A Novel Real-time Motion Control Approach for Omni-directional Mobile Robots

Conference Paper · July 2018

DOI: 10.23919/ChiCC.2018.8484052

5 authors, including:

Junkai Ren
University of Alberta
7 PUBLICATIONS **8** CITATIONS
SEE PROFILE

Junhao Xiao
National University of Defense Technology
9 PUBLICATIONS **16** CITATIONS
SEE PROFILE

Huimin Lu
National University of Defense Technology
62 PUBLICATIONS **293** CITATIONS
SEE PROFILE

A Novel Real-time Motion Control Approach for Omni-directional Mobile Robots

Junkai Ren[1], Wei Dai[1], Kuangye Xie[2], Junhao Xiao[1], Huimin Lu[1]

1. National University of Defense Technology, Changsha 410073, P. R. China
E-mail: jkren@nudt.edu.cn

2. Huazhong University of Science and Technology, Wuhan 430074, P. R. China
E-mail: ky2099@foxmail.com

Abstract: In highly dynamic environments, mobile robots are required to perform fast, accurate and stable tracking of high-speed trajectories. However, most research focus on the control algorithm for high-speed tracking, and rare works are reported which specifically deal with the communication problem and the structure of control system. This paper proposes a novel composed approach to improve the motion control performance and real-time reaction ability of mobile robot. First and foremost is to organize a hierarchical control structure for the robot motion control system. In this structure, the PD controller for trajectory tracking is improved based on the behavior of the robot. Secondly, in order to improve the communication speed, Process Data Object (PDO) is chosen as the communication object in CANopen communication protocol between EtherCAT and Elmo (motor controller). In the end, the pose estimation part of the robot is improved, and motor encoder data are used as feedback signal to adjust the robot's localization directly, thus providing more precise pose information. The improved motion control system illustrated in this paper was tested on a real omni-directional soccer robot, and the experimental results show that the robot can track high-speed trajectories effectively with small tracking errors and tiny communication time.

Key Words: Motion Control, Trajectory Tracking, High-speed, Mobile Robot

1 Introduction

In recent years, the risk of the fierce collision between robots increases in highly dynamic RoboCup Middle Size League (MSL) competitions [1]. As shown in Fig. 1, the robots in one team have to compete against the other team , and accomplish the task of attack, defense and cooperation. Every robot is supposed to track high-speed trajectories with sudden changes in direction and orientation where the robots have to elbow its way [2]. Along with the technique advance in the hardware and software of RoboCup MSL soccer robot, the maximum linear velocity and acceleration can reach 6 m/s and 2.8 m/s^2, and the maximum angular velocity can reach 6π rad/s.

Fig. 1: RoboCup 2016: NuBot VS TU/e

The robot's velocity and flexibility are becoming more important in MSL competition, especially when the two teams hold comparable intelligent level. The trajectory tracking ability appears to be increasingly important once the robot can move faster [2]. A well-behaved robot should move smoothly at high speed, avoiding sharp and frequent shake, thus can reduce the self-localization error from the camera shaking, which is beneficial for the robot to attack, avoid obstacles and fight against the opponent players.

Many new works based on the traditional methods have studied on using different controllers for tracking problem in recent years, [3] presented a model predictive control (MPC) scheme incorporating neural-dynamic optimization to achieve trajectory tracking of nonholonomic mobile robots (NMRs). [4] proposed a new robust trajectory tracking error-based control approach for unmanned ground vehicles.[5] studied neural indirect sliding mode method and adaptive sliding mode approach for the trajectory tracking control of mobile robots. The authors in [6] developed a leader-follower formation controller for networked unicycle-type vehicles. In [7], a nonlinear robust nonlinear controller was designed for uncertain quadrotors. In addition, some learning control methods were present in [8, 9].

However, seldom progress were made and illustrated about the control system structure and the communication efficiency of the robot system, which are both very important for the trajectory tracking problem in the real world.

