Universität Leipzig

Fakultät für Mathematik und Informatik
Institut für Informatik

– Bachelorarbeit –

Programmierung einer Browser-Extension zur Anzeige von Datenschutzinformationen im PlayStore sowie Evaluation der Extension-Performance

Author
Alexander Prull
ap62puny@studserv.uni-leipzig.de
Institut für Informatik

First Supervisor	**Second Supervisor**	**External Supervisor**
Prof. Nummer 1	Prof. Nummer 2	Extern Nummer 1
ggg@informatik.uni-leipzig.de	ttt@uni-leipzig.de	rrr.eee@uuu.com
Fancy Computer Science	Institute of Rocket Science	Something AG

27. Oktober 2018

Abstract

In dieser Arbeit wird sich mit den Eigenheiten der Browser Extension Programmierung auseinander gesetzt. Speziell geht es um Extension die Webseiten um bestimmte Informationen erweitern. Diese werden von einem Backend empfangen und zur Ladezeit der Seite eingespeist. Dabei setzt sich die Arbeit mit zwei Punkten auseinander. In erster Linie geht es darum die Ladezeit der Webseite durch das Anfordern von Informationen so wenig wie möglich zu beeinflussen. Also die Performance der Extension zu maximieren. Auf der anderen Seite wird durch die Nutzung der Extension von einer steigenden Nutzerzahl der Backend-Server mit einer steigenden Anzahl von Anfragen belastet. Um diese Probleme zu lösen werden in der Arbeit verschiedene Möglichkeiten zur Speicherung von Daten betrachtet und eine Auswahl der Methoden auf ihre Performance hin getestet.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Im Zeitalter der Digitalisierung werden große Mengen Informationen immer schneller und detaillierter verarbeitet. -?- Jeder, der im Internet unterwegs ist, hinterlässt dabei wertvolle, persönliche Daten. Dabei spielt der Datenschutz eine wichtige Rolle, denn nicht immer werden diese Daten freiwillig preisgegeben. So reguliert die Datenschutzerklärung, welche Nutzerdaten verarbeitet werden. Denn in dieser Erklärung muss jeder Dienstanbieter dem Nutzer zu Beginn des Nutzungsvorgangs über Art, Umfang und Zwecke der Erhebung und Verwendung personenbezogener Daten sowie über die Verarbeitung seiner Daten in Staaten außerhalb des Anwendungsbereichs [...] in allgemein verständlicher Form zu unterrichten"(Telemediengesetz Paragraph 13 1 1)

Das Projekt Privacy Guard hat sich damit beschäftigt, inwiefern diese Datenschutzerklärungen den Vorgaben entsprechen und im Speziellen analysiert, welche Applikationen unvollständige oder mangelhafte DSEs vorweisen. Im Rahmen dieses Projektes entstand die Idee, bereits vor der Installation von Anwendungen, deren Datenschutzerklärungen zu untersuchen und den Nutzer auf mögliche Bedenken hinzuweisen.

1.1 Aufgabenstellung

Die Arbeit befasst sich mit den folgenden Aufgaben:

1. **"Programmierung einer Browser-Extension zur Anzeige von Datenschutzinformationen im PlayStore"**
2. **"Evaluierung von Caching Methoden einer Browser Extension"**

Hauptaugenmerk ist die Erläuterung von Browser-Extensions, Umsetzung eines Beispiels und Limitationen. Welche Arten von Speicher stehen einer Extension zur Verfügung und welche Performance-Ersparnisse kann durch Abspeichern von

Daten die die Extension wiederholt benötigt eingespart werden. Welche Entlastung erfährt der Server mit Backend. Aufbau und Einbindung des ausgewählten Kandidaten

1.2 Aufbau der Arbeit

Zu Beginn werden Recherche Ergebnisse vorgestellt und ausgewertet. Aus den dadurch gewonnenen Resultaten die Aufgaben genauer Definiert. Auf Basis der Recherche entsteht im 1. Teil eine Extension wobei der Fokus darauf liegt, dass diese möglichst übersichtlich bleibt und zur Evaluierung von Speichermethoden dient. Anschließend werden verschiedene Testläufe präsentiert bei denen bestimmte Methoden zur lokalen Speicherung von Daten unter den gleichen Rahmenbedingungen verwendet werden. Die Ergebnisse werden verglichen und den Erwartungen gegenübergestellt. Zuletzt wird ein Fazit gezogen.

