

Autonomous Emergency Navigation to a Safe Roadside Locations

November 2020 | Final Report



SAFE-D
SAFETY THROUGH DISRUPTION



**Texas A&M
Transportation
Institute**



**SAN DIEGO STATE
UNIVERSITY**
Leadership Starts Here



VT | **TRANSPORTATION INSTITUTE**
VIRGINIA TECH.

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by a grant from the U.S. Department of Transportation's University Transportation Centers Program. However, the U.S. Government assumes no liability for the contents or use thereof.

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. 03-073	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle Autonomous Emergency Navigation to a Safe Roadside Location		5. Report Date November 2020	
		6. Performing Organization Code: Department of Mechanical Engineering, Virginia Tech	
7. Author(s) Tomonari Furukawa Lei Zuo Robert G. Parker Lisheng Yang		8. Performing Organization Report No. Report 03-073	
9. Performing Organization Name and Address: Safe-D National UTC Virginia Tech Transportation Institute 3500 Transportation Research Plaza Blacksburg, VA 24061		10. Work Unit No.	
		11. Contract or Grant No. 69A3551747115/Project 03-073	
12. Sponsoring Agency Name and Address Office of the Secretary of Transportation (OST) U.S. Department of Transportation (US DOT)		13. Type of Report and Period Final Research Report	
		14. Sponsoring Agency Code	
15. Supplementary Notes This project was funded by the Safety through Disruption (Safe-D) National University Transportation Center, a grant from the U.S. Department of Transportation – Office of the Assistant Secretary for Research and Technology, University Transportation Centers Program.			
16. Abstract In this project, we developed essential modules for achieving the proposed autonomous emergency navigation function for an automated vehicle. We investigated and designed sensing solutions for safe roadside location identification, as well as control solutions for autonomous navigation to the identified location. Sensing capabilities are achieved by advanced fusion algorithms of 3D Lidar and stereo camera data. A novel control design, based on dynamic differential programming, was developed to efficiently plan navigation trajectories while dealing with computation delay and modelling errors. Preliminary validation of proposed solutions was carried out in a simulated environment. The results show strong potential for success, especially for the control module. Hardware integration in a real vehicle has been ongoing in a parallel fashion to enable field tests of developed modules in future work. Key sensing equipment was installed and calibrated and used to collect data for offline analysis. The retrofitting of the vehicle's actuation mechanism was finished with the whole drive-by-wire system in place. Future work will involve road testing the developed systems.			
17. Key Words Publication, guidelines, report, brochure, communication, marketing		18. Distribution Statement No restrictions. This document is available to the public through the Safe-D National UTC website , as well as the following repositories: VTechWorks , The National Transportation Library , The Transportation Library , Volpe National Transportation Systems Center , Federal Highway Administration Research Library , and the National Technical Reports Library .	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 30	22. Price \$0

Abstract

In this project, we developed essential modules for achieving the proposed autonomous emergency navigation function for an automated vehicle. We investigated and designed sensing solutions for safe roadside location identification, as well as control solutions for autonomous navigation to the identified location. Sensing capabilities are achieved by advanced fusion algorithms of 3D Lidar and stereo camera data. A novel control design, based on dynamic differential programming, was developed to efficiently plan navigation trajectories while dealing with computation delay and modelling errors. Preliminary validation of proposed solutions was carried out in a simulated environment. The results show strong potential for success, especially for the control module. Hardware integration in a real vehicle has been ongoing in a parallel fashion to enable field tests of developed modules in future work. Key sensing equipment was installed and calibrated and used to collect data for offline analysis. The retrofitting of the vehicle's actuation mechanism was finished with the whole drive-by-wire system in place. Future work will involve road testing the developed systems.

Acknowledgements

This project was funded by the Safety through Disruption (Safe-D) National University Transportation Center, a grant from the U.S. Department of Transportation – Office of the Assistant Secretary for Research and Technology, University Transportation Centers Program.

Table of Contents

1.INTRODUCTION.....	1
1.1 Original Research Proposal.....	1
1.2 Research Problems Summarization.....	2
2. ENVIRONMENT PERCEPTION SOLUTIONS.....	3
2.1 Related Work.....	3
2.2 Estimation Objective.....	4
2.3 System Configuration.....	4
2.4 Roadside Detection and Tracking Using Stereovision.....	5
2.4.1 B-Spline Principles.....	6
2.4.2 3D Roadside Surface Modeling.....	6
2.4.3 Road Vertical Profile Estimation.....	7
2.4.4 Road Surface Depth Map Construction.....	8
2.4.5 Boundary Objects Detection and Tracking in Stochastic Occupancy Grids.....	9
2.4.6 Roadside Tracking and Smoothing using Kalman Filter.....	10
2.5 Results from Offline Analysis.....	10
3.TRAJECTORY PLANNING AND CONTROL SOLUTIONS.....	12
3.1 Control Problem Statement.....	13
3.2 Emergency Stop Autonomous Control System.....	15
3.2.1 DDP Iteration.....	15
3.2.2 Recursive EKF.....	16
3.2.3 Receding Horizon Reinitialization.....	17
3.3 SIMULATION RESULTS.....	18
4.CONCLUSIONS AND RECOMMENDATIONS.....	20
5.ADDITIONAL PRODUCTS.....	21
5.1 Education and Workforce Development Products.....	21
5.2 Technology Transfer Products.....	21

REFERENCES23

APPENDIX A: PERCEPTION HARDWARE CONFIGURATION25

APPENDIX B: CONTROL HARDWARE CONFIGURATION27

Vehicle Longitudinal Control Actuator Modification28

Vehicle Lateral Control Actuator Modification.....30

List of Figures

Figure 1. Overall system architecture.	2
Figure 2. Roadside space estimation illustration.	4
Figure 3. Roadside perception system configuration.....	5
Figure 4. Stereovision based roadside area detection and tracking block diagram.	6
Figure 5. Detected boundary points and roadside width as the vehicle slows down.....	11
Figure 6. The estimated mean roadside width and its 3σ errors as the vehicle slows down.	12
Figure 7. Illustration of emergency stop navigation control problem	14
Figure 8. Overall control system diagram.....	15
Figure 9. Model for EKF estimation.....	17
Figure 10. Trajectories with no lag	19
Figure 11. Trajectories with 1-second lag.....	19
Figure 12 Trajectories in cases of narrow roadside segment.....	19
Figure 13. Vehicle sensor hardware wiring configuration.....	25
Figure 14. Power supply configuration of onboard computer and sensors.....	26
Figure 15. Equipment installation picture of the test vehicle.	27
Figure 16. Golf car retrofitting architecture for slow speed tests.	27
Figure 17. Drive-by-wire controller wiring configuration.....	28
Figure 18. A pulley is drilled and mounted on the steel hub of the motor shaft for brake actuation.	29
Figure 19. Motor driver gets feedback from CAN about the actual pedal position.....	29

1.Introduction

Today's prototype automated vehicles can track lanes effectively in normal operation, but when road emergencies such as system malfunction, automotive breakdown, etc. occur, these vehicles (for example, the Cadillac Super Cruise CT6 and Tesla Model S) simply stop in the lane, creating a potential safety risk for themselves and other vehicles. And for non-automated vehicles, emergency situations requiring safe stopping can occur as a result of medical incapacitation, seizures, or falling asleep. Whatever the case for the emergency stop, the safety need is for the vehicle to have the ability to navigate itself to stop at a safe roadside location. Our research objective is to develop Autonomous Emergency Navigation by perceiving the surrounding environment and road conditions, determining a safe roadside location to stop, specifying a trajectory, and applying proper steering and braking to autonomously drive the vehicle to the safe location outside the lane.

Original Research Proposal

As shown in Figure 1, our original plan was to design two modules for road perception: a localization and mapping module for far-field environment perception and a road surface estimation module for near-field road details profiling. The proposed approach would integrate the Simultaneous Localization and Mapping (SLAM) technology for autonomous driving and the road surface estimation technology for road condition monitoring, both of which were recently developed by Principal Investigator Tomonari Furukawa [1]. For the sake of real-time navigation, the localization and mapping technology performs its functions separately using a different set of sensors and numerical techniques. Here, it was proposed that the environment map be constructed using stereovision and robotic computer vision techniques. These techniques include Structure from Motion (SfM), which consists of feature tracking, outlier rejection, and data association. The use of vision data up to the maximal range is desired since the vehicle moves 29 m/s at 65 mph. The environment map was constructed for a range of 80 m. The map of the road, similar to the environment map, was constructed using stereovision and robotic computer vision techniques. The estimated road surface conditions include 3D geometry, friction coefficients, and any presence of defects such as potholes and cracks. For these features, the range was constrained to a shorter distance of 15–25 m where visibility was sufficient for measurement.

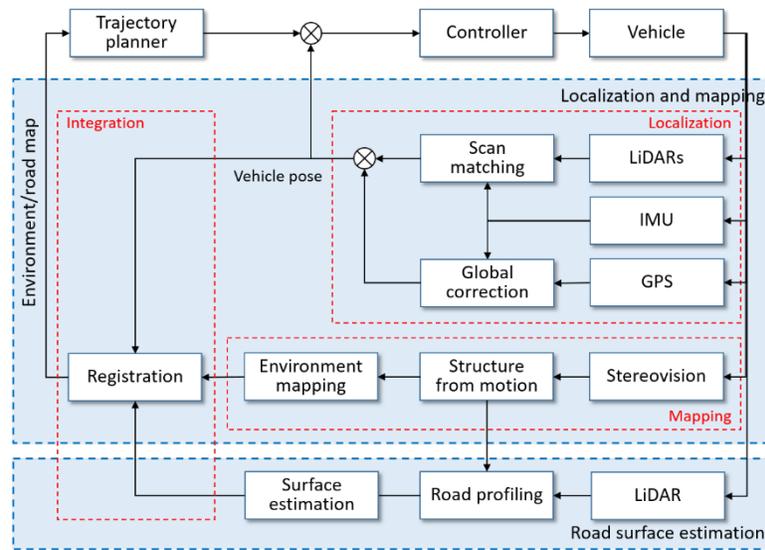


Figure 1. Overall system architecture.

Once the environment and the road surface can be measured, the remaining tasks are to identify a safe roadside location, determine the entire trajectory through a complete stop, and apply closed-loop control methods to execute the planned trajectory. While the identification of a safe roadside location can be considered equivalent to finding the first feasible location, the trajectory will be optimized by having the value function determined *a priori*. Since the autonomous vehicle shares the right-of-way with other objects, the trajectory generation needs to avoid collision with others (i.e., “the trajectory generation will be formulated as a non-holonomic motion planning problem with dynamic obstacles”). The pose and motion of the host vehicle and other objects will be predicted in a prediction horizon, based on the real-time identification of surrounding environments, including moving or standing vehicles and pedestrians. The Time-to-Collision between the host vehicle and other traffic participants will then be estimated and the host vehicle’s position will be modified accordingly.

The vehicle control in this collision free evasion maneuver will be closely and dynamically coupled with the trajectory planning. Both the longitudinal and lateral vehicle dynamics control will be integrated to track the trajectory reference, while the vertical control will maintain the vehicle’s comfort and stability. In the case of an evasive maneuver, the steering wheel and braking (accelerating) pedals will be controlled simultaneously to realize the dynamic trajectory, while taking into account the road cohesion coefficient and road conditions estimated by the road-profile measurement technology. The feedforward control will also be integrated for fast response.

