2020 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
AUTONOMY, ARTIFICIAL INTELLIGENCE & ROBOTICS (AAIR) TECHNICAL SESSION
AUGUST 11-13, 2020 - NOVI, MICHIGAN

PREDICTING ERROR PROPAGATION IN AUTONOMOUS GROUND VEHICLE SUBSYSTEMS

Daniel W. Carruth, PhD[1], Christopher Goodin, PhD[1], Lalitha Dabbiru, PhD[1], Nicklaus Scherer[1], Paramsothy Jayakumar, PhD[2]

[1]Center for Advanced Vehicular Systems, Mississippi State University, Mississippi State, MS
[2]US ARMY CCDC Ground Vehicle Systems Center, Warren, MI

ABSTRACT

Self-driving or autonomous vehicles consist of software and hardware subsystems that perform tasks like sensing, perception, path-planning, vehicle control, and actuation. An error in one of these subsystems may manifest itself in any subsystem to which it is connected. Errors in sensor data propagate through the entire software pipeline from perception to path planning to vehicle control. However, while a small number of previous studies have focused on the propagation of errors in pose estimation or image processing, there has been little prior work on systematic evaluation of the propagation of errors through the entire autonomous architecture. In this work, we present a simulation study of error propagation through an autonomous system and work toward developing appropriate metrics for quantifying the error at both the subsystem and system levels. Finally, we demonstrate how the framework for analyzing error propagation can be applied to analysis of an autonomous systems with a lidar-based sensing and perception system.

Citation: D. W. Carruth, C. Goodin, L. Dabbiru, N. Scherer, P. Jayakumar, "Predicting Error Propagation in Autonomous Ground Vehicle Subsystems", In *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium* (GVSETS), NDIA, Novi, MI, Aug. 11-13, 2020.

1. INTRODUCTION

Autonomous ground vehicles (AGV) have become more capable in recent years [1]. However, the struggle of these vehicles with environmental conditions like weather has been well documented, qualitatively [2]. Nevertheless, there has been relatively little quantitative evaluation of errors induced by environmental factors like rain and dust.

Past work in quantitative error measurement tends to focus either directly on the sensor data or on the overall system-level performance. For example, there have been several studies that quantify the influence of phenomena like rain or snow on lidar sensor performance [3]. Similarly, there have been some laboratory studies on the effect of dust on lidar [4]. System level analyses have focused on

high-level metrics like average speed or distance traveled [5]. Additionally, some early studies related sensor level-errors for inertial navigation systems to mission performance [6][7].

While there is value in sensor-level or system-level error studies, an understanding of how error propagates through the chain of autonomous subsystems is also needed. How do errors in lidar sensors operating in rainy conditions affect the point cloud, the navigation maps derived from the point cloud, and the path planned through those maps? Is there a level of rain for which the resulting path and map are not significantly affected, and at what rain rates would errors start to manifest in the planned path? These questions are critical to answer as self-driving vehicles transition to the consumer market.

The primary difficulty in systematically measuring the relationship between environmentally induced sensor errors and system-level performance is the difficulty in controlling the environmental error sources. It is logistically difficult to perform repeatable, controlled experiments in conditions like dust or rain. For this reason, some recent studies have utilized simulation to inject errors and faults in camera systems [8-11].

Despite these past research efforts, some basic questions for predicting AGV performance remain unanswered. How do environmentally induced sensor errors propagate to the "downstream" subsystems of the AGV? Can subsystem errors be predicted for various environmental conditions? In order to answer these questions, this study develops a method, utilizing simulation, for studying error propagation through the subsystems of an AGV. The study will be described in the following sections with an overview of the method, followed by a presentation of the experiment results and analysis.

2. METHOD

The method used to study error propagation through the AGV systems is outlined below:

- Define the test scenario
- Define the AGV to be tested
- Define sources of error
- Define the system and subsystem level metrics
- Perform experiments
- Find correlations between error sources and metrics

Obstacle detection and avoidance (ODOA) is a critical capability for any autonomous or semi-autonomous system [12]. Therefore, an ODOA scenario was selected for this work to demonstrate error propagation in the AGV. In the scenario, the test vehicle navigated a 90-meter-long test lane. At the center of the lane, 45 meters from the starting position, was a jersey barrier, 1-meter tall x 2 meters wide. The vehicle had to avoid the jersey barrier, return to the test lane, and reach the goal point that lies 45 meters beyond the jersey barrier. The 90-meter test lane was preceded by a 100-meter lane in which the vehicle accelerated to reach the test speed of 10 m/s. The test course is shown in Figure 1.

3. AUTONOMOUS GROUND VEHICLE

The simulated test vehicle was a generic four-wheel-drive truck with a gross vehicle weight of 3000 kg. The simulated vehicle has a real-time kinematic sensor [13] for odometry measurements and a 3D lidar sensor mounted to the roof.

