Conceptual Design Review Report - Spring 2024

PERCIV

(Perception Enabled Remote Control Interface for Vehicles)

Precise Teleoperation Using External Perception

Team C:

Dhruv Gupta
Sivvani Muthusamy
Khush Agrawal
Sri Sashank Undavalli
Shahram Najam Syed

Sponsor: Nissan Automotive Inc.
Liam Pederson & Viju James

2nd May 2024

Abstract

Annually, Nissan manufactures approximately 9 million cars, equating to 2 cars produced every minute. The challenge arises in efficiently transporting these vehicles to the storage unit, located 3 miles away from the manufacturing facility. Currently, Nissan employs around 100 manual laborers tasked with parking the cars in the storage unit. This process involves drivers shuttling between the manufacturing unit and storage area repeatedly, consuming significant time and energy resources.

This is where we need PERCIV: Precision Enhanced Remote Controlled Interface for Vehicles. PERCIV revolutionizes this labor-intensive process with its advanced teleoperation system. By leveraging PERCIV's intuitive controls, a single operator can seamlessly navigate a car from the manufacturing unit to the storage facility without the delays associated with traditional transportation methods. This streamlined approach eliminates the downtime spent traveling back and forth, maximizing operational efficiency and minimizing resource consumption. PERCIV's teleoperation system offers precise control over vehicle movement, ensuring safe and accurate navigation through the manufacturing environment and onto the storage area. Through the integration of advanced sensors and real-time feedback mechanisms, PERCIV enhances the operator's situational awareness, facilitating smooth and efficient parking maneuvers.

With PERCIV, Nissan can significantly reduce labor costs, enhance productivity, and optimize the overall manufacturing workflow. By automating the transportation and parking process, PERCIV empowers operators to focus on more value-added tasks, ultimately driving innovation and efficiency within the automotive manufacturing industry.

Table of Contents

1. Project Description

Car production facilities function at a swift rate, generating vehicles in an ongoing manner. The notable buildup of finished vehicles at the conclusion of the assembly line presents a considerable obstacle for car manufacturers. Addressing this concern involves directing these completed vehicles to a shipping yard, usually situated a short distance away. The traditional method of coordinating non-autonomous vehicles, utilizing human drivers, is marked by its repetitiveness, labor intensity, costliness, and sluggishness. While automated driving using external perception presents a potential solution to streamline this repetitive process, it comes with certain limitations. The main issue when using external perception is ensuring precise parking once the vehicle reaches the shipping yard. Other scenarios, such as navigating through challenging situations like construction sites or handling situations where autonomy fails, further highlight the need for an alternative solution.

Supervised Autonomy or "Human-on-the-loop"[1] is a potential solution to improve the performance and reliability of a fully autonomous solution when operated using external perception. With PERCIV, we aim to develop a system which allows a human operator to quickly and reliably teleoperate a car using external perception. PERCIV will help achieve this goal by enabling the operator to realistically visualize the scene around the car through mixed reality and seamlessly interact with the car through various modes of control. Following the timeline of the MRSD Project Course, PERCIV shall demonstrate a proof of concept of this solution on a scaled-down model. For this project, PERCIV will enable a human operator to precisely teleoperate and park an RC car through external perception on a test track.

2. Use Case

Nissan, an international automotive OEM, runs several car manufacturing plants in the United States. The expansive vehicle assembly facility in Smyrna, Tennessee, spans 884 acres and manufactures over 820 vehicles each day. In order to streamline the distribution process to dealerships throughout America, the completed vehicles need to be guided to a loading station positioned 5 miles from the assembly facility. Addressing the ongoing expenses and supply chain constraints, a solution involves the installation of infrastructure sensors along the route[2]. These sensors enable the vehicles to be autonomously and safely driven, ensuring precise lane-keeping and obstacle avoidance. This initiative aims to reduce costs and enhance the efficiency of the distribution process[3]. This solves the problem of reaching the shipping yard, but the inability to precisely park the cars at desired locations autonomously leads to congestion and inefficient use of space, which indirectly affects the throughput of the plant shown in Figure 1. At the time if the system is unable to park the car at a fast pace, it will then again bottleneck the supply chain, leading to reduced throughput shown in Figure 2. With the car now at the storage unit, the next challenge was to park it precisely between two parked cars. An operator logged into the PERCIV control interface, fusing data from multiple external sensors, was used to render a 3D replica of the car's surroundings. The operator is now able to intuitively perceive and effortlessly imagine the real scenario through multiple views, which won't be possible with raw camera feeds. Now, the operator remotely initiated the car's engine and provided steering commands to drive it from the storage unit to the desired parking area.

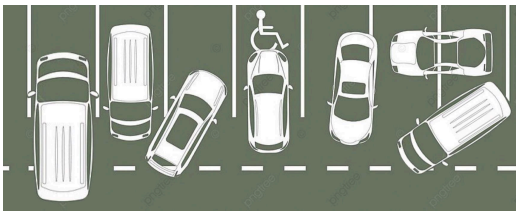

Figure 1: Imprecise parking

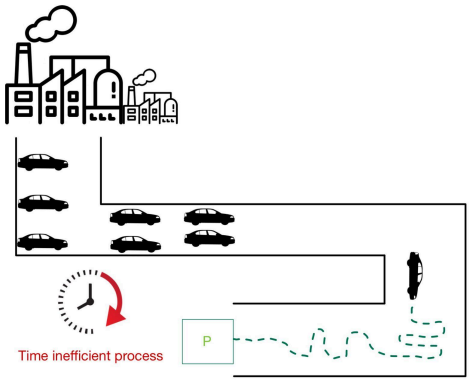

Figure 2: Time inefficient parking

The PERCIV system enhances operator control over the vehicle by displaying the car's predicted trajectory. During the motion a person walks in front of the car, the PERCIV system alerts the operator through visual cues; moreover, if the operator ignores those cues, the system overrides the commands and brings the vehicle to a halt. In addition to teleoperation, the system offers waypoint-based control, allowing the operator to select a set of points for the car to travel. Once reaching the desired parking station, the operator can take control for precise parking. During operation, an onboard proximity sensor detects obstacles and alerts the operator, who can then reassess the surroundings through various system views. This comprehensive approach ensures safe navigation and precise parking, reducing congestion and optimizing parking time. Refer to Figure 3 for visualizing the proposed workflow. Finally, the car is parked within the desired tolerance limit, reducing congestion and optimizing the time required for parking, hence increasing the overall throughput and saving cost for the automaker refer to Figure 4.

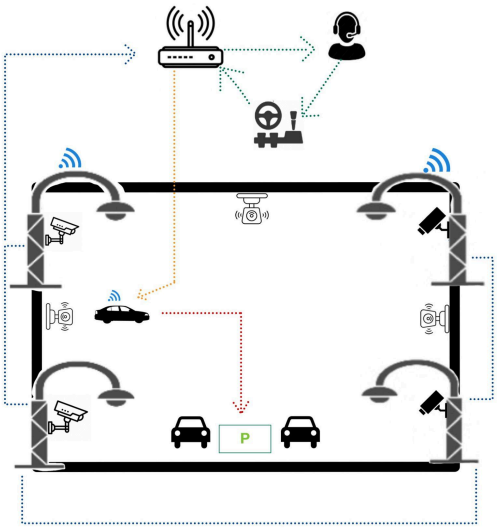

Figure 3: Proposed Workflow

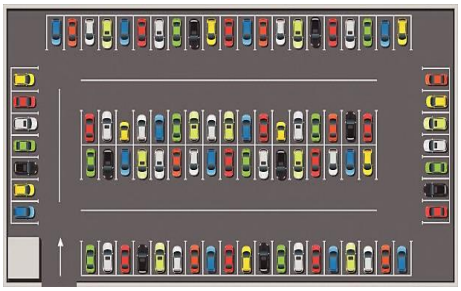

Figure 4: Final Output

3. System Requirements

System requirements are specifications that describe the functional and technical needs of a system. They serve as a guide for the development and evaluation of the system, ensuring that

the final product meets the needs of the user. System requirements are typically divided into two main categories: Functional and Non-Functional requirements.

3.1 Functional Requirements

Table 1: Functional Requirements		
Serial no.	Functional Requirements	FR ID
1	Create a virtual environment	FR1
2	Localize the vehicle in the virtual environment	FR2
3	Drive the car using external commands	FR3
4	Identify safety issues and alert/override driver commands	FR4
5	Park the car within the given tolerances	FR5

The Table 1 above lists down the functional requirements of our proposed solution, which are further explained below:

FR1: Create a virtual environment

This involves developing a detailed, dynamic 3D map and bird's eye view (BEV) of the vehicle's surroundings. The system utilizes a variety of off-board perception sensors like cameras, and LiDAR along with on-board odometry data to generate comprehensive visualizations. This detailed virtual representation is crucial for the teleoperator to understand and navigate the environment safely and efficiently.

FR2: Localize the vehicle in the virtual environment

This function involves accurately determining the vehicle's position within the virtual map. The accuracy of this localization is vital, as even small errors can lead to significant navigational mistakes.

