Software Requirements Specification

ARC - Autonomous RC

Tao Chen, Cierra Shawe, Daniel Stoyer

Version 0.1

November 21, 2016

CONTENTS

1 INTRODUCTION

This section provides a scope description and overview of everything included in this SRS document. The purpose of this document and a list of abbreviations and definitions are provided.

1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the ”Autonomous RC System” or ARCS. The purpose and declaration for the development of ARCS will be explained. This document will also explain system constraints, interface decisions, and interactions with other external applications and hardware. This document is primarily intended to be a customer proposal for approval and a development team reference for the first version of the system.

1.2 Scope

ARCS is a software-hardware interface designed to retrofit RC cars for autonomous operation, using commodity hardware. The software and hardware specifications should be available free to download and modify at the users will.

Users’ should be able to purchase and install the specified hardware, and implement a version that can autonomously navigate to a given destination, or a within a pre- defined space.

The software will need to be installed on specific hardware, and flashed onto the main processing unit, along with control software on a computer or tablet. Hardware that we expect to need is a base station that includes a transceiver, an RC retrofitted with the main processing unit, a transceiver to send and receive information, a controller to send signals to sensors and actuation devices (motors, servos, etc...) and a vision system to aide in navigation.

ARCS will be expected to be able to receive input from a user base station, and react within the environment based on a destination that the system receives. It should also be able to navigate to the destination without user intervention, as fast as possible.

1.3 Definitions, Acronyms, and Abbreviations

1) **IMUs:** Inertial measurement unit. Used to measure acceleration, angular acceleration, and orientation of the vehicle.
2) **Operator/User:** The person who is giving commands such as destination to the system.
3) **Protocol:** Defines the data format to be transferred.
4) **Telemetry Data:** Data that contains the status information of the vehicle, such as speed, temperature, location, battery, etc.
5) **Emergencies:** Emergencies include:
   - The vehicle drifts off course significantly.
   - Vehicle flips upside down.
   - On-board components fall off.

- Etc.

6) **Visual Unit:** Visual components that provide image streams to the primary computer. The primary computer will extract information from the images, such as road condition, obstacles, and depth.
7) **Actuators:** Motors and servos.
8) **Initialization:** From the system perspective, the initialization process will include self-checking and sensor calibration. From the user's perspective, it contains making sure all the hardware is attached, battery is at least 80% full, and giving the system a destination.
9) **Success:** The vehicle successfully navigates itself to the destination.

1.4 References

1.5 References

[1] I. S. E. S. Committee, "Ieee std 830-1998, ieee recommended practice for software requirements specifications," http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=720574, published: October 20, 1998.

1.6 Overview

The remainder of this document includes three sections and the appendixes.

Section two provides an overview of system functions and intersystem interaction. System constrains and assumptions are also addressed.

Section three provides the requirement specification in detailed terms and system interface descriptions. Different specification techniques are used in order to specify requirements for different audiences.

Section four priorities requirements and includes motivation for the chose prioritization and discusses why other methods were not chosen.

Appendixes at the end of the document include results of the requirement prioritization and a release plan based on the requirments. [1]

2 Overall Description

This section provides a system overview. The system will be explained in its context to create a better understanding of how the system interacts with other systems and introduce the basic system functionality. This section will also describe what types of stakeholders will use the system and what functionality is available for each type. Lastly, constraints and assumptions for the system will be presented.

2.1 Product Perspective

The ARC system (ARCS) will be designed to integrate into an RC car using commodity hardware, and open to anyone who is interested in using it. This makes ARCS a component of a larger system, namely the RC car.

ARCS will consist of three parts:

1) Base-station used for user interaction
2) Hardware attached to RC car to be able to connect to base-station and operate the vehicle
3) Software implementation for control and communication between hardware and software

In order for the user to interact with the vehicle, commands will be sent via some form of receiver to the car. This will need to be done via a base-station that has software able of providing a way to communicate with the receiver, which then transmits data to the receiver on the car, which is then handled by the on-board computer. The control flow is described in figure 1.

