Human action and motion prediction in industrial human-robot shared environments using probabilistic decision-making methods

Petković, Tomislav

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Tomislav Petković

HUMAN ACTION AND MOTION PREDICTION IN INDUSTRIAL HUMAN-ROBOT SHARED ENVIRONMENTS USING PROBABILISTIC DECISION-MAKING METHODS

DOCTORAL THESIS

Zagreb, 2022

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Tomislav Petković

HUMAN ACTION AND MOTION PREDICTION IN INDUSTRIAL HUMAN-ROBOT SHARED ENVIRONMENTS USING PROBABILISTIC DECISION-MAKING METHODS

DOCTORAL THESIS

Supervisor: Associate Professor Ivan Marković, PhD

Zagreb, 2022

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Tomislav Petković

PREDVIĐANJE RADNJI I GIBANJA ČOVJEKA U SURADNIM INDUSTRIJSKIM PROSTORIMA LJUDI I ROBOTA ZASNOVANO NA VJEROJATNOSNIM METODAMA ODLUČIVANJA

DOKTORSKI RAD

Mentor: Izv. prof. dr. sc. Ivan Marković

Zagreb, 2022.

Doctoral thesis was written at the University of Zagreb, Faculty of Electrical Engineering and Computing, Departement of Control and Computer Engineering.

Supervisor: Associate Professor Ivan Marković, PhD

Thesis contains 119 pages

Thesis no.:

ABOUT THE SUPERVISOR

Ivan Marković was born in Osijek, Croatia in 1985. He finished a math-oriented high school in Osijek in 2003 and received the master and PhD degrees in Eectrical Engineering under the mentorship of Prof. Ivan Petrović from the University of Zagreb Faculty of Electrical Engineering and Computing (FER), Zagreb, Croatia in 2008 and 2014, respectively. In 2008 he was employed by S.C.A.N. Ltd. where he worked on developing control algorithms and supervision interfaces for terminal automation systems. In 2009 he joined FER and the Department of Control and Computer Engineering as a research assistant where he enrolled the PhD studies in the same year. In 2009 and 2012 he was a member of the National Organizing Committee and in 2022 the Programe Committee Chair of international scientific conferences. In 2009 he became the coordinator of the technical editing team of the journal "Automatika – Journal for Control, Measurement, Electronics, Computing and Communications" and since 2018 he serves as the deputy Editor-In-Chief. In collaboration with Siemens Croatia Inc. he has been a course trainer for programmes developed for industry participants and in 2018. he recieved the recognition for his contribution in advancing education for SIMATIC automation systems. He actively participated as a researcher or work package leader in 6 national and 7 international research projects. During 2013 and 2014 he was a visiting researcher in the laboratory led by Prof. Francois Chaumette at INRIA Rennes-Bretagne Atlantique u Rennesu, France. He published 23 journal papers and more than 40 conference papers in the fields of estimation theory, sensor fusion, and robot vision with applications to autonomous mobile robot and vehicle control. He is a member of the Institute of Electrical and Electronics Engineers (IEEE), Technical Committee for Robotics of the International Federation of Automatic Control (IFAC), and the Croatian Society for Communication, Computing, Electronics, Measurement and Control (KoREMA). In 2018 he received the young scientist award "Vera Johanides" of the Croatian Academy of Engineering, in 2014 silver plaque "Josip Lončar" of FER for an outstanding doctoral dissertation, in 2008 the "INETEC award" of the Institute of Nuclear Technology for academic excellence during the undergraduate and graduate studies, and in 2006 the "Josip Lončar" award of FER for student excellence in the respective academic year.

O MENTORU

Ivan Marković je rođen 1985. godine u Osijeku. Treću gimnaziju u Osijeku završio je 2003. godine te je diplomirao i doktorirao u polju elektrotehnike pod mentorstvom prof. dr. sc. Ivana Petrovića na Sveučilištu u Zagrebu Fakultetu elektrotehnike i računarstva (FER), 2008. odnosno 2014. godine. Po završetku studija zapošljava se u tvrtki S.C.A.N. d.o.o. gdje radi na razvijanju algoritama upravljanja i nadzora u sustavima automatizacije naftnih terminala. U 2009. godini zapošljava se u Zavodu za automatiku i računalno inženjerstvo FER-a u zvanju znanstvenog novaka u suradničkom zvanju asistenta te iste godine upisuje i doktorske studije. U 2009. i 2012. godini bio je član organizacijskog odbora, a u 2022. predsjednik programskog odbora međunarodnih znanstvenih konferencija. Od 2009. godine koordinator je tehničkog uredništva časopisa "Automatika – časopis za automatiku, mjerenje, elektroniku, računarstvo i komunikacije", a od 2018. je zamjenik glavnog i odgovornog urednika. U suradnji s tvrtkom Siemens Hrvatska d.d. bio predavač je na tečajevima za polaznike iz industrije te je 2018. godine primio priznanje za poseban doprinos unaprijeđenu nastave i edukaciji korištenjem SIMATIC automatizacijskih sustava. Aktivno je sudjelovao kao suradnik ili voditelj radnog paketa na 6 nacionalnih i 7 međunarodnih znanstvenih projekata. Tijekom 2013. i 2014. godine bio je gostujući istraživač u grupi prof. dr. sc. Francoisa Chaumettea na ustanovi INRIA Rennes-Bretagne Atlantique u Rennesu, Francuska. Objavio je 23 znanstvena rada u časopisima i više od 40 znanstvenih radova u zbornicima skupova u području estimacije, fuzije senzora i robotskog vida s primjenom u upravljanju autonomnim mobilnim robotima i vozilima. Član je društva Institute of Electrical and Electronics Engineers (IEEE), tehničkog odbora za robotiku društva International Federation of Automatic Control (IFAC) i Hrvatskog društva za komunikacije, računarstvo, elektroniku, mjerenja i automatiku (KoREMA). Godine 2018. primio je nagradu mladom znanstveniku "Vera Johanides" Akademije tehničkih znanosti Hrvatske, 2014. srebrnu plaketu "Josip Lončar" FER-a za posebno istaknutu doktorsku disertaciju, 2008. "INETEC nagradu" Instituta za nuklearnu tehnologiju za izvrsnost postignutu tijekom studija, a 2006. godine nagradu "Josip Lončar" FER-a za izvanredan uspjeh u toj akademskoj godini.

ACKNOWLEDGEMENTS

To my supervisor Assoc. Prof. Ivan Marković for his invaluable advice and help.
To Prof. Ivan Petrović for wisdom and guidance in crucial moments of my journey.
To the LAMOR group for helping in my academic endeavors and making my stay with them full of joy.
To my friends, old and new, for accepting me for who I am and sharing both the best and the worst moments of my life with me.
To my family for their unconditional support and love.
To Ana for being with me.

ABSTRACT

As robots are progressing towards being ubiquitous and an indispensable part of our everyday environments efficient and safe collaboration and cohabitation become imperative. Given that, such environments could benefit greatly from accurate human action and motion prediction. In addition to being accurate, human action prediction should be computationally efficient, in order to ensure a timely reaction, and capable of dealing with changing environments, since unstructured interaction and collaboration with humans usually do not assume static conditions. In this thesis, we focus on probabilistic decision-making models for human action and motion prediction in industrial human-robot shared environments. We first introduce a framework for human intention recognition in the robotized warehouse environment. This framework is based on Markov Decision Process action validation and the Hidden Markov Model intention recognition module. The warehouse floormap is searched to find optimal paths towards potential goals using Generalized Voronoi Diagrams and D* graph search algorithm. We continued by utilizing this framework for precise human motion prediction that served as input for the human-aware planning algorithm. The goal of human-aware planning is to reroute robots in the worker's way thus improving efficiency and retaining safety. We ran multiple experiments to test the proposed algorithms: in a real-world laboratory warehouse with a worker, wearing augmented reality glasses, a virtual reality warehouse twin as well an in-house developed warehouse simulator. Finally, we utilized Long Short-Term Memory networks to predict the next object human is going to pick in a collaborative environment. In order to reduce execution time, we crafted two dimensionality reduction methods. The first one is a feature selection method that relies on signal correlation and individual merit while the second one is a feature extraction method based on the autoencoder model. Heavy emphasis was put on the best predictor for human action prediction, the eye gaze, and an effort was made to estimate it using Multilayer Perceptron architecture. We used motion capture data from the publicly available dataset as well as on a smaller in-house recorded dataset.

KEYWORDS: human action prediction, human motion prediction, Markov decision process, hidden Markov model, probabilistic decision-making methods, recurrent neural networks, long short-term memory networks, feature dimensionality reduction, autoencoders, collaborative environments

SAŽETAK

Predviđanje čovjekova djelovanja i gibanja u suradnim industrijskim prostorima ljudi i robota zasnovano na vjerojatnosnim modelima odlučivanja

Robotski sustavi su sveprisutni u modernim industrijskim okruženjima poput tvornica automobila, prehrambenih postrojenja i automatiziranim skladištima. Oni uključuju autonomne mobilne robote i robotske ruke koji svojom preciznošću, lakoćom održavanja i efikasnošću omogućuju efikasniju i sigurniju proizvodnju. S razvojem suvremenih tehnoloških rješenja takvi sustavi bivaju potentniji i sofisticiraniji. Integriranje mnoštva senzora poput kamera, sustava za bilježenje pokreta i inercijalnih mjernih jedinica omogućuje precizno upravljanje robotima te napredno percipiranje okoline. Percepcija okoline podrazumijeva svijest robota o svojstvima ostalih objekata u sceni: njihov položaj, brzinu, semantičku interpretaciju. Poboljšanje stanja tehnike omogućilo je da okruženje robota, osim objekata, uključuje i ljude te takva okruženja nazivamo suradnim industrijskim prostorima. Primjeri takvih okruženja automazitirana su skladišta u kojima mobilni roboti pomažu radnicima noseći težak teret ili police s predmetima. Uvođenje ljudi u radni prostor robota povećava zahtjeve za sigurnošću sustava. Također, za razliku od robotskog djelovanja koje je određeno algoritmom upravljanja, ljudsko je djelovanje autonomno te unosi nesigurnost u sustav upravljanja automatiziranom okolinom. Zbog toga je precizno predviđanje čovjekovog djelovanja i gibanja nužan uvjet za efikasnu i sigurnu suradnju robota i ljudi.

