To appear in the International Journal of Robotics Research (IJRR)

The Ballbot: An Omnidirectional Balancing Mobile Robot

Umashankar Nagarajan, George Kantor, and Ralph Hollis

***Abstract*—The ballbot is a human-sized dynamically stable mobile robot that balances on a single ball. Unlike statically stable mobile robots, the ballbot is tall and narrow with a high center of gravity and a small footprint. Moreover, its dynamic stability enables it to be physically interactive. These characteristics make it better suited to navigate and interact in cluttered human environments. This paper presents the evolved hardware design of the ballbot with a four-wheel inverse mouse-ball drive to actuate the ball, and a yaw drive mechanism that enables unlimited rotation about its vertical axis. The ballbot also has a triad of legs that provide static stability when powered down. This paper presents a detailed description of the ballbot's control architecture, and it presents several experimental results that demonstrate its balancing and locomotion capabilities. This paper also presents a trajectory planning algorithm that plans for body lean motions, which when tracked result in the desired rest-to-rest motions of the robot. Finally, the paper illustrates some interesting human-robot physical interaction behaviors that can be achieved as a result of the ballbot's dynamic stability.**

I. Introduction

There are a number of challenges in building wheeled mobile robots that will operate and interact in human environments. One of the fundamental challenges is in locomotion and navigation. Traditionally, wheeled robots have been statically stable, while recently, more groups have been interested in developing dynamically stable wheeled robots that actively balance. The work presented in this paper aims at exploring the feasibility of developing dynamically stable mobile robots that are human-sized, dynamically agile, slender enough to easily maneuver cluttered environments and readily yield when pushed. This paper presents the ballbot, shown in Fig. 1, which is the first successful single-wheeled, omnidirectional balancing mobile robot to the best of our knowledge (Lauwers et al. 2005). The ballbot was developed as a simple test bed to study locomotion and physical interaction characteristics of balancing mobile robots in human environments.

U. Nagarajan, G. Kantor and R. Hollis are with The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. email: umashankar@cmu.edu, kantor@ri.cmu.edu, rhollis@cs.cmu.edu.

Balancing robots have underlying dynamic properties that can be exploited in order to carry out agile, efficient motion. Balancing robots can be tall enough to interact with people at eye level, narrow enough to easily negotiate cluttered environments, and they can move with speed and agility comparable to that of humans. They are also capable of safe and gentle physical interaction.

To understand why balancing robots offer a revolutionary departure from traditional statically stable mobile robots one must consider that statically stable (non-balancing) robots of the traditional variety generally have multiple wheels or treads. When at rest, the gravitational vector from the robot's center of mass must pass through the base of support. When the robot accelerates, the vector sum of its acceleration and gravity vectors must be encompassed by the base to avoid tipping. If the statically stable robot is tall, has heavy arms, or carries a heavy payload it becomes much more difficult to avoid potentially disastrous tipping.

In 2005, we introduced the ballbot, a dynamically stable mobile robot moving on a single spherical wheel (Lauwers et al. 2005), and it was popularized in (Hollis 2006; Lauwers et al. 2006). Unlike its two-wheeled counterparts, the single spherical wheel provides omnidirectional motion making the ballbot more suitable for navigation in human environments with constrained spaces. Moreover, the ballbot is tall and narrow, which enhances its ability to interact with human environments.

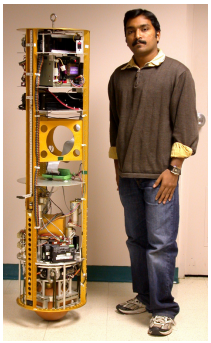


Fig. 1. Ballbot balancing.

In (Hollis 2006; Lauwers et al. 2006, 2005), we presented the first version of the ballbot with preliminary results of its balancing behavior. It used an inverse mouse-ball drive with only two active motors to drive the ball, which resulted in an undesirable "hopping motion." Although its controller was capable of balancing the robot reasonably well, it was not robust to large disturbances. Moreover, it produced jerky motions while tracking desired ball motions.

A. Contributions

This paper exclusively deals with the design and operation of the ballbot, which is the first successful ball balancing robot to the best of our knowledge. The ballbot was designed as a simple test bed to study locomotion and interaction characteristics of balancing mobile robots in human environments. This paper demonstrates the feasibility of developing such simple balancing platforms, and highlights the advantages of balancing robots. It also discusses the drawbacks and the challenges that need to be addressed in developing balancing mobile robots.

This paper presents an evolved version of the ballbot with a four-motor inverse mouse-ball drive, wherein all four rollers driving the ball are active. This arrangement overcame the undesirable "hopping motion" that was observed in (Hollis 2006; Lauwers et al. 2006, 2005). This evolved version included a yaw drive mechanism that enabled the ballbot to rotate about its vertical axis. This paper presents a detailed description of the entire improved control architecture of the ballbot, including balancing control, stationkeeping and velocity control, leg control, and yaw control. It presents controllers that enable the ballbot to successfully transition from a statically stable state (SSS) to a dynamically stable state (DSS) and vice versa. Unlike in (Hollis 2006; Lauwers et al. 2006, 2005), the balancing controller presented here enables the ballbot to be robust to large disturbances, including shoves and kicks, and also collisions against walls and tables.

This paper also presents an offline trajectory planning algorithm that plans body lean trajectories so as to move the robot to a desired position on the floor while satisfying the dynamic constraints associated with balancing. The trajectory planner and the control architecture enable the ballbot to achieve good tracking of desired ball motions, and do not produce jerky motions as was observed in (Hollis 2006; Lauwers et al. 2006, 2005). This paper presents experimental results of the ballbot achieving smooth rest-to-rest motions on the floor. Finally, the paper demonstrates how properties of dynamic stability naturally provide for safe, useful human-robot physical interactions with balancing robots like the ballbot. This paper is an improved version of the work presented in (Nagarajan et al. 2009a,b,c).

II. Related Work

A. Balancing Mobile Robots

The idea of balancing with wheeled robots has been around for some time. Of course the cart-pendulum system has long been considered as a canonical problem in the controls literature, but one of the first real two-wheeled inverted-pendulum type mobile robot was designed and controlled in (Ha and Yuta 1997). Two-wheeled balancing robots became popular especially after the introduction of the Segway RMP (Nguyen et al. 2004). Dean Kamen introduced iBot (iBOT 2003), a two-wheeled balancing wheelchair, and Anybots introduced a balancing tele-presence robot on two wheels (Anybots 2010). Rod Grupen and his group introduced uBot (Deegan et al. 2006), a two-wheeled dynamic mobile manipulation platform and showed that balancing robots can be effective mobile manipulators (Deegan et al. 2008). Mike Stilman introduced Golem Krang (Stilman et al. 2010), a two-wheeled balancing mobile manipulator that can autonomously stand and sit. Such two-wheeled balancing platforms have dominated the field of dynamically stable mobile robots, but they have kinematic constraints (*i.e.*, wheels that cannot slip sideways) that restrict their direction of motion. Moreover, they balance in only one vertical plane and are statically stable in the other. On the contrary, single-wheeled balancing mobile robots like the ballbot (Hollis 2006) balance in both the vertical planes and are omnidirectional in motion.

B. Single-wheeled Omnidirectional Balancing Robots

Since the introduction of the ballbot at Carnegie Mellon University in 2005, several other groups around the world have also begun to explore and build single-wheel omnidirectional balancers (Havasi 2005; Kumagai and Ochiai 2008, 2009; Rezero 2010). In 2005, Laszlo Havasi from Hungary independently developed another ball balancing robot called *ERROSphere* (Havasi 2005). However, the robot did not balance reliably and there was no further work presented. In 2008, Masaaki Kumagai developed a ball balancing robot called *BallIP* (Kumagai and Ochiai 2008) at Tohoku Gakuin University, and his group demonstrated its capability to carry loads and achieve cooperative transportation (Kumagai and Ochiai 2009). In 2010, a group of mechanical engineering students at ETH Zurich developed a ball balancing robot called *Rezero* (Rezero 2010).

