Chat und 4-gewinnt-Spiel mit Docker und Node.js

Projekdokumentation Microservices

an der Dualen Hochschule Baden-Württemberg Stuttgart

von

Hanna Siegfried und Nahku Saidy

12.06.2019

Bearbeitungszeitraum	nothing
Matrikelnummer, Kurs	5946066,XXXXX, STG-TINF17-ITA
Ausbildungsfirma	Daimler AG, Stuttgart
Betreuer	nothing

Inhaltsverzeichnis

Abkürzungsverzeichnis

AABB Axis-Aligned Bounding Box

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1 Einleitung

1.1 Aufgabenstellung und Ziel der Projektarbeit

Die Aufgabe bestand in der Entwicklung einer Beispielanwendung unter Verwendung zweier Aspekte aus der Vorlesung. Wir haben uns für die Entwicklung eines 4-gewinnt-Spiels auf Basis der Chat-Anwendung entschieden. Dieses soll mittels Docker deploybar gemacht werden. Die Chat-Teilnehmer sollen mit einer Registrierungsmöglichkeit in einer MongoDB verwaltet werden, welche auch in einen Docker-Container verpackt werden soll. Die Entwicklung der Anwendung wird mit Node.js durchgeführt. -docker -mongo.db -docker-compose -node.js

1.2 Aufbau der Projektarbeit

Diese Projektarbeit gliedert sich in fünf Kapitel. Im ersten Kapitel wird eine Einleitung in die Aufgabenstellung und die geplante Anwendung gegeben. Das zweite Kapitel beinhaltet die notwendigen Grundlagen, auf denen die Anwendung basiert. Dabei werden die einzelnen Technologien und ihre Möglichkeiten kurz vorgestellt. Darauffolgend wird in Kapitel 3 das Konzept der Anwendung und ihrer Komponenten dargestellt. Dabei wird auch auf die Vernetzung zwischen zum Beispiel der Datenbank und dem Chat eingegangen. In Kapitel 4 werden dann einzelne Teile der Implementierung vorgestellt, die essenziell für die Funktion der Anwedung sind. Im letzten Kapitel wird noch einmal auf die Aufgabenstellung Rückbezug genommen und eine kritische Würdigung der erzielten Ergebnisse vorgenommen. Zusätzlich dazu wird ein Ausbilck auf mögliche Weiterentwicklungen gegeben.

1.3 Wer hat was gemacht?

2 Grundlagen und Stand der Technik

2.1 Node.JS

2.2 Websockets

Websockets sind die Grundlage für die Chat-Anwendung.

2.3 Docker

Docker ist eine Software ... In dieser Projektarbeit wird Docker verwendet, um das einfache deployen der Anwendungen, unabhängig von den konkreten Servern zu ermöglichen.

2.4 MongoDB

In der Anwendung wird eine MongoDB zum Speichern der Zugangsdaten der Chat-Teilnehmer genutzt.

3 Konzept

3.1 Übersicht

4 Implementierung

4.1 Spiellogik

4.2 Erkennung des Spielendes

Es gibt zwei Szenarien, unter denen das Spiel zu Ende ist. Entweder gewinnt ein Spieler oder das Spiel endet unentschieden. Unentschieden ist das Spiel, wenn das Spielfeld komplett gefüllt ist, aber kein Spieler eine Reihe aus vier Steinen aufbauen konnte. Bei einem Gewinn sind drei verschiedene Szenarien zu unterscheiden. Eine Reihe aus vier Spielsteinen des gleichen Spielers kann horizontal, vertikal oder diagonal auftreten. Wenn die Reihe diagonal gebildet wird, kann noch zwischen von links unten nach rechts oben und von rechts unten nach links oben unterschieden werden. Alle diese Fälle müssen geprüft werden.

