ATLANTIS PRESS Journal of Robotics, Networking and Artificial Life, Vol. 4, No. 4 (March 2018) 275–282

Motion Improvement of Four-Wheeled Omnidirectional Mobile Robots for Indoor Terrain

Amornphun Phunopas[1]
Department of Production Engineering, King Mongkut's University of Technology North Bangkok, 1518 Pracharat 1 Road Bangsue, Bangkok, 10800, Thailand[1]

Shinichi Inoue[2]
Department of Interdisciplinary Informatics, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka Fukuoka, 820-0067, Japan[2]
E-mail: amornphun.p@eng.kmutnb.ac.th, inoue.shinichi.1800@gmail.com
www.kmutnb.ac.th

Abstract

The four-wheeled omnidirectional platform is great to use for an indoor mobile robot. It can increasingly move and change heading directions. However, the robot is easy to slip when it is moving. One or more wheels are sometimes not touching the ground. This paper approaches to solving these problems by computational simulation in locomotion. The mathematical models simulate the robot movement. The robot struggles to go to the target with randomly simulated slips and untouched ground circumstances. The robot can estimate the positions using the Kalman filter and readjust itself to the planned path. Consequently, this paper demonstrates the motion improvement and compares the results of decreasing errors.

Keywords: Omnidirectional wheel, Indoor mobile robot, Kalman filter, Motion improvement

1. Introduction

It is known that an omnidirectional mobile robot is a holonomic motion. It can travel in multi-directions and change direction rapidly. This platform is famous for use in the automated guided vehicles (AGVs), which use is growing in service robots.[1] The robots can support home care. The robots can flexibly add on many functions for works or for human assistance.

The omnidirectional mobile robot has many wheels that are placed in different directions. It has a complicated mathematical model to control the movement of robots.[2] There is a constraint when moving on a flat, indoor terrain. However, it is difficult to move the robot to the desired position autonomously because the robot's motion is slippery. Therefore, the robot may miss its planned path or, in the worst-case scenario, get lost due to unrecognized circumstances.

There are several techniques to solve the robot motion slippery of the omnidirectional type wheel. The design performance of the four-wheeled Omni wheelchair, with a suspension mechanism that utilizes a hydraulic shock absorber, was evaluated.[3] The results claimed that the proposed mechanism design had lesser wheel slippage and vibration than the other existing designs. Moreover, the omnidirectional mobile can be used with the wheelchair platform, which added the human upper body model to the robot model.[4] Normally, the dynamic model and modeling control methods are implemented in the system. The motion control has a dynamic model of three Omni wheels, which is formulated to stabilize the programmed motion and to track the trajectory of a robot. The performance investigations are based on the direct Lyapunov method and its extensions. Andreyev et al. created control laws and simulated it by using the numerical method.[5] Similarly, the problem of trajectory tracking for a four-wheeled omnidirectional robot is

