Requirements Spec Section 2

Team TechOps

Table of Contents

1 – Introduction....1
1.1 – Purpose....1
1.2 – Intended Audience and Reading Suggestions....2
1.3 – Product Scope....2
2 – Overall Description....2
2.1 – Product Perspective....2
2.2 – Product Functions....2
2.3 – User Classes....3
2.4 – Operating Environment....3
2.5 – Design and Implementation Constraints....3
2.6 – User Documentation....3
2.7 – Assumptions and Dependencies....4
3 – External Interface Requirements....4
3.1 – User Interfaces....4
3.2 – Hardware Interfaces....4
3.3 – Software Interfaces....5
3.4 – Communications Interfaces....5
4 – System Features....5
4.1 – Capture & Monitor Images....5
4.1.1 – Description and Priority....5
4.1.2 – Stimulus/Response Sequences....5
4.1.3 – Functional Requirements....5
5 – Other Nonfunctional Requirements....6
5.1 – Performance Requirements....6
5.2 – Software Quality Attributes....6

1 – Introduction

1.1 – Purpose

The product that is covered by this SRS relates to a parking monitoring system. This parking monitoring system will keep track of available spaces in a parking lot. This SRS will cover the entire requirements of the parking monitoring system.

1.2 – Intended Audience and Reading Suggestions

The intended audience for this SRS are developers and testers, because this system is more of a research project than a product built for a client. This SRS document covers the purpose of the system, and the functional and nonfunctional requirements of the system.

The functional requirements include the interface requirements and system features. The non-function requirements will include features such as performance and security, as well as any user functions under specific circumstances.

1.3 – Product Scope

The main purpose of the parking monitoring system is to provide a way of monitoring available spaces in the parking lot by using an elevated camera. The system will track available spaces and output the number of free spaces in an external interface. This will help drivers in a vehicle to see which parking lot has available spaces instead of going through a parking lot blindly without knowing if there are any places to park. Another objective of this system is to offer a system with minimal hardware cost by using budget cameras to capture data of the parking lot.

2 – Overall Description

2.1 – Product Perspective

The product is meant to be a standalone system that accomplishes the tasks laid out in the following Product Functions section. Specifically, the product will allow monitoring of parking spaces and report to an end user the overall status of the area.

The product will provide interfaces for setting up capture areas from provided images, and for seeing most-recent or archived parking space conditions.

2.2 – Product Functions

The user will be able to set up defined areas in an image that will be given to the system from a fixed camera. Parking spaces will be able to be delineated from the images, and this information be stored for future use by the product.

After this setup process, the product will check the defined areas in future images given to it and report if there is a vehicle in each marked area.

This information will be displayed to the user, as well as stored for a time in a file that will be accessible to the user.

2.3 – User Classes

There will be two classes of user that will interact directly with the system, and a third that will passively view results produced by it.

The first direct user will handle initial setup of the system and verify that it is functioning correctly, and the second will view information reported by the system and possibly make decisions such as closing the parking lot when full or redirecting drivers.

The user that handles setup will use a graphical interface on a computer to create and save location data for each parking are the system captures. Additionally, this user will be able to view system reports and gathered data, in order to spot check the results gathered.

The second user class will have a similar graphical interface, but functionality will be limited to viewing reports generated by the system, past and present. This user will not be permitted to change or create new monitoring areas.

The final class of user is the operator of vehicles in the parking lot that the system is monitoring. Their only interaction will be viewing a display that shows how many parking spaces are free in the monitored area.

2.4 – Operating Environment

The product will primarily exist on a self-contained computer, which will run Ubuntu 14.04 due to software compatibility needs. Software dependencies include the deep learning library Caffe (release candidate 3), the image processing library OpenCV (v 2.4.13), and Python (v 2.7).

The system will expect a camera to be connected to the computer it is deployed on, and a stream of images to be transferred from the camera to the computer.

2.5 – Design and Implementation Constraints

Limitations in the software product will mainly affect the accuracy of the data the system will gather.

Camera position and lighting conditions may be hard to control in some situations, and can greatly affect the quality of images, which in turn affects the effectiveness of the entire system.

Another limitation is in the actual design and implementation of the neural network that will classify the images gathered from the camera. The design and layout of a network is very open to change, and techniques to increase effectiveness are changing and improving rapidly. This, added to the large amount of overhead in learning to set up such a system, will also affect accuracy of the system. That is to say, there will likely always be optimizations to make, and time limitations will be a deciding factor in when to stop making them.

2.6 – User Documentation

Instructions detailing the initial setup of the system, as well as how to view information gathered, will be provided to the end users. These instructions will preferably be provided online as either web pages or downloadable documents.

