VIC

Vehicle Intersection Control

McMaster University

Draft Component Design

SE 4G06

Group 6

Alex Jackson	-	1302526
Jean Lucas Ferreira	-	1152120
Justin Kapinski	-	1305257
Mathew Hobers	-	1228607
Radhika Sharma	-	1150430
Zachary Bazen	-	1200979

Table of Contents

List of Tables

List of Figures

1 Revisions

Table 1: VIC Table of Revisions

Date	Revision Number	Authors	Comments
January 23, 2017	Revision 0	Alex Jackson Jean Lucas Ferreira Justin Kapinski Mathew Hobers Radhika Sharma Zachary Bazen	N/A

2 Introduction

2.1 Document Purpose

The purpose of this document is to provide a comprehensive system overview of the VIC (Vehicle Intersection Control) system. In addition, it is also intended to provide comprehensive subsystem details that will allow the system to be implemented.

2.2 Document Overview and Intended Audience

This document contains three main sections. The first section, Module Guide, provides an overview of all the system modules and a natural language description of each module. Each module description includes information on intended behaviour, inputs, outputs, initialization and timing constraints. Module Interface Specifications and Module Internal Design details are included in the second section. The third section contains system sequence diagrams.

When reading this document it will be helpful to note that there are two main components to VIC. These two components are the intersection component and the vehicle component. Each section is organized with this division.

The intended audience for this document is Sean Marshall (the engineering team leader at GM) who proposed the problem, Dr. Alan Wassyng and the teaching assistants as supervisors of the project, and ourselves as designers of the system.

2.3 Definitions

Table 2: Definitions

VIC	The entire system including the intersection controller, the vehicles, and their corresponding controllers.
IC	The Intersection Controller is the system that tracks the arrival and departure of the vehicles, as well as determining the order in which the vehicles must proceed through the intersection.
VC	The Vehicle Controller is the system that will allow the 1/10 scale RC car to follow lanes, maintain a desired speed, steer itself, and send requests to the intersection controller.

2.4 Naming Conventions

Table 3: Naming Conventions

m_ic_variableName	Monitored variable for intersection controller
c_ic_variableName	Control variable for intersection controller
m_vc_variableName	Monitored variable for autonomous vehicle controller
c_vc_variableName	Control variable for autonomous vehicle controller
ICD#	Intersection Controller Design Component ID
ICM#	Intersection Controller Module Guide ID
VCD#	Vehicle Controller Design Component ID
VCM#	Vehicle Controller Module Guide ID

3 Module Guide

3.1 Module Overview

3.1.1 Intersection Controller Modules

ID	Name	Responsibilities	Secrets
ICM.1	TrafficController	Determine order of car progression	Scheduling algorithm
ICM.2	VehicleDetection	Know when a car is on top of one of the intersection sensors, and the corresponding sensor	Relationship between magnetic sensor and car
ICM.3	Communication	Interpret receiving car signals and sending signals to a car	Communication protocol
ICM.4	IC_Main	Control information flow of intersection controller	Manages intersection modules

3.1.2 Vehicle Controller Hardware Modules

ID	Name	Responsibilities	Secrets
VCM.1	PWM	Convert a software signals to a physical PWM signal	How to convert a software value to PWM
VCM.2	SpeedConverter	Convert wheel rotation count to a speed value	Speed calculation algorithm
VCM.3	ServoController	Set a physical wheel angle	How to convert a software value to a PWM (Pulse Width Modulation) signal
VCM.4	MotorSpeedController	Control PWM signal	How to convert speed into a PWM signal

3.1.3 Vehicle Controller Software Modules

ID	Name	Responsibilities	Secrets
VCM.6	ImageProcessing	Interpret image into environment state	Image processing algorithm
VCM.7	VehicleNavigaton	Control the navigation of the car	How the car navigates on the track
VCM.8	Communication	Interpret signal from Intersection Controller. Prepare and send signal to the Intersection Controller	Communication Protocol
VCM.9	VC_Main	Control information flow of the car	Manage car modules

ICM.1 - TrafficController

Behavioural Description

Will receive input from IC_ Main only when IC_ Main have input from Communication and Vehicle Detection. Traffic Controller will then assess the input and decide if the vehicle may proceed or not. If more than one car is approaching the intersection, Traffic Controller will notify the vehicles whether they may proceed or not based on the intersection conditions.