Research Problems Summarization

1. Can the proposed localization and mapping provide the means to correctly identify a safe roadside harbor by:
 - a. Processing conventional camera images to yield an accurate, rich, 3D road profile within a 15–25 m radius

- b. Integrating localization and mapping with the road profile information
2. With the identification of safe roadside harbor, is it possible to:
 - a. Plan a suitable vehicle path trajectory
 - b. Accomplish low-level (feedback) steering and braking control for the vehicle navigation

2. Environment Perception Solutions

In this work, we aimed to detect a safe roadside location for parking in emergency situations. We defined a safe roadside area as the area outside of the rightmost solid lane marking with the same ground level as that of the main road. Using this definition, the problem of safe roadside location detection boils down to two sub-problems of solid lane detection and road obstacle detection. We need to first detect the rightmost solid lane, find any obstacles on the right side of that lane, and then calculate the available space between them. Finally, we need an optimal selection strategy to find a location that has enough space to park and also enough distance for our vehicle to pull in.

Related Work

Lane marking detection has been extensively studied in the context of lane tracking. Wang et al. [2] developed a B-snake based algorithm to extract lane markings from monocular images. Since the processing is all performed in the image domain, this method doesn't require camera parameters. Nedevschi et al. [3] adopted stereo-vision to recover real world dimensions, which are encoded in a lane model. This lane model is tracked by a Kalman filter and new lane markings are searched in a small region proposed by the Kalman prediction model. Later, Danescu et al. [4] extended this work by using particle filters to accommodate more challenging situations like lane discontinuity. While lane markings generally follow certain rules, and hence can be modelled and tracked, road obstacles can be more random and thus their detection requires more efforts. Kubota et al. [5] used dynamic programming to do a global segmentation between the road and the obstacles. Their technique is based on the v-disparity map [6] constructed from stereovision. Badino et al. [7] extended their work by introducing stochastic occupancy grids to track detected obstacles across frames. Wedel et al. [8] further extended their work by considering the road undulation in the v-disparity map construction. While these works were concerned more with global navigable space segmentation, other works placed more emphasis on road boundary detection. Li et al. [9] fused Lidar and image information to detect road curbs and proposed optimal drivable regions. Zhang et al. [10] used stereo-vision to find the road horizon and vanishing point, then used a graph cut method to find the road boundary by calculating the two shortest paths from the vanishing point to the image bottom.

These previous works have established mature methods for the detection and tracking of important road features, including lane markings and road obstacles. However, when it comes to roadside estimation for the vehicle emergency stop purpose, additional efforts need to be taken to integrate lane detection and road boundary detection techniques under a unified framework.

Estimation Objective

Figure 2 shows the estimation objective of the designed perception module. The vehicle is driven on the righthand side as required in the U.S. and thus focuses on navigable roadside on the right. As the lane change is not the problem of concern within this paper, we assume that the vehicle has already changed to the rightmost lane of the road prior to activation of the subject logic.

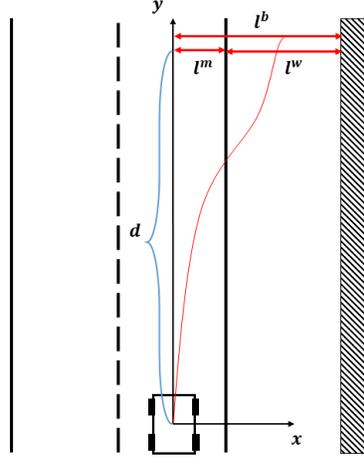


Figure 2. Roadside space estimation illustration.

Since the vehicle's geolocation is irrelevant to the automated control for an emergency stop, the inertial coordinate frame is set to align with the vehicle body frame where the emergency stop starts. This beginning time is set as time step 0. Our goal is to estimate the roadside space at a distance d ahead of the car. The distance should be sufficient to allow the car to perform a complete stop. Assume that the car can perform a complete stop after the travel distance in the K^{th} time step. Let the distance between the car center and the rightmost lane marking and the distance between the car center and the right road border be l_K^m and l_K^b respectively. The width of the roadside at time step K , l_K^w is given by:

$$l_K^w = l_K^b - l_K^m \quad (1)$$

As a result, the problem is equivalent to estimating x_k^o at time step $k \in 1, \dots, K$ via a sequence of given observations: $p(x_k^t | z_{1:k}^t)$, where $p(\cdot)$ is a probability density function, $x_k^t \equiv [l_k^b, l_k^m]^T$ is the state to estimate, and $z_{1:k} \equiv \{z_{1:k}^b, z_{1:k}^m\}$ is the observation of the road border and the lane marking from time step 1 to K .

System Configuration

Figure 3 shows our perception module configuration. The key idea is to fuse information from different sensors under one Bayesian estimation framework so that we can make the most of the strengths of each type of sensor installed in the vehicle. Instead of having a separated road profiling subsystem as we originally proposed, we switched to a 3D Lidar based quick profiling approach which requires fewer computation resources and thus allows robust real-time

performance. Our system can handle the strict requirement of estimating up to 80 m ahead when the vehicle is traveling at over 55 mph through such a fusion structure.

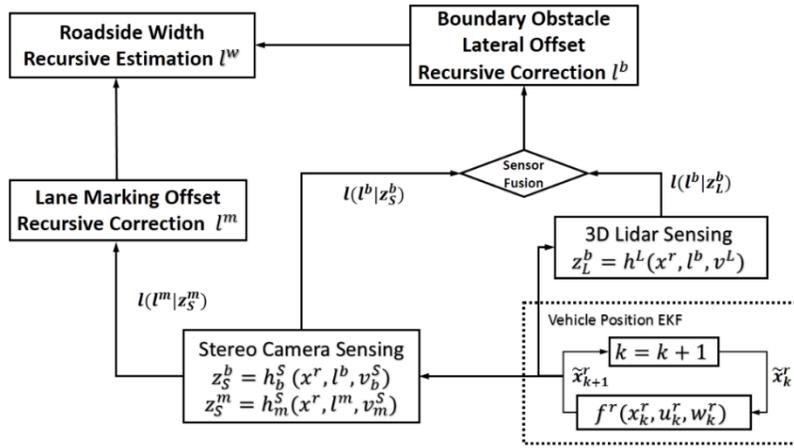


Figure 3. Roadside perception system configuration.

Specifically, the stereovision subsystem will be responsible for far-field (> 20 m) environment mapping, which gives us a preliminary roadside space estimation allowing feedforward prediction planning and control. The 3D Lidar and stereovision system combined will generate a detailed roadside profile in the near-field (< 20 m) for a vehicle to accurately park itself at the end of the maneuver. The major challenges here are to develop efficient image processing techniques for the stereovision subsystem to simultaneously detect boundary lane markings and road border obstacles. This places demanding requirements on enabling new reasoning capabilities beyond conventional stereovision algorithms; it is necessary not only to triangulate points, but also to determine if the points are above road surface and located on the right side of the lane demarcation. Our novel stereovision reasoning components help address this problem.

Roadside Detection and Tracking Using Stereovision

As we define the right boundary of the roadside to be an area where surfaces rise above the road height (such as areas with guard rails, retaining walls and utility poles, etc.), we proposed a novel stereo-vision based boundary detection technique to detect the right boundary. Then, combining this technique with an image-based lane marking detection method for left boundary detection, we can measure the roadside area with a pair of calibrated stereo cameras. The technique we propose first constructs a vertical road profile using a B-spline model, which is later used to generate a road surface depth map. Then, based on the depth map, boundary objects are detected and tracked in an occupancy grid. The technique overview is shown as a block diagram in Figure 4.

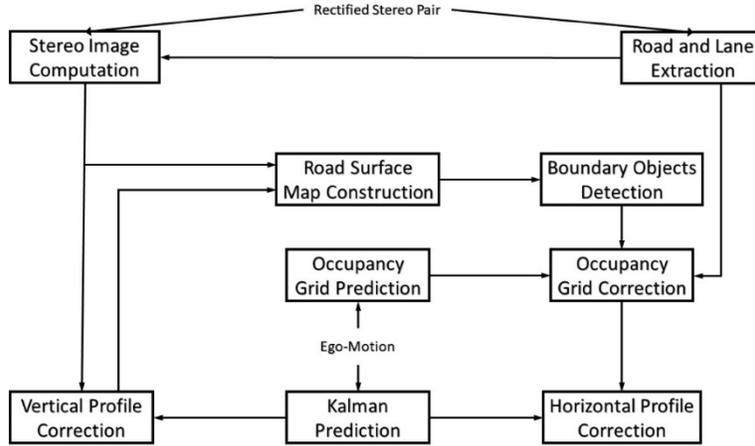


Figure 4. Stereovision based roadside area detection and tracking block diagram.

B-Spline Principles

B-spline curves are a general way of modeling 2D curves. Theoretically, B-spline curves can represent any polynomial functions using a set of predefined base functions and an adjustable control vector. The advantage of B-spline curves over other quadratic or cubic curves is their versatility for handling inflections and discontinuity. This quality is especially useful when we need to model a long distance ahead. A B-spline is defined as a linear combination of a set of base functions defined by a knot vector and the order of the curve. A knot vector $[t_1, t_2, \dots, t_{k+(n+1)}]$ defines the non-zero intervals of each base function at zero-order. Specifically, zero-order base function $N_{i,0}(t)$ is defined as:

$$N_{i,0}(t) = \begin{cases} 1, & t_i \leq t < t_{i+1} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Higher order base functions are calculated recursively from zero-order functions:

$$N_{i,k}(t) = \frac{t-t_i}{t_{i+k}-t_i} N_{i,k-1}(t) + \frac{t_{i+k+1}-t}{t_{i+k+1}-t_{i+1}} N_{i+1,k-1}(t) \quad (3)$$

The coefficients for the linear combination form the control vector are $[C_0, C_1, \dots, C_n]$, and the final curve function is given as:

$$B(t) = \sum_{i=0}^n N_{i,k}(t) C_i, t_{min} \leq t \leq t_{max} \quad (4)$$

Since we are dealing with an open loop spline, we also need to satisfy the following conditions: $t_1 = t_2 = \dots = t_d$, $t_d < t_{d+1} < \dots < t_{n+1}$ and $t_n + 1 = t_{n+2} = \dots = t_{n+d}$ where d is the order of the basis function.

3D Roadside Surface Modeling

From our definition of roadside, we know that we only need two 3D curves to fully define the roadside area. If we have a fixed knot vector, our B-spline functions will be fixed and can be pre-computed. All that remains to be estimated are the control points. However, estimation in 3D space boosts the state vector dimension and thus is unfavorable in terms of online performance. To minimize the state dimension, we decided to project the 3D curves onto two orthogonal planes and

represent the resulting 2D curves as a 1-dimension B-spline. This is based on the assumption that the road always extends forward and will not turn back. So, we can use the depth coordinate as the independent parameter t of the B-spline formulation and reduce the 2D control points to 1D control coefficients.

