# Comparing the performance of lateral control algorithms on long rigid vehicles in urban environments

Artyom Boyarov1

1 Whitgift School, Croydon, United Kingdom

Email address: [artyom.boyarov@gmail.com](mailto:artyom.boyarov@gmail.com)

ORCID: 0000-0001-9043-7393

## Keywords

Autonomous vehicles; control; lateral control; path following; kinematic control

## INTRODUCTION

Self-driving cars have been developing at an increasing rate over the past few decades, starting with basic highway driving in the 1980s to almost full autonomous driving capability showcased in the DARPA Grand Challenges of the 2000s [2]. Autonomous vehicles feature software handling perception, navigation, and vehicle control [3]. The perception module uses sensor data to produce a map of the environment surrounding the vehicle. Navigation modules consist of mission and motion planners – the mission planner finds the route the car should take to reach its destination, while the motion planner generates a trajectory, based on the map of the surroundings provided by the perception module, for a car to follow locally to reach the next waypoint that the mission planner has given. For example, the BOSS vehicle from Carnegie Mellon University used a model predictive trajectory generator as its motion planning module in the DARPA Urban Challenge [3]. The controller of the car manages lateral and longitudinal control. Lateral control ensures that the vehicle follows the given trajectory and at each iteration the control algorithm used returns the steering angle needed to follow the trajectory. The longitudinal controller ensures the car follows the given speed, usually using a PID controller.

Multiple algorithms for lateral control have been developed, based mainly on kinematic models and predictive models. Algorithms based on kinematic models use a bicycle model to represent the vehicle, where the car is represented as a two-wheeled body with the front wheel providing turning [4]. Kinematic model-based controllers are simple to implement, involve a low computation overhead and provide reasonable performance in fixed scenarios at a moderate speed, such as in urban settings where the speed usually does not exceed 30 miles per hour [4]. Kinematic controllers were used by multiple vehicles in the DARPA Urban Challenge, including the winning vehicle, Stanley. A further three vehicles used the pure pursuit kinematic controller in the same challenge [4].

Existing research focuses on comparing the effectiveness of such algorithms when applied to conventional cars. Not much research focuses on autonomous buses, however, they can be more beneficial for the environment and can improve public transport [1]. If full autonomous capability is to be achieved, especially in urban environments, vehicles such as buses or trucks also need to be made autonomous. Buses and trucks have a longer length and therefore the kinematic model will produce different results. For example, the angular velocity of the vehicle will decrease as the length between the two axles increases, making turning harder. Investigating the performance of existing kinematic control algorithms on longer vehicles would give valuable insights on how the algorithms could be adjusted to cope with longer lengths (such as how gain parameters could be adjusted) or whether new algorithms need to be designed. Some research has been conducted into various control algorithms for articulated vehicles [5], however for non-articulated vehicles of a long length the performance of control algorithms has not yet been discussed, which is the basis of my investigation.

Therefore, the hypothesis of my investigation is to compare the performance of various lateral control algorithms when used to control a long vehicle on a trajectory like one in an urban environment. The investigation compared the Stanley and pure pursuit control algorithms, as well as two algorithms based on the Stanley and pure pursuit controllers which aim to counteract their shortcomings. The Stanley and Pure Pursuit algorithms were implemented based in the available literature and publicly available code [4,6]. The first new controller implemented was a combined Stanley and Pure Pursuit controller which would operate based on the current cross track error. If the cross-track error is larger than the set threshold, the vehicle used the Pure Pursuit control algorithm to return to the trajectory. Otherwise, the Stanley control algorithm is used as it can effectively return a car to its trajectory with minimal steering. The second new control algorithm implemented was a modified Stanley controller with lookahead turning: the controller would use the heading of a point on a trajectory ahead of the vehicle’s position to set the current steering angle. This algorithm was chosen as for longer vehicles, the larger distance between the axles would make turning harder, and so by turning earlier the vehicle would be able to go around tight turns easier. The investigation measured the cumulative cross track error (measured from the car’s front axle) as well as the steering angle over time. The major conclusion drawn from the research was that no control algorithm was able to provide a low cross track error as well as smooth steering, indicating that an optimal control algorithm is yet to be developed, or other controllers using predictive models could perform better. This investigation could be extended to environments similar to urban ones, such as a warehouse where space is limited and robots travel on fixed paths, or in an airport where transit buses and other vehicles have designated roads for them to travel on.

