Joint Space Trajectory Planning Considering Physical Limits with Convolution Operator for Mobile Robots

Gil Jin Yang and Byoung Wook Choi*
Department of Electrical and Information Engineering
Seoul National University of Science and Technology, Seoul, South Korea
*yang6495@gmail.com, *bwchoi@seoultech.ac.kr*

Abstract

A practical and useful trajectory planning that considers the physical limits of mobile robots using the convolution operator in joint space are presented. Smooth joint velocity commands that consider the maximum velocity of a mobile robot and systems limit along the path are computed. The effectiveness of the algorithm is shown through various Bezier curves in a robot simulator. Three curved path were conducted to determine whether the mobile robot will be able to meet the physical limits. Each experiment had a different start point and end point. Results show that the mobile robot was able to meet the velocity limit and follow the predetermined path while satisfying physical constraints in the joint space.

Keywords: *Bezier curve, Mobile robots, Physical Limits, Joint Space Trajectory*

1. Introduction

A mobile robot is a nonholonomic system and recently becoming widely used as cleaning robots and intelligent service robots. Various trajectory planning approaches have been addressed by researchers [1-4].

Considering the physical limits of a mobile robot is crucial. During path planning and trajectory generation, the physical limits should be considered in order to avoid potential damage to a mobile robot. If the velocity profiles of a mobile robot exceeded its physical limits, then the mobile robot might not be able to follow the velocity profiles. As a consequence, the mobile robot will end up going to a wrong end point.

The continuous path designed in task space translates to the requirement for second order differentiable trajectories in joint space. This means continuous velocity commands for each of two wheels. A convolution-based trajectory generation method satisfying system specifications was suggested in [4-7].

This paper utilizes a Bezier curve-based two-differentiable path planning that satisfies a robot's heading angles and follows the designed path while considering velocity limits of translational velocity and rotational velocity [8]. A convolution operator is used to generate the central velocity to travel the planned path [5]. In this process, the velocity trajectory can be generated while considering the maximum velocity and acceleration according to the physical limits of mobile robots. The velocity trajectory gained through convolution is a trajectory along which a robot travels given distance that does not consider the rotating angle of mobile robots. In order to consider a rotating angle of mobile robots, a transformational method for the trajectory is presented that consists of segmented paths along designed Bezier curve with the central velocity generated through convolution [4].

The trajectory obtained through transformational process can be used for mobile robots to smoothly follow the planned path while staying within the physical limits. Finally, an algorithm providing smooth velocity profiles in joint space which are actuator commands to the mobile robot while considering the velocity limits is suggested. In order to determine the effectiveness of the proposed method, numerical simulations were performed. The application of the planned trajectory to a simulator showed that the robot carried out desired tasks while staying within its physical limits. This trajectory can be used for path planning to optimize time and energy

* Corresponding author

consumption. Three paths, including a C-curve, an S-curve and an arbitrary curve were applied to the mobile robot to determine whether it will be able to meet the physical limits according to various predetermined paths. The path error in terms of sampling time when making trajectory that follows the predefined Bezier curved path was also examined.

2. Smooth trajectory planning in joint space considering velocity limits

The trajectory denotes the path that a mobile robot should traverse as a function of time. For a mobile robot, a trajectory in task space is its position ($x(t)$, $y(t)$) in the Cartesian coordinate, and orientation $\theta(t)$, so that its configuration for the center of the mobile robot is $q_c(t) = [x_c(t), y_c(t), \theta_c(t)]^T$ [4].

When planning a path for a mobile robot, the position and direction angle at its start point and end point should be considered. A curved trajectory is commonly generated using Bezier curves [8]. To create a smooth path that considers the configurations at the robot's start point and end point, a trajectory is commonly generated using a third degree Bezier curve consisting of a start point $P_i(A_0, B_0)$, an end point $P_f(A_3, B_3)$, and control points $C_1(A_1, B_1)$ and $C_2(A_2, B_2)$. The trajectory takes a smooth Bezier curve-based path $\rho(u) = [x(u), y(u)]^T$ that considers the angle of rotation and uses a constant parameter u as input and tries to find the velocity profiles in joint space.

