Motion compensation of omnidirectional wheel robot using neural networks

Yonghai Wu
College of Mechanical and Energy Engineering
Zhejiang University
liunian@zju.edu.cn

Zhenfu Yuan
College of Mechanical and Energy Engineering
Zhejiang University

Abstract

This paper describes a method to compensate the omnidirectional robot's motion control. Due to the variation of individual mechanical characteristics, there always have some errors between the given command and the robot's execution.The paper explains how we cope with the execution errors by two neural networks modeling the rotation and translation individually.And we have successfully field-tests the compensation method at several RoboCup events with our ZjuNlict team. The results shows that the compensation can significantly improve the accuracy of robotic play.

1 Introduce

An omnidirectional wheel can highly improve the mobilities of a vehicle robot. The robot equipped with omnidirectional wheels is able to rotate while simultaneously translating in a free direction. So it can move along a straight path between any two points without having to rotate first, instead of rotating and translating in different time. This is useful in some dynamic and competitive environments [7].

There are many different types of omnidirectional wheel[3–5, 7].But they are all based on the same general principle: while the wheel proper provides traction in the direction normal to the motor axis, it can roll passively in the direction of the motor axis. And it is also called orthogonal wheels. Figure 1(a) is an example of this wheel we design for our F180 team in the SSL competition. There are 30 smaller passive wheels attached along the periphery of the main wheel.Compare to the other,this kind of wheel's distance between the assembly and the geometrical center of the robot can keep a constant(Figure 1(b)) when it roll on the ground. It can drive the robot move smoothly and also greatly simplify the process of deriving the kinematics.

The current motion control methods [3–5, 7] for this kind of omni-wheel robot are almost base on the suppose that there is not friction between the small passive wheels and the main wheel. As the friction is very small, this suppose is acceptable. ignored .Using this suppose, the translation velocity of the robot can be transformed to the main wheel's rotational velocity. Because most papers are introduce the control method for three wheels robot not for four wheels which is used in this paper, we will introduce the four wheel robot's motion control using current method in the following section.

Due to the special structure of the wheel, the robot is sensitive to the precise of machining and assembly. A little error may bring some unstable factors into motion control. Such as, some of the small passive wheels can rotate smoothly, or the four omni-wheels can touch the ground at the same time. In fact these situations are very common. This is the reason that it is hard to drive this kind of robot fast as well as accurately . we propose a method to compensate the robot motion control by using neural networks. Because the rotation movement and translation movement of the omni-wheel robot are independent, two neural networks are used to compensate the two kind of motion individually.

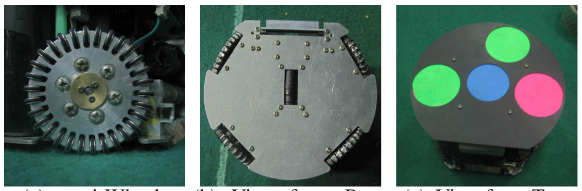

(a) omni-Wheel (b) View from Bottom (c) View from Top

Figure 1. (a) Our robot's omnidirectional wheel. (b) This display the assembly of our omni-wheels. (c) The mark on the top of our robot.With it the overhead camera can identify our robot's position and orientation.

And we use ZjuNlict[1],a RoboCup [2] Small Size

[1]http://www.nlict.zju.edu.cn/robocup2006/welcomepage.html
[2]RoboCup is an international joint project to promote AI, robotics, and related field. www.robocup.org

League(SSL) team,as the research platform. SSL, also known as F180 league,is one of RoboCup league divisions.In a SSL soccer match, both teams have five robots,each of which must physically fit inside a cylinder with a diameter of $180mm$ and a height of $150mm$.The competition field is a $4.9 \times 3.4m$ rectangular,with an orange golf ball acting as the soccer ball.Teams use the global overhead cameras as their main sensor.The cameras can locate robot's position and orientation by identify the color mark (Figure 1(c)) on the top of robots. We also use this global vision system to record robot's position in following experiments.

This compensation method greatly improve our control effect.This help us win the runner-up of China RoboCup 2004 SSL competition,and be qualified RoboCup 2004,2005 and 2006 successively[6].At RoboCup 2006 hold in Bremen (14-20, June), we are one of the *BEST EIGHT*. This is the best performance for Chinese teams.

The following section introduce the motion control method for four omni-wheel robot.Then we illustrate the method of motion control we design to compensate the error between command sent and the robot's execution,and also present the experimental result. Finally,we summarize the work of this paper and discuss the future work.

