Pure Pursuit Revisited: Field Testing of Autonomous Vehicles in Urban Areas

Conference Paper · October 2016

DOI: 10.1109/CPSNA.2016.10

Naoki Akai
Nagoya University

Pure Pursuit Revisited: Field Testing of Autonomous Vehicles in Urban Areas

Hiroki Ohta*, Naoki Akai†, Eijiro Takeuchi*, Shinpei Kato‡ and Masato Edahiro*
* Graduate School of Information Science
Nagoya University
† Institutes of Innovation for Future Society
Nagoya University
‡ Graduate School of Information Science and Technology
University of Tokyo

***Abstract*—**
In this paper, we aim to explore path following. We implement a path following component by referring to the existing Pure Pursuit algorithm. Using the simulation and field operational test, we identified the problem in the path following component. The main problems identified were with respect to vehicles meandering off the path, turning a corner, and the instability of steering control. Therefore, we apply some modifications to the Pure Pursuit[1] algorithm. We have also conducted the simulation and field operational tests again to evaluate these modifications.

I. Introduction

In the near future, autonomous vehicles will bring substantial changes to traffic infrastructure and our daily lives. Automotive companies, electric manufacturers, and information technology (IT) service providers are interested in such technology. Academic organizations have contributed to the development of prototype systems[2],[3]. In 2007, the DARPA Urban Challenge[4] showed that their vehicle was able to drive autonomously in urban environments[5],[6],[7],[8],[9].

We can roughly classify the technology needed for autonomous driving into environmental perception, path planning and vehicle control. A vehicle perceives the surrounding environment using sensors, plans a path to a given destination, and calculates operation commands to follow the determined path. High performance three dimensional Light Detection and Ranging (LiDAR) sensors and Global Navigation Satellite Systems (GNSS) enable us to estimate position with high accuracy in various locations, such as public roads and urban areas. Thus, autonomous vehicles can follow a path to a predetermined destination.

In this paper, we explore path following. We implement a path following function in reference to an existing approach, identify problems through simulation and field operational tests, and present ways to resolve the problems. There are various control methods for path following[10]. Pure Pursuit is a well-known algorithm that follows a given path. This algorithm calculates the angular velocity to reach a target point using the current position and linear velocity of a robot and the curvature passing the target point. This algorithm has been used for many years at a robotics institute, for example Massachusetts Institute of Technology (MIT). A paper published by MIT[7][11] about the DARPA Urban Challenge discussed Pure Pursuit algorithm. Therefore, we decided to revisit the Pure Pursuit algorithm to determine if it was suitable to investigate path following for a vehicle.

We implemented the Pure Pursuit algorithm in an autonomous driving platform called Autoware[12], which has module oriented frame work, and applied flexibly to various vehicles.

However, simulations and field operational tests revealed the following problems.

- The vehicle sometimes overshoots the path and oscillates along the path (meandering drive).
- The discrete operational command leads unstable of the steering control.
- The vehicle sometimes passes through the inside of sharp curves, such as a left turn at an intersection.

These Problems are caused by the delay of environmental perception or processing. To solve the above problems, we examine the Pure Pursuit algorithm and consider solutions obtained by improving parameter tuning and implementation. We then perform simulations and field operational tests to determine if the improvements resolve the identified problems.

II. Related Work

A. Autonomous Driving in Urban Environments: Boss and the Urban Challenge[5]

Boss is an autonomous vehicle that won first place in the DARPA Urban Challenge. Boss has a three-layer planning system. The planning system combines mission, behavioral and motion planning to facilitate driving in urban environments. The mission planning layer determines the best route to the mission goal. The behavioral layer determines when to change lanes, how to handle intersections and performs error recovery. The motion planning layer creates a path to the desired goal and tracks the vehicle's trajectory based on the given path. However, the algorithm used to follow the path generated by the motion planner is not described.

B. Junior: The Stanford Entry in the Urban Challenge[6]

This article describes Junior, a robotic vehicle capable of driving in urban environments autonomously. Junior won second place in the DARPA Urban Challenge. This vehicle has navigation modules to determine its behavior. The navigation modules consist of a number of motion planners and a finite state machine. In the motion planner, the path to the mission goal is generated; however this article does not explain how path following is determined.