5 Projektabschluss, Fazit & Ausblick

5.1 Fazit

Das Ziel dieser Projektarbeit war die Entwicklung eines Konzepts für eine **LiDAR!** (**LiDAR!**)-basierte Objekterkennung und eine prototypische Implementierung dieser. Dazu wurden bereits existierende Clustering-Verfahren diskutiert und deren Vor- und Nachteile für die Verwendung im Kontext dieser Projektarbeit dargestellt. Außerdem wurden konkrete Probleme des **DBSCAN!** (**DBSCAN!**)-Algorithmus mit dem Clustering von **LiDAR!**-Daten herausgearbeitet. Diesen wurde durch eine Erweiterung des Algorithmus mit einer distanzabhängigen Berechnung des Parameters ϵ entgegengewirkt (Kapitel **??**).
Die erkannten Objekte werden mit eindeutigen IDs und Axis-Aligned Bounding Box (AABB)s im Sensordatenbild visualisiert.
Zusätzlich wurde eine Tracking-Funktion entwickelt, die das Tracken eines einzelnen Objekts mittels des Kalman-Filter ermöglicht und den Bewegungsverlauf zusätzlich zu den Clustering-Ergebnissen im Sensordatenbild darstellt. Dieses Konzept wurde mittels Matlab implementiert. Dabei wurde eine Anpassung des **DBSCAN!**-Algorithmus vorgenommen, welche die Laufzeit reduziert, da die Analyse eines Messzyklus sonst ca. 70 - 100 Sekunden gedauert hätte (siehe Kapitel **??**). Nach der Verbesserung ergaben sich je nach Messumgebung Laufzeiten von ca. 10 - 50 Sekunden.
Somit konnten die zu Beginn in Kapitel 1.1 definierten Ziele erreicht werden und zusätzlich die Tracking-Funktion entwickelt werden, die es ermöglicht, nicht nur einen Messzyklus zu analysieren, sondern die zukünftige Position eines Objekts anhand der Analyse vorheriger Messzyklen zu schätzen.

5.2 Ausblick

Es gibt einige Bereiche, in denen die Funktion der Objekterkennung erweitert und verbessert werden kann.

Es besteht die Möglichkeit, eine Sensorbewegung in die Berechnungen miteinzubeziehen. Dies bietet sich an, wenn der Sensor auf einem beweglichen Objekt montiert ist. Bisher wurde die Annahme eines stationären Sensors getroffen. Mit einem bewegten Sensor müssten die Koordinaten, die immer relativ zum Sensor aufgenommen werden, in ein Welt-Koordinatensystem umgerechnet werden. Außerdem könnte die Objekterkennung an Echtzeitanforderungen angepasst werden. Durch die hohe Anzahl an Punkten, die sich in einer Punktwolke befinden, ist die Analyse dieser rechenintensiv. Deshalb müsste voraussichtlich eine Vereinfachung der Sensordaten vor der Analyse vorgenommen werden, um die Ausführungszeit zu reduzieren. Aktuell wird der Matlab-Programmcode für jede Ausführung der Objekterkennung neu interpretiert. Würde der Code der Objekterkennung in ein ausführbares Programm übersetzt werden, so ließe sich die Ausführungszeit weiter reduzieren.
In Bezug auf die Tracking-Funktionalität ist es denkbar, diese um ein Tracking von mehreren Objekten zu erweitern. Dazu müsste eine neue Strategie zur Data-Association entwickelt werden. Es ist dabei zu beachten, dass die Tracking-Funktion auf das Tracken von Personen ausgerichtet ist. Sollen auch Objekte anderer Art getrackt werden, so muss auch ggf. eine Anpassung des Bewegungsmodells für den Kalman-Filter in Betracht gezogen werden.
Neben der Funktion des Trackens mehrerer Objekte könnte das Tracking um die Funktion eines automatischen Starts des Trackings erweitert werden. Bisher wählt der Benutzer aus, welches Objekt getrackt werden soll. Durch eine Analyse, welche Objekte beweglich sind, könnte dieser Prozess automatisiert werden, sodass keine Benutzereingabe mehr erforderlich ist.

Literatur

Publikationen

[1] Peter Knoll. *Fahrstabilisierungssysteme und Fahrerassistenzsysteme*. Wiesbaden: Vieweg+Teubner Verlag, 2010. ISBN: 9783834813145.

Anhang

A. Screenshot NameNode Web-Interface

B. DVD Inhalt

C. DVD

A. Screenshot NameNode Web-Interface

C. DVD Inhalt