Inputs The module will get information about the current intersection state, and the list of cars dictating which direction they are coming from, and where they are going.

Outputs
TrafficController will output whether or not the vehicle can proceed

Initialization Description
TrafficController will be initialized once, when the IC_Main has been initialized. It does not require any specific initialization procedure.

Derived Timing Constraints
The TrafficController will have a soft deadline of 1 second, however a hard deadline will not be enforced.

ICM.2 - VehicleDetection

Behavioural Description
Vehicle Detection will make use of a camera to view the intersection, from a bird's eye view. It will create and update a data structure which will hold the state of the intersection.

Inputs
This module will have no inputs from other modules.

Outputs
VehicleDetection will output a data structure containing the state of the intersection.

Initialization Description
IC_Main will initialize this module. Upon initialization, VehicleDetection will turn on the web camera.

Derived Timing Constraints
Time constraint for VehicleDetection must be bounded by the time constraint set by the IC_Main.

ICM.3 - Communications

Behavioural Description It will be responsible for the two-way communication from the car to the intersection controller. It will create a list of communications coming in and coming out.

Inputs
A Bluetooth signal containing information from the requesting vehicle. Such as the car the orientation and destination of the car, and communication information.

Outputs
The module will have two outputs. One will the information of inbound cars to the IC_Main, the other will be a response to the cars that it can proceed or not.

Initialization Description
The inbound and outbound list of cars will be initialized empty, and the module will remain running continuously.

Derived Timing Constraints

Limited to the speed of the Bluetooth connection.

ICM.4 - IC_Main

Behavioural Description
It will be responsible for controlling the flow of information between all components of the intersection controller. Also maintains a list of the current traffic state, and updated accordingly.

Inputs
Inbound communication from the Communications module, the current state of the intersection from the VehicleDetection module, and the response from the TrafficController.

Outputs
The output will be cars going into the outbound list of cars that may proceed through the intersection.

Initialization Description
Ensure the list of active cars in the intersection is cleared. It also initializes all other modules connected to it.

Derived Timing Constraints

The timing constraint is based on the speed of the car, and the length of the intersection. In order to detect when a car has entered and left the intersection, we must process the current intersection state within this given time.

$$processing\_Time = \frac{intersection\ Segment\ Length}{car\ Speed}$$
$$processing\_Time = \frac{\frac{1}{6} * 0.6\ m}{1.4\ m/s}$$
$$processing\ Time = 0.07\ seconds$$

3.2 Vehicle Software Module Description

VCM.6 - ImageProcessing

Behavioural Description The image processing module of the vehicle controller is responsible for detecting lane positioning and obstacles. Once these factors have been detected it must produce digital information pertaining to the state of the track as seen in the current image. It must then pass this information on to the vehicle navigation module which will make appropriate adjustments.

Inputs
Webcam video stream.

Outputs
Digitally interpreted information regarding the state of the car with respect to the track.

Initialization Description
The image processing module will be initialized once the car has been turned on. It does not have any specialized initialization procedure.

Derived Timing Constraints
The time constraint for the imageProcessing is bounded by the time constraint in VC_Main.

VCM.7 - VehicleNavigation

Behavioural Description The vehicle navigation module of the vehicle controller is responsible for ensuring that the vehicle follows the correct lane on the track, stops at intersections, and avoids obstacles. This module will apply any necessary adjustments in trajectory to ensure that these responsibilities are met.

Inputs
Information regarding the current state of the car with respect to the track.

Outputs
Adjustments to the vehicle's steering, acceleration, and braking necessary to continue following the track.

Initialization Description
The vehicle navigation module will be initialized once the vehicle has been turned on. The vehicle's steering angle, speed, and acceleration with be initialized at zero (i.e. standing still with the steering pointed directly forwards).

Derived Timing Constraints
The vehicle navigation module has negligible processing time constraints.

VCM.8 - Communication

Behavioural Description Responsible for establishing communication from the car to the intersection controller, when it begins to approach an intersection. It will also receive the response from the intersection controller if it can proceed through the intersection.

Inputs
The state of the car, such as the current orientation and intended direction.
Bluetooth signal from the intersection controller.

