AI-based Autonomous Driving

How we won RDW's Self Driving Challenge 2024

Edwin van den Oetelaar, Sieuwe Elferink, Teade Punter
SDC Team Fontys, High Tech and Embedded Systems (HTES) Research Group,
Fontys University of Applied Sciences, ICT

Self driving cars might help to change our society in several ways. One might be that traffic safety improves by reducing the number of accidents. Self driving cars might also enable disabled people to drive safely without being dependent on others which might help to improve their quality of life. Self driving cars might also enable efficient and eco-friendly traffic.

The Dutch national road transport authority RDW organizes the Self Driving Challenge (SDC). It is a yearly event aimed at gaining knowledge about complex technology for autonomous decision-making in a car. Each year, a goal is set for the challenge and participating student teams are tasked with developing software for their vehicle to best achieve this goal. The SDC is a competition for Dutch universities and their students to build a self-driving solution for a car-like vehicle. The vehicle must navigate a designated track, follow road signs, stop at traffic lights, and park in marked spaces. The challenge has two categories: *Closed*, in which participants use an RDW-provided electric kart with a standard Intel i5 computer, and *Open* in which the teams could come up with their own solution. Team Fontys won the SDC open category in 2024. The open category enabled us to develop our own vehicle with custom hardware and software. In this way we wanted to explore the possibilities of Machine Learning (ML) -based software development by our students. The SDC "Open" category enabled us to focus on machine learning-based solutions rather than classical algorithms, allowing for greater freedom in experimentation.
Expecting high processing demands, we chose the NVIDIA Orin platform with CUDA acceleration [20] running on Linux, instead of the standard i5 setup.

We designed a custom Quad to better explore our own electronics and mechanical systems. This paper reports our design and our experiences when developing our car.

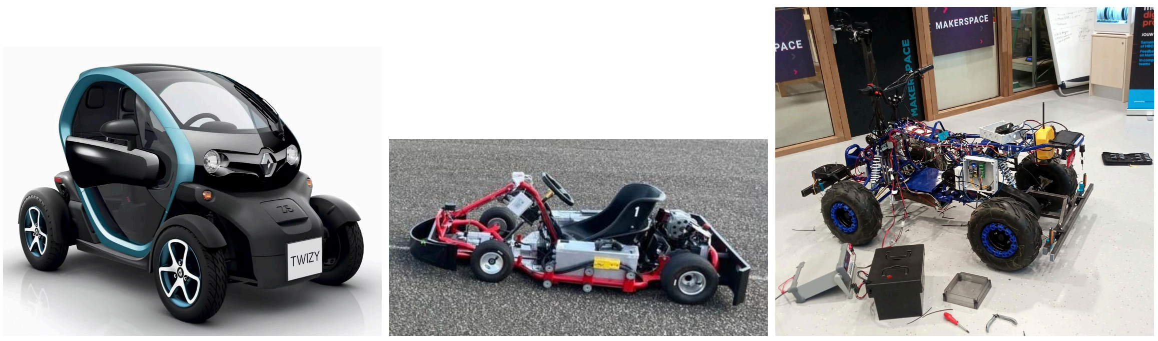
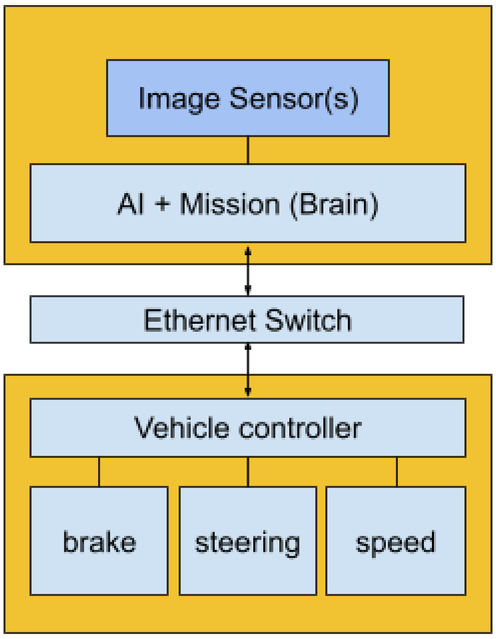


Figure 1 - Vehicles in 2024 challenge: Twizzy (open) , RDW kart (closed), Fontys Quad (open)



We have split our system and development group in 2 parts : **”vehicle controller”** (speed, steering, odometry low latency 1000 Hz control loop) and **“the brain**” (running mission control, planning, and sensory processing, independent in slower control loop) connected using a reliable wired 100Mbit ethernet our version of **“spinal cord”.**

Our SDC vehicle-control posed four main challenges: steering, speed control, braking and communication.