In NuBot soccer robot [10], vision information from the omni-directional camera and odometry data from motor encoder were fused online to calculate it's pose. Similar works can be find in [11], which estimate the global position of the mobile robot online using natural visual features measured by a vision system and its orientation and velocity measured by odometry and attitude and heading reference system sensors. However, since the frame rate of the camera is only 30 FPS, which leads to a low update frequency of localization estimation, the whole system can not meet the requirement of good trajectory tracking ability.

Considering the problems mentioned, this paper proposes a novel composed approach in order to improve the robot's

This work is supported by National Natural Science Foundation (NNSF) of China under Grant 61773393 and 61503401.

trajectory tracking ability in complex competition environment [12]. First, a hierarchical control structure based on the robot's kinematics model is proposed, where the PD controller which generates the robot's expected velocity based on pose information is adjusted according to the robot's action policy. The second is increasing self-localization update frequency. The last is increasing the communication speed from motion controller to the motor. The real-time motion control approach presented in this paper has been developed in the application in the NuBot soccer robots of the RoboCup MSL, which specifically deals with highly dynamic and uncertain environment.

The paper is organized as follows: in Section II, an omnidirectional mobile robot is presented, and an action-based PD controller in the upper control layer of its control system is illustrated. Section III shows the methods used for improving the real-time motion control performance of the robot. A serious of experiments are developed and the results are analyzed in Section IV. Finally, the conclusions and future work are discussed in Section V.

2 Hierarchical Control Structure

The motion control [13] system of NuBot soccer robot adopts a hierarchical structure based on it's kinematics model: the upper layer completes trajectory tracking task and the lower layer deals with speed tracking task. The controller in this hierarchical control structure is relatively easy to design and expand. This section will optimize the motion controller of this system based on the robot's kinematics model, according to the competitive environment.

2.1 The NuBot Soccer Robot System

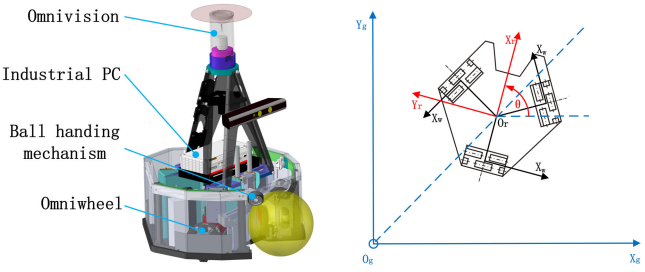

Fig. 2: The regular soccer robot of NuBot team

The NuBot Soccer Robot serves as the experimental platform of the proposed method, it is an omni-directional robot with three omniwheels [14]. The main parts of the robot and the omniwheels' layout are illustrated in Fig. 2. The wheel orientation in the body coordinate system is equal to 60 degrees, and each omniwheel is driven by a DC motor. Beckhoff and EtherCAT [15] modules are used for the data acquisition. These modules are connected to an industrial PC running all control algorithms. To be able to kick, a solenoid is designed. The ball handling mechanism enables the robot to catch and dribble a ball during the game. Besides the robot is equipped with omnivision which is used for self localization and obstacle detection.

2.2 The Robot Motion Control System

As shown in Fig. 3, the trajectory tracking part in the upper layer adopts a PD controller, which outputs expected speed V_1; the speed tracking part in the lower layer adopts a PI controller, which outputs expected speed V_2, then V_2 will be distributed into motor speeds for the three motors based on the no-sliding kinematics model. After that, the PI controller will output expected current command which ensures that the input electrical current to the motor keeps in the motor's safe range.

Besides, the motor speed control is implemented by Elmo (a digital servo motor driver). Since Elmo has high precision and real-time response efficiency, the time delay caused by this hierarchical structure could be reduced to a relatively low extent.

2.3 The Upper Adaptive PD Control Layer based on Robot Action

To analyze the upper layer of the hierarchical control system, a simple control loop is presented by simplifying the whole system. As shown in Fig. 4, the trajectory tracking controller is a simple PD controller, the input signal e of this controller is the robot's pose error, and the output is the expected velocity.