Predviđanje čovjekovog djelovanja i gibanja problem je koji u posljednje vrijeme pokušavaju riješiti mnoge istraživačke zajednice. Osim suradnih prostora robota i ljudi, predviđanje čovjekovog djelovanja korisno je u područjima poput autonomnih vozila i naprednih nadzornih sustava. To je vrlo izazovan problem čiju težinu podiže, osim kompleksnosti čovjekova uma, brojnost unutarnjih i vanjskih podražaja koji mogu utjecati na proces odlučivanja čovjeka. Na ponašanje čovjeka utječu vlastite želje i ciljevi te konfiguracija okoline koja može uključivati i prisutnost drugih živih i neživih agenata. U slučaju da se u radnoj okolini čovjeka nalaze drugi ljudi, u obzir je potrebno uzeti društvene norme i međuljudske odnose. Zbog svega navedenog je rekonstruiranje čovjekovog procesa odlučivanja jedan je od najvećih izazova i tema istraživanja u području suradnje robota i ljudi.

Prvi pokušaji estimiranja čovjekove namjere i gibanja temeljili su se na predviđanju čovjekove putanje i trajektorije u dvodimenzionalnom prostoru. Te su metode najčešće koristile mrežastu kartu zauzetosti prostora te pokušavale pogoditi cilj čovjekova kretanja. Za to su korišteni matematički modeli poput inverznog optimalnog upravljanja i familije Markovljevih modela koji uključuju Markovljev proces odlučivanja, skriveni Markovljev model

i djelomično osmotriv Markovljev model. Zajednička karakteristika ovih pristupa pretpostavka je da se čovjek ponaša skoro-optimalno s obzirom na svoj cilj. Skoro-optimalnost podrazumijeva da se čovjek uglavnom ponaša optimalno prema nekakvom kriteriju koji može biti brzina dolaska do cilja, glatkost trajektorije ili izbjegavanje potencijalne opasnosti. Imajući to na umu, spomenute metode uspoređuju osmotreno gibanje čovjeka s optimalnim gibanjem naspram potencijalnih ciljeva u prostoru te zaključuju najvjerojatniji cilj i trajektoriju s obzirom na rezultate usporedbe. Bitan je dio tih modela planiranje čovjekove putanje prema potencijalnim ciljevima radi usporedbe hipoteza o osmotrenom ponašanju. Najčešće korišteni algoritmi za tu svrhu su za mrežastu kartu zauzetosti A^* i njegova dinamična verzija D^*, a u slobodnom prostoru često se koriste brzorastuća slučajna stabla i Gaussovi procesi. Glavni nedostatak metoda baziranih na skoro-optimalnosti je oslanjanje na pretpostavke o ponašanju ljudi koje ne moraju vrijediti u stvarnom svijetu. Također je potrebno estimirati čovjekovu pozornost i informiranost o stanju okoline jer su čovjekove akcije vođene vlastitom percepcijom vlastitog okruženja, a ne stvarnim stanjem okoline. Nepotpuna informiranost o čovjekovim unutarnjim stanjima može dovesti do propuštanja nekih obrazaca ponašanja koje bi fleksibilniji modeli mogli obuhvatiti. U novije vrijeme se problemu predviđanja čovjekovog djelovanja pristupa korištenjem metodama koje se zasnivaju na učenju iz podataka. Snimanje velikih skupova podataka omogućilo je korištenje algoritama strojnog, potpornog i dubokog učenja za predviđanje kretanja čovjeka i zaključivanje o njegovim namjerama. Najčešće su korišteni temporalni modeli poput povratnih neuronskih mreža i njihove izvedenice, mreže s dugom kratkoročnom memorijom. Ti modeli su sposobni učiti obrasce izravno iz podataka bez potrebe za dodatnim pretpostavkama o optimalnosti i računanjem putanja do ciljeva. Također, korištenjem naprednih metoda odabira značajki poput analize korelacijie ili autoenkoderskih arhitektura moguće je izlučiti najbitnije značajke za predviđanje čovjekova djelovanja i gibanja. Time je moguće znatno ubrzati metodu predviđanja te smanjiti količinu potrebnih senzora u suradnom prostoru bez većeg gubitka moći predviđanja i sigurnosti.

Cilj je ove disertacije razviti programski okvir predviđanja čovjekova djelovanja i gibanja u suradnim prostorima ljudi robota. Prvo poglavlje opisuje problem predviđanja čovjekova djelovanja, znanstvene doprinose disertacije i kratki sadržaj preostalih poglavlja. Drugo poglavlje donosi teoretski pregled korištenih programskih okvira, matematičkih modela i algoritama. Opširno se opisuju metode pronalaženja optimalnog puta i razlaganja prostora radi smanjenja vremena pretraživanja. Daje se podroban teoretski pregled vjerojatnosnih modela odlučivanja za predviđanje djelovanja i gibanja čovjeka te se opisuju metode odabira značajki korištene u disertaciji. Nadalje, donosi se pregled metoda za predviđanje sekvenci poput čovjekova kretanja baziranih na kinematičkom modelu i učenju iz podataka. Četvrto, peto i šesto poglavlje opisuju ostvarene vlastite znanstvene doprinose, dok sedmo poglavlje zaključuje disertaciju i daje uvid u daljnji razvoj algoritama predviđanja čovjekova djelovanja i gibanja. U nastavku slijedi kratki opis tri znanstvena doprinosa ostvarena u ovoj disertaciji.

#1 *Metoda predviđanja čovjekova djelovanja pomoću skrivenog Markovljeva modela za integrirane skladišne sustave zasnovana na mjerenju položaja čovjeka i podacima iz virtualne stvarnosti.*

Integrirani skladišni sustavi podrazumijevaju blisku suradnju ljudi i autonomnih robota u automatiziranim fleksibilnim skladištima. U takvim je sustavima najčešće implementiran centralni sustav upravljanja flotom vozila (*Fleet Manager System* - FMS) koji je odgovoran za planiranje njihovih trajektorija. Te se trajektorije planiraju s obzirom na unaprijed zadani skup zadataka koji se u skladištu moraju izvršiti. Osim planiranja, FMS vodi računa o lokacijama svih subjekata i objekata u skladištu te nadgleda izvršavaju li se svi zadaci u skladu s planom. U slučaju nepredviđenog događaja poput pojave novih zadataka, kvarova ili promjene namjera čovjeka, FMS mora pravovremeno reagirati i preusmjeriti robote da bi se održala efikasnost i osigurala sigurnost ljudi u skladištu. Kako bi se to ostvarilo, potrebno je precizno estimirati buduće djelovanje čovjeka u stvarnom vremenu i voditi računa o njemu prilikom planiranja novih trajektorija robota.

Jedan od preduvjeta za prepoznavanje namjera i djelovanja čovjeka poznavanje je pozicije čovjeka te razumijevanje njegove okoline. Konkretno, potrebno je precizno odrediti lokacije svih zanimljivih objekata u čovjekovoj blizini te ustvrditi može li ih čovjek vidjeti i je li svjestan njihove prisutnosti. Za to je potrebno poznavati poziciju čovjeka i njegov smjer gledanja, odnosno orijentaciju njegove glave što se vrši algoritmima lokalizacije. Lokalizacija ljudi i robota vrši se naprednim algoritmima koristeći razne senzore poput kamera, radiofrekvencijske identifikacije i lasera. S obzirom na napredak u razvoju tehnologije virtualne stvarnosti (*Virtual Reality* - VR) i proširene stvarnosti (*Augmented Reality* - AR), AR uređaji postaju lakši i ugodniji za nošenje te vrlo funkcionalni. Njihove mogućnosti uključuju vrlo precizno kartiranje i lokalizaciju, prikazivanje naprednih holograma u vidnom polju korisnika i reproduciranje zvuka. Takvi se uređaji prirodno mogu integrirati u okruženje robotiziranog skladišta jer na njima je moguće prikazivati relevantne podatke radniku poput sljedećeg zadatka i stanja okoline. Korištenje podataka AR uređaja zato omogućuje prepoznavanje djelovanja čovjeka te pospješuje interakciju čovjeka i robota. S obzirom na to da treniranje osoblja za korištenje AR uređaja može biti dugotrajan proces koji bi zahtijevao njihovo izlaganje opasnostima u skladištu ili njegovo zaustavljanje, moderna skladišta koriste digitalne blizance napravljene u virtualnoj stvarnosti. Takva skladišta tlocrtom i izgledom odgovaraju pravom skladištu uz jednaku funkcionalnost nosivog uređaja.