## experimental METHODS

A kinematic simulation of a two-axled vehicle was developed in the Python programming language with the Matplotlib library providing a graphical frontend. The bicycle kinematic model was used to simulate the vehicle travelling. The simulation was run at a fixed time step of 0.1 seconds. Trajectories for the simulation were made up of straight segments and curved line segments; points on the trajectories were generated using linear interpolations and geometric slerps. Each control algorithm was implemented as a separate function which would return a steering angle on each step based on the current position of the car. Control algorithms were implemented using the available literature [4] as well as the *PythonRobotics* GitHub repository [6]. While the simulation was running, values for the steering angle and cross track error (measured as the distance from the front axle to the nearest waypoint) were recorded and then stored in a csv file at the end of the simulation. These values were then processed and analyzed. The code is publicly available at <https://github.com/heemogoblin/trajectory-following-simulation>.

## RESULTS

The control algorithms were compared in a simulation which simulated a vehicle travelling at 50km/h and responding to the steering angle provided by the lateral control algorithm being tested. The vehicle’s cross track error from the front axle and the steering angle were recorded. The cross-track error was chosen to be measured from the front axle as having the front axle of the car on the given trajectory will minimize the possibility of the front of the car hitting obstacles. A lower total cross track error also indicates that the controller successfully follows the trajectory. The steering angle was measured for two reasons; firstly, large turns will cause a vehicle to risk overturning. Secondly, having sudden changes in steering angle will cause the vehicle to jolt, thereby reducing passenger comfort. After results were collected in the simulation they were processed and analyzed. The car’s speed was set to 50 km/h, roughly 30 mph, as this is the speed limit for urban areas in many countries around the world [7].

The first course chosen featured two sections of straight track connected with two turns. The pure pursuit controller offered low steering angles and smooth turns, whereas the Stanley, Stanley with lookahead and Stanley with Pure Pursuit controller had a larger steering angle and more sudden changes in steering angle (Figure 1)[[1]](#footnote-1). However, for cross track error, the Stanley and combined Stanley and pure pursuit controllers had a much lower cross track error than the pure pursuit and Stanley with lookahead controllers (Figure 2).

In the second course the vehicle took a three-quarter turn around a roundabout. The pure pursuit controller had smoother steering than the other controllers. The Stanley controller had sudden changes in steering angle and for some sections of the course was at the maximum steering angle. The Stanley with lookahead and combined Stanley and Pure pursuit controller had more sudden changes in steering angle, however they did not have such a high steering angle as the Stanley controller (Figure 3). The Stanley controller had the lowest cross track error, however the difference in cross track error was not so large compared to the previous course (Figure 4).

The last course involved a vehicle traversing a roundabout fully. Similarly in the previous experiments, the pure pursuit controller had smooth steering, however at times it had a large steering angle. The other controllers all had large steering angles, as well as sudden changes in steering angle (Figure 5). The Stanley controller had the lowest cross track error. The Stanley with lookahead had a similar total cross track error to the combined Stanley and pure pursuit controller, and the pure pursuit controller had the largest cross track error (Figure 6).

## DISCUSSION

Each of the experiments presented a similar pattern of results: the pure pursuit controller would have smooth steering and a low steering angle, and the Stanley controller would have the lowest cross track error. For some experiments the cross-track error of the other controllers compared did not differ greatly from the cross track error of the Stanley controller, namely the three-quarter turn on the roundabout course. The combined pure pursuit controller and Stanley controller with lookahead had a lower cross track error than the pure pursuit controller, however both controllers had sudden changes in steering angle as well as a large steering angle. The pure pursuit controller was unable to steer round the turns without having a large cross track error. For the Stanley controller and the two new controllers, the sharp steering angle resulted from the steering angle changing during the turns as the controller changed its steering angle based on the current heading of the trajectory.

The simulation was quite accurate as it was not affected by external factors. Also, the measurements used were entirely accurate. When the experiments were repeated, the results obtained were very similar. However, the simulation had the speed fixed to 50 km/h, but in an actual scenario the speed of the vehicle will change, especially while turning. A more realistic simulator could have been used for the experiment, such as the CARLA simulator. This simulator features advanced vehicle physics and so the effect of turning on the speed of the vehicle could have been observed.

The results therefore indicate that no single controller offers optimal control of a vehicle. The Stanley controller offers a good cross track error, but at the cost of sudden changes in steering, which would be uncomfortable for passengers or goods. The pure pursuit controller had smoother turns; however, it had a large cross track error. A controller which can avoid the large change in steering angle while having a low cross track error is a possible next step for development. A possible implementation may be a modified Stanley controller which outputs a value for the rate of change of steering angle, which is clamped within a defined margin, so that the vehicle does not change the steering angle suddenly and instead gradually changes it while turning.