Kapitel 2

Vorarbeit

2.1 Recherche zu Browser Extensions

2.1.1 Extension-Programmierung allgemein

Unter einer Extension versteht man ein Programm, welches den Browser um neue Funktionen ergänzt. Durch eigene Oberflächen oder Manipulation der Website erleichtern diese Erweiterungen das Nutzen des Browser. Im Gegensatz zu Plug-Ins haben Extensions Zugriff auf Browser-spezielle Funktionen und sind in der Lage über die Webseite hinaus zu agieren. Plug-Ins werden direkt in eine Webseite eingebettet und sind auf diese beschränkt. Der Oberbegriff „Add-on“ wird heutzutage Hauptsächlich als Synonym für Extension verwendet. Jeder größere Browser stellt eine Plattform zur Verfügung auf denen Extensions angeboten und installiert werden können. In der Regel sind Diese kostenlos. Können auch von außerhalb installiert werden zu Entwicklungszwecken oder wenn nicht auf der Plattform angeboten. Extensions werden in HTML, JavaScript und CSS implementiert. Dabei können alle Bibliotheken verwendet werden, welche den Browserstandards für Extensions entsprechen. Kapitel 2.3.2 befasst sich genauer damit, welche Bedingungen für diese Bibliotheken in Google Chrome gelten. Bekannte Beispiele sind Werbeblock wie UBlock Origin und VPN-Anwendungen wie Hola.

2.1.2 Existieren bereits vergleichbare Extensions

Gesucht wurde nach einer Extension die auf der Play Store Seite den Nutzer datenschutzrelevante Informationen zu den angebotenen Apps liefert, Eine Datenschutzwertung im Playstore vergibt oder den Nutzer Apps nach Berechtigungen die Apps vorschlägt. Extensions werden nach ihrer Kurzbeschreibung in den Suchergebnissen überprüft und bei nicht eindeutig Aufgabenbeschreibung die Infoseite

aufgerufen(Bsp. Safe.ad im Web Store „ecosystem“).Nur deutsche und englische Ergebnisse werden berücksichtigt.

Die Recherche hat ergeben, dass unter den genannten Suchkriterien keine Chrome oder Firefox Extension gefunden wurde die Aufgabenbereich der geplanten Extension abdeckt. Einige aufgeführte Beispiele implementieren einen Teil der geplanten Funktion (Umsortierung, Tracker checken) , aber keine Extension erfüllt alle gewünschten Aufgaben.