FR3: Drive the car using external commands

This aspect refers to the vehicle's ability to effectively interpret and execute driving commands issued by a teleoperator. These commands could be specific driving instructions such as steering input and velocity in manual mode of operation or simple waypoints in case of the semi-assisted mode.

FR4: Identify safety issues and alert/override driver commands

The system continuously monitors for potential safety hazards, such as obstacles on the road, unexpected actions from nearby vehicles and pedestrians etc. If it detects a potential risk, the system can alert the teleoperator, or take corrective action automatically if the issue still persists.

FR5: Park the car within the given tolerances

This function involves the system's ability to enable the teleoperator to park the vehicle within a specified area, adhering to predefined tolerances. The teleoperator should be comfortably able to park the vehicle within the tolerances using the visual environment provided by the system.

3.2 Performance Requirements

Performance requirements stem from functional requirements and provide measurable criteria for the results of a functional requirement. Performance requirements quantify the system's capability to meet the functional requirements.

Table 2: Mandatory Performance Requirements

Serial no.	Mandatory Performance Requirements	PR ID
1	Update Frequency of 10 hz	PR1
2	Successful parking is detected: 80%	PR2
3	Stationary tolerance: Y: ±5 cm; X: ±10 cm; Rotational ±20; Degrees	PR3
4	Vehicle speed should be around 10 cm/s	PR4
5	Safety Tolerance: 10 cm	PR5

The Table 2 above lists down the mandatory performance requirements of our proposed solution, which are further explained below:

PR1: Update frequency of 10 hz

This means the system updates the information displayed to a user 10 times per second. A 10 Hz update rate ensures that the teleoperator has real-time visualizations and data for decision making which is essential for safe and effective teleoperation.

PR2: Successful Parking Detection Rate of 80%

This requirement signifies that the system must accurately detect successful parking instances with a rate of at least 80%. Achieving this level of accuracy ensures reliable feedback to the operator, fostering confidence in the system's performance and minimizing the risk of errors during parking maneuvers. Associated Functional Requirements: FR5

PR3: Stationary Tolerance: Y: ±5 cm; X: ±10 cm; Rotational: ±20 Degrees This specification defines the acceptable tolerance limits for the vehicle's position and orientation while stationary. The system must maintain the vehicle's position within a range of ±5 cm vertically, ±10 cm horizontally, and ±20 degrees rotationally. Adhering to these tolerances

ensures precise and consistent parking outcomes, minimizing the likelihood of collisions or misalignments. Associated Functional Requirements: FR2,FR3, FR5

PR4: Vehicle Speed Around 10 cm/s
This requirement mandates that the vehicle's speed during teleoperation should be maintained at approximately 10 cm/s. Controlling the vehicle within this speed range ensures smooth and steady movement, facilitating precise maneuvering and parking in confined spaces while minimizing the risk of accidents or overshooting.
Associated Functional Requirements: FR3

PR5: Safety Tolerance: 10 cm
This refers to the minimum distance the vehicle should maintain from obstacles or other objects at all times. Associated Functional Requirements: FR3, FR4, FR5

The performance requirements outlined are essential and non-negotiable for our system's functionality. Nonetheless, our aim is to surpass these baseline requirements, striving to achieve additional desired outcomes, as detailed below in Table 3.

Table 3: Desired Performance Requirements

Serial no.	Desired Performance Requirements	PR ID
1	Update Frequency of 30 hz	PR6
2	Successful parking is detected: 90%	PR7
3	Stationary tolerance: Y: ±5 cm; X: ±10 cm; Rotational ±20; Degrees	PR8
4	Vehicle speed should be around 15 cm/s	PR9
5	Safety Tolerance: 5 cm	PR10

3.3 Non Functional requirements

Non-functional requirements refer to the criteria that define how a system should operate, rather than specific behaviors or functions the system performs as listed below in Table 4.

Table 4: Non Functional Requirements

Serial no.	Non Functional Requirements	NFR ID
1	Infrastructure cameras placed at oblique angles	NFR1
2	Infrastructure Lidar/Depth Sensor	NFR2

3	Onboard odometry	NFR3
4	Onboard proximity sensors	NFR4
5	Car Scale: 1/10th of real world	NFR5
6	Forward Parking	NFR6
7	Backward Parking	NFR7
8	Parallel Parking	NFR8

4. Functional Architecture

The functional architecture of our system provides a detailed overview of its operations, beginning with data acquisition and culminating in vehicle control. The architecture is divided into several key components, each playing a crucial role in ensuring the system's efficiency and safety as can be seen in Figure 5 below.

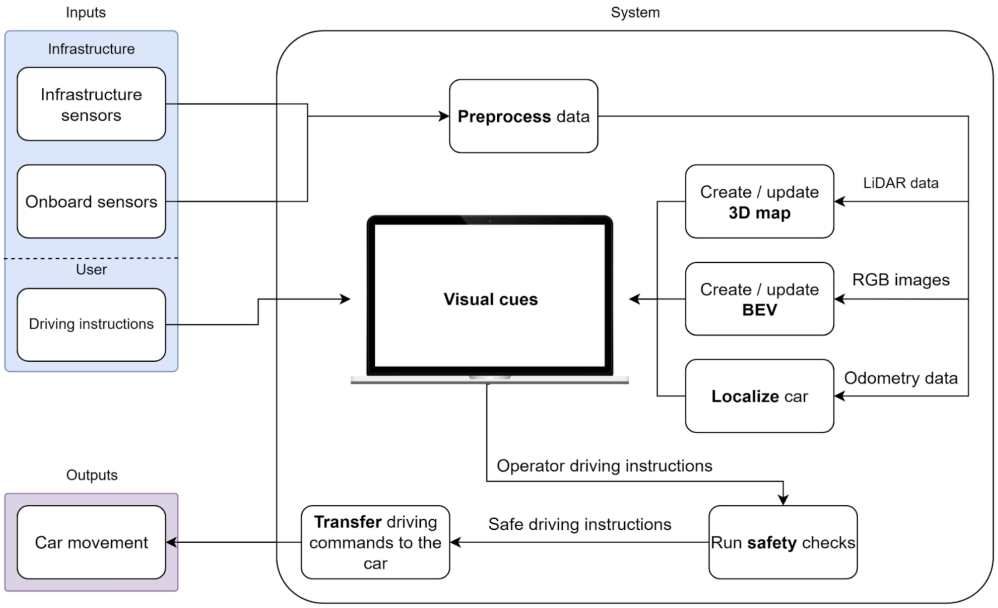

Figure 5: Functional Architecture

4.1 Data Acquisition and Pre-Processing:

The process starts with gathering data from a range of sensors. Off-board sensors, such as cameras and depth sensors, provide external environmental information. Simultaneously, on-board sensors, including odometry offer vital data about the vehicle's internal state. This data is then pre-processed as necessary. Pre-processing involves filtering and refining the data to ensure accuracy and usability, which is essential for the subsequent stages of the system.

4.2 Perception Pipeline:

Following data acquisition, our system boasts a sophisticated perception pipeline. This advanced framework leverages pre-processed data to generate detailed environmental representations, enhancing the system's understanding of its surroundings. It produces both a 2D Bird's Eye View (BEV) spatial context, facilitating navigation and obstacle avoidance for the vehicle. Additionally, as part of our ongoing development efforts, we will also provide 3D environmental representations, further enriching the system's perception capabilities and enhancing its ability to navigate complex environments with depth information.

4.3 UI/UX Pipeline:

The generated maps are then integrated into our UI/UX pipeline. Here, visual cues are overlaid on the maps, taking into account the data from on-board sensors and inputs from the teleoperator. This step is critical in visualizing the vehicle's environment in a way that is intuitive and informative for the teleoperator. It helps in making informed decisions regarding vehicle control and navigation.

4.4 Safety Pipeline:

The teleoperator's inputs are subsequently fed into the safety pipeline. This system is designed to rigorously check for any violations of predetermined safety tolerances. It acts as a failsafe, ensuring that the operator’s commands do not lead the vehicle into unsafe conditions. By continuously monitoring for potential risks and ensuring compliance with safety standards, the safety pipeline plays a pivotal role in maintaining the overall safety of the system.