The coordinate system we used here has the origin at the middle of the front footprint of the ego vehicle, with the X -axis pointing toward the right of the vehicle, the Y -axis pointing upwards from the ground, and the Z -axis pointing forward. The two projection planes are the YZ plane for vertical profile projection and the XZ plane for horizontal profile projection. Here, we assume the two boundary curves share the same vertical profile; that is, their projection on the YZ plane is the same. This means we didn't consider the road shoulder's banking angle, which was negligible in most situations we evaluated. Then we can set the horizontal equation of the left boundary as:

$$X_l(z) = \sum_{i=0}^n N_{i,k}^l(z) C_{li}, z_{min} \leq z \leq z_{max} \quad (5)$$

The horizontal equation of the right boundary as:

$$X_r(z) = \sum_{i=0}^n N_{i,k}^r(z) C_{ri}, z_{min} \leq z \leq z_{max} \quad (6)$$

And the shared vertical profile equation as:

$$Y(z) = \sum_{i=0}^n N_{i,k}^y(z) C_{vi}, z_{min} \leq z \leq z_{max} \quad (7)$$

Note that each of these three curves can have a different B-spline function due to both inherent differences and measurement availability. For example, the right boundary allows larger and more irregular curvature than the left boundary and thus requires a higher polynomial degree. And because stereovision has more resolutions in the near field, the knots at the front need to be more closely spaced compared to those at the rear. We will discuss this in more detail in following sections.

Road Vertical Profile Estimation

The reason for first estimating the vertical curve $Y(z)$ is that we need to know the ground level of the road surface in order to calculate the height map, which is the height relative to the road surface. When the road ahead is flat, there is a technique called v-disparity, which can be used to efficiently calculate the road surface height. However, a flat road assumption doesn't always hold, especially when we want to estimate a long range ahead. Therefore, our solution is to fit a curve to the detected road surface points and use the fitted curve to generate the road height map.

Specifically, when we have M measurements from the stereo image matching, we can calculate the depth Z of these points from the disparity and then use a camera projection matrix to get their height Y . After that, we have m independent measurements $\{Y_m, Z_m\}_{m=1}^M$ with corresponding standard deviations σ_m .

With these measurements, the curve fitting problem becomes an optimization problem:

$$C^* = \min_c \left\{ \sum_m \frac{1}{\sigma_m^2} (Y(Z_m) - Y_m)^2 \right\} \quad (8)$$

After substituting equation (7) in (8), the problem reduces to finding the optimal control coefficients C_{vi}^* that minimize the weighted sum:

$$C^* = \min_c \left\{ \sum_m \frac{1}{\sigma_m^2} (\sum_{i=0}^n N_{i,k}^y(Z_m) C_{vi} - Y_m)^2 \right\} \quad (9)$$

This then becomes a least-square problem:

$$C^* = \min_c \left\{ \begin{bmatrix} \frac{1}{\sigma_1^2} N_Y(Z_1)^T \\ \vdots \\ \frac{1}{\sigma_M^2} N_Y(Z_M)^T \end{bmatrix} C - \begin{bmatrix} Y_1 \\ \vdots \\ Y_M \end{bmatrix} \right\} \quad (10)$$

where $N_Y(Z_m)^T$ is the vector $[N_{0,k}^y, N_{1,k}^y, \dots, N_{n,k}^y]$.

After the coefficients are estimated, we have a curve for the vertical coordinates of the road, which we can use to calculate a road surface depth map combining with the camera model.

Road Surface Depth Map Construction

Now we construct a depth map for the road surface similar to the familiar v-disparity map. That is, we want to know the disparity (depth) of the road projected on the v^{th} row of the image. In order to do this, we trace back a ray of light incident at the v^{th} row and calculate its intersection with the vertical B-spline profile. Let f be the camera focal length, α be the camera pitch angle, R be the image row number, p_v be the pixel height, H , and Δ_h be the camera height and height offset. We can see the incident coordinate at the image screen is:

$$\text{incident point} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} f \\ -\frac{R}{2} p_v + p_v v \end{bmatrix} + \begin{bmatrix} 0 \\ H + \Delta_h \end{bmatrix} \quad (11)$$

Since the camera center is located at $(0, H + \Delta_h)$, we can calculate the resulting line function of the light ray as:

$$Y = \frac{-f \sin\alpha + \cos\alpha (p_v v - \frac{R}{2} p_v)}{f \cos\alpha + \sin\alpha (p_v v - \frac{R}{2} p_v)} Z + H + \Delta_h \quad (12)$$

Because the pitch angle of the camera may influence the results, it is necessary to have a good estimate of the pitch, especially for accurate depth calculation in the far field. Danescu et al. [4] designed a method for pitch estimation based on a polar histogram. Their approach is based on two important assumptions:

1. The road points should be on a horizontal plane in the near field (typically ≤ 15 m)
2. Most of the points in the 3-D space are above the road surface.

We built our estimation method based on the same assumptions. However, unlike the methods used by Danescu et al. [4], we simultaneously estimated the camera pitch angle and height offset, since vehicle vibrations tend to result in both kinds of movement of the camera. We adopted a robust fitting method called Hough transformation and constructed a 2D Hough space for parameter estimation. Specifically, we set the domain for pitch angle from -2 degrees to 2 degrees

and the domain for height offset from -2 cm to 2 cm. Then we discretized the parameter space into bins of 0.1 degree and 0.1 cm for parameter voting. The algorithm for Hough transformation is outlined below:

First initialize all $H(\alpha, \Delta_h)$ to 0

For each triangulated 3D point (x, y, z) ,

If $z \geq 15m$, go to the next point;

For $\Delta_h = -2cm: 0.1cm: 2cm$

$\alpha = \tan^{-1}(y + \Delta_h)/z$

If $\alpha < -2^\circ$ or $\alpha > 2^\circ$

go to next point;

Else $H(\alpha, \Delta_h) = H(\alpha, \Delta_h) + 1$ (need to first find the index of corresponding bin)

End

End

After the transformation, we just need to find the values of (α, Δ_h) where $H(\alpha, \Delta_h)$ is a local maximum. For each image row v , we can calculate the depth Z_v of the road surface at that row as:

$$\frac{-f \sin \alpha + \cos \alpha (p_v v - \frac{R}{2} p_v)}{f \cos \alpha + \sin \alpha (p_v v - \frac{R}{2} p_v)} Z_v + H + \Delta_h = \sum_{i=0}^n N_{i,k}^y (Z_v) C_{vi}^* \quad (13)$$

After all Z_v values are obtained, the v-disparity map for the road surface follows easily as:

$$d(v) = \frac{B * f}{Z_v} \quad (14)$$

where B is the baseline length of the stereo rig.

Boundary Objects Detection and Tracking in Stochastic Occupancy Grids

As previously defined, the right roadside boundary is marked by objects rising above road surface. One typical feature of those objects is that they usually form a vertical surface perpendicular to the ground (e.g., buildings and billboard poles). This feature provides a powerful cue for detecting these objects using stereo-vision, since such vertical surfaces display the same disparity values in a column of the disparity image.

Our detection algorithm is as follows:

While $i < R$

If $d(i, u) > d_{max}$, $i = i + 1$;

Else $t = VAR[d(i, u), d(i + 1, u), \dots, d(i + N_1, u)]$,

If $t > \text{threshold}$

$i = i + N_2$;

Else find $N > N_1$ such that $N = \underset{N}{argmax}\{VAR[d(i, u), d(i + 1, u), \dots, d(N, u)] < \text{threshold}\}$

$d = MEAN[d(i, u), d(i + 1, u), \dots, d(N, u)]$;

If $N > R - v(d) + N_3$, $i = i + N_2$;

Else for $j = N$ to $R - v(d) + N_3$ count the number C of $d(j, u) < d$;

If $C > \text{threshold}$, $i = i + N + 1$;
Else $M(k)=(u,d)$, $k=k+1$;

End While

Here, R is the row number of the disparity image, $d(i, u)$ is the disparity of the pixel at the i th row and the u th column. The detection algorithm tries to find a vertical surface by continuously searching for segments of the same disparity in a column. The detection is initialized by observing a minimal number of consistent disparity values N_1 and extends to all following similar values. The road surface depth map built in *section B* serves as a strong constraint for rejecting possible outliers, since any visible vertical surface should be above road surface. A final check comes from the concerns of hanging objects. While our algorithm is robust to partial occlusions since we only need parts of the object surface as the detection evidence, there are cases when this assumption fails. For example, in urban environments, crossovers and billboards hanging in the air could be mistakenly detected as boundary objects. In order to address this problem, we enforce a check on the disparity values of the pixels between the lower end of a detected surface and the road surface. If the number of pixels with smaller disparity than the detected surface exceeds a threshold, we can deduce that this is a hanging surface.

After detection, we get measurement data stored in M where each entry (u, d) represents a possible object in a grid cell (u_{ij}, d_{ij}) . We constructed the grids in terms of image columns and disparity values because of the efficiency advantages provided by such configurations. Specifically, since we don't need to go through projection transformation when updating the grids, the correction implementation can be much faster. We also only need to update a quarter of the grid cells that conventional Cartesian grids use because of the symmetry about both axes. However, the cost of this is a significant drop of resolution in the far field. A trade-off has to be made between far-field detection accuracy and real-time computation performance.

Roadside Tracking and Smoothing using Kalman Filter

After the boundary measurements are obtained, the curve fitting process is exactly the same as that of vertical profile fitting discussed in section 2.4.3. However, curve coefficients estimated in this way are susceptible to noises or incorrect measurements and can change abruptly in continuous frames. Therefore, we embedded the curve fitting problem in a Kalman filter framework to track the curves across frames and filter out any outliers. In this sense, the Kalman filter correction step is a variation of equation (10) and the prediction is performed by taking vehicle odometry data and projecting the coefficients of old B-splines to new B-splines.

Results from Offline Analysis

In this project we were able to load a vehicle with all the sensing equipment mentioned above. See Appendix A for a detailed description of all the equipment installed and the wiring and configuration around the vehicle. Using the installed equipment, we were able to collect real data and test our algorithms in different real scenarios. Real data also comes with a lot of problems and corner cases where our detection algorithm may fail. To mitigate these issues, we leverage the

accuracy of Lidar points in the near field to prefilter the stereo image data and take several preprocessing steps before running the main program. Specifically, before using the 2D edge detector for point cloud intensities, we first perform a plane segmentation to cluster out all those points belonging to the road. This step proves to be essential for the success of subsequent detection and tracking. In order to segment out the points, we first use a bilateral filter to remove noise points and then use a sample consensus-based surface normal Euclidean cluster extraction method to obtain the road surface plane. We use the prior knowledge of road markings to search for the boundary of the road shoulder where no marking is detected. Here, the primary cue for boundary detection is road surface roughness, which we quantified using a histogram of surface normal. Basically, we first use an octree-based k nearest neighbor searching algorithm to determine a unique neighborhood for each point and then compute the surface curvature and normal of the points in that neighborhood. Next, variance of surface normal is calculated based on predetermined neighborhoods and a certain threshold is applied for unpaved area detection.

We validated our approach using the vehicle equipped with stereo cameras and 3D Lidar. The test was performed in highway scenarios. The vehicle began traveling in the right lane at 45 mph (20 m/s), then began to decelerate after identifying a roadside stopping place 80 meters away. After decelerating, at 30 m away from the identified stopping place, the vehicle turned right into the emergency lane. Here, Figure 5 shows both lane offsets and the road boundary in the image domain are detected and tracked while only the road boundary is detected in the Lidar point cloud. The two sources of information were fused in the Extended Kalman filter (EKF) correction phase to enhance accuracy. Figure 6 shows reduced variance of the roadside width estimation as the vehicle slows down to the roadside location.

