Technische Universität Braunschweig

ISF

Capture the flag

Team 2

Software-Entwicklungspraktikum (SEP)
Sommersemester 2015

Abnahmetestspezifikation

Auftraggeber
Technische Universität Braunschweig
Institut für Programmierung und Reaktive Systeme
Prof. Dr. Ursula Goltz
Mühlpfordstraße 23
38106 Braunschweig

Betreuer: Benjamin Mensing

Auftragnehmer:

Name	E-Mail-Adresse
Patrik Brendel	p.brendel@tu-bs.de
Anika Christmann	a.christmann@tu-bs.de
Carsten Hoppe	c.hoppe@tu-bs.de
Björn Kottutz	b.kottutz@tu-bs.de
Sebastian Lübbe	s.luebbe@tu-bs.de
Marina Porrmann	marina.porrmann@tu-bs.de
Vanessa Wolters	vanessa.wolters@tu-bs.de

Braunschweig, 13. Mai 2015

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Das vorliegende Dokument befasst sich mit der Planung und Dokumentation der Softwaretests. Um den im Pflichtenheft angeführten Qualitätskriterien und Funktionen gerecht zu werden, soll die zu entwickelnde Software während des gesamten Entwicklungsprozesses kontinuierlich getestet werden. Gleichzeitig wird das Dokument erweitert, damit am Ende des Projekts vollständige Abnahmetestspezifikationen als Grundlage für die Abnahmetests vorliegen.

Das im Rahmen des SEP 2015 zu entwickelnde Spiel *Capture the Flag* setzt sich aus mehreren zu testenden Komponenten zusammen:
Grundlegend läuft das Spiel auf einem Server, der die Spielmechanik ausführt und eine Benutzerschnittstelle bereitstellt. Die Serversoftware setzt sich zusammen aus einer GUI und einer KI zur Berechnung der Spielzüge. Zusätzlich kommuniziert die Serversoftware mit den Lego Mindstorms EV3-Robotern und der Kommunikationsschnittstelle des gegnerischen Teams. Für die Funktionsfähigkeit des Spiels ist eine korrekte Kommunikation zwischen allen Komponenten sowie den beiden Teams von äußerster Wichtigkeit.
Der physische Teil des Spiels läuft auf einer auf dem Boden abgeklebten Karte mit EV3 Robotern als Spielfiguren ab. Zur korrekten Visualisierung des Spiels müssen die Roboter alle Befehle vollständig annehmen und ausführen können.

Weitere nichtfunktionale Anforderungen ergeben sich aus der Tatsache, dass die Software ein Spiel ist. Um einen leichten Einstieg zu gewährleisten, muss die Benutzeroberfläche und die Spielmechanik leicht verständlich und zu bedienen sein. Auch die Installation und die einfache Anpassung von Spielparametern müssen gegeben sein, sodass diese Anforderungen objektiv getestet werden müssen.

2 Testplan

Im folgenden Abschnitt wird der Testplan für *Capture the Flag* dargelegt. Es wird auf die einzelnen Komponenten und Funktionen eingegangen, aus denen *Capture the Flag* besteht. Außerdem wird die Vorgehensweise der einzelnen Tests beschrieben.

2.1 Zu testende Komponenten

Im folgenden Abschnitt werden die vier zu testenden Komponenten beschrieben. Diese sind der Server, die GUI, die KI und der Lego Mindstorms EV3-Roboter.

- **Der Roboter**: Der Roboter muss eine funktionierende Wlan-Schnittstelle besitzen, um mit der KI kommunizieren zu können. Außerdem müssen die Akkus der Roboter vollständig geladen sein. Des Weiteren muss der Roboter zum Testen eingeschaltet sein. Damit eine Teilnahme der Roboter am Spiel möglich ist, muss das Programm per W-Lan oder USB-Stick auf den Roboter geladen werden.
- **Der Server**: Der Computer bzw. das Notebook, auf dem das Spiel ausgeführt werden soll, muss über passende Anschlüsse für ein Massenspeichergerät verfügen, da die Server-Software auf einem solchen Speicher vorliegt. Außerdem muss der Computer eine W-Lan-Schnittstelle besitzen, um mit dem Server des anderen Teams und den Robotern kommunizieren zu können.
  - **Die KI**: Auch die KI liegt auf einem Massenspeicher vor und wird von dort gestartet. Die KI ist ebenfalls wie der Server und die GUI angebunden.
  - **Die GUI**: Die GUI-Software liegt ebenfalls auf einem Massenspeicher vor und kann von dort ausgeführt werden. Zum Testen dieser Software wird das gleiche TCP/IP Netzwerk wie beim Server benötigt. Über die GUI kann der Roboter manuell gesteuert werden.