2 Current motion control for four omni-wheels robot

We use four omni-directional wheels to drive our robot. The wheel distribution is shown in Figure 1(b). In order to explain the velocity transformation more clearly, we present a diagram (Figure 2), which represents the omni-directional wheel's contact points relative to the geometrical center of the robot, and the wheel drive direction angles(ω_i) relative to the forward direction ($\mathbf{X}\ axis$) of the robot. The following explains the other symbols: (v_x, v_y) denote the robot's translate velocity, v_θ denotes the rotation velocity around the robot's geometry center, L denotes the distance between the wheel's center to the robot's geometry center, and W_i denotes the wheel's speed.

Base on the suppose that there is no friction between the small passive wheel and the main wheel,we can get two kind of velocity transformation.One is robot integer velocity to robot's wheel velocity, and the other is transformation from four omni-wheels velocity to robot integer velocity.

2.1 Velocity transformation from velocity of the geometric center of the robot to the four wheels

Use the method presented in [2] we can obtain the coordinate transformation (1) between the three-component vector velocity of the geometric center of the robot,represented by $[V_x, V_y, V_\theta]^T$,and the four wheel velocities,$[W_1, W_2, W_3, W_4]^T$.

$$\begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = \begin{bmatrix} cos(\omega_1) & sin(\omega_1) & L \\ cos(\omega_2) & sin(\omega_2) & L \\ cos(\omega_3) & sin(\omega_3) & L \\ cos(\omega_4) & sin(\omega_4) & L \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_\theta \end{bmatrix} \tag{1}$$

If we want the robot move at any given velocity vector $[V_x, V_y, V_\theta]^T$,we just need send the command $[W_1, W_2, W_3, W_4]^T$ to the four wheels.

2.2 Velocity transformation from velocity of the four wheels to the geometric center of the robot

As for our robots,the wheel drive direction angles are

$$[\omega_1, \omega_2, \omega_3, \omega_4]^T = [2.3911, -2.5307, -0.7854, 0.7505]^T$$

and the distance between the assembly and the geometrical center of the robot is

$$L = 8cm$$

Let us denote the transformation matrix in (1) by B,therefor

$$B = \begin{bmatrix} -0.7314 & 0.6820 & 8.0000 \\ -0.8192 & -0.5736 & 8.0000 \\ 0.7071 & -0.7071 & 8.0000 \\ 0.7314 & 0.6820 & 8.0000 \end{bmatrix} \tag{2}$$

if the inverse of $(B^T B)$ exists, the Moore-Penrose Matrix Inverse [1] of B,let us denote it by B^+,is just the transformation matrix which we need:
because

$$rank(B^T B) = 3 \tag{3}$$

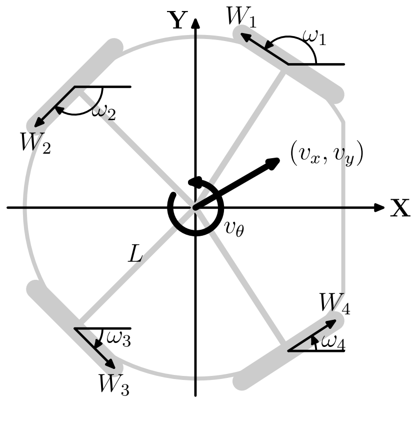


Figure 2. Robot Geometry used in Velocity Transformation.

therefor, the inverse of $(B^T B)$ exists. So the velocity transformation matrix is B^+

$$\begin{bmatrix} V_x \\ V_y \\ V_\theta \end{bmatrix} = B^+ \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} \tag{4}$$

here B^+,the Moore-Penrose Matrix Inverse of B, is:

$$\begin{aligned} B^+ &= (B^T B)^{-1} B^T \\ &= \begin{bmatrix} -0.3097 & -0.3578 & 0.3234 & 0.3441 \\ 0.3713 & -0.3439 & -0.4091 & 0.3816 \\ 0.0292 & 0.0309 & 0.0334 & 0.0315 \end{bmatrix} \end{aligned} \tag{5}$$

For any given wheels velocity vector $[W_1, W_2, W_3, W_4]^T$,the robot will move at the velocity vector $[V_x, V_y, V_\theta]^T$.