There has been research that the path generation method may use a convolution operator to create a central velocity trajectory of a mobile for smooth path generation while satisfying physical limits [5]. If the function $y_n(t)$ is a resulting function to which the nth convolution is applied, the result of convolution $y_0(t)$ and $h_1(t)$ can be represented as $y_1(t)$ and $y_2(t)$ denotes the result of $y_1(t)$ and $h_2(t)$ after convolution, where the nth-applying convolution function $h_n(t)$ is defined as a square-wave function with the unit area. The velocity function for the center of the mobile robot $v_c(t)$ generates the velocity command of the differentiable S-curve that considers the maximum velocity v_{max} for the robot to travel the distance S.

The central velocity is obtained using the convolution operator. The central velocity, which considers the velocity limit, travels along the distance of the path. The path $\rho(u(t))$, which is obtained by computing the central velocity, considers the heading angles of a mobile robot. Joint velocities could not satisfy the physical limits when the angular velocities are high. The distance traveled is calculated using a Bezier curve to generate the central velocity trajectory for the robot to travel along the distance S. The curved distance B_d along the path $\rho(u)$ from the initial to final position is calculated as follows:

$$B_d = \sum_{u=0}^{1} \Delta\rho(u) = \sum_{u=0}^{1} \sqrt{(x(u+\Delta u)-x(u))^2 + (y(u+\Delta u)-y(u))^2} \quad (1)$$

In order to generate the center velocity trajectory of a mobile robot using convolution, the distance S is thus used as an input value. Therefore, if the center velocity trajectory $v_c(t)$ is generated to have the traveling distance as $S = B_d$, then the trajectory using the advantages of convolution while considering velocity limits can make a smooth path.

The generated central velocity trajectory of $v_c(t)$ travels along the distance S. However, the central velocity trajectory of the mobile robot does not consider the direction of the robot. In other words, for any position ($x(u_i)$, $y(u_i)$), the robot travels with velocity $v_c(t_i)$. In order to consider the positions in task space that depend on velocities in paths with direction angles, the parameter $u(t)$ of Bezier curve for the distance during the sampling time should be determined and calculated using Equation (2). The trajectory $\rho(u(t))$ with the direction angle can be obtained by inputting the determined $u(t)$ into the Bezier curve equation. In $\rho(u(t))$, if the sampling time is shorter, the path can more accurately follow $\rho(u)$ as generated by constant parameter value u.

$$u(t) = \sum_{t=0}^{t_0+t_1+t_2} v_c(t) \Big/ B_d \quad (2)$$

Here, $u(t)$ is defined as $0 \le u(t) \le 1$ and represents the parameter of the Bezier curve that depends on the central velocity. The trajectory generated by using $u(t)$ satisfies the maximum velocity allowed by the physical limits of a mobile while following the curved path with respect to the direction angles.

The actual command for actuating the mobile robot is the angular velocities, where ω_c is the rotating velocity for the center, ω_r is the right wheel's rotating velocity, and ω_l is the left wheel's rotating velocity. The command can generate wheel velocity commands in joint space using equations (3) and (4):

$$\omega_r = \frac{1}{r}(v_c + D/2 \cdot \omega_c)$$
$$\omega_l = \frac{1}{r}(v_c - D/2 \cdot \omega_c) \quad , \quad (3)$$

$$v_r = r\omega_r$$
$$v_l = r\omega_l \quad (4)$$

where D denotes the distance between its two wheels and r denotes the radius of a robot's wheel.

In this study, joint velocity commands were calculated that do not satisfy physical limits even though the center velocity satisfies physical limits. The joint velocity commands can be maintained within the maximum velocity limit by correcting the center velocity as shown in Equation (5) when the central velocity trajectory is generated. By doing so, a trajectory can be generated that satisfies an actual actuator's physical limits.

$$v_{comp} = \frac{|max(v_r - v_l)|}{2},$$
$$v'_{max} = v_{max} - v_{comp} \quad (5)$$

By doing convolution again with the modified velocity limit v'_{max}, the joint velocity profiles that satisfy the robot's actuator's physical limits can be generated. If any of the velocity profiles (central velocity, left wheel velocity and right wheel velocity) exceeds the robot's physical limits, then the robot's actuator will not be able to follow the velocity profile.