2.1.3 Vergleich führender Browser als Plattform für die Extension

2.2 Projekt PGuard

2.2.1 Vorstellung

2.2.2 API-Anbindung für die Extension

2.3 Implementierung einer Google Chrome Extension

2.3.1 Eigenschaften

Die Architektur einer Chrome Extension stellt ein Paket aus mehreren Dateien dar und ist vergleichbar mit anderen Web-Technologien wie zum Beispiel Webseiten. Grundvoraussetzung für eine funktionierende Extension ist die manifest.json, welche grundlegende Informationen für den Browser bereitstellt und festlegt mit welchen Dateien und Rechten die Extension aufgebaut ist. Hinzu kommt mindestens eine HTML-Datei zur Darstellung der Inhalte und mindestens ein Skript zur Umsetzung der Funktionalität. Erweitert werden diese oft durch CSS-Dateien. Externe Bibliotheken wie JQuery können ebenfalls eingebunden werden, müssen aber aufgrund der Policies von Google Chrome vollumfänglich lokal vorliegen. Mehr dazu Im nächsten Abschnitt. Die Manifest-Datei ist im JSON-Format aufgebaut und beinhaltet sämtliche Informationen über die Extension. Wichtige Punkte sind Name der Extension, Beschreibung, Rechte und Aufbau. Unter Rechten oder „permissions“ werden alle APIs aufgelistet, welche die Extension benötigt um ordnungsgemäß zu funktionieren. Bevor ein Nutzer später die Extension installiert, muss er diesen „permissions“ zustimmen. Mehr zu den APIs im entsprechenden Abschnitt. Der Aufbau wird unter „content scripts“ (BILD?) in drei Eigenschaften unterteilt: unter welchem URL sind die Skripte aktiv, welche Skripte sind dort aktiv und welche CSS-Dateien werden dort von der Extension eingesetzt. HTML-Dateien werden als „User-Interface Elemente“ zusammengefasst und beinhalten im Normalfall eine popup.html zur Darstellung des Fensters der Extension in der oberen rechten Ecke des Browser-Fensters (BILD?). Je nach Funktionsumfang

können weitere UI-Elemente eingebunden sein, um zum Beispiel die besuchte Webseite zu erweitern.

Die vorhandenen Skripte werden normalerweise in zwei Kategorien eingeteilt. Das sogenannte „Background-Skript“ dient als Event-Handler und kommuniziert zwischen Extension und Browser. Alle restlichen Skripte sind „Content-Skripte“. Sie beinhalten die eigentliche Funktionalität der Extension.

2.3.2 Funktionsumfang

2.3.3 Darstellung im Browser

2.4 Caching-Methoden

2.4.1 Welche Rolle spielt Performance?

2.4.2 Verwendete Methoden und deren Eigenschaften

Kapitel 3

Hauptteil

This chapter will first outline the problems that constitutes the main portion of this work. Each problem is described separately starting with the available data sources followed by a detailed description of the proposed solutions. After that, the proposed solutions are evaluated by empirical means and the results are presented. Performance studies are conducted to provide suitable recommendations concerning the real world application.

3.1 Erlaeuterung der Aufgabenstellungen

3.2 Aufgabe 1: Implementierung einer Browser-Extension zur Anzeige von Datenschutzinformationen im Play-Store

3.2.1 Anwendungsszenario

QUELLE STATISTIK?

Während vor einigen Jahren Applikationen hauptsächlich auf eigenen Webseiten zum Download angeboten wurden, haben sich die AppStores mitlerweile durchgesetzt. Vorteile für diese Plattformen sind unter anderem: erleichterter Zugang, Vergleiche mit anderen Applikationen und individuelle Empfehlungen. Bei der Wahl für eine bestimmte Applikation achten Nutzer auf Aspekte, wie Preis, Anzahl der Downloads und Bewertungen von anderen Nutzern.

Immer wichtiger aber auch die Frage: Welche Daten gebe ich der Applikation frei und wie werden diese verarbeitet. Der PlayStore bietet zwar einen groben

Überblick, welche Daten eine Applikation von dem Handy ausließt, aber nicht wie diese vom Anbieter verarbeitet werden.

Dadurch entstehen beim Nutzer Fragen, welche der Playstore nicht beantwortet:

1. Handhabung der Daten: Wie werden die Daten verarbeitet und an wen werden diese weitergeleitet? Wird ein Profil anhand der Daten erstellt? Welche Sicherheit besteht bei der Übertragung der Daten?

2. Vor- und Nachteile der Datenverarbeitung: Kann der Anbieter die Applikation dadurch komfortabler gestalten? Wird Werbung in der Applikation personalisiert? Besteht Gefahr vor Missbrauch der Daten?