The vehicle autonomous software was conceptually divided into three parts: sensing/perception, path planning, and vehicle control. The architecture was implemented in ROS [14]. In the autonomy system being studied, a 3D point cloud was generated by the lidar sensor. Point clouds were generated at a rate of 10 Hz and registered using the odometry published by the simulated real-time kinematic sensor.

A slope-map was calculated from the lidar point cloud and converted to a cost-map. A path through the cost-map was calculated using A* [15]. Throttle

and steering commands were calculated from the path using the pure pursuit algorithm [16].

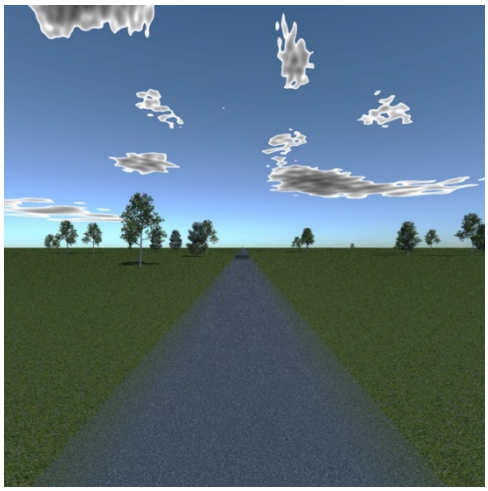

Figure 1: The obstacle avoidance test course. The jersey barrier can be seen near the center of the image.

3.1. Sensing and Perception

A simulated lidar was the primary sensing modality in these experiments. The output of the perception algorithm was a 3D point cloud in sensor coordinates that was registered with the odometry from the real-time kinematic sensor.

The simulated lidar was based on a 64-beam Ouster OS1 lidar. The sensor was mounted on the roof of the vehicle in a standard vertical orientation. The rotation rate was set to 10 Hz and the point cloud was published to a ROS PointCloud2 message after each rotation.

3.2. Path Planner

For path planning, a gridded slope map was created from the point cloud. At each time step, the point cloud was registered to world coordinates using the current odometry. Points were then assigned to a grid location. The grid was centered at (0,0) in local East-North-Up (ENU) coordinates, centered on the jersey barrier obstacle. The grid dimensions were 250x250 meters, while the cell resolution was 0.5 meters.

For the slope-based perception, the value of the highest and lowest points was tracked in each cell and the other points were discarded. The slope of the cell was calculated by dividing the height different between the highest and lowest points by the cell resolution. Cells with a slope greater than 1.0 were flagged as occupied; otherwise a cell was unoccupied.

The grid was updated at 10 Hz with new point cloud data. After each update, the path was re-planned with the current grid and the updated vehicle position. An example of the A* calculation is shown in Figure 2. In the image, black represents free cells while red and purple dots represent occupied cells, color coded by slope. The green line shows the history of the vehicle and the blue line shows the planned path. The green circle is the goal point.

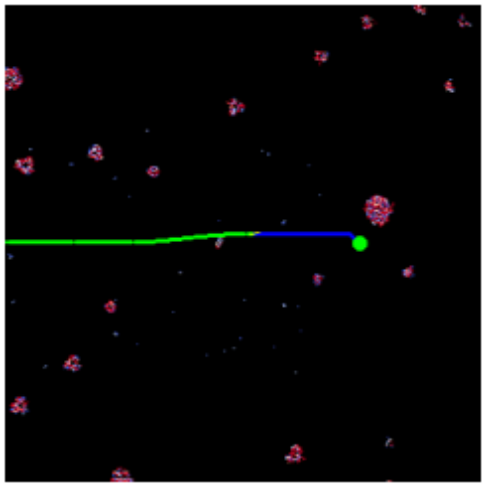

Figure 2: A* grid showing path path (green line) and planned path (blue line) of the AGV.

The vehicle control subsystem used the pure pursuit algorithm [16] to convert the proposed path into throttle and steering commands. In the first step of the pure pursuit algorithm, a local goal point is found on the path at a look-ahead distance along the path from the current vehicle position. The look-ahead distance is a function of the vehicle

speed. Next, the steering angle necessary to reach the goal points is calculated from the vehicle wheelbase.

4. SOURCES OF ERROR

Macfarlane and Stroila [17] identified three main classes of uncertainties in autonomous navigation: sensors, maps, and situations. Uncertainties in sensors include sensor noise and inaccuracy as well as environmentally induced errors caused by rain, dust, fog, or other phenomena. Map uncertainties include errors in object detection and localization. Finally, situational uncertainties pertain to predicting the future state of the dynamic environment. Since the environment in these experiments is static, situational uncertainties were not measured. Instead, the focus was on sensor and map errors. A summary of the error sources in these experiments and the ranges over which they were varied is listed in Table 1.

