Software Requirements Specification (SRS)

Active Park Assist: Group 2

Authors: Timothy Guertin, Rachel Grunder, Vladimir Otchere, Jacob Lawler, Robert Sulaka

Customer: Ms. Eileen Davidson, Ford Motor Company

Instructor: Dr. James Daly

1 Introduction

This is the Software Requirements Specification document for the Active Park Assist system. This document contains subsections for the overall description, specific hardware and software requirements, modeling requirements, a discussion of the prototype, as well as a list of references used. Additionally, product functions, user characteristics, use cases, software and hardware constraints, and example scenarios for the product are among the topics that are discussed in this document.

1.1 Purpose

This document’s purpose is to restate the required functionality of the Active Park Assist system in terms of its inputs and outputs, as well as to devise criteria to fit the customer’s requirements. The intended audience for this document consists of the engineers responsible for the design and development of this system, as well as the customers and relevant stakeholders of this software project.

1.2 Scope

The Active Park Assist (APA) system is designed to make parking easier and safer for the driver. The general rationale for the Active Park Assist system is simple: automated systems have been shown to perform better than humans in tests performing various parking maneuvers. In a 2015 American Automobile Association test, “five [automated] systems tested performed better in four key areas than human drivers” [1]. One results of the test found that “compared to … AAA engineers who manually parked with the aid of a backup camera, autonomous parking systems experienced 81 percent fewer curb strikes and used 47 percent fewer maneuvers” [1]. Parking lots are also potentially busy areas with people getting in and out of cars and perhaps trying to park themselves. Due to the high level of activity surrounding them, drivers may be distracted while parking, which only adds to the risk involved. An automated system has the ability...
to be safer than a human driver in that the automated system is able to avoid distracted driving entirely.

The APA system will be able to automatically park the vehicle through the use of ultrasonic sensors -- sensors that use sound waves above the maximum frequency audible to humans to detect objects -- and front/rear facing cameras that will identify valid parking spots. Once a valid parking spot is identified the system will then proceed to guide the parking maneuver. Permission for the APA system to proceed will be authorized by the vehicle operator through the Human Machine Interface (HMI). Embedded systems will be implemented to allow for the control of automotive subsystems by the APA. The HMI will be a higher level application responsible for handling user input such as parking slot verification, authorization for the system to proceed, or user input that cancels the parking procedure.

While parking the vehicle, the APA’s cameras will identify where the lines are, and sensors figure out what is surrounding the vehicle. All of these are converted into angles to figure out the orientation, such that if the parking type selected by the user is perpendicular, then the system is looking for the vehicle to be lined up in parallel with the lines and/or other vehicles. During the automated parking process, the system continuously monitors the environment to avoid any potential accidents. During the process any input from the driver such as braking, accelerating, or steering will instantly stop the automated process and restore full control of the vehicle back to the driver. The system will keep the driver up to date on the current conditions through the use of notifications displayed on the HMI.

1.3 Definitions, acronyms, and abbreviations

Terms & Abbreviations
1. APA : Active Park Assist
2. Powertrain : consists of the engine, transmission, differential, and other aspects involved in vehicle propulsion
3. HMI : Human Machine Interface; the interface through which the vehicle operator will interact with the Active Park Assist
4. AAA : American Automobile Association
5. Ultrasonic : term referring to a sound wave that is of a frequency above the threshold of human hearing (greater than 20 kHz)
6. CAN -> Controller Area Network
7. ECU -> Electronic Control Unit
8. RAM -> Random Access Memory

1.4 Organization

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
The remaining portions of this document will go into further detail regarding the product, its use cases, specific modeling requirements, as well as a general overview of the prototype. First, the context and functions of the product will be examined. Any constraints on the hardware or software, as well as any environmental assumptions will also be brought to the reader’s attention. User characteristics will be analyzed in order to understand the target user of this system. An analysis of the specific requirements to the project will follow, as well as necessary class, state, and use case diagrams. The document will finish by explaining the necessary systems and plug-ins for the prototype, as well as its design and any other technical details.

2 Overall Description

This section will give a detailed breakdown of how the APA system exists in context within its operating environment. The functionality of this product will also be covered. User characteristics and expectations will also be defined, as well as constraints, dependencies, and the apportion of requirements.

