Universität Ilmenau

Car - Android - Arduino

Validierung Carduinodroid

Bearbeiter:
Behlau, Benjamin
Haueisen, Sven
Lewandowski, Benjamin
Lewandowski, Felix
Schulze, Christian
Strenger, Robin
Thorwirth, Paul
Vogel, Lars

Betreuer:
Dietrich, Thomas

zuletzt geändert am 03. Juli 2012

Inhaltsverzeichnis

1 Wunsch- und Pflichtkriterienvergleich

1.1 Android-Applikation

1.1.1 Pflichtkriterien

- Akkudaten werden nicht an den PC gesendet, da das Mobiltelefon über die USB-Verbindung durchgehend geladen wird und somit nur der Akkustand des Fahrzeugs von Interesse wäre, welcher sich jedoch nicht ohne weiteres auslesen lässt.
- Auslesen der Verbindungsqualität bedeutet, dass die Art der Verbindung (WLAN/Mobiles Internet) gesendet wird.
- Wiederverbindungsversuche nach Verbindungsabbruch ist noch in Arbeit.

1.1.2 Wunschkriterien

- Automatische Regulierung der Bildauflösung/-qualität anhand der Verbindungsqualität wurde nicht umgesetzt, da es sich als zu aufwendig erwiesen hat, um es in der gegebenen Zeit zu implementieren.

1.2 Computerprogramm

1.2.1 Pflichkriterien

- Auf der Seite des Computers gibt es kein Loglevel, hier werden die gesamten Kommunikationsinformationen ausgegeben, bis auf die Steuerbefehle da das Programm in einer Sekunde 10 Steuerbefehle sendet, was in kürzester Zeit zu großen Datenmengen im Log führen würde.
- Anzeigen der Verbindungsqualität wurde insofern realisiert, dass die Information über die Verbindungsart, die das Mobiltelefon sendet, ausgegeben wird.
- Informationsmeldungen bei beispielsweise Verbindungsabbrüchen oder Bildqualitätsänderungen
- Da die Akkudaten aus oben genannten Gründen nicht ausgelesen werden, müssen sie folglich auch nicht angezeigt werden.

1.2.2 Wunschkriterien

- Anzeige der Geschwindigkeit anhand der GPS-Daten wurde aufgrund von Zeitmangel nicht Implementiert.
- Die Anzeige der verbleibenden Akkulaufzeit ist nicht möglich, da hierfür der Akkustatus des Fahrzeugs ausgelesen werden müsste.

1.2.3 Energiemanagment

- Konnte in einigen Punkten nicht umgesetzt werden, da z.B. bei der Android-Oberfläche das Display nicht ausgeschaltet wird, weil die Kamera des Mobiltelefons nicht im Hintergrund laufen kann, dadurch ist der Stromverbrauch erhöht, da der Bildschirm dauerhaft aktiv ist.

2 Unterschied zwischen Endprodukt und dem Pflichtenheft

2.1 Controller

2.1.1 Controller-Android

- Im Konstruktor wird die Activity übergeben.
- Die Methode „sendData()“ wurde duch „packData()“ ersetzt, die alle relevanten Statusdaten wie zum Beispiel Verbindungsstatus und momentane GPS-Koordinaten in ein Datenpaket schreibt und dieses zurückgibt. Diese Methode wird vom Netzwerkteil aufgerufen und der zurückgebene String an das PC-Programm gesendet.
- „ReceiveControlSignal()“ wurde in „receiveData()“ umbenannt. Die Funktion des Aufschlüsselns des übergebenen Strings und der Ausführung der in ihm enthaltenen Befehle bleibt erhalten.
- „ CameraPicture()“ wurde entfernt. Die Kamerafunktionalität wird von der Klasse „Cam“ übernommen.

2.1.2 Record_Sound

- Im Konstruktor dieser Klasse wird als Parameter der Log übergeben.
- „startRecordSound(path: Path)“ wurde in „start()“ umbenannt. Der Parameter „path“, welcher den Dateinamen der Sounddatei angibt, wurde entfernt, da ein einheitliches Namensschema mit Datum und Uhrzeit der Erstellung verwendet wird.
- „stopRecordSound()“ wurde in „stop()“ umbenannt, jedoch wurde die im Entwurf beschriebene Funktionalität beibehalten.
- Die neue Methode „init()“ übernimmt die Initialisierung des MediaRecorders.

2.1.3 GPS

- Im Konstruktor wird der Log und die Activity übergeben. Des Weiteren wird ein LocationListener definiert und beim Android-System registriert um auf Änderung der GPS-Daten zu reagieren.
- Die Methode „getGPS()“ hat nun lediglich den Zweck die Felder „longitude“, „latitude“ und „altitude“ durch Semikolon getrennt in einem String zurückzugeben.
- Die neue Methode „reset()“ setzt „longitude“, „latitude“ und „altitude“ auf 0. Dies wird dazu benötigt um anzuzeigen dass keine GPS-Koordinaten verfügbar sind.