Cilj je prvog znanstvenog doprinosa razviti metodu predviđanja čovjekova djelovanja za integrirane skladišne sustave zasnovanu na mjerenju položaja čovjeka i podacima iz virtualne stvarnosti. Pretpostavka ovog modela jest postojanje ograničenog broja unaprijed poznatih zanimljivih lokacija u skladištu, a te lokacije zovemo potencijalnim ciljevima čovjekova djelovanja odnosno kretanja. Položaj čovjeka estimira se korištenjem senzora ugrađenh u Microsoft HoloLens naočale za proširenu stvarnost dok je položaj robota u prostoru dobiven fuzijom signala odometrije i detekcije markera na podu skladišta. Pomoću tih se signala agenti smještaju na unaprijed dostupnu dvodimenzionalnu kartu skladišta. Korištenjem generaliziranih Voronoijevih dijagrama i D* algoritma pretraživanja grafa pronalazi se optimalna putanja do svakog cilja. Analizom kretanja čovjeka i usporedbom s optimalnom putanjom računa se relativno odstupanje za svaki cilj. Taj se signal obrađuje predloženim

skrivenim Markovljevim modelom. Predloženi skriveni Markovljev model sadrži po jedno skriveno stanje za svaki cilj te dva dodatna stanja: iracionalni agent i nepoznati cilj. Dodatna su stanja uvedena kako bi se modelu na prirodan način omogućilo izražavanje nesigurnosti predviđanja. Pomoću Viterbijevog algoritma za računanje očekivane distribucije skrivenih stanja dobiva se vjerojatnost svakog cilja što predstavlja izlaz modela. Predloženi je model testiran u robotiziranom skladištu i u digitalnom blizancu skladišta pomoću VR tehnologije. Skalabilnost modela pokazana je korištenjem većih skladišta s većim brojem robota u VR okruženju.

#2 *Metoda predviđanja čovjekova gibanja za planiranje trajektorija robota svjesno blizine čovjeka u integriranim skladišnim sustavima.*

U integriranom robotiziranom skladištu FMS vodi računa o poziciji svih agenata (robota i ljudi) i njihovim zadacima, a istovremeno pazi na efikasnost i sigurnost. Centralna zadaća FMS-a jest planiranje trajektorija za sve agente što je vrlo kompleksan zadatak i temelji se na preslikavanju strukturirane karte prostora u graf resursa, gdje svaki resurs ima zadani vremenski tijek. Vremenski tijek podrazumijeva slobodne i zauzete vremenske prozore koji ukazuju na to je li određeni resurs slobodan ili zauzet od strane nekog agenta. Iterirajući po agentima planira se optimalna putanja za svakog pojedinačno uz ograničenja koja osiguravaju nesmetano rješavanje konflikata. Također, potrebno je planirati trajektorije na način koji ne ugrožava ljudski život. Zbog toga se uvode tri razine ograničenja koje se nazivaju sigurnosne regije definirane pomoću udaljenosti od čovjeka u skladištu. Prva sigurnosna regija nalazi se u neposrednoj blizini čovjeka i robot se zaustavlja u slučaju da se nađe u njoj. Drugu sigurnosnu regiju definiramo većim radijusom od prve i FMS ne šalje robote u tu regiju prilikom planiranja. Treća sigurnosna regija predstavlja područje snižene brzine, ali je kroz nju moguće planirati robotovu putanju. Predstavljeni planer daje prioritet čovjekovom nesmetanom gibanju te je vrlo osjetljiv na njegovo slijeđenje unaprijed zadanje putanje.

U slučaju da čovjek ne slijedi unaprijed zadanu putanju potrebno je takvo ponašanje detektirati i pravovremeno reagirati. Cilj je drugog znanstvenog doprinosa razviti metodu predviđanja čovjekova gibanja za planiranje trajektorija robota svjesno blizine čovjeka u integriranim skladišnim sustavima. Prvi korak predloženog algoritma utvrđivanje je devijacije od zadane putanje pomoću definiranog dozvoljenog područja devijacije. U slučaju da se čovjek ne nalazi unutar dozvoljenog područja dulje od jedne sekunde pretpostavlja se da više ne slijedi zadanu putanju. Tada se šalje upit modulu za predviđanje čovjekova gibanja koji ovisno o razini pouzdanosti u svoje predviđanje dojavljuje novu trajektoriju ili poruku da se ona ne može zaključiti s visokom razinom pouzdanosti. Predviđanje trajektorije vrši se pomoću već predstavljenog skrivenog Markovljevog modela i pretpostavke o jednolikom gibanju čovjeka. Kako bi se ispitao utjecaj predloženog modula za detekciju devijacija i predviđanje trajektorije, potrebno je bilježiti aktivnost u skladištu: broj uspješnih dostava robota i čovjeka kao i bliske susrete robota i ljudi te ukupan broj replaniranja putanja u minuti.

#3 *Metoda predviđanja čovjekova djelovanja zasnovana na kinematičkom modelu čovjeka i vjerojatnosnom odlučivanju u suradnim prostorima ljudi i robota.*

Zahvaljujući razvoju tehnologije napredni robotizirani sustavi nisu ograničeni na kohabitaciju s ljudima, već dolazi do povećanja suradnje robota i ljudi u industrijskim okruženjima. Takva okruženja donose sa sobom nove izazove, ponajviše vezane za efikasnost i sigurnost zbog stohastičnosti ljudskog ponašanja i gibanja. U robotiziranim skladištima najčešći zadatci su prenošenje predmeta i slaganje na police. Za takve sustave moguće je postići zavidnu razinu sigurnosti i efikasnosti preciznim predviđanjem buduće lokaciju čovjeka u dvije dimenzije, to jest na karti skladišta. Tada nije potrebno predvidjeti gibanje cijelog kinematičkog lanca čovjeka, odnosno položaja udova u vremenu. Međutim, u suradnim prostorima poput varionica i operacijskih sala gdje robotski manipulator pomaže čovjeku u izvršenju zadatka potrebno je precizno osmotriti gibanje čovjekova kinematičkog lanca i pomoću njega zaključiti ciljeve čovjeka i njegovo buduće gibanje.

Cilj je trećeg doprinosa razviti metodu predviđanja čovjekova djelovanja zasnovanu na kinematičkom modelu čovjeka i vjerojatnosnom odlučivanju u suradnim prostorima ljudi i robota. Odabrana metoda za rješavanje ovog problema zasnovana je na povratnoj mreži s dugom kratkoročnom memorijom (*Long Short-Term Memory Network* - LSTM) koja je nadograđena verzija povratne neuronske mreže (*Recurrent Neural Network* - RNN). Navedene se arhitekture temelje na učenju uzoraka iz podataka. S pojavom javnih skupova podataka koji sadrže snimljene podatke o kinematičkom lancu čovjeka te smjera pogleda u scenariju uzimanja i odlaganja predmeta moguće je naučiti parametre navedenih arhitektura. S obzirom da su navedene metode slojevite, odnosno ovisno o stupnju složenosti mogu sadržavati velik broj parametara, a dostupan broj podataka je ograničen, potrebno je odrediti ulazne značajke u navedene modele. Snimljeni kinematički lanac sadrži nekoliko desetaka položaja i orijentacija čovjekovih zglobova te se predlaže metoda odabira značajki temeljena na korelaciji. Dodatno, predaže se alternativna metoda odabira značajki direktno iz podataka temeljena na autoenkoderskoj arhitekturi. Snimljen je vlastiti skup podataka koji umjesto cijelog kinematičkog lanca bilježi samo položaj ruke i orijentaciju glave te su rezultati uspoređeni s prethodnom metodom. Konačno, s obzirom na to da je uređaj za bilježenje smjera pogleda skup, nezgrapan i ne može se koristiti s naočalama, razvija se metoda estimacije smjera čovjekova pogleda.

KLJUČNE RIJEČI: predviđanje čovjekova djelovanja, predviđanje čovjekova gibanja, Markovljevi procesi odlučivanja, skriveni Markovljev model, vjerojatnosni modeli odlučivanja, povratne neuronske mreže, autoenkoder, odabir značajki, povratna mreža s dugom kratkotrajnom memorijom, suradni prostori

CONTENTS

1

Introduction

The introduction chapter presents and elaborates the motivation behind the research conducted in the thesis. We shall start by discussing what exactly we mean by *human action and motion prediction in industrial human-robot shared environment* and how we approached solving this problem. Then, we discuss probabilistic decision-making models and supplementary frameworks used for solving them. We shall also discuss and break down the title of the thesis and see how it relates to the contents of the thesis research. The main challenges in solving this problem are laid out, followed by a discussion of various approaches taken. Afterward, the original scientific contributions of the thesis are presented with more detailed elaborations. In the end, the outline and structure of this thesis are sketched, with a short description and main aspects covered in each of the chapters.

1.1 Motivation and problem statement

The European e-commerce turnover managed to increase 10% to €757 billion in 2020, leading the way in the challenging times and environments [1]. With the internationalization of distribution chains, the key to success lies within efficient logistics, consequently increasing the need for larger warehouses and their automation. As robots are becoming more capable and sophisticated, we are witnessing growth in their presence and integration in both private and professional human environments. Nowadays, such shared environments, besides cohabitation, often include close human-robot collaboration and interaction. Examples of such environments are warehouse automation solutions such as Swisslog's CarryPick Mobile system and Amazon's Kiva system [2] which use movable racks that can be lifted by a fleet of small, autonomous robots. By bringing the product to the worker, productivity is increased by a factor of two or more, while simultaneously improving accountability and flexibility [3]. However, current automation solutions are based on strict separation of humans and robots; the worker is not allowed to enter the shop floor during operation due to safety reasons, since a robot with a rack can weigh together up to a ton. When moving they are posing a significant risk to all humans nearby. Therefore, when human intervention is needed on the shop floor, the whole fleet of mobile robots has to be stopped and remain stopped until the worker has again left the shop floor. With the increasing size of warehouses, such events immensely impact operational efficiency. Therefore, a novel integrated paradigm arises where humans and robots will work closely together and these integrated warehouse models will fundamentally change the way we use mobile robots in modern warehouses.

