Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Primož Črnigoj

Sistem za vodenje viličarjev v skladišču

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: akad. prof. dr. Ivan Bratko

Ljubljana 2014

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00105 / 2013
Datum: 11.4.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **PRIMOŽ ČRNIGOJ**

Naslov: **SISTEM ZA VODENJE VILIČARJEV V SKLADIŠČU**
A SYSTEM FOR MANAGING FORKLIFTS IN A WAREHOUSE

Vrsta naloge: Diplomsko delo univerzitetnega študija prve stopnje

Tematika naloge:

Optimizirano avtomatsko usmerjanje viličarjev v skladišču je ključnega pomena za izkoriščenost viličarjev ter učinkovito delovanje skladišča. V tej diplomski nalogi razvijte sistem, ki za dano nalogo zbiranja blaga viličaristu prikaže čim boljšo pot za izvedbo te naloge, pri čemer sistem minimizira dolžino poti in dolžino t.i. prazne vožnje. Sistem naj določa položaj viličarja avtomatsko s kamero in s pomočjo oznak na tleh.

Mentor:
akad. prof. dr. Ivan Bratko

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:
prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Primož Črnigoj, z vpisno številko **63090027**, sem avtor diplomskega dela z naslovom:

Sistem za vodenje viličarjev v skladišču

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom akad. prof. dr. Ivana Bratka,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 14. marca 2014 Podpis avtorja:

Na tem mestu bi se rad zahvalil mentorju akad. prof. dr. Ivanu Bratku za vso pomoč in usmeritev pri izdelavi diplomskega dela. Zahvalil bi se tudi staršem, ki so mi omogočili študij, ter podjetju Epilog d.o.o., ki mi je nudilo opremo za izdelavo diplomskega dela.

Kazalo

Seznam uporabljenih kratic

RTLS - Real Time Locationing System
SOAP - Simple Object Access Protocol
URL - Uniform Resource Locator
USB - Universal Serial Bus
XML - Extensible Markup Language
JVM - Java Virtual Machine
JAR - Java Archive
HTML - Hypertext Markup Language

Povzetek

Cilj diplomske naloge je vizualizacija in optimizacija poti viličarjev. Z uporabo sistema Pilot Pro smo razvili sistem za vodenje viličarjev. Sistem s pomočjo kamere in talnih označb izračunava lokacijo viličarja v skladišču. S tem je omogočeno natačno usmerjanje viličarista in sledenje zaloge v skladišču, ne da bi jo morali označevati. Rezultat diplomske naloge je spletna aplikacija, ki viličaristu prikazuje trenutni nalog. Prikaže mu najkrajšo pot za izvedbo ter projicira paleto na končno mesto. Sistem naloge viličaristom razporeja tako, da je čim manj tako imenovanih praznih voženj. To so vožnje, pri katerih viličar ne prevaža tovora. S tem dosežemo, da so viličarji bolje izkoriščeni. Viličaristom tudi ni potrebno iskanje določene zaloge po skladišču, saj ga sistem vedno vodi do ciljne lokacije. Tako prihranimo pri času in gorivu s čimer zmanjšamo obratovalne stroške skladišča.

Ključne besede: vizualizacija, skladišče, spletna aplikacija, optimizacija, večagentno iskanje poti, vodenje, viličar

Abstract

The goal of this thesis is the visualisation and optimization of forklift routes. With Pilot Pro system, we have developed a forklift guidance system. The system uses a camera and floor markers to accurately calculate the position of the forklift in the warehouse. This allows accurate guidance for the driver and palette tracing, without the need to label them. The result of the thesis is a web application that displays the current tasks to the forklift driver. It displays the shortest path to the realization of the driver's task and projects the goal position of the palette. The system allocates tasks to the forklift drivers in a way that empty runs are minimized. These are the runs in which the forklift does not carry any load. In that way forklifts are used efficiently. The forklift drivers also do not need to search for stock, because the system always leads them to the target location. This saves time and fuel in order to reduce the operating costs of the warehouse.