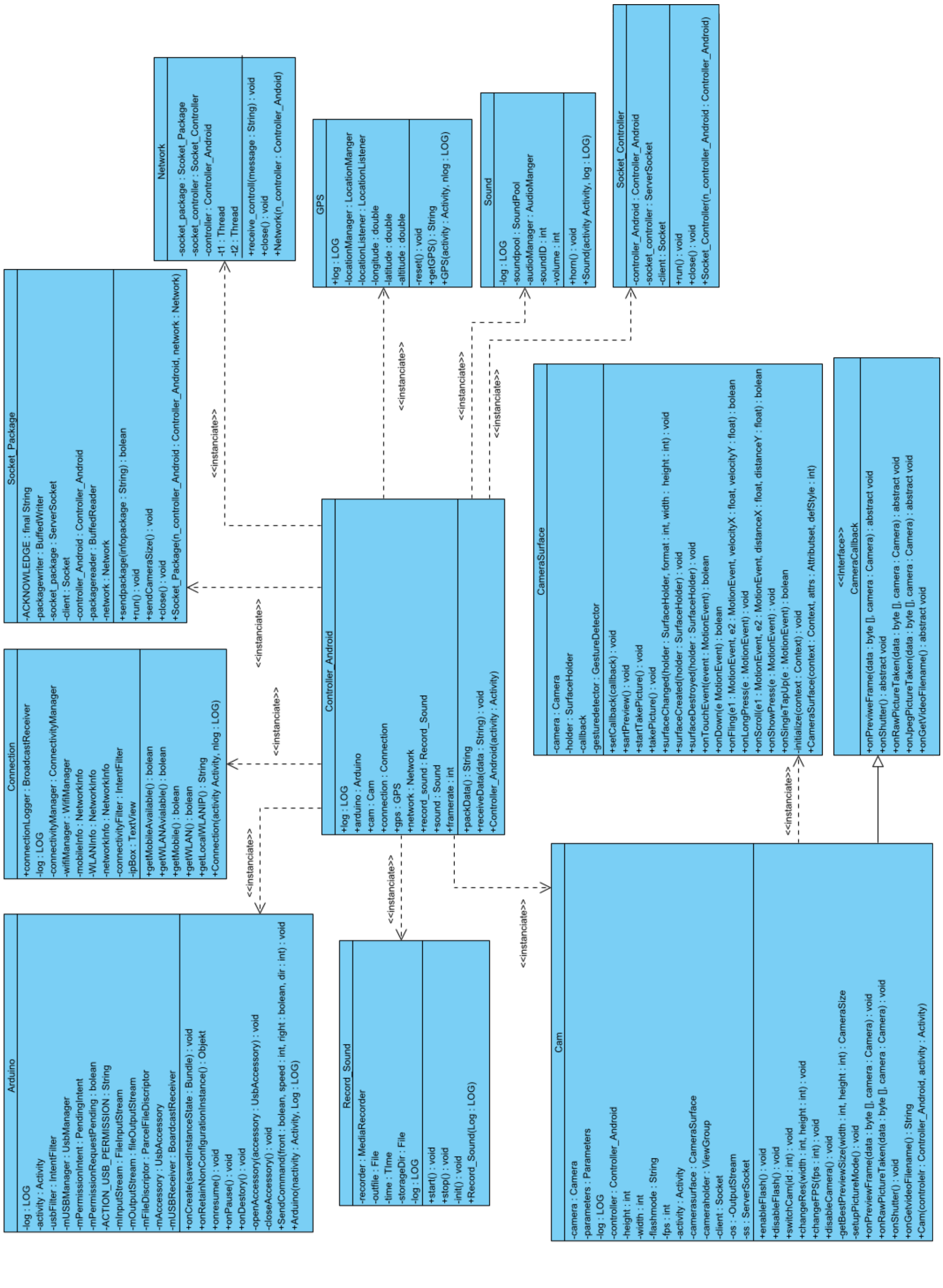

Abbildung 1: Controller_Android

2.1.4 Arduino

- Im Konstruktor wird zusätzlich ein „BroadcastReceiver“ definiert und registriert, der auf das Verbinden des Smartphones mit dem Arduino-Board reagiert.
- Die Methode „SendCommand(Command: string)“ wurde in „SendCommand(boolean front, int speed, boolean right, int dir)“ geändert. „front“ gibt an ob sich das Fahrzeug vorwärts oder rückwärts mit der Geschwindigkeit „speed“ bewegen soll. „right“ gibt an ob nach rechts oder links gelenkt wird. „dir“ gibt den Lenkwinkel an.
- Die neue Methode „openAccessory(UsbAccessory accessory)“ bereitet die Verbindung zum Arduino-Board für das senden von Steuerkommandos vor.
- Die neue Methode „closeAccessory()“ beendet die Verbindung mit dem Arduino-Board sicher.

2.1.5 Connection

- Im Konstruktor wird nun zusätzlich ein „BroadcastReceiver“ definiert und beim Android-System registriert, der auf Änderung des Verbindungsstatuses reagiert und diese in das Log schreibt.
- Die neue Methode „getMobileAvailable()“ gibt die Verfügbarkeit des mobilen Internets zurück.
- Die neue Methode „getWLANAvailable()“ gibt die Verfügbarkeit eines Drahtlosnetzwerks zurück.
- Die neue Methode „getLocalWLANIP()“ gibt die Lokale WLAN-IP des Smartphones zurück, falls verfügbar.
- „getMobile()“ gibt zurück ob das Smartphone mit dem mobilen Internet verbunden ist.
- „getWLAN()“ gibt zurück ob das Smartphone mit einem Drahtlosnetzwerk verbunden ist.

2.2 Model

2.2.1 Sound

- Die Klasse „Sound“ ist jetzt im Controller-Paket zu finden.
- Im Konstruktor wird die Activity und der Log übergeben.
- Zum Ändern der Medienlautstärke wird die Androidklasse „AudioManager“ verwendet.
- Die Methode „play()“ wurde in „horn()“ umbenannt. Vor dem Abspielen der Sounddatei wird die Medienlautstärke auf maximal gesetzt und dannach wieder auf den vorigen Wert.