Besides safety, as the fundamental requirement in every human-robot interaction (HRI) scenario, the proposed system has to take usability into account. It has to be ensured that the worker is assisted and not impeded during work. One way of achieving this is to, on the one hand, ensure that robots are avoiding the area near workers and, on the other hand, instruct the worker to reach their goal through robot-free zones and corridors. Given that, we assert that a future warehouse system, which will have to orchestrate and coordinate human workers and robots, would significantly benefit if it were able to estimate workers' intentions correctly and control the robots accordingly, so that warehouse operation efficiency is ensured. While robots operating in these environments are fully controllable, human behavior, on the other hand, is not. Because of that, in order to ensure efficiency and safety, robots in human proximity should be aware of possible changes in human intentions and react accordingly. Having that in mind, one of the main challenges in collaborative environments is to capture the uncertainty and nuances of human behavior, intentions, and actions. In order to do that, the supervisory system has to be aware of the context the human is in, measure appropriate features, and craft a method that works reliably and in real time.

Let us continue now by decomposing the title of the thesis. The "human action prediction" pertains to a wide range of possible human activities in industrial environments. It is important to note that the focus of this thesis is not *human action recognition* which, while sounding similar, is quite a different problem that aims to understand human behavior and assign a label to each action. For example, a problem falling into the human action recognition domain is determining whether the observed subject is sitting, running, or jumping from video or data from a fusion of various sensors. On the other hand, *human action prediction*, also called *human intention recognition* in the literature, limits the type of action with respect to the environment. Namely, in the manufacturing domain, we try to predict which object a human is going to pick next, and in the integrated warehouse domain, we guess the next shelf or warehouse exit a human is going to visit. By limiting ourselves to a concrete set of expected outcomes, we can concentrate on discriminating between these and polishing the method for timely and precise prediction of the next goal of a human. There exists a plethora of challenges in human intention estimation, because of the subtlety and diversity of human behaviors [4]. Contrary to some physical characteristics, such as position and velocity, human intention is not directly observable and usually needs to be estimated from human actions. It is also imperative to put those actions in context because even basic behaviors, such as walking, running, and jumping, are interpreted differently in, e.g., sports events, office, and warehouse environments. Furthermore, because human intention estimation should serve as input to decision-making processes [5], the intentions should be estimated in real-time, and overly complicated models should be avoided. Having that in mind, only the actions with the greatest influence on intention perception should be considered as inputs to the human intention recognition model. For example, in the warehouse domain, workers' orientation and motion as well as mobile robots' movement have a large effect on goal intention recognition. On the contrary, observing, e.g., a worker's heart rate or perspiration could provide very little, if any, information on the worker's intentions. Therefore, such measurements should be avoided in order to reduce model complexity and ensure real-time operation [4].

The "human motion prediction" aims to generate a set of future frames of human motion based on an observed sequence. Generally, it can leverage inputs from cameras, wearable devices, or motion capture systems and use various mathematical models and frameworks to yield the prediction. The nature of the aforementioned prediction can be a sequence of locations on a map, skeleton poses, or probabilistic distributions. In this thesis, we observe human action and motion prediction as coupled problems and approaches to solving them simultaneously.

There are plenty of domains of the human enterprise including numerous outdoor and indoor environments. Devising a general model for human action prediction is hard, if not impossible, without aiding it with some spatial context. For example, certain actions and movement nuances can carry different meanings based on the environment they are manifested. An observed behavior pattern can be interpreted differently during robot-aided operations, hunting, or shopping for groceries. Because of that, in the scope of this thesis, we put emphasis on "human-robot shared environments". These environments are constituted of one or more human workers paired with one or more autonomous robots, together commonly labeled as agents in the literature. Humans and robots coexist and even collaborate in such shared environments and the actions of humans are heavily coupled by the behavior of the robot because controlling robots is dependent on human behavior. The aforementioned environments are very complex and this thesis focuses on two common modalities. The first one is integrated robotized warehouse systems where a fleet of mobile robots can move under the racks as well as carry them while human workers perform maintenance, inspection, and picking duties. These warehouses can be exceedingly large and safe path planning while retaining efficiency is a major challenge in such an environment. Modern-day solutions equip workers with artificial intelligence wearable devices pointing them in the right direction, listing tasks, and monitoring their status. Such devices can be leveraged to send information about human movement and status to the supervisory system which can utilize this information to precisely infer their actions and future movement. The other environment the focus of this thesis is the collaborative environment in the object-picking domain. Examples of such environments could be hospital operations with the help of a robotic manipulator or a robot-aided welding manufacturing shop. In these environments, the actions and movements of human are more delicate and the sensor system capturing them are more sophisticated. These systems, such as motion capture systems, are utilized to reconstruct the human kinematic model which can be used to precisely predict actions such as picking and carrying objects in the scene.

Finally, we discuss what the "probabilistic decision-making methods" term encapsulate. Solving a human action prediction problem is not a trivial or easy task. It is necessary to consider spatial context, previously observed sequences, and potential goals of the observed behavior. The models used for this must be complex, utilize available data, and work in real-time to be useful to the supervisory system and they also need to couple intent and action estimation with the degree of certainty of current inference. Because of this, models and mathematical frameworks in this thesis are probabilistic - meaning they deal with the uncertainty in a natural way modeling it with probabilistic distributions and interpreting it accordingly. Because humans often behave nearly rationally [6] with respect to their desires the natural approach to decode human behavior is to try to formulate a mathematical

decision-making model mimicking the result - a predicted behavior. Having that in mind, we focus on utilizing probabilistic decision-making methods, both experimentally devised and tuned as well as learned from data, to predict future human actions and movement in industrial human-robot shared environments.

1.2 Original scientific contributions

The original contributions of the thesis essentially revolve about crafting both data-driven and manually-tuned probabilistic decision-making methods for human action and motion prediction in human-robot shared environments. The contributions and a brief elaboration follow in the sequel.

- A method for human action prediction using a hidden Markov model for integrated warehouse systems based on human pose measurements and virtual reality data.

  This contribution covers algorithms and frameworks developed for human action prediction. Firstly, a map of the environment is extracted and a generalized Voronoi diagram (GVD) is calculated. Nodes of the GVD are used for associating human positions obtained from Microsoft HoloLens Augmented Reality wearable device. As robots move in the warehouse, the optimal path towards certain interesting points in the warehouse, called goals, changes and is recalculated using the D^* algorithm. Distance between associated nodes and goals is used as observation for the proposed Hidden Markov Model. Alternatively, these distances are obtained using Markov Decision Process in the simulated warehouse environment. The Hidden Markov Model is solved using the Viterbi algorithm and goal probabilities are considered predictions of future human intention and action.

- A human motion prediction method for human-aware robot trajectory planning in integrated warehouse systems.

  This contribution encompasses the application of the previous contribution in synergy with the Human Aware Planner. The human deviation algorithm is presented and human trajectory prediction is performed using estimated goal and constant velocity assumption. Human deviations from the original plan are detected detailed procedure for replanning is crafted in order to maximize warehouse throughput and minimize potential human-robot collisions and needed replanning.

- A method for human action prediction based on their kinematic model and probabilistic decision-making in human-robot shared industrial environments.

  This contribution comprises a method for human action prediction based on their kinematic model. The kinematic model of humans is obtained using a Motion Capture suit and the human gaze is sampled from a dedicated wearable device. These signals are used as input to a proposed ensemble of Recurrent Neural Networks (RNNs) and Long Short-Term Memory Networks (LSTMs). In order to improve execution time, features are carefully selected using correlation analysis and an autoencoder-inspired multilayer perceptron architecture.

1.3 OUTLINE OF THE THESIS

The thesis is organized into six chapters. Each chapter begins with a short abstract which serves to present generally the content of the chapter, its results, and the insights it offers. Afterward, the reader is gradually introduced to the problem and the related work in the field. After the body of the chapter, in the end, a summary is given which restates some of the main results of the chapter and its contributions. Hereafter, we present the outline of the thesis with a short summary of the contents.

CHAPTER 2 .This chapter presents the general mathematical background for the thesis and sets up the context of the problem at hand. Space sampling methods, namely occupancy grid and General Voronoi Diagrams are introduced together with A^* and D^* path planning algorithms. Furthermore, probabilistic decision methods used in the thesis are presented with a focus on Markov models and neural network modalities such as RNN and LSTM. Finally, the correlation of temporal signals is analyzed together with Multilayer Perceptron-based methods used for feature selection.

CHAPTER 3 .A novel approach to human action prediction for integrated robotized warehouses based on Markov decision models is introduced in this chapter. The generalized Voronoi diagram algorithm is used for the warehouse map partitioning and searches on the generated graph is performed with A^* and D^* graph search algorithms. Distances between nodes of the graph and the human position associated with them serve as observations for the proposed hidden Markov model which yields probabilities of goals for human actions. The algorithm's performance is evaluated using data from the real warehouse and its larger digital twin designed in Virtual Reality.

CHAPTER 4 .In this chapter, a method for human motion prediction in the integrated robotized warehouse is given. An overview of the Fleet Management System is given and a procedure for planning paths of human workers and autonomous ground vehicles is presented. A novel human deviation detection algorithm is proposed with an emphasis on industrial warehouse domain application. Previous findings on human action prediction are expanded to yield predicted trajectories used for rerouting robots if a high probability of collision is anticipated. Detailed testing of key metrics is performed including the average number of human deliveries, robot deliveries, replanning number, and warehouse shutdowns.