Die jeweiligen Programme liegen alle als kompilierbarer Java-Quellcode vor. Zum Testen dieser Komponenten werden die im nachfolgenden Abschnitt beschriebenen Funktionen und Merkmale benötigt.

2.2 Zu testende Funktionen/Merkmale

Im vierten Kapitel des Pflichtenhefts, werden die Produktfunktionen näher beschrieben. In diesem Abschnitt wird das Testen dieser Funktionen näher erläutert.

- $\langle F10 \rangle$ Fahren: $\langle RM1 \rangle$
- $\langle F20 \rangle$ Angriff: $\langle RM7 \rangle$
- $\langle F30 \rangle$ Gittererkennung: $\langle RM2 \rangle$
- $\langle F40 \rangle$ Blinken: $\langle RS4 \rangle$
- $\langle F50 \rangle$ Flaggenaufnahme: $\langle RM7 \rangle$
- $\langle F60 \rangle$ WLAN-Datenverbindung: $\langle RM3 \rangle$, $\langle RM8 \rangle$
- $\langle F70 \rangle$ Spielzug berechnen: $\langle RM11 \rangle$
- $\langle F80 \rangle$ Spielstart: $\langle RM7 \rangle$
- $\langle F90 \rangle$ Spielabbruch: $\langle RM7 \rangle$
- $\langle F100 \rangle$ Spielende: $\langle RM7 \rangle$
- $\langle F110 \rangle$ Spieldarstellung: $\langle RM6 \rangle$, $\langle RM13 \rangle$, $\langle RM14 \rangle$
- $\langle F120 \rangle$ Karte erstellen: $\langle RM12 \rangle$
- $\langle F130 \rangle$ Karte wählen: $\langle RM6 \rangle$
- $\langle F140 \rangle$ Spielsteuerung wählen: $\langle RM11 \rangle$, $\langle RM14 \rangle$
- $\langle F150 \rangle$ Manuelle Steuerung: $\langle RM14 \rangle$

Zusätzlich müssen folgende Musskriterien, die nicht durch die Produktfunktionen abgedeckt werden, getestet werden.

- $\langle RM4 \rangle$ Rechteckiges Spielfeld
- $\langle RM9 \rangle$ Server verwaltet Roboteraufenthaltsort
- $\langle RM10 \rangle$ Server verwaltet Status der Spielteilnehmer

2.3 Nicht zu testende Funktionen

Das Kriterium $\langle RM5 \rangle$ (durch Koordinaten beschreibbares Spielfeld) wird nicht getestet. Da das Spielfeld aus Knoten und Kanten besteht und rechteckig ist, kann immer ein Koordinatensystem angelegt werden.

2.4 Vorgehen

Der folgende Abschnitt beschreibt die übergeordnete Vorgehensweise für die Funktions- und Systemtests. Das Testvorgehen bedient sich des Bottom-Up-Prinzips des V-Modells. Zunächst werden die Komponenten und ihre Funktionsweise einzeln getestet, anschließend folgt ein Integrationstest, der das Zusammenspiel einzelner Komponenten miteinander überprüft. Zur Vorbereitung auf den Abnahmetest mit dem Kunden wird das gesamte System einem Systemtest unterzogen, um die funktionalen und nichtfunktionalen Anforderungen am Gesamtsystem zu überprüfen.

1. Komponententests
   Bei den Komponententest sollen die in Punkt 2.1 genannten Komponenten möglichst isoliert voneinander auf eine korrekte Funktion überprüft werden. Wenn sichergestellt werden kann, dass die einzelnen Komponenten für sich genommen fehlerfrei arbeiten, können eventuell auftretende Fehler in späteren Entwicklungsphasen besser eingegrenzt werden.

