Dokumentation

Fabian Bruhns, Jan-Henrik Bruhn, Sven Mehlhop

26. Februar 2018

Inhaltsverzeichnis

Fabian:

- Einleitung
- Hauptteil Algorithmen subsection enforceGraph
- System GUI
- Ende Subsection Optimierung

Jan-Henrik:

- Hauptteil Algorithmen subsection Spielgraph
- Ende Ergebnisse
- Ende aufgetretene Fehlher

Sven:

- Layout
- Hauptteil Algorithmen subsection Controller to review
- System Umgebungsvariablen ?????
- System anforderungen ?????

1 Einleitung

Kurze Projektbeschreibung.... Ziel des Projekts etc...

2 System

2.1 Umgebungsvariablen

Roboter vs Kind Bewegungsmuster Spielfeld um Begrenzung und Mauern erweitert

2.2 GUI

Die Benutzeroberfläche zur Visualisierung des Problems und dem anschließenden Lösungsweg, wurde so gewählt das der Nutzer zu jedem Zeitpunkt die Schritte das Programmes nachvollziehen kann. Dazu wurde die GUI von der Logik getrennt und besteht als eigenständiger Teil im Projekt. Die GUI bezieht von einem Spielfeld wo sich die Objekte befinden. Dieses Spielfeld wird von der Prozesslogik verändert und von jener ausgelesen. Nach jeder Veränderung wird das Feld als Benutzeroberfläche neu gezeichnet.
Dem Benutzer ist es zu aller erst möglich das Spielfeld durch das Platzieren von Objekten zu gestalten. Dabei stehen dem Benutzer Wände, Roboter, Kind, Ladestationen und Zielfelder zur Verfügung. Der Roboter beschreibt das Feld auf dem der Roboter auf dem gekachelten Spielfeld startet. Dem entsprechend verhält sich das Kind zu seiner Figur. Die Batterien stellen Ladestationen für den Roboter da, welche er in einem Spielmodus besuchen muss, um seine Energie/Ladung wieder aufzufüllen. Die Zielfelder beschreiben die Felder die der Roboter in chronologischer Reihenfolge abarbeiten muss. Dabei wird das als nächstes angestrebte Feld grün markiert.
Ist das Spielfeld erstellt kann der Benutzer im folgenden die Bewegungsmöglichkeiten von Kind und Roboter frei wählen. Dabei sind auch Variationen möglich mit welchen der Roboter sein Ziel nicht erreichen kann, da die GUI mit Rückmeldungen an den Benutzer ausgestattet ist, welche dieses zur Laufzeit wiedergeben.
Um das Programm zu starten ist der Start-Button zu drücken. Dieser ermöglicht es dem Benutzer nach dem Laden der Logik das Programm zu Beschleunigen oder gänzlich zu stoppen. Für das Beschleunigen steht ein Regler zur Verfügung, welcher die Dauer zwischen den Schritten beschränkt oder verlängert. Das Stoppen wird durch einen gleichnamigen Button realisiert, welcher das Programm wieder in den Bearbeitungszustand setzt.
In den Menüreitern können weitere Einstellungen getroffen werden, wie Sprache, oder ob die Tankfüllung beachtet werden soll. Im Programm werden dann dynamisch die Sprache angepasst und die Tankfüllung angezeigt (oben rechts im Feld).
Sollten Fragen zum Spiel aufkommen können diese jederzeit mit dem Hilfefenster versucht geklärt zu werden. Weiterhin ist es noch möglich die Spielfeldgröße zu verändern.

3 Algorithmen

probleme übersicht roboter probleme übersicht gruppe

3.1 Spielgraph

To review!
Da der Roboter seine Entscheidungen basierend auf den Enforce-Algorithmen treffen soll, müssen wir einen endlichen Spielgraphen erstellen. Ein solcher Graph besteht aus Zuständen, in denen der Spieler (in diesem Fall der Roboter) und Zuständen in denen die Umgebung (in diesem Fall das Kind) einen Zug macht. Ein Zug wird dabei von einer Kante im Graphen dargestellt.

In unserem Fall sind die möglichen Züge die jeweiligen Bewegungsmuster, die den Spielern mitgegeben werden. Demnach besteht unser Spielgraph $G = (Q, I, Q_0, Q_1, A_0, A_1, \Omega)$, orientiert an den Vorlesungsunterlagen, aus diesen Elementen:

- Q: Menge aller Zustände
- $I \subseteq Q_0$: Menge aller Startzustände
- Q_0 und Q_1: Teilmengen von Q
  - Q_0: Zustände in denen der Roboter am Zug ist
  - Q_1: Zustände in denen das Kind am Zug ist
- A_0: Züge des Roboters
- A_1: Züge des Kindes
- $\Omega \subseteq Q$: Zustände, in denen der Roboter auf dem Ziel steht

Die Zustände enthalten dabei die Positionsinformation des Roboters und des Kindes, und welcher Spieler am Zug ist.