CHAPTER 5 .In this chapter, a method for human action prediction based on their kinematic model is presented. The human kinematic model is obtained from a motion capture system and multiple positions and orientations of joints are observed. Distance towards potential goals and joint orientations towards them are considered observations for the proposed ensemble of Long Short-Term Memory Networks which perform prediction of the next picking actions. In order to speed up the inference, a feature selection method based on correlation and individual merit is proposed as well as a data-driven feature selection method based on neural networks. The proposed framework is thoroughly tested

on a public dataset and an in-house recorded dataset with a reduced number of observed joints is recorded and a gaze estimation method is proposed and tested.

↬ CHAPTER6 .This chapter brings conclusions and a summary of the scientific contributions. Some ideas for future work are given as well.

2

General Background

This chapter presents the mathematical background and several tools utilized in the thesis. Used sensors and a basic overview of their principles will be brought out in order to better understand the implementation implications of the proposed algorithms. The emphasis is set on probabilistic decision-making methods which include Markovian models and neural networks. Mathematical details of used frameworks such as the Markov Decision Process, and Hidden Markov Model are presented and algorithms used for solving them are laid out. These algorithms include the iterative Viterbi algorithm, and policy and value iteration algorithms. Real-time human action prediction depends on the fast search for the optimal path in the integrated warehouse domain. Because of that, we will introduce occupancy grid maps, a representation of warehouse plans, and common algorithms used for searching for optimal paths utilized in this thesis. These algorithms include space partitioning Generalized Voronoi Diagram, as well as graph searching algorithms A* and its dynamic version D*. Furthermore, we will introduce neural network frameworks and concepts, namely Multilayer Perceptron architecture, encoders, decoders, and autoencoders as well as recurrent architectures; the Recurrent Neural Network, and its more advanced version Long Short-Term Memory Network. We will discuss network architecture which includes hidden dimensions, layers, learning methods, and optimization. Special importance will be given to a feature selection process for which we will introduce signal correlation. In the end, we will give a brief overview of testing techniques used for obtaining statistically significant results, with an emphasis on the cross-correlation method.

2.1 Introduction

The main goal of this thesis is to precisely and timely predict human actions and motion in human-robot shared industrial environments. These environments include multiple types of robots such as collaborative robots, fixed-base manipulators, mobile robots, and mobile manipulators. When it comes to sharing an environment with humans, literature [7] distinguishes three types of human-robot interaction (HRI):

- Human-Robot Coexistence, where humans and robots share the same working space but perform tasks with different aims. Examples of such environments are integrated warehouses where humans perform maintenance work while robots carry pallets.
- Human-Robot Cooperation, in which humans and robots perform different tasks

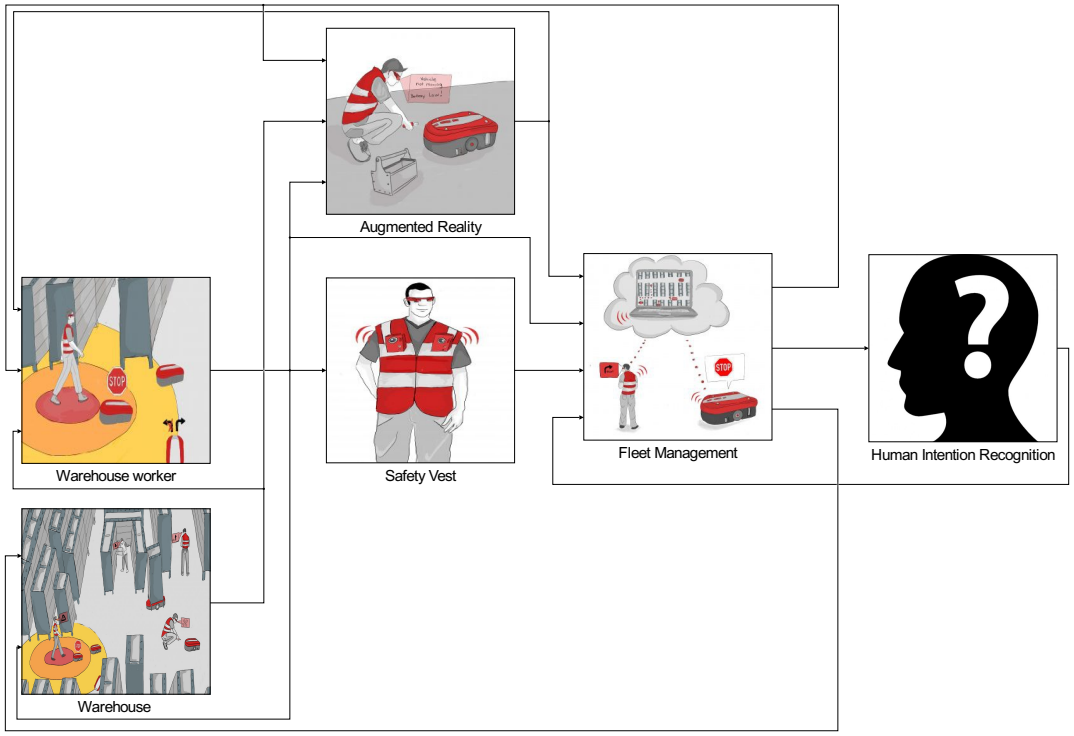


Figure 2.1: Human intention recognition as part of the SafeLog Project architecture. The fusion of sensors on a Safety Vest, Augmented Reality glasses, and those positioned around the warehouse is leveraged as input to the human intention and action recognition algorithm. The algorithm reports back estimations of the human intentions to the Fleet Management System which takes this information into account to make decisions about rerouting robots.

but with the same objectives that should be fulfilled simultaneously in terms of time and space. Such environments can be imagined as an advanced integrated warehouse where a robot brings pallets to a human worker who picks needed objects, further inspecting them.

- Human-Robot Collaboration, where a direct interaction is established between the human operator and the robot while executing complex tasks. We can imagine a collaborative welding environment, where robots help the human worker with preparatory tasks and directly assist in the process.

Each of these levels of interaction requires robots to be aware of human positions, intentions, actions, and future motion. In order to achieve this, human-robot shared environments are equipped with a plethora of sensors observing the positions and states of all agents. For example, the Safe human-robot interaction in logistic applications for highly flexible warehouses (SafeLog) project aims to bring human-robot interaction to life in a warehouse domain. The human interaction module as part of the proposed Fleet Management system in the context of integrated warehouse sensors can be seen in Fig.2.1where

it serves a crucial role in managing unpredictable situations accounting for uncertainty in human intentions and motion. As already stated, the human intention recognition and action prediction problem are complex because it follows from estimating human desires that are nuanced and often prone to sudden change. Because of that, it is important to feed a model as much context and information about the surrounding world as possible. In the warehouse domain, such information might consist of the known environment (floor map), the position of robots, and details about interesting or dangerous spots in the warehouse. We would naturally look at the position of a human worker on the warehouse map and where is the worker looking as our cues for inferring desires. If we are interested in object picking domain or some other cooperative or collaborative environment, then it is essential to capture more details of human movement which includes an effort to reconstruct the human kinematic model. Also, recording the direction of the subject's eye gaze has shown to be pivotal to human action prediction problems in recent research [8, 9, 10, 11]. Having this in mind, we dedicate Section2.2to sensors we utilized for observing human motion, eye-gaze direction, and environment. We will discuss their principle of operation, common use, and implications they have for our use case.

After we introduce sensors used for the perception of the environment and human motion, we will address methods and frameworks used for human action recognition and motion prediction. The leitmotifs of these sections will be real-time execution and uncertainty handling via probabilistic interpretation of inputs. Firstly, in Section2.3we will discuss path-planning algorithms in the shared environment domain. These algorithms help us to set a baseline for the comparison of observed human motion to the optimal one with respect to the predefined set of goals. In order to find these paths, we will introduce optimal and complete graph search algorithms - the A* and D* algorithms. Their pseudocode will be laid out and optimality and completeness considerations will be given. We will discuss choices of environment representation and introduce occupancy grid maps as our method of choice. In order to reduce the search space and improve execution time, the Generalized Voronoi Diagrams are used. Their definition and applications are discussed in Section2.4. The optimality of a given path sometimes usually relates to total time, traveled distance, or smoothness of the resulting trajectory but sometimes can not be explicitly stated. For example, avoiding noisy, narrow or for some other reason, unsafe parts of the scene or environment can not be trivially embedded into a graph representation of the environment. Because of that, we introduce the Markov Decision Process model which assigns each state (location) a reward. By solving for an optimal reward using value and policy iteration algorithms one can find an optimal path and evaluate human behavior.

Furthermore, in Section2.5we introduce probabilistic decision-making methods used in the scope of this thesis. Their role is to interpret observed motion obtained via utilized sensors and put it into context with the help of path-planning methods. For this task, we will select two approaches, the first one being the Hidden Markov model, a reinforcement learning-based framework that extends MDP with the addition of the hidden state concept. The hidden state naturally mimics human intention or desire that needs to be concluded and reconstructed from observed motion and actions. We will explain the details of HMM architecture and principles of the Viterbi algorithm, the iterative algorithm used for solving the most probable sequence of the hidden state. The other approach we will take for the

estimation of human actions and motion is based on the neural network paradigm. The core principles of neural networks, especially Multilayer Perceptron will be laid out with an emphasis on key elements such as layers, backpropagation, activation functions, and architecture details. In order to interpret the time-series data, such as observed human motion, it is necessary to propagate information from previous time steps into the inference method. We introduce Recurrent Neural Networks (RNNs) and the more advanced Long Short-Term Memory (LSTM) networks for this task, with an in-depth explanation of their innovative recurrent connection. Since the architecture is just one part of any neural network model, we will discuss training and testing methods as well as statistically significant result interpretation, giving emphasis on the cross-correlation method. As the human action and motion, prediction problem must be solved in real-time to be useful to the supervisory system [4], and neural network approaches run time can rise considerably with the number of input features and layers [12], recent works aim to reduce feature number while obtaining satisfactory precision. We will introduce the mathematical background of correlation signal analysis which is one of the methods used in literature for reducing the feature space thus improving execution time. Finally, we will summarize the introduced sensors, models, and algorithms in Section2.6.