3. Experiments

Three experiments were conducted. In each experiment, the start point and the end points were different. The values of the physical limits during the experiments were v_{max} = 0.5 m/s, a_{max} = 0.2m/s^2, and j_{max} = 0.2m/s^3. The sampling time in all the experiments was 100 ms.

3.1 C-curved path

The first experiment's path is a C-curved path. The velocity trajectory satisfied the physical limits while moving from (0, 0, 0°) to (4, 4, 90°) as shown in Figure 1. Joint trajectories with considering the center velocity limit and considering the actuator velocity limit in the joint space are shown in Figure 2 and 3, respectively. The simulation results applied to the robot simulator are shown in Figure 4 and 5 [9].

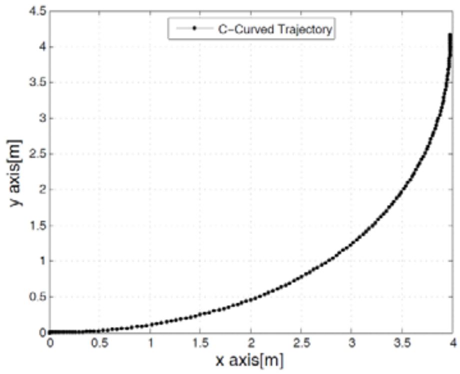

Figure 1. C-curved trajectory in the Cartesian space

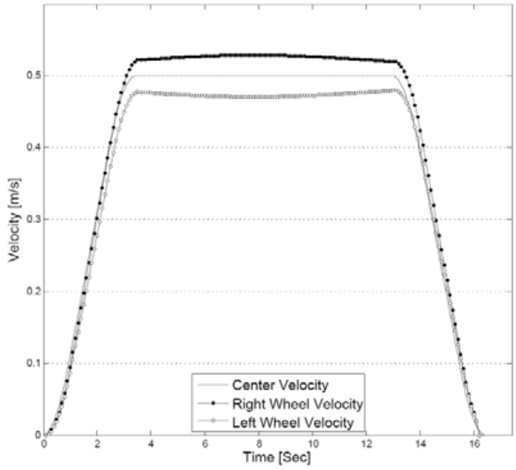

Figure 2. Joint velocity trajectories considering the central velocity limit

Figure 2 shows that when the physical limits are v_{max} = 0.5 m/s, a_{max} = 0.2 m/s^2 and j_{max} = 0.2 m/s^3, the central velocity trajectory satisfies the physical limits while moving from the start point (0, 0, 0°) to the end point (4, 4, 90°). The joint velocity commands for the two wheels are used to drive the two wheels to follow the Bezier curve-based trajectory.

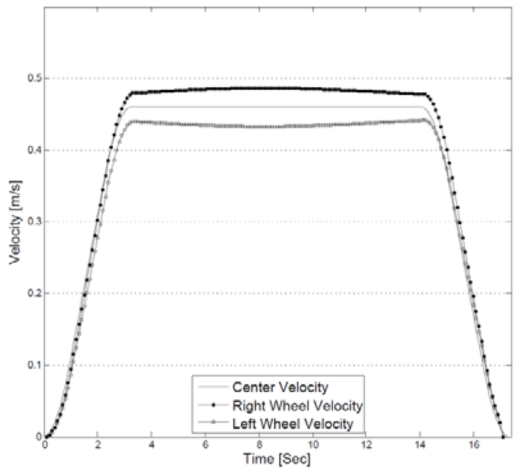

Figure 3. Joint velocity trajectories considering the actuator's velocity limit

The velocity commands for the two wheels satisfying the actuator's physical limits are shown in Figure 3. Figure 3 travels along the C-curved path shown in Figure 1. The distance traveled along the generated joint velocity commands are constant due to the convolution operator's characteristics. The travel time slightly increased because of the limited velocity.

The proposed algorithm was applied to the robot simulator, Marilou Robotics Studio in anyKode [9]. Figure 4 and 5 shows the simulation results in task space. They also show the trace of the velocity commands for the two wheels, wherein the mobile robot dimensions are r = 10 cm and D = 40 cm. Figure 4 and 5 have the same desired end points but in Figure 4, the mobile robot missed the end point because the mobile robot used Figure 2's velocity profiles, which exceeded the physical limits of the mobile robot during Figure 4's simulation. Equation 5 was applied to the joint velocity trajectories to satisfy the robot's actuator's physical limits. The result of the application of Equation 5 to the simulation done in Figure 4 is seen in Figure 3 and 5. Figure 3 shows the joint velocity trajectories while Figure 5 shows the trace of the robot's driven path.