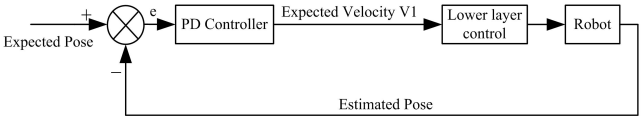

Fig. 4: The upper control layer based on PD controller

The traditional PD controller can be expressed as equation (1). From the experimental experience, we get that $K_p = 5$, $K_d = 2 \cdot K_p$.

$$u(t) = K_p \cdot e(t) + K_d \cdot \frac{de(t)}{dt} \tag{1}$$

However, during the robot soccer game, the robot needs to switch to different roles such as the attacker and defender according to the situation. Then relative actions such as avoiding the obstacle and searching the ball will be executed. Because it involves role switching and action choosing when the robot playing a game, which makes the robot's motion state usually changes [16], the control processes here have nonlinear, time-varying and uncertain properties. So it is very difficult for the traditional PD controller with fixed parameters to keep the robot moving smoothly. In this case, an adaptive PD controller based on the robot's real-time action is proposed.

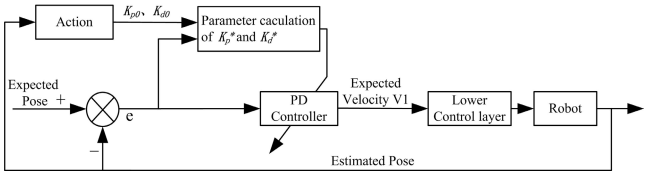

Fig. 5: The upper control layer based on adaptive PD controller

The upper control layer based on self-adaptive PD controller is showed in Fig. 5. A group of initial parameters (K_{p0}, K_{d0}) would be set according to the robot's behavior, then the robot's pose error is given to the regulator to regulate these parameters, which can help the control system achieve

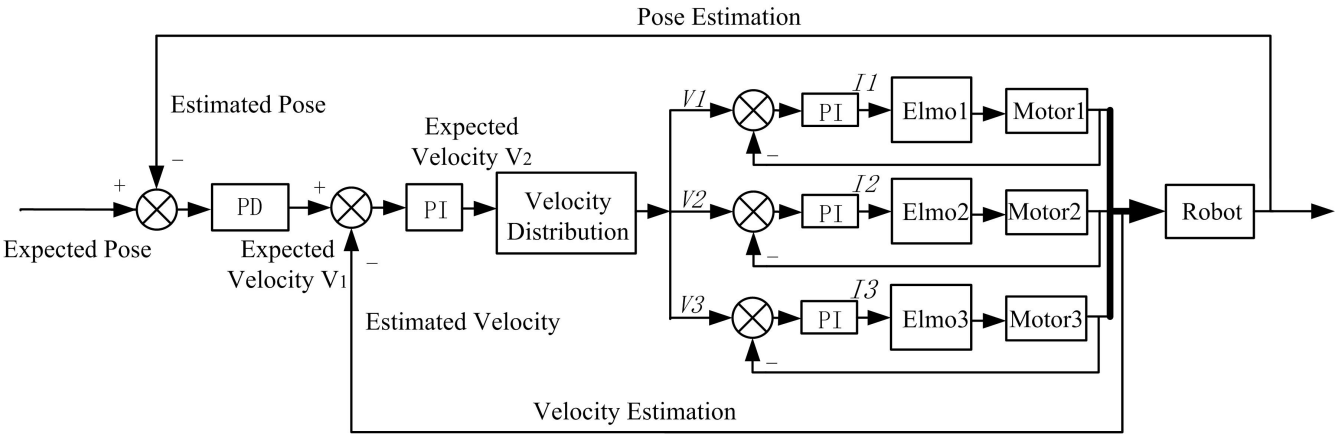

Fig. 3: The motion control system of NuBot soccer robot

Table 1: The initial PD parameters in calculating expected orientation speed

Action	Search Ball	Avoid Obstacle	Run with Ball
K_{p0}	5	1	2
K_{d0}	10	2	5

Table 2: The initial PD parameters in calculating expected linear speed

Action	Search Ball	Avoid Obstacle	Run with Ball
K_{p0}	3	1	3
K_{d0}	8	2	8

better control performance. The design of this adaptive PD controller can improve the robot's trajectory tracking ability and smooth it's moving trail. At the same time, it can reduce the moving oscillation and steady-state error.