The “**Vehicle Controller**” was based on ESP32, connected through **wired 100Mbit Ethernet**. For signal integrity, we created opto-isolation PCB’s. Additionally, we implemented a PID control system within the motor control loop and steering control loop and integrated a protocol stack that supports fail-safe mechanisms, ensuring consistent and reliable operation. **Steering** posed the biggest engineering challenge, leading us to test multiple solutions. Early attempts included rotating servos, linear actuators, and even a power steering system from a Suzuki Alto, but these were either too weak or too slow. Finally, we developed a high-torque motor system with an integrated gearbox, using a sturdy metal gear directly mounted onto the steering column. This provided the necessary power and response needed to control the vehicle. To ensure precise steering control, we integrated a MAQ473 **Hall-based absolute angle encoder** [22] and added mechanical end-stops with **limit switches** that cut motor power to prevent over-rotation and potential damage. **Speed control** was managed with a galvanically isolated PWM to 0-5V output circuit that we designed and built ourselves.

For **communication**, we combined Wired 100MBit Ethernet (with isolation transformers) with the open source **IP/Zenoh protocol** [21] to support autonomous control and incorporated a 6-channel PPM-based RC input for manual overrides.

We integrated a **certified RF emergency stop (E-stop) system**, designed to instantly cut motor power at the hardware level. This was a crucial safety measure, ensuring immediate shutdown in case of unexpected behavior. Autonomous driving demands strict safety controls, and the challenge organizers mandated an emergency stop system as a requirement for participation.

We chose to handle perception using a **single ZED-X stereo depth camera** [3] rather than LiDAR or RADAR. Our approach is AI-driven, and current deep-learning models are more readily applied to camera data, making this the most practical choice. LiDAR adds complexity and cost, while our setup remains simple and low-cost—key principles in our design. The main computing platform is an **NVIDIA Jetson Orin AGX** [20], providing the necessary processing power for real-time vision and decision-making.

Mechanical modifications

We reinforced the vehicle with welded metal bumpers on the front and back, equipped with collision detection switches.

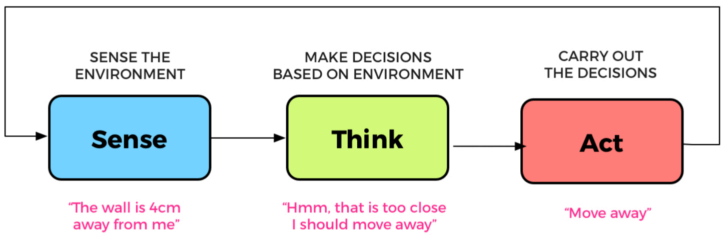
Power came from the original 48V DC lead-acid battery pack, from which we generated all necessary voltages using isolated DC/DC converters (24V for steering, 12V for the Orin, and 5V for microcontrollers).

An electrically actuated hydraulic brake was designed and added but was damaged during testing and therefore not operational during the challenge. We resorted to an electric braking approach, we shorted the motor windings of the BLDC motor to come to instant stop.

The hardware, electronics, and software were developed in parallel, requiring careful coordination of APIs to enable integration. Team members relied on each other to complete their parts, with changes and updates implemented across the entire system right up until the final days before the challenge.