Error type	Units	Range	Step
Range Noise	mm	5 – 35	5
RMS Pose Error	mm	200 – 1400	200
Dust Density		1-8	1
Rain Rate	mm/hour	4-28	4

Table 1: Sources of error.

4.1. Lidar Range Noise

Lidar sensors are subject to noise errors in measurement accuracy. A survey of reported lidar specifications shows that typical RMS range errors are between 1-5 cm [23-25]. In this work, error is added as Gaussian noise to the raw sensor signals to approximate range error for the lidar.

4.2. Rain

The influence of rain on lidar has been well documented [3]. The primary effect of rain on lidar sensors is to reduce the range of the sensor and increase the range error. In the simulations reported in this work, the rain rate was varied from 0-28 mm/h, with 28 mm/h representing an unusually heavy rain.

4.3. Dust

While it is well-known that dust can obscure lidar targets, there has been little work on quantifying this effect for automotive lidar sensors. It has been shown that the optical depth of the dust cloud correlates well with the probability of the dust obscuring the lidar target [4]. The primary error mode is for the lidar to return from the dust cloud itself, rather than the surfaces in the environment. Since the optical properties of the dust cloud are more important for lidar interaction than the mass properties, dust particles were added in nine increments from no dust to a totally opaque dust cloud. The dust cloud was added to the scene directly in front of the target. This mimics the situation where dust could obscure a target like a vehicle or pedestrian.

4.4. GPS with Real-time Kinematic Noise

Real-time kinematic sensors are subject to noise errors in measurement accuracy. A survey of real-time kinematic lateral position errors indicates that error typically ranges from 0.5 to 3.0 meters [18]. Therefore, as with the lidar range noise, Gaussian noise is added to the raw lateral position to approximate error for the real-time kinematic sensor.

5. METRICS

Information flows through the subsystems from perception to mapping to planning to control. Therefore, metrics are proposed for each of these subsystems. These metrics are summarized in Table 2.

Time to complete is the total amount of time from the time the vehicle starts moving until it reaches the goal point at the end of the course.

A trial ends in one of four states. If the vehicle reaches the goal point within 90 seconds, the vehicle successfully completes the trial and "Completion" is true, otherwise it is false. If the vehicle collided with the obstacle, it fails to complete the trial and "Collision" was true, and false otherwise. If the vehicle rolled over (flipped), the vehicle failed to complete the trial and "Rolled Over" was true, and false otherwise.

Point cloud error quantifies the impact of environmental factors like rain and dust on the accuracy of the point cloud. Each time a scan is published in the simulation, the point cloud is compared to the "true" point cloud in the absence of rain or dust. The average point-wise error is computed by comparing the resulting clouds point by point.

Odometry error is the average difference between the vehicle's actual position and its position as measured by the odometry at each time step.

Grid error quantifies the propagation of error from the perception subsystem to the mapping subsystem. The "true" grid with perfect ground truth is compared to the grid created from the sensor data with errors. Error is quantified as the average difference in measured slope in each cell.

Path error quantifies the propagation of error from the perception subsystem to the planning subsystem. The "ideal" path follows the centerline of the test lane, except for a deviation around the obstacle. The path error metric quantifies the average deviation of the vehicle from this ideal path during the experiment.

6. Simulator

The AGV simulator was the MSU Autonomous Vehicle Simulator (MAVS) [19]. The MAVS provides a software library for physics-based simulation of lidar and camera sensors. In this work, the MAVS library was integrated with ROS

Metric	Units
Time to complete	seconds
Completion	Boolean
Collision	Boolean
Rolled Over	Boolean
Point cloud error	meters
Odometry Error	meters
Grid Error	meters / cell
Path Error	meters

Table 2: System and sub-system metrics

such that the simulated sensor data was published to standard ROS topics such as "PointCloud2", "Image", and "Odometry".

The Chrono multibody dynamics engine [20] was used to simulate the dynamics of the vehicle. The simulated vehicle reported the "true" pose of the vehicle to the sensor simulations.

MAVS was used to simulate the effect of rain using the model presented in [21]. An example of the effect of rain on a lidar scan is show in Figure 3, which shows the reduced range of scans in rain. Dust was simulated in MAVS using a particle system model [22] with optical properties derived from laboratory measurements of lidar-dust interaction [4].