Um einen Spielgraphen für unser Spielfeld zu erhalten generieren wir also, ausgehend von einem Initialzustand $Q0$, alle möglichen darauf folgenden Transitionen, abhängig davon, ob der daraus entstehende Folgezustand valide ist. Es werden dabei jeweils entweder die Bewegungesmuster des Kindes oder die des Roboters gewählt. Die daraus folgenden Zustände werden in die Datenstruktur mit aufgenommen. Wenn jene zuvor noch nicht bekannt waren, werden auch von diesen Zuständen die Folgezustände generiert, bis keine neuen Zustände mehr generiert werden.

Da schon der Spielgraph illegale Zustände vermeiden muss, wird hier bereits beachtet, ob der Roboter mit dem Kind kollidiert oder nicht. Die offensichtliche, einfache Lösung, unabhängig davon wer am Zug ist illegale Folgezustände zu verbieten entspräche natürlich nicht der Aufgabenstellung, weil dann auch das Kind dem Roboter aktiv ausweichen würde. Es soll aber der Roboter dem Kind ausweichen, also ggf. auch vorhersehen ob er mit dem Kind kollidiert. Diese Funktionalität wird umgesetzt, indem für alle generierten Folgezustände des Roboters zusätzlich die darauf folgenden Möglichen Züge

des Kindes berechnet werden. Enthält diese zweite Menge einen Zustand, in dem das Kind in den Roboter hineinspringt, wird der aktuelle Roboterzustand verworfen, da er zu einem illegalen Zug führen kann.

Wird die Graphengenerierung so umgesetzt erhalten wir bereits einen Roboter, der dem Kind aktiv ausweichen kann, egal welchen Zug das Kind macht. Dies gilt natürlich nur unter der Voraussetzung, dass die angegebenen Bewegungsmuster dies Zulassen. Kann das Kind bspw. sich frei bewegen ($A_1 = u, d, l, r$), der Roboter jedoch gar nicht ($A_0 = e,$), so kann der Roboter auch keine Kollision vermeiden.

3.2 Enforce

3.3 Controller

To review!

Der Controller ist für die Schnittstelle zwischen dem Graphen, inklusive des Enforce-Graphen und der GUI zuständig. Er bezieht aus der GUI die Positionierungen des Kindes und des Roboters, sowie die Standorte der Mauern, Zielfelder und Batterien. Er wird mit Start des Programms aus der GUI-Umgebung gestartet. Daraufhin initialisiert er den Graphen und berechnet mit hinzunahme der aktuellen Position des Kindes sowie des Roboters,den Enforce-Graphen. Sollte festgestellt werden, dass es keine Lösung für das Problem gibt, terminiert das Programm. Sollte dies nicht der Fall sein wird der Enforce-Graph für den ersten und die fortlaufenden Schritte des Roboters genutzt. Die Schritte des Kindes wird auch aus diesem Graphen bezogen, dabei bekommt der Controller die möglichen Moves aus dem Graphen und wählt zufällig einen. Mit der Hinzunahme des Energie-Problems wird der Controller um die Aufgabe erweitert das Energie-Level zu überprüfen. Sollte der Worst Case Pfad zum nächsten Zielfeld, zuzüglich des Weges zum nächstbesten Energiefeld, größer als das Energie Level des Roboter sein, bewegt sich der Roboter zum nächsten Batteriefeld. Sollte auch dies nicht möglich sein terminiert das Programm. Der Weg zum nächstbesten Batteriefeld, ist der geringste Weg mit dem geringsten Aufwand aus den Worst Case Pfaden aller Batteriefeld. Als Energiefeld gelten die Felder um die Batterie.

4 Buzzword

4.1 Optimierung

4.2 Fehler

Aufgetretene Fehler

4.3 Ergebnisse

Folgendes haben wir während des Projekts ermittelt.

5 Arbeitsverteilung

Teilaufgabe	Name	Anteil
1. Graphischer Editor mit Eingabe der Definitionen	Fabian	90%
	Jan-Henrik	5%
	Sven	5%

6 Erklärung

Hiermit versichern wir, dass wir diese Arbeit selbständig verfasst und keine anderen, als die angegebenen Quellen und Hilfsmittel benutzt, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Arbeit in der aufgeführten Arbeitsverteilung im Abschnitt 5 erledigt.

Sven Mehlhop