The appropriate initial PD parameters based on different actions can be obtained from multiple experiments, and some of these values are listed in table1 and table2.

Supposing that α and β stand for the position error and orientation error respectively. Then if the value of α is large, the control system will need fast response performance in order to help the robot move towards the target as quickly as possible, so the value of K_p needs to be higher. However, because the estimated robot localization updates only at 30 Hz, which makes the response speed of the system slow, so the value of β will become lager, resulting in the appearance of frequent shaking and unsmooth motion. So it is necessary to reduce the value of K_p properly after β increases to a certain degree. At last, the relationship between the new parameter K_p^* and the initial parameter K_p can be written as equation (2). Similarly, the relation between K_d^* and K_d can be deducted and written as equation (3). (The constant term c_{13} and c_{23} are used to avoid zero denominator.)

$$K_p^* = c_{11} \cdot \alpha \cdot K_p + \frac{c_{12}}{\beta + c_{13}} \cdot K_p \tag{2}$$

$$K_d^* = c_{21} \cdot \alpha \cdot K_d + \frac{c_{22}}{\beta + c_{23}} \cdot K_d \tag{3}$$

To test the performance of this action-based adaptive PD controller, we designed an experiment as follows. We take the action of static positioning (usually happens after the command of kick off) as an example. The start and end coordinates of robot movement are set as (200 cm, 0 cm, 0) and (650 cm, 0 cm, π) respectively, which means the moving distance is 450 cm and the difference between the start and end orientation is π. During the experiment, we record the moving trajectories of the robot using adaptive PD controller and traditional PD controller.

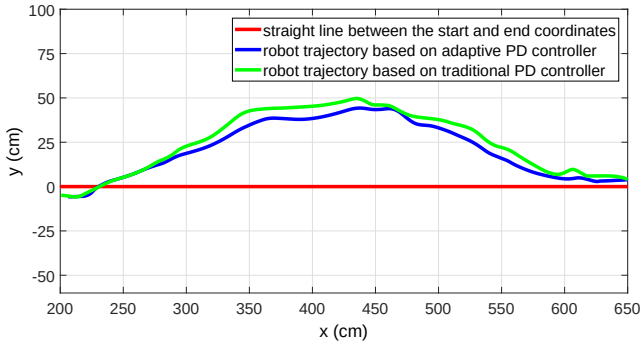

Fig. 6: The robot moving tragectories between two points

As illustrated in Fig. 6, the red line is a straight line connecting the start point and end point; the blue line and green line are moving trajectories when using adaptive PD controller and traditional PD controller respectively. It can be seen that the blue line is closer to the red line. This is because the parameters of the adaptive PD controller are adjusted according to the robot action. When the robot executes static positioning action, only the robots in our side can move on the field, the opponent robots are required to keep static, and no confrontation exists. Therefore, the velocity and acceleration could be properly reduced in order to get better trajectory tracking performance.

Similarly, the initial PD parameters are set according to other different behavior decisions chosen from competition condition, and then the initial PD parameters will be regulated by the regulator according to the real-time pose error. In the end, the newly calculated PD parameters will be used by the PD controller to control the robot's motion and improve the actual movement performance.

3 Improving Real-time Performance of Robot Motion Control

3.1 The Electrical and Control System of NuBot Robot

As a standard control system, the motion control section of NuBot soccer robot is composed of intact sensors, controllers, and actuators. The sensors include vision system, motor encoders and linear displacement sensors which were used to measure the ball-handling extent [17].

To improve the robustness of the robot control system, the current electrical system using PC-based control technology is designed [18], where all control and perception tasks can be carried out by a powerful central CPU and decentralized I/Os. Thus the electrical system is highly scalable. Also, the system adopts the Ethernet-based field bus system EtherCAT [18] and the Twin-CAT system to realize high-speed communication between the industrial PC and the connected modules.