2.1 Product Perspective

System Interface

The Active Park Assist interfaces with multiple subsystems. Because it is responsible for automating the task of parking a vehicle, it must interface with the powertrain, steering, and braking systems of the vehicle it is installed on. Because the APA system is interfacing with additional systems, it is paramount that the APA have appropriate access to those systems, and the necessary security permissions to send control data such as steering or brake input. Below is a Data-Flow Diagram that explains the context in which the APA will operate in. The APA must also ensure to sanitize the input that it feeds into those control systems, to ensure the data it is sending is not overloading the consumers of that data. For example, the APA must be sure to not overload the steering control to avoid the risk of damaging the steering system.
User Interface

The HMI’s user interface will be displayed upon a touch screen within the vehicle’s dashboard entertainment system. Therefore, the APA must also be able to display its own view on the screen, and be able to accept user input through the touch display. The HMI must also be properly abstracted away from any internet connected aspects of the dashboard entertainment system, to ensure the safety and security of the APA. A user should be able to click on buttons to approve or deny the APA’s choice of parking spot, to turn the system on and off, and to cancel the system while in the middle of a parking maneuver. A user should be able to understand the HMI’s user interface with only a few hours of training.

Another part of the APA’s user interface includes any steering or brake input initiated by the user. Though this ties into the system interface constraints, the APA must take user steering and brake inputs and send the appropriate message to the Park Control subsystem to halt parking maneuver.

Hardware Interface

The APA system will interface with many other hardware components in the vehicle, such as the powertrain, brake, and steering systems. Based on data collected by the sensors, the software will determine how it should perform the maneuver. For example, while performing a parking maneuver, the APA will send the steering system how much the wheels need to turn at a given point in time. The powertrain system will likewise make the car move and the APA system software will tell the powertrain how fast the car should be going. As the vehicle gets closer to another object, the software will send the brake system inputs on how much the vehicle should be braking. The sensors will

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
constantly be collecting data, so inputs to these systems will likely be re-adjusted as proximity to other objects changes. All these interactions will take place over a wired network within the car, most likely a Controller Area Network. All electronic control units for the different automotive subsystems will be on this network for easy access and data transfer. For security reasons, however, the APA will also have its own isolated CAN specifically for internal communication. The Audio system will need to be interfaced as well in order to allow for auditory alerts to sound if the ultrasonic sensors detect that the vehicle is too close to another object.

**Software Interfaces**

The software in the APA system includes a variety of different subsystems. The central subsystem is the Park Control system, which is responsible for interpreting data inputs from other subsystems and using it to control how the hardware systems (powertrain, brake, steering) move to achieve the maneuver. Upon interpretation of the data received by the other subsystems, the ParkControl system may also send messages to the HMI subsystem. For example, ParkControl would send a message to the HMI subsystem in the event of an obstacle that would cause the maneuver to be cancelled. The HMI subsystem would then take this message and display the appropriate information to the user to explain what is going on.

The electronic control units, or ECU’s, that provide input to the steering, braking, and powertrain systems are on the Controller Area Network, or CAN, bus. The ParkControl will also be connected to the CAN bus, in order to send control information to those ECU’s.

**Communication Interfaces**

The primary communication interface and command center is the ParkControl system. This system is responsible for sending instructions to: Powertrain Control, Brake Control, Steering Control, as well as sending instructions to the HMI system. Since most vehicles still use a Controller Area Network (CAN) bus for communication, the APA will need to be able to interface with this network to communicate with other ECU’s present in the powertrains, brakes, and steering mechanisms. Kent Lennartsson acknowledges that industrial ethernet has higher bit rates, but points out that due to the minimum required ethernet frame size, the “[extra overhead] can make data transfers less efficient, despite the technology’s impressive bit rates” [3]. Because CAN also has no need for an IP address, it also means that the APA system can be isolated from potential malicious attacks coming from the internet. The APA will still have its own isolated CAN network to ensure security and encapsulation of the system.