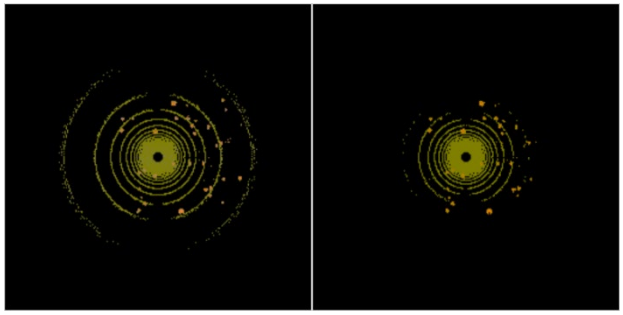

Figure 3: Example of the impact of rain on a lidar point cloud. The left image is a top-down view of a scan during clear conditions, while the right is a scan from the same scene in 10 mm/h rain.

7. Experiments

Experiments were conducted for the four error types shown in Table 1. Each error type was divided into increments. Additionally, simulations with no injected errors were run for comparison. In

total, 3000 simulations were run, amounting to about 48 hours of simulated experiments. An example of output from a simulation is shown in Figure 4.

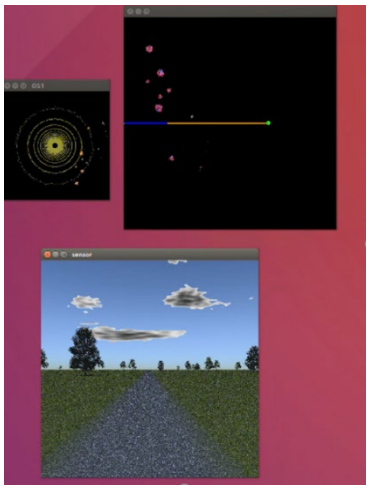

Figure 4: Example output from a simulation.

8. Results

8.1. System Performance

The primary measure of system-level performance for the AGV is whether it successfully reached its objective and avoided a collision with the barrier and did not rollover the vehicle. Figures 5-8 show the percentage of trials that resulted in failure. For the tested system, the overall failure rate was fairly low (6.5% of trials) and less than 1% in no- or low-error trials (ideal trials + smallest level of error).

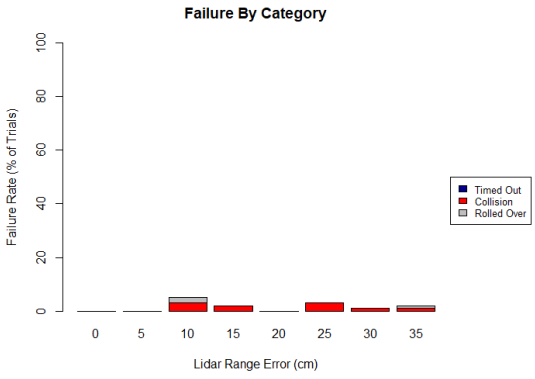

Figure 5: Collisions and rollovers for lidar range error from 0-35 mm.

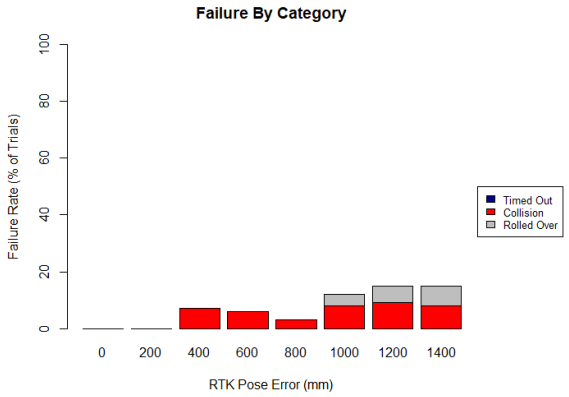

Figure 6: Collisions and rollovers for GPS with Real-time Kinematics pose error.

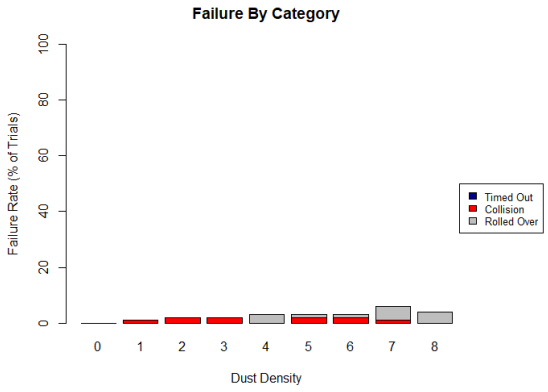

Figure 7: Collisions and rollovers for dust.

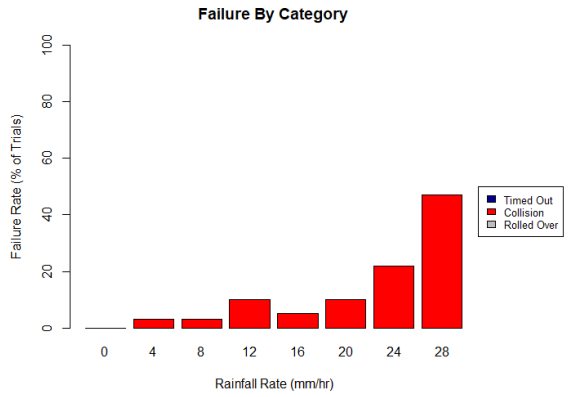

Figure 8: Collisions and rollovers in rain (0 to 28 mm/h).