Key words: visualisation, warehouse, web application, optimisation, multi-agent a*, forklift guidance system

Poglavje 1

Uvod

Z razvojem vse hitrejših procesorjev se odpirajo nove in nove možnosti uporabe računalniških tehnologij. Posebej zanimivo je tudi področje računalniškega vida in z njim povezane rešitve. V podjetju Zenotrack so zasnovali sistem, ki s pomočje kamere na viličarju izračunava položaj in smer viličarja. S pomočjo njihovega sistema pa smo v podjetju Epilog d.o.o. naredili sistem za vodenje viličarjev. Glavna prednost sistema je sledenje paletam skozi skladišče, brez označevanja le teh. Sistem omogoča viličaristu, da brez interakcije s sistemom prepelje določeno paleto. Glede na podatke o lokaciji viličarja pa sistem sam opravi potrebno knjiženje. Sistem smo dopolnili tudi z večagentim iskanjem poti. To nam omogoča, da viličaristom naloge razdelimo tako, da je vsota razdalj prepepljanih poti najmanjša. To pomeni manj praznih voženj skozi skladišče in večji izkoristek viličarjev.

Poglavje 2

Pregled podobnih sistemov

Na trgu je že kar nekaj podobnih sistemov. V glavnem jih lahko ločimo na dve skupini. Sistemi z samodejnim knjiženjem in sistemi z ročnim knjiženjem.

2.1 Sistemi z ročnim knjiženjem

Pri sistemih z ročnim knjiženjem, mora viličarist ročno vnesti opravljene naloge v sistem. Tako mora izbrati paleto, ki jo je vzel in lokacijo, kamor je paleto odložil. Prednost ročnih sistemov je cena, saj so nekjakrat cenejši od sistemov s samodejnim knjiženjem. Pomankljivosti ročnih sistemov pa so daljše uvajanje viličarista za uporabo sistema, težje sklepamo, kje se viličarji največ vozijo, vnašanje podatkov je zamudno in prihaja do napak pri vnosu.

2.2 Sistemi z samodejnim knjiženjem

Sistemi z samodejnim knjiženjem na podlagi lokacije viličarja sami prepoznajo, katero paleto je natovoril in kje jo je odložil. Viličaristi tako ne potrebujejo ničesar vnašati v sam sistem, kar poveča njihovo efektivnost in zmanjša možnost napak pri vnosu. Sistem zahteva vnos uporabnika le v izjemnih primerih. Šolanje uporabnikov za uporabo takšnega sistema je mnogo krajše, saj jih sistem sam vodi po nalogah, ki jih morajo opraviti. Omogoča

tudi navigacijo skozi skladišče, saj v vsakem trenutku ve, kje se nahaja viličar in kje je ciljna lokacija. Na podlagi lokacije viličarja lahko sistem tudi optimalnejše razporeja opravila med viličaristi. Viličaristu lahko izpisujemo tudi obvestila, da se vozi v območju z omejeno hitrostjo, da je v bližini še en viličar in obstaja nevarnost trčenja in podobno.

Poglavje 3

Uporabljene tehnologije

Sistem za vodenje viličarjev sestavljajo mnoge tehnologije. V nadaljevanju bomo opisali delovanje glavnih gradnikov sistema in njihove funkcionalnosti.