C. Team MIT Urban Challenge Technical Report[7]

This technical report describes the MIT autonomous driving vehicle that participated in the DARPA Urban Challenge. This vehicle has a number of planning systems that demonstrate basic navigation and traffic behaviors. For example, pure-pursuit control is used for lane-following.

III. Assumptions and Scope

In this paper, we assume the following in simulations and field tests with an autonomous driving platform.

- Our autonomous driving employs path following rather than adaptive cruise control or lane-keeping based on white line recognition.
- Paths are defined as a series of discrete points (waypoints). The distance between waypoints is set to 1 [m]. A waypoint represents $(x, y, heading, velocity)$, a position and a direction on a two-dimensional space and the vehicle's target speed as information. The velocity along the path is planned in consideration of a traffic signal or curve.
- We used automotive four-wheel robots in our experiments.
- Localization is performed using scan matching by 3D LIDAR and a three-dimensional map. The location is assumed to be highly accurate. The location is defined as $(x, y, z, roll, pitch, yaw)$, a position and a orientation in three-dimensional space.

IV. Implementation

A vehicle drives automatically by estimating its position with high accuracy, planning the path to the goal, and following the given path.

Pure Pursuit is a well-known algorithm for following a given path. We have designed a path-following algorithm based on a previously proposed Pure Pursuit algorithm[1].

1) Obtain the vehicle's position (two dimensional position and orientation).
2) Search the next target waypoint. The target waypoint is the closest waypoint outside of a given search circle. The origin of the search circle is the center point of the rear axle of the vehicle. The radius of the circle is set as a parameter (i.e., Lookahead Distance).
3) Calculate the curvature of the circle passing the vehicle position and the target waypoint.

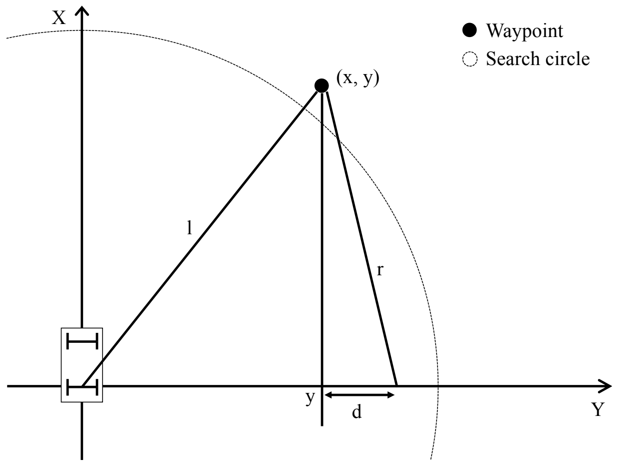


Fig. 1: Illustration about the Pure Pursuit Algorithm[1]

4) Calculate the angular velocity from the current linear velocity and the curvature assuming that the vehicle moves in a uniform circular motion.
5) Update the vehicle position

Figure 1 illustrates the implemented Pure Pursuit algorithm. Calculation of the curvature κ needed for the Pure Pursuit is shown below.

$$y + d = r \tag{1}$$
$$x^2 + y^2 = l^2 \tag{2}$$
$$x^2 + d^2 = r^2 \tag{3}$$

From above equations,

$$\begin{aligned} x^2 + (r - y)^2 &= r^2 \\ x^2 + r^2 - 2ry + y^2 &= r^2 \\ l^2 &= 2ry \\ r &= \frac{l^2}{2y} \\ \kappa &= \frac{2y}{l^2} \end{aligned}$$

We identified the problem by testing this algorithm in the simulation and real field. To address problems described in Section I, we reviewed the implemented algorithm and considered parameter tuning and improving the implementation process as ways to resolve the problems.

The following subsections explain our parameter tuning and improved implementation.