Navigation system

The navigation system is responsible for finding the best vehicle controls in order to optimize given goals. For the SDC these goals are the challenges that were given by the RDW. A goal is for example following the road surface or stopping safely for a traffic light. To find these optimal vehicle controls we opted to use the **Sense, Think, Act architecture,** although we are aware of the drawbacks [19]. This commonly used architecture has been deployed in a wide variety of related robotics research like for self driving as shown in [1] [2]. This architecture separates the task of navigation into three distinct tasks. The Sense task is responsible for **capturing information** of the vehicles surrounding. This can be the distance to a vehicle or the location of the road surface. This information can be collected from a range of sensors on or outside the vehicle like cameras or map data. The Think task is responsible for calculating the **best path** in order to optimize the navigation system goals. The Act task is responsible for following the best possible path by **physically controlling** the vehicle's controls like the steering wheel or speed pedal. Combined these modules form a navigation pipeline that is processed in a continuous loop. We achieved a combined pipeline speed of 0.2s resulting in a 5hz update frequency.


SENSE THE ENVIRONMENT
MAKE DECISIONS BASED ON ENVIRONMENT
CARRY OUT THE DECISIONS
Sense
Think
Act
"The wall is 4cm away from me"
"Hmm, that is too close I should move away"
"Move away"

Sense, Think, Act overview

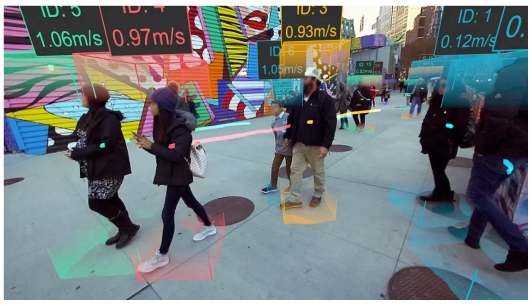
Sense

The sense task begins by capturing data on the vehicle's surroundings using the ZED X **stereo vision** system. This system uses two cameras for stereovision, achieving depth perception with <2 cm accuracy in bright outdoor settings [3]. It also provides 2K RGB imaging from the left camera, accessible through the ZED SDK, with depth processing handled on the Jetson Orin device. To avoid bottlenecks and simplify the system, we chose not to include additional sensors like: extra cameras, Lidar or wheel odometry.

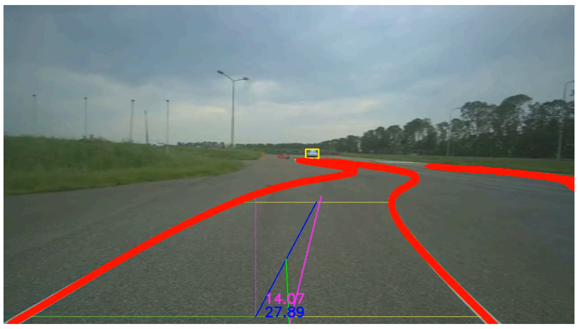
The image data from the ZED X camera is analyzed using a set of parallel running detectors in order to extract autonomy related information from the raw image data. These detectors are all based on Machine Learing (ML) because ML can adapt well to a wide range of external factors like shadows and cloud cover which change the perceived image [16]. Traditional Computer Vision based techniques focus on analyzing an image without context which results in failure when external factors can not be controlled like in the SDC.

Pedestrian

Pedestrian detection has been achieved by using the pedestrian detector and tracked build into the ZED SDK. This uses a ML based model trained by stereolabs that uses a combination of depth and rgb images to provide full pose detection of people. This allows extraction of location and distance of pedestrians.



Vehicle + Lane



Vehicle and lane detection were in the original system overview as shown in [Appendix 1] separate modules. However during implementation we found a model named **YOLOPv2** that combined the two tasks into a single ML model that achieves state of the art results [4]. This model uses the popular YOLO architecture for its encoder backbone. The model differentiates itself by combining multiple tasks into a single model. This approach is named a **hydranet** which combines multiple decoders with the same embedding space named “heads” to perform different tasks like road segmentation, line detection and obstacle detection. Hydrants are a common trend in the field of self driving with notably Tesla focusing heavily on them with their new “End-2-End” approach [5].

In the right image the red lines are the lane output of the YOLOPv2 model. The yellow box shows the vehicle detection in action. This yellow box represents the (xy) location in the image frame. However the distance to the vehicle (z) is also needed for safe navigation. To retrieve the distance the bounding box coordinates of the ML detection in the RGB image are used as a **mask** in the **depth image** from the ZED X sensor. Using this mask the (z) coordinates are extracted to retrieve a distance. Using the RGB coordinates as a mask is possible because both the Depth and RGB image of the ZED X have the same coordinate system and origin.

