Dribbling Control of Omnidirectional Soccer Robots

Xiang Li, Maosen Wang and Andreas Zell

***Abstract*— This paper focuses on the dribbling control problem of an omnidirectional mobile robot. Because the movement of the dribbled object must be considered, dribbling control is more challenging than normal mobile robot motion control. A new feedback control algorithm, which steers a reference point to follow the desired movement and keeps the ball near to this point simultaneously, is proposed. To dribble a rolling ball along a given path, the robot should provide the ball with appropriate force by consecutive pushing operations when they travel in an environment with obstacles. Based on the analysis of the forces acting on the ball with respect to the mobile robot coordinate system, a constraint for robot movement in the dribbling process is also introduced. The simulation and real-world experiments address the performance of this control algorithm.**

I. INTRODUCTION

Dribbling control is a challenging task in the RoboCup Middle Size domain. As the rules only allow one third of the ball to be covered by the dribbler, the dribbling behavior has high requirements to the robot motion control. Some teams fulfill the dribbling task by planning robot moving paths [1] or designing the behavior-based approach [2], where the mobile robot is controlled towards the desired postures. The postures are generated by a planner based on the robot dynamic or kinematic model. However, how the robot can travels without losing the ball is not considered.

Considering the difficulty of determining the ball's motion when it is in continuous contacts with a robot, many RoboCup teams use neural networks ([3], [4]) and reinforcement learning methods ([5], [6], [7]) to learn some basic skills of the robot, such as kicking and dribbling. Although suitable simulation systems can support the learning before the experiments with a real robot, the skill learning needs long time and high computational cost.

In this paper we address the dribbling control problem for an omnidirectional robot. With the analysis of the force exerted on the ball inspired by [8], we obtain a constraint of robot movement in the dribbling process for keeping a rolling ball. Under this constraint, the task of dribbling a ball to follow a given path is completed by approaching a reference point to the given path and steering the ball to move around this point simultaneously.

II. ROBOT KINEMATIC MODEL

The mobile robot used in this dribbling task is an omnidirectional robot, whose base is shown in Fig. 1. It has three Swedish wheels mounted symmetrically with 120 degrees from each other. Each wheel is driven by a DC motor and has a same distance L_w to the robot's center of mass R.

Xiang Li, Maosen Wang and Andreas Zell are with the Department of Computer Architecture, Wilhelm-Schickard-Institute, University of Tübingen, Sand 1, 72076 Tübingen, Germany {xiang.li, maosen.wang, andreas.zell}@uni-tuebingen.de

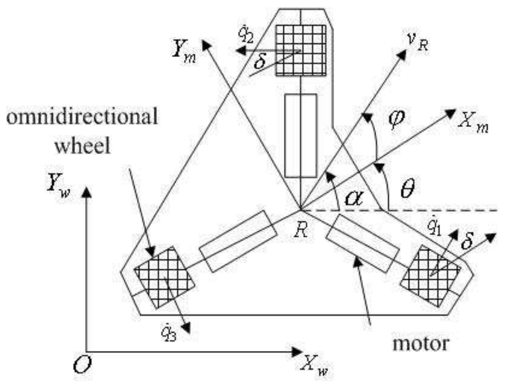


Fig. 1. Kinematics diagram of the base of an omnidirectional robot

Besides the fixed world coordinate system $[X_w, Y_w]$, a mobile robot fixed frame $[X_m, Y_m]$ is defined, which is parallel to the floor and whose origin locates at R. θ denotes the robot orientation, which is the direction angle of the axis X_m in the world coordinate system. α and φ denote the direction of the robot translation velocity v_R observed in the world and robot coordinate system, respectively. The kinematic model with respect to the robot coordinate system is given by :

$$\mathbf{v} = \begin{bmatrix} \sqrt{3}/3 & -\sqrt{3}/3 & 0 \\ 1/3 & 1/3 & -2/3 \\ 1/(3L_w) & 1/(3L_w) & 1/(3L_w) \end{bmatrix} \dot{\mathbf{q}}, \qquad (1)$$

where $\mathbf{v} = [\dot{x}_R^m \ \dot{y}_R^m \ \omega]^T$ is the vector of robot velocities observed in the robot coordinate system; $\dot{x}_R^m$ and $\dot{y}_R^m$ are the robot translation velocities; ω is the robot rotation velocity. $\dot{\mathbf{q}}$ is the vector of wheel velocities $[\dot{q}_1 \ \dot{q}_2 \ \dot{q}_3]^T$, and $\dot{q}_i (i = 1, 2, 3)$ is the i-th wheel velocity, which is equal to the wheel's radius multiplied by the wheel's angular velocity.