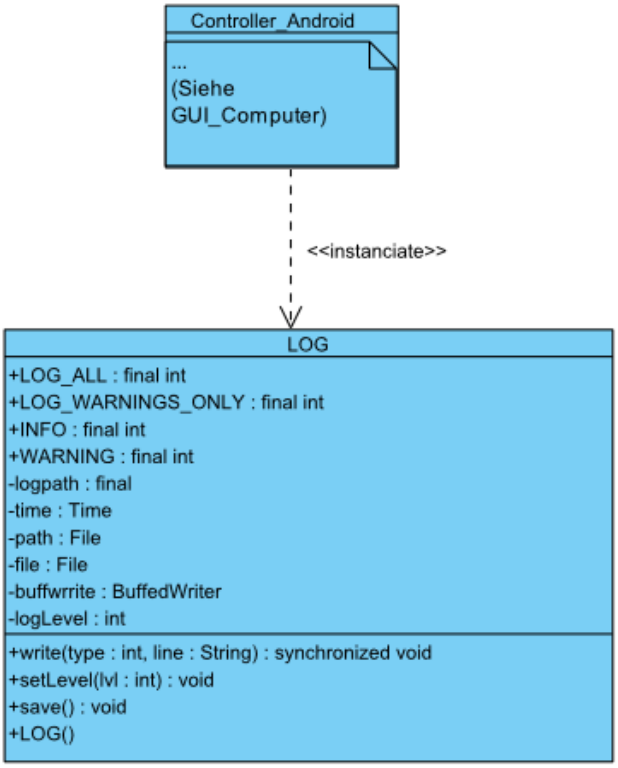

Abbildung 2: Controller_Android

2.2.2 LOG

- Der Andriod-Log hat zwei unterschiedliche Level die durch die Konstanten „LOG_ALL“ und „WARNINGS_ONLY“ definiert werden. Die Einstellung des Log-Levels wird durch die neue Methode „setLevel(int lvl)“ vorgenommen.
- Die Funktionalität von „create_log()“ wird vom Konstruktor der Klasse übernommen.
- Die Methode „write_log(String line)“ wurde in „write(int type, String line)“ geändert. Der Parameter „type“ gibt die Art des Logeintrags an und kann eine der beiden Konstanten „INFO“ und „WARNING“ sein. Des Weiteren hat die Methode das Attribut „synchronized“ was sicherstellt, dass nicht zwei Klassen zur gleichen Zeit in das Log schreiben.
- „save_log()“ wurde in „save()“ umbenannt. Die Funktionalität bleibt erhalten.

2.2.3 SoundRecording

- Diese Klasse wurde entfernt, da die komplette Tonaufnahme in der Controller-Klasse „Record_Sound“ realisiert wird.

2.3 View

Der View unserer Android-Applikation wird duch die Klasse „CarDuinoDroidAppActivity“ repräsentiert. Zum angezeigten Logo kommt ein Umschalter für das Log-Level, ein Textfeld, in dem die IP-Adresse des Smartphones angezeigt wird, und ein Schließen- Button. Außerdem wird der Anwender durch eine Statuszeilenbenachrichtigung davon in Kenntnis gesetzt, dass die Anwendung läuft. Das Minimieren der Applikation wurde durch Schließen ersetzt.