Traffic Light

The traffic light detection is performed using a **YOLOv8** ML model from [8]. This model is able to detect the (xy) location and state of the traffic light. Distance to the traffic light (z) was achieved using a bounding box shaped mask on top of the depth map. The model itself provided raw detections. However keeping track of a traffic light over time in different frames requires a tracker. A **KCF** (Kernelized Correlation Filters) tracker was used to provide each detected traffic light an ID which is necessary for accurate planning.

Crosswalk

Finding a suitable crosswalk detection model was challenging. Instead of this, a **custom model** was trained based on the YOLOv8 architecture with the ultralytics library [6]. A dataset was created using test videos from the track. Creating a dataset from images from one specific location is usually bad practice since it leads to overfitting [7]. However, specifically for the SDC, **overfitting** was found a valid method of creating a robust detector with limited resources since the vehicle only needs to ride one specific location. Distance to the crosswalk (z) was achieved using a bounding box shaped mask on top of the depth map

Traffic Sign

Traffic sign detection was achieved using another **YOLOv8** model from [8]. This model was trained on a large dataset of **European traffic signs**. It is able to provide (xy) location and sign class. For example it can differentiate between a 10km/h and 20km/h sign. Distance to the traffic sign (z) was achieved using a bounding box shaped mask on top of the depth map.

Think

The Think task receives information like location of objects of interest like traffic lights or other vehicles to make decisions. Originally a occupancy grid planner was developed by combining all the information from the sense model into a "world model". However, creating an accurate **world model** proved to be challenging with our small team and limited time. Because of this a more simplistic planner has been used that makes decisions on a per frame basis without incorporating past or future information. The system works by combining a state machine that defines high level behavior with a center line tracker to stay on a drivable surface.

The **center line tracke**r works by first finding the lane boundaries by scanning from the center outward and then calculates the midpoint between these boundaries. It uses the midpoint to determine an angle between the vehicle's center and the lane's center. This angle is then adjusted based on lane position and the field of view (FOV), helping to estimate the correct **steering angle** for staying centered in the lane.

The **state machine** defines high level behavior by defining states like "AV_DRIVING", "AV_WAITING" and "AV_CROSSWALK". Each state and state change has a corresponding **behavior**. For example "AV_DRIVING" keeps a constant speed given by the traffic sign detector and uses the center line tracker to stay on the road. Transitioning from "AV_DRIVING" to "AV_CROSSWALK" requires braking to reduce the speed to 0 km/h. In "AV_CROSSWALK" the vehicle waits until the pedestrian walks away. After this the vehicle transitions again to "AV_DRIVING" which requires throttle to get back up to the speed limit given by the traffic sign detector. The state machine incorporates **6 different states** with transitions between them to define the vehicle's behavior in the different SDC challenges.

Act

The Think task provides desired vehicle controls in the form of a desired speed and steering angle in order to safely accomplish the navigation systems goals. The Act task actually performs these desired controls on the **physical vehicle**. This responsibility is completely offloaded to the low level vehicle controller. The navigation system sends the desired speed and steering angle over the **Zenoh** protocol to the low level vehicle controller which performs the desired actions and reports the current status of the vehicle.

Evaluation

Some of the system's safety and low level logic components were individually tested and validated using the CARLA [12] simulator. After successful testing the combined system was evaluated during the SDC finale on the RDW Lelystad test track. The vehicle performed well in most of the given challenges resulting in it achieving first place. Functions like remote control (RC) helped to quickly reposition the vehicle after a failed attempt. Also the state machine demonstrated the correct behavior within each state and state transition.
However the system also showed some limitations, we discuss them below.

Issue 1- The ML based detectors of the sense task sometimes gave **false** positive or false negative **results**. Especially the audience surrounding the track during the finals, which were not attending in the tests, proved to be tricky. The stoplight detector for example gave false red light detections based on red clothing that spectators were wearing. This resulted in false state transitions like not going into the AV_DRIVING state when the traffic light turned green. Because of reliability concerns the traffic sign detector was not used and instead a speed of 15 km/h on straights and a speed of 5km/h in turns was used.