Introducing the transformation matrix from the robot coordinate system to the world coordinate system as

$${}^wR_m = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \qquad (2)$$

the kinematic model with respect to the world coordinate system is deduced as:

$$\dot{\mathbf{x}} = \begin{bmatrix} \frac{2}{3}cos(\theta+\delta) & -\frac{2}{3}cos(\theta-\delta) & \frac{2}{3}sin\theta \\ \frac{2}{3}sin(\theta+\delta) & -\frac{2}{3}sin(\theta-\delta) & -\frac{2}{3}cos\theta \\ \frac{1}{3L_w} & \frac{1}{3L_w} & \frac{1}{3L_w} \end{bmatrix} \dot{\mathbf{q}}, \qquad (3)$$

where $\dot{\mathbf{x}} = [\dot{x}_R \ \dot{y}_R \ \dot{\theta}]^T$ is the robot velocity vector with respect to the world coordinate system; $\dot{x}_R$ and $\dot{y}_R$ are the robot translation velocities; $\dot{\theta}$ is the robot rotation velocity; δ refers to the wheel orientation in the robot coordinate system and is equal to 30 degrees.

It is important to notice that the transformation matrix in model 1 is full rank, which denotes that the translation and rotation of the robot are decoupled, and guarantees the separate control of these two movements.

For the high level control laws without considering the wheel velocities, the kinematic model

$$\dot{\mathbf{x}} = G\mathbf{v} \tag{4}$$

is used in our control method, where the transformation matrix G is equal to $[{}^wR_m \ 0 \ ; \ 0 \ 1]$. Because G is full rank, the characteristics of decoupled movement is also kept.

III. DRIBBLING ANALYSIS

Dribbling is one of the most difficult skills of RoboCup robots. Dribbling refers to the maneuvering of a ball through consecutive and short contacts with a robot in the dynamical obstacles environment. By dribbling, a robot can travel through the opponents with the ball and shoot a goal. To dribble a ball from an initial position to the goal, the robot needs to keep the ball without losing it. Therefore, it is necessary to analyze the relative movement between the robot and the ball. When the ball is considered as a mass point B locating at the sphere center, the relation between ball's accelerations observed in the world coordinate system and the robot coordinate system is described in terms of vectors as

$$\mathbf{a}_B = \mathbf{a}_R + \mathbf{a}_B^m + 2\boldsymbol{\omega} \times \mathbf{v}_B^m + \dot{\boldsymbol{\omega}} \times \mathbf{r}_B^m + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_B^m), \tag{5}$$

where $\mathbf{a}_R$ denotes the robot translation acceleration; $\mathbf{a}_B$ and $\mathbf{a}_B^m$ are the ball's accelerations observed in the world and robot coordinate systems; $\boldsymbol{\omega}, \dot{\boldsymbol{\omega}}$ are the robot rotational velocity and acceleration respectively; $\mathbf{v}_B^m$ and $\mathbf{r}_B^m$ are the ball's velocity and position observed in the robot coordinate system.