The fundamental difference between the ballbot (Hollis 2006), and the other successful ball balancing robots (Kumagai and Ochiai 2008; Rezero 2010) is in the actuator mechanism for the ball. The ballbot (Hollis 2006) uses an inverse mouse-ball drive with four motors (see Sec. III-A), whereas both BallIP (Kumagai and Ochiai 2008) and Rezero (Rezero 2010) use omniwheels with three motors to drive the ball. Ordinary omniwheels produce unsteady rolling and hence, a complex and expensive omniwheel with continuous circumferential contact line was used for BallIP (Kumagai and Ochiai 2008). The same omniwheel design was adopted later in Rezero. Unlike the ballbot that needs a slip-ring assembly and a separate motor for unlimited yaw rotation (see Sec. III-B), the omniwheel balancers like BallIP and Rezero can move and yaw using just the three motors. A detailed analysis and comparison of the inverse mouse-ball drive and tri-omniwheel drive mechanisms is yet to be done. In terms of control, BallIP uses a velocity control strategy on their omniwheels instead of torque control. This is due to two reasons, one is that they use stepper motors to actuate the omniwheels, and the other is that the velocity control strategy better handles the case where the omniwheel loses contact with the ball.

C. Underactuated Systems

Balancing robots are underactuated systems, $i.e.$, systems with fewer independent control inputs than the number of degrees of freedom (Spong 1998). Underactuated balancing systems have constraints on their dynamics that restrict the family of configuration trajectories that they can follow. These constraints are called second-order nonholonomic (non-integrable) constraints (Ray 1966) or dynamic constraints. The past few decades have seen a lot of interest in trajectory planning and control for such systems.

Underactuated systems with dynamic constraints have been approached from the controls perspective ($e.g.$, acrobot swing-up (Spong 1995)) as well as from the planning perspective ($e.g.$, airship path planning (Kim and Ostrowski 2003)). A detailed analysis of underactuated manipulators with passive joints is presented in (Oriolo and Nakamura 1991). Olfati-Saber presented explicit cascade normal forms for different classes of underactuated mechanical systems and also presented partial feedback linearization techniques for controlling them (Olfati-Saber 2001). In (Rosas et al. 2001), the trajectory planning problem of an underactuated planar 2R manipulator is solved using offline planned trajectories, which are constructed with smooth sinusoids. In (Rosas et al. 2002), a class of parametric trajectories is proposed for the actuated joint of the 2R underactuated manipulator with zero gravity in order to achieve desired configurations of the system. In this paper, we use a similar class of parametric trajectories for the unactuated joint of the ballbot as will be described in Sec. VI.

III. The Ballbot Platform

The ballbot, shown in Fig. 1, is a reconfigurable cylindrical platform resting atop a ball. The body is 1.41 m tall, weighs 55 kg and has an outer diameter of 0.37 m. The ballbot was intentionally built to be of human size so that it can interact with human environments similar to the way humans do. Three aluminium channels, held together by circular decks, define the structure of the ballbot's body. Three retractable legs are attached to the lower part of these channels and are deployed to provide static stability when powered down. The ball is a 0.185 m diameter hollow aluminium sphere coated with a 13.5 mm thick layer of urethane. The body hosts a 48V lead acid battery, its charger and a single board computer on its top decks. An inertial measurement unit (IMU) sits on top of the ball drive unit and provides Kalman-filtered body angles and rates w.r.t. gravity.

A. Inverse Mouse-ball Drive

The ballbot uses an inverse mouse-ball drive to actuate the ball. Figure 2(a) shows the ball drive mechanism, which is essentially the inverse of a computer mouse. In a traditional computer mouse, the rolling ball drives rollers to produce computer input, whereas here, the rollers drive the ball to produce motion. Our previous version of the inverse mouse-ball drive (IMB) (Lauwers et al. 2006) had a pair of drive and opposing passive rollers for each of the orthogonal motion directions. This setup caused the rollers to produce an upward or downward force on the ball in addition to the torque, which resulted in an undesirable "hopping motion." The present design circumvents this problem by actuating all four rollers with individual DC servomotors that exert pure torques on the ball. The motors have 1024 cpr encoders attached to their shafts that measure ball rotation. A detailed paper describing the IMB is in preparation (Hollis et al.).

B. Yaw Mechanism

A large thin-section bearing attaches the ball drive mechanism to the body and allows yaw rotation. The yaw drive consists of a DC servomotor with planetary gears driving a pulley assembly at the center as shown in Fig. 2(b). The orientation of the body frame with respect to the ball drive unit is given by an absolute encoder

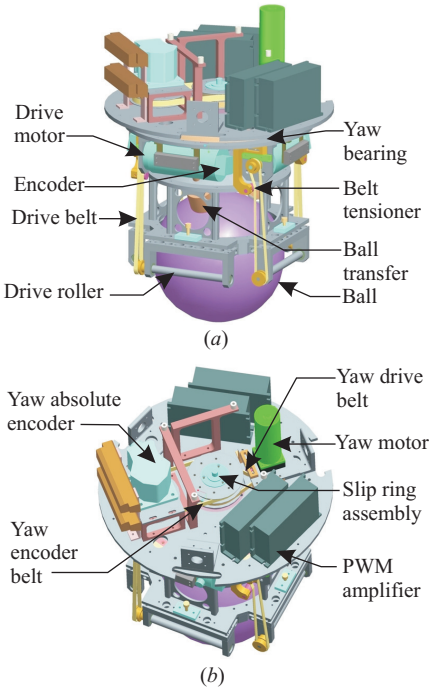


Fig. 2. Inverse mouse-ball drive and yaw drive: (*a*) view showing ball drive mechanism, (*b*) view showing yaw drive mechanism.

attached to the pulley assembly. A slip ring assembly used for drive motor currents and encoder signals permits unlimited yaw rotation.

C. Legs

Each leg is 0.48 m long and is attached to a linear screw drive with a ratio of 1.6 mm/revolution. The legs are independently driven by three DC servomotors with 500 cpr encoders. Each leg has a hoof switch and a ball castor at its tip. The hoof switches are spring loaded and are used to detect contact with the floor, whereas the ball castors allow the legs to slide on a smooth floor.

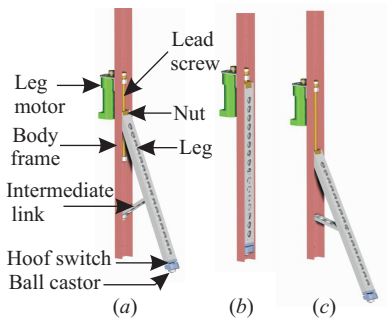


Fig. 3. Leg drive: (*a*) Various components of the leg drive mechanism, (*b*) legs completely retracted, and (c) legs completely deployed.

Different stages of operation of the leg drive mechanism along with its components are shown in Fig. 3.

IV. Dynamic Model

The ballbot is modeled as a rigid cylinder on top of a rigid sphere. A simplified planar model of the ballbot is used for both control and trajectory planning. The planar model assumes that: (i) there is no slip between the spherical wheel and the floor, (ii) the motion in the median sagital plane and the median coronal plane is decoupled, (iii) the equations of motion in these two planes are identical, and (iv) the floor is assumed to be flat and level. With these assumptions, we design two decoupled, independent planar controllers and trajectory planners for the 3D system.

The stabilizing controllers for the ballbot presented in Sec. V are designed as linear controllers. The linearized dynamics of the full 3D ballbot system about the origin is decoupled between the two orthogonal planes of motion, which validates the use of decoupled planar models for linear control design. In the full 3D ballbot model, the coupling terms between the two orthogonal planes of motion contain products of sine of the body lean angles, and hence, for small lean angles, the dynamics of two decoupled planar models well approximate the dynamics of the full 3D model. The trajectory planner presented in Sec. VI uses the nonlinear dynamics of the planar ballbot model to plan body lean motions that achieve desired rest-to-rest motions. In this work, we are interested only in simple planar rest-to-rest motions, and hence, the planar ballbot model is sufficient to capture the dynamics of the system. A detailed presentation of more sophisticated planning approaches that use the 3D ballbot model is in review (Nagarajan and Hollis 2013).