solved by obtaining the nonlinear error, which is brought to a quasi-Linear Parameter Varying (LPV). The simulation results showed the effectiveness of switching LPV control, using Linear Matrix Inequalities (LMI)-based techniques in the tracking of the desired circular trajectory.[6] The utilized trajectory can be generated for obstacle avoidance; the robot can travel in smooth vehicle velocities near obstacles. Galicki applied the Lyapunov stability theory, which is used to derive the control scheme.[7] The simulations of an omnidirectional vehicle had collision-free movement with one obstacle and in cluttered task space. Flippo et al. developed a prediction algorithm in a skid steering turn[8] to validate the use of single wheel testing to predict the full four-wheel vehicle system and to control the side-slip angle estimation based on lateral dynamics control with optimal steering angle and traction/brake torque distribution.[9] Both trajectory tracking and stabilization can be synthesized via the well-known adaptive backstepping approach for the dynamic models of nonholonomic mobile robots with dynamic effect and uncertainties.[10] The robot that can localize or recognize a shifted position can adjust itself to the correct position. In fact, the robot has sensors to refer to the absolute position. An omnidirectional mobile platform has a closed-loop control scheme to control three degrees-of-freedom motions. Cooney et al. adapted the optical mice as a sensor providing dead-reckoning for closed-loop PID control.[11] Huang et al. have built in the electronic compass and the gyroscope sensors to measure the mobile platform azimuth and feedback for platform orientation control.[12] They control the robot by knowing the robot rotation and orientation, relating to an intelligent fuzzy sliding mode controller (FSMC) for every wheel. The robot has the suspension system of Omni wheels to prevent non-ground contact on a unsmooth surface for better control. Research used a fault-tolerant odometry feedback using IMU, stereo camera, and laser scanner to recognize self-position and track the desired trajectory.[13] Sanada et al. used a camera as a velocity sensor to calculate optical flow.[14] The correct position is obtained by the optical flow sensor as opposed to the dead reckoning method. To solve the slip problem when a vehicle traverses over soft soil, use a downward-looking camera to capture images that contain both the soil surface and the wheel tire surface.[15] The wheel slip can be estimated using the vision-based optical flow algorithm without wheel odometry information. The vision-based perception using an omnidirectional mirror can obtain a panoramic view, and the omnidirectional mobile robot can then move toward desired targets and avoid the obstacles. The particle swarm optimization (PSO)-learning algorithm is proposed to generate fuzzy decision rules[16] automatically. Odometry sensor is not suitable for such drives because of wheel slippage. Kundu et al. used a single camera for 360-degree scanning of multiple markers in the area.[17] The markers were based on an augmented reality localization platform. The system required a scanning time of 5 seconds, and the robot had to stop traversing. The average error ranging was from 2.5cm to 5cm in a test area of 5m X 5m. Active Beacon System is a radio frequency signal transceiver that has multiple ultrasonic sensors which transmit a package to every beacon. It is used to determine the absolute position of the robot.[18] Algorithms are used to control the robot by reducing errors. For real implement in the robot, Hashemi et al. use a model-based PI-fuzzy control for four-wheeled RoboCup Small Size League omnidirectional robot.[19] The robot had a path planner and associated low-level control system to satisfy planning prerequisites and prevent slippage with velocity and acceleration filtering. The robot can generate a real-time trajectory by constrained dynamic inversion.[20] Another idea to improve the trajectory tracking of mobile robots is to use neural networks. The wheel slip error and external disturbance forces were defined as uncertainty.[21] The modified backpropagation neural networks have online weight updating laws for robustness. The boundary of system stability is proven using the Lyapunov method. The tracking error can be reduced according to the theoretical analysis of tracking a straight line and a U-shape trajectory.

The four-wheeled omnidirectional mobile robot has four points to make contact with the floor. Its geometry is a rectangular shape with four connecting points, which are supposed to align all wheels in the same plane. In fact, the floor is not completely flat, and after running the robot many times, each wheel will become eroded unequally. The robot becomes unbalanced or shaky even though all wheels are designed and aligned in the same plane. It is possible that some wheels will not touch the floor when it moves. It is possible that the robot will get stuck and not reach its goal. These problems can be solved by mechanically designing it for rough terrain conditions.[22] For example, the omnidirectional mobile robot employs a suspension system[23] for supporting all wheels to touch the floor in order to cancel out the mechanical noise.

The computational algorithm can reduce error from noises and make the robot move on its planned path. Dead reckoning is the motion estimation of the robot base on speed estimation, direction, and traveling time.[24] The Kalman filter is great to use in noisy systems such as localization and navigation.[25]

This research is about to use the Kalman filter to solve the slip without the wheel touching the ground. The robot will use sensors and computation to resolve the problems. The research is done in simulation by creating the possible conditions of motion randomly using MATLAB. The first section is about the kinematics model; next section is the trajectory to define the state of motion. After that, the simulation section describes the steps of simulation and the conditions of motion, and then the motion improvement section uses the Extended Kalman Filter and shows the simulation results. Finally, conclusion section discusses and concludes the results from the simulation.

2. Kinematics

The omnidirectional mobile robot uses holonomic motion and has four wheels to move in multiple directions. It can be controlled through the planned path in three degrees of freedom. There are two kinematics models: the first is forward kinematics, and the second is inverse kinematics. The robot platform uses the forward kinematics to determine the robot's velocities $(V_x, V_y, \dot{\theta})$ and the inverse kinematics to determine the wheel's angular velocities (w_1, w_2, w_3, w_4). The robot motion is in the XY coordinate as in Fig. 1. The robot's heading angle is determined by θ. The robot can move through the planned paths of motion from the kinematics models. As such, there are eight basic directions for omnidirectional movement in Table 1. Each wheel turns in different directions. CW is clockwise, and CCW is counterclockwise.