3. Kontrolle über die Daten: Welche Möglichkeiten stehen zu Verfügung im Falle von Nichteinverständnis? Ist der Umgang mit den Daten nach der Installation noch einschränkbar. Kann der Nutzer die Verwendung der Daten verbieten und trotzdem die App weiterhin nutzen?

3.2.2 Anforderungsanalyse

3.2.2.1 Funktionale Anforderungen

Aus den Fragen die bei dem Anwendungsszenario entstanden sind werden funktionale Anforderungen gebildet um konkrete Aufgaben für die Extension zu schaffen.
-Anforderungen in TEXTFORM-

-Direkt bei Apps in Anforderungen erwähnen

/F10/ Erweiterung der Informationen im PlayStore: Der Nutzer hat die Möglichkeit im Browserfenster per Aktivierung bzw. Deaktivierung der Extension zusätzliche Datenschutzinformationen zu den angezeigten Applikationen ein- bzw. auszublenden.

/F20/ Anzahl von bedenklichen Eigenschaften einer Applikation: Zu jeder Applikation erhält der Nutzer ein Feedback von der Extension, wieviele Bedenken vorliegen.

/F30/ Darstellung von kritischen Eigenschaften einer Applikation: Eigenschaften einer Applikation, welche einen erheblichen Nachteil für den Nutzer darstellen oder einen möglichen Gesetzesverstoß beinhalten werden hervorgehoben.

/F40/ Abrufen von Details zu den Bedenken: Wird ein Bedenken angezeigt, kann der Nutzer direkt Erläuterung, Handlungsempfehlung sowie Vor- und Nachteile zu diesem Bedenken abrufen.

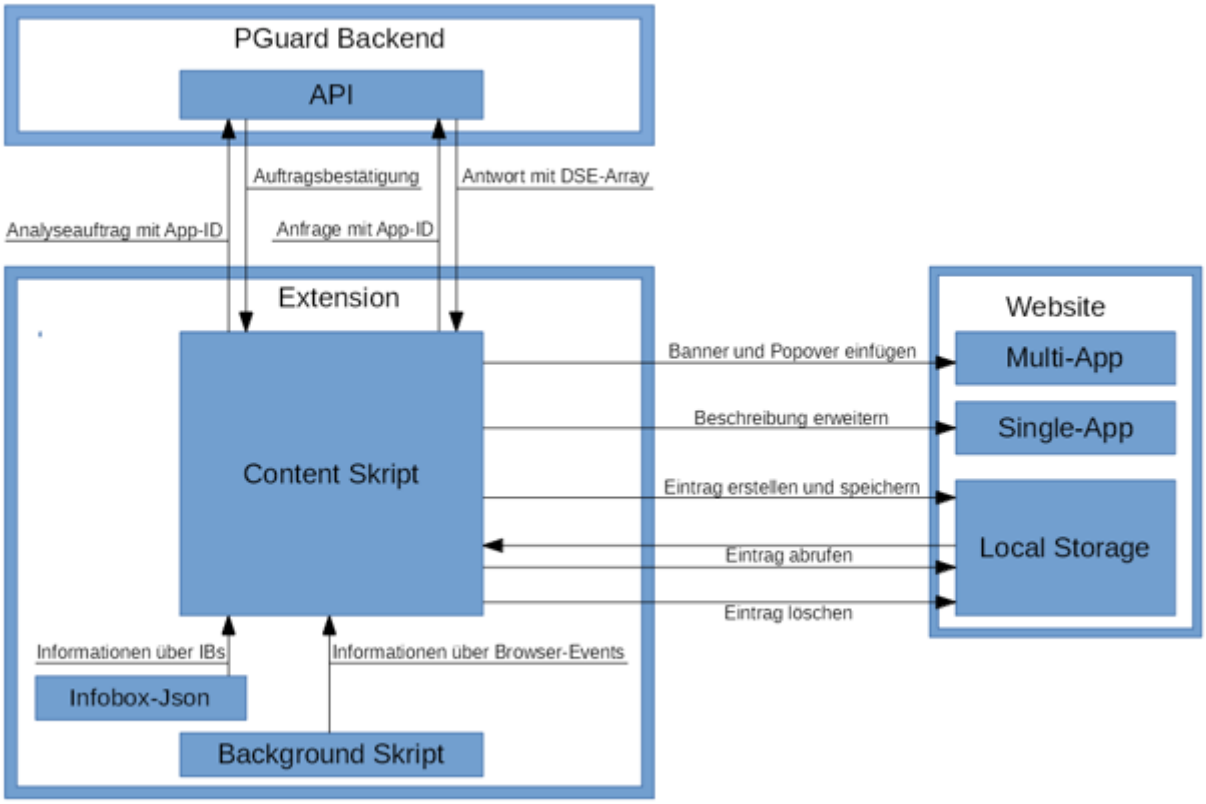
/F50/ Empfehlung bei Suchanfragen: Basierend auf den Bedenken einer Applikation kann der Nutzer die Suchanfrage so anpassen, dass ihm unbedenkliche Applikationen priorisiert angezeigt werden.