4.4 Transmission of Driving Instructions:

Finally, the driving instructions, vetted for safety, are transmitted to the vehicle. This transmission is the culmination of the process. The vehicle then operates in accordance with the desired routes and maneuvers while strictly adhering to safety protocols. In summary, our system's functional architecture is a meticulously designed framework that encompasses everything from data gathering to the execution of safe driving instructions. Each component is tailored to work seamlessly with others, ensuring the system operates efficiently, safely, and effectively

5. Cyber-physical architecture

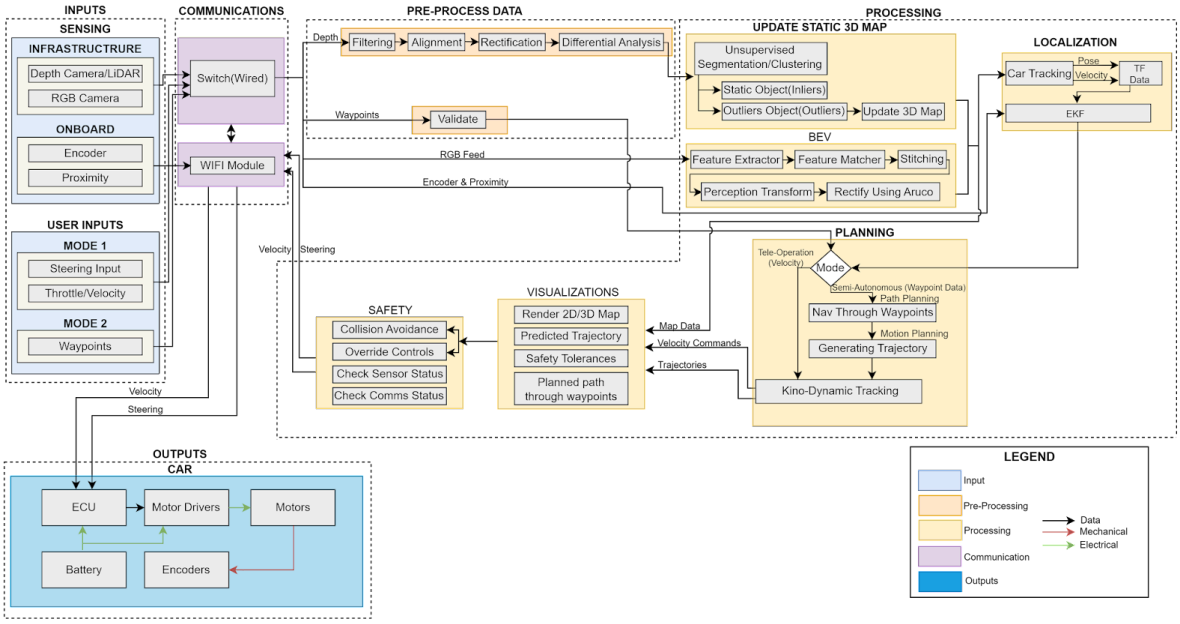

Figure 6: Cyber-physical Architecture

The cyber-physical architecture of PERCIV is presented above in Figure 6, and is delineated into various core-building blocks, with each core building block playing a pivotal role in achieving precise control and navigation of the RC vehicle without onboard sensing capabilities. Following is the detailed explanation of key features and aspects of the core-building blocks of the aforementioned cyber-physical architecture:

5.1 Inputs

The inputs sub-block of the system is divided into three components:

a. Infrastructure Sensors, using LiDAR/Depth and RGB cameras to perceive the environment and localize obstacles and the ego car, designed with scalability in mind;
b. Onboard (passive) sensors, such as encoders and proximity sensors from the ego-car, used for sensor fusion and as fallbacks in case of communication failure;
c. User-input, split into two modes - 'teleoperated' or 'fully-dependent', where the teleoperator directly controls the ego vehicle's heading and velocity, and 'semi-autonomous', where the vehicle's movement is influenced by waypoints and interactions with environmental participants/obstacles, rather than direct teleoperator control.

5.2 Communication

Once the data from the sensing sub-block and the user-input sub-block from the input core block is received, the communication interface is responsible for transferring data to an off-board compute. This communication interface consists of a LAN switch (in simplex configuration) for handling wired infrastructure sensor data stream as well as user-input through hardware haptic interfaces, in addition to a WiFi module to establish a duplex communication

with the ego vehicle for transferring real time odometry data from the vehicle to the off-board compute, and then driving instructions (velocity profile and heading) from the off-board compute to the ego vehicle.

5.3 Pre Processing

The data pre-processing stage involves four critical steps: Filtering for noise reduction, dense cloud thinning, and surface smoothing; Alignment using voxel structuring, the Iterative Closest Point algorithm, and feature-based alignment for map coherence; Rectification for global coordinate alignment and consistent merging of point clouds; and Differential Analysis for change detection and differentiation between static and dynamic objects. Additionally, it validates teleoperator-provided waypoints to prevent collision risks or inaccurate trajectories, ensuring precise navigation in dynamic environments.

5.4 Processing

The processing core-block of the cyber-physical architecture handles key functions of the offboard compute:

a. 3D Map Update, involving updating the pre-generated 3D static map with dynamic obstacles isolated using differential analysis and rendered as smoothed reconstructed mesh grids;
b. Generating a 2D BEV of the environment by stitching and transforming RGB camera streams from different viewpoints, corrected for distortions using ARUco markers;
c. Localization, using the 2D BEV, 3D updated map, and encoder odometry, tracking the car's position and orientation using Extended Kalman Filters for sensor fusion;
d. Planning, determining vehicle control based on operating mode—direct control in teleoperated mode or trajectory generation using waypoints in semi-autonomous mode;
e. Visualizations, creating the Human Machine Interface with visual cues overlaid on 3D maps and 2D BEVs to guide the teleoperator;
f. Safety, managing collision avoidance and monitoring for sensor failures or communication latency, ensuring safe vehicle operation.

5.5 Output

Finally the velocity and steering commands from the safety sub-block (of the processing core block) are transferred to the ego vehicle where the ECU is interfaced with the motors using the motor driver, which translate the steering of the Ackermann drive as well as the velocity based on the received command.

6. Current System Status

6.1 Targeted Requirements

This semester, our team aimed to achieve all system-level requirements, allowing for some flexibility. Our goal was to achieve all of the desired requirements with a lenient view on some of the parameters. For example our desired FPS was 30hz but we were able to achieve around 20-22 FPS. A detailed list of our targeted requirements is given in table 5.

Table 5: Targeted Performance Requirements			
PR ID	Requirement	Subsystem	Status
PR1	Update Frequency of 10 hz	Perception	Passed
PR2	Successful parking is detected: 80%	Perception	Passed
PR3	Stationary tolerance: Y: ±5 cm; X: ±10 cm; Rotational ±20; Degrees	Controls	Passed
PR4	Vehicle speed should be around 10 cm/s	Controls	Passed
PR5	Safety Tolerance: 10 cm	Controls	Passed

6.2 Overall System Description

The comprehensive system encompasses our sensor suite, consisting of off-board infrastructure cameras and on-board proximity sensors, alongside the G29 controller. Our software stack comprises several subsystems operating collaboratively. The data processing and Bird's Eye View (BEV) generation pipeline is tasked with delivering a precise and responsive BEV. Simultaneously, the controls pipeline captures user inputs and channels them through a safety mechanism before implementation. The UI/UX module serves as the integration hub, providing users with an accurate visualization enriched with advanced visual overlays for teleoperation. Figure 7 depicts the overall flow of the system.

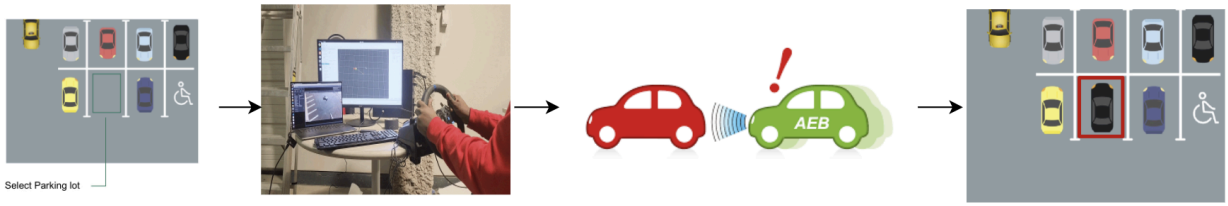

Figure 7: System Overview SVD

6.3 Subsystem Description

6.3.1 Sensing Modalities

PERCIV mostly relies on external sensing for teleoperation of RC cars in the miniaturized parking lot scenario. Currently, the external sensing involves using RGB cameras which will be expanded to incorporate depth sensing to output point clouds of the environment. Some data from onboard sensors are also used. This involves reading odometry data for estimating the position of the car and proximity sensors for estimating nearby obstacles. The onboard data is transmitted wirelessly to a computer through a ROS2 DDS node communication.

External sensors are connected to the local network over a wired connection and transmitted to a control module (computer).

6.3.2 Perception

Currently the perception pipeline consists of the following flow:

a. Receive time-synchronized image frames from the intel Realsense using software implementation of GenLock, in which a trigger is sent programmatically to the realsense sensors to control the shutters and exposures.
b. Once the time-synchronized images are received, they are stitched using a Graph-Neural Network based stitcher optimized to run in realtime and to operate in feature sparse environments.
c. Once the scene is stitched, points are selected manually to perform perspective transform into a BEV. Inverse Perspective Mapping was also experimented, but due to better fidelity of a simple homography using perspective transformation in our case we decided to stick with perspective transforms. The final stitched BEV can be seen in Figure 8.