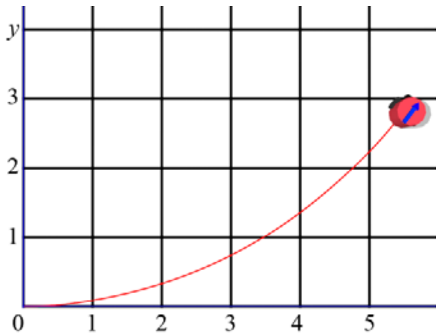

Figure 4. C-curved path result driven by joint velocity commands of Figure 2

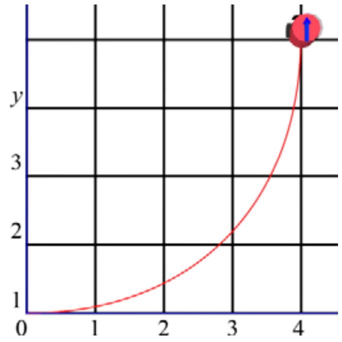

Figure 5. C-curved path result driven by joint velocity commands of Figure 3

Figure 5 shows that the mobile robot successfully followed the planned path. The travel time slightly increased after considering the robot's actuator's velocity limits. The proposed trajectory generation method can be used to generate velocity commands for robot's driving and controlling.

If any of the velocity profiles (central velocity, left wheel velocity and right wheel velocity) exceeds the robot's physical limits, then the robot's actuator will not be able to follow the velocity profile. As shown in Figure 2, the right wheel's velocity exceeded the robot's velocity limit. Therefore the mobile robot cannot follow the velocity profile. This is the reason why the mobile robot missed its supposed end point, as seen in Figure 4.

Figure 5 shows the results after getting the modified velocity profile v'_{max}. The mobile robot was able to follow the velocity profile and eventually was able to arrive at the end point.

3.2 S-curved path

The second experiment's path is an S-curved path from (0, 0, 0°) to (4, 4, 0°) as shown in Figure 6. Joint trajectories with considering the center velocity limit and considering the actuator velocity limit in the joint space are shown in Figure 7 and 8, respectively. The simulation results are shown in Figure 9 and 10.

Figure 6 shows that when the physical limits are v_{max} = 0.5 m/s, a_{max} = 0.2 m/s^2 and j_{max} = 0.2 m/s^3, the central velocity trajectory satisfies the physical limits moving from the start point (0, 0, 0°) to the end point (4, 4, 0). The joint velocity commands for the two wheels are used to drive the two wheels to follow the Bezier curve-based trajectory.

The velocity commands for the two wheels satisfying the actuator's physical limits are shown in Figure 8. Figure 8 shows that the mobile robot travels along the S-curved path shown in Figure 6. The distance traveled along the generated joint velocity commands is constant because of the convolution operator's characteristics. The travel time also increased slightly because of the limited velocity.