3.2.2.2 Nichtfunktionale Anforderungen

Die Nutzu Aus der Definition der Aufgabenstellung ergeben sich folgende Nutzerszenarien:

-

3.2.3 Programmaufbau



3.2.4 Ergebnis

3.2.5 Diskussion

3.3 Aufgabe 2: Evaluierung von Caching Methoden einer Browser Extension

3.3.1 Anforderungen

Extension erweitert die Landingpage. Diese in Kategorien unterteilt. Kategorien werden bei jedem Besuch wieder aufgerufen SSpiele mit Vorregistrierungßtellt über längere Zeit gleiche Apps dar. New + Updated Gamesünd Top-Bewertung: Spielellieferen jedes Mal ähnliche Ergebnisse =¿ überlappende Information Ëmpfehlungen für dichünd "Das könnte dir Gefallenpassen sich vorherigen Suchen an und liefern daher auch redudante Ergebnisse Selbe App oft mehrmals in der Übersicht vertreten Konklusion: viel Redundanz. Ergänzende Information werden mehrfach benötigt

Verarbeitet Diese nur zu benutzerfreundlichen Format Informationen müssen von externen Quelle entnommen werden Dadurch entstehen Anfragen an einen Server mit Antworten Antworten und Anfragen durch // redundant.

Thesen: Nutzung der Extension nach "Veröffentlichung"würde hohen Traffic verursachen mit vielen wiederholten Anfragen. Anfragen könnten ab einer bestimmten Nutzerzahl Server überlasten Performance der Extension leidet unter dieser Art der Infortionsbeschaffung Bei Ausfall der Quelle, bietet Extension keinen Mehrwert für den Nutzer mehr.

Lösung: Einrichtung von unabhängigen Speicher zur Aufbewahrung der gewonnen Informationen. Insbesondere viele/wiederholt genutzt Informationen sollen ohne erneute Anfrage zur Verfügung stehen. Neue Anfragen nur bei Veraltung der Informationen bzw. nur von neu aufgetauchten Apps Aufbau einer Struktur zur Abspeicherung der wichtigen Informationen

Verwendung des Speichers:

Welche Informationen stehen pro App zur Verfügung?(1) Welche Informationen werden pro App benötigt? (2) Wie wird der Speicher gepflegt?(3)

(1)

Wird bei Aufbau der App schon Beschrieben? Aynchron!

(2) Alter der Information: Ist die Information auf dem Aktuellen Stand wie die der Quelle? Ist die Information der Quelle veraltet? Wie oft wird so eine Information erneuert? (Neue DSE o.ä) =¿ Abspeichern des Analysedatums. Regelmäßige Überprüfungen(3 Tage), ob neue Information bei der Quelle vorhanden

Aufruf der App: Wie oft wird diese App aufgerufen? Informationen über die App im Speicher können nach gewisser Zeit gelöscht werden, wenn sie nicht erneut aufgerufen wurde. =¿ Frequency Count: Zähler im Speicher der bei jedem Aufruf erhöht wird. Regelmäßig wird der Speicher nach niedrigen Zählern durchsucht und diese Einträge gelöscht.