Multiplying (5) by the mass m_B of the ball, we get the extended Newton's second law in the robot coordinate system as

$$\mathbf{F}_B^m = \mathbf{F}_B + \mathbf{F}_{in} = m_B \mathbf{a}_B^m, \tag{6}$$

where $\mathbf{F}_B = m_B\mathbf{a}_B$. $\mathbf{F}_B$ is the vector sum of all the external forces acting on the ball referring to the world coordinate system. $\mathbf{F}_B$ is composed of the force from the robot and the friction between the ball and the floor. $\mathbf{F}_B^m$ is the vector sum of the forces referring to the robot coordinate system. $\mathbf{F}_{in}$ is the inertial force calculated as

$$\mathbf{F}_{in} = -m_B\mathbf{a}_R - m_B(2\boldsymbol{\omega} \times \mathbf{v}_B^m + \dot{\boldsymbol{\omega}} \times \mathbf{r}_B^m + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_B^m)). \tag{7}$$

Equation (6) implies that not only the external forces but also the inertial force exerted on the ball when it is observed in the robot coordinate system. Although the inertial force manifests itself as a real force, it is not the real one while it results from the non-inertial observing coordinate system but not from interactions with other bodies.

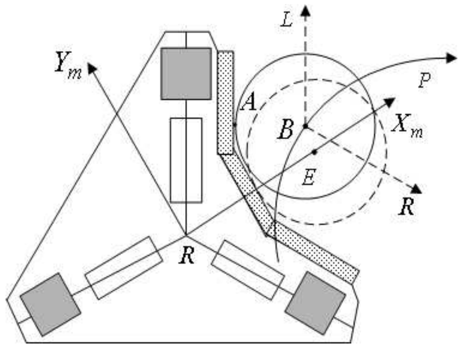


Fig. 2. Ball's relative position in the robot coordinate system. Three spongy blocks are pasted on the robot's front to increase the friction

If the ball is moving along a curve with the clockwise turning shown in Fig. 2, it is only possible for the robot to keep the ball if the force $\mathbf{F}_B^m$ has nonpositive projection on the line $\overrightarrow{BL}$ which is parallel to the left border of the robot's front. That means that

$$(\mathbf{F}_B^m)_{\overrightarrow{BL}} = (\mathbf{F}_B + \mathbf{F}_{in})_{\overrightarrow{BL}} \leq 0. \tag{8}$$

As the ball follows a curve, the external force $\mathbf{F}_B$ can be projected on the tangent and normal directions of the curve. The tangent part $\mathbf{F}_t$ determines the magnitude of the ball's velocity which can be calculated as $\mathbf{F}_t = m_B\mathbf{a}_t$ with the acceleration parameter $\mathbf{a}_t$. The value of $\mathbf{a}_t$ is hard to determine, because it refers to the friction between the ball, the robot and the floor. When the ball moves with constant velocity, $\mathbf{a}_t$ is equal to zero. The normal part $\mathbf{F}_n$ contributes to the ball's moving direction, which is pointing to the center of curvature and has the magnitude $\mathbf{F}_n = m_Bc\mathbf{v}_b^2$, where c is the curvature of the curve and $\mathbf{v}_B$ is the ball's translation velocity.

The inertial force stems from the acceleration of the reference coordinate system, which is the robot coordinate system here. In (5), the term $2\boldsymbol{\omega} \times \mathbf{v}_B^m$ is called Coriolis' acceleration; the term $\dot{\boldsymbol{\omega}} \times \mathbf{r}_B^m$ is due to the robot rotation acceleration; the term $\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_B^m)$ is called centripetal acceleration, which always points toward the axis of robot rotation.

Inequation (8) implies the constraint of robot movement in the dribbling process, under which the robot can move in the obstacles environment along curved paths avoiding losing the ball. Above analysis assumes the robot rotates in clockwise direction, but similar results can be obtained in the non-clockwise case.

IV. DRIBBLING CONTROL

In the ideal situation described in Fig. 2, the ball's center matches point E, which is always located at the front of the robot with a distance L along the X_m axis. Since we hope the ball is always in front of the robot, we can solve the dribbling problem by controlling point E to move along the desired path and keeping the ball near to point E simultaneously, with the consideration of the decoupled translation and rotation of the omnidirectional robot. In the next sections, we present the path following control of point E based on its linearized kinematic equations and the ball keeping method with a PD controller.

A. Linearizing Kinematic Model

As point E is taken as a fixed point (L,0) with respect to the robot coordinate system, its position can be transformed to the world coordinate system by

$$x_E = x_R + L\cos\theta \tag{9}$$

$$y_E = y_R + L\sin\theta, \tag{10}$$

where x_R and y_R denote the robot position in the world coordinate system; x_E and y_E denote the point E's position in the world coordinate system.