2.2Utilized sensors

The main feature distinguishing a robot from a common machine is its capability of carrying out a complex series of tasks and actions automatically, as per Oxford dictionary, [13]. One of the main assumptions for automatic or autonomous behavior is awareness of the environment the robot operates. Having that in mind, the field of robotics heavily relies on the perception of the surroundings to operate safely and efficiently [14]. Tools used for perceiving the environment are commonly known as sensors and can be defined as devices that detect and respond to some type of input from the physical environment. Sensors are commonly divided into two groups in literature, depending on the target value they are measuring:

- Proprioceptive sensors that measure the inner states of a robot. Examples of such sensors are odometry sensors (encoder, accelerometer) computing the pose of a robot and orientation sensors (gyroscope, compass).
- Exteroceptive sensors that measure external states of the environment. Examples of such sensors are cameras, lasers, and various ranging sensors.

While these sensors give useful raw data about the environment, these pieces of information have to be put in context for intelligent decision-making. For example, the most commonly used type of camera in robotics, the RGB camera, captures images with a certain resolution assigning each pixel its red, green, and blue value. The values of those pixels can be affected by the noise of the sensor and motion blur [15]. Also, in order to extract and track the exact position and orientation of humans from the camera frame, complex algorithms relying on deep learning are commonly employed [16]. These algorithms run slow and require a substantial dataset to be trained on. Algorithms such as simultaneous localization and

mapping (SLAM) [17] and Kalman filter [18] and its variations [19] leverage sensor fusion for solving problems such as mapping, localization, and object detection tracking. They are successfully implemented in various human-robot shared environments and scenarios. An example of such a complex environment robotized system, where information from multiple sensors is fused to improve the reliability of worker and robot localization in the integrated robotized warehouse, can be seen in the Figure2.2.

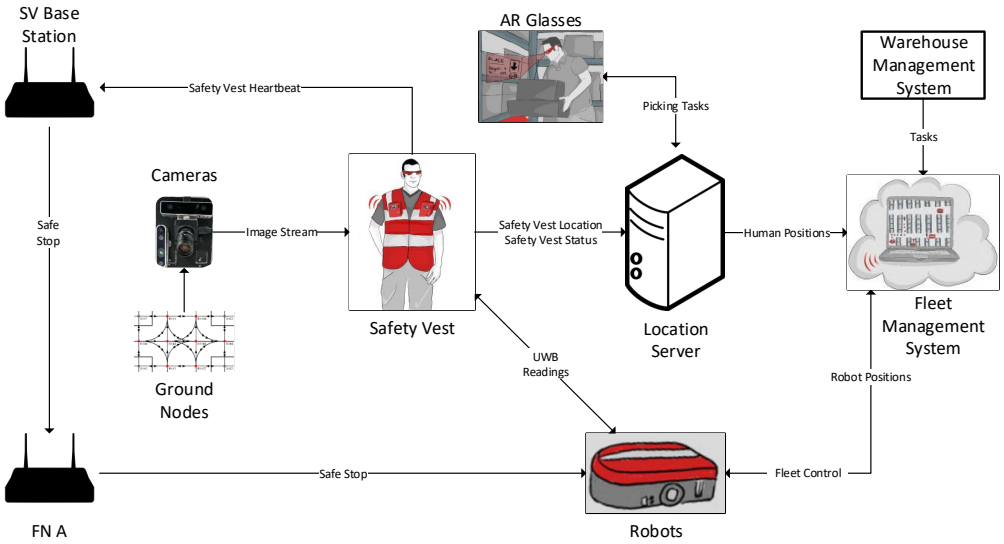


Figure 2.2: Sensors utilized in the SafeLog Project include, among others, Ultra-wideband ranging sensors, cameras detecting ground nodes, and Augmented Reality glasses aiding the localization.

In the scope of this thesis, we will be mainly interested in the precise positions and orientations of human joints in the robotized environment. While we will use information about the environment, such as the location of robots and interesting objects as well as the structure of the scene which includes mapping of the environment, proposed algorithms will not access this information directly. Because of that, we will continue by expanding on the sensors directly used as part of human action and motion prediction. First of these sensors are Microsoft HoloLens augmented reality glasses that, apart from their display capabilities, actively map the environment and localize themselves in it. As the software used for localization and mapping is proprietary, we will briefly present commonly used state-of-the-art frameworks that solve this problem. The second sensor system we will introduce is an Optitrack motion capture (MoCap) system for fast and reliable marker detection and tracking. Furthermore, the system's ability to reconstruct the human skeleton and track the position of certain joints is of great importance for the goals of this thesis. We will discuss the implications these sensors have on the performance of our algorithm and the possible limits they might impose.

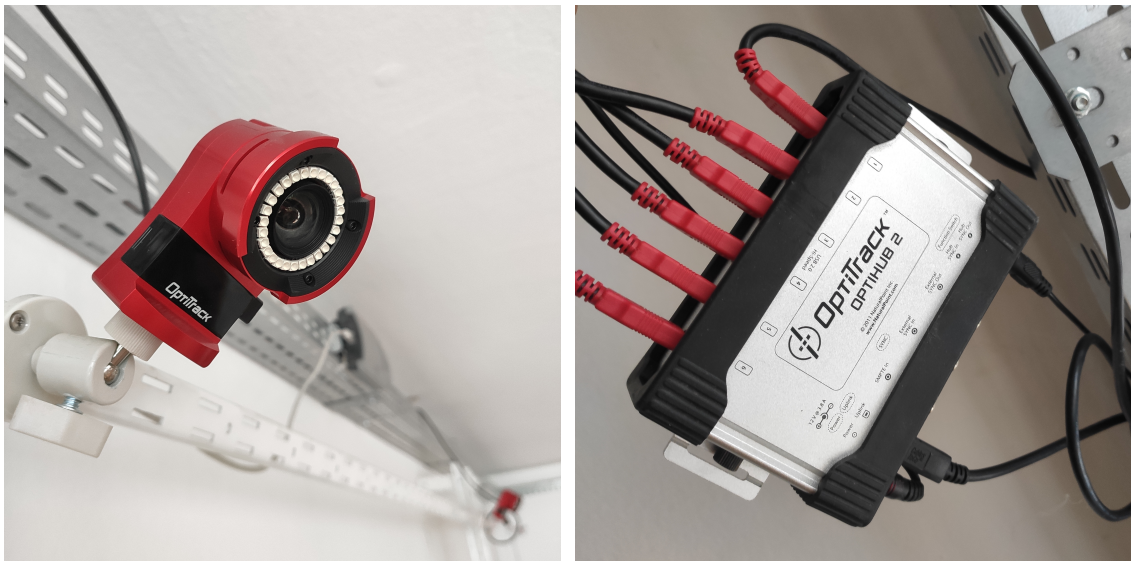
Figure 2.3: The Microsoft HoloLens 2.0 Development Edition augmented reality glasses are used for several experiments in the scope of this thesis.

2.2.1 *Augmented and Virtual Reality*

Augmented reality (AR) and virtual reality (VR) themselves have seen a big resurgence in robotics in recent years [20]. VR refers to systems where the input from the outside world is totally blocked and replaced by a system-generated input. The first VR system was built in 1968 by Ivan Sutherland [21]. The device was extremely bulky and the screen resolution was poor; however, it proved that VR was achievable. The first usable VR system came in 1992 with the CAVE system [22]. The CAVE is a special room where electromagnetic (now infrared) trackers track 3D glasses (now usually with active polarization) and projectors display the appropriate images on the room's surfaces. With the hardware advancements, the CAVE system has been replaced by cheaper and more flexible headsets such as the *Oculus Rift* or the *HTC Vive*. This has sparked a boom in the field of robotics where VR has recently seen the most use as a more intuitive method for teleoperation of stationary [23], mobile [24] and humanoid robots [25]. A natural extension of such a teleoperation system is a more immersive tool for telepresence robots [26]. It has also emerged as a method to teach virtual robots how to perform tasks [27], where the knowledge is then successfully transferred to a real robot. In manufacturing [28] VR has seen use as a virtual prototyping [29] and training tool [30]. As a training tool, VR has been shown to increase the performance of trainees in other areas as well, such as medicine [31], safety training in construction [32], and mining [33]. Although viable as a system, especially for prototyping [34], it is not flexible enough to be widely employed, requiring expensive sensors and a purposely designed room.