Issue 2- Some of the ML detectors had a **high inference time** (response time) [17] resulting in a slow update speed. The traffic light detector especially proved to be slow. This resulted in discontinuous control which caused problems like not staying in a lane.

Issue 3- The simplistic planning system, using a centerline tracker, was not able to stay on course in more **complex situations.** It performed well in the first section of the SDC course which was a straight section with two slight turns. However, when the course split into two separate lanes combined with a turn the vehicle was slowing down and speeding up on random intervals and eventually failed to stay within the lanes. Also multiple **failures** to stop for the traffic light with the correct distance to the road marking also were caused by a simplistic preprogrammed ramp down function.

Issue 4- The remote wireless safety system did not have **sufficient range**. This resulted in premature stopping of the vehicle.

Issue 5- The centerline tracker and lane detector were sensitive to the **location of the camera** on the vehicle. This needed to be changed several times resulting in a non optimal mounting of the ZED X.

Recommendations

The current system offers a solid basis showcased by **winning** the 2024 SDC open category. Based on testing data and results of the challenge day itself we have a list of recommendations for future SDC editions. We expect that these recommendations will help us to improve our system.

Current system

Problem 1 and problem 2 described in the evaluation section are both related to performance limitations of the ML detectors. Response time might be solved by using newer model architectures that use **‘smaller’** models to achieve similar accuracy. False detections can be improved by fine tuning the existing models on new and **more representative data**.

The above mentioned updates might improve the overall navigation system, but we expect more progress by improving the current **per frame** planning system of the Think-task mentioned in problem 3. By incorporating temporal information into a **world model** smoother and more concise control will be possible. Also the world model can be used to filter false detections which are inherent to any ML detector [18].

An **occupancy grid** based planning system can be used which works by fusing all the information coming from the Sense module into a single world model. This model is projected into a grid in which each cell has a cost value. A planning algorithm like A* plans a trajectory with the lowest cost value to optimize navigation goals. An occupancy grid would improve the navigation system by using past and current information to make current and future decisions instead of making a decision per time slice. This will allow more **robust navigation** in challenging environments.

Problem 4 could be solved by replacing the emergency stop trigger system with a **better range** version of at least 400m. Camera mounting (issue 5) can be solved with a new camera mount system that can be adjusted in the vehicle's driving direction. Problem 6 can be solved by improving the overall hardware packaging by using **waterproof** enclosures for electronics and improved wire routing by cutting wires to precise lengths.

New systems

Key players in the industry like Tesla are transitioning more towards replacing rule based decisions with **learning based** methods. Work like [10, 11] replaces the entire Sense task with a single hydranet ML model to create a robust world model and occupancy grid. Also End-2-End methods that replace the entire Sense, Think, Act architecture with a single **hydranet** ML model like those from NVIDIA have shown to perform reliable autonomous driving in a wide range of scenarios [13, 14, 15]. Future SDC teams can research these new industry trends to drastically improve the current explicit architecture.

Conclusion / Acknowledgments

With this paper we provided a set of ingredients that are needed to have a reasonable chance for success. Nevertheless the team and their expertise and dedication are important for real success.

Sources

[1] Claussmann, L., Revilloud, M., Gruyer, D., & Glaser, S. (2019). A review of motion planning for highway autonomous driving. *IEEE Transactions on Intelligent Transportation Systems, PP*(99), 1–23. https://doi.org/10.1109/TITS.2019.2913998

[2] Pendleton, S. D., Andersen, H., Du, X., Shen, X., Meghjani, M., Eng, Y. H., Rus, D., & Ang, M. H., Jr. (2017). Perception, planning, control, and coordination for autonomous vehicles. *Machines, 5*(1), Article 6. https://doi.org/10.3390/machines5010006

[3] Stereolabs Inc. (2024). *ZED X: Industrial stereo camera for spatial perception*. Retrieved from https://www.stereolabs.com/en-nl/products/zed-x