By differentiating equations (9) and (10) with respect to time and introducing (4), we get the velocities of point E as

$$\dot{x}_E = \dot{x}_R^m \cos\theta - \dot{y}_R^m \sin\theta - L\omega\sin\theta \tag{11}$$

$$\dot{y}_E = \dot{x}_R^m \sin\theta + \dot{y}_R^m \cos\theta + L\omega\cos\theta, \tag{12}$$

where $\dot{x}_R$ and $\dot{y}_R$ are with respect to the world coordinate system.

Combining equations (11) and (12) with the robot rotation velocity, the kinematic model of point E is deduced as,

$$\dot{\mathbf{x}} = G\mathbf{v}, \tag{13}$$

where

$$\dot{\mathbf{x}} = [\dot{x}_E \ \dot{y}_E \ \dot{\theta}]^T,$$

$$G = \begin{bmatrix} \cos\theta & -\sin\theta & -L\sin\theta \\ \sin\theta & \cos\theta & L\cos\theta \\ 0 & 0 & 1 \end{bmatrix},$$

$$\dot{\mathbf{x}}_m = [\dot{x}_R^m \ \dot{y}_R^m \ \omega]^T.$$

Although this system is nonlinear because of the trigonometric functions of angle θ, it can be linearized by inducing a simple compensator $C = G^{-1}$, because the matrix G is full rank. The linearized system $\dot{\mathbf{x}} = \mathbf{u}$ is shown in Fig. 3 and has a new input vector $\mathbf{u} = [u_1 \ u_2 \ u_3]^T$.

This linear system is completely decoupled and allows the controlling of point E to follow any reference path and θ to track any desired orientation in a separate way.

When a controller K is designed based on this simple linear system, the controller for the nonlinear system is generated as CK. The overall control loop, which consists of the nonlinear system, the compensator and the controller, is shown in Fig. 4.

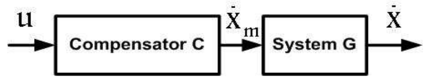


Fig. 3. Linearized system by the component C

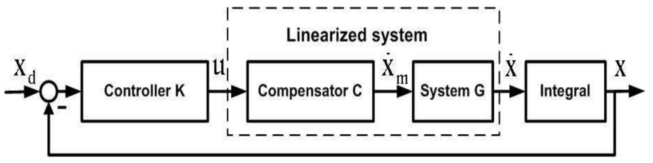


Fig. 4. Closed-loop control system

B. Path Following Control of Point E

The path following problem is illustrated in Fig. 5. P denotes the given path. Point Q is the orthogonal projection of E on the path P. $\mathbf{x_t}$ and $\mathbf{x_n}$ are the tangent and normal unit vectors at Q, respectively. The path coordinate system x_tQx_n moves along the path P. θ_P is the path tangent direction at point Q. θ_E denotes E's moving direction. The angular error is defined as $\tilde{\theta}_E = \theta_E - \theta_P$.

Based on the above definitions, the path following problem is to find proper control values of E's linear velocity v_E and angular velocity $\dot{\theta}_E$ such that the deviation distance x_n and angular error $\dot{\theta}_E$ tend to zero.

To solve this path following problem, a Lyapunov candidate function

$$V = \frac{1}{2}K_d x_n^2 + \frac{1}{2}K_\theta \tilde{\theta}_E^2 \tag{14}$$

can be considered, where K_d and K_θ are positive constants. The time derivation of V results in

$$\dot{V} = K_d x_n \dot{x}_n + K_\theta \tilde{\theta}_E \dot{\tilde{\theta}}_E. \tag{15}$$