2. Integrationstest
   Der Integrationstest überprüft das Zusammenspiel der einzelnen Komponenten. Durch den zuvor durchgeführten Komponententest können Fehler, die auf falschen Funktionsweisen einzelner Komponenten basieren, ausgeschlossen werden, sodass das Hauptaugenmerk der Test auf der korrekten Kommunikation zwischen den Komponenten liegen wird.
   Insbesondere muss getestet werden, ob die Paketkommunikation zwischen den Servern der beiden Teams korrekt funktioniert und ob die übermittelten Werte den abgesprochenen Anforderungen genügen, da die KI ansonsten auf Grundlage falscher Informationen agieren würde. Weitere Tests beziehen sich auf die Übertragung der von der KI berechneten Werte. Diese müssen der der GUI als Benutzerschnittstelle und den Robotern als Spielfiguren korrekt vorliegen, um das Spielgeschehen anzeigen zu können.

3. Systemtest
   Ist durch die beiden vorherigen Tests sichergestellt, dass die einzelnen Komponenten fehlerfrei funktionieren und kommunizieren, wird das gesamte System anhand der Kriterien des Pflichtenhefts und den daraus resultierenden funktionalen und nichtfunktionalen Anforderungen überprüft. Hierfür eignen sich Black-Box-Test, die sich lediglich auf die geforderten Spezifikationen stützen und Implementierungsdetails außer Acht lassen. Der Systemtest

konzentriert sich hauptsächlich auf die Spielmechanik und wird einen realistischen Betrieb des Systems simulieren.

2.5 Testumgebung

Nachfolgend werden die genutzte Testumgebung und Tools aufgelistet und beschrieben.

Infrastruktur
Das Entwickeln und Testen der Robotersteuerung findet im Lego-Labor des Instituts für Programmierung und Reaktive Systeme, Raum IZ 033B, statt. Der Raum bietet die Möglichkeit, dass beide Teams ungestört arbeiten können und beschränkt den Zutritt von unberechtigten Personen. Zudem ist im Lego-Labor die benötigte Infrastruktur vorhanden. Diese besteht aus einem unabhängigen WLAN-Netz, dem Teilelager der Roboter und des auf den Boden mittels kontrastreicher Klebestreifen angebrachten Gitternetzes.

Roboter
Als Spielfiguren werden Roboter der Lego Mindstorms EV3-Serie genutzt. Das mitgelieferte Lego-Betriebssystem wird durch die leJOS Firmware in Version 0.9.0 ersetzt, das eine Programmierung der Roboter in Java erlaubt. Die Kommunikation zwischen Roboter und Server erfolgt per WLAN-Modul über das WLAN-Netz des Lego-Labors.

Server
Als Server kann ein beliebiger Computer mit einem installierten Java Runtime Environment in Version 8 (JRE), Grafikausgabe, Maus- und Tastatureingabe sowie einer Möglichkeit zur Verbindung mit dem WLAN-Netz genutzt werden.
Als Java-Schnittstelle in der Entwicklungs- und Testumgebung wird das Java Development Kit 8 (JDK) eingesetzt. Die Tests des Codes werden durchgeführt mit JUnit 4.

GUI
Die Bedienbarkeit der GUI wird aufgrund des geringen Funktionsumfangs nicht formell getestet. Hier fließen die Erkenntnisse aus den Systemtests direkt in den Entwicklungsprozess ein.

3 Abnahmetest

Im Vordergrund des Abnahmetestes stehen besonders die Ergebnisse der Ausführung beliebig vieler Funktionen des Produktes, die vom Kunden abgenommen werden können.

Dabei wird auf zwei wesentliche Punkte geachtet: Zum einen muss der Kunde das Produkt auf Vollständigkeit der Spezifikationen überprüfen und zum anderen einen Test der geforderten Funktionalitäten durchführen.

Ebenfalls muss die Benutzeroberfläche auf ihre Zweckmäßigkeit getestet werden. Damit der Kunde feststellen kann, dass korrekt gearbeitet wurde, liegt eine gute Dokumentation der ausgeführten Arbeit vor.

3.1 Zu testende Anforderungen

Die folgende Tabelle führt alle zu testenden Funktionen und ihre jeweiligen Testfälle auf. Die Funktionsfähigkeit ist gewährleistet, wenn jeder Testfall überprüft wurde.

Nr	Anforderung	Testfälle	Kommentar
1	Ist der Server an einem TCP/IP fähigen Netzwerk angeschlossen? $\langle F60 \rangle$	$\langle T100 \rangle$	Damit das Spiel funktioniert, muss eine bestehende Netzwerkverbindung vorhanden sein.
2	Sind die WLAN-Schnittstellen von Server und Roboter funktionsfähig und können sie miteinander kommunizieren? $\langle F60 \rangle$	$\langle T100 \rangle$ $\langle T200 \rangle$	
3	Erhält die GUI Datenpakete vom Server und der Server von der GUI? $\langle F110 \rangle$	$\langle T200 \rangle$ $\langle T300 \rangle$	GUI und Server müssen miteinander kommunizieren damit das Spiel gestartet bzw. beendet werden kann und der Spielverlauf dargestellt wird.