	Sensor Error	Occupancy Grid Error	Path Error	Collision	Rollover
IDEAL	0.000 (0.000)	1.003 (0.073)	1.010 (0.398)	0.00%	0.00%
LIDAR	0.016 (0.007)	1.080 (0.079)	0.907 (0.295)	1.43%	0.43%
error = 5	0.004 (0.000003)	1.068 (0.071)	0.893 (0.166)	0.00%	0.00%
error = 10	0.008 (0.000007)	1.064 (0.077)	0.894 (0.354)	3.00%	2.00%
error = 15	0.012 (0.000011)	1.075 (0.081)	0.916 (0.344)	2.00%	0.00%
error = 20	0.016 (0.000013)	1.062 (0.076)	0.886 (0.185)	0.00%	0.00%
error = 25	0.020 (0.000019)	1.087 (0.067)	0.910 (0.306)	3.00%	0.00%
error = 30	0.024 (0.000019)	1.095 (0.079)	0.946 (0.414)	1.00%	0.00%
error = 35	0.028 (0.000026)	1.106 (0.090)	0.905 (0.209)	1.00%	1.00%
DUST	0.085 (0.042)	1.037 (0.079)	1.936 (0.831)	1.25%	1.75%
rate = 1	0.021 (0.009)	1.031 (0.080)	1.801 (0.625)	1.00%	0.00%
rate = 2	0.042 (0.012)	1.039 (0.076)	1.834 (0.529)	2.00%	0.00%
rate = 3	0.060 (0.013)	1.030 (0.086)	1.851 (0.552)	2.00%	0.00%
rate = 4	0.082 (0.013)	1.049 (0.085)	1.959 (0.690)	0.00%	3.00%
rate = 5	0.095 (0.015)	1.045 (0.080)	1.994 (0.600)	2.00%	1.00%
rate = 6	0.115 (0.019)	1.026 (0.070)	1.892 (0.474)	2.00%	1.00%
rate = 7	0.127 (0.018)	1.034 (0.077)	2.026 (1.149)	1.00%	5.00%
rate = 8	0.137 (0.020)	1.047 (0.073)	2.133 (1.461)	0.00%	4.00%
RAIN	5.994 (0.720)	1.429 (0.106)	0.823 (0.411)	14.29%	0.00%
rate = 4	4.650 (0.125)	1.382 (0.098)	0.923 (0.280)	3.00%	0.00%
rate = 8	5.411 (0.151)	1.406 (0.109)	0.892 (0.146)	3.00%	0.00%
rate = 12	5.863 (0.170)	1.429 (0.106)	0.880 (0.380)	10.00%	0.00%
rate = 16	6.187 (0.155)	1.438 (0.104)	0.851 (0.239)	5.00%	0.00%
rate = 20	6.440 (0.190)	1.446 (0.100)	0.844 (0.588)	10.00%	0.00%
rate = 24	6.602 (0.182)	1.439 (0.102)	0.740 (0.370)	22.00%	0.00%
rate = 28	6.802 (0.202)	1.460 (0.107)	0.633 (0.588)	47.00%	0.00%
GPS w/Real-time Kinematics	1.184 (0.672)	1.753 (0.464)	2.654 (1.580)	5.86%	2.43%
error = 200	0.304 (0.064)	1.150 (0.116)	1.102 (0.297)	0.00%	0.00%
error = 400	0.610 (0.142)	1.379 (0.159)	1.679 (0.469)	7.00%	0.00%
error = 600	0.868 (0.235)	1.563 (0.228)	2.154 (0.658)	6.00%	0.00%
error = 800	1.171 (0.315)	1.776 (0.284)	2.684 (0.996)	3.00%	0.00%
error = 1000	1.520 (0.309)	2.005 (0.242)	3.043 (1.016)	8.00%	4.00%
error = 1200	1.716 (0.466)	2.059 (0.331)	3.827 (2.032)	9.00%	6.00%
error = 1400	2.099 (0.481)	2.341 (0.358)	4.085 (1.865)	8.00%	7.00%

8.2. Sub-System Performance

Table 3 shows the average sub-system performance. In ideal conditions the sensor error, the lidar range error, and the odometry error are 0 by design. There is a minimum level of error observed in the occupancy grid in all conditions because the occupancy grid error is based on a comparison of the occupancy grid generated from the point cloud and a ground truth representation of the environment. In every case, there is error in the cells due to presence of objects on the map that are occluded and cannot be perceived even by the ideal lidar. We also observe some variability in the deviation from the path even in ideal conditions because the vehicle must leave the optimal path in order to avoid the obstacle.

