko and S. K. Agrawal, "Walk-assist robot: A novel approach to gain selection of a braking controller using differential flatness," in *Proc. American Cont. Conf.*, Jun. 2010, pp. 2799–2804.

[16] C. H. Ko and S. K. Agrawal, "Control and path planning of a walk-assist robot using differential flatness," in *Proc. IEEE/RSJ Int. Conf. Intell. Rob. Syst.*, Oct. 2010, pp. 6016–6021.

[17] H. Sira-Ramirez and S. K. Agrawal, *Differential Flat Systems*, 1st ed. New York: Marcel Dekker, 2004.

[18] C. P. Tang, P. T. Miller, V. N. Krovi, J. C. Ryu, and S. K. Agrawal, "Differential-flatness-based planning and control of a wheeled mobile manipulator—Theory and experiment," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 4, pp. 768–773, Aug. 2011.

[19] V. Sangwan and S. K. Agrawal, "Differentially flat design of bipeds ensuring limit cycles," *IEEE/ASME Trans. Mechatronics*, vol. 14, no. 6, pp. 647–657, Dec. 2009.

[20] S. M. LaValle, *Planning Algorithm*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[21] S. K. Agrawal and T. Veeraklaew, "A higher-order method for dynamic optimization of a class of linear systems," *J. Dyn. Syst., Meas. Control*, vol. 118, pp. 786–791, Dec. 1996.

[22] V. Duchaine, S. Bouchard, and C. M. Gosselin, "Computationally efficient predictive robot control," *IEEE/ASME Trans. Mechatronics*, vol. 12, no. 5, pp. 570–578, Oct. 2007.

[23] A. Grancharova and T. A. Johansen, "Design and comparison of explicit model predictive controllers for an electropneumatic clutch actuator using on/off valves," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 4, pp. 665–673, Aug. 2011.

[24] C. Y. Lin and Y. C. Liu, "Precision tracking control and constraint handling of mechatronic servo systems using model predictive control," *IEEE/ASME Trans. Mechatronics*, vol. 17, no. 4, pp. 593–605, Aug. 2012.

Chun-Hsu Ko was born in Tainan, Taiwan, in 1967. He received the M.S. degree in power mechanical engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1991, and the Ph.D. degree in electrical and control engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2003.

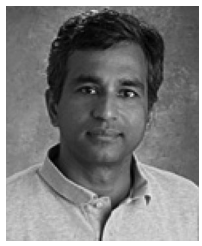
From 1994 to 1998, he was with the Industrial Technology Research Institute, Hsinchu, Taiwan, as an Associate Researcher. He is currently an Associate Professor in the Department of Electrical Engineering, I-Shou University, Kaohsiung, Taiwan. His research interests include robot control, robot walking helpers, and optimization.

Kuu-Young Young was born in Kaohsiung, Taiwan, in 1961. He received the B.S. degree in electrical engineering from National Taiwan University, Hsinchu, Taiwan, in 1983, and the M.S. and Ph.D. degrees in electrical engineering from Northwestern University, Evanston, IL, in 1987 and 1990, respectively.

Between 1983 and 1985, he was an Electronic Officer in the Taiwan Navy. Since 1990, he has been with the Department of Electrical Engineering, National Chiao-Tung University, Hsinchu, Taiwan, where he is currently a Professor. He served as the Chairman of the department from 2003 to 2006, and as the Associate Dean of the Electrical and Computer Engineering College, National Chiao Tung University, from 2007 to 2010. His research interests include robot compliance control, robot learning control, robot calibration and path planning, teleoperation, robot walking helpers, and science, technology, and society.

Yi-Che Huang was born in Pingtung, Taiwan, on January 25, 1981. He received the B.S. and M.S. degrees in electrical engineering from Tamkang University, Taipei, Taiwan, in 2004 and 2007, respectively. He is currently working toward the Ph.D. degree in the Department of Electrical Engineering, National Chiao Tung University, Hsinchu, Taiwan.

His main research interests include robot walking helpers, modeling and control of dynamic systems, learning systems, and robot path planning and force control.



Sunil Kumar Agrawal (M'92) received the Ph.D. degree in mechanical engineering from Stanford University, Stanford, CA, in 1990.

He is currently a Professor in the Department of Mechanical Engineering, University of Delaware, Newark. He has authored more than 300 journal and conference papers and two books in the areas of controlled mechanical systems, dynamic optimization, and robotics.

Dr. Agrawal is a Fellow of the American Society of Mechanical Engineers (ASME). He received a Presidential Faculty Fellowship from the White House in 1994, the Bessel Prize (Germany) in 2003, and the Humboldt U.S. Senior Scientist Award in 2007. He has been an editorial board member of several journals published by the ASME and IEEE.