Augmented reality, in contrast to VR, seeks to add information to the input from the real world. The first AR systems started to appear in the 1980s, mostly for military research. The most commonly quoted first functional AR system was the *Virtual Fixtures* system [35], developed in 1992 by Louis Rosenberg for the US Air Force. Quickly its usefulness became apparent in other fields as well, such as manufacturing, medicine, entertainment, and robotics, where the first use cases focused on assistance in robotic teleoperation via a stereo camera pair [36]. These first AR systems in robotics added data to a camera stream and displayed the enhanced view on computer monitors [37]. Today's AR can be mostly divided into projector-based, tablet or smartphone-based, head-mounted, and computer screen-based. The projector-based AR has seen use for visualizing robot's intentions and intuitive programming of robots in robot work cells [38], displaying intentions of mobile robots operating in human environments [39], as well as for debugging and rapid prototyping of robotic systems through visualization of sensor data [40]. Tablets have seen use in AR-assisted robot programming [41] as well. Devices like the *Google Tango* tablet, in addition, with inbuilt SLAM [42] can be used for markerless AR applications. The main challenge with tablets is that it occupies the hands, preventing any work while the AR information is visualized. With the recent releases of the ARCore and ARKit toolkits, for Android and iOS devices respectively, tablet and smartphone-based AR applications are becoming an economical and straightforward interaction modality for home robots [43]. Head-mounted systems can be further divided into Heads-up Display (HUD) systems and "full-AR" systems, with the most famous member of the former being the *Google Glass* and of the latter *Microsoft Hololens*. As the name implies, HUD-based systems do not have any advanced localization or computing systems and therefore are only able to display interfaces, while full AR systems are able to perform localization using SLAM and display persistent, full 3D holograms in space. HUD systems have found applications in logistics, where they have been used for *pick-by-vision* systems to quicken and ease the picking of items in warehouses [44]. Since full AR systems, starting with the *Hololens*, have not been on the market for long, research is just starting [20], with the most prominent field perhaps being AR assisted robot programming [45]. Given the described potential, in this thesis, we will leverage these systems for providing human intention cues and constructing realistic experimental scenarios. For example, AR can be used to track worker motion inside the warehouse as well as display valuable information, e.g, navigate the worker to a specific product that needs to be picked, or assist in repairing a broken robot [46], while VR can be used to construct elaborate virtual warehouses with realistic simulations of worker interaction.

For experiments conducted within this thesis, we used Microsoft HoloLens 2.0 Development Edition AR glasses shown in Figure2.3. The device is equipped with a variety of sensors including as many as four tracking cameras and a time-of-flight (ToF) range camera. One of the main advantages of the selected device is the fact that, among other information, the poses estimated by the built-in tracking system can be accessed by the user. The mapping and tracking algorithms are based on Microsoft's proprietary software and their exact algorithms are not published [47], but their quality was tested and confirmed by Khoshelham et al. and Hübner et al. in [47, 48] while we conducted similar experiment within this thesis to attest to localization capabilities in the integrated warehouse environment. The VR engine we selected for the development of twin environments real warehouses, as well as larger



(a) One of twelve Flex 13 motion capture cameras used for dataset recording.

(b) The Optihub 2 - a USB hub connected to cameras, the other hub, and a PC collecting and broadcasting recorded data.

Figure 2.4: The Optitrack motion capture system we used to record the dataset consists of two Optihub 2 USB hubs connected to twelve Flex 13 motion-capture cameras that record motion in high speed and precision.

warehouse environments, was the Unity game engine [49]. The Unity game engine bolsters with seamless integration with the HoloLens device and ease of environment creation. Also, the existence of the ROS# [1] package enables the connection with the Robot Operating System [50] middleware used for the development of other components and modules in the integrated warehouse domain.

2.2.2 *Motion Capture System*

A motion capture (MoCap) is the process of recording movement, more specifically the position and orientation of interesting objects or subjects in the observed physical space [51]. While the roots of the technology come from gait analysis application [52], these systems are now used in a wide variety of other fields including computer graphics animation [53], robotics and military use [54]. The main principle of MoCap systems is the utilization of multiple sensors to recognize and track interesting objects in the scene. The authors in [51] list the most used approaches for capturing movement including:

- Optical-passive approach where reflective markers are tracked by multiple infrared cameras [55],
- Optical-active approach with infrared LED markers tracked by multiple infrared cameras [55],
- Tracking from video using dedicated software to detect and track the object motion in each frame [56].

[1] https://github.com/siemens/ros-sharp

The main advantage of using tracking directly from the video is smaller hardware dependency and potentially larger field of operation, but these methods lack the accuracy of optical-passive and optical-active methods [57, 58]. In the scope of this thesis we have used optical-passive OptiTrack solution shown in Figure2.4consisting of twelve Flex 13 motion capture cameras (Fig.2.4a) connected to two Optihub 2 USB hubs (Fig.2.4b). The passive markers are 1.5 mm diameter spheres defined as rigid bodies enabling fast 6 degrees of freedom (DoF) tracking. Additionally, in order to capture a human kinematic model imitating a human skeleton, the specialized motion capture suit can be used [10] paired with the Motive software[2]. This way orientations and positions of each human joint can be recorded in 120 Hz [10] and fed to the human action and prediction models proposed in the scope of this thesis.

2.3PATH PLANNING

As stated earlier, a crucial part of many models concerning human action prediction and motion planning, especially those based on inverse optimal control or Markovian models, is path planning. The reason for that is these models rely on comparing observed motion and behavior with a set of optimal or nearly-optimal behaviors with the respect to given hypotheses. In the domains concerned in this thesis, these behaviors consist of moving, looking to, or away from potentially interesting locations in space we call goals. If we are to ascertain whether or not an observed sequence pertains to a goal in the set of all possible goals, it is necessary to somehow quantify this movement. The intuition followed in the rest of the thesis is that humans behave nearly rationally, meaning they will look at the goal they desire and move towards it most of the time. The human movement is observed with the introduced sensors and is given as a three-dimensional point in the workspace. Since this workspace usually consists of obstacles, for example, warehouse racks and moving robots, it can not be navigated freely and those obstacles need to be avoided. Because of that, it is important to quantify how close is the observed human movement to optimal movement toward each of the pertaining goals. For this task, we rely on path-planning algorithms.

Generally speaking, path planning algorithms aim to find a sequence of valid configurations that moves the object from the source to the destination. In the human-robot shared environments, such configurations consist of unoccupied navigable free space. Because of that, the traversable areas of the warehouse or reachable elements of the collaborative environment are called *configuration space*. In the field of robotics, configuration space can range in complexity from a two-dimensional plane to an N-dimensional space [59] and can be severely constrained based on the robot configuration. Additionally, possible approaches for solving optimal paths depend on the criteria which include minimal distance, trajectory smoothness, maximum safety, and fastest calculation [60]. With these facts in mind, it's not surprising that the field of path planning in robotics is one of the most important and researched ones. Since we are using path planning mainly as a tool for comparing the observed motion with the optimal one, our focus will shift to the family of algorithms providing us with that information. Because of that we list and define several requirements all proposed algorithms must meet:

[2] https://www.optitrack.com/software/motive/

- Optimality - the search algorithm is considered optimal if every solution it founds is the best with respect to given criteria.
- Completeness - the search algorithm is considered complete if at least one solution exists then the algorithm is guaranteed to find a solution in a finite amount of time. While some algorithms like the rapidly-exploring random tree (RRT) [61] can find a path in free space fast, they are only asymptotically complete meaning they might not be able to find the path soon enough which leads us to the last requirement.
- Real-time operation - in the scope of this thesis, we will consider a method or algorithm to be real-time if it executes the expected operation in time that is smaller than the sample time of used sensors or the smallest time constant defining the expected system dynamic. To expand on the latter, while human behavior can change abruptly [62], it manifests much slower [63] than the usual sample time of utilized sensors. Because of that, we will use real-time loosely but expect all algorithms to perform with more than 10 Hz frequency.

In the rest of this chapter, we will present specifics of path-planning algorithms used in the scope of this thesis, lay out their pseudocode, and provide an in-depth analysis of their performance and utility.

2.3.1 *A* and D* graph search algorithms*

The A*, pronounced *A star*, search algorithm was authored by Hart and al. in [64] as a graph traversal algorithm with the application in optimal path planning. It is an extension of a well-known Dijkstra's algorithm [65] with the introduced heuristics as a guide to its search. The paradigm of the A* algorithm revolves around starting from the start node and iteratively adding its children to the *open list* while expanding on the node in the *open list* with the lowest current estimated cost to the goal node. In the robotics path planning domain, the graph is usually constructed from the perceived or previously known floor map. In the warehouse domain, such a map is constructed from floor plans, using a grid of regular squares with the wanted resolution. Each square containing walls, racks, or other untraversable elements is considered occupied while other squares are considered free. The introduced procedure yields an occupancy map of the environment. Depending on the capabilities of an agent moving in this environment, the configuration space is created from the occupancy map. Because humans can move in all directions, each tile is considered to have eight neighbors, one from each side and one in all diagonal directions. Once the neighboring tiles on the occupancy grid are defined, we can interpret them as graph vertices connected via graph edges, and the edge values are set to be Euclidean distance between vertice centers in Euclidean space. The pseudocode of the A* algorithm for finding the shortest path between any two nodes (vertices) in the graph is given in Algorithm1.