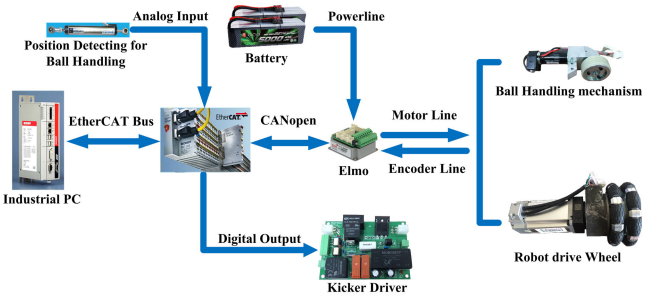

Fig. 7: The NuBot electrical system

The schematic diagram of the NuBot electrical system is illustrated in Fig. 7. All vision and control algorithms are run on the industrial PC, which communicates with the EtherCAT system via Ethernet. The Elmo Motion Control (SOL-WHI20/60) is the intelligent miniature digital servo drive for the brushless motor. The CANopen modular EL6751 embedded in the EtherCAT is used to realize communication between the industrial PC and the Elmo Motion Controls. The shooting module, which is named as kicker driver, is mainly composed of a relay and an IGBT FGA25N120ANTD. The PC can send control signals to the kicker driver for shooting or passing via the EtherCAT.

The sketch of the previous control system is depicted in Fig 8, and it can be divided into two layers. In the upper layer, the data from the motor encoder and omni-directional vision were fused to estimate the robot's current pose. Then the expected pose will be calculated through upper planner according to the robot's current pose and the competition state. After that, the robot's pose error will be used as the input signal to the upper controller to get the robot's expected speed; in the lower layer, the motion controller will transfer the expected speed into three motors' drive current commands and send them to motor drivers.

The maximum update frequency of the robot's expected speed in the upper layer is 30 Hz, which is because the updating rate of the robot's estimated pose is only 30 Hz. Supposing that the robot is moving at its maximum linear velocity (6 m/s) and maximum angular velocity (6π rad/s), the maximum linear distance and turning angle between every

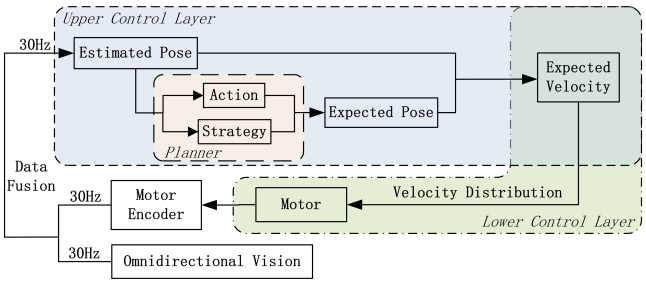

Fig. 8: The sketch of the previous control system

two updates are 0.2 m and 0.2π respectively. In this case, these two values can be lowered to be 0.06 m and 0.06π respectively if the update frequency reaches 100 Hz.

Therefore, when the update frequency of control system is relatively low, not only the linear distance error between two updates is large, but also the angular error.

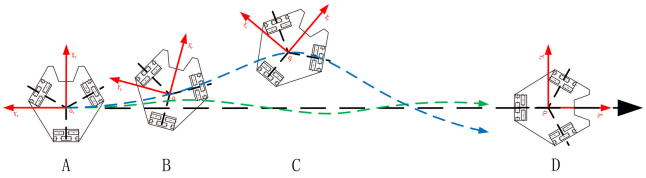

Fig. 9: The sketch of the robot's trajectories with different control frequency

As shown in Fig. 9, the start pose and end pose of the robot are set as A and D. The robot moves in a straight line and changes it's direction at the same time. When the update frequency of control system is low, the moving trajectory is the blue line, while when the update frequency is high, the moving trajectory is the green line. It can be seen that when the update frequency is low, the robot will deviate from the straight line severely, because it can not get velocity feedback and angle feedback in time, which is on the contrary when the update frequency is high. Therefore, it is particularly important to increase the update frequency of the control system and provide more specified pose feedback to the controller when the robot moves with high speed in such dynamic environments like the RoboCup MSL. In this case, we did the following two aspects of works:

- Adopting Process Data Object (PDO) in CANopen to increase the communication speed of the control system;
- Collecting the data from motor encoders to correct the robot's estimated pose directly.

