Software Requirement Specification

for

CS 308 Project

DRAWOID

Android Controlled Robot

Group 07

Gaurav Choudhary - TL

Ashish Tajane

Shikhar Paliwal

Rishabh Singhal

Submitted in partial fulfilment of the requirements of

CS 308 Embedded Real-Time Systems Lab

Table of Contents

1. Introduction

The purpose of this document is to present a detailed description of the Drawoid, Android Controlled Robot. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for the developers of the system.

1.1 Definitions, Acronyms and Abbreviations

Term	Definition
Bot / Robot	The electro-mechanical machine that is guided by the user.
Drawoid	The Bot used in this project which can draw on the flat surface the drawing / shape drawn by user on android device.
Android	Android is a Java-based operating system for mobile devices such as smartphones and tablet computers. It is developed by the Open Handset Alliance led by Google.
Software Requirements Specification (SRS)	A document that completely describes all of the functions of a proposed system and the constraints under which it must operate.
User	The person that is currently controlling the robot
Android SDK	Software Development Kit for developing Android applications
Command	A single command or a set / sequence of many commands
IDE	Integrated Development Environment
Bluetooth	Bluetooth is a proprietary open wireless technology standard for exchanging data over short distances from fixed and mobile devices, creating personal area networks (PANs) with high levels of security
API	Application Programming Interface
AVR Studios	IDE for c programming for the drawoid

1.2 References

1. Android Developers Website - http://developer.android.com/index.html
2. Bluetooth Connectivity between Drawoid and Android device - http://developer.android.com/guide/topics/wireless/bluetooth.html

2. Overall Description

2.1 System Requirement

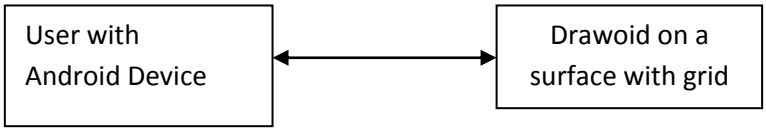

Fig. 1 - System Environment

The drawoid controlled using android touch device has two active systems and no other additional coordinating systems. The user is able to view the robot's environment directly. In the extended version, there can be additional systems like a camera on top like the ceiling or on drawoid itself which gives the location of drawoid through a screen. In that case, the user may not be able to view the drawoid directly. The commands are transmitted from android device itself. The communication between android device and robot is on a dedicated wireless frequency channel using bluetooth. The user is able to view any problem, error report or simply any feedback from robot on the screen of android device.

2.2 Product Perspective

The drawoid is controlled remotely using the Bluetooth wireless interface. The user is giving the input by drawing shapes that are recognized and captured by android device. The shapes captured by android device are translated to a particular command or a sequence of commands using android API and to signals using android platform on java (Android JDK). The drawoid may move using the variable speeds on each motor allowing speed modulation and hence draw curve shaped paths. It can also lift the pen (drawing device) up while moving so it can draw non continuous shapes also.

2.3 Product Functions

Drawoid Movement Case

Description:
The user draws a shape that is recognized by the android device. The command (or a set/sequence of commands) corresponding to the particular shape is forwarded to the drawoid. There can be many possible paths for a robot like different curves. But to simplify, we are first aiming for basic shapes like linear paths and different combinations of linear paths like a square etc.

Initial step by step description:
Before this can be initiated, the user must recognize himself/herself by entering a correct passphrase/password and pair the android device with drawoid and switch the wireless communication on.

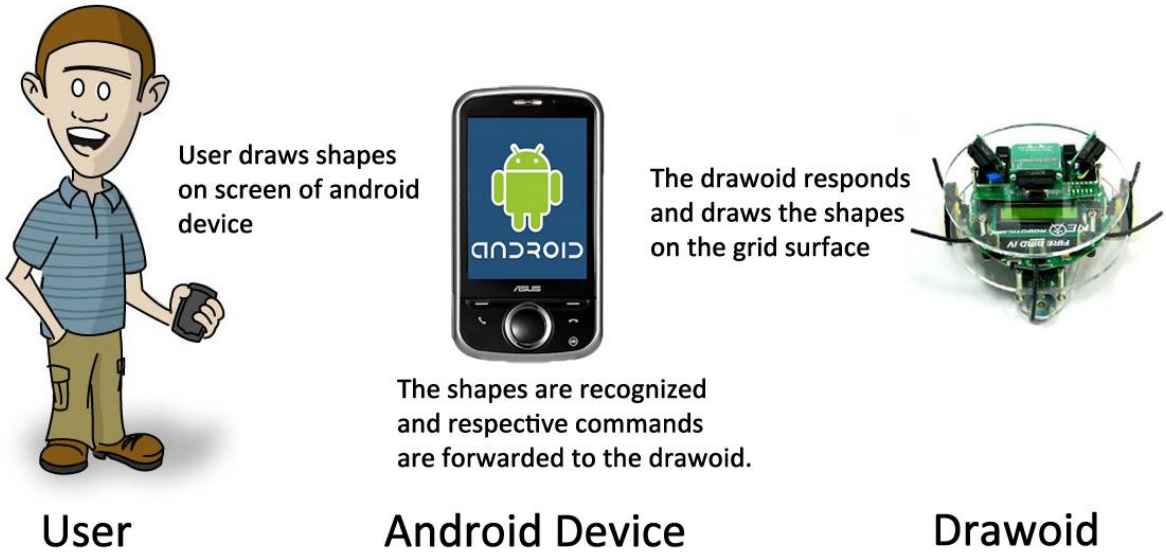

Fig. 2 - Drawing Process

1. The user decides what shape to draw on surface using drawoid
2. He accordingly draws a shape on screen of the android device
3. The shape is recognized and translated into a command by the android device and it is forwarded to the drawoid using bluetooth connectivity.
4. The drawoid accesses the command and signals corresponding movement
5. The drawoid may lift the pen up if it is a free move and not a draw move
6. The drawoid moves as per signal generated
7. The drawoid keeps track of the gridlines passed for reducing error and keeping track of its own position

