A Pseudo-Derivative Method for Sliding Window Path Mapping in Robotics-Based Image Processing

Landon Bentley*, Joe MacInnes†, Rahul Bhadani‡, and Tamal Bose§
*lcbentley@crimson.ua.edu, †jmacinnes19@wooster.edu, ‡rahulbhadani@email.arizona.edu, §tbose@email.arizona.edu

*Department of Computer Science, The University of Alabama, Tuscaloosa, AL 35487, USA
† Department of Computer Science, The College of Wooster, Wooster, OH 44691, USA
‡§ Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85719, USA

***Abstract*—A sliding window technique in robotics-based image processing applications is a common approach to path mapping from extracted features. Mapping a path inside an image requires finding a series of points representing the path. Previous approaches find these points by sliding a window along the path in fixed increments across one image dimension. After each slide, the center of the window in the other dimension is adjusted so that the window maximally covers the path in that area. This approach, however, fails to map paths that experience sharp curvature since the windows slide along only one dimension. The method proposed herein uses a pseudo-derivative approach to sliding windows that improves upon the traditional technique by dynamically adjusting the windows along both image dimensions during each slide. In this method, the directional components of a vector representing the previous slide are used as a naive estimation to perform the current slide. If this fails to map the path, the vector direction is used to enlarge the window dimensions. The method was tested in the domain of autonomous vehicles as an approach for detecting road lane markings. The algorithm proved more successful than previous sliding window approaches on perspective mapped lane images.**

I. Introduction

Path detection and modeling is an essential part of various robotics applications, such as lane keeping systems in autonomous vehicle design [1] and Unmanned Aerial Vehicle (UAV) road mapping [2]. Object detection within images is a fundamental yet challenging task in applications of computer vision, serving as the basis for several other high-level tasks such as license plate localization [3] and facial/gesture recognition [4]; however, many of these solutions implement a deep learning model that requires substantial computational power [5] [6].

The limitations associated with deep learning path mapping motivate the development of simpler, naive image processing methods. These methods have the potential to decrease the overhead of computational costs. A common technique used to map paths such as lanes is applying a series of Hough transforms into an image [7]. However, this procedure is still relatively costly when compared to some sliding window based approaches in known domains where one is able to make assumptions about the path space.

Such a sliding window path mapping algorithm is proposed in [8]. In the domain of lane mapping, one can assume that the paths begin at the bottom of the image, at certain positions. Thus, windows slide along the image from these points in fixed increments across a single image dimension (up); the entire image need not be searched.

Following each slide, the window then adjusts in the other dimension such that the window best encapsulates the path. Although this method provides satisfactory results when applied to smooth, continuous curves, there are various open issues that remain when using this technique:

1) Mapping paths with sharp curvature are not feasible.
2) Mapping discontinuous curves are also not feasible; the windows get lost between markings.

Contribution

The contribution of this paper is an improvement of the algorithm described above. The proposed algorithm uses sliding windows but changes the way in which they slide so that the windows can capture paths efficiently with sharp curvature and discontinuous paths.

II. Problem Statement

In lane detection, path mapping starts with a binary image containing the lanes, often perspective warped so that the x and y image coordinates map linearly to distance in the real world. Given such an image, a series of points must be found along each lane in order to represent the path mathematically.

The sliding window algorithm discussed in section I accomplishes the waypoints calculation; however, its performance remains poor with sharp curvature. A visualization of this algorithm implemented as presented in [8] can be seen in Figure 1. Another similar implementation can be found in [9].

In its state described above, the algorithm is unable to account for curvature above a certain threshold or discontinuous lanes. Since the windows are restricted to slide vertically up the height of the window, if no lane markings are detected inside the window, it is nonetheless slid vertically again. This causes the search window to overstep the lane in the image. These failures can be seen in Figure 2, which shows the poor tracking of the sliding windows on extremely curved lanes, one of which has a dashed marking.