A simple control law [9] is utilized here. It is only based on the deviation of point E to the given path, and controls point E moving along an exponential curve to converge to the axis x_t. The exponential curve is expressed as

$$x_n = x_{n_0} exp(-kx_t), \tag{16}$$

where x_{n_0} is the deviation and the positive constant k determines the convergence speed of the deviation. Differentiating (16) with respect to x_t, we get the tangent direction of the exponential curve as

$$\tilde{\theta}_E = \arctan(\frac{dx_n}{dx_t}) = \arctan(-kx_n). \tag{17}$$

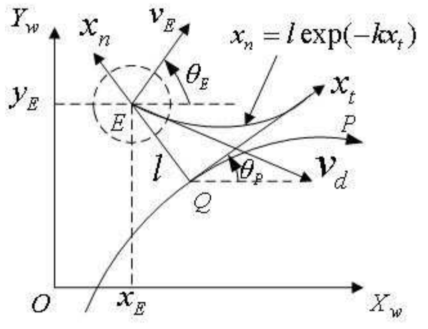


Fig. 5. Illustration of the path following problem

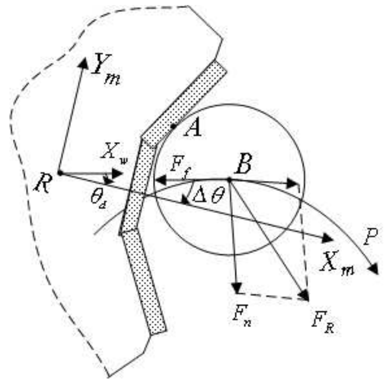


Fig. 6. Forces analysis for ball following a curve

Therefore, for a non-zero constant velocity v_d, the velocity of point E in the coordinate system x_tOx_n results in

$$\dot{x}_n = v_d \sin\tilde{\theta}_E, \tag{18}$$

$$\dot{x}_t = v_d \cos\tilde{\theta}_E. \tag{19}$$

Substituting the time derivative of $\tilde{\theta}_E$ into (15), we get

$$\dot{V} = K_d x_n \dot{x}_n + kK_\theta \arctan(-kl)\frac{-\dot{x}_n}{x_n + (kx_n)^2} < 0, \tag{20}$$

because $x_n\dot{x}_n = x_n V_d \sin(\arctan(-kx_n)) < 0$ and $\dot{x}_n \arctan(kx_n) < 0$. This solution of $\dot{V}$ guarantees the global stability of the equilibrium at $x_n = 0, \tilde{\theta}_E = 0$, which means that this control law solves the path following problem.

Transforming the velocity of point E into the world coordinate system, we get the control values of the linear system as

$$u_1 = v_d \cos(\theta_E), \tag{21}$$

$$u_2 = v_d \sin(\theta_E), \tag{22}$$

where $\theta_E = \tilde{\theta}_E + \theta_P$.

The input of this controller is the relative distance between point E and the given path, which normally can be directly obtained by the sensors on the robot. Moreover, the deviation converges to zero smoothly with the speed controlled by parameter k, which can be chosen according to the performance requirement.

C. Ball Keeping Control

As only two degrees of freedom are used to control point E to follow the given path, there is one remaining degree of freedom for the robot orientation which can be used for the ball keeping control. When a given path P performs a turning as shown in Fig. 6, the robot movement needs to be accurately controlled such that the force from the robot F_R is enough to overcome the friction F_f between the ball and the floor and provide the ball with sufficient centripetal force F_n. Although the shape of the robot's front can help the robot to exert the suitable force, it is necessary to keep the robot orientation having some deviation $\Delta\theta$ to the tangent direction of the curve.

Based on the centripetal acceleration cv_d^2, we calculate the angle deviation as $\Delta\theta = k_\theta cv_d^2$, where k_θ is the positive parameter. Then the ideal robot orientation θ_d is given by

$$\theta_d = \theta_P + \Delta\theta, \tag{23}$$

A PD controller can be used to control the robot orientation to converge to the ideal one,

$$\omega = k_p(\theta_d - \theta) + k_d(\dot{\theta}_d - \dot{\theta}), \tag{24}$$

where $\dot{\theta}_d$ and $\dot{\theta}$ are the corresponding differential values of θ_d and θ; k_p and k_d are the proportional and derivative gains, respectively.