A. Planar Ballbot Model

A ball rolling on a plane has five configurations, two configurations for the ball position and three configurations for the ball orientation. However, in this work, we are interested only in the position of the ball and not in its orientation. Figure 4 shows the planar model of the ballbot used in this work along with this planar configurations. The origin of the world frame is fixed to the initial position of the center of the ball. Since we have assumed a flat and level floor, the horizontal position of the center of the ball x_w matches the horizontal position of the ball's contact point on the floor. The body axis of the robot is given by the line connecting the center of the ball to the center of mass (CM) of the body. The body angle ϕ is defined as the angle between the vertical and the body axis, and it is directly measured by the IMU.

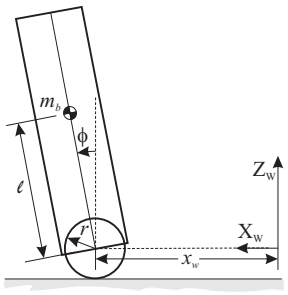


Fig. 4. Planar ballbot model with body angle ϕ and ball position x_w. The ball angular configuration θ is chosen such that $x_w = r_w(\theta+\phi)$.

The angular configuration θ of the ball is chosen such that the horizontal position of the ball x_w w.r.t. the world frame is given by $x_w = r_w(\theta+\phi)$, where r_w is the radius of the ball and the ball configuration $\theta \in [-\infty, \infty]$. There are two advantages in choosing this coordinate for the ball configuration: one is that the ball configuration θ directly corresponds to the encoder readings on the ball motors, and the other is that this coordinate choice allows one to remove the input coupling between the ball and the body from the equations of motion.

The dynamic equations of motion of the planar ballbot model[1], shown in Fig. 4, are derived using Euler-Lagrange equations. The equations of motion can be written in matrix form as follows:

$$M(q)\ddot{q} + C(q,\dot{q}) + G(q) + D(\dot{q}) = \begin{bmatrix} \tau \\ 0 \end{bmatrix}, \quad (1)$$

where $q = [\theta, \phi]^T$ is the generalized coordinate vector, $M(q)$ is the mass/inertia matrix, $C(q,\dot{q})$ is the vector of coriolis and centrifugal forces, $G(q)$ is the vector of gravitational forces, $D(\dot{q})$ is the frictional torque vector and τ is the torque applied between the ball and the body in the direction normal to the plane. The expressions for the above mentioned terms are given below:

$$M(q) = \begin{bmatrix} \alpha & \alpha + \beta\cos\phi \\ \alpha + \beta\cos\phi & \alpha + \gamma + 2\beta\cos\phi \end{bmatrix}, \quad (2)$$

$$C(q,\dot{q}) = \begin{bmatrix} -\beta\sin\phi\dot{\phi}^2 \\ -\beta\sin\phi\dot{\phi}^2 \end{bmatrix}, \quad (3)$$

$$G(q) = \begin{bmatrix} 0 \\ -\frac{\beta g \sin\phi}{r} \end{bmatrix}, \quad (4)$$

$$D(\dot{q}) = \begin{bmatrix} D_c \mathrm{sgn}(\dot{\theta}) + D_v\dot{\theta} \\ 0 \end{bmatrix}, \quad (5)$$

where $\alpha = I_w + (m_w + m_b)r_w^2$, $\beta = m_b r_w \ell_b$, $\gamma = I_b + m_b\ell_b^2$, D_c and D_v are the coulomb friction torque and the viscous damping friction coefficient respectively. Please refer to Table I for the other symbols.

As described in Sec. III-A, a pair of active opposing rollers drive the ball in each of the orthogonal motion directions. For each orthogonal motion direction, the ball torque τ is given by:

$$\tau = \frac{r_w}{r_r}(\tau_{m1} + \tau_{m2}), \quad (6)$$

where r_r is the radius of the roller, r_w is the radius of the ball, and τ_{m1}, τ_{m2} are the torques on the opposing motors. In the current setup, the amplifiers that drive the opposing motors are hardwired to command the same torque, *i.e.*, $\tau_{m1} = \tau_{m2}$.

B. Parameter Estimation

In order to facilitate the mathematical model to better represent the robot dynamics, various offline experiments were conducted to determine the principal system parameters. Compared to (Nagarajan et al. 2009c), this paper presents a more detailed description of the parameter estimation experiments and the results obtained.

1) Inertia Measurement: The moments of inertia of the ballbot's body were determined experimentally using a torsional pendulum setup (Wang et al. 2007). The body was suspended perpendicular to its length about its center of mass using a torsional spring as shown in Fig. 5, and the oscillations were observed after an initial disturbance.

The resulting angular velocity trajectories from the IMU shown in Fig. 6 were used to find the natural frequency of oscillation. The spring constant of the torsional pendulum was obtained by performing the same experiment with an I-beam whose inertia was calculated from its shape and material density. The torsional spring constant K is given by

$$K = I\omega_n^2, \quad (7)$$

where I is the inertia of the suspended object and ω_n is the natural frequency of oscillation. The moment of

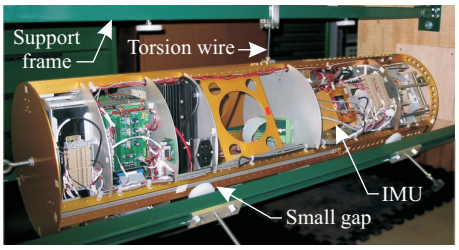


Fig. 5. Torsional pendulum setup with the ballbot suspended perpendicular to its length.

[1]It is to be noted that the model described below uses a coordinate scheme different from the one described in (Lauwers et al. 2006).

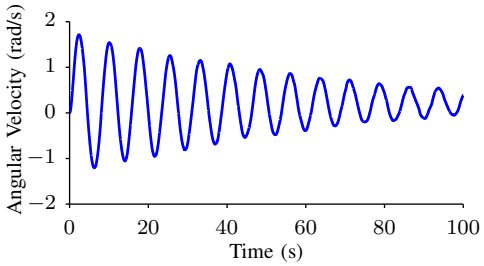


Fig. 6. Damped sinusoidal oscillation used to determine the ballbot's moments of inertia.

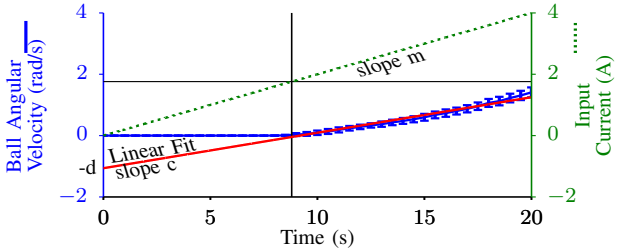


Fig. 8. Ball response to the ramp current inputs to the ball drive motors used for determing coulomb and viscous friction terms.

inertia of the body about its center of mass is given by

$$I_b = I_{I-beam}\frac{\omega_{I-beam}^2}{\omega_b^2}. \tag{8}$$

The moment of inertia about the vertical axis I_b^{zz} was determined by a similar experiment with the body suspended vertically. The measured moment of inertia values are shown in Table I.

2) Friction Modeling: The coulomb friction torque and the viscous friction coefficient were experimentally determined using the setup shown in Fig. 7, where the ballbot stood on a roller ball with its body constrained vertically. A ramp current input of slope m was given to the ball drive motors and the angular velocity of the ball was recorded. The minimum current required to start the ball rolling is called the breakaway current, which when multiplied by the torque constant K_i of the drive unit gives the coulomb friction torque D_c. The experiment was repeated with the current vector at 5° intervals.

