

RACECAR - The Dataset for High-Speed Autonomous Racing

Amar Kulkarni¹, John Chrosniak¹, Emory Ducote¹, Florian Sauerbeck², Andrew Saba³, Utkarsh Chirimar¹, John Link¹, Marcello Cellina⁴, Madhur Behl¹

¹University of Virginia, ²Technical University of Munich, ³Carnegie Mellon University, ⁴Politecnico di Milano

{ark8su, jlc9wr, etd4sv, uc6gq, jwl9vq, madhur.behl}@virginia.edu

{florian.sauerbeck}@tum.de {asaba}@andrew.cmu.edu {marcello.cellina}@polimi.it

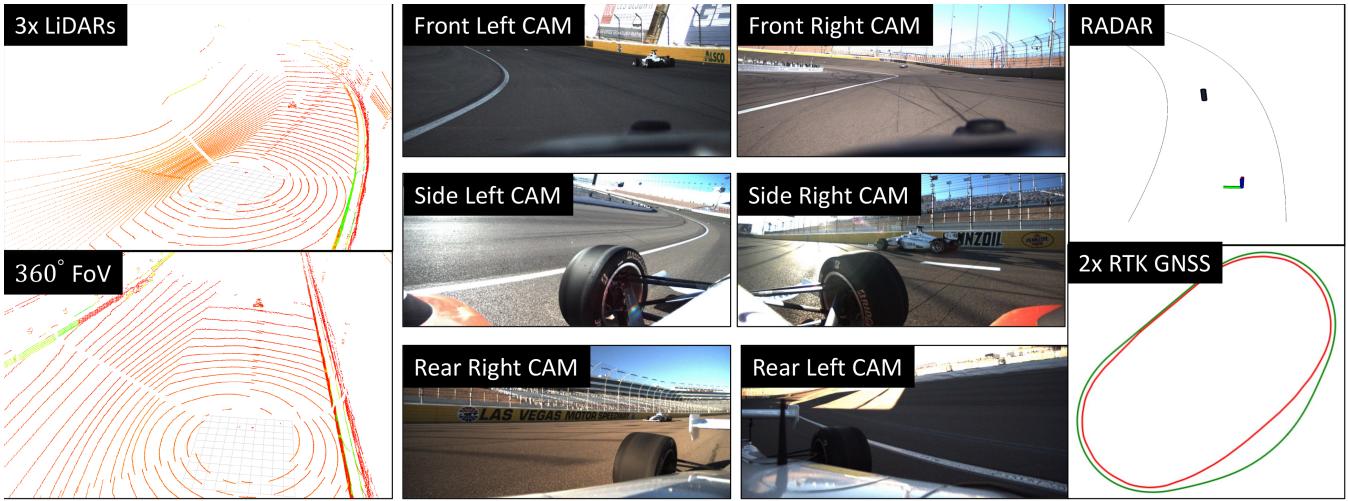


Fig. 1: RACECAR is the first multi-modal sensor data collected from fully autonomous Indy race cars operating at speeds of up to 170 mph (274 kph). The dataset spans 11 racing scenarios with over 6.5 hours of track activity.

Abstract— This paper describes the first open dataset for full-scale and high-speed autonomous racing. Multi-modal sensor data has been collected from fully autonomous Indy race cars operating at speeds of up to 170 mph (273 kph). Six teams who raced in the Indy Autonomous Challenge have contributed to this dataset. The dataset spans 11 interesting racing scenarios across two race tracks which include solo laps, multi-agent laps, overtaking situations, high-accelerations, banked tracks, obstacle avoidance, pit entry and exit at different speeds. The dataset contains data from 27 racing sessions across the 11 scenarios with over 6.5 hours of sensor data recorded from the track. The data is organized and released in both ROS2 and nuScenes format. We have also developed the ROS2-to-nuScenes conversion library to achieve this. The RACECAR data is unique because of the high-speed environment of autonomous racing. We present several benchmark problems on localization, object detection and tracking (LiDAR, Radar, and Camera), and mapping using the RACECAR data to explore issues that arise at the limits of operation of the vehicle.

I. INTRODUCTION

While autonomous vehicle research and development is focused on handling routine driving situations, achieving the safety benefits of autonomous vehicles also requires a focus on driving at the limits of the control of the vehicle. Demonstrating high-speed autonomous racing can be considered as a grand challenge for autonomous driving and making progress here has the potential to enable

breakthroughs in agile and safe autonomy. In recent years autonomous racing competitions, such as F1/10 autonomous racing [1], [2], Indy Autonomous Challenge [3], [4], [5], and Formula SAE Driverless [6], [7] are becoming proving grounds for testing motion planning and control algorithms at high speeds. While the autonomous racing community and research have grown [8] by an order of magnitude due to these competitions, the field is still very specialized and exclusive. Full-scale autonomous racing like the Indy Autonomous Challenge requires a large research team to develop the autonomous racing stack. As such, there are only 9 university teams in the world that own and operate full-scale fully-autonomous Indy race cars. The barrier to entry into autonomous racing is high and the underlying research challenges remain elusive from the reach of the larger computer vision, machine learning, and robotics communities.

This paper presents the RACECAR dataset which contains multi-modal sensor data collected from the Indy Autonomous Challenge during the 2021-22 racing season. The autonomous race cars were outfitted with a full sensor suite for localization and perception, including solid state LiDARs, high precision RTK GNSS and IMU units, multiple cameras, and radar sensors. This paper has the following contributions:

- 1) We present the RACECAR dataset describing the



Fig. 2: [Left] The AV-21 is a modified Indy Lights racecar retrofitted with 3 LiDARs, 6 Cameras, 3 Radars, 2 GNSS systems. [Right] The Indy Autonomous Challenge (IAC) held its first autonomous race at the Indianapolis Motor Speedway (IMS) track in 2021, followed by a head-to-head overtaking competition held at the Las Vegas Motor Speedway (LVMS).

unique features, and autonomous racing contexts within which this data was collected.