3.1 Zenotrack Pilot Pro

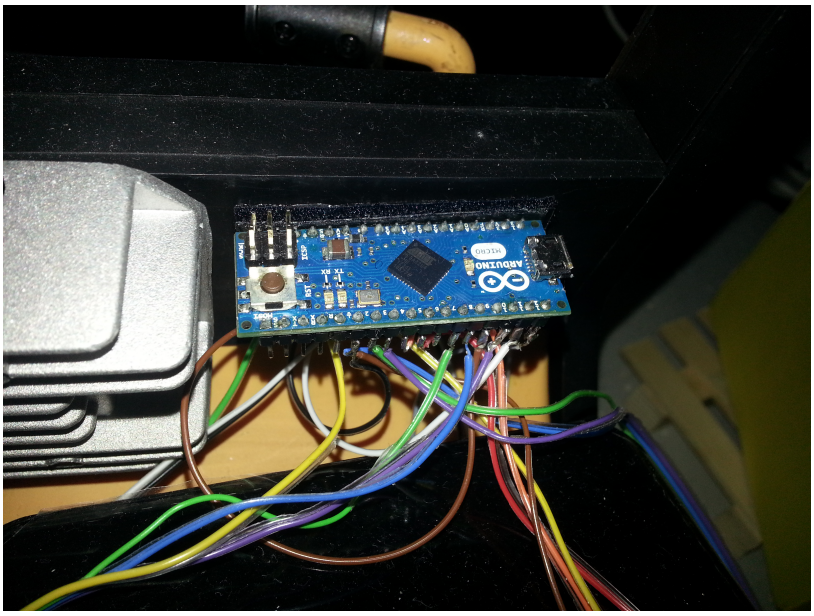
Zenotrack Pilot Pro je RTLS sistem. Sistemu za vodenje viličarjev pošilja podatke o hitrosti in položaju viličarja.
Za delovanje potrebuje kamero 3.1, senzor za tovor, računalnik na viličarju ter markerje na tleh. Položaj vsakega markerja je potrebno natačno izmeriti in njegove koordinate vnesti v Pilot Pro. Ob zagonu sistema je potrebno viličar pripeljati do markerja tako, da je le ta v vidnem prostoru kamere. Pilot Pro na sliki zazna marker in njegovo orientacijo. Na podlagi vnešenih koordinat markerja Pilot Pro izračuna položaj in orientacijo viličarja. Senzor za tovor je v produkcijskem sistemu ponavadi laserski senzor, ki meri razdaljo do najbližje točke pred viličarjem. Pilot Pro na podlagi izmerjene razdalje sklepa, ali je na vilicah viličarja naložen tovor. Pilot Pro tako na določen časovni interval pošilja podatke o tovoru, položaju, hitrosti viličarja po protokolu SOAP na konfiguriran URL. Za potrebe demonstracije, se namesto laserskega merilnika razdalje uporablja cenejši ultrazvočni senzor. Ostala oprema je identična na demonstracijskemu in produkcijskemu sistemu.

Slika 3.1: Kamera na viličarju

3.2 Arduino

Arduino sistemu za vodenje viličarjev pošilja podatke o višini vilic, stanju baterije in razdaljo do najbližje točke pred viličarjem. Iz nje lahko sklepamo, če je na vilicah naložen tovor.

Zaradi visoke cene laserskih merilnikov razdalje, smo se pri izdelavi makete viličarja odločili za uporabo cenejše senzorike. Izbrali smo odprtokodno elektrotehnično platformo Arduino. Mikrokontroler Arduino micro 3.2 omogoča enostaven priklop različnih senzorjev. Na uradni spletni strani projekta je na voljo tudi Arduino software. To je okolje, v katerem pišemo programe za mikrokontroler za Arduino micro. Okolje omogoča tudi enostavno nalaganje spisanih programov na Arduino micro preko micro USB-ja. V naši izvedbi Arduino uporablja ultrazvočni senzor za zaznavo razdalje do najbližje točke pred viličarjem. Za merjenje višine vilic je bil s pomočjo 3D tisklanika izdelan zobnik, ki je pritrjen na zgornji del vilic tako, da je v stiku z zobato letvijo na vilicah. Obrati zobnika so nato pretvorjeni v višino vilic



Slika 3.2: Arduino sistemu pošilja podatke o višini vilic in naloženem tovoru v milimetrih.

3.3 SOAP

Po protokolu SOAP se izvaja komunikacija med sistemom za vodenje viličarjev in sistemom Pilot Pro.
SOAP ali Simple Object Access Protocol je protokol namenjen izmenjavi strukturiranih informacij v decentraliziranih, distributiranih okoljih. Uporablja tehnologijo XML za definicijo razširljivega sporočilnega ogrodja. Ogrodje zasnovano tako, da je neodvisno od programskega jezika v katerem je implementirano. S tem je omogočeno, da se med sistemi v različnih programskih jezikih in na različnih platformah izmenjujejo podatki.

3.4 Java

Java je programski jezik in računalniška platforma. V programskem jeziku java smo napisali strežniško aplikacijo za interakcijo s sistemom za vodenje viličarjev. Ta aplikacija bere podatke iz podatkovne baze in izvaja glavno logiko. Programski jezik Java smo izbrali predvsem zaradi poznavanja jezika v podjetju. Dobra lastnost je tudi prenosljivost aplikacije. Isto Java aplikacijo lahko namreč poganjamo na različnih sistemih, če ti le lahko poganjajo JVM.