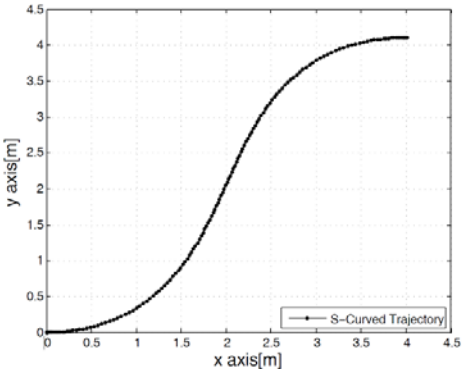

Figure 6. S-curved trajectory in the Cartesian space

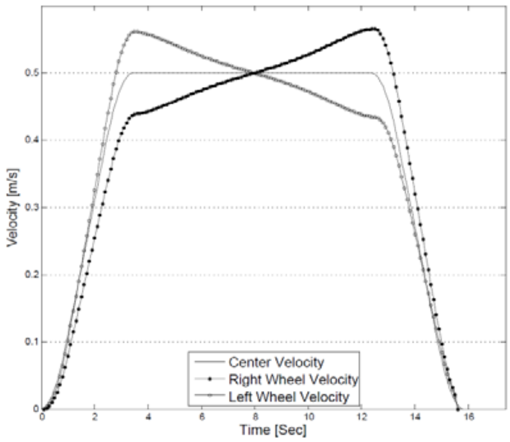

Figure 7. Joint velocity trajectories considering the center velocity limit

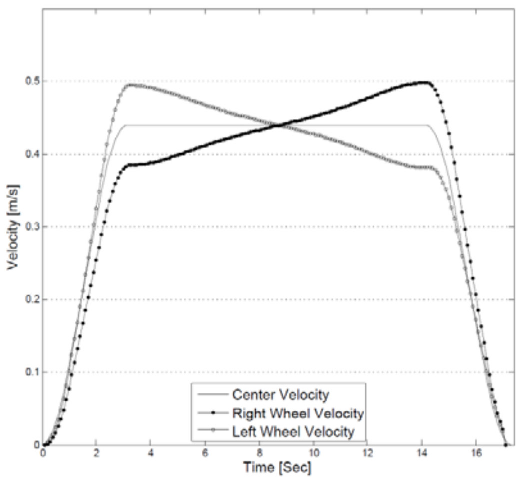

Figure 8. Joint velocity trajectories considering the actuator's velocity limit

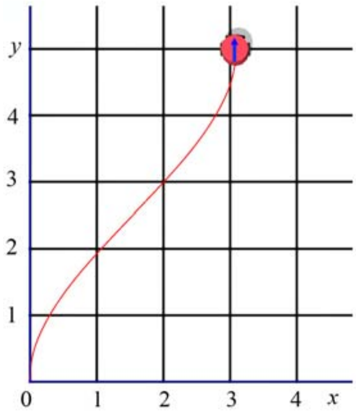

Figure 9. S-curved path result driven by joint velocity commands of Figure 7

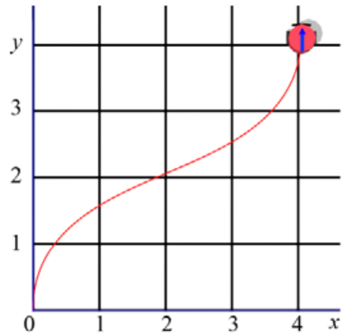

Figure 10. S-curved path result driven by joint velocity commands of Figure 8

Figure 9 and 10 shows the simulation results in task space. They also show the trace of the velocity commands for two wheels. Figure 9 and 10 have the same end points but in Figure 9, the mobile robot missed the end point because the mobile robot used Figure 7's velocity profiles. The velocity profiles exceeded the physical limits of the mobile robot during Figure 9's simulation. Equation 5 was applied to satisfy the robot's actuator's physical limits. The result of the application of Equation 5 to the simulation done in Figure 9 is shown in Figure 8 and 10. Figure 8 shows the joint velocity trajectories while Figure 10 shows the trace of the robot's driven path.

3.2 Arbitrary path

The final experiment used two paths moving from (0, 0, 45°) to (3, 1, 45°) through (1, 1, 0°) as shown in Figure 10. Joint trajectories with considering the center velocity limit and considering the actuator velocity limit in the joint space are shown in Figure 11 and 12, respectively. The simulation results are shown in Figure 13 and 14.

Figure 11 shows that when the physical limits are v_{max} = 0.5 m/s, a_{max} = 0.2 m/s^2 and j_{max} = 0.2 m/s^3, the central velocity trajectory satisfies the physical limits moving from the start point (0, 0, 0°) to the end point (4, 4, 0). The joint velocity commands for the two wheels are used to drive the two wheels to follow the Bezier curve-based trajectory.

The velocity commands for the two wheels satisfying the actuator's physical limits are shown in Figure 13. The mobile robot travels along the S-curved path shown in Figure 11. The distance traveled along the generated joint velocity commands are constant due to the convolution operator's characteristics. The travel time slightly increased because of the limited velocity.

Figure 14 and 15 shows the simulation results in task space. They also show the trace of the velocity commands for two wheels. Figure 14 and 15 have the same end points. But in Figure 14, the mobile robot missed the end point. This is because the mobile robot used Figure 12's velocity profiles.