- 2) We developed a ROS2 to nuScenes conversion library which allows us to release the dataset in both ROS2 bag file format, ubiquitously used by the robotics community, as well as in nuScenes format.
- 3) Using the RACECAR data, we provide benchmark challenges for the research community on high-speed localization, object detection, and tracking with baselines derived from techniques deployed on real autonomous racecars.

Our intention is to democratize the field of autonomous racing, making it accessible to researchers who do not have access to a racecar. In doing so, we hope that the RACECAR data will enable further advances in perception, planning, and control for autonomous driving at its limits by making the underlying algorithms more robust and stress tested at high speeds. The data and associated code are located at https://github.com/linklab-uva/RACECAR_DATA

II. RELATED WORK

Although there are several autonomous racing competitions held at different scales, there is no large-scale autonomous racing dataset available. The Formula Student Objects in Context [9] data provides annotated camera and LiDAR frames with track bounds indicated by colored cones. More generally, several large-scale autonomous driving datasets have been released in recent years, some accompanied by open benchmark challenges. The most notable among these is the KITTI Dataset [10] which has led to improvements and new methods for 3D object detection, visual odometry, and Simultaneous Localization and Mapping (SLAM). Other datasets such as the Waymo Open Dataset [11], and the Lyft Level 5 Data [12] are also noteworthy due to the scale, scenario coverage, and quality of annotations. The nuScenes Dataset [13] is another popular autonomous driving dataset that includes camera, LiDAR, Radar, GPS, and CANBus data. Also included are annotations of semantic descriptions, vehicle attributes such as velocity and pose, and cuboid bounding boxes.

The top speed of any of the vehicles within these datasets remains limited to highway driving speed. RACECAR data is not a replacement for existing AV datasets, but instead provides a unique setting that is not possible to capture in

on-road testing. Since the AV-21 racecar has a similar set of sensors as one would find on any AV prototype, the high-speed nature of the RACECAR data makes it suitable to test the limits of perception algorithms.

III. RACECAR: DATA COLLECTION

The RACECAR dataset is compiled by contributions from several teams, all of whom competed in the inaugural season of the Indy Autonomous Challenge during 2021-22. Nine university teams participated in two races. The first race was held at the Indianapolis Motor Speedway (IMS) track in Indiana, USA in October 2021 (Fig. 2[Right]). This track is a 2.5 mile (4 km) oval and is home to the famous 'Indy 500' race. The second race was held at Las Vegas Motor Speedway (LVMS) in January 2022 (Fig. 2[Right]). The track is shorter (1.5 mile) and more aggressively banked (up to 20 degrees in the turns) making it challenging to run the cars at high speeds. At IMS, teams reached speeds up to 150 mph on straights and 136 mph in turns, competing in solo vehicle time trials and obstacle avoidance. At LVMS, teams participated in a head-to-head overtaking competition reaching speeds in excess of 150 mph!, with the fastest overtake taking place at 170 mph.

A. Sensor Configuration

The AV-21 Indy Lights vehicle (Fig. 2[Left]) is outfitted with three radars, six pinhole cameras, and three solid-state LiDARs. Each of the sensor modalities covers a 360-degree field of view around the vehicle. For localization, the vehicle is equipped with two sets of high-precision Real-Time Kinematic (RTK) GNSS receivers and IMU. The chassis, as well as the steering, powertrain, and brake system, are as close as possible to the base Indy Lights race car but were controlled autonomously using a custom drive-by-wire system. The top speed of the vehicle is rated at 180 mph. Detailed sensor configuration, specification, and calibration information is available

B. Racing Scenarios

Table I shows the eleven different racing scenarios which are included in the RACECAR dataset. There are 6 scenarios $S_1 \dots S_6$ from LVMS and 5 scenarios $S_7 \dots S_{11}$ from the IMS track. The scenarios are further categorized on the basis of solo, multi-agent, slow-speed, and high-speed runs with

Scenario	Track	Description	Speeds
S_1	LVMS	Solo Slow Lap	< 70 mph
S_2	LVMS	Solo Slow Lap	70-100 mph
S_3	LVMS	Solo Fast Lap	100-140 mph
S_4	LVMS	Solo Fast Lap	> 140 mph
S_5	LVMS	Multi-Agent Slow	< 100 mph
S_6	LVMS	Multi-Agent Fast	> 130 mph
S_7	IMS	Solo Slow Lap	< 70 mph
S_8	IMS	Solo Slow Lap	70-100 mph
S_9	IMS	Solo Fast Lap	100-140 mph
S_{10}	IMS	Solo Fast Lap	> 140 mph
S_{11}	IMS	Pylon Avoidance	< 70 mph

TABLE I: RACECAR Scenarios

speeds indicated in Table I. Scenario S_6 is especially exciting since it contains several multi-agent runs between pairs of several teams at speeds of over 130mph. S_{11} is a scenario from IMS with static obstacle avoidance. By spanning these eleven scenarios, the RACECAR dataset provides an interesting mix of solo laps, multi-agent laps, overtaking situations, high-accelerations, banked tracks, obstacle avoidance, and pit entry and exit at different speeds.