3.5 PostgreSQL

PostgreSQL je odprtokodna relacijska podatkovna baza. Uporabljamo jo za trajno shranjevanje podatkov in za komunikacijo med sistemi. Izbrali smo jo, ker je brezplačna in ker so na voljo tudi brezplačna orodja za administracijo. Pri izdelavi demonstracijskega sitema smo uporabili pgAdmin.

3.6 Playframework

Playframework je programsko ogrodje za izdelavo javanskih spletnih aplikacij. Omogoča samodejno generiranje relacij v podatkovni bazi, na podlagi javanskih razredov. Med razvojem aplikacije te ni potrebno ustavljati in ponovno zaganjati. Ob vsaki spremembi kode je potrebno le osvežiti spletno stran v brskalniku. Vsebuje tudi strežnik Netty. Inštalacija aplikacije narejene v playframeworku je enostavna. V konzoli poženemo ukaz play dist in playframework izdela JAR z vsemi potrebnimi datotekami. Aplikacijo prenesemo na strežnik, ki ima instaliran JVM in jo poženemo z ukazom java jar ime aplikacije. Glavna prednost spletne aplikacije je, da jo je potrebno inštalirati le na strežniku. Odjemalci se nato preko spletnih brskalnikov povežejo na strežnik, kjer lahko dostopajo do aplikacije.

3.7 Websocket

S podporo standarda HTML5 brskalniki omogočajo full-duplex komunikacijo z aplikacijo na strežniku. Tako aplikaciji na odjemalcu ni več potrebno periodično spraševati strežniške aplikacije po novih podatkih. Strežniška aplikacija lahko s pomočjo protokola websocket sama pošlje podatke aplikaciji, ki teče v brskalniku pri odjemalcu, ko so ti pripravljeni. V aplikaciji na odjemalcu pa se nato sproži dogodek. Nato lahko aplikacija na odjemalcu primerno obdela prejete podatke. Tehnologija websocket je še posebej primerna za aplikacije, ki potrebujejo podatke v realnem času. Takšen je tudi naš sistem za vodenje viličarjev. Preko websocket-a aplikacija na strežniku ob vsaki spremembi v bazi pošilja nove podatke aplikaciji na odjemalcu.

3.8 Threejs

Threejs je odprtokodna javascript knjižnjica za prikazovanje vsebin v 3D. Uporablja WebGL, SVG ali html element platno. V našem primeru smo se odločili za uporabo WebGL, saj je ta možnost najihtrejša in najprimernejša za prikazovanje zapletenejših 3D modelov, kot so viličarji. Threejs omogoča nalaganje 3D modelov v brskalnik. Z njimi lahko nato poljubno manipuliramo v programskem jeziku javascript.

3.9 Bootstrap

Bootstrap je stilska predloga za izdelavo uporabniških vmesnikov v HTML. Izdelali so jo v podjetju Twitter in je na voljo brezplačno. Razvijalcu spletne aplikacije ponuja osnovne html gradnike in ikone. Na svetovnem spletu je na voljo tudi veliko kompleksnejših gradnikov za uporabo z Bootstrap-om. Programerjem, ki niso vešči oblikovanja vsebin takšna stilska predloga močno olajša delo in pripomore, da so uporabniki zadovoljni z izgledom končne aplikacije.