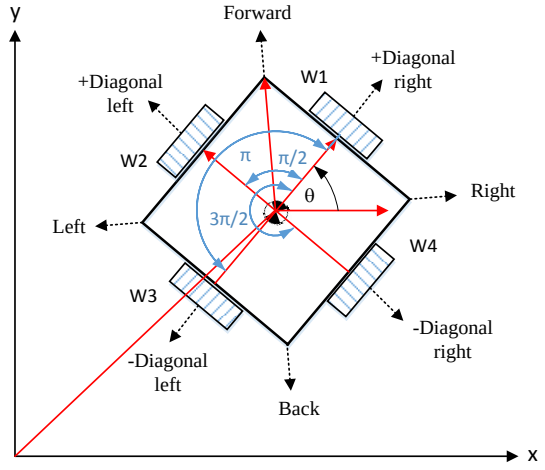

Fig. 1. The robot's configuration and eight directions of motion.

Table 1. Eight directions of the robot's motion relate to the four wheels' rotation direction.

Direction	W_1	W_2	W_3	W_4
Forward	CW	CCW	CCW	CW
Left	CW	CW	CCW	CCW
Right	CCW	CCW	CW	CW
Back	CCW	CW	CW	CCW
+Diagonal Right	0	CCW	0	CW
-Diagonal Right	0	CW	0	CCW
+Diagonal Left	CW	0	CCW	0
-Diagonal Left	CCW	0	CW	0

2.1. Forward Kinematics

The input of forward kinematics regards the wheels' speed, and the output regards the robot's velocities as in Eq. (1) to simulate the robot's motion. Converting the four wheels' speed into the robot's linear velocity requires the Jacobian matrix as seen in Eq. (2). The parameters R and r are set to 10 cm and 5 cm respectively for the simulation.

$$V_r = \begin{bmatrix} V_x \\ V_y \\ \dot{\theta} \end{bmatrix} = J \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \tag{1}$$

$$J = \frac{r}{2} \begin{bmatrix} -sin\left(\theta + \frac{\pi}{4}\right) & cos\left(\theta + \frac{\pi}{4}\right) & \frac{1}{2R} \\ -sin\left(\theta + \frac{3\pi}{4}\right) & cos\left(\theta + \frac{3\pi}{4}\right) & \frac{1}{2R} \\ -sin\left(\theta + \frac{5\pi}{4}\right) & cos\left(\theta + \frac{5\pi}{4}\right) & \frac{1}{2R} \\ -sin\left(\theta + \frac{7\pi}{4}\right) & cos\left(\theta + \frac{7\pi}{4}\right) & \frac{1}{2R} \end{bmatrix}^T \tag{2}$$

Where W_1 is the angular velocity of wheel 1, W_2 is the angular velocity of wheel 2, W_3 is the angular velocity of wheel 3, and W_4 is the angular velocity of wheel 4. R is the radius of the robot from the center to the wheel, r is the radius of the wheel, V_x is the robot's velocity on the x-axis, V_y is the robot's velocity on the y-axis, and $\dot{\theta}$ is the heading angular velocity of the robot.

2.2. Inverse Kinematics

To inverse Eq. (1) will obtain Eq. (3), which provides the robot's velocities.

$$V_w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} -sin\left(\theta + \frac{\pi}{4}\right) & cos\left(\theta + \frac{\pi}{4}\right) & R \\ -sin\left(\theta + \frac{3\pi}{4}\right) & cos\left(\theta + \frac{3\pi}{4}\right) & R \\ -sin\left(\theta + \frac{5\pi}{4}\right) & cos\left(\theta + \frac{5\pi}{4}\right) & R \\ -sin\left(\theta + \frac{7\pi}{4}\right) & cos\left(\theta + \frac{7\pi}{4}\right) & R \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ \dot{\theta} \end{bmatrix} \tag{3}$$

The planned path of the robot's velocities is assigned as input parameters to obtain the speed of four wheels.