**Memory Constraints**

Because the data coming from the cameras and ultrasonic sensors will be in the form of a stream, storage for data is not necessary for the Active Park Assist. However, enough RAM is needed to ensure that the Park Control system can process all the incoming sensor data and outgoing instructions to ECU’s without running out of memory.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Therefore, 32GB of RAM should provide sufficient headroom for heavy processing of data from the sensors and cameras.

2.2 Product Functions

The following section describes the functions of the APA under normal operations. When preparing to use the APA, the user will drive the vehicle to within range of at least one open parking spot. When the user of the APA turns the system on, they will be prompted on the HMI touch screen to input the type of parking operation they wish the APA to perform. The APA takes that selection input, and proceeds use the front and rear racing cameras to scan the area surrounding the vehicle for open parking spots. When it has found what it deems to be a valid parking spot, the APA will prompt the user through the HMI to approve the system’s choice of a parking spot.

If the APA’s selection is approved by the user, the parking procedure will begin. The appropriate gear will be selected, and the APA will then steer the vehicle into the parking spot. The APA will be guided by its array of ultrasonic sensors until the parking procedure is complete. Upon the completion of the parking maneuver, the APA will shift the transmission into park, and return operational control of the vehicle back to the driver.

2.3 User Characteristics

All operators of APA will have a driver’s license which certifies a guaranteed level of familiarity with the operation of a motor vehicle based on requirements set by state governments. Thus, it can be expected that a user would be familiar with performing a parking maneuver. However, while it can expected that a user would have some familiarity with a graphical user interface, it must assumed that not every user will have the same skill level with regards to using such software or touch screen interfaces. Thus, it is imperative that the system be able to accommodate valid drivers of all ages.

2.4 Constraints

Software Constraints

According to CSS Electronics, the latest version of CAN, CAN FD, has a maximum bit rate of 8 Mbit/s [2], however, it cannot be assumed that all cars that will have the APA system will have type of CAN. Therefore, it must the APA has the potential to be constrained to dealing with the older 1 Mbit/s version of CAN. Fortunately, CAN FD is backwards compatible and is able to be retrofitted to older systems without much difficulty [3].

Hardware Constraints

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Because of the APA’s need for an HMI, the system is limited to newer vehicles only. Older vehicles do not have any method of receiving user input via touch screens, so they will not be able to utilize this feature. Additionally, supported vehicles must have backup cameras and ultrasonic sensors, as these are crucial to sensing the surrounding environment and ensuring safety. The backup camera will send a video feed to the HMI so that the user is able to view what is going on during the maneuver. The sensors will collect data on its proximity to nearby object and send it to the software to determine if objects are too close to complete the maneuver, or if it is possible to adjust the maneuver path.

### 2.5 Assumptions and Dependencies

One assumption is that the HMI will have access to be able to communicate with the vehicles entertainment system to display its user interface on the touch screen. As the requirements currently stand, having a touch screen available to user interaction is a dependency for the system to be able to effectively operate. Another assumption is that the APA will be able to select gears from the transmission and apply throttle control to assist in the parking maneuver. Lack of either would result in the need for human intervention in terms of gear selection or input from the accelerator pedal.

The APA is dependent upon the CAN bus network to be able to connect with the various ECU’s that control the aforementioned steering, braking, and powertrain subsystems. Because the CANbus is largely unsecured, we must also assume that any input data coming from the CANbus is potentially malicious, and therefore the data must be properly sanitized.

A final assumption is that the ultrasonic sensor data can be sent as a stream of data and interpreted as such. If not, then the potential latency would mean that the data coming from the sensors may not be accurate in real time.

### 2.6 Apportioning of Requirements

A useful feature that was brought up in discussions about how to improve on the APA system is to include an option to safely exit parking spaces. This feature would be a great addition to the original description of the system but was not requested specifically by the client. For this reason, the option to safely exit parking spaces is currently out of scope for our initial implementation of the system but would be a great addition in future implementations of the system.