4	Kann eine Spielfeldkarte vom Benutzer erstellt bzw. gewählt werden? $\langle F120 \rangle$ $\langle F130 \rangle$	$\langle T300 \rangle$ $\langle T400 \rangle$	GUI und Server müssen miteinander kommunizieren damit das Spiel gestartet bzw. beendet werden kann und der Spielverlauf dargestellt wird.
5	Kann das Spiel über die GUI gestartet und abgebrochen werden? $\langle F80 \rangle$ $\langle F90 \rangle$	$\langle T300 \rangle$ $\langle T500 \rangle$ $\langle T600 \rangle$	
6	Wird über die GUI die ausgewählte Steuerungsmöglichkeit ausgeführt? $\langle F110 \rangle$ $\langle F140 \rangle$ $\langle F150 \rangle$	$\langle T300 \rangle$ $\langle T700 \rangle$ $\langle T800 \rangle$	In der GUI kann zwischen manueller oder KI Steuerung gewählt werden.
7	Berechnet die KI die beste Route zum Ziel? $\langle F10 \rangle$ $\langle F30 \rangle$	$\langle T200 \rangle$ $\langle T900 \rangle$ $\langle T1000 \rangle$	
8	Besteht eine Kommunikation zwischen Server und KI bzw. Roboter und KI? $\langle F60 \rangle$ $\langle RM10 \rangle$	$\langle T200 \rangle$	KI kommuniziert mit Roboter und gibt Roboterdaten an Server weiter.
9	Erkennen die Sensoren des Roboters das Gitternetz? $\langle F30 \rangle$	$\langle T1000 \rangle$	
10	Führt der Roboter seine angewiesenen Aktionen korrekt aus? $\langle F10 \rangle$ $\langle F20 \rangle$ $\langle F30 \rangle$ $\langle F40 \rangle$ $\langle F50 \rangle$	$\langle T1100 \rangle$ $\langle T1200 \rangle$ $\langle T1300 \rangle$ $\langle T1400 \rangle$ $\langle T1500 \rangle$	Roboter fährt, blinkt, greift Gegner an und nimmt die Flagge auf.
11	Senken sich nach einem gegnerischen Angriff die Lebenspunkte? $\langle F40 \rangle$	$\langle T1300 \rangle$ $\langle T1600 \rangle$	
12	Hat das Spiel einen definierten Endzustand? $\langle F100 \rangle$	$\langle T1700 \rangle$	Das Spiel wird durch den Sieg einer Gruppe beendet.

3.2 Testverfahren

Das Zusammenspiel der einzelnen Komponenten des Produktes werden durch das dynamische Black-Box-Verfahren getestet. Dies geschieht in möglichst praxisnaher Umgebung.

Durch die Eingabe bekannter Attribute in der Benutzeroberfläche oder durch von außen sichtbare Schnittstellen muss das Produkt auf erwartete Weise reagieren und korrekte Zwischenergebnisse und Ausgaben liefern. Es werden die beobachteten Verhalten der einzelnen Testfälle mit denen der funktionalen Anforderungen verglichen und ausgewertet.

Bei Veränderung der Daten durch Eingabe eines Benutzers werden durch Kombinationen der Grenzfälle überprüft.

Es werden dabei keine Testskripte verwendet.

3.3 Testfälle

Im folgenden Abschnitt werden die Testfälle beschrieben.

3.3.1 Testfall $\langle T100 \rangle$ - Test der Datenverbindung

Ziel

Überprüfung funktionierender Datenverbindung mittels TCP/IP und WLAN

Objekte/Methoden/Funktionen

Objekte: Server, Roboter

Funktionen: WLAN-Datenverbindung $\langle F60 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn eine Verbindung hergestellt werden kann.

Vorbedingung

Das Programm ist in Betrieb und der Roboter muss eingeschaltet sein.

Einzelschritte

Ausgabe:

1. Der Roboter zeigt, dass er mit dem Server verbunden ist.

Beobachtungen / Log / Umgebung

Beobachtet wird der Display des Roboters bezüglich erwarteter Anzeige.