2.1.1 System interfaces

There are a total of 5 system interfaces where the system can communicate with the outside world.

1) Sensors: Sensors will have a two-way communication with a secondary computer unit, where filtering and smoothing will happen before reliable data will be passed to the primary computer. The programs that reside in the secondary unit will utilize various methods to generate reliable results based on the raw data. At start-up, there will be a script executed by the system to correctly configure and calibrate each sensor. Sensors may include: battery sensor, temperature sensor, GPS, speed sensor, and IMUs.
2) Radio: This is the portal of the system where operator/user can monitor the status of the vehicle. Different protocols will be implemented for telemetry data, which will be displayed to the operator/user. This portal also allows operator/user to take control over the computer in case of emergencies.
3) Visual unit: This is the interface where the visual unit can pass streams of images to the system.
4) Actuator: The system issues commands to the motors and servos via this interface.
5) User interface: This interface is a different interface than the radio interface even though they both allow humans to interact with the system. The user interface will be disable after the vehicle starts maneuvering. This interface allows user to input operation modes and desired destinations into the system.

2.1.2 User interfaces

The user interface is a simple, concise GUI. Anyone who knows how to operate a mouse will interact with the interface with no problem. A map makes it easier for users to pin point destinations and view the current location of the vehicle. An error messages will be generated if destination is out of range, meaning that with the onboard battery the vehicle won't be able reach the desire destination. If the direct distance between the vehicle and the station exceeds the maximum radio range, an error message will be generated as well.

With proper training (in less than 30 minutes), one can understand all the indicators of the system to know the status of the vehicle.

The GUI is a single window/page arrangement. A large portion of the window is dedicated to the map. A little section at the bottom is the indicators. Error messages will directly appear on the map with alarm to warn the user.