Informationen zur App: Inhalt entspricht Kennnummern der Infoboxen. Also Abspeichern der jeweilig vorhandenen Eigenschaft/Infobox

(3) LRU Konzept oder Fehler falls Speicher voll. Aufbau des Speichers von Speichermethode abhängig (Chrome =¿ key, value String) Möglichst kurze Zusammenfassung der benötigten Informationen: Alter der Information als Tageszähler seit festem Datum, da Tagesvergleich stattfindet Frequency Counter als einzelner Integer (ÜB: casten auf einstellige Zahl ProtectFaktor?) Abfolge von Kennnummer mit Infoboxen Trennzeichen zum korrekten Auslesen der Informationen. ÄLTERTAGE(TRENNZEICHEN)FREQ(TRENNZEICHEN)IB(TRENNZEICHEN)IB ...”Bsp: ”17500—3—1—2—3—4”

FREQ FÄLLT WEG???

Worst Case: 99999—1—2—3—4—5—6—7—8—9—10—11—12—13—14—15—16—17—18—19—20—21—22—23—24—25—27—
86 Character 86Bytes

Limit = 5MB = 5242880 Bytes

5242880 / 86 = 60963 Einträge

Appanfragen pro Initialladen 70

Nach Antwort von Server werden Informationen im Speicher abgelegt mit Startcounter. Bei jedem erneuten Abrufen wird der Speicher abgefragt. Falls Information vorhanden und aktuell wird Counter angepasst und Informationen lokal ausgelesen. Ansonsten neue Anfrage an Server. Ist Antwort mit neuer Information vorhanden wird die alte gelöscht und durch die neue ersetzt.

Extension prüft in regelmäßigen Abständen den Counter-Status. Ist dieser zu niedrig wird die Information aus dem Speicher entfernt.

Speicher wird auf Maximalkapazität geprüft und eine Approximation vom FFüllstand”wird errechnet. Ist der Speicher voll, werden Einträge mit dem niedrigsten Counter gelöscht.

3.3.2 Vorauswahl

Welche Arten von Speicher stehen einer Extension für Browser (hier Chrome) zur Verfügung?

Integriert: Session Storage und lokal Storage Session Storage: Daten bleiben nur während der Sitzung erhalten. Das Schließen des Browsers bzw. das Öffnen der Website in einem anderen Tab/Browserfenster beendet die Sitzung

Local Storage: Daten bleiben über eine Sitzung hinaus erhalten und verfallen erst durch Überschreiben oder Löschen

Kapazität: 5MB

Aufbau: String-Tupel nach dem (Key, Value)-Prinzip

Zugriff: getItem(key), setItem(key,value) und removeItem(key)

Serverseitiger Speicher: Identifizierung notwendig. Aufwendig in Pflege und Wartung. Kein Mehrwert zu Anfragen an Informationsquelle

Serverseitiger Speicher fällt vor vorne herein weg aus oben genannten Grund und datenschutzrechtlichen Bedenken.

Datenbanken: Verfügbar: IndexedDB und WebSQL

WebSQL seit November 2010 von W3C nicht mehr empfohlen (veraltet).

IndexedDB: API in allen modernen Browser zur Speicherung von Daten und Dateien in einer object-orientierten Datenbank. synchron und asynchron möglich. Funktioniert nach key, value prinzip Alle Datentypen von JavaScript werden unterstützt. Kann indexiert werden um Suchen effizient zu machen. Verwendet Prinzip von Transaktionen Anfragen mit Rückgabewerten als Basis aller Operationen Verfolgt den NoSQL-Ansatz

Speicherlimit nach global Limit (1/2 Festplatte) und Gruppenlimit (1/5 von global Limit, min 10MB max. 2GB) Gruppenlimit voll = voll (Fehler) Global Limit voll = löschen bis wieder frei (Quellenabhängig komplette Elemente gelöscht)

Warum nicht IndexedDB?