A. Lookahead Distance

The Lookahead Distance parameter is used when searching for the next waypoint. For example, with greater Lookahead Distance, Pure Pursuit algorithm defines a distant waypoint as a target and gradually converges to that point (Fig. 2(a)). By contrast with a the smaller Lookahead Distance, the algorithm

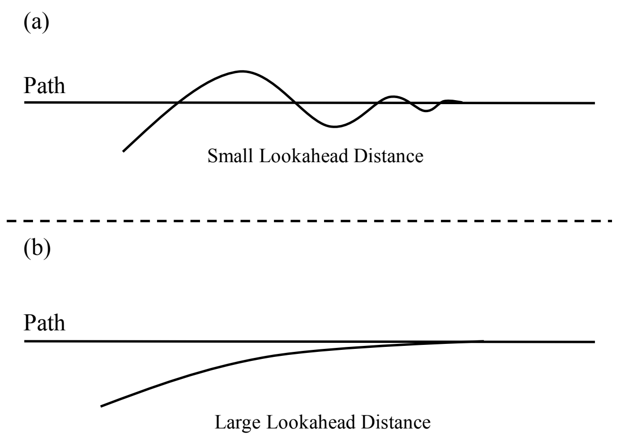


Fig. 2: Effect of Lookahead Distance[1]

determines a nearby waypoint as the target and quickly moves to the target (Fig. 2 (b)). In other words, when the Lookahead Distance value is large, the change in angular velocity is small and the vehicle tends to move straight. On the other hand, when the Lookahead Distance value in small, the change in angular velocity is large and the vehicle tend to turn.

The vehicle was unable to navigate a sharp curve at high speed. Thus, it is better to set the Lookahead Distance to a large value when the vehicle's speed is high. By contrast, Lookahead Distance should be small at low speeds.

We dynamically modify the Lookahead Distance relative to velocity. The Lookahead Distance is defined as the distance that the vehicle will reach in a certain seconds at the current linear velocity, which is expressed as follows:

$$l_d = v_c x$$

Here, l_d [m] is the Lookahead Distance, v_c [m/s] is the current linear velocity and x [s] is a magnification parameter. To prevent divergence of the control owing to tracking too near target waypoints, the minimum Lookahead Distance is also parameterized.

B. Waypoint Search

We use two waypoint search methods. One is waypoint search of the target and the other is waypoint search of the closest waypoint of the vehicle. Here, we describe how to search the closest waypoint. When the path has many waypoints when searching for the target waypoint, it leads to an increase in the calculation time required for the algorithm. We reduced the time required for calculation by searching between the closest waypoint and the next waypoint using the Lookahead Distance as follows.

1) Define the search circle.
2) Calculate the distance between the waypoint in the circle and the vehicle's position. Determine the closest waypoint from this result.
3) Predict the waypoint that the vehicle would have reached if there are no waypoints in the circle, i.e., the vehicle strays from the path. Determine the closest waypoint between the previous closest waypoint and the predicted waypoint.
4) If the vehicle strays from the path by a distance greater than a given threshold, a the system outputs a message indicating that driving is not possible.

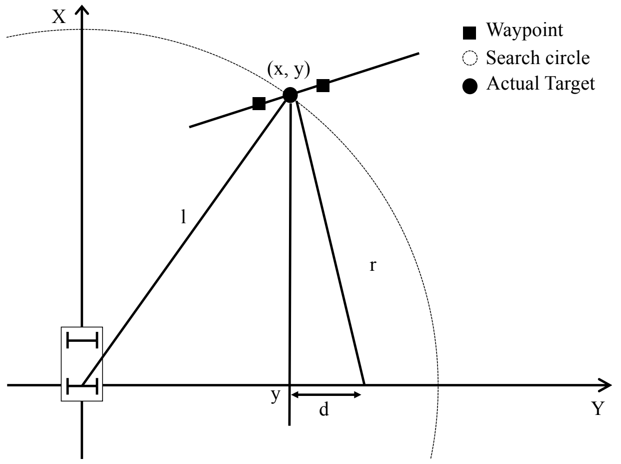


Fig. 3: Illustration about linear interpolation.

C. Linear Interpolation

Angular velocity, calculated from the discrete waypoints on the given path, is considered to be the reason for meandering or unstable steering control Therefore, we have modified the Pure Pursuit algorithm to output continuous operation commands.