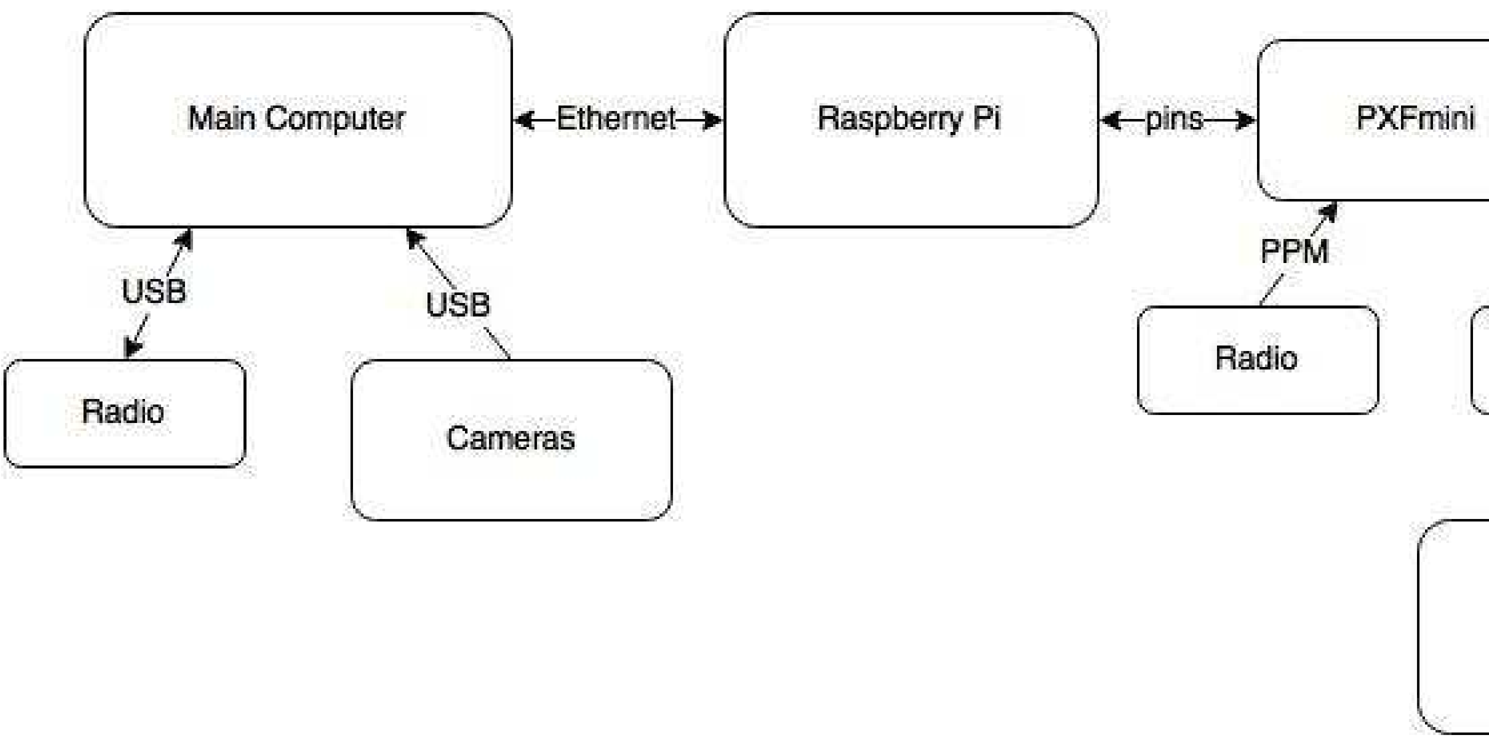


Fig. 1. Block diagram of hardware flow

2.1.3 Hardware Interfaces

We currently have two hardware systems.

1) Vision processing
2) Telemetry collection and actuator control

Vision processing, from either a stereoscopic or depth camera, will be interfaced through an i5 or better processor from an on board computer using a control board, specifically the PXSF on top of a Raspberry Pi or UP board, will be used as an interface between a computer that is doing high level computations, and the servos, motors, and other telemetry devices. And they should be able to do these things....

2.1.4 Software Interfaces

- We require three operating systems. One for a remote PC for user input, one for the primary computer for on-car data analysis, and one for the secondary computer for on-car control.
- We require a user interface to be able to give the car destination commands.
- We require software that takes input from hardware, such as video cameras, and analyze the data
- Sensor analysis software needs to be able to read data from sensors, such IMUs, and generate usable information to pass on to
- Path finding software needs to be able to integrate and analyze data from mapping and localization software products and determine a path to follow in real time.
- The primary and secondary computers will need to talk with one another. The primary computer will need to receive data from the secondary computer, analyze the data and send corresponding commands back to the secondary computer. The secondary computer will need to receive commands from the primary computer, and send data to the primary computer.
- A software interface will required on the secondary computer to convert commands received from the primary computer to instructions usable by pre-existing RC car on-board controller.
- We will require separate interfaces between the secondary computer and the visual/spacial sensors, GPS, speed sensor, and the IMU.

2.1.5 Communications Interfaces

- Radio communication will required to be able to send commands to the car and to be able to receive feedback from the car. The radio frequency will be in 27 MHz or 49 MHz range. We need software that allows us to send and receive radio signals to and from the car.
- LAN communication will be used to transmit data between the on-car computers. Existing network protocols will be utilized to perform the transmissions.
- A software interface will be required to send the commands from the secondary computer to the pre-existing RC car on-board controller.

2.1.6 Memory constraints

Since we are unsure of what hardware will be used, we are not able to set any constraints on the memory requirements that the system has to meet.

2.1.7 Operations

In most cases, operations of the system will be isolated from the user/operator, excluding emergencies and initialization.

1) At initailization, user/operator has to thoroughly check the body of the vehicle, making usre that all components are firmly attached as well as the bettary level is at least 80%;
2) Destination will be given by the user at the station during initialization;
3) User is allowed to take over control from the system in case of emergencies via the remote control;
4) User/operator is resposible for monitoring the status of the system, such as battery level, speed, location, distance, radio signal strength, pre-planned path, etc.