After breakaway, the equation of motion of the ball can be written as

$$\begin{aligned} I_w\ddot{\theta} &= \tau(t) - \tau_v - D_c, \quad \dot{\theta} > 0 && (9)\\ &= K_i m t - D_v\dot{\theta} - D_c, && (10)\end{aligned}$$

where τ_v is the viscous friction torque. The plot of ball angular velocity $\dot{\theta}$ vs. time after breakaway can be approximated by a line of constant slope c (Kelly et al. 2000) as shown in Fig. 8. Hence, the ball angular velocity $\dot{\theta}$ and angular acceleration $\ddot{\theta}$ can be written as

$$\begin{aligned} \dot{\theta} &= ct - d, \quad \dot{\theta} > 0 && (11)\\ \ddot{\theta} &= c. && (12)\end{aligned}$$

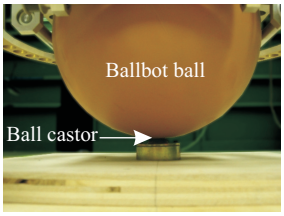
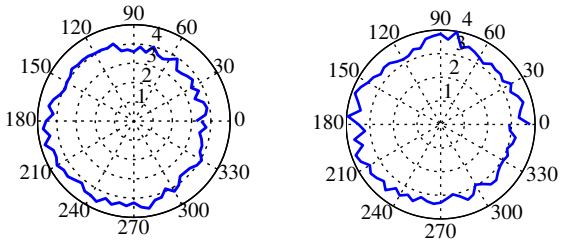


Fig. 7. Ball rolling on the roller during friction tests.



(a) (b)

Fig. 9. Radial plots as functions of drive directions: (a) Coulomb friction torque D_c (N·m); (b) Viscous friction coefficient D_v (N·m·s/rad).

The ball angular velocity $\dot{\theta}$ was measured using the encoders on the ball motors. The viscous friction coefficient D_v can be obtained by solving (10−12), and is given by

$$D_v = \frac{K_i m}{c}. \tag{13}$$

The radial plots of coulomb friction torque and viscous friction coefficient in different drive directions are shown in Fig. 9. Table I presents the average coulomb friction torque and viscous friction coefficient values that are used in simulation.

TABLE I
SYSTEM PARAMETERS

Parameter	Symbol	Value
Ball radius	r_w	0.1058 m
Roller radius	r_r	0.006335 m
Ball mass	m_w	2.44 kg
Ball inertia	I_w	0.0174 kg·m^2
Body center of mass height	ℓ_b	0.69 m
Roll moment of inertia	I_b^{xx}	12.59 kg·m^2
Pitch moment of inertia	I_b^{yy}	12.48 kg·m^2
Yaw moment of inertia	I_b^{zz}	0.66 kg·m^2
Body mass	m_b	51.66 kg
Coulomb friction torque	D_c	3.82 N·m
Viscous friction coefficient	D_v	3.68 N·m·s/rad
Ball drive torque constant	K_i	2.128 N·m/A

V. Control Architecture

This section describes the various controllers used on the ballbot. For all experimental results presented in this paper, the ball position and velocity data are obtained from the encoders of the ball motors, while the body angles are obtained from the IMU.

A. Balancing Control

The objective of the balancing controller is to balance the body about the desired angles, $i.e.$, roll and pitch angles. The desired body angles will be zero for a pure balancing operation, $i.e.$, standing still and non-zero in order to move around (Nagarajan et al. 2009a). The balancing controller consists of two independent controllers, one for each of the vertical planes. Each one is a Proportional-Integral-Derivative (PID) controller whose gains were tuned manually. The balancing controller, shown in the shaded part of Fig. 10, attempts to track the desired center of mass projection on the floor.

While balancing, the body angles remain within $\pm 0.1^\circ$ as shown in the plot for the pitch angle in Fig. 11(a). Similar results were obtained for the roll angle. The XY plot of the ball position on a carpeted floor under the action of just the balancing controller is shown in Fig. 11(b). The ball position was obtained using the data from the encoders on the ball motors. Even though the balancing controller does not attempt to maintain its ball position, the ball remains within ± 20 mm of its starting point on the floor. However, it is to be noted that the robot was not disturbed during this operation.

B. Outer Loop Control

The balancing controller is good at balancing about the desired body angles but does not achieve any desired ball position or velocity on the floor. This is achieved by using an outer control loop around the balancing controller, as shown for the stationkeeping controller in Fig. 10. The outer loop controller provides the desired body angles to the balancing controller.

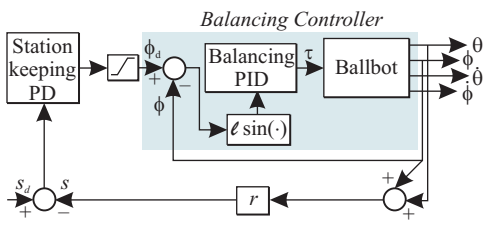
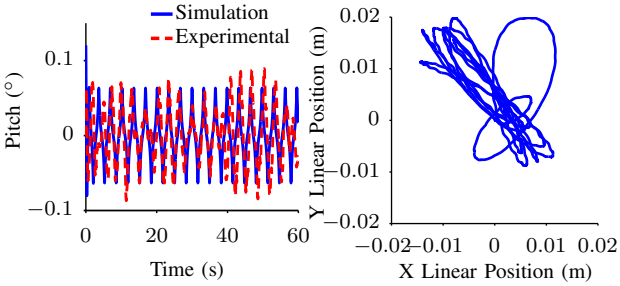
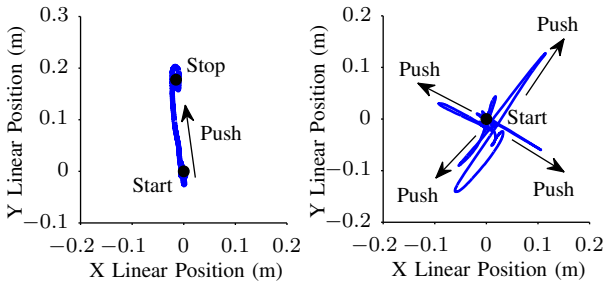


Fig. 10. Block diagram for the stationkeeping controller with the balancing control block.



(a) (b)

Fig. 11. Balancing about zero body angles: (a) Pitch angle trajectory, (b) Ball position on the carpet when the robot is not disturbed.



(a) (b)

Fig. 12. XY ball position when the ballbot is pushed with about 20 N force for 0.5 s under the action of: (a) balancing controller, (b) stationkeeping controller.

1) Stationkeeping Control: Stationkeeping is the act of balancing at a desired ball position s_d even when disturbed. The position of the ball on the floor is given by $s = r_w(\theta + \phi)$. The stationkeeping controller is a Proportional-Derivative (PD) controller that outputs desired body angles depending on the error between the ball's current and desired positions. The PD controller's angle outputs are saturated to avoid large lean angles and its gains were tuned manually. The XY plot of the robot's ball position under the action of balancing and stationkeeping controllers are shown in Fig. 12(a) and Fig. 12(b) respectively. Here, the ballbot was pushed by a human with about 20 N force for 0.5 s. While using the balancing controller, the ballbot comes to rest about 0.2 m from its initial position (Fig. 12(a)), whereas the stationkeeping controller successfully brings the robot to rest at its initial position even when disturbed (Fig. 12(b)). The ball position was obtained from the encoders on the ball motors. The companion video (Extension 1) shows the ballbot successfully stationkeeping when subjected to human pushes.

2) Velocity Control: The velocity controller is a manually tuned Proportional-Integral (PI) controller that outputs desired body angles depending on the error between the ball's current and desired velocities. The velocity

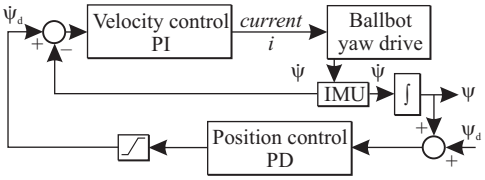


Fig. 13. Block diagram of the yaw controller.

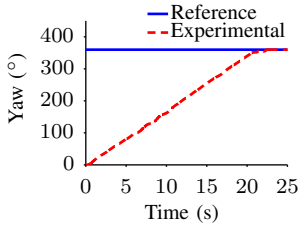


Fig. 14. 360° yaw maneuver while balancing.

controller is concerned only with the ball velocity $\dot{s}$ and not its position s. The velocity controller has two major applications, one as a stopping controller that enables the ballbot to slow down and come to rest when subjected to large disturbances; and the other for teleoperation wherein the user can provide velocity commands using a joystick. The angle outputs from the velocity controller are saturated to avoid large lean angles.