Vorteile von IndexedDB: Abspeicherung von großen strukturierten Datenmengen. Nachteile: hoher Aufwand bei Implementierung.Overhead lohnt nicht bei kleinen Datenmengen. Transaktionen blockieren bei Fehlern eventuell den Datenabruf bzw. die Aktualisierung

Storage API von Chrome ausreichend Speicher und geringer aufwand bei der Implementierung. Lediglich Strings benötigt. Indices bei gewählen value-Struktur nicht notwendig.

Vorteil von Session Storage: Speicherpflege nicht notwendig, da 5MB groß genug für Anzahl(?) an App-Informationen während einer Session im PlayStore. Informationen immer auf Stand der Quelle Nachteil: Bei erstmaligen Öffnen des Stores in neuer Browsersession werden viele Anfragen losgeschickt für Apps die bereits in der letzten Session schon angefragt wurden. Bei Serverausfällen fehlen

die Informationen Lediglich in einer Session mehrfach aufgerufene Apps ersparen erneute Anfragen. =¿ Speicherpflege fällt weg, dafür kaum Mehrwert bei Anfragen.

Vorteil von Lokal Storage: Apps werden einmal abgefragt und sind anschließend abgespeichert. Fällt der Server aus können die lokalen Informationen genutzt werden. Daten auch aus letzter Session bleiben vorhanden. Neue Anfragen werden nur dann geschickt wenn aktuelle Daten über 3 Tage alt sind. Nachteile: Speicherpflege notwendig. Dadurch wird die Information länger (Counter und Tag). Zusätzliche Rechenzeit für das Löschen von alten Informationen notwendig. Dadurch wird sichergestellt dass die 5MB nicht überschritten werden und somit Informationen ungewollt verloren gehen. Für Informationen mit hohem Counter muss regelmäßig überprüft werden, ob die Information noch aktuell ist, weil diese in der Regel lange im Speicher verweilt. =¿ Hohe Einsparung bei Anfragen an den Server möglich. Dafür müssen zusätzliche Operationen zur Speicherpflege und Prüfung der Informationen ausgeführt werden.

3.3.3 Rahmenbedingungen

Plattform: Windows 10 Rechner Build, Specs Chrome Details App Details Was wird gemessen? Limitierungen

Getestet auf:

Windows 10 Education 64 Bit Build 10.0.17134 Prozessor i7-6700K RAM: 16GB GPU: Nvidia GTX 1070

Chrome 67.0.3396.99 64 Bit

3.3.4 Vorgehensweise

3.3.5 Ergebnisse

Storage: none Ladezeit: 1435ms Start der Extension-Funktionsaufrufe: 944ms Dauer: 491ms Anzahl der Anfragen an das Backend: 0

Storage: Local Storage Ladezeit: 1711ms Start der Extension-Funktionsaufrufe: 892ms Dauer: 819ms Anzahl der Anfragen an das Backend: 125

Storage: none Ladezeit: 1761ms Start der Extension-Funktionsaufrufe: 883ms Dauer: 878ms Anzahl der Anfragen an das Backend: 131