[4] Han, C., Zhao, Q., Zhang, S., Chen, Y., Zhang, Z., & Yuan, J. (2022). YOLOPv2: Better, faster, stronger for panoptic driving perception. *arXiv preprint* arXiv:2208.11434. https://doi.org/10.48550/arXiv.2208.11434

[5] Unknown Author. (2021, July 1). *Tesla's HydraNet - How Tesla's Autopilot works.* Retrieved from https://www.thinkautonomous.ai/blog/how-tesla-autopilot-works/

[6] Ultralytics. (n.d.). *Ultralytics: AI vision for everyone.* Retrieved (2024, October 10), from https://ultralytics.com

[7] Xue, Y. (2019). An overview of overfitting and its solutions. *Journal of Physics: Conference Series, 1168*(2), 022022. https://doi.org/10.1088/1742-6596/1168/2/022022

[8] Syazvinski. (n.d.). *Traffic Light Detection and Color Classification Using Yolo v8*. GitHub. Retrieved [2024, mar 10], from https://github.com/Syazvinski/Traffic-Light-Detection-Color-Classification

[9] Flucus. (n.d.). *Traffic Sign Detection AI*. GitHub. Retrieved [2014, april 15], from https://github.com/Flucus/Traffic-Sign-Detection-AI

[10] Xu, H., Chen, J., Meng, S., Wang, Y., & Chau, L.-P. (2024). A survey on occupancy perception for autonomous driving: The information fusion perspective. *arXiv*. https://arxiv.org/abs/2405.05173 and https://github.com/HuaiyuanXu/3D-Occupancy-Perception

[11]OpenDriveLab. (n.d.). *OccNet: Scene as Occupancy [Code]*. GitHub. Retrieved (2024, october 8), from https://github.com/OpenDriveLab/OccNet

[12] Dosovitskiy, A., Ros, G., Codevilla, F., López, A., & Koltun, V. (2017). CARLA: An open urban driving simulator. *arXiv*. https://doi.org/10.48550/arXiv.1711.03938

[13] Zhou, H., Laval, J., Zhou, A., Wang, Y., Wu, W., Qing, Z., & Peeta, S. (2021). Review of learning-based longitudinal motion planning for autonomous vehicles: Research gaps between self-driving and traffic congestion. *arXiv*. https://doi.org/10.48550/arXiv.1910.06070

[14] Li, Z., Li, K., Wang, S., Lan, S., Yu, Z., Ji, Y., Li, Z., Zhu, Z., Kautz, J., Wu, Z., Jiang, Y.-G., & Alvarez, J. M. (2024). Hydra-MDP: End-to-end multimodal planning with multi-target Hydra-distillation. *arXiv*. https://doi.org/10.48550/arXiv.2406.06978

[15] Singh, K. (2024, September 30). *Tesla FSD V12.5.5 adds end-to-end highway stack; Tesla teases upcoming features*. Retrieved from https://www.notateslaapp.com/news/2284/tesla-fsd-v1255-adds-end-to-end-highway-stack-tesla-teases-upcoming-features

[16] O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Velasco-Hernandez, G., Krpalkova, L., Riordan, D., & Walsh, J. (2019). Deep learning vs. traditional computer vision. *arXiv*. https://doi.org/10.48550/arXiv.1910.13796

[17]Hazelcast. (n.d.). What is machine learning inference? *Hazelcast*. Retrieved [2024, dec 19], from https://hazelcast.com/foundations/ai-machine-learning/machine-learning-inference/

[18] Sapkota, R., Meng, Z., Churuvija, M., Du, X., Ma, Z., & Karkee, M. (2024). Comprehensive performance evaluation of YOLO11, YOLOv10, YOLOv9, and YOLOv8 on detecting and counting fruitlet in complex orchard environments. *arXiv*. https://doi.org/10.48550/arXiv.2407.12040