C. Yaw Control

The yaw controller, shown in Fig. 13, is decoupled from the balancing controller and consists of two loops: an inner Proportional-Integral (PI) control loop that feeds back the yaw angular velocity $\dot{\psi}$ and an outer PD control loop that feeds back both the yaw angle ψ and the yaw angular velocity $\dot{\psi}$. The yaw angle ψ is obtained by integrating the yaw angular velocity $\dot{\psi}$ data from the IMU. The output from the outer-loop PD controller is saturated to avoid high desired angular velocities. The PI loop also relies on output saturation and does not use anti-windup logic. When the body yaws, the IMU attached to the body frame rotates, whereas the ball drive unit does not. Hence, the body angles are transformed from the body frame to the drive unit frame using the angle offset provided by the absolute yaw encoder. A successful 360° yaw rotation of the ballbot while balancing is shown in Fig. 14, and its video can be found in Extension 1.

D. Leg Control

The three legs have independent controllers for both up and down motions. The legs-up controller is a PI velocity controller that stops when the legs hit the body, *i.e.,* when the leg velocity is less than a threshold. The

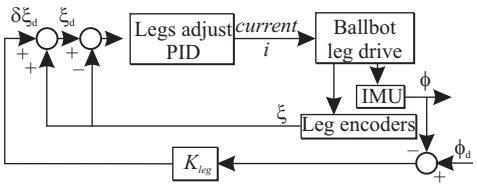


Fig. 15. Block diagram for the legs-adjust controller.

legs-down controller has an inner PI control loop that feeds back the leg velocity and an outer PD control loop that feeds back both position and velocity of the leg similar to that of the yaw controller in Fig. 13. The legs on the ballbot were not designed to snap up and down, and they are not strong enough to hold a falling robot. A combination of PI and PD control loops are used to regulate their velocities in order to minimize damage from hitting the body and the ground from legs-up and legs-down operations respectively. The three legs have independent controllers so that they can handle cases where the ballbot balances on a non-level floor.

The ballbot is in a dynamically stable state (DSS) when it is balancing on the ball and it is in a statically stable state (SSS) when it rests on its three legs. In order to be autonomous, the ballbot must have the capability to automatically transition between SSS and DSS. The ballbot must simultaneously use the legs-up controller and the balancing controller to automatically transition from SSS to DSS. This can create large, undesirable transients if the body angles are large when in SSS. The body angles will not be close to zero especially when the robot is on a non-level floor. In order to avoid this, we use a legs-adjust controller, shown in Fig. 15, to adjust the position of the legs tilting the body close to vertical when in SSS.

When the three legs are down and remain down, the ballbot forms an overconstrained spatial linkage. The ballbot's top view with all the legs deployed, shown in Fig. 16(a), suggests that the motion of leg 1 will only affect the pitch angle and not the roll angle, whereas the motion of legs 2 and 3 will affect both. Figure 16(b)

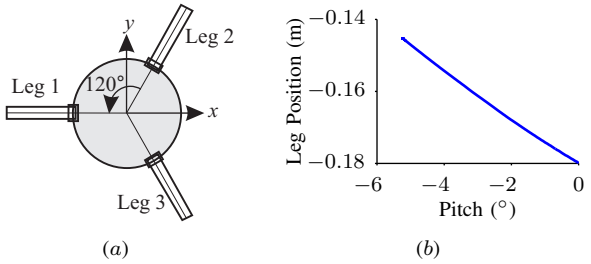


Fig. 16. (a) Top view of the ballbot with all three legs deployed; (b) Position of leg 1 as a function of body pitch.

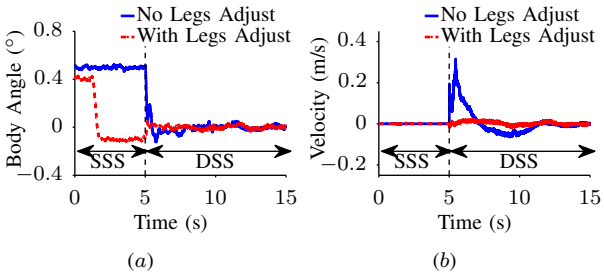


Fig. 17. Effect of Legs Adjust Controller: (a) Roll; (b) Ball Velocity.

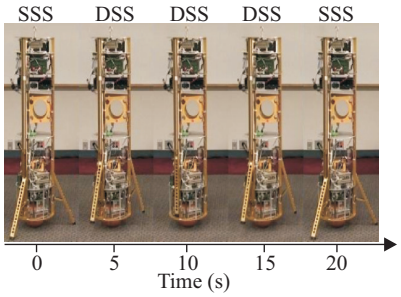


Fig. 18. Selected frames of automatic transition from SSS to DSS and vice versa.

shows a graph of the leg 1 position as a function of the body pitch angle. Similar graphs hold for legs 2 and 3. The relationship between leg position ξ and body angle ϕ is approximately linear of the form $\xi = K_{leg}\phi + c_{leg}$.

The legs-adjust controller uses the linear relationship between body angles and leg positions to achieve body angles close to zero when the ballbot is in SSS. After such an operation, the legs-up controller and the balancing controller can be simultaneously used to achieve smoother automatic transition from SSS to DSS. The transition is said to be smooth if the discontinuity in the body angle trajectory is small and the resulting ball velocity upon transition is also small.

Figure 17 shows experimental results on the ballbot with and without the use of legs adjust controller before transitioning from SSS to DSS. It can be seen from Fig. 17(a) that the use of legs adjust controller moved the body angle close to zero before transition and hence, the discontinuity in the body angle trajectory was significantly smaller than the case where the legs adjust controller was not used. Similarly, Fig. 17(b) shows that the ball velocity upon transition is significantly smaller when the legs adjust controller was used, thereby resulting in a smoother transition from SSS to DSS.

The transition from DSS to SSS is achieved by performing the legs-down operation while balancing, and by turning off balancing when the hoof switches hit the floor. Selected frames of the automatic transition from SSS to DSS and vice versa are shown in Fig. 18.

VI. Trajectory Planning between Static Configurations

The balancing controller presented in Sec. V-A is capable of keeping the ballbot upright, while the stationkeeping controller presented in Sec. V-B is capable of making the ballbot stick to its position on the floor. However, any attempt to use the stationkeeping controller to track a desired ball motion on the floor will result in jerky motions as the controllers fight against the dynamics of the system to achieve this motion. This is because the ballbot is an underactuated system with no direct actuation on the body angle ϕ. This section presents a trajectory planning algorithm that exploits the natural dynamics of the system to plan body angle trajectories, which when tracked by the balancing controller enable the ballbot to achieve desired rest-to-rest motions on the floor.

From (1), the equation of motion corresponding to the unactuated joint ϕ is given by

$$(\alpha+\beta\cos\phi_p)\ddot{\theta}+(\alpha+\gamma+2\beta\cos\phi_p)\ddot{\phi}_p - \beta\sin\phi_p\dot{\phi}_p^2 - \frac{\beta g\sin\phi_p}{r_w} = 0, \quad (14)$$

which forms a constraint on the system dynamics, and hence restricts the family of configuration trajectories that the system can follow. It is a second-order nonholonomic constraint as it is not even partially integrable (Oriolo and Nakamura 1991), and it is referred to as a dynamic constraint. Any ballbot motion must satisfy the dynamic constraint, and hence the trajectory planner presented here uses the dynamic constraint equation to plan a body angle trajectory, which when tracked will result in the desired rest-to-rest motion on the floor.

The dynamic constraint in (14) can be re-written as

$$\ddot{\theta} = f(\phi,\dot{\phi},\ddot{\phi}) = \frac{\frac{\beta g\sin\phi}{r_w}+\beta\sin\phi\dot{\phi}^2-(\alpha+\gamma+2\beta\cos\phi)\ddot{\phi}}{\alpha+\beta\cos\phi}. \quad (15)$$

Equation (15) shows that changes in the body angle can result in acceleration of the ball. Though it is not integrable, one can numerically solve the equation to find the ball angle trajectory that will result from tracking any given body angle trajectory. Given the initial ball angle θ_0 and the desired final ball angle θ_f^d, a body angle trajectory $\phi_p(t)$ for $t \in [t_0, t_f]$ is planned, which when tracked will result in a ball angle trajectory $\theta_p(t)$ such that $\theta_p(t_f) = \theta_f^d$.