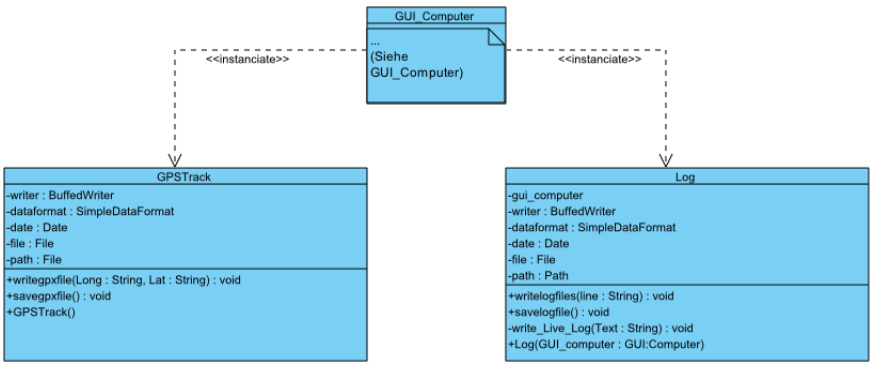

Abbildung 3: GUI_Computer

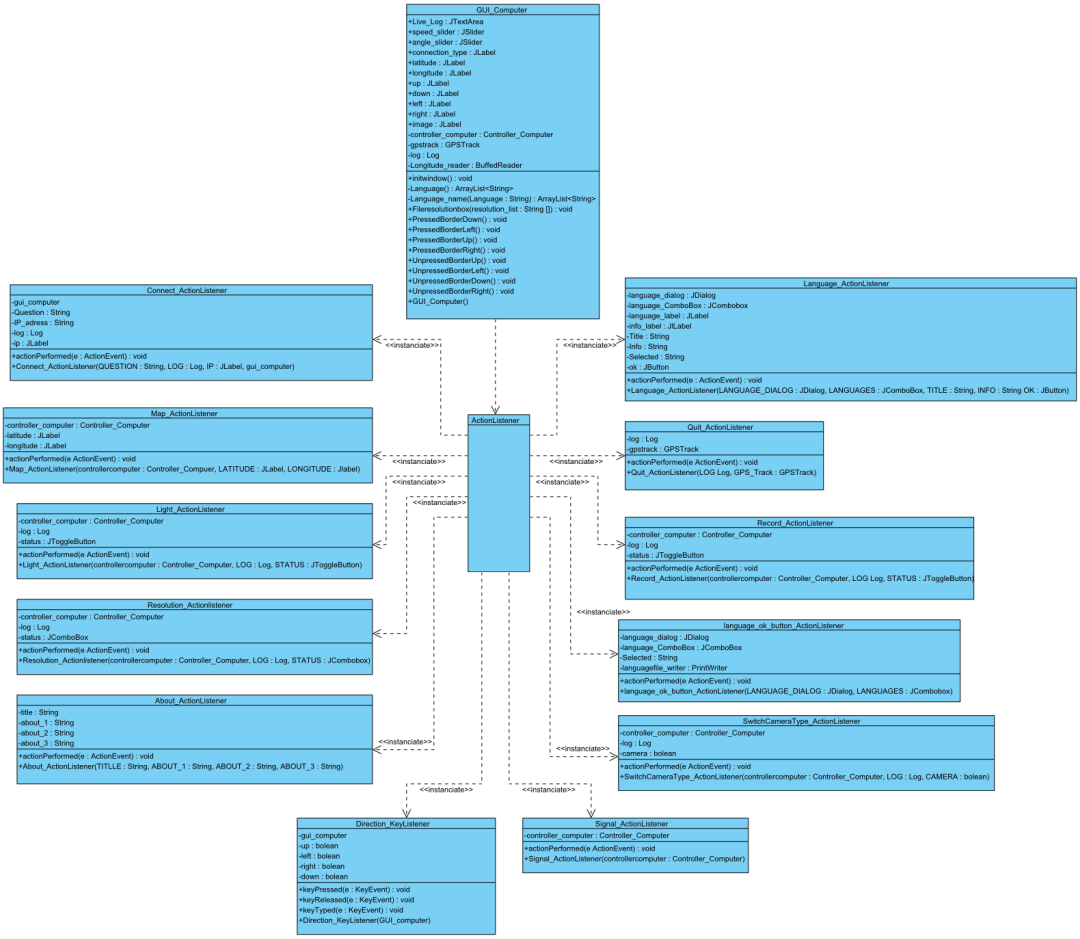

Abbildung 4: Actionlistener

2.4 Vergleich zwischen Entwurf und Implementierung bezüglich der Kamera

Ursprünglich geplant war ein Videostream, der an das Java-Programm geschickt wird und dort angezeigt werden soll. Dies wäre technisch nur mit sehr großen Aufwand möglich, weil wichtige Daten zum Abspielen der Videodaten von Android erst nach der Aufnahme bereitgestellt werden und dementsprechend erst sehr spät an das Java-Programm übertragen werden kann. Dadurch würde es eine sehr hohe Latenz ergeben, wodurch man das Fahrzeug nicht mehr steuern könnte. Dieses Problem konnten wir lösen, indem wir nun einzelne Bilder übertragen. Diese Bilder werden mithilfe der Vorschaubilder der Mobilfonkamera erstellt und an das Java-Programm übertragen. Dadurch entsteht nur eine kleine Verzögerung, aber man erhält eine weniger flüssige Darstellung der Kameradaten als mithilfe eines Videostreams.

2.5 Vergleich zwischen Entwurf und Implementierung bezüglich des Netzwerks

Ursprünglich war geplant, dass wir die gesamte Kommunikation mit dem jeweiligen anderen Teilprogramms in einer einzigen Klasse implementieren wollten. Dies haben wir aufgrund der Übersichtlichkeit umgeändert, indem wir für die einzelnen Kommunikationswege (Sockets) eigene Klassen erstellt haben und eine Netzwerkklasse für die Steuerung dieser Klassen. Für das restliche Programm hat sich dadurch nichts geändert.

Abbildung 5: Network

3 Produktfunktionen

3.1 Androidfunktionen

- /A01/ Der Verbindungsaufbau zwischen Mobiltelefon und PC ist über eine Eingabe der IP-Adresse des Mobiltelefons in das dafür vorgesehene Feld der Java-GUI realisiert worden. Folglich wird die Verbind nicht vom Mobiltelefon aus hergestellt, sondern vom PC.
- /A052/ Eine Einzelbildübertragung ist nicht möglich, da das Kamerabild in Form des „Vorschaubildes“ des Mobiltelefons an den Computer übertragen und dort angezeigt wird.
- /A055/ Die Tonaufnahmen werden nicht Live zum PC übertragen, sondern auf dem Mobiltelefon abgespeichert.
- /A0551/ Qualität: Die Soundqualität kann nicht angepasst werden, da nur eine Aufnahme stattfindet, welche nicht an den PC gesendet wird.

3.2 Javafunktionen

- /J01/ Wie oben beschrieben wird die Verbindung durch die Eingabe der IP-Adresse des Mobiltelefons hergestellt.
- /J06/ Informationsmeldungen bei auftretenden Fehlern oder sonstigen Ereignissen wurden in Form von Logeinträgen, welche auch im Live-Log ausgegeben werden, realisiert.