In the Pure Pursuit algorithm, the next waypoint is the closest waypoint outside of the given search circle. As mentioned previously, the origin of the search circle is the center of rear axle. We define a new point where the intersection of a the straight line passing through the next waypoint and the previous waypoint with the search circle is the actual target point. This approach makes continuous acquisition of the target point from the path possible. We call this approach Linear Interpolation. Figure 3 illustrates Linear Interpolation.

V. Experimental Setup

This section describes the software architecture of the autonomous driving platform, the autonomous vehicle, the simulation environment and the test fields.

A. Software Architecture

We use Autoware as the software architecture of the autonomous driving platform for our vehicle. Autoware is based on the Robot Operating System (ROS)[13], which is a meta-operating system for robots. The ROS is abstracted by nodes and topics, where nodes represent individual functions and topics store the data communicated between nodes.

Our software is roughly organized into five components:

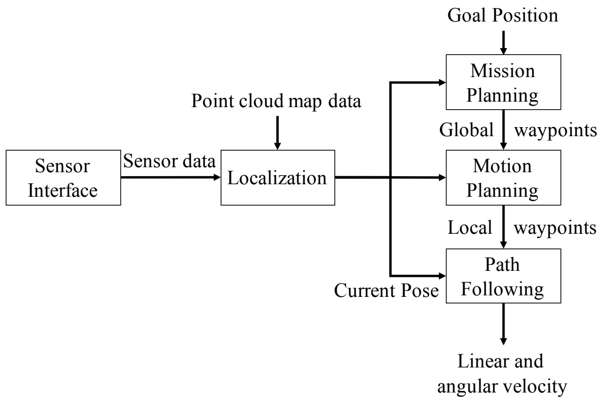


Fig. 4: Data flow of Autoware

- Sensor Interface - The sensor Interface acquires sensor data and converts the data to usable information in the software component.
- Localization - Localization estimates the position of the vehicle using sensor data and 3D map.
- Mission Planning - Mission Planning generates global waypoints, i.e., a path from current position coordinates to the goal position coordinates
- Motion Planning - Motion Planning plans the velocity in consideration of the curvature of the given path, the legal speed limit and traffic lights to generate local waypoints.
- Path Following - Path Following calculates control commands to follow the local waypoints.

Figure 4 shows the data flow between components in Autoware.

B. Vehicle

The vehicle (Fig. 5(a)) used in this study is the Robocar HV, a modified Toyota Prius, produced by ZMP[14]. Robocar HV can connect to a computer. The given computer can obtain Controller Area Network (CAN) information. In addition, when the given computer sends operational commands, Robocar HV can drive automatically .

To acquire 3D real-world data, a Velodyne HDL-32 (Fig. 5(b))[15] is mounted on the roof of the vehicle. This sensor, which has 32 lasers and spins at 10 [Hz], generates point-cloud data covering a 360-degree horizontal field-of-view and a nearly 30-degree vertical field-of-view. We determine the location of the vehicle performed using scan matching[16] by point-cloud data and the 3D map created by Mobile Mapping System (MMS)[17].

Two laptop computers are installed in the vehicle. One computer executes autonomous driving software, the other controls Robocar HV.

C. Experimental Conditions

1) Simulation: We constructed the simulation environment in consideration of ideal vehicle dynamics. In this environment, the 2D position of the vehicle is determined using the following equations.

$$
\begin{aligned}
x_{i+1} &= x_i + v\cos\theta_i \Delta t \\
y_{i+1} &= y_i + v\sin\theta_i \Delta t \\
\theta_{i+1} &= \theta_i + \omega \Delta t \\
i &: 0, 1, ..., n
\end{aligned}
$$

Here, (x_0, y_0, θ_0) is the initial position of the vehicle, v is the current linear velocity and ω is the current angular velocity. Operational commands are substituted into these equations sequentially, and the vehicle moves within the virtual coordinates. The simulation environment demonstrates that the modification works effectively.

2) Actual Field: We performed two field tests.

The first test field is performed in a mock urban environment, i.e., Toyota City Safety Learning Center. This site has narrow roads, traffic lights and crosswalks. It is difficult to operate the vehicle at high speed because the area is not less than 100 [m] wide. Therefore, we conducted an evaluation at low speed, i.e., less than 20 [km/h].