3.3.6 Diskussion

Kapitel 4

Abschließende Diskussion

4.1 Konklusion

4.2 Fortsetzung der Forschung

Kapitel 5

Appendix

5.1 Derivations

5.1.1 Example Matlab Code

Add a sourcefile directly into LaTeX

```
% simple Kalman Filter example:
% state "x" consists of position and velocity
% system model "F" is a cinematic model of constant velocity
% only the position is measured

clear     % clear all matlab variables

%%% declare matlab variables and asign default (randomly chosen) value
% simulation specifications
T = 1;              % make a measurement every T steps. also called \Delta t
                    % i.e. every 1, 2, 3, ... seconds
real_x = [0; 10];   % "real world" state, only needed in simulation context
                    % also called ground truth. start: position=0, velocity=10

% model specifications
model_F = [1, T ;   % the model we have about the real world
           0, 1];   % here: cinematic model of constant velocity
q = 9;              % controls the amount of process noise. is usually unknown
model_Q = [T^4/4, T^3/2;       % process noise
           T^3/2, T^2 ]*q;     % arises from the cinematic model

% estimations specifications
esti_x = [0; 10];   % estimated state: position and velocity
esti_P = [1, 0 ;    % estimated covariance of esti_x. refelcts
          0, 2];    % the uncertainty about the estimated state esti_x

esti_z = 0;      % estimated measured value. here: just depicting position-entry
                 % of esti_x since we are only iterested in the position. Or
                 % maybe it is only possible to measure position, but not velocity
esti_S = [0,0 ; % estimated covariance of esti_z. Will consist of process noise
          0,0]; % with added measurement noise

H = [1, 0;       % observation matrix. we only measure position values
     0, 0];      % this row could be left out, but then also modify R to 1x1
R = [1, 0 ;      % measurement noise. is usually unknown. reflects the
     0, 1];      % inaccuracy of the sensors
K = [0, 0];      % Kalman gain vector


%%% Initialization
esti_x = [0; 10];
esti_P = [1, 0 ;
          0, 2];

for step = 1:1000          % simulate for 1000 steps (simulate continuous time)
  if mod(step, T) == 0     % if it is time to take a new measurement
    % update the "real data". For simplicity: take the model F. But could be any
    % other function, possibly non-linear.
    % mvnrnd = multi variate normal random numbers
    real_x = model_F * real_x + transpose(mvnrnd([0,0], model_Q));

    %%% Step 1: Prediction Step
    esti_x = model_F * esti_x;  % estimate the new state according to the
                                % system model since we do not have any
                                % control inputs, this term is left out
    esti_P = model_F * esti_P * transpose(model_F) + model_Q;  % update the
                                % covariance of estimated state esti_x
    esti_z = H * esti_x;        % depic position value from estimated state
    esti_S = H * esti_P * transpose(H) + R;  % estimation of the covariance of
                                % the estimated measured value. inherits model
                                % noise and measurement noise

    %%% make a measurement z
    z = H * real_x + transpose(mvnrnd([0,0], R)); % make a noisy measurement

    % Step 2: Innovation Step
    K = esti_P * transpose(H) * esti_S^-1;  % calculate Kalman gain vector by
                                            % comparing model and measurement
                                            % noise
    esti_x = esti_x + K * (z - esti_z);     % update the estimated state by an
                                            % weighted sum of the measurement
                                            % and the model-estimation
    esti_P = esti_P - K * esti_S * transpose(K);  % update covariance of
                                                  % estimated state
  end
end
```

Listing 5.1: Simple example of a Kalman Filter in Matlab

Acknowledgement

First of all, I would like to express my gratitude to … for the aspiring guidance, useful comments and invaluably support throughout the whole process of this Master Thesis. Furthermore, I would like to thanks …, … and the other members of the research team for helpful discussions and constructive criticism. In addition, I would like to thank my University supervisor … for all the helpful remarks, advises and discussions.

Also, I would like to thank my parents and … who have supported me throughout the entire process by keeping me harmonious and motivated.

Last but not least, I like to thank … for funding my research and providing me with the facilities being required.

Abbildungsverzeichnis

Literaturverzeichnis

Proclamation

Hereby I confirm that I wrote this thesis independently and that I have not made use of any other resources or means than those indicated.

Forname Surname, Place, 27. Oktober 2018