3.3 Oberfläche

- /O013/ Die Einstellungen bezüglich der Bildqualität sind direkt auf der GUI ausführbar.
- /O03/ Warnmeldung bei Bildausfall bzw. Verbindungsabbruch ist noch in Arbeit.
- /O05/ Die Richtungsänderung wird durch betätigen der Pfeiltasten umgesetzt. Jedoch muss der Winkel, der bei der Betätigung eingeschlagen wird, vorher über einen separaten Regler eingestellt werden.

4 Namenskonventionen

- Im Allgemeinen werden zwischen Konstanten, Variablen und Klassennamen unterschieden. Konstanten erkennt man daran, dass lediglich Großbuchstaben verwendet werden. Konstantennamen, welche aus mehreren Worten besteht, trennt man durch einen Unterstrich („_“). Im Gegensatz dazu, beginnen Variablen mit Kleinbuchstaben. Es können jedoch Großbuchstaben folgen. Klassennamen werden groß geschrieben. Sofern Namen aus mehreren Wörtern bestehen, werden diese zusammen geschrieben und der zweite Teil beginnt mit einen Großbuchstaben.

5 Dateiorganisation

5.1 Android

- Während der Programmierung haben wir uns an das Prinzip der Fassaden gehalten. Daran orientiert wurden die Pakete aufgebaut. Die Logdateien und die Tonaufnahmen werden auf einer externe SD-Karte des Mobiltelefons wie folgt gespeichert:
  - /sdcard/carduinodroid/log
  - /sdcard/carduinodroid/Recording
- Weiterhin sind Ressourcen wie Icons, Logo und Tondateien in der .apk-Datei enthalten, auf welche durch die Felder der automatisch generierte Klasse „R“ zugegriffen werden kann.

5.2 Java

- Auf der Javaseite wird ein neuer Ordner für die Log-Datei und für die gpx-Datei angelegt. Diese befinden sich in dem src Ordner des Eclipseprojekts. Die Speicherung findet in folgenden Ordnern statt:
  - src/Logs/
  - src/gpx/
- Weitere Daten, wie die Spracheinstellungen und Icons, werden aus dem jeweiligen Paket des Projekts geladen.

6 Dokumentation und Entwicklungsumgebung

6.1 Art der Dokumentation

- Die Dokumentation des Quelltextes wurde durch Kommentare und Javadocbefehle realisiert, welche auf Englisch verfasst wurden. Somit kann das Programm international genutzt werden.
- Einzelne Methoden, Funktionen und Parameter im Quelltext, werden mit einfachen Kommentaren versehen, welches durch folgende Formatierung geschieht „//“.
- Des Weiteren wurde im Javadoc aktuellere Informationen hinzugefügt und Querverweise auf andere Klassen bzw. Methoden hergestellt. Einzelne Methoden wurden erklärt und der Klasse ein Autor und eine Versionsnummer zugewiesen. Die Formatierung ist wie folgt:
  „/**
  *
  /*“
  Für jede neue Zeile wird ein neuer „*“eingefügt. Verlinkungen von Parametern oder Autoren werden immer mit einem „@“ Zeichen gekennzeichnet. Außerdem sind Querverweise mit „@see“ zu zu makieren.

6.2 Entwicklungsumgebung und besondere Konfiguration

- Als Entwicklungsumgebung für unser Projekt wurde Eclipse, Version 4.2 gewählt. Über sogenannte Plug-ins wurde der Funktionsumfang für die Android- Entwicklung mit dem ADK erweitert. Für das wissenschaftliche Verfassen von Dokumentationen wählten wir Texclipse im Zusammenhang mit Miktex.

7 Testprotokoll

Test	Durchführung	Ergebnisse
/TA01/ /TJ01/ /TO012/	Das Java-Programm wird gestartet. Die Testperson gibt über das Menü die IP-Adresse des Mobiltelefons ein, um eine Verbindung herzustellen.	Das Java-Programm startet nach Eingabe der IP-Adresse und Druck auf den OK-Button. Das Programm verbindet sich mit dem Mobiltelefon und das Kamerabild wird sofort an den PC übertragen.
/TA03/	Es wird im Javaprogramm überprüft, ob die Anzeige für die Verbindungsqualität funktioniert. Dazu wird das Carduinodroid durch mehrere Gebiete verschiedener Netzarten navigiert.	Die Anzeige der Verbindungsqualität beschränkt sich auf die Art der Verbindung (keine Signalstärke etc.).
/TA041/ /TO031/	Der Computer wird vom Internet getrennt. Das Carduinodroid hat mit einem „Systemstopp“ zu reagieren.	Sobald das Mobiltelefon keine Befehle mehr erhält, stoppt das Fahrzeug.
/TA042/	Der Verlust der Kamerafunktion wird dadurch simuliert, dass die Android-Applikation minimiert und dann einen andere Anwendung gestartet wird, die ebenfalls die Kamera benutzt. Das Carduinodroid hat mit einem „Systemstopp“ zu reagieren. Während dieses Tests wird das Auto in der Luft gehalten.	Da eine Minimierungsfunktion nicht implementiert werden konnte, wird dieser Test nur mit Beenden der Software durchgeführt. Sobald das Programm beendet wird, stoppt das Fahrzeug automatisch.
/TA05/	In der Java-Software werden auf verschiedene Funktionen der Android-Applikation zugegriffen und deren Reaktion überprüft.	Alle Funktionen verhalten sich wie gewünscht.
/TA051/ /TO02/	Am PC werden in der Informationsbox die GPS-Daten ausgelesen und auf Aktualisierung überprüft. Weiterhin wird der Button zur Anzeige der GPS-Koordinaten auf einer Karte betätigt. Danach wird die Kartenanzeige überprüft.	Die GPS-Daten werden ständig aktualisiert. Betätigt man den Karte-Button wird der aktuelle Standort auf Online-Kartenmaterial im Browser angezeigt.