3.3.2 Testfall $\langle T200 \rangle$ - Test der Kommunikation

Ziel

Überprüfung der Kommunikation zwischen Server und Roboter, Server und Server und Server und GUI.

Objekte/Methoden/Funktionen

Objekte: Server, Roboter, GUI

Pass/Fail Kriterien

Der Test gilt als erfolgreich bestanden, wenn das Datenpaket gültig von B empfangen wurde. Im Umkehrschluss schlägt der Test fehl, wenn das Datenpaket nicht oder fehlerhaft Empfangen wurde.

Vorbedingung

Das Programm ist in Betrieb und der Roboter muss eingeschaltet sein. Zudem muss eine Netzwerkverbindung bestehen.

Einzelschritte

Eingabe:

1. Komponente A sendet ein Datenpaket zu Komponente B.

Ausgabe:

1. Änderung der Angaben, z.B. Position auf der GUI
2. Änderung der Angaben, z.B. Lebenspunkte auf dem Roboterdisplay

Beobachtungen / Log / Umgebung

Beobachtet wird der Display des Roboters und die GUI bezüglich erwarteter Anzeige.

Abhängigkeiten

Dieser Testfall hängt vom Testfall $\langle T100 \rangle$ ab, da eine Netzwerkverbindung bestehen muss.

3.3.3 Testfall $\langle T300 \rangle$ - Test der Benutzeroberfläche

Ziel

Es besteht eine funktionstüchtige Benutzeroberfläche, die mit dem Server kommunizieren kann.

Objekte/Methoden/Funktionen

Objekte: Server, GUI

Funktionen: Das Spiel muss gestartet $\langle F80 \rangle$, abgebrochen $\langle F90 \rangle$ und dargestellt $\langle F110 \rangle$ werden können

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die Benutzeroberfläche angezeigt, gestartet oder abgebrochen werden kann und das laufende Spiel angezeigt wird.

Vorbedingung

Das Programm ist in Betrieb. Eine Verbindung zum Server muss bestehen.

Einzelschritte

Eingabe:

1. Start
2. Abbruch

Ausgabe:

1. Spiel wird gestartet, die auf der GUI dargestellten Roboter setzen sich in Bewegung.
2. Spiel wird abgebrochen, die auf der GUI dargestellten Roboter stoppen ihre jeweilige Aktion.

Beobachtungen / Log / Umgebung

Beobachtet wird, ob auf dem Display des Computers eine korrekte Ausgabe erfolgt.

Abhängigkeiten

Dieser Testfall hängt vom Testfall $\langle T200 \rangle$ ab, da die GUI mit dem Server kommunizieren muss.

3.3.4 Testfall $\langle T400 \rangle$ - Test des Karteneditors

Ziel

Überprüfung der einzelnen Funktionen des Karteneditors.

Objekte/Methoden/Funktionen

Objekte: GUI

Funktionen: Es muss eine Karte erstellbar $\langle F120 \rangle$ und wählbar $\langle F130 \rangle$ sein.

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn eine neue Karte erstellt und eine vorhandene Karte geladen werden kann.

Vorbedingung

Das Programm muss gestartet und die Benutzeroberfläche geöffnet sein.

Einzelschritte

Eingabe:

1. Neue Karte
2. Karte laden

Ausgabe:

1. Anzeige der neuen Karte
2. Anzeige der geladenen Karte

Beobachtungen / Log / Umgebung

Beobachtet wird das Display des Computers, im besonderen das Fenster der GUI in dem der Karteneditor geöffnet ist.

Abhängigkeiten

Dieser Testfall hängt vom Testfall $\langle T300 \rangle$ ab, da die GUI funktionieren muss.

3.3.5 Testfall $\langle T500 \rangle$ - Start des Spieles durch Benutzer

Ziel

Das Spiel kann vom Benutzer gestartet werden.

Objekte/Methoden/Funktionen

Objekte: GUI, Server, Roboter, KI

Funktionen: Spiel starten $\langle F80 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn das gestartete Spiel auf der Benutzeroberfläche angezeigt wird.

Vorbedingung

Das Programm und die Roboter müssen gestartet sein und es muss eine Verbindung zum Server bestehen.