Figures 9-11 show the relationship between the measured error in a sub-system and the measured error in the downstream sub-system. For example, Figure 9 shows the relationship between the average error in the point cloud generated from the lidar data and the average error in the occupancy grid generated based on the point cloud error.

At this first stage of processing from the sensor data to the occupancy grid, we observe a small, limited relationship between point cloud error and occupancy grid error.

In contrast to the weak relationship between point cloud error and occupancy grid error, error in the odometry pose has a clear linear relationship with the observed occupancy grid error (see Figure 10). As the odometry pose error increases, the occupancy grid error also increases. Also, the occupancy grid error associated with odometry error is larger than occupancy grid error associated with the point cloud error.

In Figure 11, we see a complex relationship between the error observed in the occupancy grid and deviation from the ideal path. When path deviation is near 0, the outcome of the trial is a collision with the obstacle. As the occupancy grid errors increases, we generally observe increasing deviation from the ideal path. There are examples of significant path deviation at lower levels of occupancy grid error that generally lead to successful outcomes.

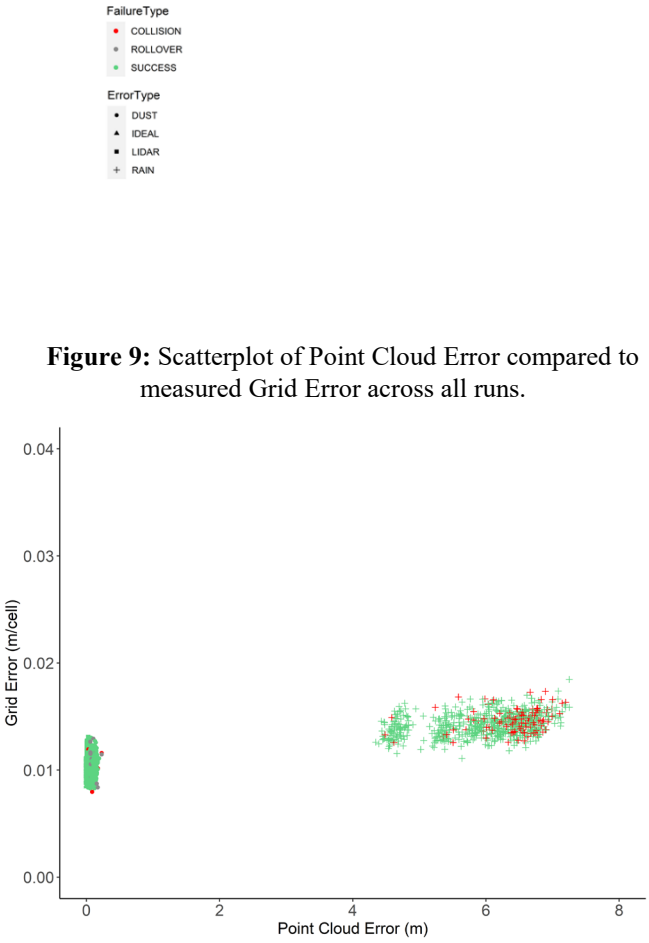

Figure 9: Scatterplot of Point Cloud Error compared to measured Grid Error across all runs.

Table 3: Mean and standard deviation in sub-system errors (sensor – lida ror for lidar, dust, and rain and pose error for real-time kinematics conditions, occupancy grid, and path error) and of failures (collisions and rollovers).

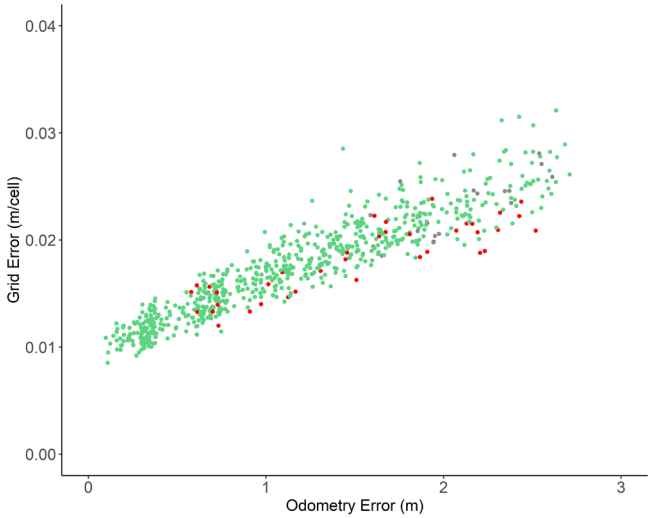

Figure 11: Scatterplot of Grid Error compared to measured Path Error across all runs. Black horizontal line indicates ideal deviation to avoid the obstacle.