IV. RACECAR: DATA ORGANIZATION

Each team's autonomous racing stack used the Robot Operating System (ROS) middleware. The raw sensor data for each scenario was logged using a ROS2 bag format. ROS2 bags are a popular database storage format and a variety of tools have been written to allow one to store, process, replay, and visualize bag data. Each data frame is composed of a serialized message and a UNIX timestamp.

A. Data Processing and Synchronization

ROS bag data has been preprocessed in several ways. We pruned the bag files such that large swaths of time spent idling in certain parts of the track are removed. The next step was converting all the GNSS data into a uniform Cartesian coordinate system across the entire RACECAR dataset. This is especially useful for multi-agent scenarios (S_5, S_6) where we have positions of both vehicles on track expressed in the same global Cartesian coordinate frame. The two data sources from different teams running in the same session were synchronized using both the UNIX timestamps in the data, as well as reported GPS satellite time.

B. Ground Truth Annotations

Currently, the data is not professionally annotated with bounding boxes, but we provide centimeter-level accurate positions of the vehicles in a common global coordinate frame to evaluate perception benchmarks. The Real-Time Kinematic (RTK) GNSS accuracy is below 1-2 cm. Included in the GNSS data message is a standard deviation value that drops below 0.05 m when a RTK fix is acquired. Also included in each packet is a solution status which denotes the uncertainty of the RTK correction. Centimeter level accuracy for a racecar of footprint 4.918 m x 1.886m is very accurate and a reasonable margin of error.

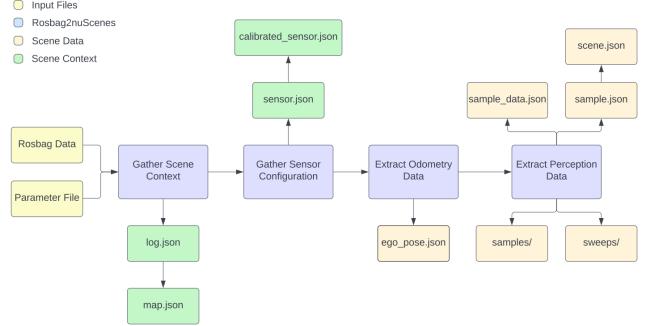


Fig. 3: ROS2 bags to nuScenes conversion process

V. ROS2 TO NUSCENES

As stated earlier, one of the motivations behind releasing the RACECAR data is to present the data to as many researchers as possible. Therefore, we also convert the entire dataset into the nuScenes format to improve the dataset's accessibility to the general self-driving community.

In nuScenes format, data is organized in a tiered structure, starting with a scene at the highest level. A scene consists of a continuous stream of data for which a scenario took place, along with information on what occurred in the scene, the location of the scene, and when it was recorded. For our data, scenes correspond to scenarios (from Table I). The continuous stream of data present within a scene is used to create samples at fixed time intervals, containing all sensor measurements (e.g. LiDAR, Radar, or camera). All sensor data is recorded with a corresponding pose of the ego vehicle in an inertial frame and an extrinsic matrix to convert sensor readings to the inertial frame. The intrinsic calibration of camera sensors is also included to project between 3-dimensional coordinates and the image plane.

The `rosbag2nuscenes` library was developed to convert the data originally stored in ROS2 bag files to the nuScenes format. An overview of the conversion process is shown in Figure 3. Using the `rosbag2` Python API, the conversion library reads through the database entries and extracts the necessary ROS2 messages, converting them to JSON files specific to the nuScenes schema. The sampling rate is set at the default rate of 2 Hz, but is adjustable as laps at higher speeds warrant higher sampling rates than those at lower speeds. Scenes also last significantly longer than the twenty-second clips in the original nuScenes dataset.

VI. AUTONOMOUS RACING BENCHMARKS

The RACECAR data contains full multi-modal sensor coverage of 11 exciting racing situations with full-scale autonomous racecars. Therefore, this dataset presents a unique, high-speed version of seminal problems in autonomous driving such as localization, object detection and tracking, and mapping. We present these three problems and demonstrate the applicability of the RACECAR dataset to establish new benchmarks for these problems within the context of high-speed autonomous racing.

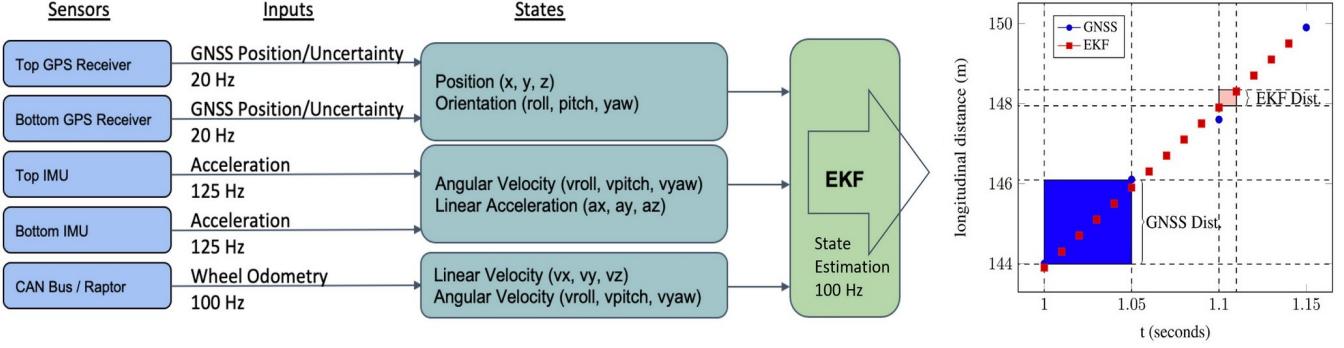


Fig. 4: [Left] Localization Pipeline, sensors produce state measurements which are used as updates for an Extended Kalman Filter to produce a vehicle Pose estimate. [Right] The blue dots represent GNSS updates at 20 Hz. The red dots are EKF predictions at 100 Hz. At approximately 40 m/s, the car travels 2 meters before another GNSS update.