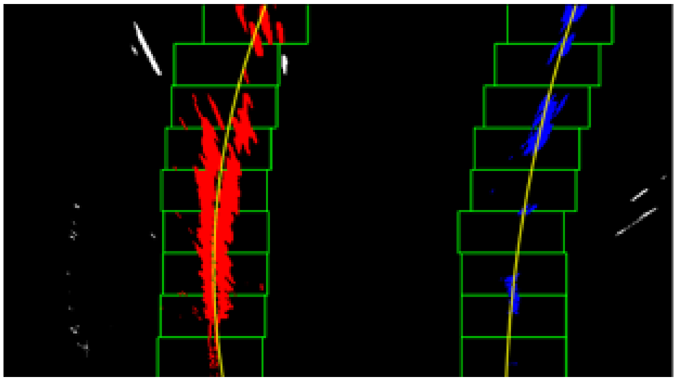

Fig. 1: A visualization of the sliding window algorithm from [8].

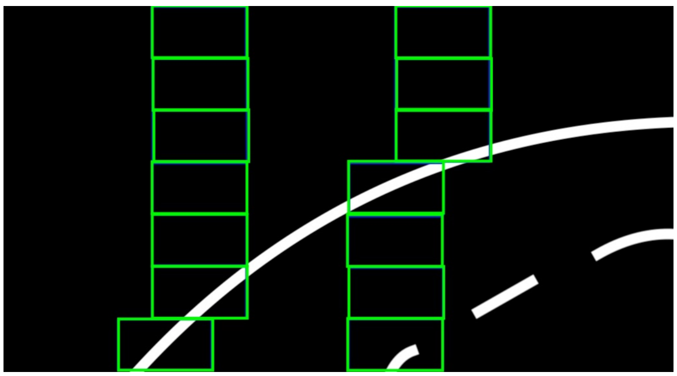

Fig. 2: The poor performance of the described sliding window approach on curved lanes.

III. Pseudo-Derivation Method

In order to improve upon the existing implementations, the proposed approach changes the window slide from a fixed vertical direction to a pseudo-derivative approach. First, the initial points of the lanes are discovered using k-means clustering on the bottom section of the image. Then, after an initial upward slide, the previous slide vectors are used to perform the current slide. We use a set step size and window size to slide the window along the lane. To account for sharp curvature in the road, we calculate a vector along which to the shift the window. At first, this vector points in the vertical direction but it changes its direction after re-centering itself around road pixels it detects. In this manner, the window is not simply shifted upwards before re-centering but moves along the path. The vector along which the window slide acts as a pseudo-derivative of the lane being mapped.

Figure 3 shows an example of this process. The gray dots represent lane marking pixels in the image space. The previous slide vector is $\boldsymbol{a}$, while the center of the resulting search window, α is shown by the left yellow circle. In order to perform the current slide, a search window β slides along the same vector, shown here as $\boldsymbol{b}$. This window then re-centers around the lane markings it captures resulting in the window γ. The vector between the center of γ and α is thus $\boldsymbol{c}$. The final vector, $\boldsymbol{d}$, used to place the next point along the lane, the right yellow circle, is the weighted sum of $\boldsymbol{b}$ and $\boldsymbol{c}$. The weights are proportional to the number of lane marking pixels detected in the corresponding search windows; therefore, $\boldsymbol{a}$ is weighted more heavily.

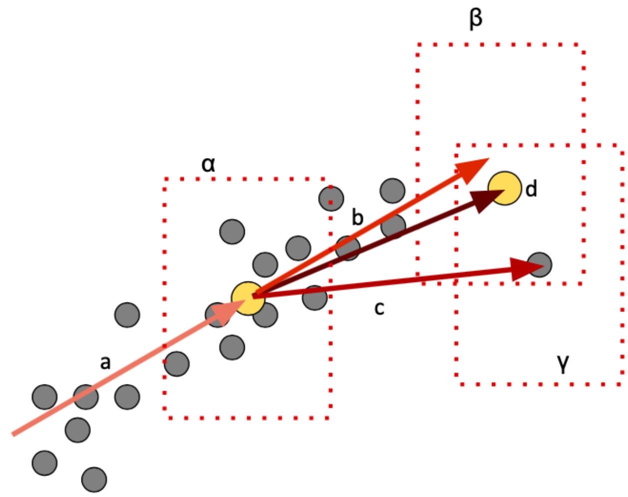

Fig. 3: The improved sliding window algorithm.