Einzelschritte

Eingabe:

1. Spiel starten

Ausgabe:

1. Gestartetes Spiel wird auf GUI angezeigt

Beobachtungen / Log / Umgebung

Beobachtet wird die GUI ob das gestartete Spiel angezeigt wird.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T100 \rangle$, $\langle T200 \rangle$ und $\langle T300 \rangle$ ab, da eine Kommunikation zwischen den einzelnen Objekten bestehen muss und die Benutzeroberfläche funktionieren muss.

3.3.6 Testfall $\langle T600 \rangle$ - Abbruch des Spiels durch Benutzer

Ziel

Das Spiel kann vom Benutzer abgebrochen werden.

Objekte/Methoden/Funktionen

Objekte: GUI, Server

Funktionen: Spieler bricht Spiel ab $\langle F90 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn ein laufendes Spiel abgebrochen werden kann.

Vorbedingung

Das Programm ist gestartet und hat eine bestehende Verbindung zum Server. Außerdem muss ein Spiel gestartet sein.

Einzelschritte

Eingabe:

1. Spiel abbrechen

Ausgabe:

1. Abgebrochenes Spiel

Beobachtungen / Log / Umgebung

Beobachtet wird die GUI um zu erkennen, ob das Spiel abbricht.

Abhängigkeiten

Dieser Testfall hängt von Testfall $\langle T500 \rangle$ ab, da bereits ein Spiel gestartet sein muss.

3.3.7 Testfall $\langle T700 \rangle$ - Manuelle Steuerung

Ziel

Das Spiel kann vom Benutzer selbst gespielt werden. Der Benutzer steuert den Roboter.

Objekte/Methoden/Funktionen

Objekte: GUI, Server, Roboter

Funktionen: Manuelle Steuerung $\langle F150 \rangle$, Steuerung wählen $\langle F140 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Benutzer das Spiel manuell steuern kann.

Vorbedingung

Ein neues Spiel muss gestartet und die mauelle Steuerung gewählt sein.

Einzelschritte

Eingabe:

1. Manuelle Steuerung
2. Roboter auswählen
3. Koordinaten des Zielfelds eingeben

Ausgabe:

1. Steuerungsoptionen für Benutzer
2. Roboter bewegt sich zu Feld

Beobachtungen / Log / Umgebung

Beobachtet wird die GUI, ob das Spiel manuell ausgeführt wird.

Besonderheiten

Diese Option kann nur vor dem Spielstart ausgewählt werden.

Abhängigkeiten

Dieser Testfall hängt von Testfall $\langle T500 \rangle$ ab, da es möglich sein muss das Spiel zu starten.

3.3.8 Testfall $\langle T800 \rangle$ - KI Steuerung

Ziel

Das Spiel wird von der entwickelten KI gespielt.

Objekte/Methoden/Funktionen

Objekte: Server, KI, Roboter, GUI

Funktionen: Spielsteuerung wählen $\langle F140 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die KI das Spiel selbstständig spielt.

Vorbedingung

Ein neues Spiel wurde gestartet und die KI Steuerung wurde ausgewählt.

Einzelschritte

Eingabe:

1. KI Steuerung

Ausgabe:

1. Die KI berechnet die Spielzüge
2. Die Roboter bewegen sich zu der von der KI vorgegebenen Position

Beobachtungen / Log / Umgebung

Beobachtet werden die GUI und der Roboter, um die Ausführung der KI Befehle zu erkennen.

Besonderheiten

Die Option kann nur vor dem Spielstart ausgewählt werden.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T500 \rangle$ und $\langle T1500 \rangle$ ab, da es möglich sein muss ein Spiel zu starten und der Roboter die Aktionen korrekt ausführen muss.

3.3.9 Testfall $\langle T900 \rangle$ - Spielzug Berechnung

Ziel

Die KI berechnet den nächsten Spielzug des Roboters.

Objekte/Methoden/Funktionen

Objekte: KI, Roboter, Server

Funktionen: Spielzug berechnen $\langle F70 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die KI einen Spielzug für den Roboter berechnen kann.

Vorbedingung

Das Spiel ist mit KI Steuerung gestartet und das Team der KI ist am Zug.

Einzelschritte

Eingabe:

1. KI erhält Daten vom Server
2. KI berechnet Zug

Ausgabe:

1. Berechneter Spielzug

Beobachtungen / Log / Umgebung

Beobachtet wird, ob der Roboter einen Befehl von der KI erhalten hat und folglich Aktionen ausführt.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T500 \rangle$ und $\langle T1500 \rangle$ ab, da es möglich sein muss ein Spiel zu starten und der Roboter die Aktionen korrekt ausführen muss.