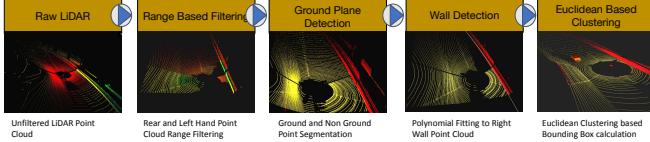


Fig. 5: LiDAR Perception: Processing raw LiDAR point clouds using Range Based Filtering, Ground Plane Detection, Wall Detection, and Euclidean Based Clustering

Birds Eye View (BEV)					
Model	Overall	0-20m	20-40m	40-60m	60m-Inf
PP AP	83.02	99.72	98.73	93.08	54.27
VR AP	75.82	97.05	90.39	83.60	40.71
3D Bounding Box					
PP AP	71.62	90.37	89.69	79.08	37.33
VR AP	63.31	86.17	85.85	65.62	23.95

TABLE II: Baseline AV-21 LiDAR Detections trained on LVMS Multi-Agent Data, using both Birds Eye View (BEV) labels and 3D Bounding Boxes. PP: PointPillars, VR: VoxelRCNN, AP: Average Precision

A. Benchmark 1: Localization

High precision and low latency localization is a key challenge of autonomous racing [14]. At a speed of 150 mph, the racecar travels up to 220 ft in one second. Since all of the racing took place outdoors and under clear sky conditions, the localization methodology adopted by several teams was mainly based on a fusion of the two GNSS signals and their IMU units using an Extended Kalman Filter [15] as shown in Figure 4. LiDAR-based [16] and camera-based [17] localization are also possibilities and the RACECAR dataset will enable such a comparison.

Using GNSS data alone results in a localization rate of only 20 Hz. While this may be fine for passenger autonomous driving, it is not adequate for autonomous racing. The AV-21 reports the rotation of the wheels at 100 Hz and accelerometer and gyroscope data at 125 Hz. This information is fused together with successive GPS readings and the pose estimate of the racecar $[x, y, z, \theta]$ is obtained at 100Hz. Here x, y, z corresponds to the Cartesian location of the car in world frame, and θ is the vehicle heading.

Figure 4[Right], shows the position estimates for the racecar traveling at 40 m/s (90 mph). A comparison between GNSS only and an EKF estimate is shown. It can be seen that at this speed the racecar can travel up to 2m blindly before another estimate of position is received via the GNSS. With an EKF running at a 5x faster rate, only 0.4m is traveled before another precise estimate is made resulting in more precise localization of the vehicle.

B. Benchmark 2: Object Detection and Tracking

Long-range and robust detection and tracking of opponents on the track is of paramount importance. At high speeds, overtaking another vehicle provides a very small window of time for the ego vehicle to react. A false negative detection could result in a collision between the attacker and the defender. Similarly, a false positive can cause erratic behavior by causing a vehicle to try and avoid a vehicle that is not there. In this section, we present three example approaches for object detection and tracking using the LiDAR, Cameras, and Radar. Ideally, one would fuse all the detections together into one cohesive detection but we present these separately to showcase that the RACECAR data can enable object detection challenges for each sensing modality as well as challenges for fused detection.

1) *LiDAR Based Object Detection and Tracking:* 3D point clouds obtained from the LiDAR are quite dense and noisy for object detection. Figure 5 shows an Euclidean clustering based pipeline for object detection. This method involves downsampling the point cloud, region of interest based filtering, ground-plane segmentation, and then an euclidean distance based clustering algorithm. This classical approach to object detection for point clouds is limited to a detection range of around 50 m.

Machine learning approaches to object detection are very popular but due to the requirement of annotated training data, exploration of auto-labeling methods, synthetic data genera-

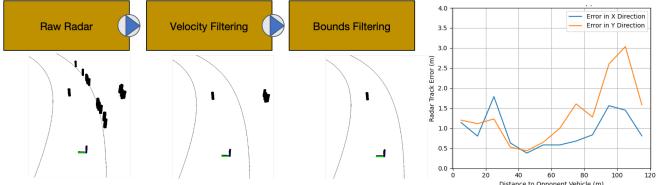


Fig. 6: [Left] Radar Filtering based on Track Boundaries and Velocity. [Right] Tracking Error when Compared to Ground Truth for an overtake between TUM Motorsopt and EuroRacing team.

tion, or hand labeling is necessary. With the RACECAR data, we hope deep learning approaches to racing problems can be further explored. We have provided results of a baseline implementation of both PointPillars [18] and VoxelRCNN [19], shown in Table II. These models were trained on two multi-agent runs within the RACECAR dataset, with labels generated from the GNSS/IMU information provided. Inference on point clouds using PointPillars resulted in maximum detections at distances up to 110 m. However, the average precision of detections dropped approximately 40% for distances over 60 m. For racing speeds of over 100 mph, 60 m detections provide a second of reaction time, and for computationally expensive planning algorithms, every additional moment counts. A benchmark challenge is to improve the detection range and reliability.