A class of parametric trajectories is proposed for the body angle that enables the robot to move between static configurations, $i.e.$, rest-to-rest motions. The ballbot must lean forward in order to move forward and

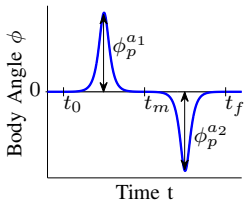


Fig. 19. Proposed parametric trajectory for body angle.

lean backward in order to come to a stop. Using this physical understanding of the ballbot motion, we follow[2] (Rosas et al. 2002) to propose the following parametric trajectory for the body angle shown in Fig. 19:

$$\phi_p(t) = \phi_p^{a_1}\,\mathrm{sech}(k\frac{2t-t_m-t_0}{t_m-t_0}) + \phi_p^{a_2}\,\mathrm{sech}(k\frac{2t-t_f-t_m}{t_f-t_m}) + \phi_p^0, \quad (16)$$

where $\phi_p^{a_1}$, $\phi_p^{a_2}$ are the amplitudes of the hyperbolic secant functions, $t_m = (t_0+t_f)/2$, $k = 9$ is a constant scalar and ϕ_p^0 is chosen such that $\phi_p(t_0) = 0$ and $\phi_p(t_f) = 0$. Positive values for $\phi_p^{a_1}$ and $\phi_p^{a_2}$ will result in a body lean trajectory that represents a lean forward and lean backward motion of the body, which is necessary for the robot to achieve a rest-to-rest motion.

Given $\phi_p^{a_1}$, $\phi_p^{a_2}$, t_0 and t_f, a smaller k value results in a wider peak, whereas a larger k value results in a narrower peak. The effect of varying k was significantly smaller than the effect of varying other parameters, and hence it was kept constant. Similarly, with no loss of generality $t_0 = 0$ was used. Therefore, the body angle trajectory $\phi_p(t)$ depends only on three parameters $\phi_p^{a_1}$, $\phi_p^{a_2}$ and t_f whose values have to be found such that the resulting ball angle trajectory $\theta_p(t)$ reaches the desired ball angle θ_f^d at $t = t_f$. For any set of parameters $\phi_p^{a_1}$, $\phi_p^{a_2}$ and t_f, the planned body angle trajectory $\phi_p(t)$ is given by (16) and the planned ball angle trajectory $\theta_p(t)$ is obtained by numerically solving (15) with the initial conditions $(\theta_p(t_0), \dot{\theta}_p(t_0)) = (\theta_0, 0)$.

The trajectory planning procedure can now be formulated as an optimization problem. The goal is to find parameters $\phi_p^{a_1}$, $\phi_p^{a_2}$ and t_f of the body angle trajectory $\phi_p(t)$ in (16) such that the objective function

$$J = w_1(\theta(t_f)-\theta_f^d)^2 + w_2\dot{\theta}^2(t_f) + \int_0^{t_f}(w_3 t + w_4\tau^2)dt \quad (17)$$

has a minimum subject to the dynamic constraint in (14). Large values are chosen for the weights w_1 and w_2 in order to ensure that the goal configuration is reached, *i.e.*, $\theta(t_f) = \theta_f^d$ and $\dot{\theta}(t_f) = 0$. The weights w_3 and w_4 determine the relative cost between time and control effort. The constraints $\phi(t_f) = 0$ and $\dot{\phi}(t_f) = 0$ are not explicitly mentioned as they are automatically satisfied when ϕ tracks $\phi_p(t)$. In this paper, the optimization is performed using Nelder-Mead simplex method (Nelder and Mead 1964), which is a direct search method that finds an optimal solution with just functional evaluations.

The planned body angle trajectory $\phi_p(t)$ is obtained by substituting the optimal parameters in (16). The planned ball trajectory $\theta_p(t)$ is determined by numerically solving the dynamic constraint equation (14) using $\phi_p(t)$, $\dot{\phi}_p(t)$ and $\ddot{\phi}_p(t)$. The optimization process finds a body angle trajectory $\phi_p(t)$ given by (16), which when tracked will enable the robot to reach the desired ball position θ_f^d at time $t = t_f$. It is to be noted that the optimal parameters $\phi_p^{a_1}$, $\phi_p^{a_2}$ and t_f obtained by the optimization process are sensitive to the initial parameter values. The trajectory planner presented here was the first successful planning algorithm on the ballbot. Since then, we have explored more sophisticated trajectory planning approaches that can generate body lean trajectories for arbitrary desired ball motions on the floor. However, an exclusive, detailed presentation of these approaches is in review (Nagarajan and Hollis 2013).

A. Feedback Trajectory Tracking Controller

Given $\phi_p(t)$ and $\theta_p(t)$, the open-loop torque input required to track the trajectories is obtained from the equations of motion in (1). However, this open-loop control input fails on the real robot due to modeling errors, nonlinear friction effects, perturbations and wrong initial conditions. In order to ensure accurate tracking of the planned ball angle trajectory $\theta_p(t)$, a feedback trajectory tracking controller is used as an outer loop controller, similar to the ones described in Sec. V-B, around the balancing controller.

The feedback trajectory tracking controller is a PID controller whose gains were manually tuned. It feeds back ball angle θ and outputs a compensation body angle ϕ_c to correct for the deviation of the ball position from its desired trajectory $\theta_p(t)$. The balancing controller tracks the desired body angle trajectory $\phi_d(t)$, which is a sum of the planned body angle trajectory $\phi_p(t)$ and the compensation body angle trajectory $\phi_c(t)$, *i.e.*, $\phi_d(t) = \phi_p(t) + \phi_c(t)$. The feedback compensation terms

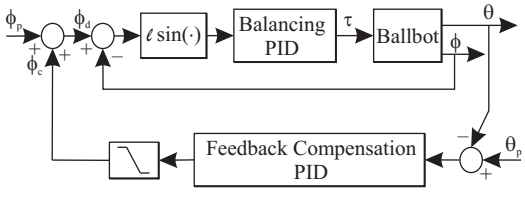


Fig. 20. Feedback Trajectory Tracking Controller

[2]In (Rosas et al. 2002), the proposed trajectory was used for the actuated joint and not for the unactuated joint as it is done here.

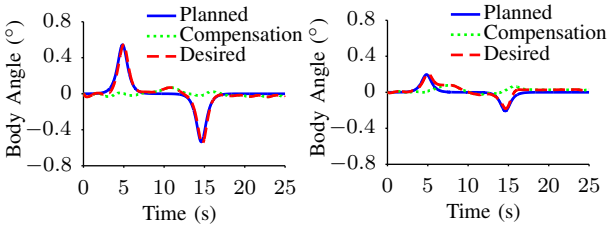


Fig. 21. Desired body angle trajectories using feedback compensation from experiments on the ballbot: (*a*) Pitch; (*b*) Roll.

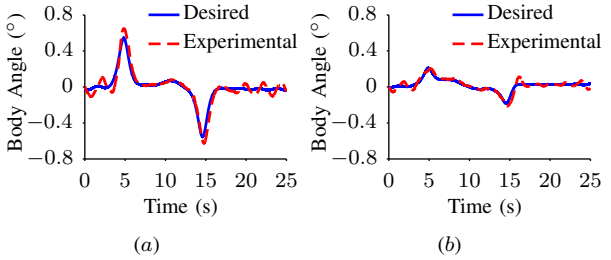


Fig. 22. Body angle trajectory tracking: (*a*) Pitch; (*b*) Roll.