2.1.8 Site Adaptation Requirements

For research and prototype purposes, no site adaptation requirements are relevant.

2.2 Product Functions

This system utilizes resources from more than one computers and controls multiple actuators. The system is designed to minimize user interference during operations. However, monitoring is strictly required. Maintainability of the system is not considered to be user-oriented, which means user/operator should not be worried about maintaining the system. Only people who work on this project or people with extreme confidence should alter the system. Otherwise, it might break.

2.3 User Characteristics

Our system should be able to be built and operated by a user with at least two years of university engineering coursework and one two years, a user with over five years of technical experience through work in either an electrical or software industry, and a RC hobbyist with over five years of experience.

2.4 Constraints

Without a list of specific hardware to be used for the project, constraints are still fuzzy at the moment. Nonetheless, we will try our best.

1) Real time image processing will limit the vehicle's ability to maneuver through obstacles.
2) Telemetry data will be slightly delayed due to long distance. There is not solutions to this issue.
3) The vehicle's natural structure will inevitably cause more uncertainty on predictions, which requires us to use more complicated algorithms.

4) Hardware may fall off during operation which will force the system to halt. Structurally more concrete design to hold the hardware is needed, which will potentially reduce battery life.
5) Telemetry data transfer are limited to radio during outdoor operations.
6) The success rate are set to 80% and above.
7) For research purpose only, the vehicle is limited to only operate in a closed area at low speed and an open area with few or no humans and animals at high speed.

