Towards Self-Driving Car: Lane Line Detection and Path Planning

Adam Jones, Vidush Vishwanath, Devashish Kashikar, Sarvagya Vatsal Singh

Department of Electrical and Computer Engineering, University of California, Davis CA 95616

Project Goals

To apply traditional computer vision to perform lane line detection and path planning.

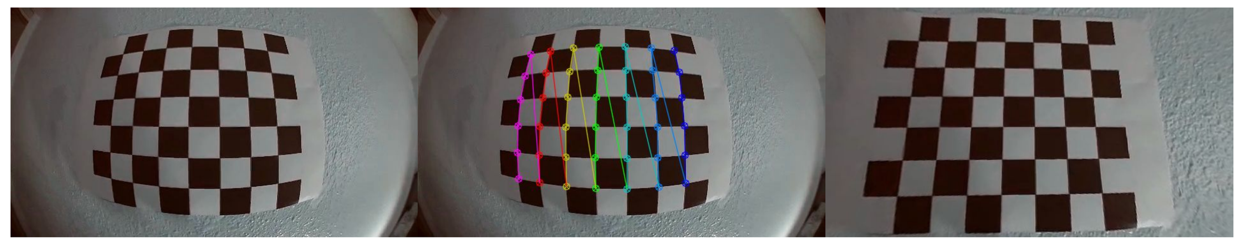
Approach and Milestones

In order to detect lane lines, we performed camera calibration, perspective transformation, color and gradient thresholding, and used a histogram-based window search. This allowed us to calculate the radii of curvature of the lane lines and the center offset of the car. In order to achieve path planning, we used SLAM and A*.

Lane Line Detection

Camera Calibration

Distorted **Drawn Corners** **Undistorted**



Since all camera lenses cause for some distortion, we first correct for these distortions in order to accurately measure the radii of curvature and the center offset. To correct for distortion, we take pictures of known shapes - in this case, a chessboard, and we estimate the calibration parameters to obtain undistorted images.

Undistorting the Lane Lines

Distorted **Undistorted**

Above shows the difference between distorted lane lines and undistorted lane lines.

Perspective Transformation

Once we have obtained the undistorted image, we apply a perspective transformation to get the birds-eye view of the track. We do this to remove unnecessary information from the image.

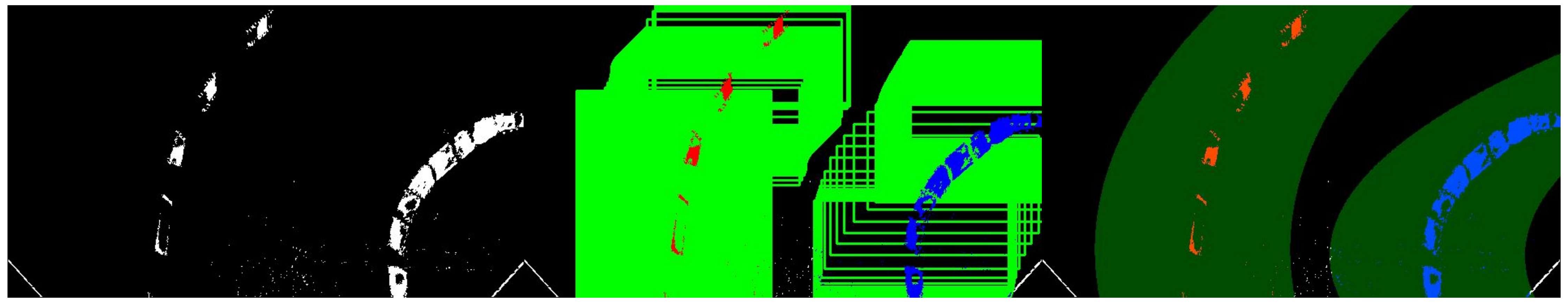
Vehicle View

Birds-Eye View

Window Search on Binary Image

We isolate the lane line pixels by applying color and gradient thresholds. Once the lane line pixels are isolated, we perform two simultaneous histogram-based window searches to classify each lane line pixel as belonging to either the left or right lane line. Finally, we fit a quadratic to each lane line and use this to calculate the radii of curvature and the center offset of the car.