### 3 Specific Requirements

1. External Interfaces

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
1.1.1. Name of item: System Interface
1.1.2. Description of purpose: Parking Automations
1.1.3. Source of input or destination of output: Powertrain, steering, and braking
1.1.4. Valid range, accuracy, and/or tolerance:
1.1.5. Units of measure: N/A
1.1.6. Timing: N/A

1.2.1. Name of item: User Interface
1.2.2. Description of purpose: Accepts user input and display information
1.2.3. Source of input or destination of output: Touch Screen
1.2.4. Valid range, accuracy, and/or tolerance:
1.2.5. Units of measure: Text
1.2.6. Timing: Must be instantaneous to accurately respond to user demands via the HMI
1.2.7. Relationships to other inputs/outputs: Steering and/or break inputs

1.3.1. Name of item: Hardware Interface
1.3.2. Description of purpose: Interfaces with powertrain, braking, and steering systems
1.3.3. Source of input or destination of output: Powertrain, steering, and braking

1.4.1. Name of item: Software Interface
1.4.2. Description of purpose: Interpret data inputs
1.4.3. Source of input or destination of output: Electronic Controlled Units (ECU’s) input and HMI subsystem output

1.5.1. Name of item: Communication Interface
1.5.2. Description of purpose: Sending Instructions to various ECU’s
1.5.3. Source of input or destination of output: Powertrain control, brake control, and steering control

2. Functions

2.1. The system shall check for valid inputs before performing any autonomous activity.
2.2. The system shall identify the parking spot, accelerate and/or brake the vehicle, choose the appropriate gear, and steer accordingly.
2.3. The system shall stop and notify the user in response to any abnormal situations.

3. Performance Requirements

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Autonomous parking should be completed in 30 seconds or less. It will handle information from the user to do so.

4. Logical Database Requirements

No database is utilized for the APA, therefore there are no requirements that apply here.

5. Design Constraints

5.1 Standards Compliance

- Must comply with all National Highway Traffic Safety Administration (NHTSA) regulations with regards to the Department of Transportation’s Automated Vehicles 3.0 (AV 3.0) Federal guidance[4].
- Also, must follow the National Conference of State Legislatures (NCSL) autonomous vehicle enacted legislation [5].

6. System Software Attributes

6.1 Reliability

- System must be as or more reliable than the equivalent human driver. The system should also be able to perform parking maneuvers consistently without error. Though it can be expected that malfunctions may occur, the error rate of the system should be kept below 1%. The system should also be able to reliability accept user input, particularly if the user inputs a command to cancel the procedure.

6.2. Availability

- The system must be available for use regardless of weather conditions. It also must be able to handle multiple types of parking situations, for example parking lots, parking structures, and parallel parking.

6.3. Security

- The system must be appropriately abstracted away from internet connected nodes within the CAN network. The APA’s internal communication should be transmitted on a secure network entirely isolated from the general car-wide CAN. Data coming from ECU’s or user input via the HMI should be sanitized to ensure malicious data entry cannot occur.

6.4. Maintainability

- The system must be able to handle updates in the electronic control units that it interfaces with, as well as the network that it operates within. It must also be able to be updated accordingly should an enhancement be made.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
4 Modeling Requirements

Use Cases

Use Case: Initiate System

Diagram:

![Diagram of Initiate System](image)

**Brief Description:**
The driver indicates on the HMI that they would like to start the Active Park Assist system, after which they are prompted to select the type of parking they wish the system to perform (perpendicular or parallel).

**Step-by-Step Description**
This use case assumes that the user has already pulled up to within a reasonable vicinity (15 feet).

1. The user switches on the APA system.
2. APA prompts the user to select the type of parking they wish the system to perform
3. User selects the parking type via buttons on the HMI screen

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Use Case: Validate Parking Spot (extends Initiate System)

Diagram:

Brief Description:
The driver indicates on the HMI that the suggested parking spot is indeed a viable space.

Step-by-Step Description:
This use case assumes that the driver has already completed the use case of initiating the APA system
1. Following the selection of the parking type, the APA scans the area for valid parkings spots
2. The APA, via the HMI, returns an image of a parking spot, and prompts the user for approval of the APA’s selection
3. User approves the parking spot selection
4. APA proceeds to park the vehicle
**Use Case:** Interrupt System (extends Initiate System)

**Diagram:**

![Diagram of Interrupt System](image)

**Brief Description:**
The driver interrupts the system by interacting with the vehicle controls, such as the brakes or steering wheel. The process is terminated and control is relinquished to the driver.