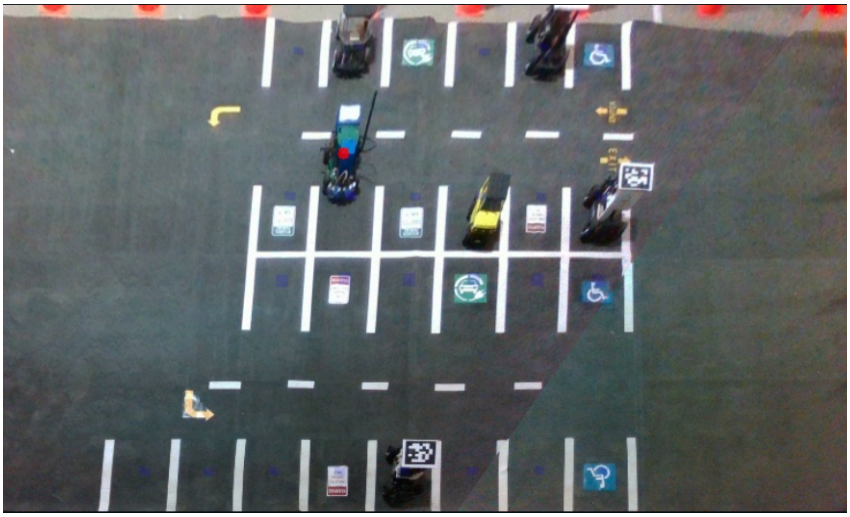

Figure 8: Stitched BEV output

d. With the BEV generated a learning-based algorithm is used for keypoint detection for explicitly pose estimation of the ego-vehicle. The mAP (mean average precision) we were getting against the ground-truth (measure using an ArUCO marker) was 97.5%. The estimated pose can be seen in Figure 9:

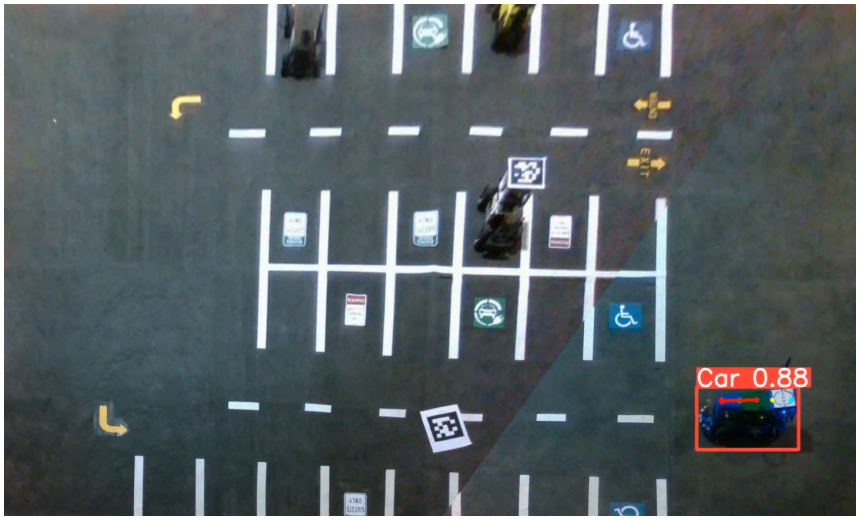

Figure 9: Estimated pose estimation for the ego-vehicle

e. With the pose of the ego-vehicle determined the kinodynamic model is used complemented by the user input to generate trajectories as a t+nth timestep based on the ackermann steering. The predicted trajectories in green can be seen in Figure 10 below:

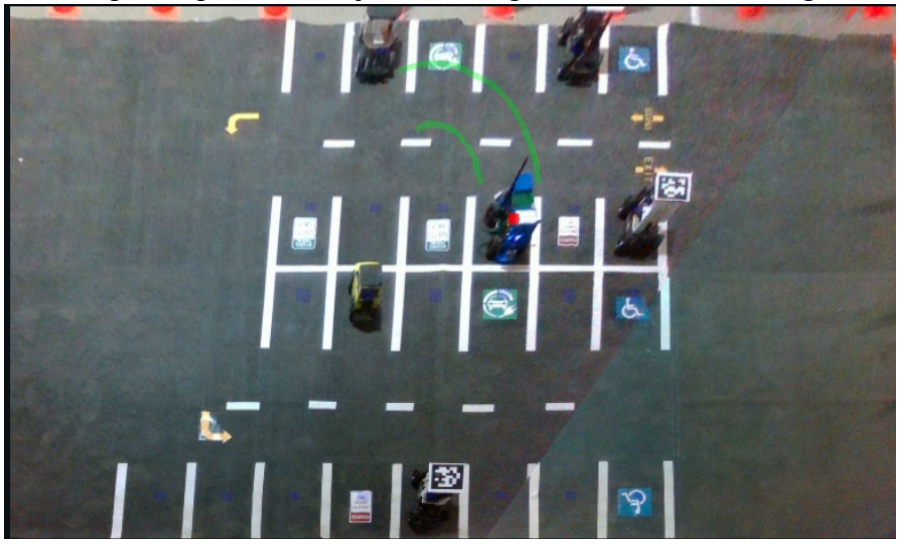

Figure 10: Predicted trajectories using estimated pose and controller input

f. Finally an independent greedy-heuristic for box-to-box center alignment and pose alignment to detect successful parking. Figure 11 depicts the translucent green overlay achieved as a result of successful parking:

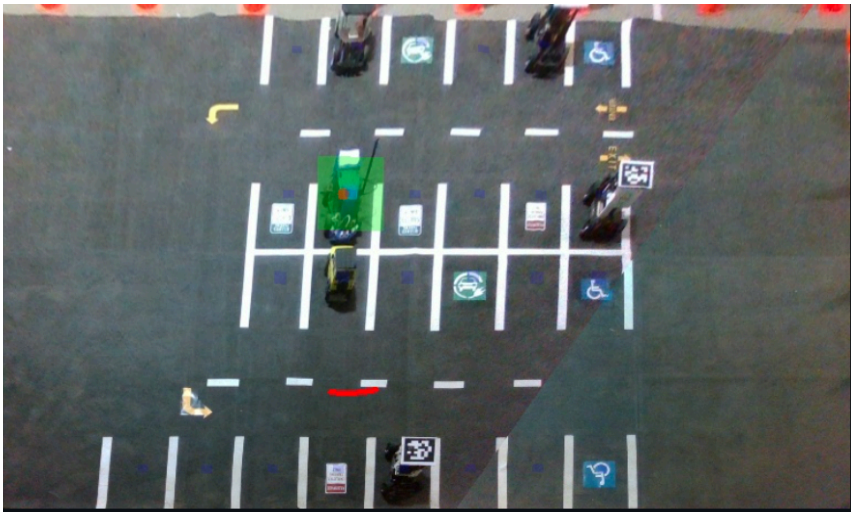

Figure 11: Green overlay depicting successful parking and within the designated tolerance

6.3.3 UI/UX

Inherently, the complete system is a human-machine interface with UI/UX as a core submodule of the complete system. Various off-the-shelf solutions which included ROS web-bridge integration with bootstrap, Glovo, and simple HTML CSS based designs, but due to limited flexibility to change the UI/UX and high latency/low throughput from the web-bridge we opted to design our custom UI/UX design using StreamLit and ReactJS. The communication between the frontend and backend was achieved using ROS topics with high compression rates for image frames for high throughput. The elements included a side navigation bar for selecting between raw CCTV footage, pre-defined parameters for stitched BEV, and user-defined parameters for stitched BEV. In addition, the user was provided with a real time feed of the environment bolstered by other graphical interfaces which included a speedometer, gear status, and safety override status. Figure 12 shows the current status of our UI/UX:

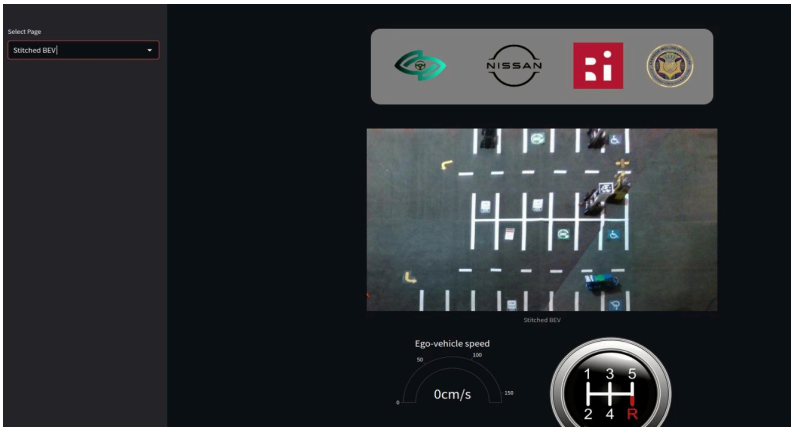

Figure 12: UI/UX depicting stitched BEV, speedometer, gear status, and safety override status

6.3.4 Planning and Control

- Steering and Throttle Input

For our setup, we used the Logitech G29 steering and pedal setup. We fine-tuned the force feedback steering option to replicate the sensation of maneuvering a real vehicle, ensuring an immersive driving experience. The pedal shifters integrated into the steering wheel offer gear changes, allowing drivers to transition through three forward gears and one reverse gear, which is also displayed on the UI/UX. The plus-minus button on the wheel can be used to change the length of the trajectory prediction. The red button can be used to disable the safety checking vehicle driving the car. All the key mapping are depicted in Figure 13.