Although a suitable dribbler mechanics can play a great role in providing friction for the robot keeping the ball, the robot movement should satisfy the condition given by (8). The required robot rotation velocity ω must be constrained based on the required robot translation velocity coming from the path following control of point E. Since point E is considered as a fix point on the robot, the movement relationship between the robot and point E in vector notion is as follows:

$$\mathbf{v}_E = \mathbf{v}_R + \boldsymbol{\omega} \times \mathbf{r}_E^m, \tag{25}$$

$$\dot{\mathbf{v}}_E = \dot{\mathbf{v}}_R + \dot{\boldsymbol{\omega}} \times \mathbf{r}_E^m + \boldsymbol{\omega} \times \dot{\mathbf{r}}_E^m, \tag{26}$$

where $\dot{\mathbf{v}}_E$, $\dot{\mathbf{v}}_R$ are the translation accelerations of point E and the robot with respect to the world coordinate system; $\mathbf{r}_E^m$ is the position vector from R to E with respect to the robot coordinate system; $\dot{\mathbf{r}}_E^m$ is the derivative of $\mathbf{r}_E^m$; $\dot{\boldsymbol{\omega}}$ is the robot rotation acceleration.

If point E is controlled with constant translation velocity moving along the given path, the tangent acceleration $\mathbf{a}_t$ is equal to zero and the centripetal acceleration is $c\mathbf{v}_E^2$. Substituting (26) into (7) and (8), the constraint of dribbling becomes

$$\begin{aligned}&\mathbf{a}_t + c\mathbf{v}_B^2 - \dot{\mathbf{v}}_E + \dot{\boldsymbol{\omega}} \times \mathbf{r}_E^m + \boldsymbol{\omega} \times \dot{\mathbf{r}}_E^m - \\ &(2\boldsymbol{\omega} \times \mathbf{v}_B^m + \dot{\boldsymbol{\omega}} \times \mathbf{r}_B^m + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_B^m)) < 0.\end{aligned} \tag{27}$$

When the ball is near to point E, $\dot{\mathbf{r}}_E^m$ approximates to zero and $\dot{\mathbf{r}}_B^m$ approximates to $\dot{\mathbf{r}}_B^m$. Then the above constraint can be reduced to

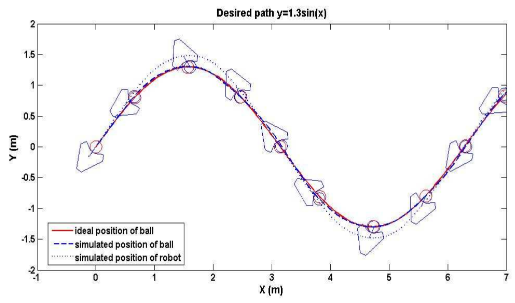
$$c\mathbf{v}_B^2 - \dot{\mathbf{v}}_E - (2\boldsymbol{\omega} \times \mathbf{v}_B^m + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_B^m)) < 0. \tag{28}$$

V. EXPERIMENTS

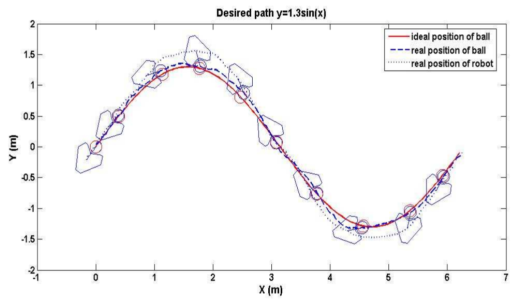
The control algorithm discussed above has been tested in both simulation and real-world environments, where the ball is required to follow a sinusoidal path $y = 1.3\sin x$ with constant translation velocity $v_d = 1.2$ m/s.

A. Simulation Experiment

In our simulator, the movements of the robot and the ball have been calculated from their kinematic equations. The pushing process is considered as a consecutive high frequency and low magnitude compact process. We utilize the basic collision equations, which describe the collisions of

