Proceedings of the 37th Chinese Control Conference
July 25-27, 2018, Wuhan, China

Pose Optimization of a Single Ball Self-balancing Mobile Robot Based on IMU and Visual Fusion

She Jing[1], Zhongli Ma[1,2], Kang Yi[1], Huixin Li[1], Zuoyong Li[2]

1. Harbin Engineering University,Harbin,150001,China
E-mail: mazhongli@hrbeu.edu.cn
2. Fujian Provincial Key Laboratory of Information Processing and Intelligent Control, Minjiang University, Fuzhou,350121, China
E-mail:journalpaper@yeah.net

Abstract: The single ball self-balancing mobile robot is a typical multivariable, high-order, nonlinear and static instability system, IMU inertial devices are generally used to obtain attitude parameters of the system. The data errors obtained by the inertial devices will accumulate over time, and the long-term pose estimation is rather unreliable. But for some quick motion over a short period of time, inertial sensors can get very accurate estimates. Monocular cameras are often used as image real-time monitoring sensors by single ball self-balancing mobile robots. The camera's data does not accumulate errors over time, so there is essentially no drift. Therefore, camera pose estimation can effectively correct the inertial sensor error. In a fast-moving environment, the shutter-roller camera cannot capture the image clearly and there may be ghosting or overlapping of two areas too little, resulting in mismatched or unmatched conditions. So that the robot makes a wrong estimate. In this paper, the non-linear optimization method is used to fuse the IMU and monocular attitude data to obtain the optimal pose estimation of the robot. Based on the modeling of robot using Lagrange's equation method, the control of single ball robot is realized by using fuzzy PID algorithm. Experiments show that this scheme can respond quickly to the change of the attitude of the ball self-balancing robot and keep the attitude of the robot relatively stationary, so that the robot can move steadily.

Key Words: Ball self-balancing robot, Attitude detection, Nonlinear optimization，Visual inertial data fusion

1 Introduction

In the field of agriculture, industry and anti-terrorism, wheeled robots with light weight, simple structure, convenient operation, high flexibility and reliability have been widely applied. However, in applications that require human-machine collaboration, wheeled robots cannot respond rapidly to human actions due to their inability to flexibly turn around and move in a small space [1]. Ball self-balancing robot is a typical wheeled robot, but different from the ordinary wheeled robot, it uses a circular ball as the driving wheel to achieve the movement in the horizontal plane. Ball self-balancing robots have the flexibility to move in a narrow space because the circular ball as a driving wheel has the advantage of commutating without turning, enabling full-scale robot movement to be achieved. The advantages are as follows:

1) The turning radius is zero: when the moving direction is adjusted, the robot moves in a small space through the rotational movement around the center of the main body;

2) There is no brake system: in the brakes, the robot does not need a reducer but through the positive and reverse torque provided by the motor controlled by the CPU;

3) Small power consumption: the robot does not need large driving power and can prolong the use time of the battery;

4) Small size: the area required is much smaller than the ordinary wheeled robot.

*This work is supported by National Natural Science Foundation (NNSF) of China(No.61772254) and Key Project of College Youth Natural Science Foundation of Fujian Province (No.JZ160467)

Although ball self-balancing robots have so many advantages, such robotic systems are very complex, non-linear and multi-degree-of-freedom. Therefore, the research and application of various control methods for this platform are of great significance.