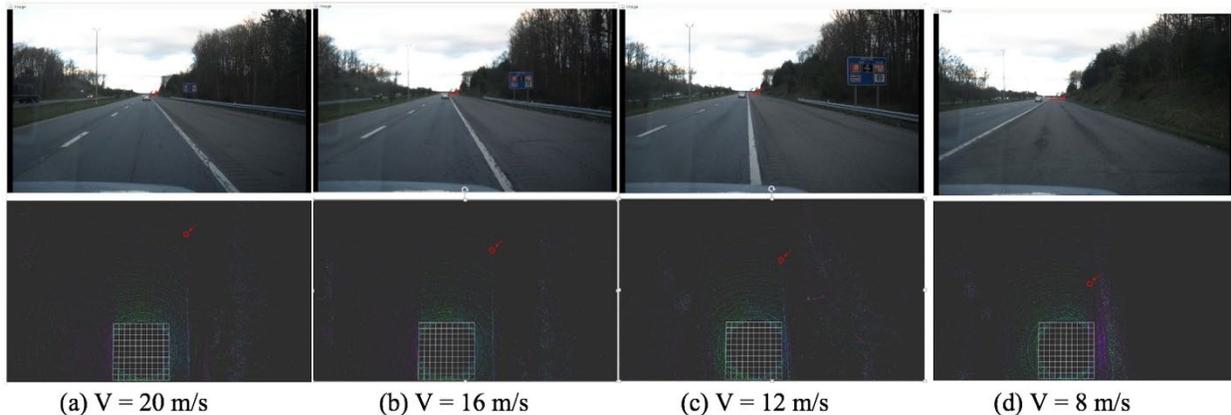


Figure 5. Detected boundary points and roadside width as the vehicle slows down.

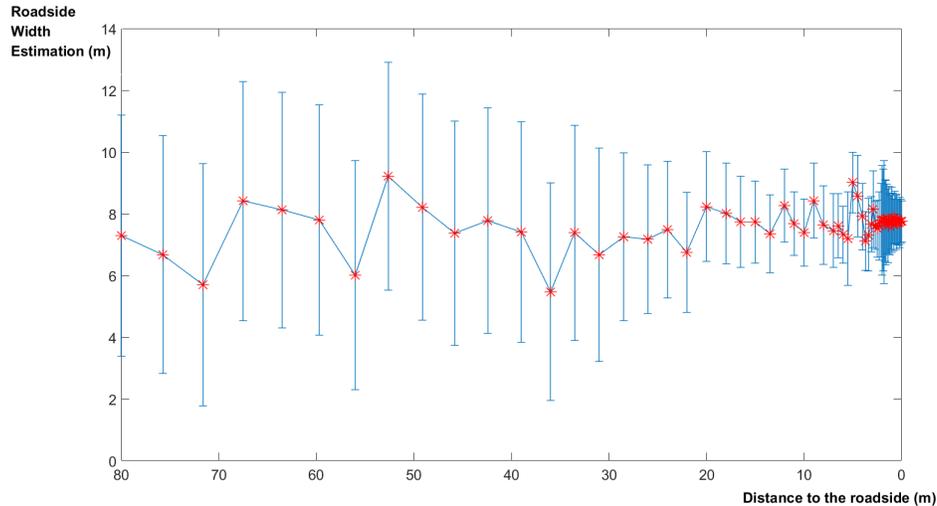


Figure 6. The estimated mean roadside width and its 3σ errors as the vehicle slows down.

3. Trajectory Planning and Control Solutions

While there have been limited focused efforts on developing planning and control solutions for autonomous emergency navigation, many other works have been performed dealing with vehicle control for automated overtaking and collision avoidance. These works deal with similar technical challenges as ours, such as simultaneous lateral and longitudinal control, prediction control with uncertain destinations, and fast computation for real-time implementation. Among these works there are two main strategies to address these challenges. One is to break the task into two subtasks of trajectory generation and trajectory tracking. For example, Petrov et al. [11] designed an adaptive nonlinear controller to track polynomial trajectories generated in real time for overtaking maneuver. In [12], He. et al developed a robust lateral motion controller to stably track a collision-free path during an emergency collision avoidance maneuver. Such strategies are usually computationally efficient since there exist many fast trajectory generation methods, like that proposed by Werling et al. [13]. This two-layer design can also draw on the abundant existing works on nonlinear feedback controllers. However, the main drawback of this strategy in real application centers on the difficulties encountered during feedback calculation. Specifically, the feedback controller needs to continuously measure the difference between current state and reference state, the so-called tracking error. This requires the vehicle to possess a high-precision self-localization and mapping capability, which is hard to achieve, especially in high-speed scenarios and in scenarios where landmark references are scarce. Our targeted scenario, an emergency stop on a highway, will involve both high-speed movement and scarce landmarks in the surrounding environment, thus making this strategy undesirable.

Another strategy is to leverage model predictive control (MPC) and cast the control problem into an optimization problem over a look-ahead time horizon. In this way, the prediction requirements are handled in a model-based feedforward manner without the need to generate predictive

trajectories. For instance, in [14], Dixit et al. propose an autonomous overtaking framework in highway speed using tube-based robust MPC as the control commands generation method. Sotoudeh et al. [15] use a similar method to ensure collision avoidance in uncertain environments. In [16] Liu et al. use MPC to simultaneously optimize speed and steering angle for obstacle avoidance maneuver in high speed while considering the handling limits of vehicle dynamics.

The optimization algorithms used in these works, however, have limitations in the length of planning horizon, so intermediary references are needed. For example in [14], the controller will generate reference points as intermediary destination states along the overtaking maneuver by measuring the states of the overtaken vehicle. For automated emergency stops, such references don't exist. We only have a rough estimation of the final location our vehicle may pull into, which is usually far from our current position and may be subject to further changes due to insufficient roadside space. Therefore, we cannot use the standard collocation-style MPC algorithms for our system.

To address the unique requirements of the automated emergency stop system, we propose a novel control system design based on a modified differential dynamic programming (DDP) technique that is shown to be able to control the vehicle to stop at a safe roadside location as determined on the fly. The backbone of the computation leverages the DDP method, which allows our system to plan over a much longer horizon compared to standard MPC techniques, thus solving the challenge of not having references along the way.

Control Problem Statement

In this section we state the mathematical formulation of the emergency pull over motion control problem and present the essential assumptions we adopted for addressing this problem. Figure 7 shows the automated roadside emergency stop problem addressed in this report where the vehicle is driven on the righthand side as required in the United States, and thus focuses on the navigable roadside on the right. As the lane change is not the area of concern within this paper, we assume that the vehicle has already changed to the rightmost lane of the road prior to activation of the pull over logic. Since the geolocation of the vehicle is irrelevant to the automated control for emergency stop, the coordinate frame to be used for the motion control is chosen along the right lane marking, which is shown in blue in Figure 7.

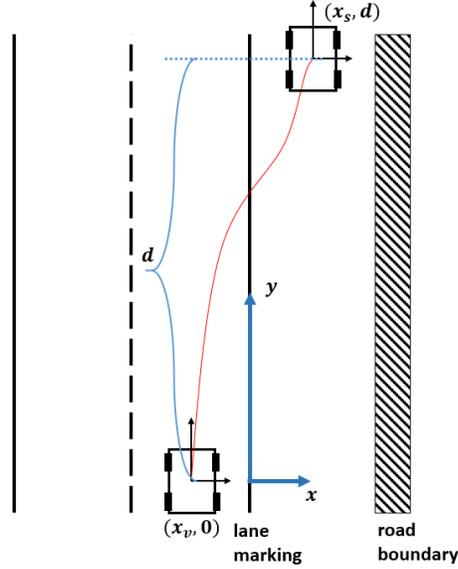


Figure 7. Illustration of emergency stop navigation control problem

The goal of the emergency stop problem is to find a set of control sequences that can drive the vehicle from the current location $(x_v, 0)$ to a complete stop at the identified roadside position (x_s, d) with the vehicle heading aligned with the road direction, which is the y axis in this case. We formulate this problem as a discrete optimal control problem. The vehicle state at step i is represented as \mathbf{x}_i . The control action of vehicle at step i is represented as \mathbf{u}_i , which comprises a physical control term and a time increment term to allow for flexibility of control duration and thus total time to stop. The state transition dynamics at each step is represented by $\mathbf{f}_i(\mathbf{x}_i, \mathbf{u}_i)$, which is assumed to be continuously differentiable in its arguments. The full problem formulation is shown as (15):

$$\begin{aligned}
 & \min_{(\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{K-1})} \quad \sum_{i=0}^{K-1} l_i(\mathbf{x}_i, \mathbf{u}_i) + l_K(\mathbf{x}_K) \\
 & \text{s. t. :} \quad \mathbf{x}_{i+1} = \mathbf{f}_i(\mathbf{x}_i, \mathbf{u}_i), \quad i = 0, 1, \dots, K-1 \\
 & \quad \quad \mathbf{x}_0 = \mathbf{x}_{init} \\
 & \quad \quad \mathbf{x}_i \in \mathcal{X}_i \\
 & \quad \quad \mathbf{u}_i \in \mathcal{U}_i
 \end{aligned} \tag{15}$$

Here l_i and l_K are integral and terminal objectives, respectively, and both are continuously differentiable in their arguments. K is the step at which the vehicle should stop. The initial state of vehicle \mathbf{x}_{init} is given as the state when the emergency stop maneuver starts. All other states are determined by the control sequence $(\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{K-1})$. \mathcal{X}_i and \mathcal{U}_i are the constraints space for vehicle state and control action at step i . Note that the terminal constraint is introduced in the terminal objective l_K .

Emergency Stop Autonomous Control System

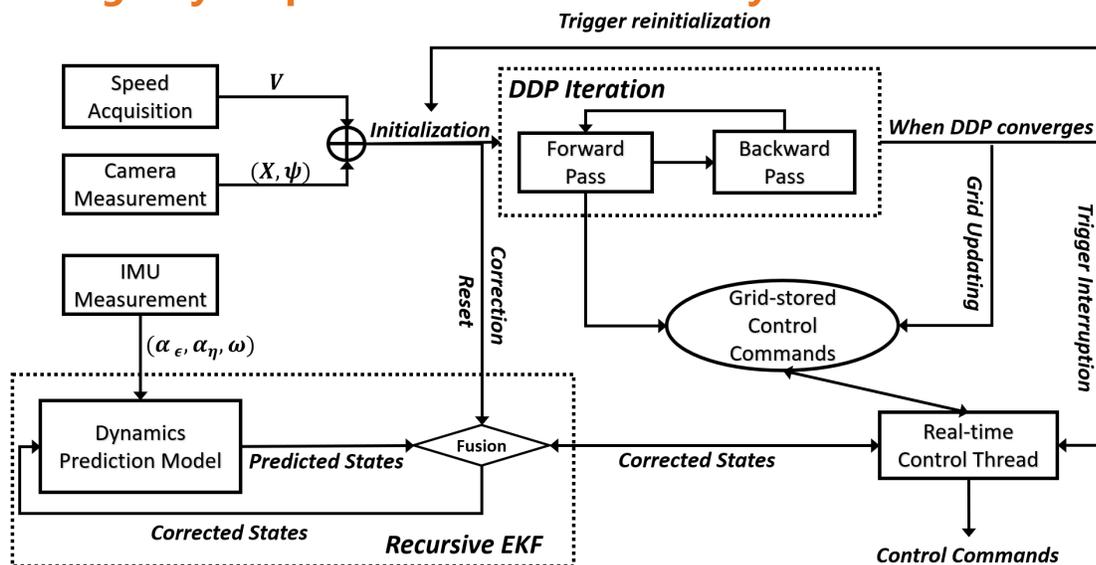


Figure 8. Overall control system diagram.