2) Radar Based Object Detection and Tracking: The front Electronically Scanning Radar (ESR) returns a list of tracked objects within its frame of view. Each object tracked is packaged with data concerning its angle, distance, forward velocity, and lateral velocity all with respect to the radar. Similar to the raw LiDAR data, at high speeds the Radar data is noisy, returning tracks for arbitrary points on the racetrack wall in addition to random objects nearby. One strategy for removing undesirable objects is to filter by velocity, dynamically adjusting based on ego speed, as well as a region of interest filter similar to the LiDAR. Figure 6 describes the error in both x and y directions between the Radar detected position and the ground truth position from a head-to-head racing scenario between TUM and EuroRacing.

3) Camera Based Object Detection and Tracking: On-board the AV-21 are six color, global shutter cameras. To maximize detection range, the front-facing two cameras utilize a narrower camera lens, as this improves far-field resolution. The remaining four cameras use a wider field of view to provide 360° coverage around the vehicle. Example images from the cameras can be seen in Figure 1.

Calibration: As part of the data set, full camera intrinsics and extrinsics are provided. Intrinsics were obtained using an off-the-shelf camera calibration package available in the ROS 2 ecosystem. Extrinsic were obtained using laser rangefinders and surveying equipment, measuring the sensor locations with respect to a fixed point on the vehicle.

Camera Object Detection Methods: Due to the higher resolution and higher frame rates, cameras have the potential to provide faster and longer-range detections of opponent



Fig. 7: Detections from YOLO on several camera views during a passing maneuver between MIT-PITT-RW and KAIST.

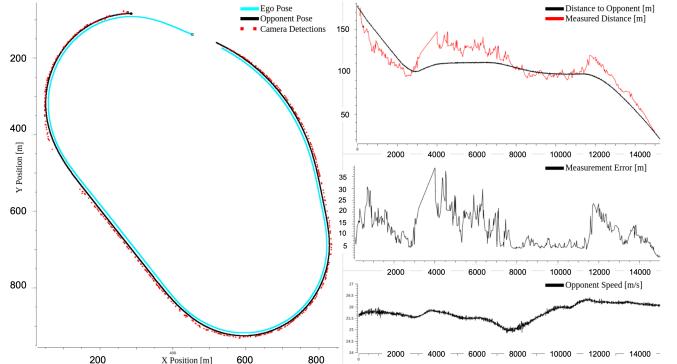


Fig. 8: (Left) RTK-GPS ground truth position of the ego vehicle and opponent, overlaid with camera detections. (Top Right) Distance from ego to the opponent. (Middle Right) Error between measured and ground truth. (Bottom Right) Speed profile of opponent

vehicles. However, camera detections are inherently noisier and more difficult to localize in 3D, due to depth ambiguity and projection error. A baseline utilizing YOLO v5 [20] was trained using a data set of other AV-21 vehicles. By exploiting the fact that the dimensions of the AV-21 are known, we can extend YOLO to report back real-time object depths by using a standard pinhole optics model. Detections from several camera views during a passing maneuver between MIT-PITT-RW and KAIST can be seen in Figure 7. The baseline was also compared to the GPS position of the opponent vehicle during a lap on the Las Vegas Motor Speedway track as shown in Figure 8.

C. Benchmark 3: Mapping

Relying on one method of localization can be risky, and in the case of having faulty or unreliable GNSS sensors, it can be useful to have an alternative solution. Implementing Simultaneous Localization and Mapping (SLAM) using 3D LiDAR point clouds or camera images for 2D visual SLAM, can provide another source of localization and an online method of mapping. Point cloud mapping aims to build a 3D point cloud map of an environment from sensor data that conveys 3D information about the surroundings of a perceiving agent, either directly like a LiDAR or indirectly as in the case of 2D visual SLAM.

The RACECAR dataset contains highly interesting data for implementing LiDAR SLAM algorithms, however several key challenges remain. Since three LiDARs are used, precise calibration is crucial to allow good scan matching. Bad calibration can be seen in Figure 9. High velocity leads to big

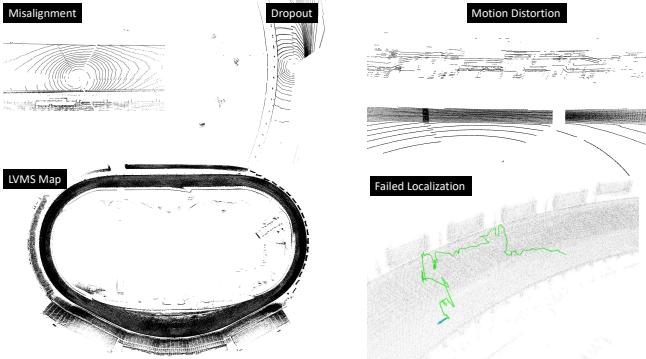


Fig. 9: Clockwise from Top Left: Faulty calibration of the LiDARs. Dropout of the rear right LiDAR. Motion distortion due to high velocity. Failed localization on a map. SLAM generated 3D point cloud map for LVMS.

jumps between the individual scans. This makes the SLAM more dependent on a good initial guess for the transformation of each frame. High velocity also leads to strong motion blur. This needs to be compensated before matching the point clouds. Figure 9 shows the effect of motion distortion on the tents behind the track barrier that are vertical in reality. Due to the blur, they appear sheared in the point cloud. The banked turns make it impossible to assume a flat ground. Available SLAM algorithms tend to create spiral maps. A new handling of the ground surface has to be implemented. **Map based Localization:** To avoid these problems, map-based localization approaches can be realized. Online, the current scan can be matched with the offline-generated map. However, this is still a difficult task as the current scans have to be corrected first, and good state estimation is necessary to provide initial guesses for scan matching. Existing SLAM packages fail to provide a precise and robust localization output on the RACECAR dataset. Figure 9 shows a failed SLAM attempt at the LVMS track. Robust SLAM at high speeds (100+ mph) is still a challenge for autonomous racing.