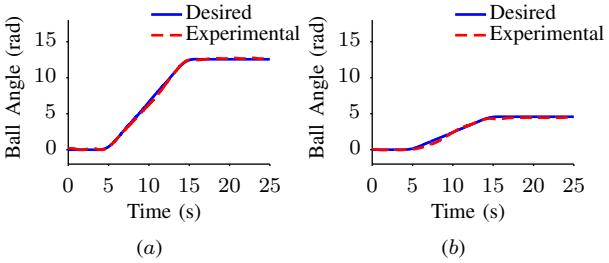


Fig. 23. Ball angle trajectory tracking: (*a*) X; (*b*) Y.

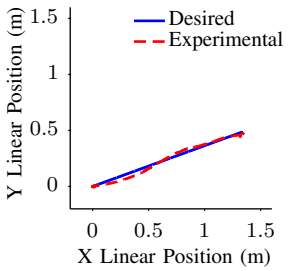


Fig. 24. Experimental trajectory tracking on the floor.

are saturated to avoid large values that might drive the system unstable.

B. Experimental Results

The trajectory planning and tracking framework presented in this section was experimentally validated on the real ballbot. The trajectory planning and optimization was done offline using MATLAB. The dynamic constraint equations were numerically solved using $ode45$ in MATLAB and the Nelder-Mead simplex method was implemented using $fminsearch$ in MATLAB.

The planned body angle trajectories for a diagonal motion of the ballbot along the floor moving 1.414 m at a direction 20° from the body X-axis are shown in Fig. 21. The feedback compensation trajectories and desired body angle trajectories are also shown in Fig. 21. A video of this motion can be found in Extension 1. The ballbot's successful tracking of the desired body and ball angle trajectories are shown in Figs. 22−24. The body angle data was obtained from the IMU, and the ball position on the floor was calculated using the odometry information from the ball motor encoders.

VII. Human–Ballbot Physical Interaction

The objective of the work presented in this paper is to explore the feasibility of developing dynamically stable mobile robots that are human-sized, dynamically agile, slender enough to easily maneuver cluttered environments and readily yield when pushed. The ballbot was developed as a simple test bed to study locomotion and physical interaction characteristics of balancing mobile robots in human environments. Sections V and VI presented the control architecture and the trajectory planner that enabled the ballbot to balance, stationkeep, yaw and achieve rest-to-rest motions. This section presents experimental results that demonstrate the robustness of the ballbot to physical disturbances that are inevitable while operating in human environments. Moreover, the dynamic stability of balancing mobile robots naturally enable them to be physically interactive (Nagarajan et al. 2009b). This section also presents several interesting physically interactive behaviors that were developed.

A. Ease of Mobility

It is desirable for robots operating in human environments to yield when pushed. Balancing mobile robots react to a disturbing force to regain balance. Only a small continuous applied force is required to move the robot from one place to another, a behavior that is intrinsically different from that of a statically stable robot. When the ballbot is balancing, a continuous force as small as 3 N is enough to keep the ballbot moving in any direction, which implies that the ballbot can be moved around with just a single finger as shown in the companion video (Extension 1). Figure 25(a) presents the linear velocity trajectories of the ballbot when pushed with three different constant forces. For a 5 s constant push, a 4.6 N force moved the robot from 0 to 0.23 m/s, while a 11.5 N force moved it from 0 to 0.5 m/s, and a 20.4 N force moved it from 0 to 0.82 m/s.

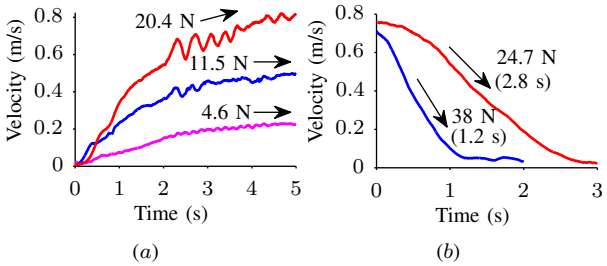


Fig. 25. Ball velocity trajectories: (a) when the ballbot is pushed with a constant force, and (b) when the ballbot is physically stopped while in motion.

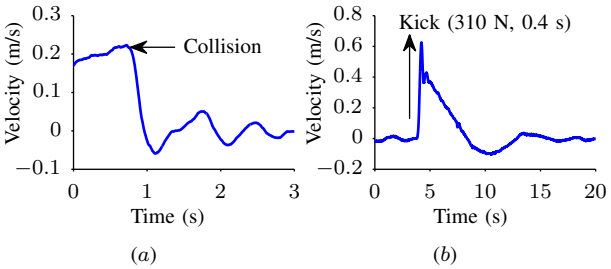


Fig. 26. Ball velocity trajectories: (a) when the ballbot collides with a rigid object; (b) when the ballbot is kicked.

Similarly, a balancing mobile robot can also be physically stopped with a small force when it is in motion. This is a key feature for a robot operating in human environments as it provides a natural way for humans to physically control the movement of the robot and stop it from running them over. Figure 25(b) shows the velocity trajectories of the ballbot when it was physically stopped while moving at 0.7 m/s and 0.75 m/s. A resisting force of 38 N applied for 1.2 s brought the ballbot to rest from 0.7 m/s, while a smaller force of 24.7 N applied for a longer time of 2.8 s brought the ballbot to rest from 0.75 m/s.

The companion video (Extension 1) demonstrates the balancing capabilities of the ballbot and also presents videos of physically moving and stopping the robot with just a single finger. It is important to note that the ease of mobility of a balancing mobile robot like the ballbot comes naturally from its dynamic stability characteristics, and is not a programmed behavior. Moreover, it does not require any extra sensing to achieve these physically interactive behaviors. However, a statically stable mobile robot needs other sensors, like cameras or tactile sensors to detect human disturbances, and also needs specially programmed behaviors to handle different scenarios.

B. Robustness to Large Disturbances

As shown in Sec. VII-A, the ballbot can be physically moved with small forces. However, it is also important for robots operating in human environments to be robust to large disturbances as well. While balancing, the ballbot can successfully handle collisions with stationary objects in the environment like furniture and walls, and bounce back as shown in Fig. 26(a).

In case of large disturbances like a kick, the ballbot is able to maintain balance, but under the action of only the balancing controller, the robot will continue to move in the direction of the push for a long time until the damping friction of the floor slows it down and brings it to rest. However, in cluttered environments, this behavior is not desired as the robot will most likely collide with other objects in the environment while recovering from the kick. Therefore, the ballbot uses a stopping controller, which is essentially the velocity controller described in Sec. V-B with zero desired velocity, to bring the robot to rest in a shorter distance. Figure 26(b) shows the ball velocity trajectory under the action of the stopping controller when the ballbot was kicked with about 310 N force for 0.4 s. Figure 27 shows the composite frames from the companion video (Extension 1), wherein the ballbot successfully recovers from a kick using the stopping controller.

C. Human Intent Detection

Humans are physically interactive with the objects in their environments, and hence, the robots operating in human environments will inevitably have physical interactions with them. These physical interactions can be used to encode interesting robot behaviors. For example, a soft push to the robot can be considered unintentional, whereas a hard push to the robot can be considered as a command to move away from its current location. This section describes an approach to detect this human intention and act accordingly.

The force exerted by a human on the ballbot directly corresponds to its acceleration, which can be used to differentiate a soft push from a hard push. The ballbot's response to such human intentions is shown in Fig. 28. The ballbot continues to stationkeep at its

Fig. 27. Composite frames from a video of the ballbot successfully recovering from a kick.

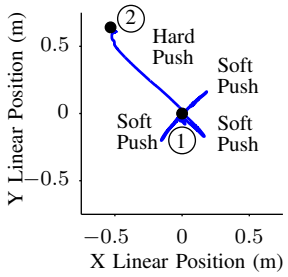


Fig. 28. The XY motion of the ball on the floor during human intent detection behavior.

current position ① when given a soft push, whereas given a hard push, it moves away and stationkeeps at a different position ②. In this experiment, any push that produces a ball acceleration greater than 0.5 m/s^2 is considered a hard push. The companion video (Extension 1) demonstrates the ballbot successfully achieving this behavior.