Poglavje 4

Implementacija

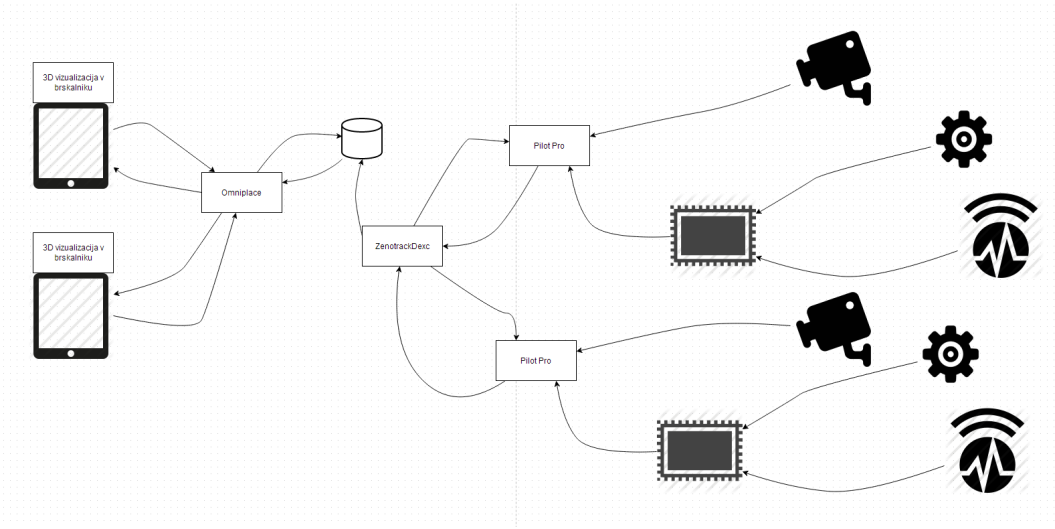
V nadaljevanju je podrobno opisano delovanje vseh delov sistema za vodenje viličarjev in njihova medsebojna povezava. Vsi sestavni deli sistema so prikazani na skici 4.1.

4.1 Podatkovna shema

Entiteta CMO predstavlja naloge za premik palet. V njej sta zapisani začetna in končna lokacija palete. Vsak nalog ima vsaj dva pick mission-a. Entiteta Pick Mission predstavlja vsako knjiženje palete na viličar ali iz njega. S tem imamo v sistemu celoten dnevnik premikov in prelaganj palet, tudi če zato ni bilo naloga. Entiteta Cont predstavlja palete in druge oblike zaloge v skladišču. Hrani podatke o logični in fizični lokaciji v skladišču. Entiteta MfsDevice predstavlja viličarje. Hrani podatke o tem, kako je viličar natovoril zadnjo paleto, kje se nahaja in vrednosti vseh njegovih senzorjev.

4.2 Pošiljanje prebranih vrednosti senzorja v Pilot Pro

Za potrebe demonstracije sistema smo razvili lastno senzoriko s pomočjo Arduin-a. Na mikrokontroler Arduino je povezan senzor za višino vilic in



Slika 4.1: Skica gradnikov sistema

ultrazvočni senzor za razdaljo do najbližje palete pred vilicami. Na 125 milisekund Arduino prebere vrednosti senzorjev in jih po serijski povezavi pošlje sistemu Pilot Pro.

4.3 Branje podatkov iz Pilot Pro

Pilot Pro nato podatke senzorjev ter podatke o lokaciji, ki jo izračuna iz slike kamere, pošlje po protokolu SOAP v modul zenotrackDexc. To je modul sistema za vodenje viličarjev, ki zna prebrati SOAP sporočila. Vsebino SOAP sporočila nato zapiše v podatkovno bazo ompl.

4.4 Obdelava podatkov

Glavni del sistema za vodenje viličarjev vsake pol sekunde prebere podatke iz podatkovne baze. Nato preveri, če je senzor za tovor zaznal paleto na vilicah. To pomeni, če je razdalja do najbližje točke pred vilicami manjša od 3cm. Če je senzor zaznal paleto, potem sistem v podatkovni bazi preveri katera

je najbližja paleta poleg viličarja in jo preknjiži iz lokacije na viličar. Če v krogu 5cm ni nobene palete, potem sistem viličaristu odpre dialog Neznana paleta. Viličarist nato vnese šifro palete v sistem in ustvari se nova paleta.

4.5 Iskanje poti

4.5.1 A*

Algoritem A* je med najbolj priljubljenimi algoritmi za iskanje poti. Pričakujemo, da bo v skladiščih dobro deloval, saj imamo veliko direktnih poti do vseh lokacij. A* deluje tako, da s pomočjo hevristike vedno raziskuje najbolj obetavno pot. Pri tem upošteva tudi že prepotovano pot. S tem se izognemo ciklom.