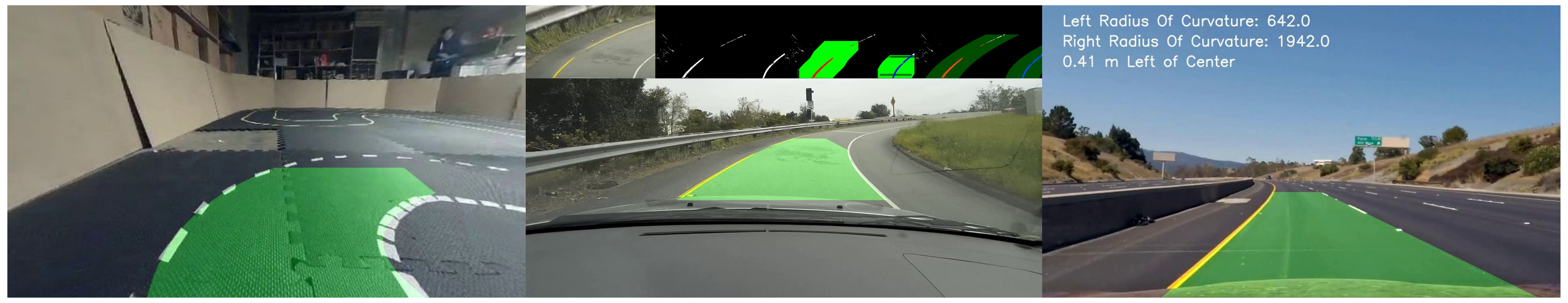
Binary Image **Window Search** **Best-fit Lines**



Lane Line Detection Final Results

Below are three examples of the final results using an RC car and two real cars.

RC Car **Real Car 1** **Real Car 2**

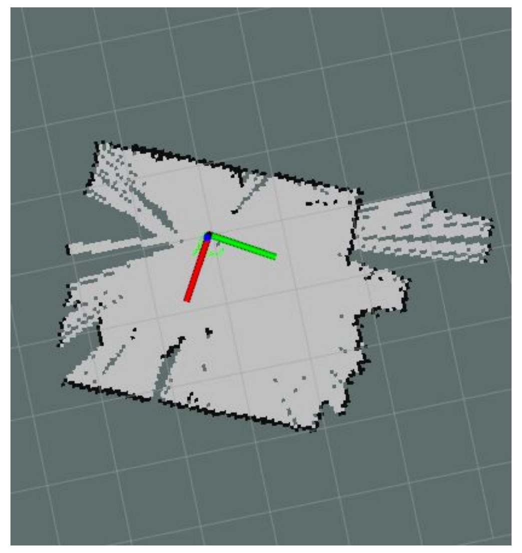


Path Planning

SLAM

The algorithm requires a representation of the area and a map to the location. We do this using LIDAR. It is used to gather information by sending a pulse of light until it hits the target. We can determine how far the target is, by determining how long it takes for the light to travel back to the sensor. Once we have this information we can start to map the area. By doing multiple 360 degree sweeps of the area, we can generate a map.

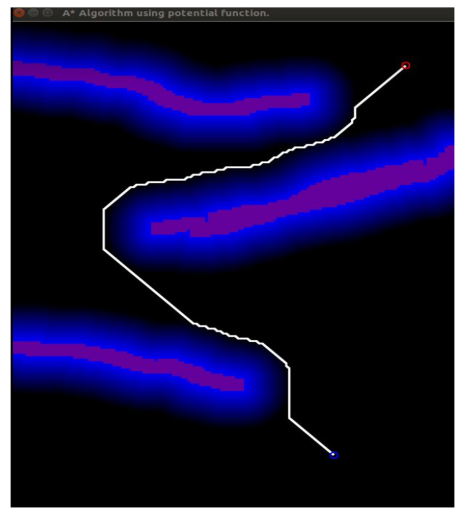
Lidar-generated Map



The A* Algorithm

With the map generated, we can implement Path Planning. The A* heuristic search is a graph algorithm that uses a greedy approach to find the most optimal path. By applying this with the possible obstacles in the area, we draw the 'desired path' to navigate the autonomous car.

Path Planning Result



Acknowledgement: The Titan XP used in this project was donated by NVIDIA Corporation. We also acknowledge the REU support from NSF CCMI-1301496 grant.