3.2 PDO-based CANopen

The CANopen communication task uses two types of objects, which are service data object (SDO) and process data object (PDO). SDO is based on a client-server communication model and allows for direct addressing of an object using its index and subindex. SDO is specific to configuration and service, at the level of local nodes [19]. However, PDO provides an efficient data transmission according to a producer-consumer model. PDO is specific to real time data exchange, at the level of processes. The data length is lim-

ited to 8 bytes but no protocol overhead is contained. One PDO object can contain the values of more than one entry of the object dictionary, but the contents of a PDO have to be defined during the initialization.

NuBot soccer robot adopted SDO as it's communication objects before because it is a Java-based data programming model and architecture for accessing and updating the data. It is intended to give applications an easy-to-use programming model for accessing and updating the data, regardless the underlying source and format of the data. However, PDO is used for the configuration of a device, uploading and downloading large data blocks, so it requires an additional protocol overhead.

That is to say: PDO is unconfirmed, while SDO is confirmed; PDO allows broadcast, while SDO is point-to-point; PDO has a payload of up to 8 bytes, using as much as required, while SDO has a payload of 4 bytes for the expedited transfer.

In NuBot soccer robot, the master station (EtherCAT) needs to communicate with five follow stations (Elmo) so as to send current commands and receive rotation velocity feedback from motor encoders [10]. Considering the demands of dynamic competition environment and the composition of the NuBot electrical system, the new control system adopted PDO instead of SDO, thus the time consumed in one control period is less than 10 ms, and the communication rate between EtherCAT and Elmo can reach 100 Hz and even higher.

3.3 Improving the Structure of NuBot Control System

The new control system is depicted in Fig. 10. It includes two layers. In the upper control layer, the robot's pose error is used to calculate expected velocity. Here we replaced the former omni-directional camera which outputs images at 30 FPS by a new camera whose frame rate is 60 FPS, so we can get the robot's estimated pose from the fused data from motor encoders and omni-directional vision at 60 Hz.

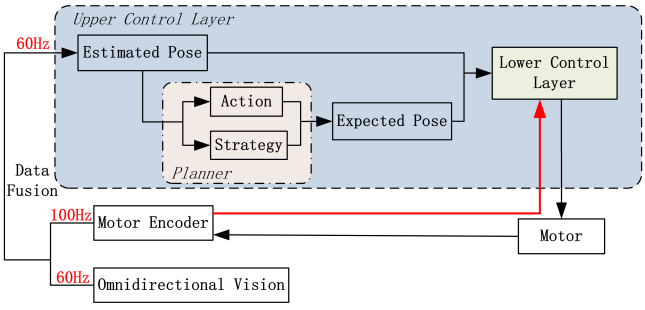

Fig. 10: The sketch of the new control system

In the lower control layer, the expected velocity is used to calculate the expected rotational speed of every motor and then drive motors running with the current command from Elmo. However, in order to provide more precise pose feedback to the upper layer, here the odometry data from motor encoder is utilized to calculate more accurate expected pose.

This new motion control system can provide estimated pose information with higher accuracy in time, which is helpful for the robot's trajectory planning of position and orientation adjustment. As a result, the robot can adjust it's velocity in time, which result in smoother trajectory.

4 Experiments

To verify the actual performance of the new control system, the following experiments are designed and executed.

4.1 Testing the communication speed of the control system based on PDO

The velocity command send from the industrial PC changes at 500 Hz. Then the velocity command is sent to the Elmo after being transferred into the current command for the motor. After that, the motor encoder will feed the rotation velocity back to the PC. In addition, the control system can receives feedback signals from motor encoders only when the speed changes.