2.3. Trajectory model

The robot can move smoothly. The trajectory is for path planning, which determines speed and acceleration of each wheel. The trajectory advantage is to increase and slow down velocity properly. The robot can rotate to change directions while it is moving. Accordingly, the trajectory of path planning will be applied to the inverse kinematics and then control the speed of four wheels. In reality, the robot has to have odometry sensors for feedback control to control the wheels' speed and the robot's direction. For simulation, the velocity of four wheels can be applied to drive the animated robot. The trajectory uses a polynomial function of time. Polynomials are simple to compute and can provide the required smoothness and boundary conditions using Peter Corke's MATLAB library.[25] A quintic (fifth-order) polynomial is used,

$$S(t) = At^5 + Bt^4 + Ct^3 + Dt^2 + Et + F \quad (4)$$

$$\dot{S}(t) = 5At^4 + 4Bt^3 + 3Ct^2 + 2Dt + E \quad (5)$$

$$\ddot{S}(t) = 20At^3 + 12Bt^2 + 6Ct + 2D \quad (6)$$

where S is function of robot motion (x, y, θ), $\dot{S}$ *is derivative of S as robot velocity* $(V_x, V_y, \dot{\theta})$, $\ddot{S}$ *is derivative of* $\dot{S}$ *as robot acceleration* $(a_x, a_y, \ddot{\theta})$, *t is interval time in step sampling, and (A,B,C,D,E,F) are coefficients.*

3. Simulation

The simulation is implemented by a kinematics model and a trajectory model. The planned path is created for the robot motion, a straight line and a curved line. The trajectory is ready to use to control the robot's position, velocity, and acceleration using Eq. (4-6). Then, the robot converts its body velocity to four wheels' speed using Eq. (3). To display the robot's motion, in animation, see Eq. (1). The Kalman filter is used to improve the robot's motion and simulate every condition of motion as in Fig. 2.

3.1. Conditions of Motion

The robot may not move as the planned path because of friction and external force. However, the robot normally has the feedback control to compensate for the motor's speed to maintain the speed input and direction of the robot. Another problem is the robot possibly getting stuck and being unable to move out from the trap field. However, the wheels still continue as the assigned wheels' speed.

There are two effects from changing the conditions of motion randomly. First, the robot position shifts from the planned path, and second, the robot cannot reach the goal's position. There are eight conditions of motion as

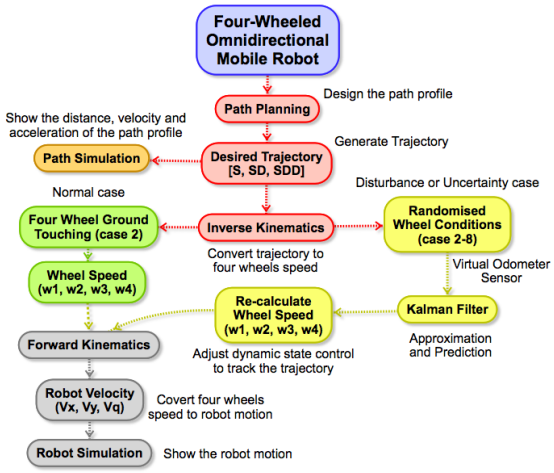

Fig. 2. The simulation procedures.

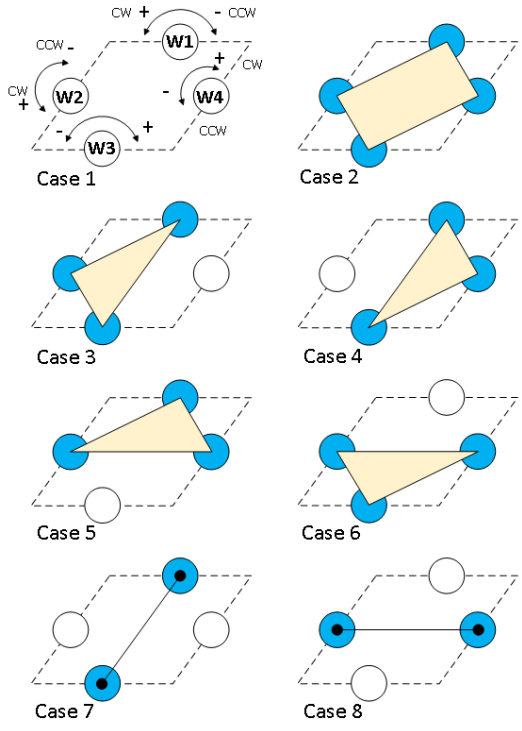

Fig. 3. There are eight possible cases involving the four omnidirectional wheels. This shows the rotation direction of each omnidirectional wheel, clockwise and counterclockwise. In case 1, not all wheels touch the ground. This is not used in the simulation.

shown in Fig. 3. The wheels can change the states of motion, as in Table 2, the four wheels drive in Case 2, the three wheels drive in Cases 3 through 6, and the two wheels drive in Cases 7 and 8.