Outputs
One output will be the information about the car to the intersection communication module. The other will be the 'go-ahead' signal to the car controller.

Initialization Description
Will be initialized when the car is initiated, and paired to the intersection controller.

Derived Timing Constraints
Limited to the speed of the Bluetooth connection.

VCM.9 - VC_Main

Behavioural Description This module is responsible for maintaining an active loop while the car is in operation. It will gather information from the ImageProcessing and Communication modules, and transpose that information to the VehicleNavigation module to make certain decision.

Inputs Parsed Bluetooth signals from the car Communication module.
Digital information pertaining to the state of the car with respect to the track.

Outputs A request for the car Communication module to signal the intersection controller.

Initialization Description Upon initialization, VC_Main will start up the VehicleNavigation module, the

ImageProcessing module and the Communication Module.

Derived Timing Constraints The exact car speed is unknown, but it will be estimated to be 1.4 m/s, and we will require to get an update on the track status every 3cm to maintain accuracy. Therefore, at this speed and this update rate, the IC_Main must process all necessary information within the following time:

$$\begin{aligned} processing\ time &= \frac{distance\ update\ rate}{car\ speed} \\ processing\ time &= \frac{0.03\ m}{1.4\ m/s} \\ processing\ time &= 0.02\ seconds \end{aligned}$$

If one instance of a loop iteration has not completed within this given time, the ImageProcessing module will be called immediately. In order to maintain an accurate position on the track.

3.3 Vehicle Hardware Component Description

Design Notes
Hall Effect sensors were chosen to measure the speed of the vehicle for the simplicity in implementation and cost.

VCM.1 - PWM

Inputs
The duty cycle and frequency for the PWM.

Outputs
Physical PWM signal on a GPIO pin.

VCM.2 - SpeedConverter

Inputs
Signal from Hall Effect sensor mounted next to wheel.

Outputs
The approximate speed of the vehicle.

VCM.3 - ServoController

Inputs
The desired angle for the servo in software.

Outputs
A physical signal on a GPIO pin telling the servo to go to the desired angle.

VCM.4 - MotorSpeedController

Inputs
The desired speed of the vehicle in software.

Outputs
A physical signal on a GPIO pin telling the motor to go at a specific speed.

4 Module Specifications

4.1 Intersection Module Interface Specification

Table 19: ICM.1 - TrafficControl

ICM.1 - TrafficControl	
TrafficControl()	Constructor to initialize the scheduling algorithm
getTrafficPermission(Queue<cars>, intersectionState[ ]) : Boolean	When called, it will evaluate all active cars and the current intersection state, then make a decision whether the most recent car can proceed or not.

Table 20: ICM.2 - VehicleDetection

ICM.2 - VehicleDetection	
VehicleDetection()	Constructor to initialize the detection of vehicles at the intersection.
getIntersectionState( ) : Boolean[ ]	Returns the state of the intersection when the function is called. Returns an array of boolean values signifying where a car is positioned in the intersection.

Table 21: ICM.3 - Communication

ICM.3 - Communication	
RecieveRequest() : Request	Function to allow the controller recieve a request to be scheduled from the car.
SendResponse(car c) : void	Function that allows the intersectrion to send a car the response to proceed through the intersection.

Table 22: ICM.4 - IC_Main

ICM.4 - IC_Main	
main() : void	Main Function for VIC.

4.2 Intersection Module Internal Design

Non-Primitive Types for the Intersection Controller

Car
Holds information pertaining to car requests to the intersection.

Fields	**Type**
direction_from	string

direction_to	string
socket	int
thread_info	Thread

ICM.1 - TrafficControl

This section does not have any private methods or objects. Functionally it will run as an algorithm. The decision to make it a separate class was made to reduce the complexity of the IC_Main module. Currently, there are no predicted changes for the Module Interface Design for this module.