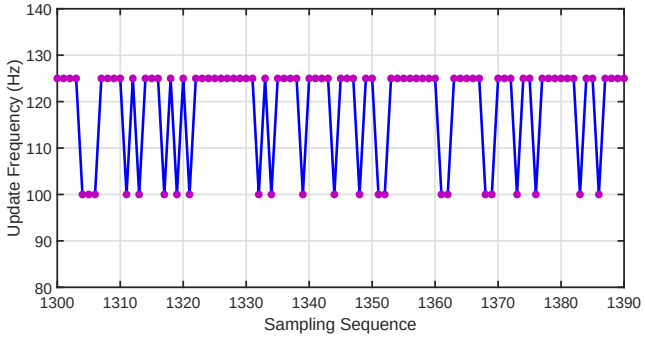

Fig. 11: The communication rate of the motion control system based on PDO

In this experiment, 3270 groups of time between every two changed motor rotation speed were recorded continuously. Then 90 groups of the data were extracted randomly to calculate update frequency of this new control system.

As shown in Fig 12, the communication speed can reach 125 Hz, and sometimes it drops to 100 Hz which is because of the data overflow. However, the overall communication speed of the new-designed control system is much higher than the previous which is only 33 Hz.

4.2 Testing the performance of adjusting the robot self-localization using odometry data

In this experiment, the robot moved from one point to another point with steady speed. The robot positions based on fused data and odometry data in the direction of y-axis were recorded respectively.

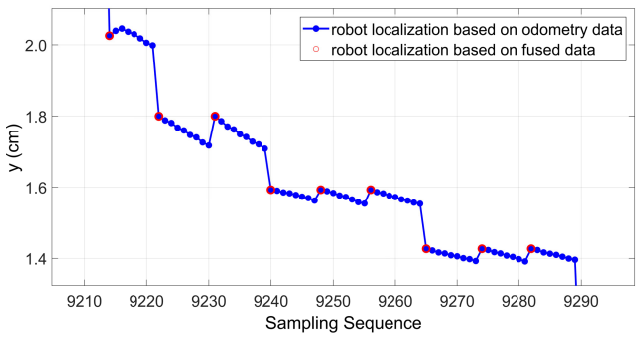

Fig. 12: The estimated positions based on fused data and odometry data respectively

The relation between the estimated positions based on fused data and odometry data is illustrated in Fig. 13. Red

points stand for fused positions, and blue points stand for odometry positions based on the former position. Because of the existing of camera positioning error, the fused robot position jumps from one point to another sharply. By feeding odometry data back to the upper control layer based on fused position, detailed position information can be calculated, which can help NuBot robot conduct real-time control.

4.3 Contrasting the performance of different control systems

In this experiment, the coordinates of the starting point and ending point were (0 cm, 0 cm, 0) and (400 cm, 0 cm, π). The robot needed to make linear motion and rotation motion at the same time. The moving trajectories of NuBot soccer robot adopting previous control system and new control system were recorded respectively.

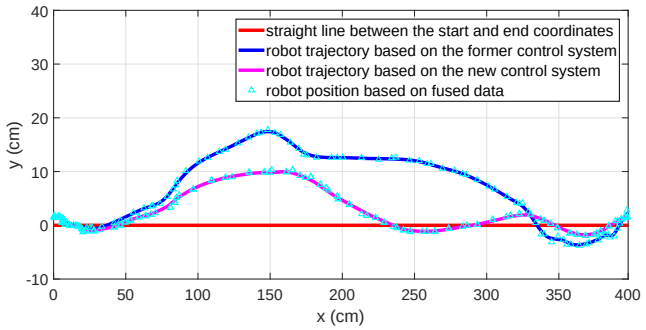

Fig. 13: Robot's moving trajectories based on different control systems

As shown in Fig. 13, the red line is a straight line from starting point to the ending point, the blue line is the trajectory based on previous control system, and the pink line is the trajectory based on improved control system.

It shows that the pink line is closer to the red line, which is because that the communication rate of control system was increased and the odometry data was adopted by the control system directly. Thus the control system could calculate estimated pose with higher frequency, and use these values to calculate expected velosity. At last, the robot could adjust velocity in time and generate smoother moving trajectory.

5 Conclusion

In summary, this paper illustrated the whole motion control system design of an omni-directional mobile robot which is based on the industrial electrical system. The results of comparative experiments verified the effectiveness of the new-designed control system. It can help high-speed mobile robot generate smoother trajectory which is particularly important for the robot self-localization and obstacle avoidance. In future work, we are going to focus on how to define the optimal trajectories [20] under different situation and implementing more robust control scheme based on the dynamic model of soccer robot.