Test	Durchführung	Ergebnisse
/TA052/ /TJ05/	Durch Betätigen des dafür vorgesehenen Buttons wird zwischen Front- und Rückkamera umgeschaltet.	Der Wechsel zwischen den Kameras funktioniert mit einer geringen Verzögerung.
/TA0521/ /TO013/ /TO03/	Im Computerprogramm wird die Bildqualität geändert. Es wird kontrolliert, ob sich jene entsprechend den Vorgaben anpasst.	Es gibt eine Möglichkeit die Auflösung zu ändern und die Bildqualität über einen Schieberegler einzustellen.
/TA0522/ /TO03/	Im Computerprogramm wird zwischen den beiden Kameras gewechselt. Die Änderung des wiedergegebenen Bildes wird überprüft.	Die Kamera wird wie vorgesehen gewechselt, jedoch kann das Bild der Frontkamera je nach Typ des Mobiltelefons um 180° gedreht errscheinen.
/TA053/ /TO07/	Durch Betätigen des „Signalton“-Buttons wird ein Ton vom Mobiltelefon über dessen Lautsprecher abgespielt.	Betätigen des Signalton-Buttons: Ein Signalton wird am Mobiltelefon ausgegeben und im Live-Log wird die erfolgreiche Übertragung des Befehls angezeigt.
/TA054/ /TO03/	Nachdem das Carduinodroid in einem Bereich mit geringer Lichtstärke platziert wurde, wird die Beleuchtung der Kamera aktiviert.	Betätigen des Licht-Buttons: Die Lampe des Mobiltelefons schaltet sich ein und im Live-Log wird die erfolgreiche Übertagung angezeigt. Durch erneutes Betätigen wird das Licht deaktiviert.
/TA055/	Mittels der Computersoftware wird das Mikrofon aktiviert. Es wird überprüft, ob akustische Signale von Android zum PC übertragen werden.	Betätigen des Tonaufnahme-Buttons: Das Mobiltelefon nimmt seine Umgebung auf und speichert eine Sounddatei ab. Im Live-Log wird die erfolgreiche Übertragung des Befehls angezeigt. Durch erneutes Betätigen wird die Aufnahme beendet.
/TA0551/	Am Computer wird die Reaktion bei Änderung der Soundqualität getestet.	Die Änderung der Soundqualität wurde nicht Implementiert.
/TJ02/ /TJ03/ /TO06/	Es werden im Computerprogramm das Kamerabild, der Umgebungssound sowie die Informationsbox überprüft. Dabei wird die Drohne in unterschiedliche Gebiete gefahren und auf Vorhandensein der Daten sowie deren korrekte Aktualisierung geachtet.	Alle Daten sind vorhanden und korrekt.

Test	Durchführung	Ergebnisse
/TJ06/ /TO03/	Die Informations-/ Fehlermeldungen können beispielweise mit dem Trennen der Internetverbindung, Fahren in Umgebungen mit sehr schwachem Netz (mobiles Internet) oder dem Öffnen der Android-Kamera- Applikation im Vordergrund überprüft werden. Nach dem Trennen der Internetverbindung, wird kontrolliert, ob die Drohne zum Stehen kommt.	Die Fehlermeldungen werden wie gewünscht im Live-Log angezeigt. Wird die Internetverbindung getrennt, bleibt das Fahrzeug stehen. Minimiert man die Applikation wird das Fahrzeug automatisch gestoppt, da die Applikation dadurch beendet wird.
/TO04/ /TA02/ /TJ04/	Mittels der Pfeil-Hoch-Taste wird das Carduinodroid beschleunigt. Dies wird jeweils mit verschiedenen Geschwindigkeitseinstellungen wiederholt. Mittels der Pfeil-Runter-Taste wird das Carduinodroid abgebremst. Nach Loslassen aller Tasten hat das Carduinodroid wieder stillzustehen.	Durch das Betätigen der Pfeil-Hoch-Taste fährt das Fahrzeug los, während der Fahrt kann die Geschwindigkeit nach Wunsch reguliert werden. Betätigt man die Pfeil-Runter-Taste fährt das Fahrzeug rückwärts. Wird keine Taste gedrückt stoppt das Fahrzeug.
/TO05/ /TA02/ /TJ04/	Mittels der Pfeil-Rechts-Taste und der Pfeil-Links- Taste wird das Carduinodroid, sofern wie in /TO04/ beschreiben beschleunigt, nach rechts oder links gelenkt. Nach Loslassen Pfeil-Rechts-Taste oder der Pfeil-Links-Taste hat das Carduinodroid wieder geradeaus zu fahren.	Nach dem Tastendruck nehmen die Vorderräder den zuvor eingestellten Lenkwinkel ein. Wird keine Taste mehr gedrückt stellen sich die Räder wieder in Ausgangsstellung.
/TO01/	Das Menü wird geöffnet.	Durch das Drücken von „Datei“ oder „Einstellungen“ öffnet sich das entsprechende Menü.
/TO03/	Es wird während einer gesamten Testfahrt überprüft, ob permanent und mit welcher Verzögerung ein Bild der Kamera zu sehen ist.	Die Verzögerung des Kamerabildes ist bei guter Verbindung kaum wahrzunehmen. Bei schlechter Verbindung kann es allerdings zu Standbildern kommen.
/TO011/	Im Anschluss an alle aufgelisteten Testszenarien wird das Java- Programm über das Menü beendet.	Das Java-Programm schließt sich.