The velocity profiles during the experiment exceeded the physical limits of the mobile robot during Figure 14's simulation. Equation 5 was applied to satisfy the robot's actuator's physical limits. The result of the application of Equation 5 to the simulation done in Figure 14 is seen in Figure 13 and 15. Figure 13 shows the joint velocity trajectories while Figure 15 shows the trace of the robot's driven path.

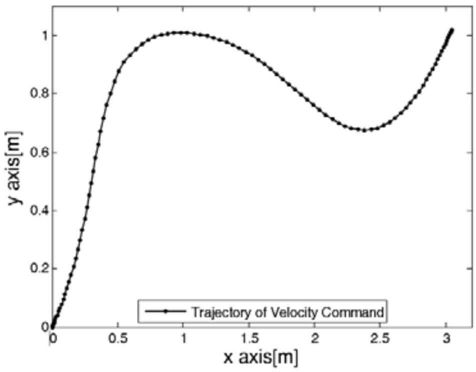

Figure 11. Arbitrary trajectory using a Bezier curve

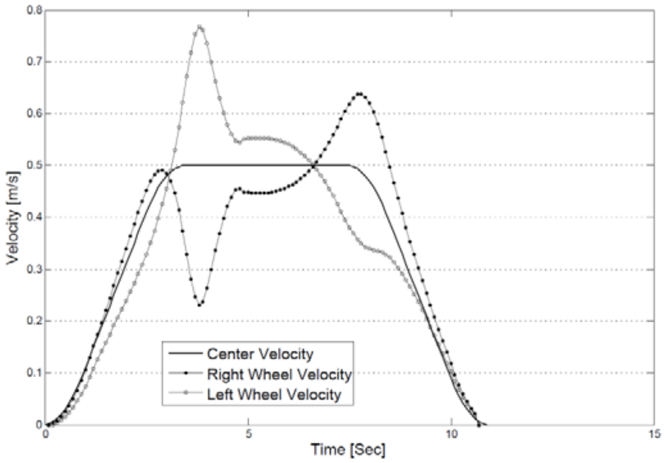

Figure 12. Joint velocity trajectories considering the center velocity limit

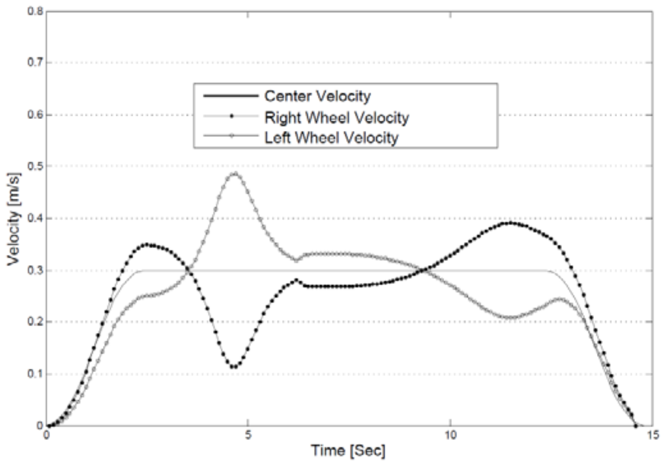

Figure 133. Joint velocity trajectories considering the actuator’s velocity limit

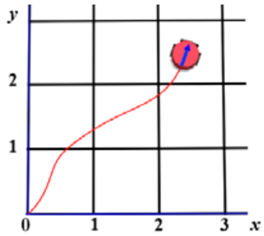

Figure 14. Arbitrary path result driven by joint velocity commands of Figure 12

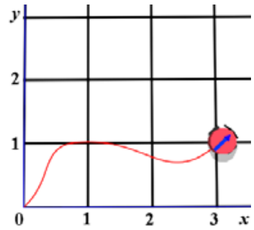

Figure 15. Arbitrary path result driven by joint velocity commands of Figure 13

4. Path error according to sampling time

When generating the velocity profiles for a mobile robot to travel smoothly along a Bezier curve-based path, the sampling time should also be considered to produce the velocity commands. Figure 15 shows trial 3's the generated trajectory according to sampling times. The results show that the error increases as sampling time increases. However, if a system cannot drive commands within sampling time, the resulting trajectory also cannot follow the predefined path.