3.3.10 Testfall $\langle T1000 \rangle$ - Roboter erkennt Spielfeld

Ziel

Der Roboter erkennt die Gitternetzlinien des Spielfeldes.

Objekte/Methoden/Funktionen

Objekte: Roboter

Funktionen: Gittererkennung $\langle F30 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Roboter das Gitternetz erkennt.

Vorbedingung

Der Roboter muss sich auf dem Gitternetz befinden und eingeschaltet sein.

Einzelschritte

Eingabe:

1. Roboter auf Gitternetz stellen
2. Roboter einschalten
3. Bewegungsbefehl erteilen

Ausgabe:

1. Der Roboter bewegt sich zielgerichtet auf dem Gitternetz
2. Auf dem Display werden Daten des Sensors ausgegeben

Beobachtungen / Log / Umgebung

Beobachtet wird, ob sich der Roboter zielgerichtet auf dem Gitternetz bewegt und ob Sensordaten auf dem Display des Bricks erscheinen.

3.3.11 Testfall $\langle T1100 \rangle$ - Roboter fährt

Ziel

Der Roboter bewegt sich von seiner aktuellen Position zu einer anderen.

Objekte/Methoden/Funktionen

Objekte: Roboter, Server

Funktionen: Bewegen $\langle F10 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Roboter sich an die gewünschte Position bewegt.

Vorbedingung

Ein Spiel muss gestartet sein und das Team des Roboters muss am Zug sein.

Einzelschritte

Eingabe:

1. Fahre zu Punkt P

Ausgabe:

1. Roboter fährt zu Punkt P

Beobachtungen / Log / Umgebung

Es kann beobachtet werden, dass der Roboter sich an den gewünschten Punkt bewegt.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T500 \rangle$ und $\langle T200 \rangle$ ab, da ein Spiel gestartet sein muss und der Roboter mit dem Server kommunizieren muss um Befehle zu erhalten.

3.3.12 Testfall $\langle T1200 \rangle$ - Roboter greift an

Ziel

Der Roboter greift einen gegnerischen Roboter an.

Objekte/Methoden/Funktionen

Objekte: Roboter, Server

Funktionen: Angriff $\langle F20 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Roboter seinen Angriff durch Blinken visualisiert und der angegriffene Roboter seine Lebenspunkte um eins verringert.

Vorbedingung

Ein Spiel ist gestartet, der anzugreifende Roboter befindet sich in Schussweite und die Sicht wird nicht durch ein Hindernis eingeschränkt.

Einzelschritte

Eingabe:

1. Greife Roboter R an

Ausgabe:

1. Greift Roboter R an
2. Blinkt

Beobachtungen / Log / Umgebung

Das Display und die Leuchten des Roboters werden beobachtet und die Visualisierung des Angriffs zu beobachten.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T500 \rangle$, $\langle T200 \rangle$ und $\langle T1300 \rangle$ ab, da ein Spiel gestartet sein, der Roboter, um Befehle zu erhalten, mit dem Server kommunizieren und blinken können muss.

3.3.13 Testfall $\langle T1300 \rangle$ - Roboter blinkt

Ziel

Der Roboter blinkt, wenn er angreift bzw. angegriffen wurde.

Objekte/Methoden/Funktionen

Objekte: Roboter, Server

Funktionen: Blinken $\langle F40 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Roboter blinkt.

Vorbedingung

Das Spiel ist gestartet und der Roboter greift an oder wurde angegriffen.

Einzelschritte

Eingabe:

1. Roboter greift an
2. Roboter wird angegriffen

Ausgabe:

1. Blinkt

Beobachtungen / Log / Umgebung

Beobachtet werden die Leuchten das Roboters um das Blinken zu erkennen.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T500 \rangle$ und $\langle T200 \rangle$ ab, da ein Spiel gestartet sein muss und der Roboter mit dem Server kommunizieren muss um Befehle zu erhalten.

3.3.14 Testfall $\langle T1400 \rangle$ - Roboter nimmt Flagge auf

Ziel

Der Roboter nimmt die gegnerische Flagge auf.

Objekte/Methoden/Funktionen

Objekte: Roboter, Server

Funktionen: Flaggenaufnahme $\langle F50 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Roboter die Flagge aufgenommen hat.