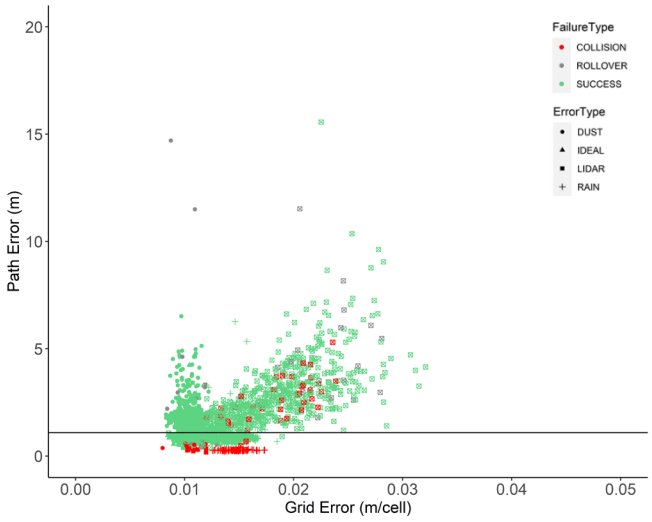

Figure 10: Scatterplot of Odometry Error compared to measured Grid Error across all runs.

9. Summary and Conclusions

In this study, a lidar-based perception system, an occupancy grid world model, an A* path planner, and a pure pursuit control system were tested in a simple obstacle detection and avoidance task. In ideal conditions, the system performed very well. Even considering performance at the lowest levels of sensor and environment error, the vehicle achieved its objective in more than 99% of trials. However, in more realistic conditions that include inherent and environmentally induced sensor error, the system failed at significantly higher rates (6.5% across all trials and as high as 47% of trials for one condition).

The increase in failures depended on the source of the error. Increases in lidar range error had little effect on failure rate and the rate of failure was not correlated with the increase in range error. Increasing odometry error led to increased risk of both collision with the barrier and rollover of the vehicle.

Odometry error presents a challenge for autonomous vehicles. Common error rates for GPS-based systems range from 500 mm to 3000 mm. The accuracy of the world model is highly dependent on the accuracy of odometry. The highest levels of odometry error resulted in failure in close to 20% of trials.

For the lidar-based perception algorithm tested here, rain has a critical impact on lidar performance that leads directly to a failure to detect and avoid obstacles even at relatively low speeds (10 m/s; 22.3 mph). At very high rainfall rates (24 mm/h and 28 mm/h), the vehicle collided with the obstacle in 22% and 47% of trials, respectively. Rainfall reduces the effective range of the lidar limiting the distance and time available for the vehicle to detect and respond to the obstacle.

The dust cloud had a localized effect on perception that led to fewer failures than the global effects of increased rainfall. In some cases, the dust cloud was perceived as an obstacle leading the system to move and inadvertently avoid the actual obstacle.

Despite being a very general measure of system performance, the system failure rates and the differences in failure rates for different sources of error provides insight into the fitness of the AGV system.

Our results also indicate the need for repeated trials. In the simulations presented, there were

random variations in vehicle start position, start orientation, and placement of trees that led to significant variations in system performance over 100s of trials. In some of our scenarios, the failure rates are less than 1% but a single failure across 100 trials in a simple obstacle avoidance task is unacceptable for an operational autonomous vehicle.

This study set out to investigate the propagation of error through the sub-systems of the autonomous vehicle control software. However, our analysis of the sub-system performance does not indicate clear relationships in observed error at each of the stages of the AGV processing system. The authors believe this is at least in part related to the aggregate nature of our selected metrics. Each of the metrics is an average of the measure over the course of a trial and may not fully characterize momentary, critical errors in the sub-systems that lead to poor overall outcomes.

Future work should revise the current sub-system metrics or identify new sub-system metrics. This should include examining time-series recordings of the sub-system metrics for analysis. In addition, the current work focuses on a particular AGV system that uses a lidar sensor paired with a slope-detection algorithm to generate an occupancy grid that is then used to plan a path that is then executed by a pure pursuit algorithm. There are many other combinations of sensors and algorithms that could be used for perception of the obstacles and investigating similarities and differences in different algorithms' responses to error is expected to increase understanding of how error affects sub-system and system-level performance.

10. REFERENCES

[1] R. W. Peterson. "The future of mobility and shifting risk," PhD thesis, Santa Clara University, 2018.

[2] K. Stock. Self-driving cars can handle neither rain nor sleet nor snow. *Bloomberg Businessweek*, Sept., 2018.