Figure 6 is a point-cloud and vector map view of the Toyota City Traffic Safety Learning Center.

The second field test is conducted on a public road in Moriyama Ward, Nagoya City, Aichi Prefecture. The length

(a) Robocar HV (b) Velodyne HDL32-E

Fig. 5: Hardware for Autonomous driving

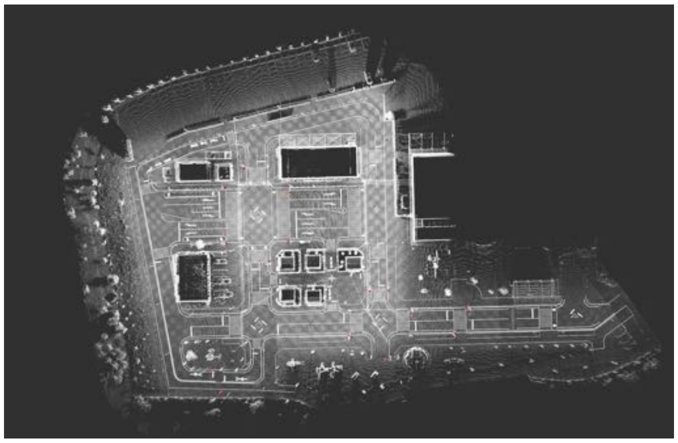
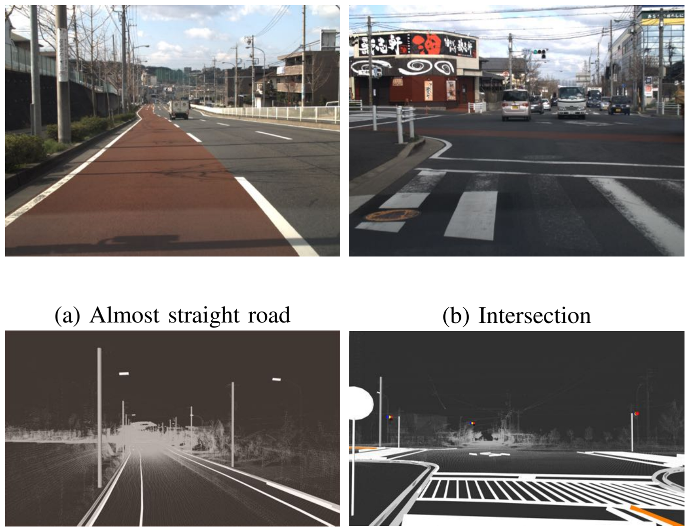


Fig. 6: Point cloud and vector map view of Toyota City Traffic Safety Learning Center