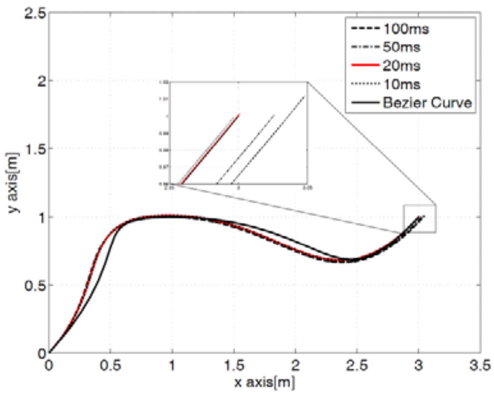

Figure 16. Trajectory according to sampling time

5. Conclusion

A practical method for generating velocity profiles for a mobile robot to travel smoothly along a curved path within the actuator's physical limits was proposed. The proposed velocity profiles satisfy the physical limits. The central velocity of the mobile robot uses the characteristics of the convolution operator [5]. The central velocity trajectory follows the smooth Bezier curve-based path.

According to the result of the experiment, the mobile robot will be able to follow the Bezier curve-based path if the proposed method is applied. The travel time slightly increased because of the limited velocity. Before application of the proposed method, the mobile robot could not follow the path. The mobile robot ended up at wrong end point because the velocity profiles exceeded the physical limits of the robot's actuators.

The proposed trajectory generation method can be used to generate velocity commands for actual driving and controlling. In the future, this trajectory generation method can be applied to obstacle avoidance algorithms that satisfy the velocity limits at the any point [10-11].

The results show that the error increases as sampling time increases. If a system cannot drive commands within the sampling time, the resulting trajectory also cannot follow the predefined path. This considered the effect of sampling time to cope with control loop.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2012-006057).

6. References

[1] J. J. Craig, Introduction to Robotics, Prentice-Hall, 2005.
[2] G. Lee, D. I. Kim and Y. J. Choi, “Faster and Smoother Trajectory Generation considering Physical System Limits under Discontinuously Assigned Target Angles”, IEEE International Conference on Mechatronics and Automation, pp. 1196-1201, 2012.
[3] M.S. Jang, E.H. Lee and S.B. Choi, “A Study on Human Robot Interaction Technology Using a Circular Coordinate System for the Remote Control of the Mobile Robot”, International Journal of Control and Automation, vol. 5, pp. 117-130, 2012.
[4] G. J. Yang and B. W. Choi, “Smooth Trajectory Planning Along Bezier Curve for Mobile Robots with Velocity Constraints”, International Journal of Control and Automation, vol. 6, no. 2, pp. 225-234, 2013.
[5] Lee. G, J. Kim and Y. Choi, “Convolution-Based Trajectory Generation Methods Using Physical System Limits”, J. Dynamic Systems, Measurement, and Control, ASME, vol. 135, pp. 011001-1-011001-8, 2013.
[6] Misel Brezak and Ivan Petrovic, “Time-Optimal Trajectory Planning Along Predefined Path for Mobile Robots with Velocity and Acceleration Constraints,” IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics, Budapest, Hungary, pp. 942-947, 2011.
[7] M. Lepetič, G. Klančar, I. Škrjanc, D. Matko and B. Potočnik, “Time optimal path planning considering acceleration limits”, Robotics and Automation Systems, vol. 45, pp. 199-210, 2003.
[8] K. G. Jolly, R. S. Kumar and R. Vijayakumar, “A Bezier Curve Based Path Planning in a Multi-Agent Robot Soccer System without Violating the Acceleration Limits”, Robotics and Automation Systems, pp. 22-33, 2009.
[9] anyKode, Marilou Robotics Studio, www.anykode.com
[10] J.S. Kim and B.K. Kim, “Efficient Minimum-Time Cornering Motion Planning for Differential-Driven Wheeled Mobile Robots with Motor Control Input Constraint,” Journal of Institute of Control, Robotics and Systems, vol. 19, pp. 56-64, 2013.
[11] J.S. Kim and B.K. Kim, “Minimum-Time Grid Coverage Trajectory Planning Algorithm for Mobile Robots with Battery Voltage Constraints,” International Conference on Control, Automation and System, pp. 27-30, Oct. 2010.