ICM.2 - VehicleDetection

Dependancies

- opencv/core.hpp
- opencv/imgproc.hpp
- opencv/highhui.hpp
- stdio.h
- math.h

Constants

Name	**Type**	**Description**
CAMERA_ID	int	integer value to access the webcam

Access Methods

Name	**Parameters**	**Description**	**Return Type**
update_intersection_s	none	Continuously convert images from the webcam and represent them as boolean values in an array	none

ICM.3 - Communication

Dependancies

- bluetooth
- deque
- threading

Constants

Name	**Type**	**Description**
HOST_ADDRESS	string	Stores the host bluetooth address

Objects, Macros, Structs, and Types

Name	Defined By	Description
arrival_buffer	ArrayList<Car>	Provides arrival buffer for multiple car requests

Access Methods

Name	Parameter	Description	Return Type
get_last_car	none	returns car at top of the communication buffer	Car

ICM.4 - IC_Main

Objects, Macros, Structs, and Types

Name	Defined By	Description
trafficQueue	ArrayList<Car>	Array-List containing the cars that wish to be scheduled

Access Methods

Name	Parameter	Description	Return Type
remove_queue	int[]	Removes cars from the traffic queue at the given indicies	none
add_queue	Request	Creates car object from request and adds it to the traffic_queue	none
check_intersection_state	Boolean[]	Determines which cars have left the intersection	int[]
run	none	Method that will facilitate the passing of information to other modules. Will determine when cars should be added or removed from the queue.	none

4.3 Vehicle Module Interface Specifications

Design Notes

Please see section 4.4 for the definitions of the non-primitive types.

Table 36: VCM.6 - ImageProcessing

VCM.6 - ImageProcessing	
get_lane_status() : ImageData	Function to capture images of the track environment from a webcam and process it into information that can be analysed by software. Will return a defined type of ImageData.
test_camera() : bool	Confirm the connected webcam is functional

Table 37: VCM.7 - VehicleNavigation

VCM.7 - VehicleNavigation	
update_navigation(struct *ImageData, struct *CarStatus)	Function to signal to the vehicle if there is a change in the navigation, and if so, what changes should be made.

Table 38: VCM.8 - Communication

VCM.8 - Communication	
send_request(struct *SignalRequest) : bool	Function to allow the car to send a request to the intersection controller. Will return true if request was successfully sent.
recieve_response() : struct* SignalResponse	Function to allow the vehicle to receive a response from the intersection controller.

Table 39: VCM.9 - VC_Main

VCM.9 - VC_Main	
main() : int	Function to initiate the vehicle controller.
init() : void	Initiate any necessary modules
run() : int	Enter the program main loop.

4.4 Vehicle Module Internal Design

Non-Primitive Types for the Vehicle Controller

ImageData
Holds information pertaining to the data gathered from the image processed by the ImageProcessing module.

Fields	**Type**
avg_left_angle	double
avg_right_angle	double
left_line_length	double
right_line_length	double
intersection_distance	double
intersection_detected	bool
obstacle_detected	bool

CarStatus
Holds information pertaining to the data about the car.

Fields	**Type**
car_id	int

current_speed	double
current_wheel_angle	double
intersection_stop	bool
obstacle_stop	bool
current_lane	int

SignalRequest
Holds information required for the car communication module to establish a connection and send message to the intersection controller.

Fields	**Type**
port	int
hostname	string
host_address	string
message	string
message_size	int
keep_alive	bool

SignalResponse
Holds information retrieved from the signal sent from the intersection controller to the car communication module.

Fields	**Type**
hostname	string
host_address	string
message	string
message_size	int

VCM.6 - ImageProcessing

Dependencies

- opencv/core.hpp
- opencv/imgproc.hpp
- opencv/highui.hpp
- stdio.h
- vector
- algorithm
- math.h
- image_processing.h

- vic_types.h

Constants

Name	Type	Description
DEFAULT_CAMERA_ID	int	Integer value to access the webcam
MIN_LINE_DISTANCE	double	Minimum distance between two lines for the Hough Transform

Objects, Macros, Structs, and Types

Name	Defined By	Description
ImageData	vic_types.h	Hold information about the captured image of the track
Point	OpenCV	Describes a two dimensional point with fields x and y
Vector	vector	Many vectors will be used to maintain collection of certain elements
VideoCap	OpenCV	Provides an API for handling image and video capturing
Mat	OpenCV	n-dimensional array for representing image data

Access Methods

Name	Parameters	Description	Return Type
lane_status	none	Evaluate and convert captured image into useful information about the state of the car on the track	ImageData
test_camera	none	Test whether the camera attached to the car is functional, return true if it is	bool
get_midpoint	Point a Point b	Find the midpoint of a line connecting two points	Point
calculate_avg_angle	vector<Point> vec Point center	Find the average angle between a list of points with respect to a center point	double