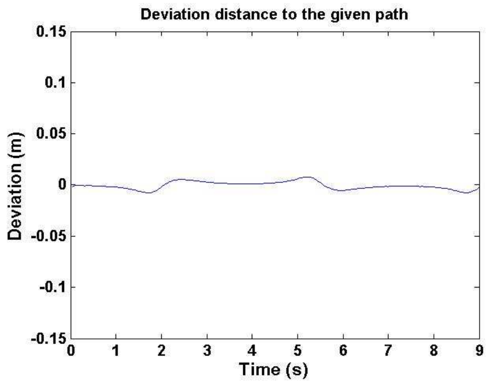


(a) Simulation

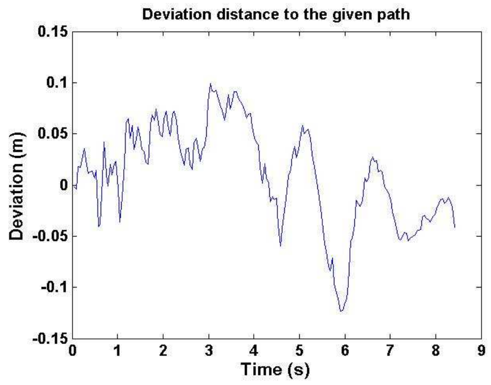


(b) Real-world experiment

Fig. 7. Ball follows the path $y = 1.3 \sin x$ with $v_d = 1.2$ m/s

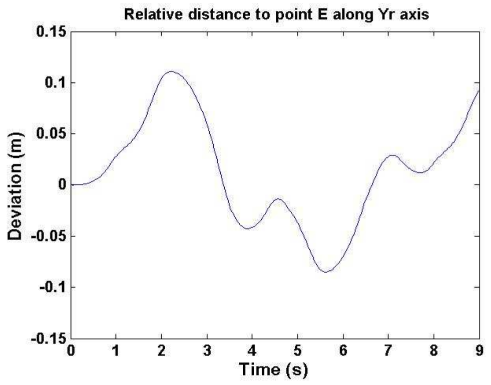


(a) Simulation

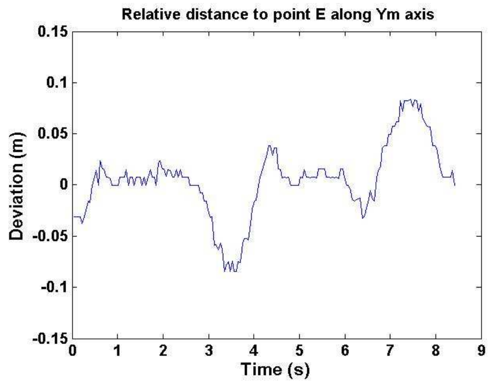


(b) Real-world experiment

Fig. 8. Deviation from point E to the desired path



(a) Simulation



(b) Real-world experiment

Fig. 9. Relative deviation from ball to point E along the axis Y_m

two rigid bodies, and Newton's law of restitution to calculate the motion of the two objects after collisions.

The difficulty in this collision simulation is the exact determination of the collision moment. As a common method, we detect whether the robot and the ball overlap by some degree due to the limited time; if they overlap, the simulation calculates n time steps backwards within the last cycle such that the overlap disappears. Since the simulation cycle is short enough, the accuracy of the collision detection is satisfiable. The parameters of our simulator, like the friction

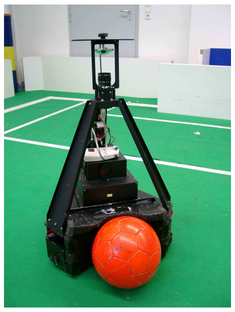


Fig. 10. The real omnidirectional robot and ball

coefficient and the restitution coefficient, are adopted as experience values. The masses and the moment of inertia of robot and ball are adapted to the real values.

The parameters of our control algorithm were chosen as $k = 3.5,\ k_\theta = 0.9,\ k_p = 10,\ k_d = 6$. These results illustrated in Fig. 7(a), 8(a) and 9(a) show us that the dribbling control method steers the point E converging to the given path with little deviation. Meanwhile, the ball does not slide away from the robot, because the ball's relative distance to point E along the axis Y_m is always less than the maximum value $\pm$15cm.