**Step-by-Step Description:**
Use case assumes that the user has completed the two previous use cases, and that the APA is currently parking the vehicle.

1. Vehicle operator gives steering or braking input, or hits cancel button the HMI’s display screen on the dashboard
2. APA applies the brakes and shifts into park upon coming to a complete stop
3. APA is disengaged and control returns to the user. The user is then prompted through the HMI to either confirm the termination of the APA, or restart the APA from the beginning.

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
Figure 2: APA Class Diagram

Class Diagram Data Dictionary

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AudioSystem</td>
<td>Plays a sound when there is an obstacle.</td>
</tr>
</tbody>
</table>

Attributes | None

Operations

PlaySound (action : string): void
Plays a sound in the audio system, depending on the context (error, complete maneuver, etc.)

Relationships
Associated with ParkControl, which tells the AudioSystem when to play a sound.

UML Extensions | None

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BrakeSystem</td>
<td>Responsible for communicating with and controlling the vehicle’s brakes</td>
</tr>
</tbody>
</table>

### Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>brakeLevel : float</td>
<td>The amount that the brake should be applied. Should be a value between 0 and 1 (a percentage)</td>
</tr>
<tr>
<td>isUserOverride : bool</td>
<td>Flag set if user interacts with brake</td>
</tr>
</tbody>
</table>

### Operations

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UpdateBraking (brakeLevel : float): void</td>
<td>Updates the brakeLevel to the parameter value.</td>
</tr>
</tbody>
</table>

### Relationships

Associated with ParkControl, which tells the BrakeSystem how much it should be braking

### UML Extensions

None
## Camera

**Element Name**: Camera  
**Description**: Collects visual data from front and rear camera positions, and communicates with VehiclePosition System

<table>
<thead>
<tr>
<th><strong>Attributes</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>CameraLocation : string</td>
<td>Distinguishes from front and rear camera</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Operations</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>GetFeed (CameraLocation : string): video</td>
<td>Given a camera location return the video feed for that camera</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Relationships</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Acts as data source for the VehiclePosition system</td>
<td></td>
</tr>
</tbody>
</table>

**UML Extensions**: None

## HMIView

**Element Name**: HMIView  
**Description**: Interface to initiate, terminate, and monitor APA operations

<table>
<thead>
<tr>
<th><strong>Attributes</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>initiateAPA: bool</td>
<td>Flag to determine APA initiation requests</td>
</tr>
<tr>
<td>userListener: bool</td>
<td>Flag to determine user interrupts</td>
</tr>
<tr>
<td>parkingType: string</td>
<td>Distinguishes parking types</td>
</tr>
<tr>
<td>spotValidated : bool</td>
<td>Flag to determine validity of parking spot</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Operations</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>initiateAPA(): int</td>
<td>Sends command to ParkControl to initiate an APA operation, and</td>
</tr>
</tbody>
</table>

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ParkControl</td>
<td>Receives commands to initiate APA from HMI. Interprets data from camera and sensor class to provide commands to SteeringControl, PowertrainControl, and BrakeControl.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attributes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>brakeValue: float</td>
<td>Calculated brake value to complete APA</td>
</tr>
<tr>
<td>steeringAngle: float</td>
<td>Calculated steering value to complete APA</td>
</tr>
<tr>
<td>torqueValue: float</td>
<td>Calculated torque value to complete APA</td>
</tr>
<tr>
<td>initialPosition: float</td>
<td>Starting position for APA operation</td>
</tr>
<tr>
<td>finalPosition: float</td>
<td>End position for APA operation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GetPosition (): float</td>
<td>Position of vehicle</td>
</tr>
<tr>
<td>Relationships</td>
<td>Sends instructions to the vehicle control subsystems (powertrain, braking, steering) to control vehicle movement</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>UML Extensions</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Element Name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>PowertrainControl</td>
<td>Receives commands from ParkControl and controls</td>
</tr>
<tr>
<td>Attributes</td>
<td>----------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>torqueValue : float</td>
<td>Determines how much torque to tell the powertrain to use</td>
</tr>
<tr>
<td>isUserOverride : boolean</td>
<td>Flag set if user interacts with accelerator</td>
</tr>
<tr>
<td>Operations</td>
<td>----------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>moveVehicle (torqueValue : float): void</td>
<td>Executes torqueValue to move vehicle</td>
</tr>
<tr>
<td>Relationships</td>
<td>Executes commands from ParkControl</td>
</tr>
<tr>
<td>UML Extensions</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReverseCameraSystem</td>
<td>Provides video feed for HMIView</td>
</tr>
</tbody>
</table>