2.7 – Assumptions and Dependencies

It is assumed that the product will take full advantage of the deep learning library to be used (Caffe). Due to the complexity of the library and the problem itself however, sufficient time may not exist to properly design and implement the solution. This will lead to reduced accuracy of system reports.

Another limit which is somewhat related to the first is the existence of data to train the network on. Presumably the data set which will be used initially will be sufficient to train the network on, but depending on final camera setup and lighting, the images the network was trained on may not be a good enough simulation of real images that will be gathered. This can be fixed after an initial prototype is deployed however, as training images that much more closely match the actual use case can be gathered easily.

In regards to software, this product has one major dependency that could be subject to change in the future. The operating system it will run on, Ubuntu 14.04, will stop receiving support in approximately nine months at the time of writing. In the (very) long run, this could lead to complications between the OS and hardware, and the product would need to be updated to run on a newer platform.

3 – External Interface Requirements

3.1 – User Interfaces

An end user of the software will have access to one of two main interfaces. Users that monitor the system directly will be presented with a GUI describing the current status of the monitored area as well as the most recent image. For setup, the same general interface will be used, but with additional features relating to the setting and modifying of the currently monitored area. Vehicle operators will be presented with a simple display of available parking spaces, and will not need to directly interact with the system in any way.

So in short, the following interfaces will be present:

- Parking indicators for space availability
- Administrative/Setup interface
- Monitoring Interface

3.2 – Hardware Interfaces

The hardware components of the parking lot application include a central computer system, client computer, and a parking camera. Each component plays an important role in delivering the system.

The central computer system is a server based system which holds data. A computer is used to store the data which contains parking lot information and pictures, which are gathered from a camera overlooking the parking lot.

Interface to access and view numerical values of available parking spaces.

3.3 – Software Interfaces

Software communication will be limited to the selected software libraries working together. This includes Caffe and all of its requirements (primarily scientific computing packages written in python) and OpenCV.

3.4 – Communications Interfaces

The main communication interface that will require specific attention will be between the operating system and the monitoring system regarding loading and processing images. The images that are captured by the camera will be saved in JPEG format and will be sent to the computer. Specifics means of getting images from the camera to the main system is outside the scope of this project. The computer will then process the image and display information regarding parking. That information will then be stored in an XML file (tbd) for later use. Any other detailed communications interfaces would mainly be handled by the underlying operating system, and are outside the scope of this document.

4 – System Features

4.1 – Capture Images

4.1.1 – Description and Priority

This features primary responsibility is to capture an image within a time interval specified by the system administrator. The image is capture using a camera that will be set up on a parking lot. These images are used later for processing through the main computer. The priority of this feature is high, because without an image, there is no way of telling if the parking spaces are available or occupied.

4.1.2 – Stimulus/Response Sequences

Stimulus: A time interval is set by an administrator to capture images
Response: The image is captured by the camera and is sent to the main computer for processing.

4.1.3 – Functional Requirements

1. The camera must observe parking spaces and obtain images
2. The camera must be able to adapt to low-light conditions.
3. TBD – errors regarding low light condition or conditions where camera is being blocked
4. The camera must send captured images to the hard drive.

4.2 – Process Images

4.2.1 – Description and Priority

This features primary responsibility is to gather the images from the hard drive. An image is then broken down into several images of individual parking spaces which will then be inputted to the neural network. The neural network will process these images that contain individual parking spaces, and output whether the parking space is available or occupied. The priority is high for this feature because this feature determines/outputs how many parking spaces are available.

4.2.2 – Stimulus/Response Sequences

Stimulus: A captured image is inputted to the neural network
Response: The neural network will output whether the images are available or occupied.

4.2.3 – Functional Requirements

1. The system must be able to receive images from the location that images were stored in by the camera.
2. The system must be able to recognize the difference between free and occupied parking spaces
3. The system must be able to output whether the space is free or occupied.
4. TBD – if the system cannot receive images from the camera.

5 – Other Nonfunctional Requirements

5.1 – Performance Requirements

- Images will be captured every 15 seconds and sent to the main system – This feature will determine how frequently the system updates its parking status results. This is a performance requirement because the system needs to output the result for users to see how many spaces are available.
- The system must output complete results before the next image arrives – This feature will demonstrate the number of available parking spaces. It is necessary for the feature to finish processing current images before the next image arrives, so that a user can accurately see how many spaces are available in a more accurate manner.
- Saving of information relating to parking images must be completed within the 15 second interval – This feature will let the administrator analyze the occupancy of parking spaces. The feature also needs to be completed within the 15 second interval in order to avoid overlapping of current images and the next set of image that arrives.

5.2 – Software Quality Attributes

Adaptability: Able to adjust to daytime and nighttime settings
Availability: The system shall be available to the administrator 24hours every day of the year