8 Statistiken

8.1 statistische Daten

- LOC:
  - Android: 1178
  - Java Programm gesamt: 1339
- Gesamt: 2517

8.2 Dateigrößen

- Android-Applikation: 130 KB
- Java- Programm: 68 KB + 35 KB Icons und Sprachdatein

8.3 SVN-Statistik

- Commits: 860
- Dateiänderungen: 1932
- Durschnittliche Commits pro Monat: 215
- Durschnittlich geänderte Dateien pro Monat: 483

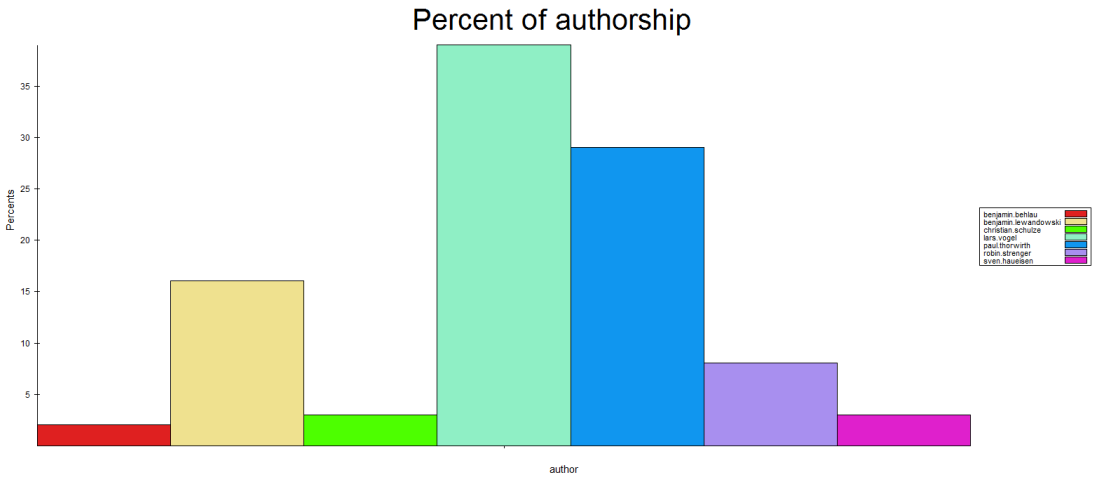

Abbildung 6: Percent of autoship

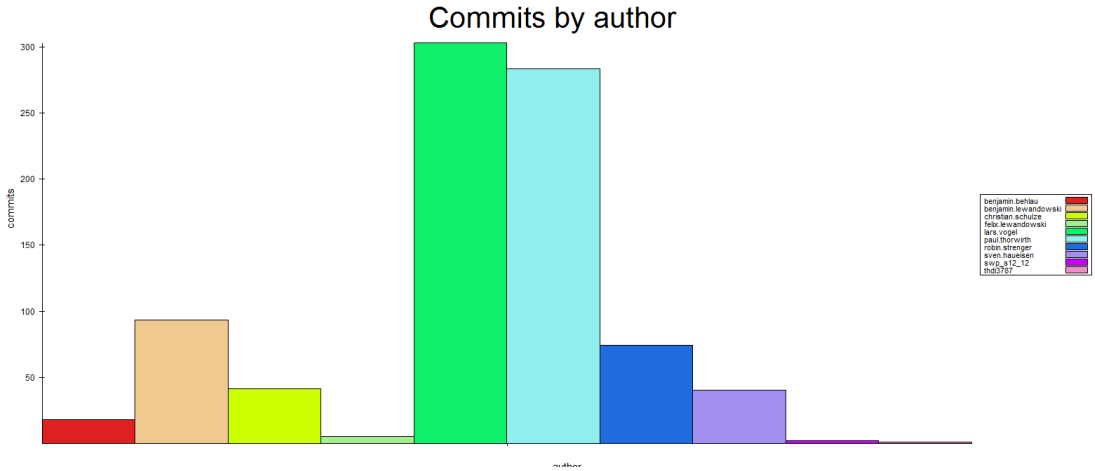

Abbildung 7: Commits by author

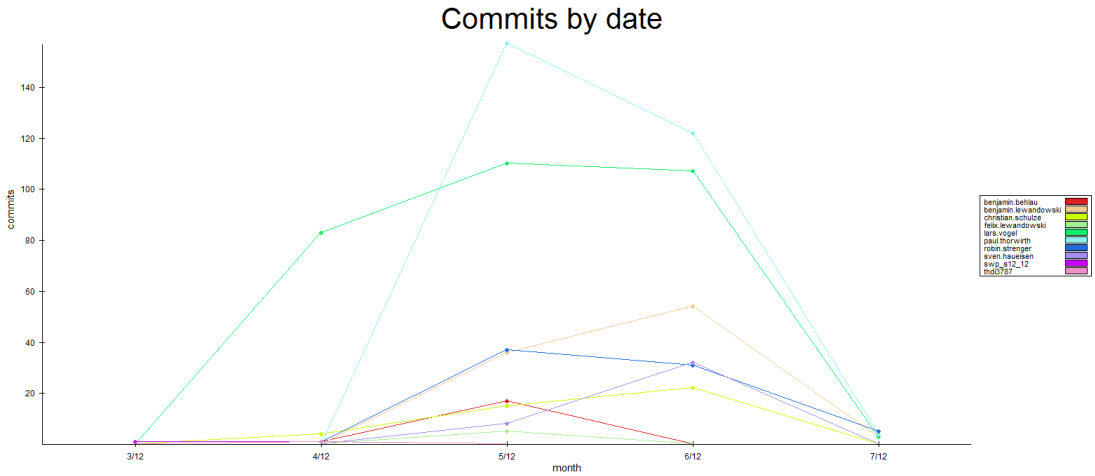

Abbildung 8: Commits by date

9 Ausblick

Die Entwicklung einer Drohne aus Konsumelektronik stellte in Anbetracht der bisher erlangten Kenntnisse aus der Studienzeit eine deutliche Herausforderung dar. Besonders im Bereich der Netzwerktechnik bleiben noch Möglichkeiten einer konsequenten, nicht netzwerkbeschränkten Nutzung offen. Ein Server, der alle laufenden Android- Applikationen verwaltet und Zugriff per Kennziffer sowie Passwort herstellt, wäre eine denkbare Alternative. Bisher ist man gezwungen unterschiedliche private Netzwerke beispielsweise über VPN zu verbinden. In Zeiten der Miniaturisierung benötigt das Energiemanagement zudem einen besonderen Betrachtungspunkt. Ausgiebige Spähmissionen, bedingt durch die Akkulaufzeiten am Mobiltelefon und Auto selbst, sind bisher noch Zukunftsvisionen. Hierbei empfiehlt es sich in Sachen der Hardware leistungsfähigere Bauteile zu integrieren. Vor allem die Lademöglichkeit per Arduino-Controller verbunden mit der neuen Hardware würde dieses Problem negieren. Die Portiermöglichkeit auf andere Plattformen (Betriebssystem Mobiltelefon) könnte in naher Zukunft durch Interesse verschiedener Nutzergruppen ein denkbares Ziel sein. Eine globale, systemübergreifende Nutzung und Vernetzung ermöglichen einerseits die abwechselnde Inanspruchnahme verschiedener Parteien und andererseits die Einsparung von Hardwaregrundlagen hinsichtlich ferngesteuerter Fahrzeuge. Durch die Erweiterbarkeit kann der programmierinteressierte Nutzer ohne weiteres die Geschwindigkeits- sowie Lenkparameter an seine gewählten Motorleistungen anpassen. Zudem bestände die Möglichkeit Einstellungen zu integrieren, die diese direkt auf der Oberfläche gewährleisten. In Sachen Erweiterung ist der kommenden Entwicklung somit keine Grenze gesetzt. Dennoch sollte für künftige Ideen immer der Aspekt der Datenströme im