In this section, we present our control system implementation based on the DDP method. Figure 8 shows the overall system architecture. There are two main computation modules: the *DDP Iteration* module and the *Recursive EKF* module. The *Recursive EKF* module is the high frequency vehicle self-localization module based on inertial measurement unit (IMU) and EKF methods. Another important module is the storage data structure of grid-discretized state space for hosting control commands to be taken at each state. During a DDP forward pass, this storage structure updates itself based on new control sequences computed. The real-time control thread will first obtain information about current states from the EKF module and then send a request to the storage structure to fetch the control commands at its interrupt time.

3.2.1 DDP Iteration

The reason we are maintaining a storage data structure to host control commands is because we cannot ensure real-time DDP convergence. In other words, the time it takes for one DDP iteration to converge is unpredictable and thus we cannot guarantee the control thread always gets the newly computed control commands when the interrupt happens. When the iteration hasn't returned, the control thread will have to reuse previous control commands. However, due to modeling errors, current estimated states may not be the same as the states predicted from the control horizon. In fact, current states may not equal any states in the trajectory computed by the previous DDP iteration. One possible solution is to calculate its nearest neighbor in the state space, but that may lead to unfavorable or even unsafe motion when the control trajectory is rapidly changing. Therefore, we instead leverage all forward pass trajectories that have a cost below a threshold. During DDP iteration, usually the cost drops quickly and stays within an acceptable range before converging to the optimal. In an original DDP implementation, those trajectories before converging are discarded (or overwritten by new trajectories). In our implementation, we record

those trajectories, as they represent feasible solutions and cover a wider state space. We accomplish this recording by discretizing the state space into grids. When a new trajectory is confirmed to be acceptable, states along the trajectory are matched to the corresponding grid cells and update the control values stored in those cells. In this way we can provide back-up control for any states the vehicle is currently in.

3.2.2 Recursive EKF

During the emergency stop maneuver, the vehicle constantly estimates its current states based on the EKF recursive fusion scheme. The EKF is a kind of Bayesian estimation method that recursively predicts a system's state based on its dynamics and corrects the predicted states based on sensor measurements. It assumes Gaussian noises so that stochastic state estimation can be represented by their mean and covariance. Specifically, we assume the system dynamics has the form:

$$\mathbf{x}_i = \mathbf{f}(\mathbf{x}_{i-1}, \mathbf{u}_{i-1}) + \mathbf{w}_{i-1} \quad (16)$$

where $\mathbf{w}_{i-1} \sim \mathbf{N}(\bar{\mathbf{w}}_{i-1}, \Sigma_{\mathbf{w}_{i-1}})$ is Gaussian dynamics noise. And sensor measurement is assumed to have the form:

$$\mathbf{z}_i = \mathbf{h}(\mathbf{x}_i) + \mathbf{v}_i \quad (17)$$

where $\mathbf{v}_i \sim \mathbf{N}(\bar{\mathbf{v}}_i, \Sigma_{\mathbf{v}_i})$ is Gaussian measurement noise. The estimated states $\mathbf{x}_{i|i} \sim \mathbf{N}(\bar{\mathbf{x}}_{i|i}, \Sigma_{\mathbf{x}_{i|i}})$ are fully represented by its mean $\bar{\mathbf{x}}_{i|i}$ and covariance $\Sigma_{\mathbf{x}_{i|i}}$, which are updated recursively as:

$$\begin{aligned} \bar{\mathbf{x}}_{i|i-1} &= \mathbf{f}(\bar{\mathbf{x}}_{i-1|i-1}, \mathbf{u}_{i-1}) \\ \Sigma_{\mathbf{x}_{i|i-1}} &= \mathbf{A}_{i-1} \Sigma_{\mathbf{x}_{i-1|i-1}} \mathbf{A}_{i-1}^T + \Sigma_{\mathbf{w}_{i-1}} \end{aligned} \quad (18)$$

$$\text{where: } \mathbf{A}_{i-1} = \left. \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right|_{\mathbf{x}=\bar{\mathbf{x}}_{i-1|i-1}}$$

$$\begin{aligned} \bar{\mathbf{x}}_{i|i} &= \bar{\mathbf{x}}_{i|i-1} + \mathbf{K}_i (\mathbf{z}_i - \mathbf{h}(\bar{\mathbf{x}}_{i|i-1})) \\ \Sigma_{\mathbf{x}_{i|i}} &= (\mathbf{I} - \mathbf{K}_i \mathbf{C}_i) \Sigma_{\mathbf{x}_{i|i-1}} \end{aligned} \quad (19)$$

$$\begin{aligned} \text{where: } \mathbf{C}_i &= \left. \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \right|_{\mathbf{x}=\bar{\mathbf{x}}_{i|i-1}} \\ \mathbf{K}_i &= \Sigma_{\mathbf{x}_{i|i-1}} \mathbf{C}_i^T (\mathbf{C}_i \Sigma_{\mathbf{x}_{i|i-1}} \mathbf{C}_i^T + \Sigma_{\mathbf{v}_i})^{-1} \end{aligned}$$

Here, (18) is often called the EKF prediction step, which requires a dynamics prediction model \mathbf{f} . We use a different model than the one used in our DDP control algorithm to fully take advantage of the high-frequency measurement capability provided by the IMU installed on the vehicle. The model is based on a 3 degree of freedom bicycle kinematics model but replaces control terms with actual measurements, as below:

$$\begin{aligned} \dot{\mathbf{v}}_\epsilon &= \alpha_\epsilon - \mathbf{v}_\eta \omega \\ \dot{\mathbf{v}}_\eta &= \alpha_\eta - \mathbf{v}_\epsilon \omega \\ \dot{\mathbf{x}} &= -\mathbf{v}_\epsilon \sin \theta + \mathbf{v}_\eta \cos \theta \\ \dot{\mathbf{y}} &= \mathbf{v}_\epsilon \cos \theta + \mathbf{v}_\eta \sin \theta \\ \dot{\theta} &= \omega \end{aligned} \quad (20)$$

where $\mathbf{v}_\epsilon, \mathbf{v}_\eta$ are the vehicle's lateral and longitudinal velocity, as shown in Figure 9; \mathbf{x}, \mathbf{y} and θ are the vehicle's coordinates in an inertial frame set at each reinitialization point ; α_ϵ and α_η are the acceleration along the lateral and longitudinal direction of the vehicle body; and ω is the yaw rate of the vehicle body. $\alpha_\epsilon, \alpha_\eta$ and ω are all measured by the on-board IMU at 100 Hz.

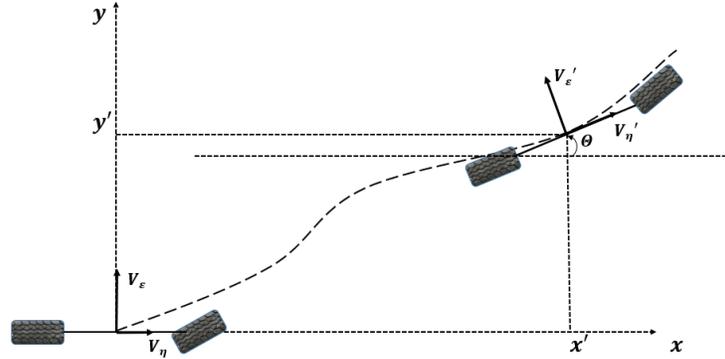


Figure 9. Model for EKF estimation.

This ensures high-resolution localization, which is crucial at high speed. The correction step (19) is to fuse sensor observations to correct errors in model-based prediction. We use a speed sensor and cameras to correct accumulated dead-reckoning errors during the prediction process. We assume cameras provide fairly accurate measurements of a vehicle's position relative to the lane marking, and thus can give us the (X, ψ) coordinates. The measurement model \mathbf{h} takes the form:

$$\begin{aligned} V &= \sqrt{\mathbf{v}_\epsilon^2 + \mathbf{v}_\eta^2} \\ X &= X' + \mathbf{x} \\ \psi &= \psi' + \theta \end{aligned} \quad (21)$$

where X' and ψ' are vehicle's coordinates at the last reinitialization point. Due to the camera frame rate limit and the CAN bus sampling rate limit, the correction frequency is 20 Hz. Therefore, as prediction and correction are asynchronous in our case, we will perform 5 steps prediction (18) before one step correction (19).

3.2.3 Receding Horizon Reinitialization

Our system works in a receding horizon fashion, meaning that when each DDP iteration returns, we will immediately begin the next round of iteration using current states as the initial state. This is similar to what is adopted in most MPC frameworks, essentially integrating feedback by constantly reinitializing the feedforward optimization process with the most recent observation. Another benefit of this scheme comes from the uncertainty associated with terminal state estimation. It's difficult for the vehicle to estimate a roadside location available for a pull over at a far distance, and naturally the estimation improves as the vehicle gets closer. So, terminal state estimation may change and continuous reinitialization can help incorporate those changes in the cost function in a timely manner. In addition, continuous DDP iteration can contribute to filling the grid storage with control commands, which can enhance the robustness of the system by

providing back-up control commands whatever the vehicle's current state. The reinitialization will also affect several other subsystems, and the execution sequence is important. Here we summarize the sequence used in our implementation:

1. **Update Grid Coordinates:** When one iteration converges and returns, we first update the coordinate frame of the discretized storage grid by requesting current state information from the EKF module.
2. **Reset EKF states:** Then we reset the EKF states to match the current DDP initial states.
3. **Interrupt Control Thread:** Finally, we need to interrupt the real-time control thread to inform it of the updated control commands' availability. The interruption is necessary because we use flexible time steps in our control formulation, and one control command can last longer than one DDP iteration.

SIMULATION RESULTS

To validate the efficacy of the proposed system before implementing it in a real vehicle, we performed tests in a simulation environment established in MATLAB/Simulink. The simulation setup includes four sub-modules: (1) a highly realistic dynamics simulator to serve as the ground truth generator, (2) a controller running DDP iterations using the simplified bicycle model, (3) an EKF estimator running at 100 HZ using simulated IMU signals, and (4) a grid-based control commands container. Since, in reality, the noises associated with camera and speed measurements are typically small, and their measurements are not for integration, we deemed those noises nonessential and assumed those measurements were accurate in our simulation. The highly realistic simulator ran at 100 HZ, so its output could be fed into the IMU simulator to generate simulated IMU signals. Camera measurements and vehicle speed acquisition were obtained directly from the highly realistic simulator at 20 HZ. The simulated scenario was a mid-size vehicle running at 20 m/s (45 mph) in the rightmost lane, and performing an emergency stop on the roadside ahead.

To evaluate our system's efficacy and its robustness to real-time computation lag and model errors, we tested two cases. In the first case, the roadside space was always sufficient for the vehicle to pull in, so there would be no identification failures. In the second case, there was a segment of road ahead with a narrow roadside space, but the vehicle could only determine that the roadside was narrow when it drove within 50 meters of that spot, in which case it had to re-identify another location ahead of the original and use the new location for DDP iteration. For the first case, we also evaluated how the system performs in the presence of DDP iteration lags. We carried out one simulation with the DDP iteration set to return immediately, and another simulation with a 1-second lag, which is a reasonable assumption of real-world worst-case execution time. The resulting vehicle state and control trajectories for each of the cases are shown in Figure 10 to Figure 12. The plotted vehicle states are ground truth states drawn from the highly realistic simulator. Note that due to the large range of the longitudinal position, we scaled all other state components' values by a factor of five to make them legible in the same plot. From the results, we can see the control system works well in all cases in terms of bringing the vehicle to a stop at a safe roadside

location. By comparing Figure 10 and Figure 11, we can see the DDP iteration lag doesn't negatively affect the results. As a matter of fact, the delay in DDP iteration allows control to be smoother because there is no need to change frequently to the new optimal policy. This smooth effect is deemed to be favorable, as it makes the vehicle's yaw motion more stable as shown in the yaw angle plots. The only negative effects of computation lags are an increase in the time it takes to fully stop and an overshoot of the steering action toward the time series end. None of these negative effects are significant and will not affect the system's overall performance.