In 2005, TBLauwerS, GA.Kantor and RLHolliS of Carnegie-Mellon University first studied the ball self-balancing robot and introduced a unicycle self-balancing robot named "Ballbot" [2] to better human-computer interaction. The biggest difference between "Ballbot" and a normal self-balancing robot is its flexible movement in any direction. The inertial measurement unit consists of a gyroscope and an encoder. Ball moving distance is determined by the angle of the robot measured by the gyroscope and the position of the motor shaft measured by the encoder. In 2008, Tohoku University in Japan developed the BallIP1 and BallIP2 self-balancing robots based on the single-wheel self-balancing robot Ballbot [3]. In order to obtain the effect of smooth and low buffeting, the robot adopts the direct drive mechanism. In 2010, Japan's Tohoku University invented a new ball drive robot - "BallIP-W". The robot's new drive is called a partial sliding roller [4]. In 2010, the Swiss Federal Institute of Technology Zurich invented the "Rezero" ball self-balancing mobile robot [5]. In order to measure the position and speed of the motor shaft, each motor is also equipped with an encoder. The drive ball is made of aluminum ball, at the same time, in order to increase the friction, the drive ball also installed three brakes. In 2013, based on the research done by the University of Zurich, Beijing University of Science and Technology developed a single ball balancing robot that uses basketball as a driving ball and basically achieves a full range of stable motion [6]. In 2017, Jiangxi University of Science and Technology reduced the complexity of a ball-balancing robot system by

using an inverted pendulum model and an IASMP model and developed a ball-balancing robot that can operate stably [7]. However, most of the researches use inertial elements for the attitude control of robots, and the pose errors measured with inertial devices increase with time, so the long-term accuracy is poor. However, the ball self-balancing robot requires very high accuracy of pose information, otherwise It is difficult for the robot to maintain homeostasis.

In this paper, we combine the IMU and monocular camera data to optimize the pose of the robot using the nonlinear optimization method, and realize the motion control of single ball self-balancing mobile robot. The robot body can maintain dynamic stability when it stands on the driving ball, resist external interference within ± 10 ° without position closed loop, and can move in any direction at the speed of 0.1-0.5m/s .

2 Research on System Modeling and Control Methods

The single ball self-balancing mobile robot shown in Fig. 1 mainly includes the robot body, driving device and driving ball. The single ball self-balancing robot's drive ball contacts the road surface with a very small area, which makes its moving more flexible. The movement of the driving ball is driven by three omnidirectional wheels locating at its upper part. The three omnidirectional wheels can provide any directional forces in the space, so that the driving ball can move on any direction in the horizontal plane [8].

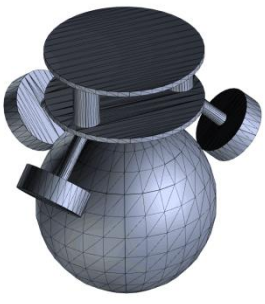

Fig. 1: Robot structure

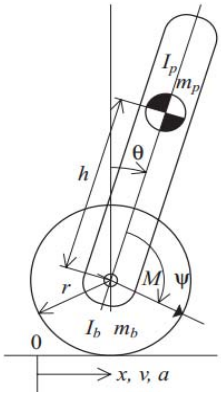

Fig. 2: YOZ plane Equivalent Model

2.1 System Modeling

A three-dimensional coordinate system XYZ, regarding gravity center of the single ball self-balancing robot as zero, is established, and three-dimensional model of the whole system is projected onto three planes XOY, XOZ and YOZ in the coordinate system. The XOY plane model is equivalent to the robot's rotation around the axis of the main body. The XOZ plane model is equivalent to the body's translation or rotation along the Y axis; the YOZ plane model is equivalent to the body's translation or rotation along the X axis. Since the three plane models are independent of each other, the model can be considered separately without considering the coupling between the three plane models. This paper does not take the rotation as the main research content, so only consider the XOZ and YOZ planes, and the two plane models are the same, so only the YOZ plane is considered. As shown in Fig.2, YOZ plane can be seen as an inverted pendulum model. In the Fig. 2, I_b 、I_p are inertia moments. m_p is the mass of the robot body. m_b is the mass of the ball. r is the diameter of the sphere. h is the length of the center of mass of the pendulum rod to the center of the ball. θ is the angle of inclination of the pendulum rod. Ψ is the relative angle of the pendulum rod to the ball. M is the moment acting on the ball. x is the movement distance. v is the speed. a is the acceleration[9].