**Attributes**

- isActive: boolean
  - flag set if the vehicle is reversing

**Operations**

- CameraFeed (): video
  - Returns video feed data

**Relationships**

- Acts as a data source for the HMIView. Is equivalent to a back-up camera on current back-up systems.

**UML Extensions**

- N/A

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor</td>
<td>Collects positional data from sensors placed along the perimeter of the vehicle.</td>
</tr>
</tbody>
</table>

**Attributes**

- SensorID: int
  - Designates what sensor data is from

**Operations**

- GetFeeds (id : int): float
  - Returns the current sensor value for a given sensor

**Relationships**

- Acts as a data source for VehiclePosition

**UML Extensions**

- N/A

---

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
<table>
<thead>
<tr>
<th><strong>Element Name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Steering Control</td>
<td>Handles all steering input for the vehicle</td>
</tr>
<tr>
<td>Attributes</td>
<td></td>
</tr>
<tr>
<td>Angle: float</td>
<td>The angle at which to turn the steering</td>
</tr>
<tr>
<td>isUserOverride: boolean</td>
<td>Flag set if user is providing external steering input to override the system</td>
</tr>
<tr>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>PerformSteer (angle : float): void</td>
<td>Rotates steering column to provide steering for the vehicle</td>
</tr>
<tr>
<td>Relationships</td>
<td></td>
</tr>
<tr>
<td>Interfaces with the Park Control subsystem, takes in steering input from the Park Control.</td>
<td></td>
</tr>
<tr>
<td>UML Extensions</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Element Name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>UserListener</td>
<td>-lists for user input via the HMI to determine if the user is attempting to cancel the Park procedure while the maneuver is being completed</td>
</tr>
<tr>
<td>Attributes</td>
<td></td>
</tr>
<tr>
<td>isCancel: boolean</td>
<td>Flag to determine if user wants to cancel procedure</td>
</tr>
<tr>
<td>Operations</td>
<td></td>
</tr>
<tr>
<td>cancelProcedure (isCancel : boolean): boolean</td>
<td>Returns to Park Control if user wants to cancel the procedure</td>
</tr>
<tr>
<td>Relationships</td>
<td></td>
</tr>
<tr>
<td>Listens to the HMI while the Park Control is performing a procedure. If user attempts to cancel the procedure, this class informs Park Control to cancel the operation</td>
<td></td>
</tr>
<tr>
<td>Element Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>VehiclePosition</td>
<td>Determine the vehicle's position within a 3D space, and determine when vehicle is parked.</td>
</tr>
</tbody>
</table>

**Attributes**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x: int</td>
<td>X coordinate.</td>
</tr>
<tr>
<td>y: int</td>
<td>Y coordinate.</td>
</tr>
<tr>
<td>z: int</td>
<td>Z coordinate.</td>
</tr>
</tbody>
</table>

**Operations**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>positionByCameraDegree () : void</td>
<td>Calculates current location based upon camera feed data</td>
</tr>
<tr>
<td>positionBySensorDegree() : void</td>
<td>Calculates current location based upon sensor feed data</td>
</tr>
</tbody>
</table>

**Relationships**

Takes data from camera and position sensors to provide position data to ParkControl. Cameras identify where the lines are, and sensors figure out what is surrounding the vehicle.
Representative Scenario

Above is a sequence diagram in which the user initializes the Active Park Assist via the HMI. The HMIView is the only class that is in this sequence, and takes the user input to start up the system.