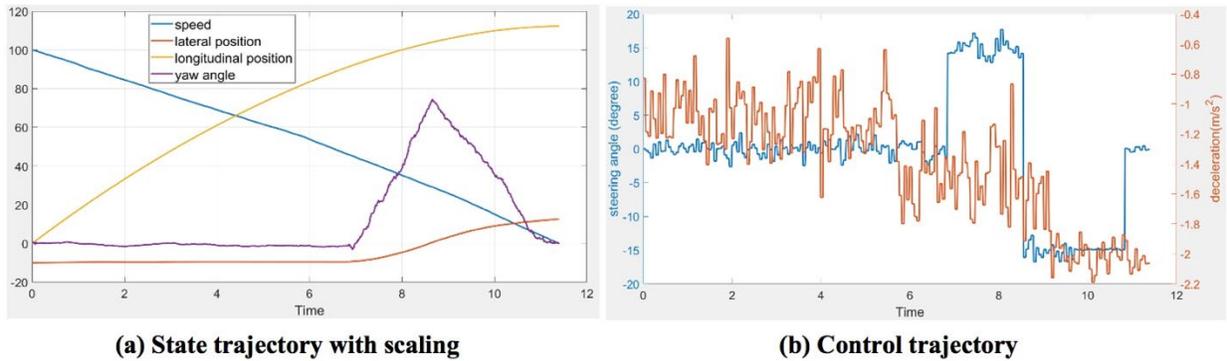


Figure 10. Trajectories with no lag.

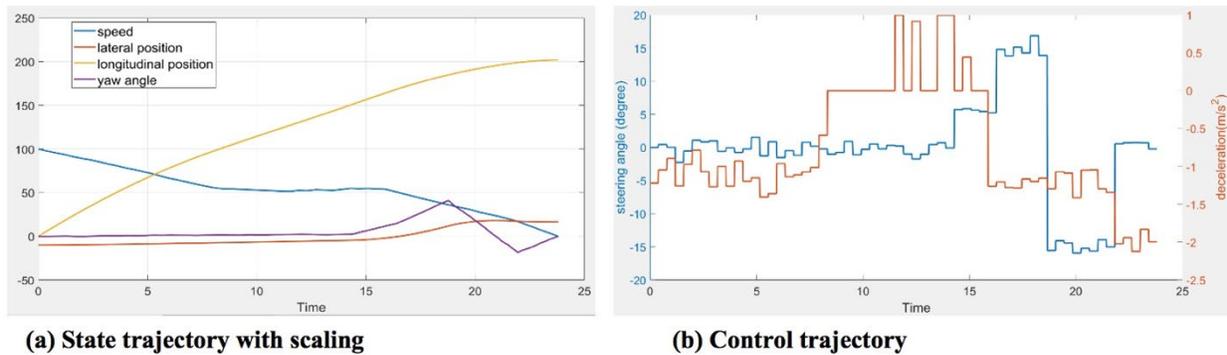


Figure 11. Trajectories with 1-second lag.

In Figure 12 we show the case with narrow roadside, which shows that our system can react properly when it detects a narrow roadside space and the vehicle can maintain speed with occasional acceleration when planning for another location ahead.

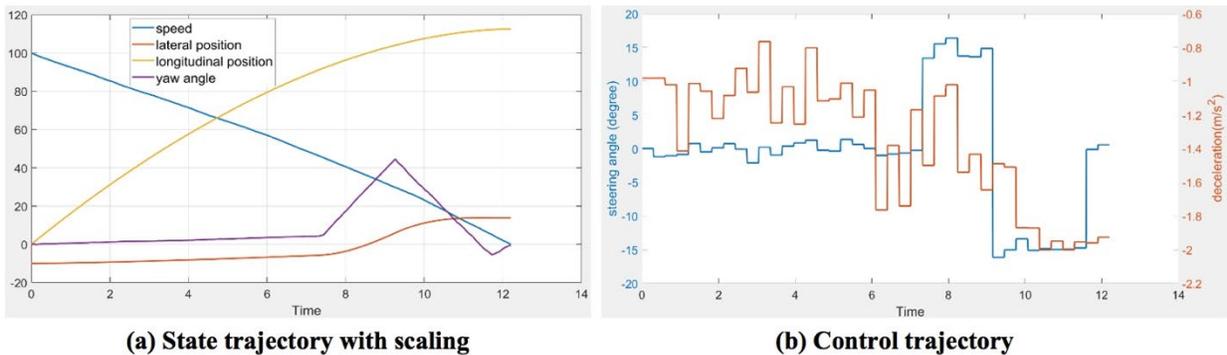


Figure 12 Trajectories in cases of narrow roadside segment.

4. Conclusions and Recommendations

This project aims to build a prototype system for an emergency autonomous navigation function that, once triggered, can autonomously navigate the automated vehicle out of the driving lane to pull off and park at a safe place by the roadside. We investigated and tested state-of-art robotics and control techniques, evaluated different solutions in real tests and a simulated environment, then developed algorithms and system architecture that can robustly achieve the proposed function.

For the first main task of the project, the roadside environment perception and mapping, we first proposed using SLAM as the major technique. We installed a 64 channel 3D Lidar and tested several state-of-the-art point cloud processing algorithms for fast voxelization and landmark registration. Our attempts showed that with 64 channel Lidar running at a speed over 20 m/s, the point cloud density was not enough to identify and track landmarks beyond 25 m. The effective tracking area was reduced to under 15 meters if the vehicle was running on the highway, where the environment doesn't have enough distinguishable landmarks and the algorithm can easily confuse similar-looking point clusters like trees and rails along the side of the road. Therefore, we decided to instead use stereovision and image processing to help make reasonable estimations for navigable roadside areas in the far field. The technique was built on classic lane and boundary segmentation solutions, with the added robustness brought by stereovision-based 3D road profile construction. Field tests showed our methods performed well in normal urban and highway scenarios, but in environments where there were not many vertical objects by the roadside, or the vertical objects were not easy to triangulate, the system may fail to find an acceptable parking place for miles. Once a place was identified and the stop maneuver started, the estimation continued to be refined by new measurements, which tended to be more accurate as the vehicle drove closer. And when the vehicle's speed dropped below 10 m/s, the Lidar SLAM began to work, which combined with stereovision, allowed construction of a detailed roadside map for final parking location selection. This means that even if our far-field perception module fails, we can choose to let the vehicle slow down and then leverage sensor fusion for parking place identification.

For the planning and control task, we originally intended to use a sequential path planning and feedback control technique. However, due to the incapability of the perception module to construct an accurate enough map for path tracking references, we instead used a DDP-based prediction control technique that can directly optimize trajectories towards the parking location. This technique employs an efficient second order shooting method which iteratively adjusts control trajectory by solving a series of small-scale quadratic programming problems. This method is innately suitable for "hot start," meaning once there is an initial solution, subsequent computation for changed problem conditions will be significantly accelerated. This feature was critical since we needed to frequently reinitialize the problem conditions due to model inaccuracy. Our introduction of grid-based control commands storage structure further increased

the robustness of real-time control. Simulation results showed good performance in typical straight road scenarios. The reinitialization mechanism worked well in situations where the originally-identified roadside location turned out to be too narrow to pull in. In addition, by comparing cases with and without control computation time delay, we found our system maintained good performance in the worst case delay scenario of as long as 1 second.

For future work, we recommend creating roadside maps and integrating map information for more robust perception. We also plan to implement the control system on a real vehicle for future field tests. Our retrofitting of the drive-by-wire system has been ongoing and the progress to date is reported in Appendix B.

5. Additional Products

Education and Workforce Development Products

During the project, the Principal Investigators engaged any students in their courses who were interested in the topic to participate in regular weekly discussion. Some technical challenges have been made as course projects and students are encouraged to modify software packages to test their ideas. Dr. Furukawa also supervised the Self-Driving Vehicle Team (SDVT) senior design team. Some undergraduate students in the team first gained experience by retrofitting a golf car and then contributed to the retrofitting of the real vehicle for future field tests. The skills developed in the project have helped these students become competitive in the job market for traditional vehicle manufacturers as well as emerging autonomous driving startups.

Undergraduate students Jeronimo Cox and Cameron Miksch accumulated substantial experience in vehicle hardware programming during the project and successfully began their career at NASA thereafter. Undergraduate student Zhengdao gained experience in stereovision algorithms and was admitted into WPI's graduate program. Graduate student Lisheng investigated the vehicle control problem in addition to working on the hardware and made steady advances towards his degree. In addition to presenting this material to enrolled students, there are also plans to disseminate this project's material to K-12 students during summer science camps, etc.

Technology Transfer Products

During the 5th International Symposium on Future Active Safety Technology Toward zero traffic accidents (FAST-zero) in 2019 in Blacksburg VA, we presented the perception part of our work to both academic and industrial participants. Some company employees at the conference have shown interest in our technology, including Honda R&D. We will continue reaching out after we further increase the robustness of the system by integrating maps.

We also summarized our planning and control work in a paper to be published at the ASME Dynamic Systems and Control Conference in 2020. We hope that presenting at these leading

conferences will expose our work more and attract external funding to help us improve algorithms and do field tests.

Data Products

A dataset was not provided since the project team's algorithm was tested in simulation environments.

References

1. Hu, Y. and T. Furukawa, *A high-resolution surface image capture and mapping system for public roads*. SAE International Journal of Passenger Cars-Electronic and Electrical Systems, 2017. **10**(2017-01-0082): p. 301-309.
2. Wang, Y., E.K. Teoh, and D. Shen, *Lane detection and tracking using B-Snake*. Image and Vision computing, 2004. **22**(4): p. 269-280.
3. Nedeveschi, S., et al. *3D lane detection system based on stereovision*. in *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No. 04TH8749)*. 2004. IEEE.
4. Danescu, R. and S. Nedeveschi, *Probabilistic lane tracking in difficult road scenarios using stereovision*. IEEE Transactions on Intelligent Transportation Systems, 2009. **10**(2): p. 272-282.
5. Kubota, S., T. Nakano, and Y. Okamoto. *A global optimization algorithm for real-time on-board stereo obstacle detection systems*. in *2007 IEEE Intelligent Vehicles Symposium*. 2007. IEEE.
6. Labayrade, R., D. Aubert, and J.-P. Tarel. *Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation*. in *Intelligent Vehicle Symposium, 2002. IEEE*. 2002. IEEE.
7. Badino, H., U. Franke, and R. Mester. *Free space computation using stochastic occupancy grids and dynamic programming*. in *Workshop on Dynamical Vision, ICCV, Rio de Janeiro, Brazil*. 2007. Citeseer.
8. Wedel, A., et al., *B-spline modeling of road surfaces with an application to free-space estimation*. IEEE Transactions on Intelligent Transportation Systems, 2009. **10**(4): p. 572-583.
9. Li, Q., et al., *A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios*. IEEE Transactions on Vehicular Technology, 2013. **63**(2): p. 540-555.
10. Zhang, Y., et al., *When dijkstra meets vanishing point: a stereo vision approach for road detection*. IEEE transactions on image processing, 2018. **27**(5): p. 2176-2188.
11. Petrov, P. and F. Nashashibi, *Modeling and nonlinear adaptive control for autonomous vehicle overtaking*. IEEE Transactions on Intelligent Transportation Systems, 2014. **15**(4): p. 1643-1656.
12. He, X., et al., *Emergency steering control of autonomous vehicle for collision avoidance and stabilisation*. Vehicle system dynamics, 2019. **57**(8): p. 1163-1187.
13. Werling, M., et al. *Optimal trajectory generation for dynamic street scenarios in a frenet frame*. in *2010 IEEE International Conference on Robotics and Automation*. 2010. IEEE.
14. Dixit, S., et al., *Trajectory planning for autonomous high-speed overtaking in structured environments using robust MPC*. IEEE Transactions on Intelligent Transportation Systems, 2019. **21**(6): p. 2310-2323.
15. Sotoudeh, S.M. and B. HomChaudhuri. *Ensured Collision Avoidance Over a Finite Time Horizon for Autonomous Vehicles in Presence of Uncertainty*. in *Dynamic Systems and Control Conference*. 2019. American Society of Mechanical Engineers.