VII. CONCLUSION

The paper presents the RACECAR dataset for high-speed autonomous racing. Multi-modal sensor data has been collected, processed, and converted into ROS2 bag files and nuScenes format for wider accessibility. The data was collected from AV-21 autonomous Indy Light racecars during the 2021-22 Indy Autonomous Challenge held at the Indianapolis Motor Speedway and at the Las Vegas Motor Speedway tracks which witnesses overtaking at speeds of 170 mph. There are 27 racing sessions spanning 11 racing scenarios and over 6.5 hours of data that have been provided. We provide ground truth locations for the vehicle which enables use of the data for several benchmark problems in autonomous racing - localization, object detection and tracking, and mapping. Baseline algorithms are presented for each benchmark. The dataset, accompanying processing scripts, and the ROS2 to NuScenes conversion library are all open source.

REFERENCES

- [1] M. O’Kelly, V. Sukhil, H. Abbas, J. Harkins, C. Kao, Y. V. Pant, R. Mangharam, D. Agarwal, M. Behl, P. Burgio, *et al.*, “F1/10: An open-source autonomous cyber-physical platform,” *arXiv preprint arXiv:1901.08567*, 2019.
- [2] V. S. Babu and M. Behl, “f1tent. dev-an open-source ros based f1/10 autonomous racing simulator,” in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pp. 1614–1620, IEEE, 2020.
- [3] “Indy autonomous challenge.” url=<https://www.indyautonomouschallenge.com/>, journal=Indy Autonomous Challenge.
- [4] A. Wischnewski, M. Geisslinger, J. Betz, T. Betz, F. Fent, A. Heilmeier, L. Hermansdorfer, T. Herrmann, S. Huch, P. Karle, *et al.*, “Indy autonomous challenge-autonomous race cars at the handling limits,” in *12th International Munich Chassis Symposium 2021*, pp. 163–182, Springer, 2022.
- [5] G. Hartmann, Z. Shiller, and A. Azaria, “Autonomous head-to-head racing in the indy autonomous challenge simulation race,” *arXiv preprint arXiv:2109.05455*, 2021.
- [6] J. Kabzan, M. I. Valls, V. J. Reijgwart, H. F. Hendrikx, C. Ehmke, M. Prajapat, A. Bühler, N. Gosala, M. Gupta, R. Sivanesan, *et al.*, “Amz driverless: The full autonomous racing system,” *Journal of Field Robotics*, vol. 37, no. 7, pp. 1267–1294, 2020.
- [7] M. Zeilinger, R. Hauk, M. Bader, and A. Hofmann, “Design of an autonomous race car for the formula student driverless (fsd),” in *Oagm & Arw Joint Workshop*, 2017.
- [8] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, “Autonomous vehicles on the edge: A survey on autonomous vehicle racing,” *IEEE Open Journal of Intelligent Transportation Systems*, 2022.
- [9] N. Vödisch, D. Dodel, and M. Schötz, “Fsoco: The formula student objects in context dataset,” *SAE International Journal of Connected and Automated Vehicles*, vol. 5, no. 12-05-01-0003, 2022.
- [10] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.
- [11] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, “Scalability in perception for autonomous driving: Waymo open dataset,” *CoRR*, vol. abs/1912.04838, 2019.
- [12] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet, “Level 5 perception dataset 2020.” <https://level-5.global/level5/data/>, 2019.
- [13] H. Caeser, V. Bankiti, A. H. Lang, S. Vora, V. E. Lioni, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “nuscenes: A multimodal dataset for autonomous driving,” in *CVPR*, 2020.
- [14] K. Lingemann, A. Nüchter, J. Hertzberg, and H. Surmann, “High-speed laser localization for mobile robots,” *Robotics and autonomous systems*, vol. 51, no. 4, pp. 275–296, 2005.
- [15] R. J. O. R. E. Kopp, “Linear regression applied to system identification for adaptive control systems,” vol. 1, pp. 2743 – 2748 vol.3, 10 1963.
- [16] X. Chen, I. Vizzo, T. Läbe, J. Behley, and C. Stachniss, “Range image-based lidar localization for autonomous vehicles,” *CoRR*, vol. abs/2105.12121, 2021.
- [17] Y. Wu, “Image based camera localization: an overview,” *CoRR*, vol. abs/1610.03660, 2016.
- [18] A. H. Lang, S. Vora, H. Caeser, L. Zhou, J. Yang, and O. Beijbom, “Pointpillars: Fast encoders for object detection from point clouds,” pp. 12689–12697, 2019.
- [19] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, “Voxel r-cnn: Towards high performance voxel-based 3d object detection,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 1201–1209, May 2021.
- [20] G. Jocher, A. Stoken, J. Borovec, NanoCode012, Christopher-STAN, L. Changyu, Laughing, tkianai, A. Hogan, lorenzomammana, yxNONG, AlexWang1900, L. Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, F. Ingham, Frederik, Guilhen, Hatovix, J. Poznanski, J. Fang, L. Yu, changyu98, M. Wang, N. Gupta, O. Akhtar, PetrDvoracek, and P. Rai, “ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements,” Oct. 2020.

APPENDIX

A. Complete Sensor Specifications

The data set was captured from a suite of sensors chosen to help teams run a full autonomous driving stack. The racecar was designed with the entire driver cockpit removed and replaced with a computer, drive-by-wire system, and sensors.