Table 2. The omnidirectional wheels touch and do not touch the floor in the conditions of motion. Eight cases have been determined.

case	W_1	W_2	W_3	W_4
1	0	0	0	0
2	1	1	1	1
3	1	1	1	0
4	1	0	1	1
5	1	1	0	1
6	0	1	1	1
7	1	0	1	0
8	0	1	0	1

Note: 1 is touching and 0 is not touching the ground.

In the example, W_1 and W_3 are stuck, and the robot cannot keep moving in the desired direction in case 8. The robot needs to get out by struggling to the left and right, back and forth, as shown in Fig. 4.

In motion case 2, all wheels are touching the floor while the robot is moving. This is the ideal condition for motion. However, not all wheels can touch the ground all the time in cases 3 through 8.

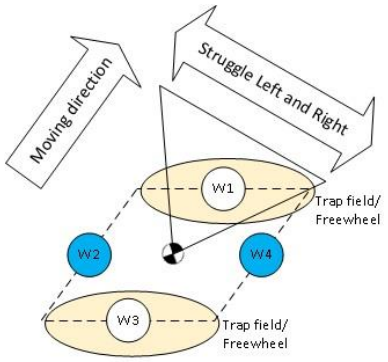

Fig. 4. The omnidirectional mobile robot attempts to move under the conditions of motion case 8. The trap field, or the freewheel, is random in the simulation. This means that the omnidirectional wheel is not touching the floor.

4. Motion Improvement

The planned path trajectory is assigned to the robot's moving path profile. This omnidirectional mobile robot cannot move along to the goal position when it is in motion case condition 3-8. The robot needs to adjust its position to the goal even when the robot goes into a trap field. Motion improvement aims to help the robot approach the goal position using the Kalman filter.

4.1. Kalman Filter

The Kalman filter was invented by Rudolf Kalman. It is a well-known algorithm that is used to estimate the output from a system model and sensor measurements. The Kalman filter was invented for the linear system, but many problems are non-linear. Thus, it needs to linearize the system model. Estimating the measurements of a non-linear system requires a tool known as the extended Kalman filter (EKF). This paper uses the EKF for the four-wheeled omnidirectional mobile robot. The algorithm is iterative to predict, and an update follows Eq. (7-11). However, much research is needed in order to implement the EKF within the omnidirectional mobile robot. Guidelines following the algorithm's steps are described in Ref. 26-27. A difference in this work as compared to other works is the use of random states of motion for the state process.

Predict:

$$\hat{x}_t^- = f(\hat{x}_{t-1}^-, \hat{u}_t^-) + w_t \quad (7)$$

$$P_t^- = F_x P_{t-1} F_x^T + F_v U_t F_v^T + Q_t \quad (8)$$

Update:

$$K_t = P_t^- H_t^T (H_t P_t^- H_t^T + R_t)^{-1} \quad (9)$$

$$\hat{x}_t = \hat{x}_t^- + K_t(z_t - \hat{z}_t) \quad (10)$$

$$P_t = (I - K_t H_t) P_t^- \quad (11)$$

Where $\hat{x}$ is the estimated state. F is the state transition matrix (translation between states). $\hat{u}$ is the control variables. w is the modelled random white noise. P is the state variance matrix (error of estimation). U is the control input error covariance. Q is the process variance matrix (error due to process). z is the measurement variables. H is the measurement matrix (mapping measurements onto state). K is the Kalman gain. R is the measurement variance matrix (error from measurements). Subscripts are as follows: t current time period and t-1 previous time period. Superscript (-) is intermediate steps.

The EKF framework has various sensors for the step update that is referred to as a "sensor fusion.[27]" The values from simulation compared to the original sensors are the heading angle from a compass, the yaw rate from

a gyroscope, the wheel's speed from an encoder, the freewheel check from the wheel's rotation and the robot's movement using an encoder and an accelerometer, and target bearing from a camera.