D. Ballbot Interface and Teleoperation

The ballbot has a highly interactive graphical user interface that allows wireless teleoperation of the robot using a joystick. The joystick commands desired velocities to the velocity controller presented in Sec. V-B. The ballbot has been reliably teleoperated at fast walking speeds for hundreds of meters over surfaces ranging from vinyl tile to carpet to rough concrete to metal gratings. The ballbot was successfully teleoperated on ramps with angles up to about 4° as shown in Fig. 29(a) and also over Ethernet cables on the floor as shown in Fig. 29(b). The videos of successful teleoperation of the ballbot can be found in Extension 1. The ballbot was able to drive into and out of elevators and over the cracks and misalignment between elevator cars and floors with ease.

VIII. Drawbacks and Challenges

In the above sections, we have experimentally demonstrated both the feasibility and the advantages of balancing mobile robots like the ballbot over traditional statically stable mobile robots. However, there are a number of challenges that remain to be addressed. One of the fundamental challenges is in the safety of such robots in human environments. The ballbot can fall down due to hardware and software problems. Therefore, the design, verification and guarantee of safety procedures is a necessity for such systems before they can be reliably deployed in human environments.

(a) (b)

Fig. 29. Teleoperation with a joystick: (a) climbing a ramp of 4°; (b) driving over wires on the floor.

In the case of the ballbot, the legs are neither strong enough to stop the falling robot, nor fast enough to be rapidly deployed. The design of better leg mechanisms with rapid deployments can be explored. Moreover, the leg design should also guarantee that the dynamics of the falling robot would not tip the robot over when the legs are deployed. Redundant sensing and actuator mechanisms like in the Segway (Nguyen et al. 2004) can also be used to increase the safety margin.

IX. Conclusions

The control implementations and their successful and robust operation on the ballbot were demonstrated. Planning body lean trajectories to achieve desired ball positions was also demonstrated. Successful trajectory tracking between static configurations was experimentally verified. Several interesting physical interaction behaviors with the ballbot were also presented.

X. Future Work

Future work include developing high-level motion planning strategies to enable the ballbot to navigate in human environments. This would require the addition of extrinsic sensors to enable localization and object detection in the environment to further enhance ballbot's locomotion and interaction capabilities. The work to date opens up a wide range of possibilities for ballbot's actions. With these considerations, it is reasonable to believe that the ballbot represents a new class of wheeled mobile robots capable of agile, omnidirectional motion.

Acknowledgments

We wish to thank Eric Schearer, Kathryn Rivard, Suresh Nidhiry, Jun Xian Leong, Jared Goerner and Tom Lauwers for their great efforts on the ballbot project. This work was supported in part by NSF grants IIS-0308067 and IIS-0535183.

REFERENCES

Anybots. http://anybots.com, 2010.

P. Deegan, B. Thibodeau, and R. Grupen. Designing a self-stabilizing robot for dynamic mobile manipulation. *Robotics: Science and Systems - Workshop on Manipulation for Human Environments*, 2006.

P. Deegan, R. Grupen, A. Hanson, E. Horrell, S. Ou, E. Riseman, S. Sen, B. Thibodeau, A. Williams, and D. Xie. Mobile manipulators for assisted living in residential settings. *Autonomous Robots, Special Issue on Socially Assistive Robotics*, 24(2):179–192, 2008.

Y.S. Ha and S. Yuta. Indoor navigation of an inverse pendulum type autonomous mobile robot with adaptive stabilization control system. In *Int'l. Symp. Experimental Robotics IV*, pages 529–37, 1997.

L. Havasi. ERROSphere: an equilibrator robot. *Int'l Conf. on Control and Automation*, pages 971–976, 2005.

R.L. Hollis. Ballbots. *Scientific American*, pages 72–78, Oct 2006.

R.L. Hollis, Y. Luo, and U. Nagarajan. An inverse mouse-ball drive mechanism for a dynamically stable single wheel mobile robot. *In Preparation*.

iBOT. http://www.ibotnow.com, 2003.

R. Kelly, J. Llamas, and R. Campa. A measurement procedure for viscous and coulomb friction. *IEEE Trans. on Instrumentation and Measurement*, 49(4): 857–861, 2000.

J. Kim and J.P. Ostrowski. Motion planning a aerial robot using rapidly-exploring random trees with dynamic constraints. *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2:2200–2205, 2003.

M. Kumagai and T. Ochiai. Development of a robot balancing on a ball. *Intl. Conf. on Control, Automation and Systems*, pages 433–438, 2008.

M. Kumagai and T. Ochiai. Development of a robot balancing on a ball - application of passive motion to transport. In *Proc. IEEE Int'l. Conf. on Robotics and Automation*, pages 4106–4111, 2009.

T. B. Lauwers, G. Kantor, and R.L. Hollis. A dynamically stable single-wheeled mobile robot with inverse mouse-ball drive. In *Proc. Int'l. Conf. on Robotics and Automation*, May 15-19 2006.

T.B. Lauwers, G. Kantor, and R.L. Hollis. One is enough! In *Proc. Int'l. Symp. for Robotics Research*, October 12-15 2005.

U. Nagarajan and R. Hollis. Shape space planner for shape-accelerated balancing mobile robots. *Int'l Journal of Robotics Research*, 2013. (Under Review).

U. Nagarajan, G. Kantor, and R.L. Hollis. Trajectory planning and control of an underactuated dynamically stable single spherical wheeled mobile robot. *Proc. IEEE Int'l. Conf. on Robotics and Automation*, pages 3743–3748, 2009a.

U. Nagarajan, G. Kantor, and R.L. Hollis. Human-robot physical interaction with dynamically stable mobile robots. *ACM/IEEE Int'l. Conf. on Human-Robot Interaction*, 2009b. (Short paper and video).

U. Nagarajan, A.K. Mampetta, G. Kantor, and R.L. Hollis. State transition, balancing, station keeping, and yaw control for a dynamically stable single spherical wheel mobile robot. *Proc. IEEE Int'l. Conf. on Robotics and Automation*, pages 998–1003, 2009c.

J.A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1964.

H. G. Nguyen, J. Morrell, K. Mullens, A. Burmeister, S. Miles, N. Farrington, K. Thomas, and D. Gage. Segway robotic mobility platform. In *SPIE Proc. 5609: Mobile Robots XVII*, October 2004.

R. Olfati-Saber. *Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles*. PhD thesis, Massachusetts Institute of Technology, 2001.

G. Oriolo and Y. Nakamura. Control of mechanical systems with second-order nonholonomic constraints: underactuated manipulators. *Proc. IEEE Conf. on Decision and Control*, 3:2398–2403, 1991.

J.R. Ray. Nonholonomic constraints. *American Journal of Physics*, 34:406–408, 1966.

Rezero. http://www.rezero.ethz.ch, 2010.

J.A. Rosas, J. Alvarez, and R. Castro. Trajectory planning and control of an underactuated planar 2R manipulator. In *Proc. IEEE Int'l Conf. on Control Applications*, pages 548–552, 2001.

J.A. Rosas, J. Alvarez, and R. Castro. Control of an underactuated planar 2R manipulator: Experimental results. In *Proc. 15^{th} IFAC World Congress*, 2002.

M.W. Spong. The swing up control problem for the acrobot. *Control Systems Magazine, IEEE*, 15(1):49–55, 1995.

M.W. Spong. Underactuated mechanical systems. In *Control Problems in Robotics and Automation*. Springer-Verlag, 1998.

M. Stilman, J. Olson, and W. Gloss. Golem Krang: Dynamically stable humanoid robot for mobile manipulation. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 3304–3309, 2010.

H. Wang, G.G. Grindle, S. Connor, and R.A. Cooper. An experimental method for measuring the moment of inertia of an electric power wheelchair. In *Proc. IEEE Conf. on Eng. Med. Biol. Soc.*, pages 4798–4801, 2007.

APPENDIX A: INDEX TO MULTIMEDIA EXTENSIONS

The multimedia extensions to this article are at: www.ijrr.org.

Extension	Media Type	Description
1	Video	Demonstrates the capabilities of the ballbot and its robustness, along with planning and physical interaction experiments