4.5.2 Večagentni A*

V našem sistemu nas zanima skladišče kot celota in ne le posamezen viličar. Želimo povečati skupno efektivnost viličarjev. To pomeni, da moramo iskati minimalno vsoto poti vseh viličarjev, za izpolnitev določenega naloga. V ta namen smo implementirali večagentni A*. V primeru enega viličarja ima A* na voljo 4 premike in sicer naprej, nazaj, levo, desno. V primeru da preiskujemo dva viličarja hkrati, pa imamo na voljo 16 premikov in sicer prvi naprej in drugi naprej, prvi naprej in drugi nazaj in tako naprej.

4.5.3 Implementacija več agentnega A*

Večagentni A* smo implementirali v programskem jeziku Scala. Rešitev kličemo s parametri, kot so seznam viličarjev, sezam nalogov, ter seznam ovir. Algoritem iz vsakega stanja razvije n^4 novih stanj, pri čemer je n število viličarjev. Hevristika je vsota obteženega števila preostalih naročil, obteženege dolžine trenutno prevožene poti ter obteženege vsote razdalj vseh viličarjev do njihovega najbližjega naloga. V rezultatu dobimo podan seznam parov viličar - nalog iz katerega razberemo kateri viličar naj prevzame kateri nalog.

4.6 Zlagalna strategija

Sistem trenutno omogoča dve strategiji zlaganja palet. Prvo smo poimenovali zlaganje spodaj levo. Strategija deluje tako, da paleto najprej postavi v desni zgornji kot lokacije. Nato jo pomika levo toliko časa, dokler se ne prekriva z drugo paleto ali pa pade izven lokacije. Potem začne paleto pomikati po Y osi navzdol, dokler se ne prekriva z drugo ali pa pade izven lokacije. Ta dva koraka se izmenjaje ponavljata, dokler se koordinate palete spreminjajo. Ko se ne spreminjajo več, smo prišli do končne lokacije. Druga strategija zlaganje spodaj desno deluje na enak način, le da sta zamenjani strani levo in desno.

Poglavje 5

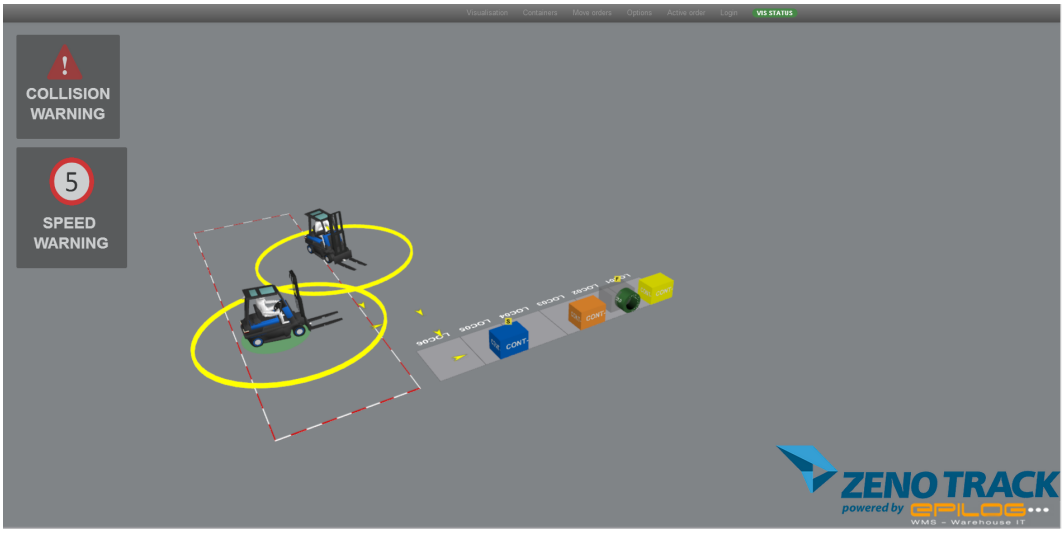
Uporaba in funkcionalnosti

5.1 Inštalacija

Za poganjanje sistema za vodenje viličarjev potrebujemo javo in postgresql. Najprej moramo ustvariti podatkovno bazo z imenom Ompl. Nato na računalnik prenesemo mapo Omniplace. Ta vsebuje JAR Omniplace, ki ga poženemo z ukazom java -jar omniplace.jar. Ob zagonu sistem samodejno ustvari primerno shemo na podatkovni bazi Ompl. S tem je sistem pripravljen za uporabo.