2.5 Assumptions and Dependencies

2.6 Apportionment of Requirements

Fig. 2. ARC Fall Schedule Gantt Chart

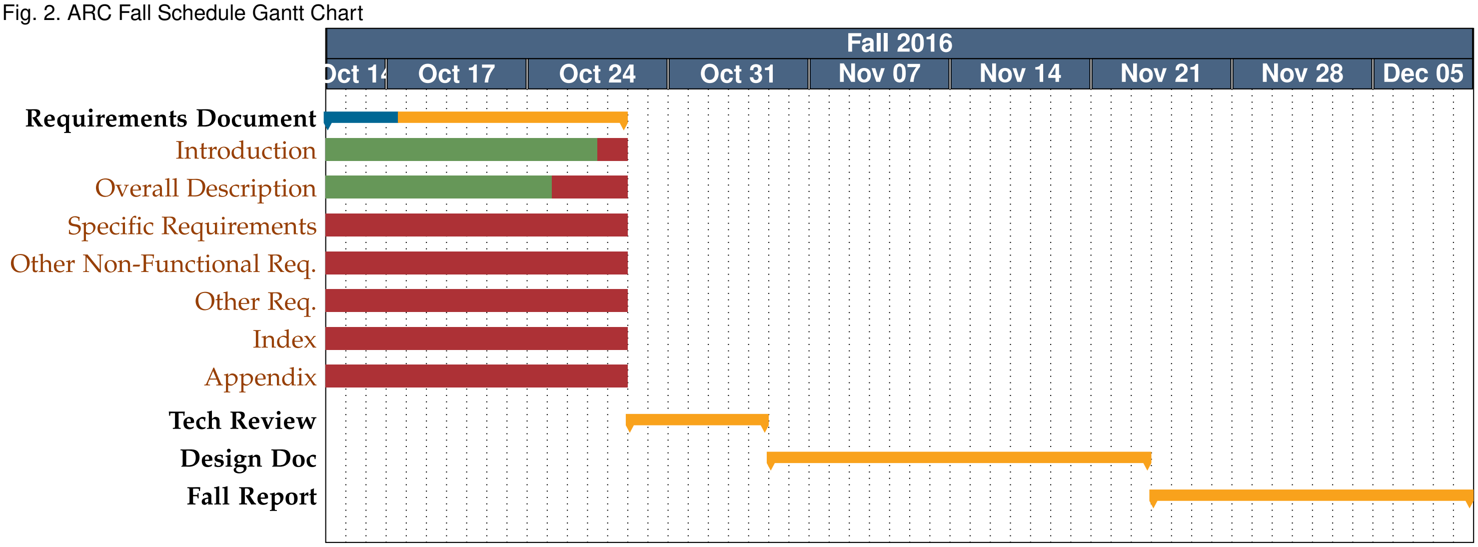


Fig. 3. ARC Winter Schedule Gantt Chart

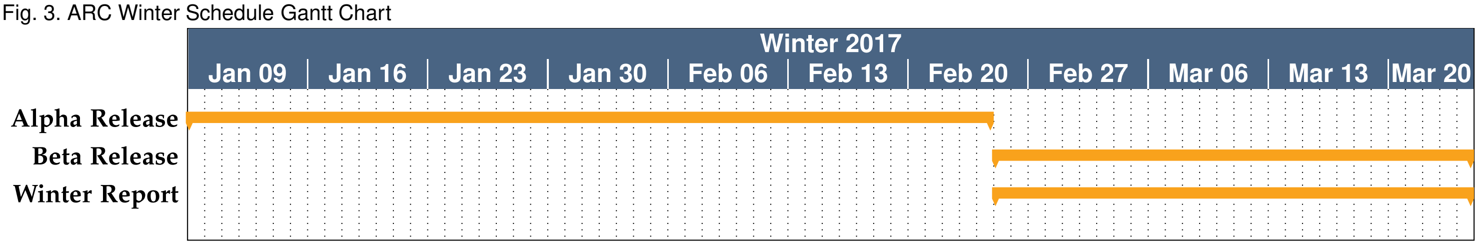
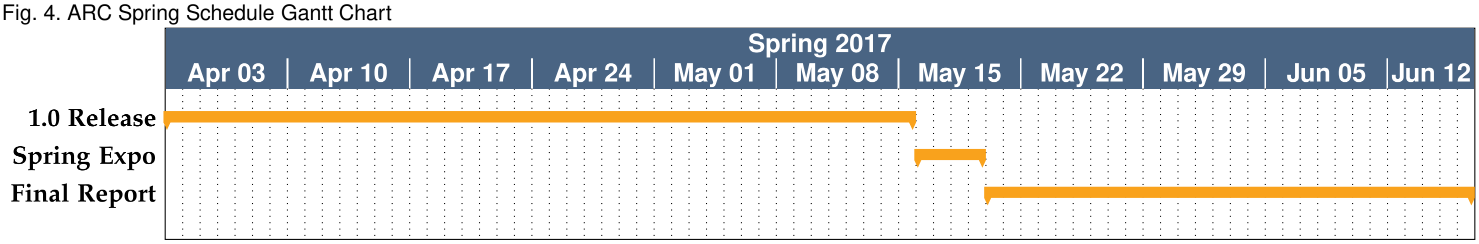


Fig. 4. ARC Spring Schedule Gantt Chart



3 Specific Requirements

This section describes the hardware, software, and performance requirements necessary for the ARC system to function. It lays out requirements for input and output of the system through interfaces. It also covers the major systems features and functional requirements for those features.

3.1 External interface requirements

This section describes all of the interfaces required for input to and output from the ARC system.

3.1.1 User interfaces

There should be a user interface (UI) for viewing a map of the surrounding area and the vehicle progress, as well as setting waypoints. There should be UI where all telemetry from the vehicle can be seen.

3.1.2 Hardware interfaces

There should be hardware interfaces for manual control of the vehicle. This interface should be able to override software control so that users can safely commandeer control of the vehicle from the autopilot.

3.1.3 Software interfaces

There should be software interfaces between

3.1.4 Communications interfaces

3.2 System features

Describes what can be expected for the ARC system to do in terms of its major features.

3.2.1 System Feature: Image Analysis

3.2.2 System Feature: Sensors

3.2.3 System Feature: Navigation

The ARC vehicle needs to navigate a course autonomously. The following functional requirements describe what is required to achieve this.