(a) Almost straight road

(b) Intersection

(c) Almost straight road shown by pointcloud and vector data

(d) Intersection shown by pointcloud and vector data

Fig. 7: Moriyama Ward

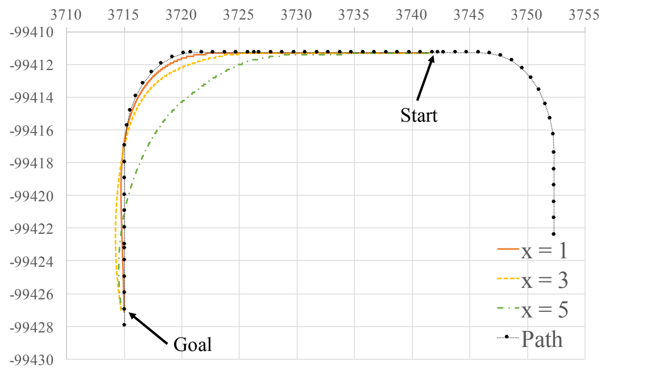


Fig. 8: Lookahead Distance results due to changing parameter x

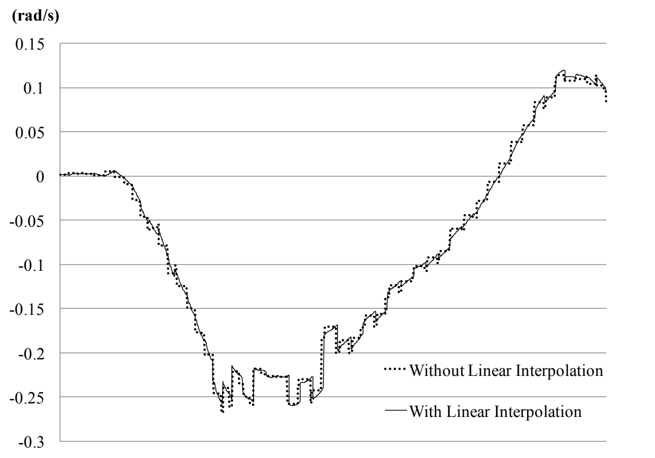


Fig. 9: Angular velocity with and without Linear Interpolation

of this course is approximately 5 [km]. This course is a long smooth road with some intersections, as shown in Fig. 7(a) and 7(b), respectively. We travel along the driving lane at the maximum legal speed, that is, 60 [km/h].

VI. Results

Here, we present out the evaluation results of parameter tuning and modification of the Pure Pursuit algorithm. In addition, we describe the field operational tests in detail.

A. Simulation

We tuned the Lookahead Distance parameter and applied Linear Interpolation for the Pure Pursuit algorithm and evaluated their performance.

1) Lookahead Distance: Figure 8 shows the differences in angular velocity caused due to changes in the Lookahead Distance parameter. The vehicle moved along the straight road at 10 [km/h] (2.78 [m/s]), and along the curved road at 5.5 [km/h] (1.52 [m/s]). The minimum Lookahead Distance was set to 4 [m].

- $x = 1$
  The trajectory nearly matched the Path. In this situation, the Lookahead Distance at the curve was the minimum value.
- $x = 3$
  The vehicle drove slightly inside the curve .
- $x = 5$
  The vehicle drove significantly inside the curve due to the distant target waypoint.

2) Linear Interpolation: Angular velocity changes with or without Linear Interpolation are plotted in Fig. 9 (dotted line, angular velocity without Linear Interpolation; solid line, angular velocity with Linear Interpolation). The dotted line indicates discrete control steps, and the solid line is smooth, that is, continuous control steps.

B. Field Operational Test

We conducted field operational tests in the mock environment(Fig. 11) and on the public road (Fig. 12). Videos of the autonomous driving experiment are available.

- Public Road (Moriyama Ward)
  https://www.youtube.com/watch?v=Zz5bKOxncVU
- Mock Urban Environment (Toyota City Traffic Safety Learning Center)
  https://www.youtube.com/watch?v=NWNpkBpdmAc

In this time, we tested on the corner which is difficult for driving with the Pure Pursuit algorithm. The autonomous driving trajectory in the mock urban environment is plotted in Fig. 10 (dotted line, given path; solid line, vehicle trajectory). The vehicle turned outside of the curve safely and followed the straight line.

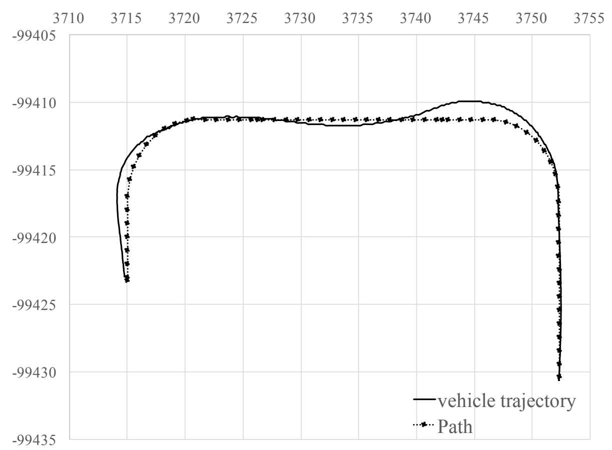


Fig. 10: Comparison between the given path and trajectory obtained by modified Pure Pursuit

Fig. 11: Toyota City Traffic Safety Learning Center field test

VII. Conclusion

In this study, we set path following as the main objective and implemented the Pure Pursuit algorithm in an autonomous driving platform. Through simulations and field tests, We identified problems.