VCM.7 - VehicleNavigation

Dependencies

- stdio.h
- algorithm
- math.h
- vehicle_navigation.h
- vic_types.h

Constants

Name	Type	Description
MAX_SPEED	double	Maximum allowed speed of the car
MAX_ANGLE	double	Maximum allowed wheel rotation angle

Objects, Macros, Structs, and Types

Name	Defined By	Description
ImageData	vic_types.h	Hold information about the captured image of the track
CarStatus	vic_types.h	Hold information about the car

Access Methods

Name	Parameters	Description	Return Type
get_speed	none	Returns the current speed of the vehicle	double
get_angle	none	Returns the current steering angle of the vehicle	double
calculate_angle	ImageData img	Calculate an appropriate steering angle based on the ImageData information	double

VCM.8 - Communication

Dependencies

- stdlib.h
- bluetooth.h
- rfcomm.h
- communications.h
- vic_types.h

Constants

Name	Type	Description
CAR_HOSTNAME	char*	hostname of the car
INTERSECTION_HOSTNAME	char*	hostname of the intersection controller

Objects, Macros, Structs, and Types

Name	Defined By	Description
SignalRequest	vic_types.h	Information for the signal to sent to the intersection controller
SignalResponse	vic_types.h	Information for the signal received fromthe intersection controller

Access Methods

Name	Parameters	Description	Return Type
send_request	SignalRequest	Send signal request to the intersection controller	bool
receive_response	none	Receive signal fromthe intersection controller	Signal Response

VCM.9 - VC_Main

Dependencies

- stdlib.h
- communications.h
- image_processing.h
- vehicle_navigation.h
- vic_types.h

Constants

Name	Type	Description
CAR_ID	int	car identification value

Objects, Macros, Structs, and Types

Name	Defined By	Description
ImageData	vic_types.h	Hold information about the captured image of the track
CarStatus	vic_types.h	Hold information about the car

Access Methods

Name	Parameters	Description	Return Type
main	none	Function to initiate the vehicle controller.	int
init	none	Initiate any necessary modules	int
run	none	Enter the program main loop	int

4.5 Vehicle Hardware Module Interface Specification

Table 61: VCM.1 - PWM

VCM.1 - PWM	
set_PWM(pin_number, duty_cycle) : void	Used to output a PWM signal to a given GPIO pin at a given duty cycle.

Table 62: VCM.2 - SpeedConverter

VCM.2 - SpeedConverter	
get_speed( ) : double	Returns the current speed of the vehicle as measured by the Hall Effect sensor.

Table 63: VCM.3 - ServoController

VCM.3 - ServoController	
set_angle(angle : double) : void	Outputs a physical signal to the servo to go to the specified angle.

Table 64: VCM.4 - MotorSpeedController

VCM.4 - MotorSpeedController	
set_speed(speed : double) : void	Outputs a physical signal to the motor to go at a specified speed.

4.6 Vehicle Hardware Module Internal Design

VCM.1 - PWM

Generate PWM signals on GPIO pins of the Raspberry PI.

Variables
None.

Methods

set_PWM(pin_number, duty_cycle) : void	Used to output a PWM signal to a given GPIO pin at a given duty cycle.

VCM.2 - SpeedConverter

Convert the physical signal from the Hall Effect sensor mounted next to the wheel into a velocity.

Variables
None.

Methods

get_speed( ) : double	Returns the current speed of the vehicle as measured by the Hall Effect sensor.

VCM.3 - ServoController

Generate physical signals that will control the steering servo of the vehicle.

Variables

current_angle : double	Stores the current direction of the servo so it can continuously send the correct angle to the servo.

Methods

set_angle(angle : double) : void	Outputs a physical signal to the servo to go to the specified angle.

VCM.4 - MotorSpeedController

Generate physical signals that will control the motor of the vehicle.

Variables
None.

Methods

set_speed(speed : double) : void	Outputs a physical signal to the motor to go at a specified speed.