5.2 Nevarnost trčenja

Sistem vsak trenutek pozna lokacije, smeri in hitrosti vseh viličarjev v skladišču. S temi podatki lahko viličarista opozori, da obstaja nevarnost trčenja. Najmanjšo razdaljo med viličarjema lahko konfiguriramo v dialogu Options. Z drsnikom Collision perimeter nastavimo željeno razdaljo pri kateri naj se sproži opozorilo.



Slika 5.1: Opozorilo o omejeni hitrosti, ter o možnosti trčenja.

5.3 Omejitev hitrosti

V mapi seedData najdemo datoteko speed locations.csv. V njej so zapisane koordinate območij z omejitvijo hitrosti. To so lahko nakladalna območja, servisna območja in podobno. Ko se viličarist zapelje v določeno območje, ga sistem opozori 5.1, da je hitrost omejena. Ob spremembi te datoteke je potrebno v vizualizaciji pritisniti Reset DB. To povzroči brisanje zapisov iz baze in ponovno branje konfiguracijskih datotek.

5.4 Nalog za premik palete

Ob kliku na gumb CMO se viličaristu odpre dialog z trenutnim nalogom 5.2 za premik palete. Na njem sta napisani izvorna in ciljna lokacija določene palete. V sami vizualizaciji je paleta za premik označena z veliko črko S, na končni lokaciji pa se izriše projekcija palete in na njej velika črka F. V sistem lahko vnašamo naročila za premik palete preko konfiguracijske datoteke scenario. V datoteko v zapišemo, katero paleto želimo premakniti na katero mesto. Sistem nato po vrsti jemlje naročila in jih prikazuje viličaristu. Na-

Slika 5.2: V vizualizaciji je prikazana pot, ter projekcija, kje mora biti odložena paleta. Viličaristu se jasno izpiše, katero paleto mora natovoriti in kje jo raztovoriti. V kolikor viličarist paleto odloži na napačni poziciji, ga sistem opozori in mu ustvari nov nalog.

log za premike palet lahko dodamo tudi preko uporabniškega vmesnika. Ta možnost se nahaja pod opcijo Move orders. Odpre se nam seznam vseh nalogov z opisi in stanjem. Ob kliku na gumb new se nam odpre dialog v katerem izberemo željeno paleto in željeno lokacijo. Ob kliku na gumb save se ustvari nov nalog in se postavi v vrsto za izvajanje. Preden se nalog začne izvajati, sistem preveri, če je paleta že na željeni lokaciji. Če ni, pred začetkom izvajanja izračuna točne koordinate na lokaciji, kjer mora biti paleta po določeni zlagalni strategiji odložena. Če je viličarist ne odloži na predvidenem mestu, potem ga sistem opozori in ustvari nov nalog za premik, ki se začne izvajati takoj.

5.5 Neznana paleta

V primeru, da senzorji zaznajo tovor, v bližini viličarja pa v sistemu ni palet, se viličaristu odpre dialog v katerega mora vnesti šifro natovorjene palete. Če je viličarist opravlja nalogo raztovarjanja se v sistemu samodejno ustvari nova paleta z novo šifro.

5.6 Urejanje palet

Sistem omogoča tudi ročno urejanje palet. Urejamo jih tako, da kliknemo na gumb Containers. Odpre se nam nova stran, na kateri je seznam vseh palet. S klikom na gumb New dodamo novo, s klikom na gumb Edit pa lahko urejamo obstoječe palete. Začetna konfiguracija palet je zapisana v datoteki cont.csv. Ob pritisku na gumb Reset DB se vse palete izbrišejo in znova se v sistem naloži začetna konfiguracija iz datoteke cont.csv.