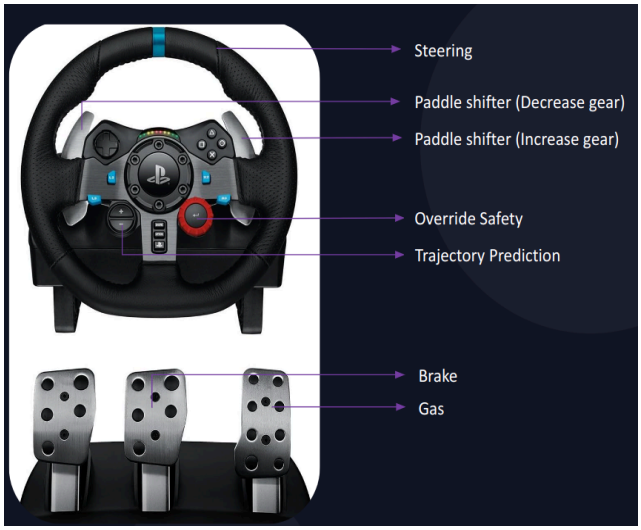

Figure 13: User Controller

- Trajectory prediction

Trajectory prediction stands as a pivotal feature, offering drivers a proactive understanding of the vehicle's forthcoming position based on present throttle and steering inputs. This tool enhances situational awareness, allowing drivers with foresight to navigate with precision and confidence. Because we are using an Ackerman type of car, we used the peppy model to predict car position and orientation given the steering commands. The lookahead distance was based on the length that the driver selected using the plus-minus button on the steering.

- Collision Avoidance

In order to mimic the actual car, we had 3 sensors in the front and 1 sensor in the back, as shown in figure 14. These were low-cost proximity sensors and a very narrow FOV of 5 degrees. The logic involved reading these ultrasound sensor data topics, as well as the teleoperation setup input data topic (raw_cmd_vel) to filter out unsafe command velocity components and only publish velocities that are safe to execute (do not move the car in the direction of possible collisions). Therefore, these safe velocities are published in a ROS topic (cmd_vel). These safe velocities are then subscribed directly to the car and executed.

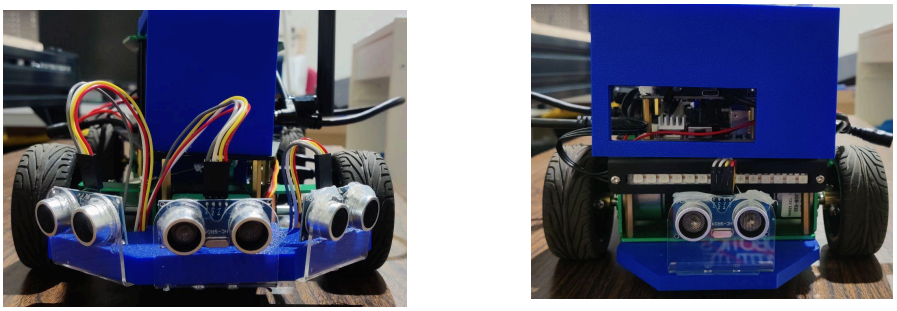

Figure 14: Sensor placement on the car

6.4 Modeling, Analysis and Testing

This subsection defines the key experiments allowing re-iteration over the design and implementation decisions taken in individual subsystem units and for integration during the spring 2024 semester.

Table 6: Targeted Performance Requirements for Spring Semester

Test	Success Criteria	Result
BEV Generation Using Classical Perspective Transform Test	The BEV accurately represents the test environment from a top-down perspective. 2. Distortions are minimized, and 70% of key features are correctly aligned with their real-world positions	The BEV is able to capture almost all details of the tested environment within the camera's field of view.
Test car with Steering and pedals.	The car responds to user input verified visually.	The car was smooth responding to user inputs.
Predict trajectories from car's odometry data	Predicted trajectories look reasonable in terms of direction.	The predicted trajectories reasonably represent the car's actual motion.
BEV Stitching Using Super Point and Super Glue Test	1. The final stitched BEV presents a seamless panoramic view with high accuracy in feature alignment. 2. No significant stitching artifacts are present, indicating successful integration of the images.	The stitched BEV is able to capture almost all details of the entire test environment capture by two cameras' field of view.
Add basic collision / safety checking	Safety warnings are displayed when the RC car is closer than a specified threshold.	Safety warnings are displayed when obstacles are closer than 15 cm.
Advanced Overlay Visualizations Test	1. Visualizations accurately overlay on the BEVs and provide meaningful assistance to the user. 2. Users report improved situational awareness and operational efficiency due to the visualizations.	Safety and trajectory predictions overlays work consistently on the GUI and multiple users report their effectiveness.

6.5 SVD Performance Evaluation

The set of system capabilities that were demonstrated in the SVD and SVD Encore are shown in Table 7.

Table 7: Demonstrated Performance Requirements for Spring Semester		
Procedure	Success Criteria	Requirements Satisfied
Demo 0: Operator tele operates the car using the standard camera feed.	The car moves smoothly and responsively in accordance with operator commands.	PR1, PR4
Demo 1: The car is teleoperated using the PERCIV system.	The car demonstrates precise and expert maneuvering, showcasing optimal teleoperation.	PR2, PR3, PR4
Demo 2: The car is parked using visual overlays on the base environment for parking assistance.	The car successfully parks within the specified tolerance limits using the visual overlays.	PR2, PR3, PR4
Demo 3: Implementation of collision avoidance algorithm is demonstrated.	The car effectively detects and avoids collisions with static obstacles, ensuring safe navigation.	PR5
Demo 4: Safe implementation during camera movement and when car not within visual range	The system detects and responds to camera movement, ensuring continued safe operation and minimal disruption to teleoperation.	PR5

In the spring validation demo, we successfully demonstrated all seven scenarios outlined above, showcasing the versatility and robustness of our system. Each demo was executed according to the specified procedure, with the success criteria met in each case. These demonstrations encompassed various aspects of our project, including teleoperation, parking assistance with visual overlays, collision avoidance, camera movement detection, and safety measures when the car is not within visual range. By meeting all performance requirements in these demonstrations, we validated the effectiveness and reliability of our system, marking a significant milestone in its development and deployment.

6.6 Strengths and Weaknesses

Our progress through the semester led to a successful demonstration during SVD and SVD Encore. While there are several strong areas of our system, there are also areas that need improvement. The strong and weak points of our system are highlighted below.

Strengths:

- Precise Teleoperation throughout: Our system excels in providing precise control over the vehicle, ensuring accurate navigation and parking maneuvers.
- Consistent car detection and tracking: The system reliably detects and tracks the vehicle's movements, facilitating seamless teleoperation and navigation.
- Car teleoperated with minimal latency: Operators experience minimal delay between input commands and vehicle response, enhancing real-time control and maneuverability.
- Meaningful visualizations for the operator: The system provides operators with clear and informative visualizations, aiding decision-making and enhancing situational awareness.

- All systems time-synchronized: The synchronization of all system components ensures seamless coordination and operation, minimizing discrepancies and enhancing overall system performance.
- Fail-safe collision detection: The system incorporates fail-safe mechanisms for collision detection, enhancing safety and mitigating risks during operation.

Weaknesses:

- Inclusion of autonomy: The system's reliance on teleoperation may limit its autonomy in certain scenarios, necessitating further development in autonomous capabilities.
- Induced latency to closely mimic real-world: In order to closely mimic real-world conditions, the system may introduce latency, potentially impacting operator experience and responsiveness.
- User experience: While the system offers robust functionality, there may be areas for improvement in user experience, such as interface design and ease of use, to optimize operator satisfaction and efficiency.