Generally, the motion modeling of the inverted pendulum can build and analyze using Lagrange's equation [10], and then the control algorithms of the model can use LQR, PID and so on[11]. Overall, the system can be equivalent to the synthesis of two inverted pendulum models. Therefore, through installing an accelerometer and a gyroscope in the axe center, the angles and accelerations of the two axes can be detected; using the angles and accelerations can inform a closed loop feedback to realize dynamic balance and fast, accurate response of the system. The relationship between the speed of the three omnidirectional wheels and the speed of the robot's main body shows as following:

$$\begin{cases} v_{s1} = -v_y * \cos\theta \\ v_{s2} = \left(\frac{\sqrt{3}}{2}v_x + \frac{1}{2}v_y\right) * cos\theta \\ v_{s3} = \left(-\frac{\sqrt{3}}{2}v_x + \frac{1}{2}v_y\right) * cos\theta \end{cases} \quad (1)$$

Where, v_{si} denotes the speed of the ith omnidirectional wheel, v_x denotes the robot's speed along the x-axis, v_y denotes the robot's speed along the y-axis. θ is the angle between the motor and the central axis, where θ takes 45 degrees. According to the above derivation, only the desired running direction and speed need to be given, and the corresponding speed of the three omnidirectional wheels can be converted and output, and the basic motion control can be realized through the closed-loop feedback of the system.

2.2 Fuzzy PID algorithm

The ball self-balancing robot belongs to the statically indeterminate system, and it is seldom in the equilibrium state, so, the static difference has less effect on the system stability. Moreover, the noise signals that cannot be filtered are included in pose information from inertia sensor, and the noise signals gradually accumulate through the integrator and eventually produce control errors. Therefore, we need to select the appropriate control algorithm to reduce the error.

The application of PID algorithm in real life is very common. However, in complex system control, it is difficult to achieve accurate control with only constant parameters. Fuzzy control method does not need to establish an accurate system mathematical model. As long as the measurement information is processed by fuzzy processing, the output is determined through reasoning mechanism, and finally the output is precisely processed. The fuzzy PID control method formed by the complementary improvement of the fuzzy algorithm and the PID algorithm is widely used and has better effect on the nonlinear system.

Due to the complicated structure of the ball self-balancing vehicle system, the change of the external environment and the delay of the sensor data, this paper adopts the fuzzy PID control algorithm. The fuzzy PID adjusts its own parameters according to the fuzzy rules set in advance, so the

parameters are easier to adjust and the system is more robust. The relationship between the PD controller input e(t) and the output u(t) is as follows:

$$u(t) = K_P e(t) + K_D \frac{de(t)}{dt} \tag{2}$$

In the mathematical expressions of the PD algorithm, adjusting the K_p coefficient and the K_d coefficient can adjust the dynamic response of the control system. In this paper, the input of the upright ring PD controller is the deviation between the robot's own angle and the desired angle, and the output of the controller is the motor speed, as shown in formula (3)

$$\begin{cases} V_x = K_{p1} \times \theta + K_{d1} \times \dot{\theta} \\ V_y = K_{p2} \times \gamma + K_{d2} \times \dot{\gamma} \end{cases} \tag{3}$$

Where, K_{p1}, K_{d1} is the controller parameter of the XOZ plane projection model, K_{p2} , K_{d2} is the controller parameter of the YOZ plane projection model, θ is the pitch angle, and γ is the roll angle.

According to the characteristics of the robot movement and the requirement of the system performance, the appropriate fuzzy rules are formulated. The changes in the parameters K_p and K_d in the formula follow the rules in Table.1.