To address problems described in Section I, we reviewed the Pure Pursuit algorithm. We modified the algorithm by tuning parameters such as the Lookahead Distance and incorporated Linear Interpolation.

We then performed simulations and field tests to evaluate our modifications. we conclude that it is important to set the Lookahead Distance parameter appropriately in consideration of the sharpness of the curve and the current velocity. In addition, we verified that Linear Interpolation result in the continuous change of angular velocity.

Fig. 12: Field Operational Test in Moriyama Ward

References

[1] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-92-01, January 1992.

[2] Y. J. Kanayama and F. Fahroo, "A new line tracking method for non-holonomic vehicles," in *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, vol. 4, Apr 1997, pp. 2908–2913 vol.4.

[3] R. W. Brockett, "Asymptotic stability and feedback stabilization," in *Differential Geometric Control Theory*. Birkhauser, 1983, pp. 181–191.

[4] "DARPA Urban Challenge," http://archive.darpa.mil/grandchallenge/.

[5] C. Urmson, J. Anhalt, H. Bae, J. A. D. Bagnell, C. R. Baker, R. E. Bittner, T. Brown, M. N. Clark, M. Darms, D. Demitrish, J. M. Dolan, D. Duggins, D. Ferguson , T. Galatali, C. M. Geyer, M. Gittleman, S. Harbaugh, M. Hebert , T. Howard, S. Kolski, M. Likhachev , B. Litkouhi, A. Kelly , M. McNaughton, N. Miller, J. Nickolaou, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, V. Sadekar, B. Salesky, Y.-W. Seo, S. Singh, J. M. Snider, J. C. Struble, A. T. Stentz , M. Taylor , W. R. L. Whittaker, Z. Wolkowicki, W. Zhang, and J. Ziglar, "Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I*, vol. 25, no. 8, pp. 425–466, June 2008.

[6] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun, "Junior: The Stanford Entry in the Urban Challenge," *Journal of Field Robotics, Special Issue on the 2007 DARPA Urban Challenge, Part II*, vol. 25, no. 9, pp. 569–597, Sep. 2008.

[7] J. Leonard, D. Barrett, J. How, S. Teller, M. Antone, S. Campbell, A. Epstein, G. Fiore, L. Fletcher, E. Frazzoli, A. S. Huang, T. Jones, O. Koch, Y. Kuwata, K. Mahelona, D. Moore, K. Moyer, E. Olson, S. Peters, C. Sanders, J. Teo, and M. Walter, "Team MIT Urban Challenge Technical Report," Tech. Rep., 2007. [Online]. Available: leonardTR07.pdf

[8] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Reinholtz, D. Hong, A. Wicks, T. Alberi, D. Anderson *et al.*, ""Odin: Team victortango's entry in the darpa urban challenge"," *Journal of Field Robotics*, vol. 25, no. 8, pp. 467–492, 2008.

[9] I. Miller, M. Campbell, D. Huttenlocher, F.-R. Kline, A. Nathan, S. Lupashin, J. Catlin, B. Schimpf, P. Moran, N. Zych *et al.*, ""Team Cornell's Skynet: Robust perception and planning in an urban environment"," *Journal of Field Robotics*, vol. 25, no. 8, pp. 493–527, 2008.

[10] J. M. Snider, "Automatic steering methods for autonomous automobile path tracking," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-09-08, February 2009.

[11] S. F. Campbell, "Steering control of an autonomous ground vehicle with application to the DARPA urban challenge," *University of Notre Dame*, 2005.

[12] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, "An open approach to autonomous vehicles," *IEEE Micro*, vol. 35, no. 6, pp. 60–69, 2015.

[13] "ROS Wiki," http://wiki.ros.org/ja/tf.

[14] "RoboCar ® PHV/HV," http://www.zmp.co.jp/products/robocar-phv.

[15] "Velodyne Lidar," http://www.velodynelidar.com/lidar/lidar.aspx.

[16] E. Takeuchi and T. Tsubouchi, ""A 3-D scan matching using improved 3-D normal distributions transform for mobile robotic mapping"," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, 2006, pp. 3068–3073.

[17] "Mobile Mapping System（MMS)," http://www.whatmms.com/.