5. Simulation Results

The planned path was a linear path from the start position (x = 20, y = 20) to the goal position (x = 150, y = 150). It started moving to the +diagonal right, as in Table 1, and then attempted to rotate itself because the heading angle was changed. The robot's heading angle started from 0 rad, as in the configuration in Fig. 1. Finally, the robot's heading angle stopped at π ⁄4 rad. Moving under the conditions of motion randomly, the robot shifted from the planned path and had not reached the goal position when the time ended. The robot's actual position is shown in Fig. 5.

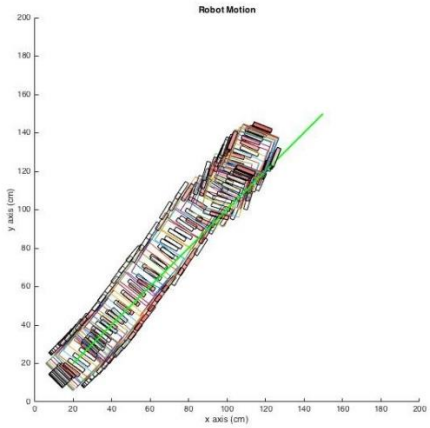

Fig. 5. The robot randomly moved under the eight conditions of motion, following the planned path from the start position to the goal position.

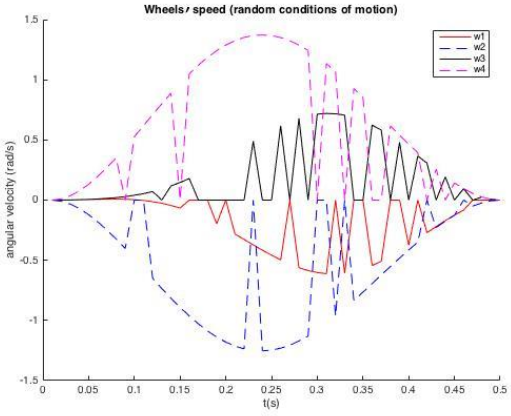

Fig. 6. The four wheels' speed trajectory is related to the planned path. W_1 and W_2 rotated counterclockwise. W_3 and W_4 rotated clockwise.

Accordingly, each wheel of the robot had the wheel's speed, as shown in Fig. 6. The wheel's speed dropped to zero in the trap field in case of free running, or the wheel did not touch the floor.

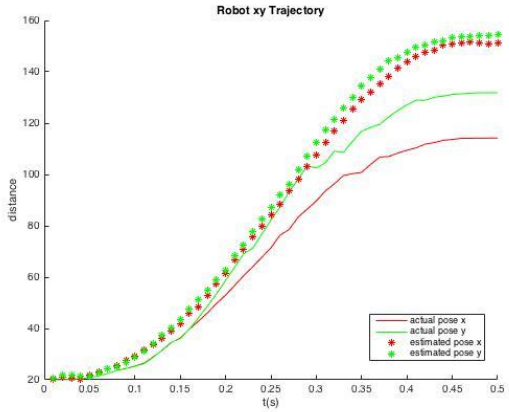

Fig. 7. The robot's actual positions x and y shifted from the planned path. The estimated position x and y from the EKF had a better motion.

The total simulation time was 0.5 seconds, with every 0.01 second per step. After applying the EKF to the robot, the estimated positions x and y were smooth and could reach very close to the goal position as shown in Fig.7.

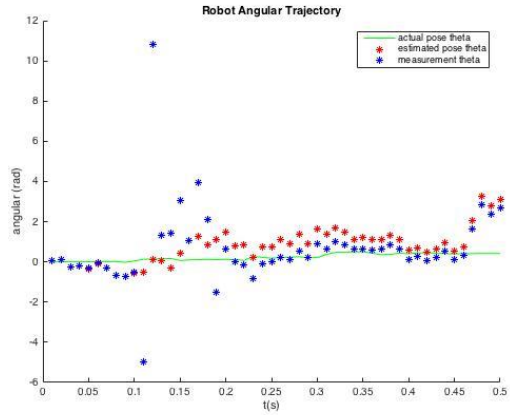

Fig. 8. The robot's heading angle simulated the actual position. The estimated and measured heading angle were the motion improvement from the EKF.

The robot maintained the position on the planned path. However, the robot's heading angle from measurement and estimation changed a lot as shown in Fig. 8.