Figure 4: Successful Parking Scenario

The first scenario is when the user successfully completes a parking maneuver through the use of the APA. The user communicates their approval via the HMIView to approve the parking space. The VehiclePosition class then calculates the appropriate values to figure out where the vehicle is, and communicates these values to the ParkControl class. The ParkControl takes in the positioning data from VehiclePosition and uses it to figure out where the vehicle is, and what sort of inputs should be sent to the steering, braking, and powertrain subsystems.
The second scenario depicts when the user wishes to cancel the APA through the HMI. Here, the user inputs their input to cancel the procedure into the HMI, which then passes it on to the UserListener which will update ParkControl and tell it to stop the procedure.

The third scenario depicts when the user approves the APA’s parking spot selection. The HMIView takes the user’s approval, and the VehiclePosition proceeds to set the coordinates of the vehicle’s location. ParkControl then takes over, and proceeds to the same steps as the successful sequence diagram above in Figure 4.
The fourth and final scenario depicts when the user declines the APA’s parking spot selection, in which the APA tries again to find a valid parking spot. The user inputs their selection via the HMIView.

The APA Parking Process state diagram shown up demonstrates the normal use cases for using the APA. When the user opts to turn on the APA system, the system will prompt the user to select the type of parking maneuver they would like the system to perform. Upon receiving that user selection, the system will then identify available parking spots, ultimately selecting one. The user will then be prompted to approve the APA’s choice of a parking spot. If the user denies the AP’s choice, the APA will revert back to the previous state and select a new parking spot. If the user does approve the spot, the system proceeds to the parking state, in which the parking maneuver is performed. If the parking maneuver is able to be completed successfully, the APA system will return control back to the driver and terminate. Should any system failure occur, such as a failure to receive sensor data, or any other warning flag raised by any one of the subsystems, the system will proceed to the Halt System state. In this state, user control of the vehicle is restored. From here, the user can either restart the APA system and begin from the beginning, or terminate the operation altogether.

5 Prototype

Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)
The prototype will demonstrate the functionality of the HMI, as well as define a user experience for interface as a whole. The prototype will show how the user will interact with the system and approve or decline parking spots selected by the APA. It will also demonstrate the process flow for how a user would cancel the APA parking procedure from the HMI.

5.1 How to Run Prototype

The prototype will be a simple web based application that will be able to run on any platform. For the first version, the prototype will consist of a simple screen with buttons to ask the user to select the parking style. Following their choice, a mock image of a parking spot will be brought up to the user. The user can then approve or decline the APA’s parking spot selection. The purpose of the prototype is to demonstrate a user’s experience in interfacing with the APA.

5.2 Sample Scenarios

The following is a sample scenario of a user interacting with the Active Park Assist:

A user pulls up to a parking space, and selects the Active Park Assist option. The user must be within 15 feet of the parking space, since that is the maximum range of detection for this system. After the user inputs their parking selection, APA scans the area for open parking spots, and returns a selection to the user. For this instance, we will say the user selects a perpendicular parking style. If the user validates the APA’s selection, the APA initiates the parking procedure. During the parking procedure, the Park Control sends the appropriate throttle, braking, and steering input to their appropriate sub-systems. Park Control also takes in input from the ultrasonic sensors to determine vehicle range from various objects potentially surrounding the vehicle. The system will look for a range of 2 feet between the vehicle and its surroundings on either side of the vehicle, and a range of 6-10 inches on the front/back. Upon detecting the vehicle’s location to be appropriately nested within the parking space, the Park Control will stop the vehicle, shift into park, and return control to the user. A message will display on the HMI informing the user that the procedure is complete.

Due to the fact that the Version 1 of this document is due before the Prototype Version 1, this document will be updated accordingly with views of the Prototype once it is complete.

6 References


Template based on IEEE Std 830-1998 for SRS. Modifications (content and ordering of information) have been made by Betty H.C. Cheng, Michigan State University (chengb at chengb.cse.msu.edu)


Website found at: https://www.cse.msu.edu/~guertint/

7 Point of Contact
For further information regarding this document and project, please contact Prof. Betty H.C. Cheng at Michigan State University (chengb at cse.msu.edu). All materials in this document have been sanitized for proprietary data. The students and the instructor gratefully acknowledge the participation of our industrial collaborators.