Hinterkopf behalten werden. Priorisierung wird dahingehend notwendig, wenn konkurrierende Datenflüsse zeitgleich die Bandbreite belasten. Als letzter Punkt für Veränderungen in naher Zukunft wird die Übertragung und Darstellung der Bilddaten herausgegriffen. Für weitere Umsetzungen bedarf es einer Bearbeitung mit weitreichenden Codec-Kenntnissen, da Verzögerungszeit mit Datenqualität im guten Kompromiss stehen muss. Summa summarum wurde eine Grundlage für weitere Entwicklungen gelegt, die einen großen Funktionsumfang aufweist.

10 Glossar

Begriff	Erklärung
Android	ist ein Betriebssystem für Smartphones.
Applikation	ist ein Computerprogramm, das eine für den Anwender nützliche Funktion ausführt.
Arduino	ist eine aus Hard- und Software bestehende Mikrocontroller-Umgebung.
Arduino-Board	ist eine aus Soft- und Hardware bestehende Plattform.
Carduinodroid	ist die Zusammensetzung der Wörter Car, Arduino und Android zu einem Wort.
Eclipse	ist ein quelloffenes Programmierwerkzeug zur Entwicklung von Software.
GPS	ist ein globales Navigationssatellitensystem zur Positionsbestimmung und Zeitmessung.
GUI	oder „Graphical User Interface“ bezeichnet die grafische Benutzeroberfläche des Systems.
Icon	lässt sich als Symbol oder Sinnbild übersetzen.
Implementierung	bedeutet die Umsetzung von festgelegten Strukturen in einem System.
IP-Adresse	ist eine Adresse in Computernetzen, die auf dem Internetprotokoll basiert.
Java	ist eine objektorientierte Programmiersprache.

Begriff	Erklärung
Konstruktor	werden in der Programmierung spezielle Prozeduren bzw. Methoden bezeichnet, die Objekte und Variable aufrufen.
Latenz	ist der Zeitraum zwischen 2 Ereignissen.
Log	ist die Zusammenfassung und Aufzeichnung bestimmter Ereignisse.
Loglevel	bezeichnet die Möglichkeit zwischen verschiedenen Log konfigurationen zu wechseln.
Logo	stellt ein Produkt oder Firma in Form einer Grafik dar.
Mobiles Internet	bezeichnet die verschiedenen Übertragungsarten, wie zum Beispiel: EDGE, HSDPA et cetera.
Namenskonvention	ist die Vereinbarung von Programmierern für ein Projekt.
Socket	dient zur Verbindung eines Computerprogramms mit einem Netzwerk.
String	ist ein Datentyp in der Informatik, der eine Zeichenkette als Wert annimmt.
Systemstopp	ist das Anhalten der Motoren mit der Anzeige einer Warnung.
View	bezeichnet die Ansicht auf ein bestimmtes Objekt.
WLAN	ist die Abkürzung für ein drahtloses lokales Netzwerk.