1) *LiDARs*: The car is outfitted with three Luminar H3 Solid State LiDARs. The LiDAR's output raw 3D point clouds comprised of x, y, z coordinates, and intensity for each laser scan. The rate at which these messages are published is dependent on the density of the point cloud, with more laser scans producing a higher density point cloud but a lower frame rate. The scan pattern of the LiDARs can also be configured in Uniform, Trapezoidal, or Gaussian distributions allowing higher point cloud density in focused areas. As seen in the bottom right quadrant of Figure 10, the three LiDAR's field of view covers 360 deg, with a detection range of 250 m.

2) *Global Navigation Satellite System (GNSS)*: GNSS and Inertial Measurement Unit (IMU) information were collected using two Novatel Pwrpk 7d Receivers and was the primary source of localization for the vehicles. There exist two receivers for the purpose of redundancy, and as they are stacked on each other within the vehicle they are referred to as the top and bottom receivers. The receiver uses two symmetrically placed antennas on the AV21 seen in the top left of Figure 10. Both receivers were configured with a Real Time Kinematic (RTK) system to provide positioning information with up to centimeter level accuracy. The on-board IMUs provide angular velocity and acceleration data, which help drive an Inertial Navigation System (INS) to dead reckon position, velocity, and heading. The raw data provided here includes latitude, longitude, altitude, and their respective standard deviations from each receiver. Also included are the acceleration and angular velocity collected from each IMU.

3) *Cameras*: Six Allied Vision Mak G319C Cameras were installed on the vehicle. The cameras resolution, frame rate, focal length, and optical center can be configured, and output raw uncompressed images. These are uncompressed images and require further processing for translation into usable point clouds. The cameras possess functionality for Precision Time Protocol and allow device synchronization in the order of microseconds.

4) *Radar*: The AV-21 has both long range and medium range Aptiv RADAR devices installed. An electronically scanning radar is placed for long range detection on the front of the vehicle, and two mid range radars are placed on the side of the vehicle. Raw radar data is interfaced through the CANBus and after processing from a driver, provides detected vehicle speed, a covariance matrix, and relative velocity of detected objects. The two radars range and field of view is described in Table IV and can be seen in Figure 10.

5) *Drive-by-wire*: The car was outfitted with a New Eagle Raptor drive-by-wire system to provide an electronic control interface for the vehicle actuators. The computing stack

on the onboard computer provided steering angle, throttle input, brake input, and gear shifting commands to the Raptor interface. Battery voltage, engine temperature, engine RPM, current gear, and the wheel speed of the vehicle were all consistently monitored and assisted in control of the vehicle.

6) *Communication*: Each car used a Cisco Ultra Reliable Wide Band radio to connect to a track-wide mesh network. This network provided an internet connection for GNSS RTK corrections, as well as connection to a basestation computer used for live observation of the vehicle telemetry. The AV21 also communicated with race control, racetrack supervisors providing command flags indicating when cars should stop, slow down, or return to the pits. Race control sent flags to the AV21s using MyLaps, a popular sports timing system which also provided vehicle telemetry to race control.

7) *Onboard Computing*: Onboard computing of the car was handled by an ADLINK AVA-3501. The computer possessed a Intel Xeon processor, 64 GB of RAM, 3 TB of storage, and a Quadro RTX 8000 GPU. The computer was intended as a platform to run every component of the self driving computing stack, including any potential machine learning methodologies. Each computer was setup with Ubuntu 20.04 and ran ROS2 to interface with each component of the vehicle. All the sensors, as well as the drive-by-wire system used ROS2 drivers to provide a consistent common interface.

8) *Chassis*: The vehicle chassis was manufactured by Dallara. The car resembles an AV-21 Indy Lights race car, but has been retrofitted to accommodate the various sensors and computing platforms described. All of the cockpit and safety features for a human driver were removed and replaced with a platform to house all the electronic components.

B. Extended Kalman Filter

Formulating and processing the EKF algorithm is a well known process, to estimate the full 3D (6 Degrees of Freedom) pose and velocity of a robot over time.

The first step is to represent one's process using a nonlinear dynamic system.

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1} \quad (1)$$

\mathbf{x}_k represents the robot's current 3D pose at time k, f is the state transition function, and \mathbf{w}_{k-1} is added noise towards the process. The vector x contains the robot's 3D pose, 3D orientation, and their derivatives.

$$\hat{\mathbf{x}}_k = f(\mathbf{x}_{k-1}) \quad (2)$$

$$\hat{\mathbf{P}}_k = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^T + \mathbf{Q} \quad (3)$$

The algorithm begins with a prediction of the future state, using the current state estimate, the transition function, and estimated covariance. \mathbf{P} is the estimated covariance, \mathbf{F} is the Jacobian of the non linear transition function, and \mathbf{Q} is the process noise covariance that perturbs the system. The estimated covariance can be initialized to some reasonable

Scenario (Teams)	GNSS	LiDAR	RADAR	Camera
$S_1(C, M, P)$	51,486	58,375	9,924	157,146
$S_2(C, M, K)$	79,748	60,020	1,148,205	301,065
$S_3(M, E)$	52,194	44,855	31,444	387,756
$S_4(T, E)$	23,852	23,722	15,725	0
$S_5(C, M, P, T, E, K)$	129,914	118,156	23,525	122,204
$S_6(T, E, P)$	52,504	67,253	35,023	0
$S_7(C, K)$	16,596	8,025	0	0
$S_8(C, K)$	22,140	0	0	270,648
$S_9(T, E)$	44,510	40,569	23,129	0
$S_{10}(P)$	13,153	12,476	0	0
$S_{11}(T, K)$	22,594	29,746	6,440	278,009