B. *Real-world Experiment*

The real-world experiment was made in our robot laboratory having a half-field of the RoboCup middle-size league. The ball's observation values come from our omnidirectional viewing system and object detection process. Our omniderectional viewing system consists of a AVT Marlin F-046C color camera with a resolution of 780×580, which outputs images up 50 times per second. In order to achieve a complete surrounding map of the robot, the camera is assembled pointing up towards a hyperbolic mirror, which is mounted on the top of our omnidirectional robot, as shown in Fig. 10. After obtaining the image from the camera, a fast object detection algorithm is used to get the ball's real world position, as described in [10].

The parameters of our control algorithm were chosen as $k = 3.5,\ k_\theta = 0.9,\ k_p = 5,\ k_d = 3$. Although measurement noise, environment disturbance and the actuator delay of the robot were induced, the experiment results illustrated in Fig. 7(b), 8(b) and 9(b) demonstrate that the control method has good performance.

VI. CONCLUSIONS

In this paper a new control method for an omnidirectional robot dribbling a rolling ball is presented. This approach solves the control problem of the consecutive mobile robot pushing operation by introducing a reference point as the controlled object, which is also regarded as a fixed point of the robot itself. Feedback control methods are used to steer the reference point to follow the given path and the ball to move around the reference point simultaneously. In order to keep the ball, the relative movement between the robot and the ball has been analyzed and a constraint of robot movement is presented, with which controller outputs, namely the required translation and rotation velocities, have been limited. With the Lyapunov stability theory, the global stability of the path following control law has also been proven.

The simulation and real-world experiments used a sinusoidal curve as the ideal path, and required a constant translation velocity of the ball. The results show that this control method can control the omnidirectional robot to apply appropriate pushing operations such that the ball follows the given path and does not slide away from the robot.

VII. ACKNOWLEDGMENT

The authors would like to thank the reviewers for their constructive comments and suggestions, which have greatly improved the quality of this manuscript.

REFERENCES

[1] T. Weigel, W. Auerbach, M. Dietl, B. Dmler, J.-S. Gutmann, K. Marko, K. Mller, B. Nebel, B. Szerbakowski, and M. Thiel, "Cs freiburg: Doing the right thing in a group," in *RoboCup 2000: Robot Soccer World Cup IV*, Melbourne, Australia, 2001, pp. 52–63.

[2] R. Emery and T. Balch, "Behavior-based control of a non-holonomic robot in pushing tasks," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, 2001, pp. 2381–2388.

[3] V. Ciesielski and S. Y. Lai, "Developing a dribble-and-score behaviour for robot soccer using neuro evolution," in *Workshop on Intelligent and Evolutionary Systems*, Dunedin,New Zealand, 2001, pp. 70–78.

[4] B. Hecht, M. Simon, O. Tenchio, F. Wiesel, A. Gloye, and R. ùl Rojas. (2005) A neural coach for teaching robots using diagrams. [Online]. Available: http://page.mi.fu-berlin.de/∼rojas/2005/NeuralCoach_Hecht_Rojas.pdf/

[5] M. Riedmiller and A. Merke, "Using machine learning techniques in complex multi-agent domains," 2003.

[6] P. Jonker, B. Terwijn, J. Kuznetsov, and B. van Driel, "Algorithmic function of the clockwork orange robot soccer team," in *Proc. 6th Int. Workshop on the Algorithmic Foundations of Robotics*, Zeist/Utrecht, Netherland, 2004, pp. 1–10.

[7] T. Gabel and M. Riedmiller, "Learning a partial behavior for a competitive robotic soccer agent," *KI - Künstliche Intelligenz*, vol. 20(2), pp. 18–23, 2006.

[8] B. D. Damas, P. U. Lima, and L. M. Custódio, "A modified potential fields method for robot navigation applied to dribbling in robotic soccer," in *RoboCup 2002: Robot Soccer World Cup VI*, Fukuoka, Japan, 2003, pp. 65–77.

[9] A. Z. Alexander Mojaev, "Tracking control and adaptive local navigation for nonholonomic mobile robot," in *Proc. of the IAS-8 conference*, Amsterdam, Netherland, 2004.

[10] P. Heinemann, T. Rueckstiess, and A. Zell, "Fast and accurate environment moddelling using omnidirectional vision," in *Dynamic Perception*, H. U.Ilg, H.Bueltfhoff, Ed. Infix, 2004.