Table.1. Parameter change rules table

Condition	Parameter change rules
The angle at which the robot deviates from its equilibrium position increases	K_p increases to bring the robot back to equilibrium quickly
The direction of the deviation E is the same as the direction of the rate of change of deviation EC	K_p should be increased. the system can quickly make the corresponding. K_d should take a smaller value
The direction of deviation E is opposite to the direction of deviation change rate EC	K_p takes a smaller value to reduce overshoot. Take smaller K_d to reduce EC effects
The system is in balance	K_p takes a smaller value to reduce overshoot and concussion. K_d should be smaller, in order to adjust the balance position sensitivity
The system is far from equilibrium	K_d should be reduced to avoid differential oversaturation caused by rapid increase of error

3 Pose optimization

In this paper, the robot carries two main sensors, inertial sensor (IMU), image sensor (monocular camera). Two kinds of sensor data are used to make an optimal estimation of pose of the robot in a non-linear optimization way to achieve the advantages of two kinds of sensor measurement data complement each other.

Nonlinear optimization is generally divided into three steps, setting the objective function, looking for gradient direction, iterative update to convergence. The last two steps in most of the problems on the approach taken are similar, so this paper mainly describes how to set the target function.

Let x be the state vector to be optimized. $x = [R, v, p]$. Where, R is the rotation matrix of the robot relative to the world coordinate system, $R \in SO(3)$. The rotation matrix contains the yaw, roll, and pitch angles to be optimized, where the roll angle and the pitch angle are the input of the above erect controller. v is the speed of the robot for speed closed-loop control. $v \in \mathbb{R}^3$. p is the robot's space position coordinates as the robot position closed loop input. $p \in \mathbb{R}^3$. The basic form of the objective function can be deduced according to the maximum posterior conditional probability [12] of the observed and state variables, as shown in expression (4):

$$\mathcal{L}(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} e_r^{i,jT} W_r^{i,j} e_r^{i,j} + \sum_{i=1}^{m} e_s^{iT} W_s^{i} e_s^{i} \tag{4}$$

Where, i represents the i-th frame of image, j represents the j-th landmark, e_r represents the error term of camera re-projection, and e_s represents the error term of IMU. The first term in the expression indicates the error cost incurred by the camera, the second term represents the error penalty generated by the inertial device. The next goal is to introduce a specific objective function and adjust the robot's variable x by a gradient descent so that the cost function $\mathcal{L}(x)$ is minimized.

3.1 Camera re-projection error

The PNP algorithm is used to find the three-dimensional coordinates of the waypoint P in the first frame. The three-dimensional coordinates of the road marking point P in the image of the second frame are obtained by rotating and panning. And a P observation p2 is obtained on the imaging plane of the camera, as shown in Fig.3.

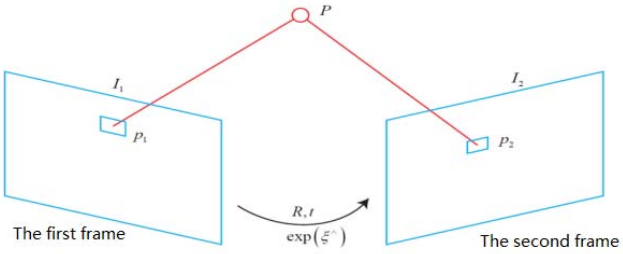

Fig.3: Projection of landmarks in two frames of image

According to the projection equation [13], the expression of the projection error term can be directly obtained as shown in equation (5):

$$e_r^{i,j} = z^{i,j} - f_i(R_j, p_i, \rho_j) \tag{5}$$

Where, $z^{i,j}$ is the pixel coordinate of the roadmap point observed in the current camera coordinate system, and ρ_j represents the space coordinate of the roadmap point in the current camera coordinate system.