References

[1] R. Soetens, R van de Molengraft, B. Cunha, Robocup msl-history, accomplishments, current status and challenges ahead, in *Robot Soccer World Cup*. Springer-Verlag, 2005: 624–635.

[2] Z.W. Zeng, H.M. Lu, Z.Q. Zheng, High-speed trajectory tracking based on model predictive control for omni-directional mobile robots, in *Proceedings of 25th Chinese Control and Decision Conference*, 2013: 3179–3184.

[3] Z. Li, J. Deng, R. Lu, et al, Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach, *IEEE Tran. Systems, Man, and Cybernetics: Systems*, 46(6): 740–749, 2016.

[4] E. Kayacan, H. Ramon, W. Saeys, Robust trajectory tracking error model-based predictive control for unmanned ground vehicles, *IEEE/ASME Tran. Mechatronics*, 21(2): 806–814,2016.

[5] F. G. Rossomando, C. Soria, R. Carelli, Sliding mode neuro adaptive control in trajectory tracking for mobile robots, *Journal of Intelligent and Robotic Systems*, 74(3–4): 931–944, 2014.

[6] X. Yu, L. Liu, Leader-follower formation of vehicles with velocity constraints and local coordinate frames, *Journal of Science China Information Sciences*, 60(7): 070206, 2017.

[7] T.P. He, H. Liu, S. Li, Quaternion-based robust trajectory tracking control for uncertain quadrotors, *Journal of Science China Information Sciences*, 59(12): 122902, 2016.

[8] Z. Chu, D. Zhu, S.X. Yang. Observer-based adaptive neural network trajectory tracking control for remotely operated vehicle. *IEEE Trans. neural networks and learning systems*, 28(7): 1633–1645, 2017.

[9] L. Tai, G. Paolo, M. Liu. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017: 31–36.

[10] D. Xiong, J.H. Xiao, H.M. Lu, et al, The design of an intelligent soccer-playing robot, *Industrial Robot: An International Journal*, 43(1): 91–102, 2016.

[11] K. Wang, Y. Liu, L. Li, Visual servoing trajectory tracking of nonholonomic mobile robots without direct position measurement, *IEEE Trans. Robotics*, 30(4): 1026–1035, 2014.

[12] L. Biagiotti, C. Melchiorri, Trajectory planning for automatic machines and robots, Spring-Verlag, Berlin, 2008.

[13] T. Tsubouchi, Motion planning for mobile robots in a time-varying environment: A survey, *Journal of Robotics and Mechatronics*, 8(1): 15–24, 1996.

[14] R. Rojas, AG. Förster, Holonomic control of a robot with an omnidirectional drive, *KI-Knstliche Intelligenz*, BöttcherIT–Verlag, 20(2): 12–7, 2006.

[15] Beckhoff, EtherCAT, Ultra high-speed communication, *http://www.beckhoff.de/english.asp?ethercat/default.htm*, 2009.

[16] C.T. Leng, Q.X. Cao, Motion planning for omni-directional mobile robots based on anisotropy and artificial potential field method, *Industrial Robot: An International Journal*, 36(5): 477–488, 2009.

[17] J.K. Ren, C.G. Xie, J.H. Xiao, et al, A control system for active ball handling in the RoboCup middle size league, in *Proceedings of 28th Chinese Control and Decision Conference*, 4396–4402, 2016.

[18] D. Jansen, H. Buttner, Real-time Ethernet: the EtherCAT solution, *Computing and Control Engineering*, 15(1): 16–21, 2004.

[19] K. Erwinski, M. Paprocki,L. M. Grzesiak, et al, Application of ethernet powerlink for communication in a linux rtai open cnc system, *IEEE Transactions on Industrial Electronics*, 60(2): 628–636, 2013.

[20] D. Bruijnen, J. van Helvoort, Realtime motion path generation using subtargets in a rapidly changing environment, *Journal of Robotics and Autonomous Systems*, 55(6): 470–479, 2007.