16. Liu, J., et al., *Combined speed and steering control in high-speed autonomous ground vehicles for obstacle avoidance using model predictive control*. IEEE Transactions on Vehicular Technology, 2017. **66**(10): p. 8746-8763.

Appendices

Appendix A: Perception Hardware Configuration

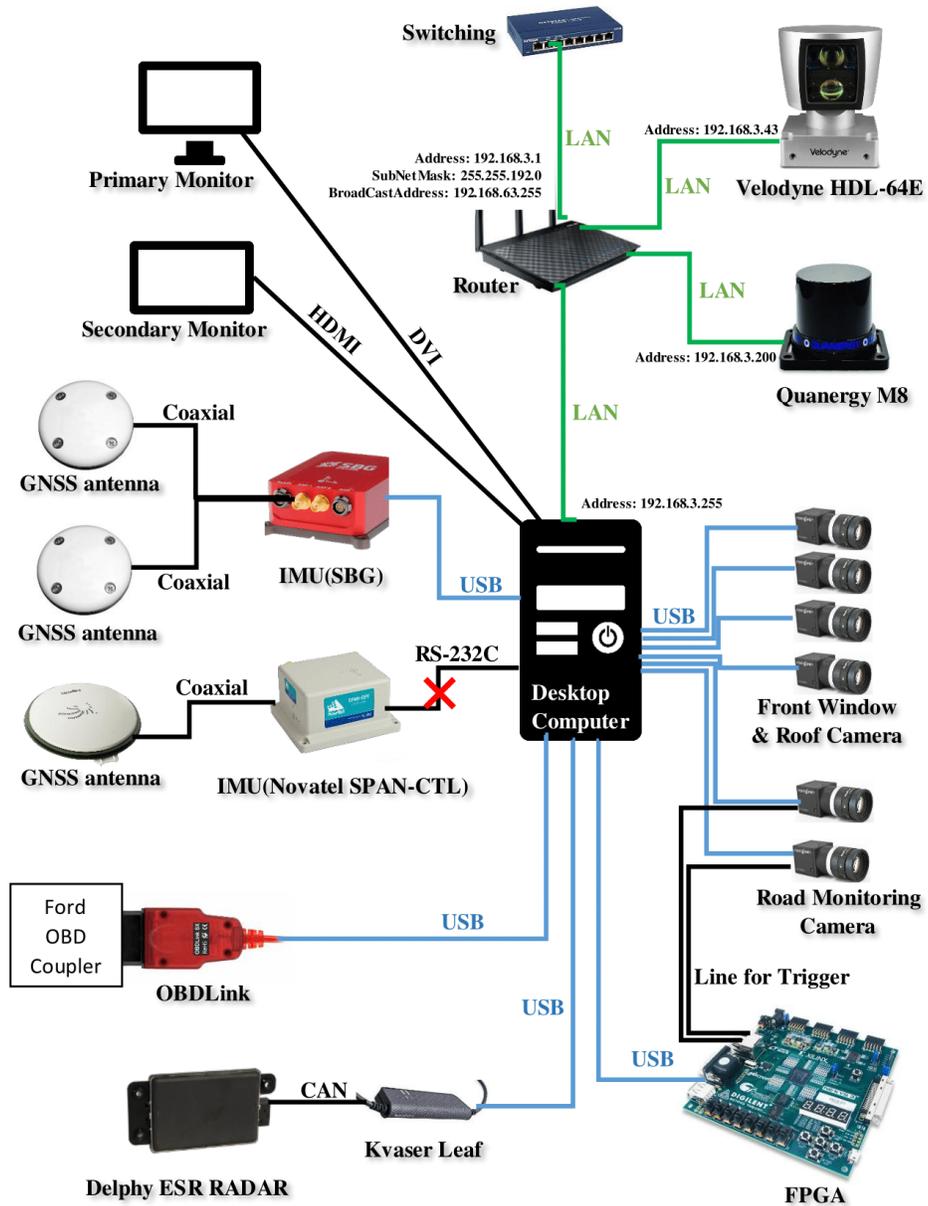


Figure 13. Vehicle sensor hardware wiring configuration

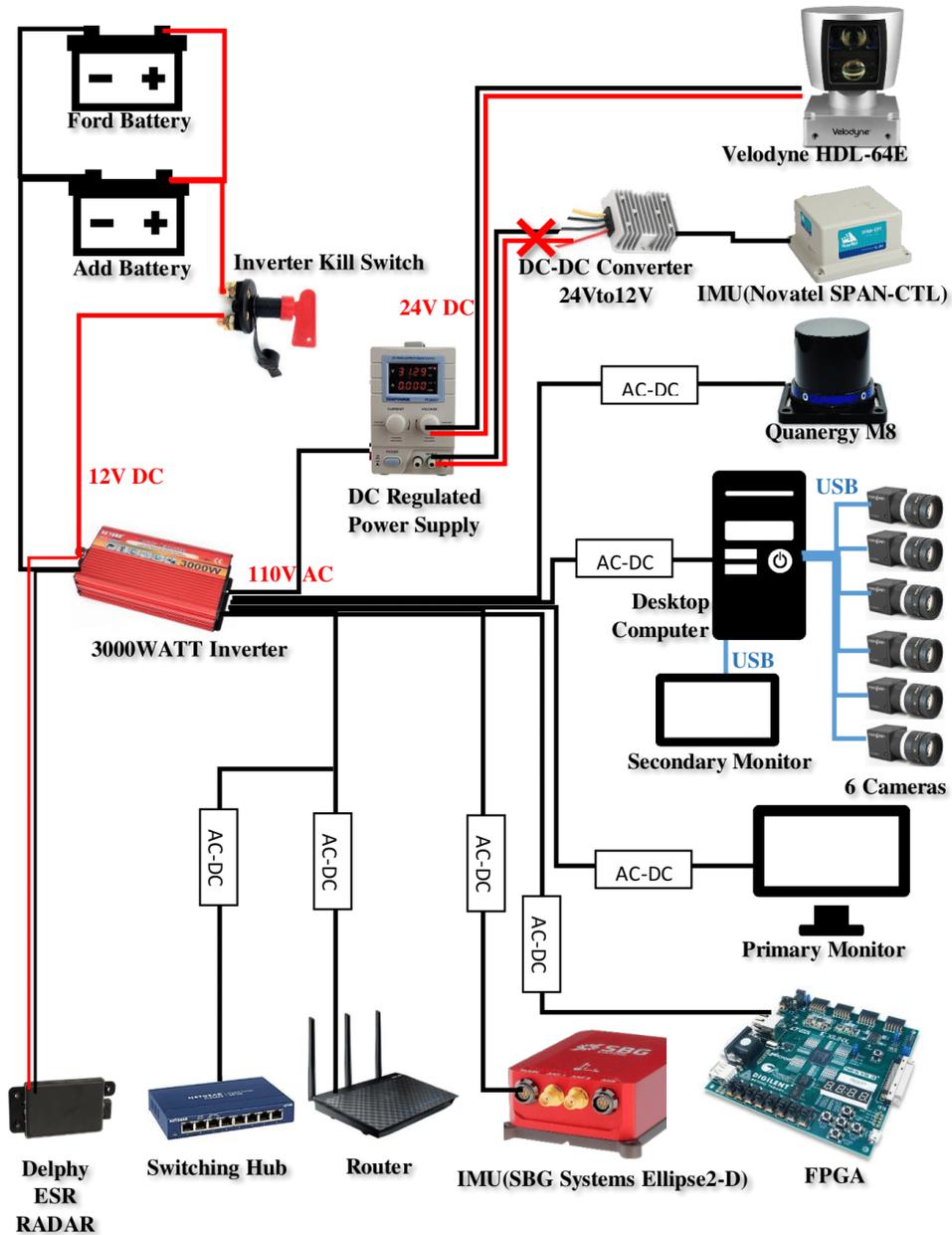


Figure 14. Power supply configuration of onboard computer and sensors.



Figure 15. Equipment installation picture of the test vehicle.