3.2 Inertial device error

The measurement of the inertial element is made up of three parts, the true value, the drift error and the white noise,

so the measurement of the acceleration and angular velocity of the IMU has the following expression:

$$\tilde{\omega}(t) = \omega(t) + b^g(t) + \eta^g(t) \quad (6)$$

$$\tilde{a}(t) = a(t) + b^a(t) + \eta^a(t) \quad (7)$$

Where, $\tilde{\omega}(t)$ is the measurement of angular velocity, $\omega(t)$ is the true value, and $b^g(t)$ is the gyro drift error , $\eta^g(t)$ is the white noise of the gyroscope. (7) Same as above. The gyroscope's motion model is as follows:

$$\dot{R} = R\omega^{\wedge}, \quad \dot{v} = a, \quad \dot{p} = v \quad (8)$$

Combining equations (6) through (8) yields the following equation:

$$R(t+\Delta t) = R(t)Exp((\tilde{\omega}(t) - b^g(t) - \eta^g(t))\Delta t$$
$$v(t+\Delta t) = v(t) + g\Delta t + R(t)(\tilde{a}(t) - b^a(t) - \eta^a(t))\Delta t$$
$$p(t+\Delta t) = p(t) + v(t)\Delta t + \frac{1}{2}R(t)(\tilde{a}(t) - b^a(t) - \eta^a(t))\Delta t^2 \quad (9)$$

According to the above integration method, the data of the IMU can be integrated in an iterative manner. The pose data of the i + 1th key frame can be iteratively solved through the pose data of the ith keyframe. So that you can use IMU data and camera data in the optimization. But if the integral method above is used directly, there are two problems[14]:

1) In the above integral process, the pose information of the previous moment is coupled to the integral process. In this case, if we want to derive the pose of the previous frame, we need to derive each data through the chain rule, which consumes a great amount of computing resources.

2) In the optimization process, the pose of each frame has been slightly changed. However, if you use the above integral method, as long as the posture of the previous frame has changed a little, then the integral will be recalculated.

In order to solve the above problem, the preintegration method proposed by Christian Forster is used to calculate the residual error[15]. As equation (10), the error term for inertial elements is $e_s = [e_{\triangle R_{ij}}^T, e_{\triangle v_{ij}}^T, e_{\triangle p_{ij}}^T]^T \in \mathbb{R}^9$.

$$e_{\triangle R_{ij}} = Log\left(\left(\triangle \tilde{R}_{ij}\left(\bar{b}_i^g\right) Exp\left(\frac{\partial \Delta \bar{R}_{ij}}{\partial b^g}\delta b^g\right)\right)^T R_i^T R_j\right)$$

$$e_{\triangle v_{ij}} = R_i^T\left(v_j - v_i - g\Delta t_{ij}\right) - [\Delta \tilde{v}_{ij}\left(\bar{b}_i^g, \bar{b}_i^a\right) + \frac{\partial \Delta \bar{v}_{ij}}{\partial b^g}\delta b^g + \frac{\partial \Delta \bar{v}_{ij}}{\partial b^a}\delta b^a]$$

$$e_{\triangle p_{ij}} = R_i^T\left(p_j - p_i - v_i\Delta t_{ij} - \frac{1}{2}g\Delta t_{ij}^2\right) - [\Delta \tilde{p}_{ij}\left(\bar{b}_i^g, \bar{b}_i^a\right) + \frac{\partial \Delta \bar{p}_{ij}}{\partial b^g}\delta b^g + \frac{\partial \Delta \bar{p}_{ij}}{\partial b^a}\delta b^a] \quad (10)$$

In the formula, the variable wiping the wavy line indicates the pre-integral value. Details can refer to reference [15].

3.3 Optimization

Through the above, this paper has got a concrete objective function, then we use the gradient descent method to find the pose variable that minimizes the objective function. The Jacobian matrix [16] is generally used to locally linearize the function to find the gradient direction. In order to optimize the convenience, this paper uses the method of logarithmic mapping to remove the constraint of the rotation matrix itself.

The specific optimization algorithm has Gaussian Newton method, Levenberg - Marquardt method [17]. This paper uses the latter to optimize pose. The specific optimization process is shown in Fig.4. Through the iterative method, the cost function is continuously decreased along the gradient direction, so that the error is reduced and the optimization is achieved.