3.2.3.1 Functional requirement 1:

1) ID: FR-NG1
2) Title: Motor Control

3) Description: To limit the uncertainty caused by the motor under 25% when going forward and backward.
4) Rationale: Maximize the accuracy of motor control to make prediction easier.
5) Dependencies: Probabilistic Analysis for motion

3.2.3.2 Functional requirement 2:

1) ID: FR-NG2
2) Title: Servo Control
3) Description: To limit the uncertainty caused by the servo under 25% when turning.
4) Rationale: Maximize the accuracy of servo control to make prediction easier.
5) Dependencies:

1) ID: FR-NG3
2) Title: Probabilistic Analysis for motion
3) Description: Use probability to estimate the current location of the vehicle with less than 25% error.
4) Rationale: Tires may slip and wobble, which will cause uncertainty on the location of the vehicle.
5) Dependencies:

1) ID: FR-NG4
2) Title: Motion Model
3) Description: Motion model is significant as it decides what commands should be given by the computer under various circumstances.
4) Rationale: This is like learning how to drive a car. Knowing the configuration of the car is critical for the computer to know how to control it. Different combinations of speed and turning angle will result in different paths. Under different surface and weight distributions, the vehicle will also act differently.
5) Dependencies:

1) ID: FR-NG5
2) Title: Global Path Planning Algorithm
3) Description: When operating outdoor, the vehicle needs to go from one location to another following a legal path that is determined by algorithms using sensor data.
4) Rationale: A legal path is a path that goes on concrete surfaces and does not run into any objects.
5) Dependencies:

1) ID: FR-NG6
2) Title: Local Path Planning Algorithm
3) Description: When operating indoor and given the map of the indoor area, the vehicle needs to go from one location to another within the area following a legal path that is determined by algorithms using sensor data.
4) Rationale: A legal path is a path that does not run into any objects.
5) Dependencies:

1) ID: FR-NG7
2) Title: Obstacle Avoidance Algorithm

3) Description: When approaching an object, the vehicle needs to decide whether to turn left or right and by how much. The algorithm can overwrite the path planned by the path planning algorithms.
4) Rationale: When operating both indoor and outdoor, crashing into objects needs to be avoided.
5) Dependencies:

1) ID: FR-NG8
2) Title: Parallel Parking Algorithm
3) Description: Given sensor data and the estimate of the current location, the algorithm outputs the next optimal action (turning the front wheel a certain angle and go forward/backward at a certain speed).
4) Rationale: A dedicated algorithm for parallel parking is necessary because path planning algorithms do not promise the orientation of the vehicle. The parallel parking algorithm will make sure the vehicle is align with the curb/wall when finished.
5) Dependencies:

3.2.4 System Feature: Hardware mounting

3.2.1.3.x Functional requirement 1

1) ID: FR-HW1
2) Title: Minimize Port and Hardware Exposure
3) Description: Seal any unused ports, encase electronic components to minimize environmental exposure.
4) Rationale: Minimizing dust and other elements to electronic components to reduce wear and tear.
5) Dependencies: N/A

3.2.1.3.x Functional requirement 2

1) ID: FR-HW2
2) Title: Secure Hardware Mounting
3) Description: Hardware should be mounted in such a way, that in the event that the vehicle rolls over or high-speed impact, hardware remains in place and moves no more than 1 cm from its original location.
4) Rationale: In the event of an impact or rollover, the hardware should remain in place, and not detach from the vehicle.
5) Dependencies: FR-HW1

3.2.5 System Feature: Communications

3.2.5.1 Introduction/Purpose of feature:

This is a test of the introduction on a new line.

3.2.5.2 Stimulus/Response Sequence:

3.2.1.3.x Functional requirement 1

1) ID: FR-1
2) Title:
3) Description:
4) Rationale:
5) Dependencies:

.

.

.

3.2.1.3.n Functional requirement n

3.3 Performance Requirements

3.4 Design constraints

3.5 Software System Attributes

3.6 Other Requirements

4 INDEX

5 APPENDIX

5.1 Appendix A: Glossary

5.2 Appendix B: Analysis Models

5.3 Appendix C: To Be Determined List