3 Motion compensation

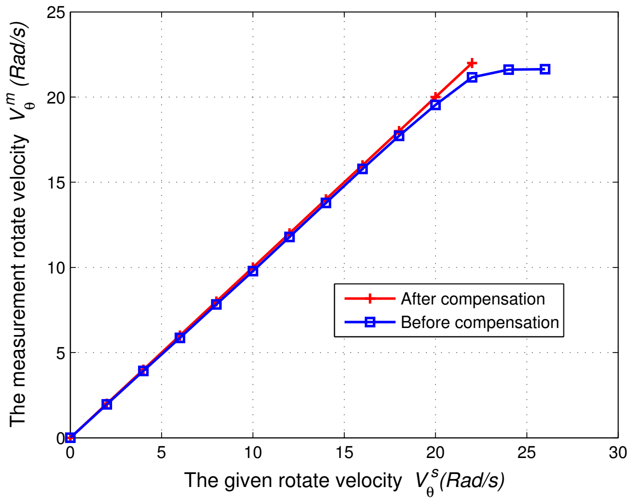
Use the formula (1),(4), robots can be controlled at any given velocity vector(V_x, V_y, V_θ) in theory.But in practice, if the robot is controlled by the given command without any compensation under open loop control conditions, it is hard for the robot to drive at the given direction,velocity,and angular velocity due to the variation in individual mechanical characteristics. Such as the four wheels maybe can touch the ground at the same time or some small passive wheels can rotate smoothly. These can't be avoid due to the the precise of machining and assembly. But fortunately these factors are inherent after the robot is built. So we can use neural networks to model the errors between the given command and the robot's real velocity. Then these models are used to compensate the motion control of the robot.

In order to get the data for the training of the neural networks, we use the camera mounted overhead of the field to record the robot's position and orientation which the precise is below $1cm$ and $3°$ individually.

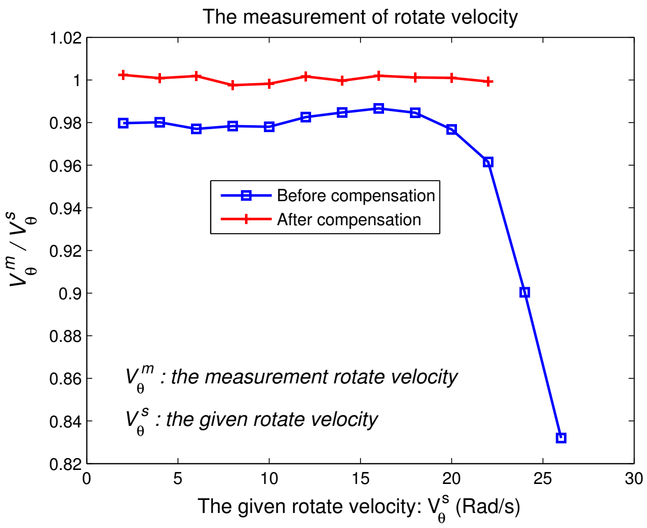
After measuring the errors between the given command and the velocity the robot really execute, we find that the errors are almost identical for the same given command, just like our expect. As mentioned in the introduce section and our relative research [6],we assume the translational and rotational invariance.This means that we can do the translational compensation and rotational compensation individually,which highly simplify the compensation problem.

3.1 Rotational compensation

we only give four wheels at a equal speed (include direction),then the robot can get a rotate velocity V_θ around it's geometrical center.



(a) First part of rotation compensation



(b) Second part of rotation compensation

Figure 3. The measurement of rotate velocity and the compensation result

3.1.1 Neural network architecture

We use a very simple network architecture with only 1 input vector ,1 output vector and 5 hide nodes.

The input data is the rotate velocity measured by the vision system V_θ^m and the output is corresponding command V_θ^s send to the robot.We train the neural network using the standard back-propagation algorithm.

3.1.2 Data Collection and Constraints

If the collected data can cover all regions of the input space,the neural network trained by these data will have a strong generalization ability. We increase the given rotate velocity V_θ^s between $2Rad/s$ and $26Rad/s$ with a step of $2Rad/s$.

In order to decrease the errors of measurement, we repeat the measurement 3 times and use the average of the 3 measurement results as the input data.The measurement result is shown in Figure 3(a).In this figure, we can tell that the max rotate velocity of the robot is $22Rad/s$. So the input scope of the neural network is $[0, 22]Rad/s$.

3.1.3 Compensation result

Figure 3 shows the experimental result of the compensation. This prove that our compensation method is better than the current control method. Because the small passive wheels' speed are all equal to 0 during the rotation movement, the result also shows that our robot's main wheels' speed coincide with the given well.

3.2 Translational compensation

3.2.1 Neural network architecture

In this case ,we use a 2 input and 2 output architecture,the number of hide nodes is 10.

The input vector is $(|V|, \theta)$,which is the polar coordinate of (V_x, V_y).The relation between this two can be described by such equation:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} |V|cos(\theta) \\ |V|sin(\theta) \end{bmatrix} \tag{6}$$

We also train the neural network using the standard back-propagation algorithm.