Poglavje 6

Testiranje in evaluacija

Pri testiranju smo ugotovili, da je programsko ogrodje playframework zelo stabilno. Tudi sam sistem za vodenje viličarjev ni pokazal večjih pomanjkljivosti v delovanju. Izgube omrežnih povezav aplikacija brez težav prenese in jih znova vzpostavi, takoj, ko je to mogoče. Uspeli smo pridobiti tudi mnenje, sicer v angleščini, partnerjev iz podjetja Zenotrack za katere smo razvili sistem za vodenje viličarjev.

" The visualization created great impression at the LogiMat. As I told you during the fair, we focus with your visualization on the safety functionalities, i.e. collision warning and speed warning. They both worked without problems. Your software ran uninterrupted without crashing a single time, that was nice.

The only thing we had to do every day during preparations before the fair started was to set the proper camera view and the initial position of the simulation forklift (to coincide with the one placed on the mate). This did not take too long, nevertheless It would be an option to consider, to implement a functionality for saving a given camera position as well as initial position of the simulation forklift.

To be sincere, we did not encounter any other drawbacks regarding your visualization during the fair. Other points regarding further development are noted in the Google Doc. I will keep updating it as soon as we find any other

possible improvements.

One thing that was very comfortable is inherent to the concept of your visualization, which is browsed-based. During the fair, it allowed us to zoom in and out to adapt the content of the visualization regarding to our needs (screen size and resolution). This was very handy. Of course, nowadays everything browser-based causes good impression."

Testiranje je bilo torej uspešno. Našli so še nekaj pomankljivosti, kot je nezmožnost shranjevanja položaja kamere. Položaj se namreč ob vsakem zagonu aplikacije ponastavi. V mnenju je govora tudi o premikanju simulacijskega viličarja. Ta problem je bil kasneje rešen s podporo za prikazovanje več viličarjev. Sprva je bil sistem zmožen prikazovati le enega. Demonstracija sistema za vodenje viličarjev na sejmu je vidna na sliki 6.1.

Slika 6.1: Sejem Logimat 2014, Stuttgart, Nemčija

Poglavje 7

Možne izboljšave

V prihodnosti želimo animirano vožnjo in ne prestavljanja modelov. Dobro bi bilo, če bi sistem omogočal ustvarjanje nalogov za viličariste tako, da paleto kar z miško premaknemo na vizualizaciji. Pri strankah je zaželjen prikaz pogostosti voženj skozi določeno pot v skladišču, prikaz vsebine palete ob kliku nanjo in posnetek voženj viličarjev. S tem bi lahko upravitelj skladišča dobro in hitro videl, kje in kako se vozi njegova flota viličarjev. Dobro bi bilo implementirati tudi iskanje poti z sistemom za izogibanje trčenja. Torej bi sistem dva viličarja poslal po dveh različnih poteh. Trenutno sistem le opozarja, da lahko pride do trčenja, viličarist pa mora potem sam izbrati drugo pot.

Poglavje 8

Sklepne ugotovitve

Dnevno se na trgu pojavljajo nove rešitve za pozicioniranje objektov v zaprtih prostorih. S tem se ustvarjajo priložnosti za izdelavo sistemov, kot jih do sedaj nismo poznali. Trenutne rešitve za pozicioniranje viličarjev stanejo do nekaj tisoč evrov na viličar. Že v času pisanja diplomske naloge pa se na trgu pojavljajo rešitve, katerih cena ne presega tisoč evrov. Z zamenjavo tehnologije za cenejšo, bo postal takšen sistem podajanja nalogov viličaristom in sledenja materialu zanimiv tudi za manjša skladišča z že tremi viličarji. Na tujih trgih je za takšne rešitve precej zanimanja.

Literatura

[1] Playframework (2014) Dostopno na:
http://www.playframework.com/

[2] Introduction to A* (2014) Dostopno na:
http://theory.stanford.edu/ amitp/GameProgramming/AStarComparison.html

[3] Integrated forklift guidance system for unitop warehouse (2014) Dostopno na:
http://www.gob.de/en/erp-software/warehouse/forklift-truck-guidance-system.html

[4] R. Nissim, R. Brafman (2013) Multi-Agent A* for Parallel and Distributed Systems Dostopno na:
http://www.cs.bgu.ac.il/ raznis/mafs.pdf