[3] R. H. Rasshofer, M. Spies, and H. Spies. "Influences of weather phenomena on automotive laser radar systems," *Advances in Radio Science*, vol. 9, pp. 49–60, 2011, doi:10.5194/ars-9-49-2011.

[4] C. Goodin, P. J. Durst, Z. T. Prevost, and P. J. Compton. "A probabilistic model for simulating the effect of airborne dust on ground-based lidar," in *Active and Passive Signatures IV*, vol. 8734, International Society for Optics and Photonics, 2013, pp. 87340D.

[5] P. J. Durst, C. T. Goodin, D. T. Anderson, and C. L. Bethel. "A reference autonomous mobility model," in *2017 Winter Simulation Conference (WSC)*, 2017, pp. 4026–4037.

[6] B. W. Welch and J. W. Connolly. "Autonomous navigation error propagation assessment for lunar surface mobility applications," NASA Glenn Research Center, Tech. Rep. NASA/TM-2006-214354, May 2006.

[7] C. A. Borja, J. M. Mirats Tur, and J. L. Gordillo. "State your position," in *IEEE Robotics & Automation Magazine*, vol. 16, no. 2, pp.82-90, June 2009, doi: 10.1109/MRA.2009.932523.

[8] G. Li, S. K. Sastry Hari, M. Sullivan, T. Tsai, K. Pattabiraman, J. Emer, and S. W. Keckler. "Understanding error propagation in deep learning neural network (dnn) accelerators and applications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis,* pp. 8-20, Nov. 2017, doi: 10.1145/3126908.3126964.

[9] S. Jha, S. S. Banerjee, J. Cyriac, Z. T. Kalbarczyk, and R. K. Iyer. "Avfi: Fault injection for autonomous vehicles," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pp. 55–56, 2018.

[10] S. Jha, T. Tsai, S. Hari, M. Sullivan, Z. Kalbarczyk, S. W. Keckler, and R. K. Iyer. "Kayotee: A fault injection-based system to assess the safety and reliability of autonomous

vehicles to faults and errors," in *3rd IEEE International Workshop on Automotive Reliability & Test*, 2018.

[11] R. Michelmore, M. Kwiatkowska, and Y. Gal. "Evaluating uncertainty quantification in end-to-end autonomous driving control," arXiv preprint arXiv:1811.06817, 2018.

[12] J. Oroko and G. N. Nyakoe. "Obstacle avoidance and path planning schemes for autonomous navigation of a mobile robot: a review," in *Proceedings of Sustainable Research and Innovation Conference*, pp. 314–318, 2014.

[13] M. Skoglund, T. Petig, B. Vedder, H. Eriksson, and E. M. Schiller. "Static and dynamic performance evaluation of low-cost rtk gps receivers," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 16–19, 2016.

[14] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, pp. 5, Kobe, Japan, 2009.

[15] P. E. Hart, N. J. Nilsson, and B. Raphael. "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[16] R. C. Coulter. "Implementation of the pure pursuit path tracking algorithm," Tech. Rep., Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, 1992.

[17] J. Macfarlane and M. Stroila. "Addressing the uncertainties in autonomous driving," *SIGSPATIAL Special*, vol. 8 no. 2, pp. 35–40, 2016.

[18] T. Mahmoud and B. R. Trilaksono. "Integrated ins/gps navigation system," *International Journal on Electrical Engineering and Informatics*, vol. 10, no. 3, pp. 491–512, 2018.

[19] C. Goodin, M. Doude, C. Hudson, and D.W. Carruth. "Enabling off-road autonomous navigation-simulation of lidar in dense vegetation," *Electronics*, vol. 7, no. 9, pp.154, 2018.

[20] A. Tasora, R. Serban, H. Mazhar, A. Pazouki, D. Melanz, J. Fleischmann, M. Taylor, H. Sugiyama, and D. Negrut. "Chrono: An open source multi-physics dynamics engine," in *International Conference on High Performance Computing in Science and Engineering*, pp. 19–49, 2015.

[21] C. Goodin, D. W. Carruth, M. Doude, and C. Hudson. "Predicting the influence of rain on lidar in adas," *Electronics*, vol. 8, no. 1, pp. 89, 2019.

[22] J. X. Chen, X. Fu, and J. Wegman. "Real-time simulation of dust behavior generated by a fast traveling vehicle," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 9, no. 2, pp. 81–104, 1999.

[23] LIDAR, Velodyne. "High Definition LIDAR HDL-64E S2 Specifications."

[24] C. L. Glennie, A. Kusari, and A. Facchin. "Calibration and stability analysis of the VLP-16 laser scanner," *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences,* vol. 9, 2016.

[25] M. Mittet, H. Nouira, X. Roynard, F. Goulette, and J-E. Deschaud. "Experimental assessment of the quanergy m8 lidar sensor," 2016.