5 Scheduling

Figure 1: Intersection Component Sequence Diagram

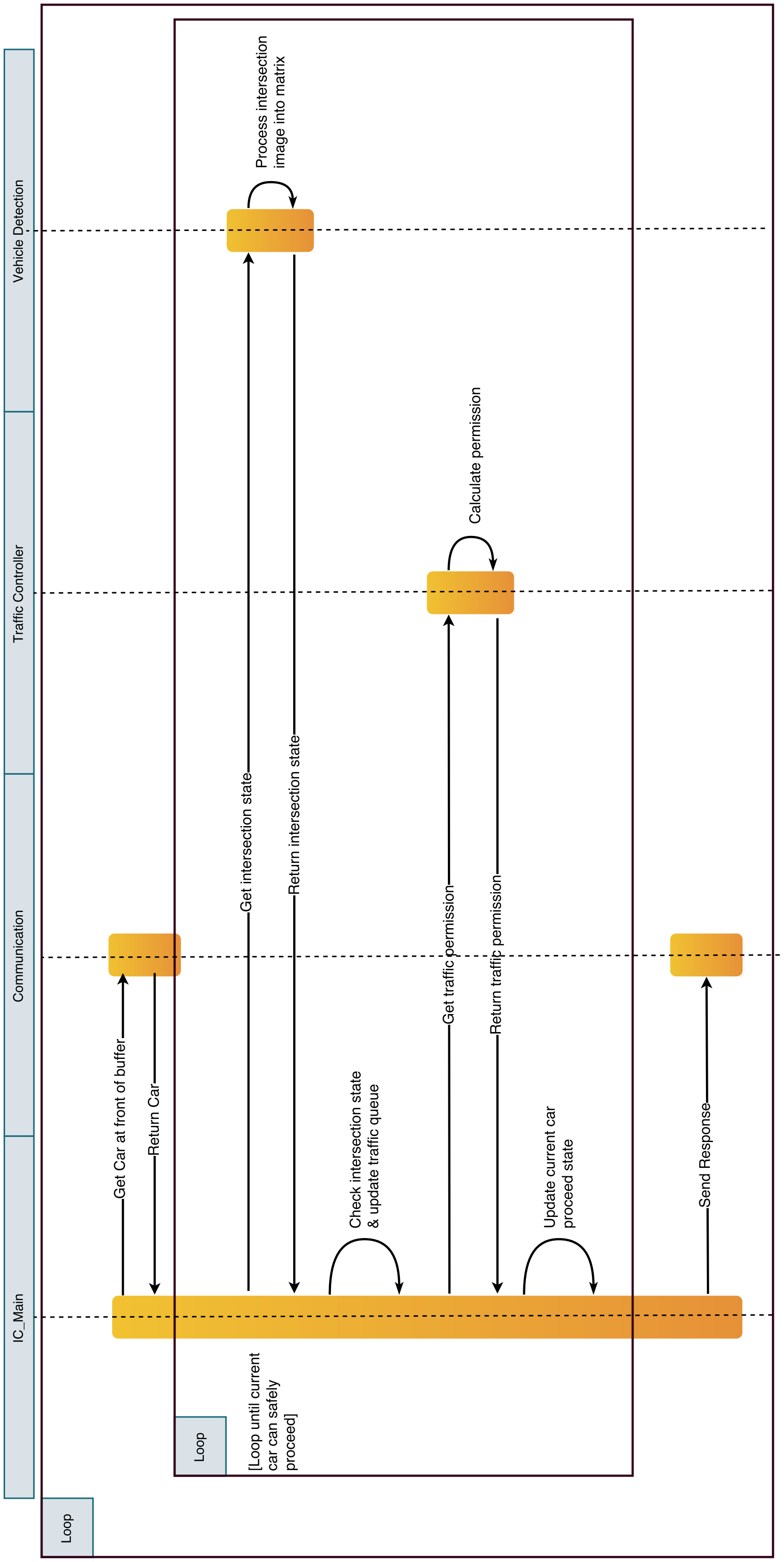
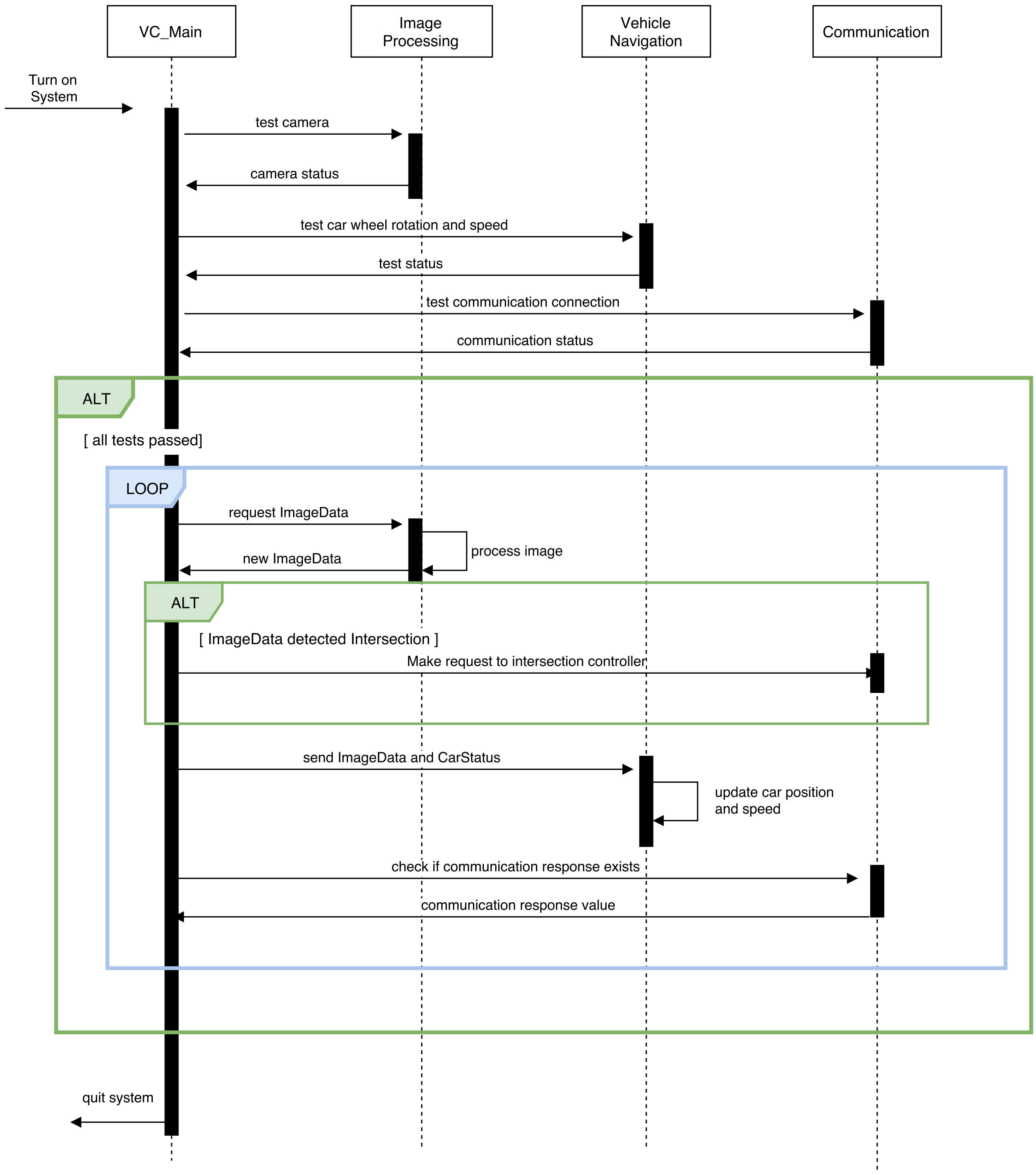

Loop
Loop
[Loop until current car can safely proceed]
IC_Main
Communication
Traffic Controller
Vehicle Detection
Get Car at front of buffer
Return Car
Get intersection state
Process intersection image into matrix
Return intersection state
Check intersection state & update traffic queue
Get traffic permission
Calculate permission
Return traffic permission
Update current car proceed state
Send Response

Figure 2: Intersection Component Sequence Diagram


VC_Main
Image Processing
Vehicle Navigation
Communication
Turn on System
test camera
camera status
test car wheel rotation and speed
test status
test communication connection
communication status
ALT
[all tests passed]
LOOP
request ImageData
process image
new ImageData
ALT
[ImageData detected Intersection]
Make request to intersection controller
send ImageData and CarStatus
update car position and speed
check if communication response exists
communication response value
quit system