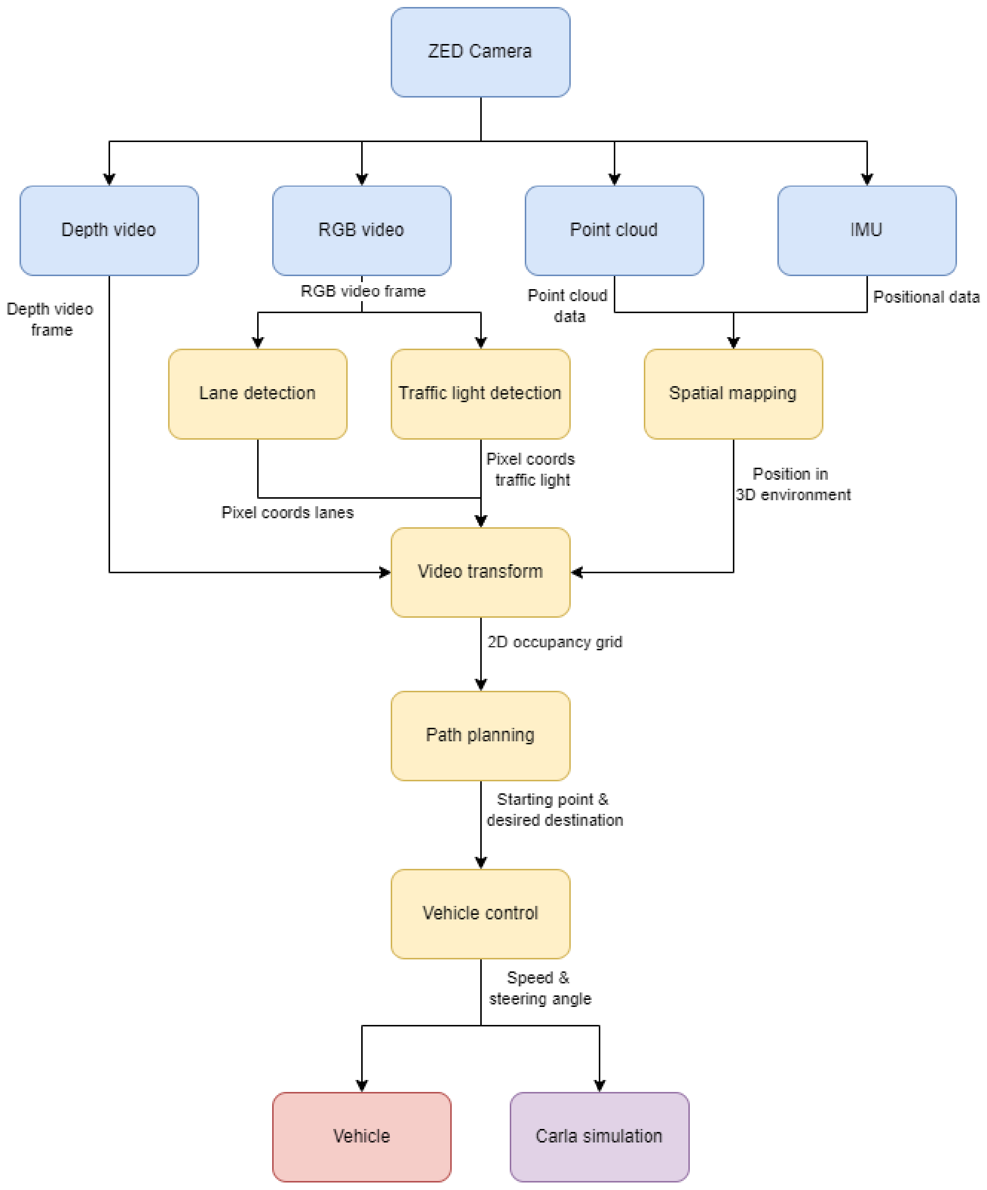
[19] Siegel, M. (2003). The sense-think-act paradigm revisited. *1st International Workshop on Robotic Sensing, 2003 (ROSE '03)*. IEEE. https://doi.org/10.1109/ROSE.2003.1218700

[20] NVIDIA Corporation, "Jetson Orin AGX: High-performance AI computing for edge devices," 2024. [Online]. Available: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/.

[21] Eclipse Foundation, "Eclipse Zenoh: Zero overhead pub/sub, store/query and compute," 2024. [Online]. Available: https://github.com/eclipse-zenoh/zenoh.