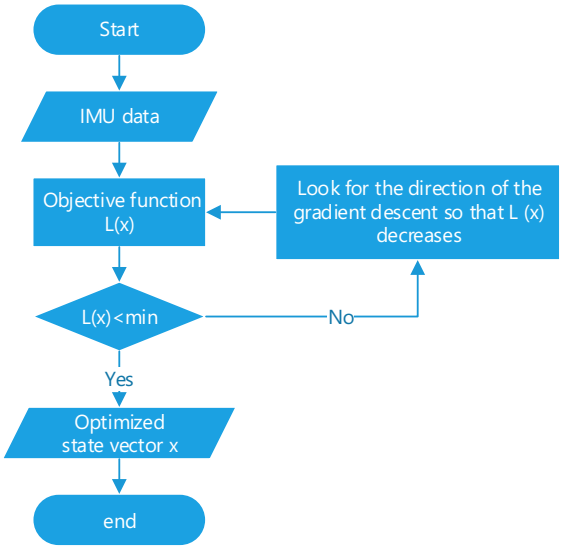

Fig.4: Optimize program flow chart

3.4 Posture optimization effect

After optimizing the pose of the system by combining the data of the camera and the inertial device, as shown in Fig.5, it can be seen that the optimized pose curve is much smoother than before optimization, which shows that the optimized pose is more accurate.

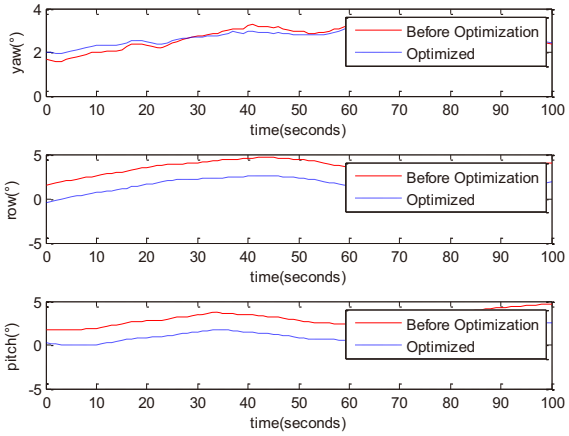

Fig.5: Pre and post-optimization rotation angle changes

4 System testing and results

The experimental site is a site built in the laboratory. Light is normal indoor light, no direct sunlight. The system uses li-po battery-powered. li-po battery provides 11.1V power supply for the control panel's own various functions debugging and the overall debugging of the robot system. After power-on test, all aspects of the control panel are working properly.

In this paper, many experiments have been performed on the ball-balancing robot to verify its stability and robustness. First of all, the erection test is performed. The power-on self-balancing robot is placed on the basketball, the robot can successfully be in a state of dynamic equilibrium without external support. Test process shown in Fig.6, the robot can be basically stable in the upright state.

Fig.6: Robot upright testing

The response of the robot's posture under external disturbance is shown in Fig.7. Assuming that the system is in the beginning in a state of equilibrium with a declination angle of zero, the system can immediately return to equilibrium after the disturbance has disappeared. It can be seen that the response curve after pose optimization has smaller steady-state error than before optimization, which shows that the optimized data error decreases and the system has better anti-disturbance effect.

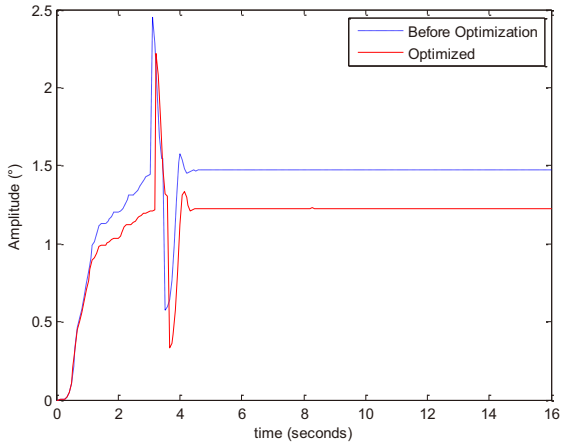

Fig.7: Response curves of the system before and after the pose optimization under perturbation

After the camera and the inertial device data are used to optimize for attitude and posture, the robot attitude angle changes as shown in Fig.8. When the pose expectation angle is (0,0), the roll and pitch angle of the robot can be kept in the range of (-0.6 ~ 0.6 °).