TABLE III: Dataset Content: Raw ROS2 message frame counts for each sensor

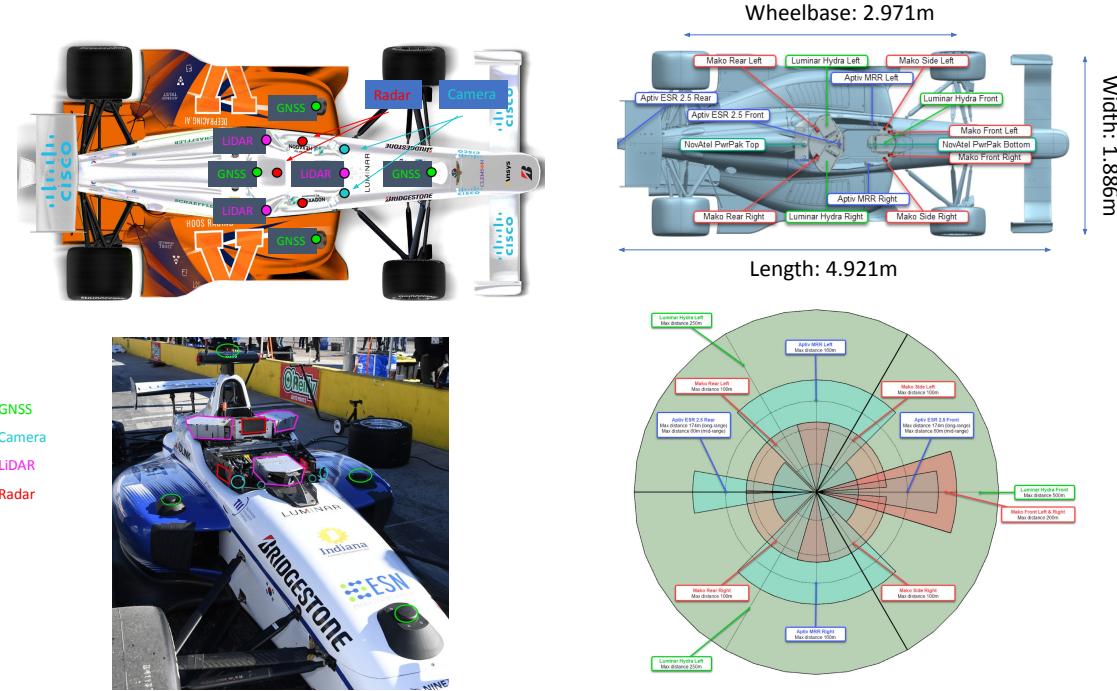


Fig. 10: Top Left: Rendering of the racecar with visible sensors highlighted. Top Right: Vehicle diagram showcasing the length, width, and wheelbase of the vehicle. Bottom Left: Inside view of cockpit with various sensors highlighted. Bottom Right: Sensor ranges for LiDAR, Radar, and Cameras are overlayed on top of the vehicle.

value, but the process noise covariance should represent the uncertainty between the real life system model, and the approximate predictions made by the transition function. Tuning these values carefully requires experimentation and iteration, and should be kept at a relatively small value initially.

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \quad (4)$$

Sensor measurements \mathbf{z}_k are received at time k , h is a sensor model that correctly maps measurement inputs into a state space vector, and \mathbf{v}_k is the measurement noise. These sensor measurements can be any state measurement from the IMU, wheels, GNSS measurements, or LiDAR laser scans. The measurement noise can be approximated from testing the hardware directly.

$$\mathbf{K} = \hat{\mathbf{P}}_k \mathbf{H}^T (\mathbf{H} \hat{\mathbf{P}}_k \mathbf{H}^T + \mathbf{R})^{-1} \quad (5)$$

$$\mathbf{x}_k = \hat{\mathbf{x}}_k + \mathbf{K}(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}_k) \quad (6)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}\mathbf{H})\hat{\mathbf{P}}_k(\mathbf{I} - \mathbf{K}\mathbf{H})^T + \mathbf{K}\mathbf{R}\mathbf{K}^T \quad (7)$$

After the prediction step and a measurement update is received, the estimated state and it's associated covariance is updated in a correction step. \mathbf{K} represents the Kalman gain, which is calculated from the measurement covariance \mathbf{R} and the estimated covariance \mathbf{P} . This gain is used to update the state vector and it's covariance matrix. After the correction step, another prediction step is made, and the process repeats.

AV21 Specifications	
GNSS	PwrPak7 E1
GNSS Positional Accuracy (L1)	1.5m
GNSS Positional Accuracy (RTK)	1cm + 1ppm
GNSS Data Rate	0-20 Hz
IMU Data Rate	125 Hz
LiDAR	H3 Prototype
Range (10% Reflectivity)	250m
Horizontal Field of View	120°
Configurable Vertical Field of View	0-30°
Frames per Second	10-30
Front Radar	Front Facing Aptiv ESR
Long Range Detection	174m
Mid Range Detection	60m
Long Range FOV	±10 deg
Mid Range FOV	±45 deg
Update Rate	20 Hz
Side Radar	Side Facing Aptiv MRR
Long Range Detection	160m
Mid Range Detection	40m
Horizontal Field of View	90 deg
Vertical Field of View	5 deg
Camera	Mako G-319
Max Resolution	2064 x 1544
Max frame rate at full resolution	37.6 fps
Spectral range	300-1100 nm
Chassis	Dallara AV-21
Overall Length	192 in / 4876 mm
Overall Width	76 in / 1930 mm
Overall Height	45.5 in / 1156.5 mm
Wheelbase	117 in / 2971 mm
Weight	1600 lbs / 726 kgs

TABLE IV: AV21 - Sensor Specifications