The robot in x and y position could be compared in

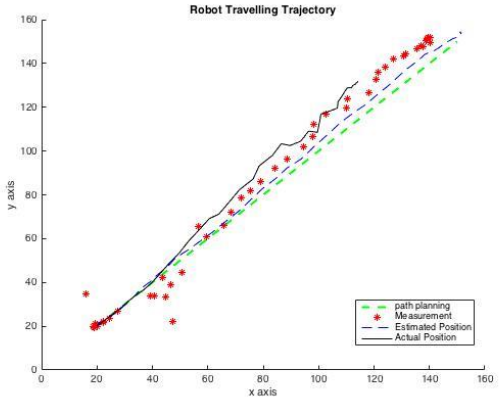

Fig. 9. Comparing the motions of the robot from path planning, measurement, estimation, and actual position.

Fig. 9. The measured position was noisy and affected by the error of the actual position.

For the motion improvement, each wheel's speed was generated from the estimated positions. The estimated wheel's speed was very noisy compared to the actual

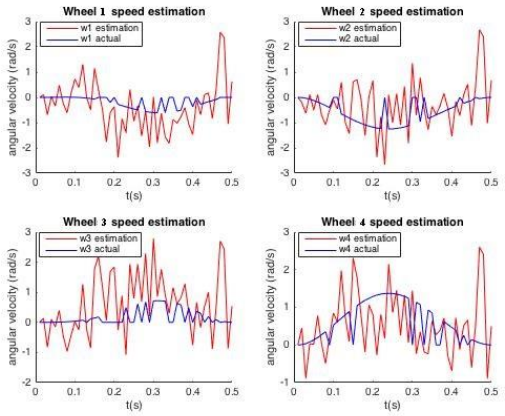

Fig. 10. Comparing the estimated and the actual speed of the four wheels.

wheel's speed because the robot struggled to the new positions as in Fig. 10.

6. Conclusion

The robot randomly changed its states of motion in four-wheel, three-wheel, and two-wheel drive. The main error came from the robot motion model, which changed the number of the driving wheel. Generally, the EKF reduces the effects of slip noise and sensor noise in the dead reckoning problem. However, the simulation result shows that the EKF produces better robot motion along the planned path. The robot struggles to traverse its planned path with its assigned trajectory.

The behavior of the robot during the real test may considerably differ from the simulation. Noises from signals are reduced, but the dynamic motion of the robot may not work like the control signals. For real tests in future work, the omnidirectional mobile robot has to have speed control sensors to monitor its wheel speed, and it will be important to check the freewheel and the floor touchless condition when some of the wheels rotate, but the robot does not move, or it moves in the wrong direction.

Acknowledgements

This work was funded by Science and Technology Research Institute, King Mongkut's University of Technology North Bangkok.

References

1. Zou, J. T., The development of the omnidirectional mobile home care robot, In *Mobile Robots-Current Trends* (2011). InTech, pp.345-362.
2. Rojas, R., & Förster, A. G., Holonomic control of a robot with an omnidirectional drive, *KI-Künstliche Intelligenz, 20(2)* (2006), pp.12-17.
3. Kundu, A. S., Mazumder, O., Lenka, P. K., & Bhaumik, S., Design and Performance Evaluation of 4 Wheeled Omni Wheelchair with Reduced Slip and Vibration, *Procedia Computer Science*, 105(2017), pp.289-295.
4. Urbano, J., Yang, Y., Terashima, K., Miyoshi, T., & Kitagawa, H., Navigation with comfort of omni-directional wheelchair driven by joystick, *IFAC Proceedings Volumes*, 38(1) (2005), pp.379-384.
5. Andreyev, A. S., & Peregudova, O. A., The motion control of a wheeled mobile robot. *Journal of Applied Mathematics and Mechanics*, 79(4) (2015), pp.316-324.
6. Rotondo, D., Nejjari, F., & Puig, V., Model reference switching quasi-LPV control of a four wheeled omnidirectional robot, *IFAC Proceedings Volumes*, 47(3) (2014), pp.4062-4067.