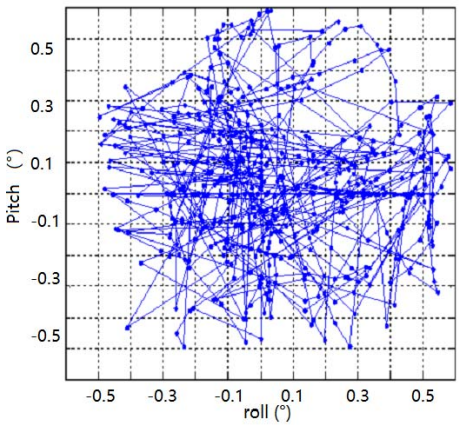

Fig.8: Attitude at rest

As shown in Fig.9. the robot's position moves between (-4cm, -6cm) and (5cm, 6cm) when the desired position for a given position is (0cm, 0cm).

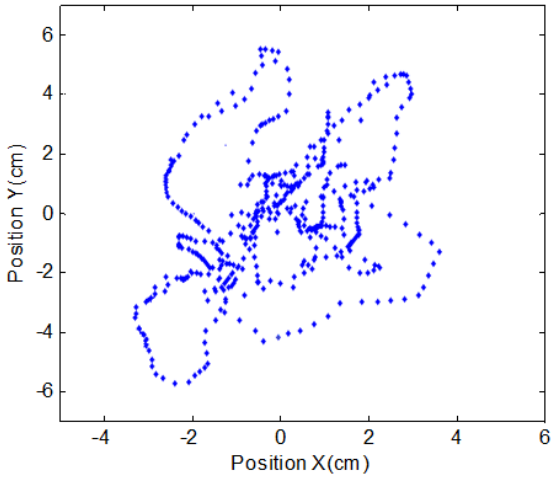

Fig.9: Position of the given robot position (0,0)

5 Conclusion

This paper studies the ball-self-balancing robot system's motion control and upright control. The fuzzy PID algorithm is used to realize the basic dynamic balance control of the system. A novel method is proposed to optimize the pose of the robot. The robot pose is optimized by combining the data of the two sensors for nonlinear optimization. As can be seen from the experimental results, the ball self-balancing robot standing on the driving ball can maintain dynamic stability. And only about $\pm 0.6°$ jitter occurs in the equilibrium position. When the robot is at rest position deviation can be maintained at about 6cm, and robots can move at 0.1-0.5m / s speed in any direction. It can be seen that the system can operate normally. Under the fuzzy PID algorithm and non-linear optimization, the system can maintain high precision operation and resist the interference in the range of ± 10 °, basically achieving the goal of this paper. However, due to the complexity of the ball self-balancing robot system, there are still many shortcomings and deficiencies in this subject. Mainly as follows:

1) However, the fuzzy PID control algorithm used in this paper is not very good. The control method still needs to be improved.

2) Because there is no standard for data filtering, solution and fusion of the inertial sensor, it is impossible to determine the error, thus reducing the accuracy.

3) The fusion of visual sensor and inertial sensor information can eliminate the error to a certain extent and improve the accuracy, but the method using nonlinear optimization requires a lot of computing resources.

In the follow-up study, we can try to use the self-interference algorithm [18-19] as the control algorithm or improve the existing fuzzy algorithm to self-organize and self-study the fuzzy rules [20]. And based on the exact model of the system, the optimization algorithm uses the traditional filtering [21-22] methods such as extended Kalman filter.