In the event that the search window does not detect any lane markings, it continues to move along its trajectory for a finite path until it does. If it still does not, it repeats this process by rotating the sliding vector within a threshold $\pm\epsilon$. The lane is considered mapped if these also fail to detect points. Finally, the average of the two-lane paths from detected lanes closest to the bottom center of the image (where the vehicle is located) is used to find the midpoints of the lane.

We tested this improved algorithm with the University of Arizona's autonomous research vehicle called the CAT Vehicle (Cognitive and Autonomous Test Vehicle) [10] and a stereocamera mounted on the top of the vehicle. Its results on an image captured by the vehicle are shown in Figure 4. Here, the centers of the search windows are shown in green. The purple points are the midpoints of the lanes.

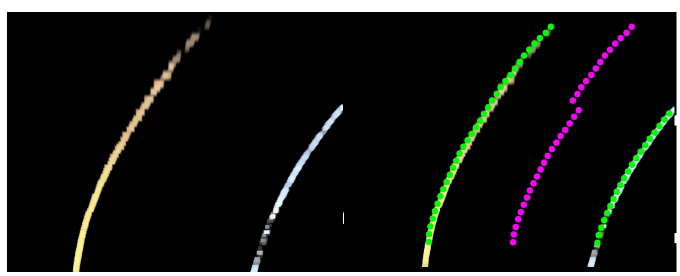

Fig. 4: The improved sliding window algorithm results on a real life perspective transformed image.

All source code used for the work mentioned in this paper are available open-source and can be download from the github repository `Lane Finder` **`https://github.com/catvehicle/Lanefinder`**.

IV. Conclusion

We propose an improvement to an existing algorithm used for path mapping in lane detection. The improvement increases the algorithm's ability to map curving lanes and lanes consisting of broken up lane markings. It does not sacrifice the algorithm's computational simplicity of beginning the path mapping where we would expect to find lanes.

ACKNOWLEDGMENT

This research was supported by National Science Foundation under award 1659428 and 1521617.

REFERENCES

[1] S. Huang and C. Tsai, "Vehicle lane detection and following based on vision system and laser scanner," in *2017 International Conference on Applied System Innovation (ICASI)*, May 2017, pp. 1446–1449.

[2] W. Feng, W. Yundong, and Z. Qiang, "Uav borne real-time road mapping system," in *2009 Joint Urban Remote Sensing Event*, May 2009, pp. 1–7.

[3] K. I. Kim, K. Jung, and J. H. Kim, "Color texture-based object detection: An application to license plate localization," in *Pattern Recognition with Support Vector Machines*, S.-W. Lee and A. Verri, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 293–309.

[4] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1–54, Jan 2015. [Online]. Available: https://doi.org/10.1007/s10462-012-9356-9

[5] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *CoRR*, vol. abs/1511.00561, 2015. [Online]. Available: http://arxiv.org/abs/1511.00561

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[7] F. Rovira-Más, Q. Zhang, J. F. Reid, and J. D. Will, "Hough-transform-based vision algorithm for crop row detection of an automated agricultural vehicle," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 219, no. 8, pp. 999–1010, 2005.

[8] M. Virgo, "Lane detection with deep learning," https://towardsdatascience.com/lane-detection-with-deep-learning-part-1-9e096f3320b7, 2017, online; accessed 24 June 2018.

[9] K. Ficici, "Curve lane detection," https://www.hackster.io/kemfic/curved-lane-detection-34f771, 2018.

[10] R. Bhadani, J. Sprinkle, and M. Bunting, "The CAT Vehicle Testbed: A Simulator with Hardware in the Loop for Autonomous Vehicle Applications," in *Proceedings 2nd International Workshop on Safe Control of Autonomous Vehicles (SCAV 2018), Porto, Portugal, Electronic Proceedings in Theoretical Computer Science*, vol. 269, 04/2018 2018, Workshop.