7. Project Management

7.1 Work Plan and tasks

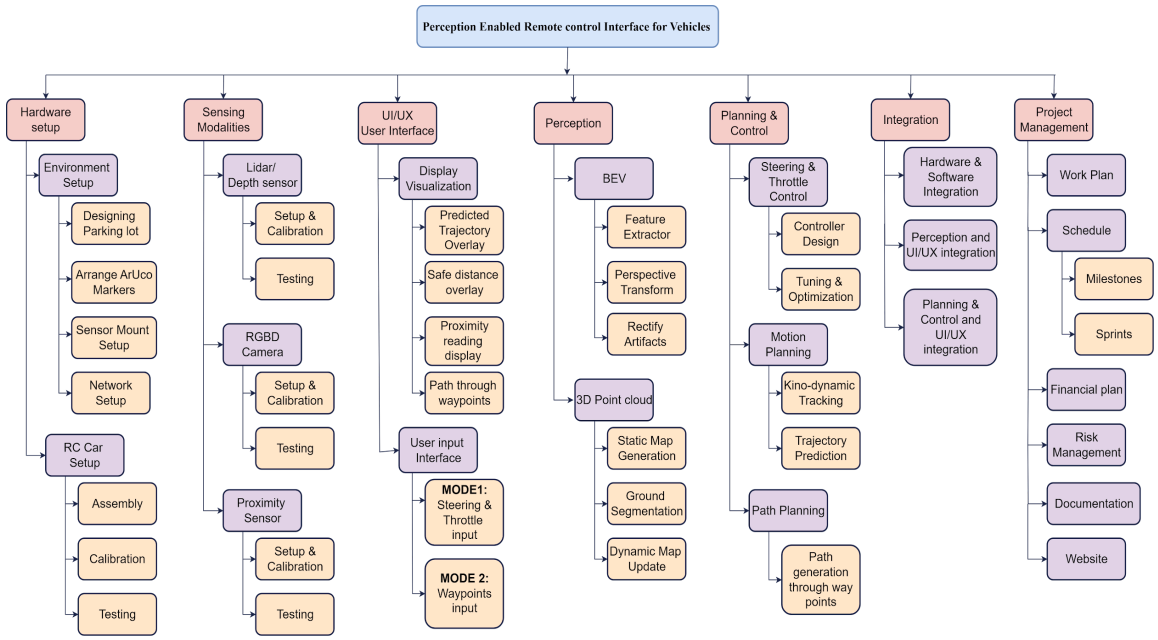

Figure 15: Work breakdown structure

The project's primary objective is to demonstrate a teleoperated parking system using external perception, blending product and process-oriented elements within a meticulously structured Work Breakdown Structure (WBS). Central to this endeavor are key product elements such as Hardware Setup, Sensing Modalities, UI/UX Interface, and the Perception Subsystem, all developed in accordance with the V-model of systems engineering. Hardware development is divided into Environment Setup and RC Car Setup, encompassing various phases including procurement, integration, and testing of mechanical and embedded hardware upgrades.

Additionally, onboard software development for control systems and communication modules is a crucial aspect, ensuring the creation of a stable mechatronic system for thorough subsystem testing and documentation.

Moving forward into the Fall term, the project will pivot towards customizing and replicating more Jetacker cars, a critical step in scaling the system. This phase will involve implementing advanced technologies such as localization and tracking of multiple cars, integrating waypoint-based control mechanisms, and constructing a multi-agent system to enhance the system's efficiency and robustness. Simultaneously, a key focus will be on providing comprehensive 3D visualization of the environment, enabling operators to monitor and interact with the system seamlessly. These advancements are pivotal in advancing the project's objectives and showcasing its capabilities effectively.

Furthermore, agile management principles will continue to underpin project operations, ensuring continuous tracking of lower-level work products, effective budget management, and proactive risk mitigation strategies. Regular meetings dedicated to assessing project progress and addressing emerging challenges will be instrumental in maintaining project momentum and ensuring timely completion. Documentation efforts will proceed alongside development activities, capturing the evolution of the system and providing valuable insights for future iterations. Overall, this structured approach, coupled with the integration of cutting-edge technologies, positions the project for success in achieving its objectives and delivering a compelling demonstration of the teleoperated parking system.

7.2 Schedule

The work schedule is represented using a Gantt chart in Figure 16. This is derived from the work plan defined in the previous subsection and associated dependencies of the tasks considering team member responsibilities in various phases of the project.

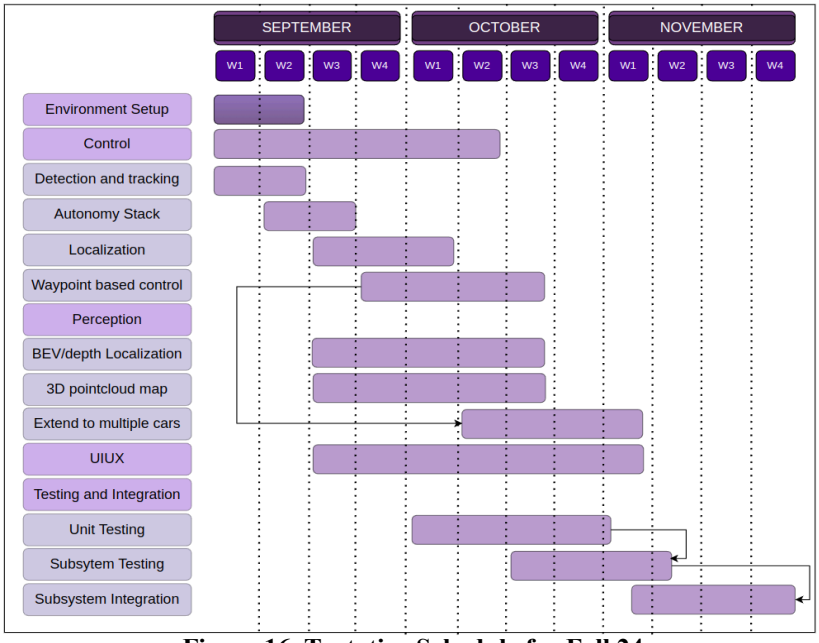

Figure 16. Tentative Schedule for Fall 24

Our objective is to finish the Full-System Development at least four weeks before the Fall Validation Demonstration to allow sufficient time for testing and debugging. As of now, we are on track with our schedule and will proceed with our tasks as planned.

7.3 Test Plan

During the forthcoming Fall Semester, we aim to augment our existing system significantly by implementing additional functionalities. Specifically, we plan to introduce enhancements such as 3D visualization, and to refine our capabilities in localization, control, tracking, and obstacle avoidance throughout navigation.

Table 9: Fall Test Plan	
Setup	Test bed (parking lot), 2x Intel RealSense D435i, 2x ZED Stereo Cameras, 2x tripods, 2x infrastructure stands, 2 RC car, Raspberry Pi, External Compute Unit, LogiTech Steering and Pedals, Wi-Fi Router, Monitor, Stopwatch
Objectives	FVD.1: Detect, track, and estimate the pose of 1 car using BEV/Depth FVD.2: Setup autonomy stack for the car FVD.3: Localisation based on BEV/Depth + Odometry for 1 car FVD.4: Demonstrate mode 2: Waypoint based control on 1 car FVD.5: Scale BEV/Depth based localisation to 2 cars FVD.6: Demonstrate mode 2: Waypoint based control on 2 cars FVD.7: Demonstrate 3D view in UI/UX FVD.8: Park RC car successfully 80% of the time using pedals and steering FVD.9: Drive the RC car at an average speed of 10 cm/s
Success Criteria	SC.1: Successful parking is detected 80% of the times SC.2: Safety tolerance of 10 cm is maintained at all times from obstacles
Verification Method	Comparison of PERCIV against ground truth

In the upcoming semester, our focus for the Fall Validation Demo (FVD) revolves around a sequence of actions aimed at advancing our system's capabilities. Firstly, we will prioritize detecting, tracking, and estimating the pose of one car using BEV and Depth sensing technologies. Following this, we'll set up an autonomy stack for the car, laying the foundation for subsequent autonomous functionalities. Localisation becomes paramount next, as we combine BEV/Depth data with Odometry to precisely determine the car's position. We'll then introduce Mode 2: Waypoint-based control for one car, enhancing navigation accuracy. Scaling up, we'll extend BEV/Depth-based localisation to two cars, further refining our system's robustness. Accompanying this, we'll demonstrate Mode 2 control for two cars simultaneously and introduce a 3D view in our user interface, enhancing user experience and comprehension.

Throughout these milestones, we will implement advanced sensor fusion techniques, refine control algorithms, and optimize system performance to achieve our objectives efficiently.

7.3.1 Fall System Capabilities

Table 10: Fall Capability Milestone

PR	System Capabilities	Date
PR 7	The team is tasked with fully customizing two RC cars, making comprehensive modifications to both hardware and software components. Furthermore, the team should implement detection, tracking and pose estimation of one car. **FVD.1:** Detect, track, and estimate the pose of 1 car using BEV/Depth **FVD.2:** Setup autonomy stack for the car	Early-Sep
PR 8	Program the RC cars to understand and respond to motion and parking signals. Further, localize one car in the PERCIV environment using BEV/depth and the odometry data. **FVD.3:** Localisation based on BEV/Depth + Odometry for 1 car	Mid-Sep
PR 9	Implement waypoint based control in one car and integrate with UI/UX to obtain user input. Utilize the localization and pose estimation of the car. **FVD.4:** Demonstrate mode 2: Waypoint based control on 1 car	Early-Oct
PR 10	Expand the localization and control system to operate multiple cars simultaneously. **FVD.5:** Scale BEV/Depth based localisation to 2 cars	Mid-Oct
PR 11	The system must showcase the ability to control two cars independently using separate commands. Additionally, it should demonstrate point cloud visualization and the capability to detect multiple cars within the environment. **FVD.6:** Demonstrate mode 2: Waypoint based control on 2 cars **FVD.7:** Demonstrate 3D view in UI/UX	Mid-Nov
PR 12	The system should be capable of operating multiple cars within an environment containing obstacles. Additionally, it needs to undergo repeatability tests to ensure consistent performance, ensuring that the entire system functions seamlessly as one cohesive unit **FVD.8:** Park RC car successfully 80% of the time **FVD.9:** Drive the RC car at an average speed of 10 cm/s	Late-Nov

7.3.2 Testing Activities

Here are the testing activities outlined:

1. Successful Parking: Out of the total trials conducted, the aim is to ensure that the car successfully parks precisely 80% of the time. The success of parking is determined by the system itself, utilizing the predefined boundaries within the environment.