7. Galicki, M., Collision-free control of an omni-directional vehicle, *Robotics and Autonomous Systems*, 57(9) (2009), pp.889-900.
8. Flippo, D. F., & Miller, D. P., Turning efficiency prediction for skid steering via single wheel testing, *Journal of Terramechanics*, 52(2014), pp.23-29.
9. Li, B., Du, H., Li, W., & Zhang, Y., Side-slip angle estimation based lateral dynamics control for omni-directional vehicles with optimal steering angle and traction/brake torque distribution, *Mechatronics*, 30(2015), pp.348-362.
10. Huang, H. C., & Tsai, C. C., Adaptive trajectory tracking and stabilization for omnidirectional mobile robot with dynamic effect and uncertainties, *IFAC Proceedings Volumes*, 41(2) (2008), pp.5383-5388.
11. Cooney, J. A., Xu, W. L., & Bright, G., Visual dead-reckoning for motion control of a Mecanum-wheeled mobile robot, *Mechatronics*, 14(6) (2004), pp.623-637.
12. Huang, S. J., & Shiao, Y. W., 2D path control of four omni wheels mobile platform with compass and gyroscope sensors, *Sensors and Actuators A: Physical*, 234(2015), pp.302-310.
13. Oftadeh, R., Aref, M. M., Ghabcheloo, R., & Mattila, J., Mechatronic design of a four wheel steering mobile robot with fault-tolerant odometry feedback, *IFAC Proceedings Volumes*, 46(5) (2013), pp.663-669.
14. Sanada, A., Ishii, K., & Yagi, T., Self-localization of an omnidirectional mobile robot based on an optical flow sensor, *Journal of Bionic Engineering*, 7(2010), pp.S172-S176.
15. Song, X., Seneviratne, L., & Althoefer, K., A Vision Based Wheel Slip Estimation Technique for Mining Vehicles, *IFAC Proceedings Volumes*, 42(23) (2009), pp.179-184.
16. Feng, H. M., Chen, C. Y., & Horng, J. H., Intelligent omni-directional vision-based mobile robot fuzzy systems design and implementation, *Expert systems with applications*, 37(5) (2010), pp.4009-4019.
17. Kundu, A. S., Mazumder, O., Dhar, A., Lenka, P. K., & Bhaumik, S., Scanning Camera and Augmented Reality Based Localization of Omnidirectional Robot for Indoor Application, *Procedia Computer Science*, 105(2017), pp.27-33.
18. Shaikh, M. M., Hwang, W., Park, J., Bahn, W., Lee, C., Kim, T., & Kim, K. S., Mobile Robot Vision Tracking System Using Dead Reckoning & Active Beacons, *IFAC Proceedings Volumes*, 44(1) (2011), pp.9379-9384.
19. Hashemi, E., Jadidi, M. G., & Jadidi, N. G., Model-based PI–fuzzy control of four-wheeled omni-directional mobile robots, *Robotics and Autonomous Systems*, 59(11) (2011), pp.930-942.
20. Kalmár-Nagy, T., Real-time trajectory generation for omni-directional vehicles by constrained dynamic inversion, *Mechatronics*, 35 (2016), pp.44-53.
21. Hoang, N. B., & Kang, H. J., Neural network-based adaptive tracking control of mobile robots in the presence of wheel slip and external disturbance force, *Neurocomputing*, 188 (2016), pp.12-22.
22. Udengaard, M., & Iagnemma, K., Analysis, design, and control of an omnidirectional mobile robot in rough terrain, *Journal of Mechanical Design*, *131*(12) (2009), 121002.
23. Song, J. B., & Byun, K. S., Design and control of an omnidirectional mobile robot with steerable omnidirectional wheels, In *Mobile Robotics, Moving Intelligence* (2006), InTech.
24. Dixon, R. C., Bright, G., & Harley, R., Robust localisation of automated guided vehicles for computer-integrated manufacturing environments, *South African Journal of Industrial Engineering*, *24*(1) (2013), pp.81-90.
25. C. Peter, Robotics Vision and Control, (*Springer Tracts in Advanced Robotics*, 2011), pp.107–128.
26. Suliman, C., Cruceru, C., & Moldoveanu, F., Mobile robot position estimation using the Kalman filter, *Scientific Bulletin of the" Petru Maior" University of Targu Mures*, *6* (2009), pp.75.
27. Surrécio, A., Nunes, U., & Araújo, R., Fusion of odometry with magnetic sensors using kalman filters and augmented system models for mobile robot navigation, In *Proceedings of the IEEE International Conference on Industrial Electronics* (2005), pp.1551-1556.