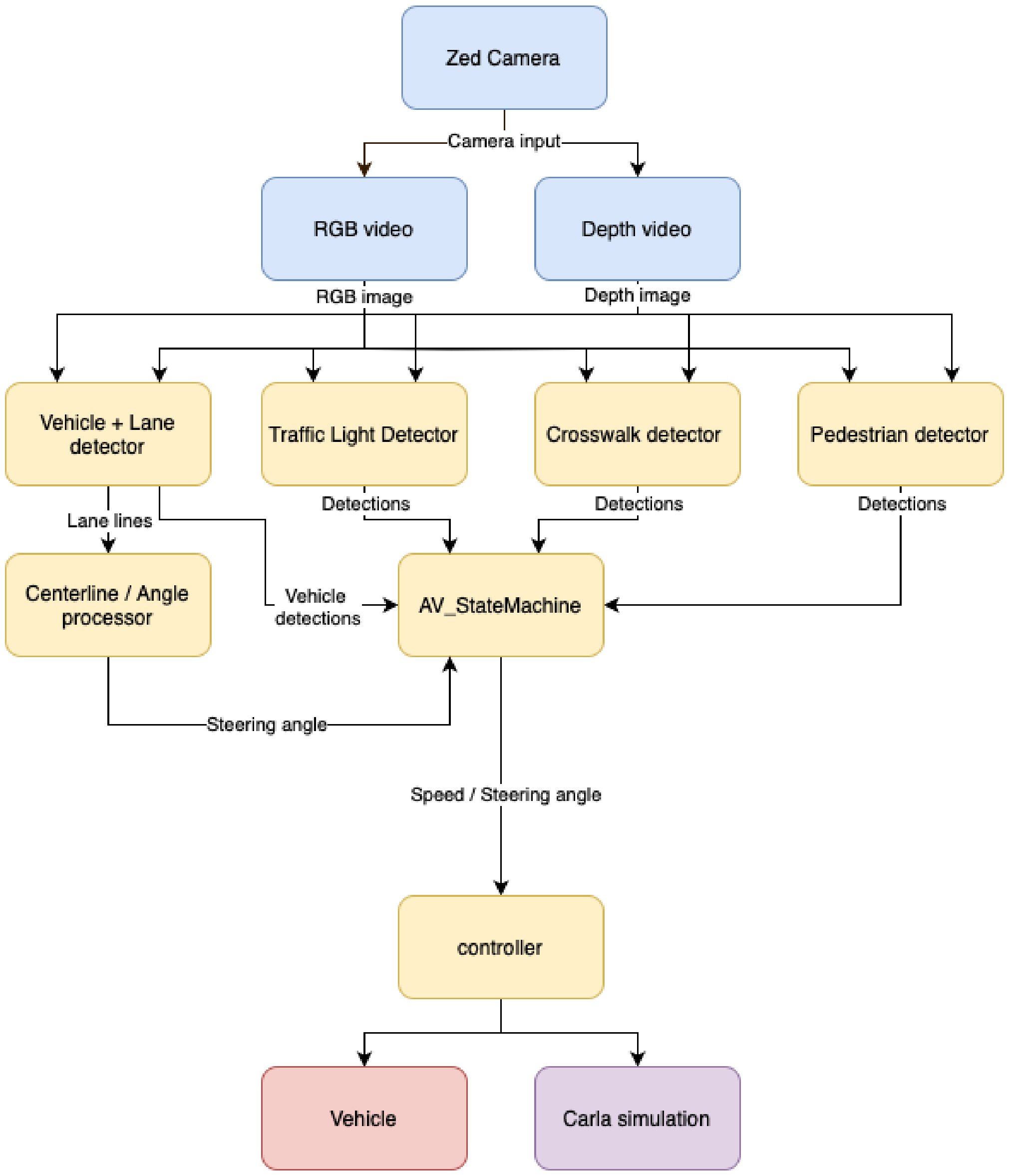
[22] Monolithic Power Systems, "MAQ473: 9-Bit to 14-Bit MagAlpha Automotive Angle Sensor with ABZ Incremental and PWM Outputs," 2024. [Online]. Available: https://www.monolithicpower.com/en/products/automotive-aecq-grade/sensors/maq473.html.

Appendix 1



Original system overview

Appendix 2



Final system overview