2. Environment Visualization Test: This test ensures that at least 80% of the features within the environment are clearly visible and perfectly aligned. Additionally, it assesses the efficiency of the teleoperator in visualizing the environment and maneuvering the car within the parking scene.

3. Collision Detection: In the Spring Validation Demo, we successfully demonstrated the system's ability to ensure safe stops when collisions were encountered with static obstacles. For the Fall Validation Demo, where multiple cars will be introduced and operated within the environment, the system's capabilities will be extended to seamlessly detect collisions with other moving cars and obstacles in the environment.

7.4 System Validation Experiments

7.4.1 Fall Validation Demo

- **Location:** Newell-Simon Hall B level using the Modular Test Track
- **Equipments:** Disassembled Test track, 4 x RGB cameras, 2 x Depth cameras, RC car, adjustment blocks, Tool box, external computer, Monitor showing outputs
- **Test Bed Description:** 5X5 meters miniature parking lot setup

7.4.2 Procedure

Table 11: FVD Procedure

S.No	Procedure	Success Criteria
1	Position the tripods with the RealSense D435i cameras at oblique angles to ensure overlapping coverage for both cameras and setup Lidars.	Overlapping visual feed from both cameras
2	Power on the intranet WiFi router, off-board sensors, and RC units for all cars.	Update frequency 10 Hz (PR1)
3	Configure each RC car and establish connections to the off-board compute.	Connection with RC car established
4	Launch the software stack for all subsystems, including the RC cars, off-board compute, data aggregation unit, and external compute.	Successful launch of UI/UX visual and control interface
5	Conduct hierarchical predefined tests for each subsystem, assessing both hardware and software functionalities.	All subsystems functioning
6	Provide the operator with instructions for selecting a specific car from	Operator is able to select and

	the multiple cars available.	control the desired car
7	Instruct the operator to teleoperate the chosen car, using either BEV visualization and 3D map visualization from the UI/UX, as per their preference.	Car operated at speed 10 cm/s (PR4) and maintains 10 cm safe distance from obstacles (PR5)
8	The system indicates the parking status as completed once the task is accomplished	Successful parking detected 80% (PR2)
9	Allow the operator to explore waypoint-based manipulation of the car, where they can define waypoints for the car to navigate to autonomously.	Car operated at speed 10 cm/s (PR4) and maintains 10 cm safe distance from obstacles (PR5)
10	Repeat steps 6-9 for each additional RC car in the environment, allowing the operator to choose and control different cars.	Repeatability test
11	Document the performance of each RC car against defined requirements and validate the system's overall functionality with multiple cars in the environment.	Complete documentation

7.4.3 Success Criteria

One RC car is driven in a parking lot using the 3D map at a maximum speed of 10 cm/s [PR4] and parking is demonstrated for forward [NFR6], backward [NFR7], and (stretch) parallel [NFR8] parking scenarios. The car maintains [PR2] tolerance values while in motion and [PR3] tolerance values while stationary. The UI/UX will be deemed successful if it has a minimum update frequency of 10 Hz [PR1] and showcases BEV and 3D maps of the environment.

- Metrics for Success Criteria:
    - SC.1: Successful parking is detected 80% of the time.
    - SC.2: A safety tolerance of 10 cm is maintained at all times from obstacles.

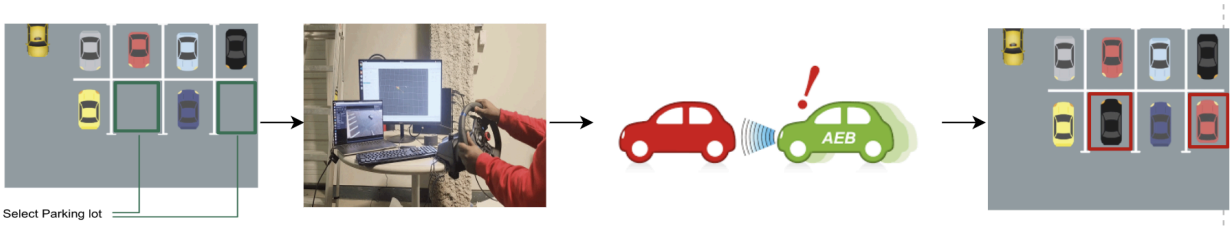

Figure 17: System Overview FVD

7.6 Budget

As of the current stage of the project, significant purchases have been made in line with the allocated budget. A total expenditure of $3108 has been utilized out of the total budget of $5000, representing 62% of the allocated funds. These expenditures encompass crucial components necessary for the development and implementation of the teleoperated parking system, including the procurement of Jetacker cars, infrastructure, and HMI equipment. Detailed breakdown of component purchases are outlined in Table 12.

Table 12: Budget		
S.No	**Item Description**	**Price**
1	Yahboom Jeston Nano Smart Robot Car Kit(R2 Standard)	$643
2	NEEWER Tripod Fluid Head	$38.15
3	8 Pieces Camera Screw Adapter	$10.49
4	Frgyee 12 Pcs 1/4" 3/8" Light Stand Adapter	$11.65
5	Lockport White Gaffer Tape 2 Inch - 2 Pack – 30 Yards	$21.19
6	Lockport Black Gaffers Tape 3 Pack	$26.34
7	NEEWER Tripod Fluid Head	$107.97
8	waveshare Power Supply for Jetson Nano	$38.97
9	SanDisk Ultra 128GB	$95.94
10	8 Pieces Camera Screw Adapter	$10.49
11	Frgyee 12 Pcs 1/4" 3/8" Light Stand Adapter	$11.65
12	Logitech G29 Driving Force Racing Wheel and Floor Pedals	$379.99
13	Beikell Memory Card Reader	$17.98
14	Memory Card Case	$6.99
15	12 Inch Traffic Training Cones	$53.98
16	12 Inch Traffic Training Cones	$26.99
17	Dell Multi-Device Wireless keyboard and mouse	$76.99
18	Dell Wireless Keyboard and Mouse - KM3322W	$24.99
19	Acer Monitor 27 inch	$109.99
20	Cat 6 Cables 100ft	$30.38
21	Anker usb C hub(6 in one)	$70.00
22	hc sr04 ultrasound sensors 10 pc	$14
23	ITD ITANDA 10FT USB Extension Cable USB 3.0 Extension Cord Type A Male to Female 15 feet	$48
24	Yahboom AI Robotic Car Chassis kit Autopilot Training Ackerman	$360
25	NVIDIA Jetson Nano Developer Kit (945-13450-0000-100)	$298
26	Arduino Mega 2560 REV3 [A000067]	$97.80
27	TP-Link Archer T2U Plus AC600 High Gain Dualband USB WLAN Adapter	$72
28	120pcs 10cm Dupont Wire Male to Female Breadboard Jumper Wires 3.9 inch 1pin-1pin 2.54mm Connector	$7

29	120pcs 20cm Dupont Wire Male to Female Breadboard Jumper Wires 7.9 inch 1pin-1pin 2.54mm	$7
30	500PCS 1 Pin Header Connector Housing for Dupont Wire Jumper Compact	$7.89
31	Lockport Black Gaffers Tape 3 Pack	$27.89
32	Lockport White Gaffers Tape 2 Inch - 2 Pack White Tape – 30 Yards	$19.88
33	10PCS/LOT Ultrasonic Sensor Mounting Bracket for HC-SR04 Smart Car AL HC-SR04 HC SR04 HCSR04 Distance Sensor (10PC hc-sr04 Mounting Bracket)	$16.98
34	Explux High-Intensity Yellow LED PAR38 Flood Light Bulb, 250W Equivalent Ultra Bright Yellow Light, Full-Glass Outdoor Waterproof & Anti-Ageing, Dimmable, Yellow Color Spotlight, 2-Pack	$45
35	Monoprice 1.5-Feet USB 2.0 A Male to B Male 28/24AWG Cable (Gold Plated) (105436),Black	$25
36	USB to USB Cable [13cm 5 inch],USB 3.0 Male to Male Type A to Type A	$40

7.7 Risk management

Risk Management is essential to identify and prepare for uncertainties that might hinder the completion of the project within the set schedule, budget or scope. It is important to continually assess possible risks and determine the likelihood and consequences of them occurring. For our project, we have identified five major risks and outlined their mitigation strategies in Table 13. These risks and their risk reduction strategies are further detailed in the Appendix.

Table 13: Risk Management Table

Risk ID	Description	Risk type	L	C	Mitigation	Severity
R1	Components not arriving on time	Schedule	3	3	Ordered from reliable vendors / ahead of time	Medium
R2	Unable to meet the budget requirements	Cost	2	4	Use existing setup	Medium
R3	Communication failure	Technical	2	5	Setup local network and use suitable protocol	High

R4	Depth scaling	Technical	5	5	Use sensor fusion techniques (LiDAR)	High
R5	Fisheye effect in BEV	Technical	3	5	Use DL based techniques	High

Table 14 indicates whether the identified risks have materialized and evaluates the effectiveness of our strategies in mitigating them. Additionally, it outlines the team members responsible for monitoring these risks.