3.2.2 Data Collection and Constraints

We use the current control method to drive the robot at a given command, then use the camera tho record the track. We use these data to get the robot's real velocity. Our robot can run at a maximum translational velocity is under $200cm/s$.To cover all the regions of the input space of $(|V|, \theta)$,We increase $|V|$ from $20cm/s$ to $200cm/s$ with a step of $20cm/s$. And at any given $|V|$,the translational velocity's direction in local coordinate changes between $0°$ and $360°$ with a step of $15°$.As it is shown in Figure 4.This means we must measure $24 \times 10 = 240$ translational velocities.To simplify the measure process,we have developed a program to complete the measurement process automatically. To decrease the measurement errors we also repeat the total measurement 3 times and use the average as the input data.

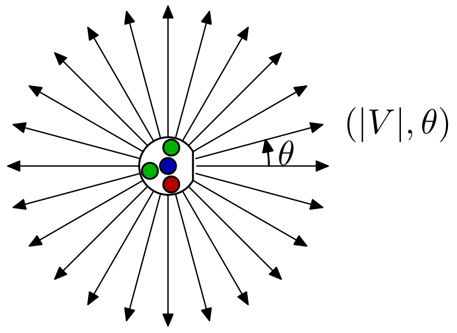


Figure 4. The measurement experiment for translational compensation

Figure 5 shows the result of the measurement.We can see the big errors between the given commands and the executions.

3.2.3 Compensation result

Using the trained neural network, the motion of our robot improved greatly just as we expected.Because there are too many input data, we can't display them all like rotational compensation. We chose some of them to show the compensation result at

$$|V| = 120cm/s, \theta = 0°, 15°, \cdots, 345°$$

Figure 5 shows the difference between the current control method and our compensation method.

4 Conclusion and further work

Due to the variation in individual mechanical characteristics, the robot can't execute the given command correctly. In this paper we using neural network to compensate our robot's motion control and make use of them in the testing field of RoboCup Small Size League. As a result these compensations greatly improve the precise of motion control. Our team benefit a lot from this improvement. A further work would be to integrate the neural networks using in the compensation into a simulator as the robot's model.We can

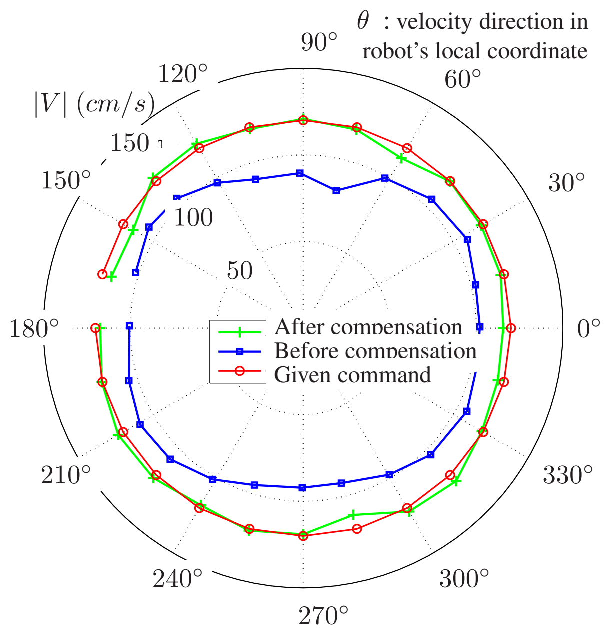


Figure 5. Translational compensation at $|V| = 120cm/s$

test our new strategies on the simulator without interacting the external world.

References

[1] A. Ben-Israel, T. Greville, and T. Greville. *Generalized Inverses: Theory and Applications*. books.google.com, 2003.

[2] K. Byun, S. Kim, and J. Song. Design of a four-wheeled omnidirectional mobile robot with variable wheel arrangement mechanism. *Proceedings of 2002 IEEE International Conference on Robotics and Automation (ICRA)*, 2002.

[3] R. Damoto, W. Cheng, and S. Hirose. Holonomic omnidirectional vehicle with new omni-wheel mechanism. *Proceedings of 2001 IEEE International Conference on Robotics and Automation (ICRA)*, 2001.

[4] D. Feng, M. Friedman, and B. Krogh. The servo-control system for an omnidirectional mobile robot. *Proceedings of 1989 IEEE International Conference on Robotics and Automation (ICRA)*, 1989.

[5] F. Pin and S. Killough. A new family of omnidirectional and holonomic wheeled platforms for mobile robots. *IEEE Transactions on Robotics and Automation*, 1994.

[6] Y. Sheng and Y. Wu. Motion prediction in a high-speed, dynamic environment. *Proceedings of 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 2005.

[7] J. Tang, K. Watanabe, and Y. Shiraishi. Design and traveling experiment of an omnidirectional holonomic mobile robot. *Proceedings of 1996 IEEE International Conference on Intelligent Robots and Systems (IROS)*, 1996.