Apart from looking for an improved kinematic lateral control algorithm, the possible next step may be to look at predictive control algorithms such as model predictive control, which may offer improved performance, albeit at a higher computational cost. Articulated vehicles may also be investigated. Articulated trucks are very common, and articulated buses are used around the world. Articulated buses offer higher passenger capacity and faster transportation and so are a viable choice for urban transport systems [8].

## conclusion

Overall, the research showed that no control algorithm offered optimal performance. The pure pursuit algorithm had smooth steering, but a high cross track error. The Stanley control algorithm had a lower cross track error but had large changes in steering angle. The combined Stanley and pure pursuit control algorithm and the Stanley with lookahead algorithm had a lower cross track error than the pure pursuit controller, however they had high steering angles and sudden changes in steering angle. This indicates that no kinematic control algorithm discussed provides optimal results, and so an algorithm which addresses the shortcomings of the Stanley and pure pursuit controllers or a predictive algorithm such as model predictive control may produce better results.

## REFERENCES

1. Mouratidis, Kostas, and Victoria Cobeña Serrano. “Autonomous Buses: Intentions to Use, Passenger Experiences, and Suggestions for Improvement.” *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 76, 2021, pp. 321–35, doi:10.1016/j.trf.2020.12.007.

2. Campbell, Mark, et al. “Autonomous Driving in Urban Environments: Approaches, Lessons and Challenges.” Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 368, no. 1928, 2010, pp. 4649–72. Crossref, doi:10.1098/rsta.2010.0110.

3. Urmson, Chris, et al. “Autonomous Driving in Traffic: Boss and the Urban Challenge.” *AI magazine*, vol. 30, no. 2, 2009, p. 17. Crossref, doi:10.1609/aimag.v30i2.2238.

4. Paden, Brian, et al. “A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles.” IEEE Transactions on Intelligent Vehicles, vol. 1, no. 1, 2016, pp. 33–55. Crossref, doi:10.1109/tiv.2016.2578706.

5. P. Bolzern and A. Locatelli, "A comparative study of different solutions to the path-tracking problem for an articulated vehicle," Proceedings of the International Conference on Control Applications, 2002, pp. 427-434 vol.1, doi: 10.1109/CCA.2002.1040224.

6. Atsushi Sakai, et al. "PythonRobotics: a Python code collection of robotics algorithms." (2018).

7. Wikipedia contributors. “Speed Limits by Country.” Wikipedia, 24 Feb. 2022, en.wikipedia.org/wiki/Speed\_limits\_by\_country.

8. El-Geneidy, Ahmed, and Nithya Vijayakumar. “The Effects of Articulated Buses on Dwell and Running Times.” Journal of Public Transportation, vol. 14, no. 3, 2011, pp. 63–86. Crossref, doi:10.5038/2375-0901.14.3.4.

## Statements & Declarations

### Funding

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

### Competing Interests

Author Artyom Boyarov declares he has no financial interests.

### Authors’ Contributions

The main author, Artyom Boyarov, performed all of the research involved in writing the manuscript. He developed the simulation used; carried out the research; analyzed the data and wrote the manuscript.

### Ethics Approval

This study did not involve any living subjects, hence does not require an ethical approval.

### Consent for Publication

All participants consent to publish this manuscript.

### Code or Data Availability

The code for the simulation used is available at <https://github.com/heemogoblin/trajectory-following-simulation>. The results are also available in this repository under the ‘results/’ folder.

### Acknowledgements

I would like to thank Professor Howie Choset of Carnegie Mellon University for reading through my manuscript. I would also like to thank Ilya Makarov for proof-reading my manuscript and helping me with refactoring. I would also like to thank my parents Evgeny Boyarov and Anna Gavrilova for their assistance and support during my study and research.

## Figures and Figure Captions

**Fig. 1.** Steering angle of vehicle against time at 50 km/h on the straight track course

**Fig. 2.** Total cross track error of vehicle at 50 km/h on the straight track course

**Fig. 3.** Steering angle of vehicle against time at 50 km/h on the three-quarter turn course

**Fig. 4.** Total cross track error of vehicle at 50 km/h on the three-quarter turn course

**Fig. 5.** Steering angle of vehicle against time at 50 km/h on the roundabout course

**Fig. 6.** Total cross track error of vehicle at 50 km/h on the roundabout course

1. The steering angle for the Stanley controller is not so visible in figure 1. This is because the Stanley and Combined Stanley and Pure Pursuit controllers had the same steering angle throughout the simulation. [↑](#footnote-ref-1)