References

[1] Fu K S, Gonzalez R C, Lee C S G. Robotics: control, sensing, vision, and intelligence[J]. Robotica, 1987.

[2] Lauwers T, Kantor G, Hollis R. One Is Enough![J]. Proc of International Symposium of Robotics Research, 2005, 1(June):12--15.

[3] Kumagai M, Ochiai T. Development of a robot balancing on a ball[C]// International Conference on Control, Automation and Systems. IEEE, 2008:433-438.
[4] Kumaga M, Ochiai T. Development of a robot balanced on a ball: application of passive motion to transport[C]// IEEE International Conference on Robotics and Automation. IEEE, 2009:4106-4111.
[5] Hertig L, Schindler D, Bloesch M, et al. Unified state estimation for a ballbot[C]// IEEE International Conference on Robotics and Automation. IEEE, 2013:2471-2476.
[6] ZHANG Shao-kun. Linear quadratic optimal control of on-ball balancing mobile robot[D].Beijing: Beijing Information Science and Technology University.2013.
[7] LIU Feife，LIU Longx，GAO Tangpan. System Modeling and Self-balancing Control of A Single-ball Self-balancing Mobile Robot[J]. Machine Design and R esearch,2017,33(03):40-44+49.
[8] Chen Shibo. VImeel．side Dri C ving Caster Wheels Modular ontrol System [D].Nanjing. Southeast University.2012:121-128.
[9] Zhou Aiguo, Hong Jia. Modeling and Control Design for A Ballbot[J]. Mechatronics, 2012, 18(11):13-19.
[10] Song Junlie,Xiao Jun,Xu Xinhe. Modeling and Control Method of the Inverted Pendulum System [J]. Journal of Northeastern University,2002,23(4):46-50.
[11] Hu Jinming. Research of modeling and balance and motion control for a unicycle robot [D].Beijing:Beijing University of Technology, 1998:12-33.
[12] Li, Mingyang, Mourikis, Anastasios I. High-precision, consistent EKF-based visual–inertial odometry[J]. International Journal of Robotics Research, 2013, 32(6):690-711.
[13] Xiang Gao, Tao Zhang, Yi Liu, Qinrui Yan, 14 Lectures on Visual SLAM: From Theory to Practice, Publishing House of Electronics Industry, 2017.
[14] Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., & Furgale, P. (2015). Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, *34*(3), 314-334.
[15] Forster C, Carlone L, Dellaert F, et al. IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation (supplementary material)[J]. Georgia Institute of Technology, 2015.
[16] Huang, G. P., Mourikis, A. I., & Roumeliotis, S. I. (2009). A first-estimates Jacobian EKF for improving SLAM consistency. In Experimental Robotics(pp. 373-382). Springer Berlin Heidelberg.
[17] byJorgeNocedal, Wright S. Numerical Optimization[M]. Springer New York, 2006.
[18] Brown H B, Xu Y. A single-wheel, gyroscopically stabilized robot[C]// IEEE International Conference on Robotics and Automation, 1996. Proceedings. IEEE, 1996:3658-3663 vol.4.
[19] Juang H S, Lum K Y. Design and control of a two-wheel self-balancing robot using the arduino microcontroller board[C]// IEEE International Conference on Control and Automation. IEEE, 2013:634-639.
[20] Gu Y, Wang H O, Tanaka K, et al. Fuzzy control of nonlinear time-delay systems: stability and design issues[J]. 2001, 6..
[21] Chun-Xia L I. Improved algorithm of Kalman filter and its application[J]. Journal of Heilongjiang Commercial College, 2002.
[22] Carpenter J, Clifford P, Fearnhead P. Improved particle filter for nonlinear problems[J]. IEE Proceedings - Radar, Sonar and Navigation, 2002, 146(1):2-7.