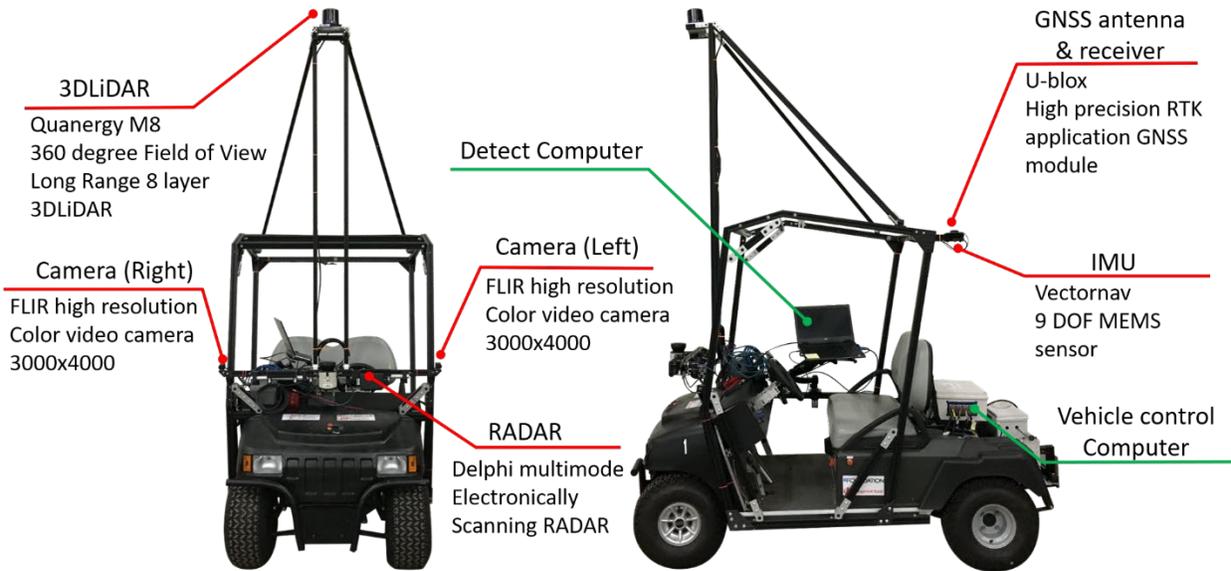
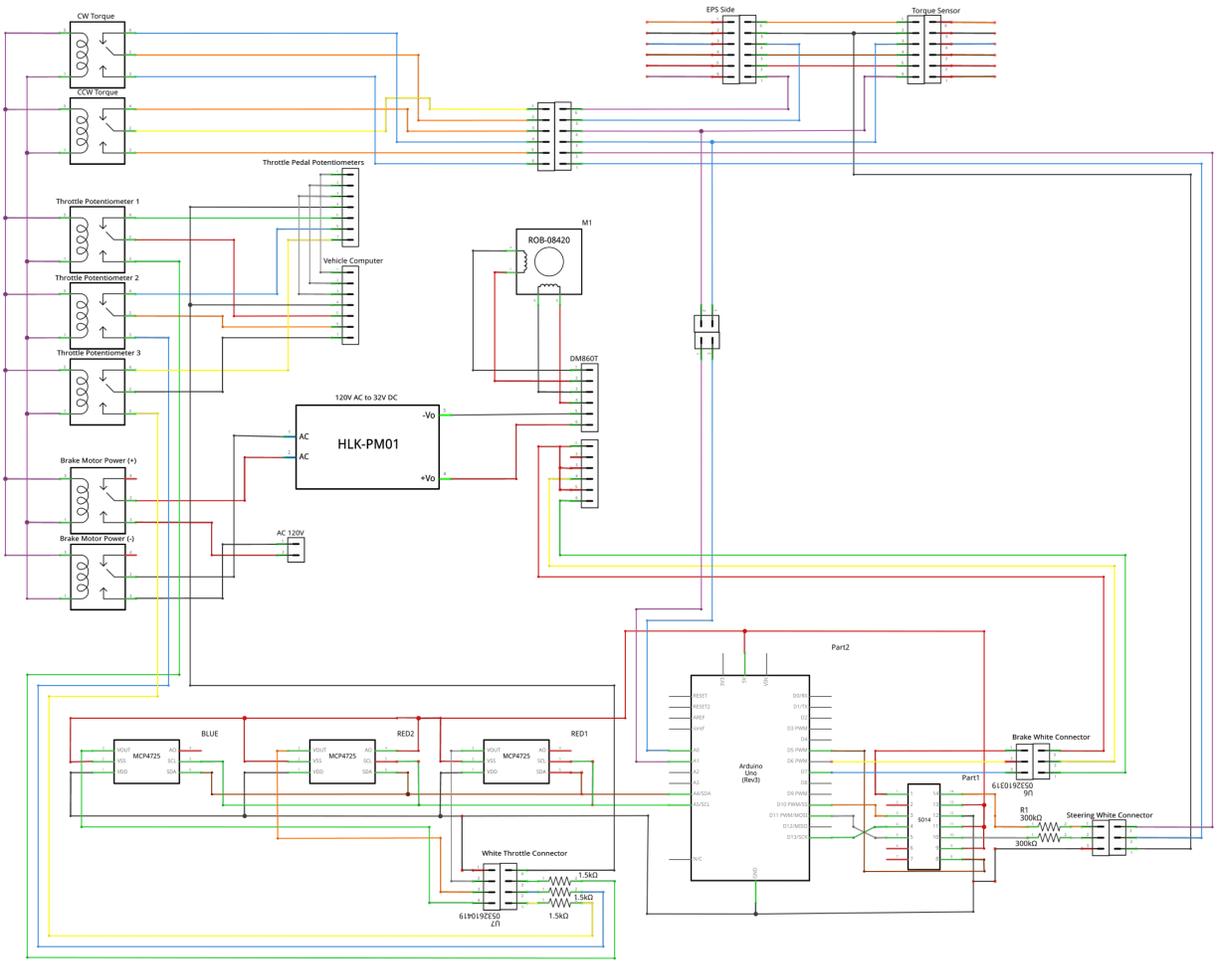


Figure 16. Golf car retrofitting architecture for slow speed tests.

Appendix B: Control Hardware Configuration



fritzing

Figure 17. Drive-by-wire controller wiring configuration.

Vehicle Longitudinal Control Actuator Modification

The brake-by-wire system has been completed. Using a stepper motor with a driver controller to turn a pulley wheel with a hub for attachment to the motor axle as shown in Figure 18, the brake is actuated using the feedback of the brake pedal position from the vehicle CAN network. Originally the power source intended to power the stepper driver was actually greater than its operating range. That leads to the need of replacing the stepper driver as well as designing a suitable power source based on what was available. The 12 V DC power from the car battery wasn't enough voltage to reliably operate the stepper driver, so a 110V AC to 36V DC step down was added. Based on the feedback received from the car (Figure 19), an integer signifying how far the brake motor should rotate from its initial position is used to pulse the stepper CW or CCW. A brake calibration procedure is undergone before each operation, moving the motor step by step until the initial depression of the brake pedal, assuming that the tension may be variable with each operation. This can later be

used as a transition from park to drive. The depression of the brake pedal is related to a deceleration of the vehicle in motion.

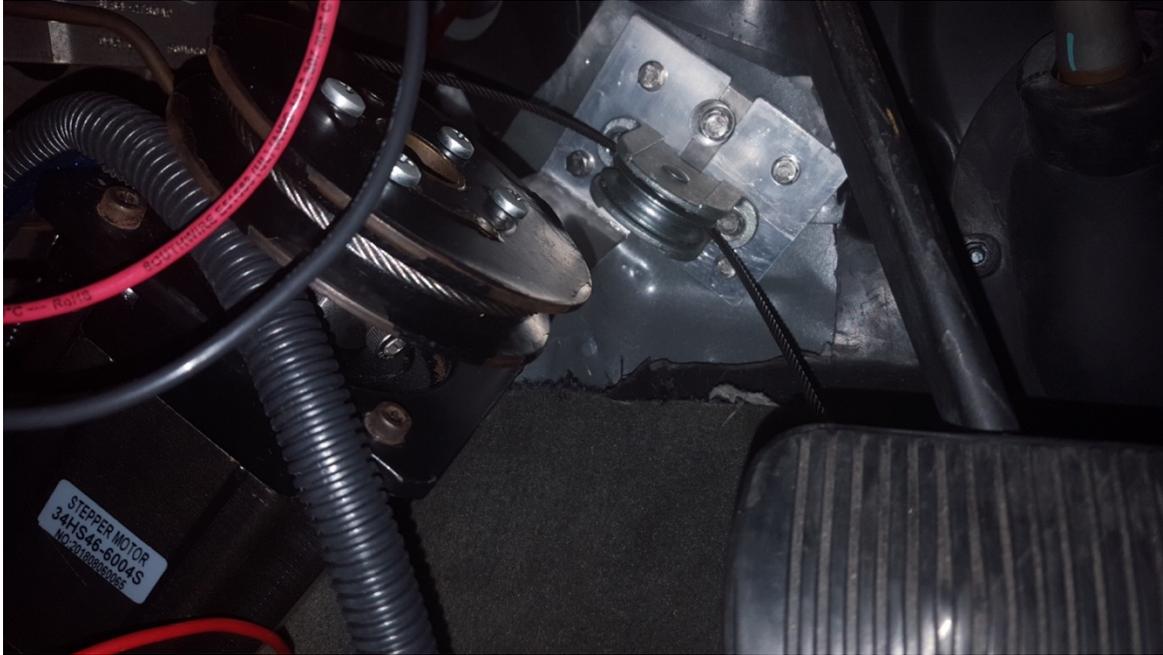


Figure 18. A pulley is drilled and mounted on the steel hub of the motor shaft for brake actuation.

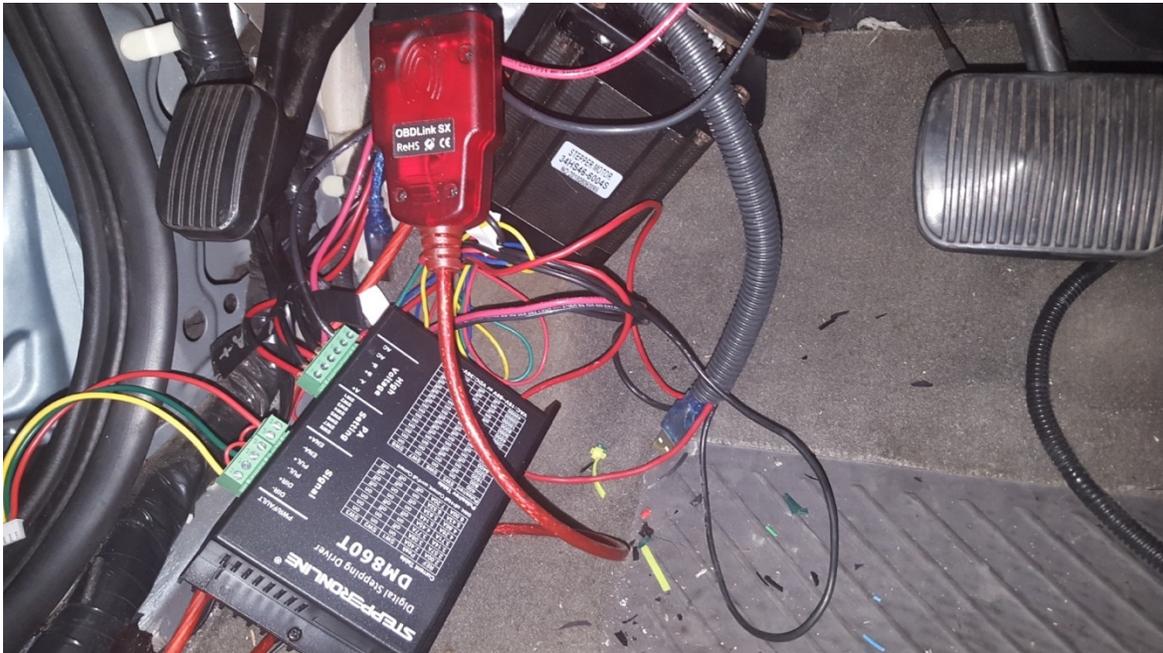


Figure 19. Motor driver gets feedback from CAN about the actual pedal position.

The throttle pedal potentiometers are mimicked by three digital-to-analog converters which is operated by inputting what percentage of throttle pedal depression is needed. Previous work was

undocumented aside from microcontroller code for operation, so documentation of the three MCP4725 wiring has been produced. The brake and throttle operation are both successfully working with the microcontroller, as well as under operation with the autonomous-manual mode switch relays.

Vehicle Lateral Control Actuator Modification

The steering-by-wire can be produced by manipulation of solely the torque sensor, as the Electric Power Steering (EPS) serves as an assist to reduce the amount of torque needed to turn the wheel by the aid of a motor. Mimicked torque outputs can operate the motor in the EPS to rotate the steering wheel, however if the torque signal is also the assist for subduing vibrations produced from steering in motion, the operation of the EPS using a 2-channel digital converter (one for each degree of rotation) will need to operate with the feedback from the torque sensor. Lateral motion of vehicles is dependent down to the flexibility of the tires, so for now, the steering wheel position has been used as the metric for automation of the steering in the vehicle. A PID controller is used to get the steering wheel to a position with applied torques to the wheel. As the automated steering could be erroneous when in use, the torque signals recorded when in autonomous mode are recorded to the microcontroller and sent to the main computer, as input to the EPS is modified to disregard the position-based input coming from the python script operating the steering. Rather than using individual channel DACs like the MCP4725s used for the throttle controller, a 2-channel DAC is used for the CW and CCW signals of the torque sensor. No torque application kept both outputs of the torque sensor in the medial range of 5 volts that it outputted, as one sensor increases in voltage for one direction while the voltage decreases on the other output. The amperage ranges for both outputs also had to be measured to prevent damaging the EPS system after matching the power range using added resistances to the two channels of the DAC.