Table 14: Risk Tracking

Risk ID	Description	Occurred?	Mitigation effective?	Tracked By
R1	Components not arriving on time	No	Yes. Ordering ahead of time has been extremely effective to mitigate this risk	Sashank
R2	Unable to meet the budget requirements	No	Usage of existing components/setup has been pivotal to stay within the budget.	Sashank
R3	Communication failure	Yes	Setting up our dedicated network helped in solving network issues.	Dhruv
R4	Depth scaling	No	We have not yet faced this risk this semester.	Shahram
R5	Fisheye effect in BEV	No	Usage of superglue and super point has been pivotal to handle such artifacts	Shahram

8. Conclusion

8.1 Lessons Learnt

During the spring semester, our team encountered and embraced valuable lessons that significantly contributed to our project's progression. Firstly, we reaffirmed the importance of effective communication among team members. Through regular meetings and open channels of

communication, we ensured that everyone remained aligned with project goals and tasks, facilitating smoother collaboration and problem-solving.

Additionally, the implementation of weekly stand-up meetings emerged as a critical practice in keeping everyone informed about individual progress, challenges, and potential roadblocks. These meetings provided an opportunity to address any issues promptly and adjust project strategies accordingly, fostering a proactive and agile approach to project management.

Another critical lesson learned was the importance of minimizing experimentation with multiple methods for the same task. For example, in our project, we initially explored various techniques for generating Bird's Eye View (BEV), including Inverse Perspective Mapping (IPM), ArUco tags, and deep learning approaches. While each method offered its advantages, we found ourselves continuously iterating and experimenting to optimize results. However, after careful evaluation, we ultimately reverted to utilizing the basic classical pipeline for generating BEV. This experience highlighted the need to balance innovation with practicality and efficiency, emphasizing the importance of committing resources judiciously and focusing on solutions that align closely with project requirements and constraints.

Moreover, the utilization of project management tools, such as Jira, proved indispensable in organizing tasks and workflows. For instance, Jira enabled us to streamline task assignment, track progress, and manage dependencies effectively. By leveraging Jira's features, we were able to allocate resources efficiently, identify bottlenecks, and prioritize tasks, ultimately enhancing overall project coordination and productivity. This scenario highlights how investing in the right tools can significantly alleviate project management complexities and streamline workflows, contributing to the project's success.

8.2 Key Fall Activities

Looking ahead to the next semester, our project is poised to advance significantly, with several strategic plans in place to enhance functionality and performance. Firstly, we aim to advance towards multi-agent systems, introducing capabilities for coordinated operation among multiple vehicles. This progression will not only amplify the scope and versatility of our teleoperated parking system but also pave the way for more sophisticated applications in real-world scenarios.

In addition to advancing multi-agent systems, we plan to integrate a parallel parking setup into our system. By incorporating this feature, we will expand the repertoire of parking scenarios our system can handle, thereby increasing its practical utility and relevance. Moreover, we prioritize enhancing safety and responsiveness, focusing on optimizing system algorithms and controls to ensure seamless and reliable operation in diverse environments and scenarios.

Furthermore, our plans entail increasing the frequency and rigor of testing iterations. By intensifying testing protocols, we aim to identify and address potential issues more efficiently, accelerating the refinement process and bolstering overall system robustness. Lastly, we aspire to elevate the complexity of the environment in which our system operates. This entails introducing more intricate and challenging parking scenarios, thereby pushing the boundaries of our system's

capabilities and fostering continuous innovation and improvement. Collectively, these strategic initiatives underscore our commitment to advancing the capabilities and performance of our teleoperated parking system, positioning it as a cutting-edge solution in the field of autonomous vehicle technology.

9. References

[1] Verbruggen, Maaike, and Vincent Boulanin. "Mapping the development of autonomy in weapon systems." (2017).
[2] Vignarca, Daniele, et al. "Infrastructure-Based Vehicle Localization through Camera Calibration for I2V Communication Warning." Sensors 23.16 (2023): 7136.
[3] Atharv, Dhanesh, Jash, Ronit, and Shreyas. "Conceptual Design Review." https://mrsdprojects.ri.cmu.edu/2023teame/78-2/conceptual-design-review. 2022.

10. Appendix

10.1 Risk Management

10.1.1 Depth sensing won't scale in the similar trend as infrastructure

Mitigation:

- Adjust the placement of LiDAR cameras
- Use higher density depth cameras (ZED)

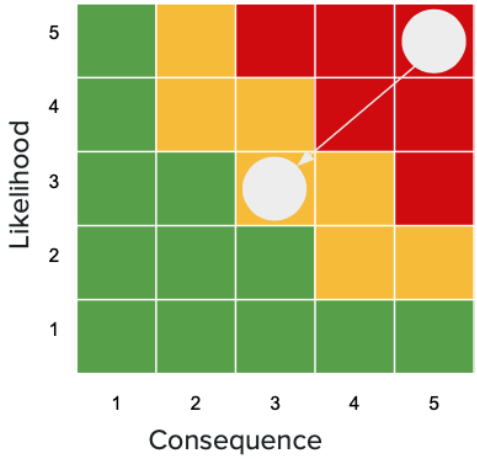

Figure 18: Likelihood-consequences graph for Risk 1.1

10.1.2 Classical methods for combining camera feeds might have fisheye effect in bird's eye view (BEV)

Mitigation:

- Use deep learning techniques

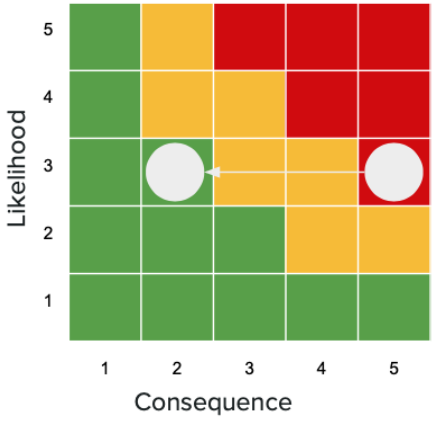

Figure 19: Likelihood-consequences graph for Risk 1.2

10.1.3 Components not arriving on time

Mitigation:

- Use more reliable vendors for sourcing components

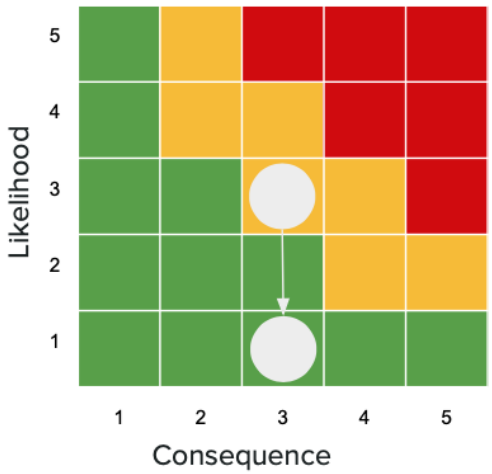

Figure 20: Likelihood-consequences graph for Risk 1.3

10.1.4 Communication failure

Mitigation:

- Setup local network and use suitable protocol

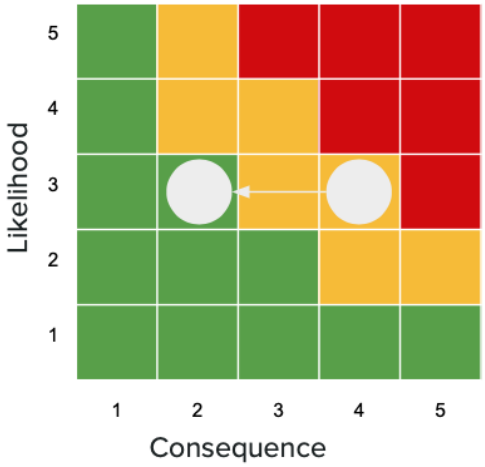

Figure 21: Likelihood-consequences graph for Risk 1.4

10.1.5 Unable to meet the budget requirements

Mitigation:

- Reuse hardware from previous teams

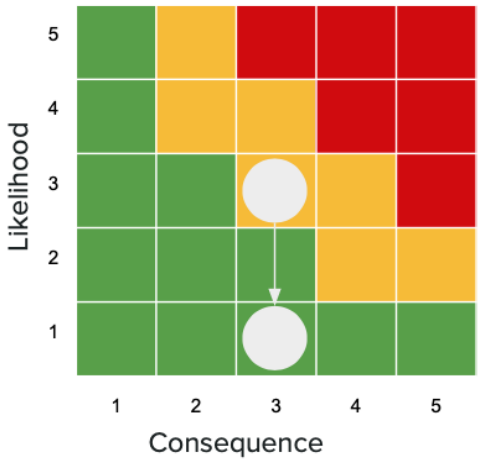

Figure 22: Likelihood-consequences graph for Risk 1.5