2.4 User Characteristics

The user is supposed to draw shapes properly with good contact with the screen of android device so that it can be detected properly by the pattern recognition on the android device.

2.5 Constraints

The software developed is restricted to work on touch devices using Android OS although the model can be easily extended with suitable changes to work on touch devices supporting other OS like iOS.

The android device requires a good processor for real time execution of the commands and communication with drawoid.

2.6 Assumptions and Dependencies

The bandwidth of Bluetooth communication should be able to handle the load of communications

- Downlink - commands from android device to the drawoid
- Uplink - response from drawoid to android device

The drawoid is at contactable distance from android device i.e. the drawoid is inside the communication threshold range of android device.

2.7 Requirements subset

- Transmission of commands
- Interpretation of commands
- Signal generation based on commands
- Actions / movements performed based on signals.

3. Details

3.1 Functionality

This is a Android Drawoid System for a user. This system will be designed to maximize the user's productivity by providing an interface to control the Drawoid for drawing. The path through which robot must traverse and draw shapes can be drawn on the Android platform.

More specifically, this system is designed to allow a user to control and communicate with the Drawoid via Android platform wireless connectivity. The wireless network is configured to provide point to point connectivity between the coordinator and the Drawoid ensuring no external interference on the communication channel. The software will facilitate recognition of the patterns made by user which in turn are converted to signals used to control the robot. A set of preformatted responses from the robot can be used to detect any problems faced by the robot to perform the action or provide information on the environment.

3.2 Supportability

3.2.1 Basis for future work

This will form the basis of more sophisticated projects using Android in the coming years.

3.2.2 Distinguished Modules of the Project

There are two distinguished modules of the project:

1. Android programming using Android SDK.
2. Bluetooth communication module for wireless communication.

Any of these modules can potentially be used in other projects requiring their functionality.

3.2.3 Documentation

Documentation in terms of embedded comments and other explicit write-ups will help others in future.

3.3 Design Constraints

The restrictions being implemented in the code are primarily due to those imposed of the system by the hardware being used. They are as follows:

- To develop a application in Android we need Android SDK which runs better on Linux platform.
- While turning we must ensure that current position of pen is used as axis for turning
- Pen up and Pen down will be represented by LED on and LED off because of lack of mechanical lift.

3.4 Interfaces

3.4.1 User Interfaces

This is the part of the system that will be used to interact with the user i.e. command sent from android used for robot navigation. They are as follows:

- A drawing area where path can be specified.
- Start, Pause and Stop button.
- Arrow button to control the robot in manual mode.

3.4.2 Hardware Interfaces

This section states the essential physical equipment required to interact with the application designed. They are as follows:

- Android supported Phone with Bluetooth connectivity.
- Bluetooth communication module.

3.4.3 Software Interfaces

This section states the requirements to development the application which will use the data feed from the various interfaces and use them to control the bot or display information for the user. They are as follows:

- Android SDK
- Ubuntu / Windows / Mac (any operating system) with Eclipse Classic (3.7 - Indigo)
- Atmel AVR Studios

3.4.4 Communications Interfaces

This section states the requirements to create a communication between the drawoid the coordinator. They are as follows:

- Bluetooth Modules

4. Quality Control

4.1 Recognizing Various Paths

Various Paths will be recognized by Android as a series of co-ordinates. Same path may or may not have same co-ordinates.

4.2 Speed of Robot

Speed bar will be given with step size of 20th part of max speed attainable by robot. so There will be 20 different speed attainable by Robot

4.3 Feed-back

It will move on 6x6 grid of black lines on white back-ground. Whenever it crosses a black line its actual co-ordinate will be updated. This will help in minimizing the error.

4.4 Turn

Drawoid will turn using current pen position as axis. Since pen doesn't have any lift/handle which will keep its distance fixed from centre of robot, there will be an implicit error in angle by which it will turn.

5. Risk Management

1. Path/Pattern Recognition: A single touch gesture pattern made by the user will never be an exact match to itself if given as input repeatedly. So, the recognition app will have to be sufficiently lenient so as to allow some liberties to the user but not to misinterpret a path.

Fall Back Plan: co-ordinates will be displayed after path is recognized and user can adjust them every time.

2. Turning by X degree: For turning X degree, we will write a program which will turn itself and adjust pen to its starting position so that the figure is continuous.

Fall Back Plan: If difficult to implement for any angle X, then we will make physical measurements and do it for certain angles like 90 degree, 45 degree.

3. Out Of Range: If at any point of time, it will detect that it is out of range then it will behave according to its default behaviour. Default behaviour will be to halt.

Fall Back Plan: User will option if changing default behaviour to buzzer on or back-trace the path.