Vorbedingung

Das Spiel ist gestartet und der Roboter befindet sich auf dem Flaggenfeld.

Einzelschritte

Eingabe:

1. Flagge aufnahemen

Ausgabe:

1. Nimmt Flagge auf

Beobachtungen / Log / Umgebung

Beobachtet wird das Display des Roboters um die visualisierung der aufgenommenen Flagge zu sehen, da keine physische Flagge auf dem Spielfeld steht.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T500 \rangle$ und $\langle T200 \rangle$ ab, da ein Spiel gestartet sein muss und der Roboter mit dem Server kommunizieren muss um Befehle zu erhalten.

3.3.15 Testfall $\langle T1500 \rangle$ - Roboter führt Aktionen korrekt aus

Ziel

Der Roboter soll vom Server empfangene Befehle korrekt ausführen.

Objekte/Methoden/Funktionen

Objekte: Roboter, Server

Funktionen: Bewegen $\langle F10 \rangle$, Angriff $\langle F20 \rangle$, Flaggeaufnehmen $\langle F50 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Roboter alle vom Server gesendeten Befehle korrekt ausführt.

Vorbedingung

Das Spiel ist gestartet, der Roboter ist mit dem Server verbunden und erhält Befehle von diesem.

Einzelschritte

Eingabe:

1. Fahr zu Punkt P
2. Greif Roboter R an
3. Nimm Flagge auf

Ausgabe:

1. Fährt zu Punkt P
2. Greift Roboter R an
3. Nimmt Flagge auf

Beobachtungen / Log / Umgebung

Überwacht wird der Roboter um die korrekte Ausführung aller Aktionen zu beobachten.

Abhängigkeiten

Dieser Testfall hängt von den Testfällen $\langle T500 \rangle$, $\langle T200 \rangle$, $\langle T1100 \rangle$, $\langle T1200 \rangle$ und $\langle T1400 \rangle$ ab, da ein Spiel gestartet sein, der Roboter mit dem Server kommunizieren und die angegbenen Aktionen korrekt ausführen muss.

3.3.16 Testfall $\langle T1600 \rangle$ - Virtuelle Lebenspunktanzeige

Ziel

Der Roboter kann seine verbleibenden Lebenspunkte auf dem Display anzeigen.

Objekte/Methoden/Funktionen

Objekte: Roboter

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn auf dem Display des Roboters seine Lebenspunkte angezeigt werden.

Vorbedingung

Das Spiel ist gestartet und der Roboter ist eingeschaltet.

Einzelschritte

Ausgabe:

1. Lebenspunkte

Beobachtungen / Log / Umgebung

Auf dem Display des Roboters können die Lebenspunkte beobachtet werden.

Abhängigkeiten

Dieser Testfall hängt von Testfall $\langle T500 \rangle$ ab, da das Spiel gestartet sein muss.

3.3.17 Testfall $\langle T1700 \rangle$ - Spiel beenden

Ziel

Der Server beendet das Spiel und der Spielstand wird ausgegeben.

Objekte/Methoden/Funktionen

Objekte: Server

Funktionen: Spielende $\langle F100 \rangle$

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn das Spiel beendet wurde.

Vorbedingung

Es ist ein Spiel gestartet.

Einzelschritte

Eingabe:

1. Spiel beenden

Ausgabe:

1. Spielstand

Beobachtungen / Log / Umgebung

Beobachtet wird die GUI, auf der der Spielstand ausgegeben wird.

Abhängigkeiten

Dieser Testfall hängt von Testfall $\langle T500 \rangle$ ab, da ein Spiel gestartet sein muss um es zu beenden.

4 Glossar

Nachfolgend werden verwendete Fachbegriffe kurz erklärt.

Lego Mindstorms EV3:
Roboter der Produktserie Mindstorms von Lego, die mittels eines programmierbaren Legosteins (genannt *Brick*), Elektromotoren sowie Sensoren autonom Aufgaben erfüllen können.

Lego Mindstorms EV3 Brick:
Hierbei handelt es sich um den programmierbaren Rechner der Lego Roboter.

leJOS:
Java-Betriebssystem für Lego Mindstorms Roboter. Es erlaubt die Steuerung in Java zu programmieren und bietet einen Mehrwert gegenüber der originalen Lego-Software.

Karte:
Dies ist die visuelle Repräsentation der *Capture the Flag*-Welt auf Serverseite, die das Spiel für den Nutzer überschaubar macht.