The main advantage of using the A* algorithm in the human-robot shared environment is its fast performance and ensured completeness. Depending on the selection of the heuristic function, the A* is also optimal. For omnidirectional agents, the heuristic function is usually selected to mimic the actual cost of moving with the Euclidean distance (L_2) as follows:

$$h_{x,y} = \sqrt{(x_a - x_g)^2 + (y_a - y_g)^2}, \tag{2.1}$$

Algorithm 1: The A* algorithm

Input: graph, startNode, goalNode
Output: path

```
1:  openList ← startNode
2:  closedList ← emptyList
3:  path ← emptyList
4:  while True do
5:      currentNode ← openList.first
6:      if currentNode is None then
7:          return None
8:      end if
9:      if currentNode == goalNode then
10:         return reconstructPath(currentNode)
11:     end if
12:     openList.removeFirst()
13:     closedList.append(currentNode)
14:     for neighbor in CurrentNode.neighbors do
15:         openList.add(neighbor)
16:     end for
17:     openList.sortInplace()
18: end while
```

while the less agile agents that can move only up, down, left, and right use more appropriate Manhattan distance (L_1) as follows:

$$h_{x,y} = |x_a - x_g| + |y_a - y_g|. \tag{2.2}$$

In the above equations we have denoted agent's position in xy coordinate system with (x_a, y_a) and goal position with (x_g, y_g). The sorting of the $openList$ in the Algorithm1is done based on each node's f value, obtained via:

$$f_{node} = g_{node} + h_{node} \tag{2.3}$$

where the g_{node} is the current consumed distance from the start node to the current node. Note that, in order for the A* algorithm to be optimal, the heuristic function has to be optimistic meaning it has to be greater or equal to the cost of the actual transition between two nodes. The main drawback of the A* algorithm in the human-robot shared environment is its space complexity because it needs to store all generated nodes in memory, sometimes multiple times if they are reached from the different parent nodes. On top of that, the A* algorithm thrives in static environments but is very sensitive to even a slight change in the floor plan as it leads to new edges appearing in the graph while moving robots remove them. The A* algorithm is not designed to handle such occurrences and the search has to start from scratch even if the modification to the original graph is just a minor one.

To accommodate this, Stentz proposed the D* algorithm in [66]. The algorithm's name is shortened from Dynamic A star and it was made for environments where a robot has

to navigate to given goal coordinates in unknown parts of the environment making assumptions about the unseen space. Usually, these assumptions are that this unseen space is navigable or free and the algorithm finds the shortest path under these assumptions, saving information about the cost of each vertex. Once the new information is available, this additional information is leveraged to replan fast depending on the new cost to reach each vertex. Unlike nodes in the A* Open List, which only save their perceived distance to the goal, the nodes in the D* Open List are marked as having one of several states:

- NEW - Meaning it is newly added in the Open List
- RAISE - Indicating that its cost is now higher than the last time it was in the OpenList
- LOWER - Indicating that its cost is now lower than the last time it was in the OpenList

Unlike the A* algorithm, the D* algorithm starts backward, from the goal node and updates states as the environment is being explored. This enables fast replanning in dynamic environments without sacrificing optimality or completeness. Finally, both of these algorithms return an optimal path with well-defined distances between its nodes. Because of that, it is possible to know the exact traversable distance between any two points in the configuration space which makes A* and D* algorithms suitable for our application.

2.3.2 *Markov Decision Process*

While the A* and D* algorithms enable fast searching of a graph, they require known costs of transition between adjacent graph nodes. As we already stated, this cost is usually set to be the distance between the two, although it can be modified to embed transition time information or fuel or battery consumption. The problem with this approach is inflexibility when it comes to modeling preference. Two paths can be optimized with respect to distance or travel time, but with significant features and differences. Consider a robot navigating in an environment full of cliffs and wet terrain. If possible, we would like to steer the robot away from such adversity sacrificing path optimality. Similarly, if we plan for humans in the integrated warehouse environment, there might exist certain areas a human would not prefer walking through, e.g. the area is noisy, not well lit or the passage is narrow. Handling this occurrence via graph search algorithms would require modification of cost towards the less desirable nodes. Since increasing these costs would render the commonly used Manhattan and Euclidean distance heuristics to be pessimistic, these algorithms would regress to basic Dijkstra's algorithm losing their speed and fast replanning ability. Furthermore, determining which nodes should be affected by adversary effect sometimes is not a trivial task, and the pathfinding problems would benefit greatly if only the source of adversity could be modeled with its effect naturally taking effect during the search.

The framework and mathematical model which can fit in with the previously mentioned requirement are a family of models called Markov decision processes (MDPs). Formally, MDPs constitute a mathematical framework that models a system taking a sequence of actions under uncertainty to maximize its total rewards [4]. More precisely, an MDP is a discrete time stochastic process represented as tuple $(\mathcal{S}, \mathcal{A}, T, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is set of states and $\mathcal{A}$ is set of actions. After an action $a \in \mathcal{A}$ is taken, system moves from the

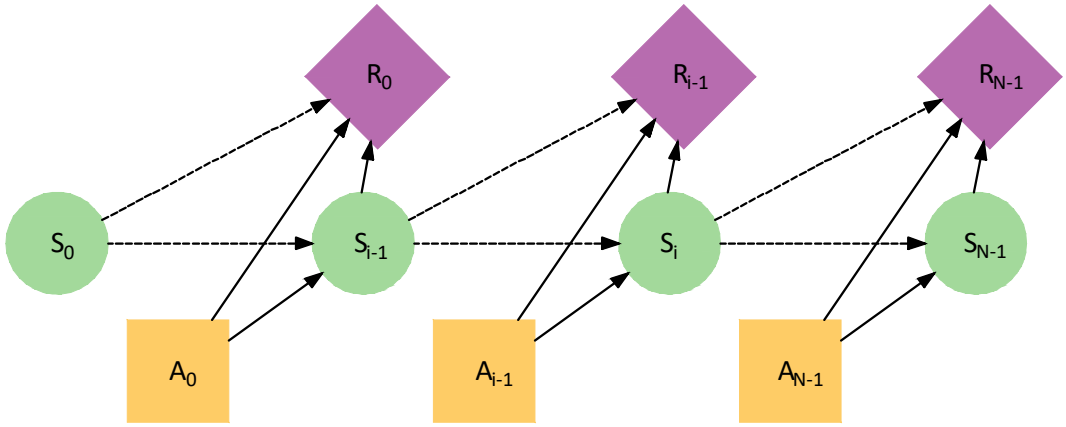


Figure 2.5: The Markov Decision Process framework consisting of states S, actions A that lead to probabilistic transition between them yielding rewards R.

current state $s \in \mathcal{S}$ to a new state $s' \in \mathcal{S}$. We define the conditional probability function $T(s,a,s') = p(s'|s,a)$ which gives the probability that the system lies in s' after taking the action a in state s, thus capturing system's uncertainty. Taking an action also yields an immediate reward $R(s,a)$ and the goal of the system is to choose the sequence of actions that maximizes the expected total reward $E\big(\sum_{t=0}^{\infty} \gamma^t R(s_t, a_T)\big)$. To prevent an infinite-horizon case where all positive rewards sum to infinity [67], one uses a discount factor $\gamma \in (0,1)$ which reflects the system preference of immediate rewards over future ones. In the path planning domain, it is common to set the value of the goal to a positive constant, associate adversary area or terrain with a large negative constant, and set all other fields to a small negative value, although the last step is not necessary if a nonzero discount factor is used. The overview of the MDP framework can be seen in Fig.2.5.

Once the appropriate MDP states, actions, transitions, and rewards are selected it is necessary to solve it. Solutions of MDP with finite state and action spaces may be found through a variety of methods, most notably revolving around dynamic programming. The two most common iterative ways of solving the MDP are the value iteration algorithm introduced by Bellman in [68] and the policy iteration algorithm introduced by Howard in [69]. They revolve around assuming the initial value or policy for each state and iteratively updating it propagating newly available information.

The value iteration algorithm is given in Algorithm2where Bellman's equation

$$V_{i+1}(s) = \max_a \sum_{s',r'} p_{s,s',a}(r_{s'} + \gamma V(s')) \tag{2.4}$$

is solved in iterations for all states until the maximum difference of previous and newly calculated state values $V_i(s)$ falls below threshold θ. This threshold is usually set to a small number and depending on its value the algorithm will have a trade-off between speed and precision towards the optimal policy π^*.

Unlike the value iteration algorithm the policy iteration algorithm puts emphasis on finding optimal policy instead of finding the most appropriate value for each state. It performs value iteration and then queries for change in all policies. If there is no change found, the algorithm is concluded and the optimal policy is considered to be found. These algorithms are useful for our problem of human motion valuation because they calculate

Algorithm 2: The value iteration algorithm

Input: θ, iteration threshold
Output: π, a policy

for s in S **do**
  $V(s) \leftarrow 0$
end for
$\Delta \leftarrow 0$
while $\Delta < \theta 0$ **do**
  for s in S **do**
    $v_{tmp} \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r'} p_{s,s',a}(r_{s'} + \gamma V(s'))$
    $\Delta \leftarrow max(\Delta, |v_{tmp} - V(s)|$
  end for
end while
for s in S **do**
  $\pi(s) \leftarrow \arg\max_a \sum_{s',r'} p_{s,s',a}(r_{s'} + \gamma V(s'))$
end for

state values $V(s)$. As the human agent moves, we are able to follow observed behavior and note consumed values of each state thus generating information as the agent moves towards or away from pertaining goal, while elegantly embedding unwanted areas and the probabilistic nature of the human movement.

2.4 Space Partitioning

In the previous section, we have described A* and D* graph search algorithms that excel at fast pathfinding and planning while maintaining completeness and optimality guarantees. While these algorithms serve their purpose in a plethora of applications, they still can not guarantee real-time performance in very large search spaces. As we already stated, the search space is a graph constructed from an occupancy grid map. If the map is really large, as modern robotized warehouses and other human-robot shared environments can be, the only way of reducing the size of a graph is to sacrifice grid density. However, there exists a certain limit to this procedure as reducing the density heavily impacts precision. For example, if a grid size is one square meter, the human could move one meter in any direction without their position being updated on the graph. This would lead to a slow reaction of the supervisory algorithm making the resulting human action and movement prediction useless. Because of that, it would be beneficial to enable search space reduction without great sacrifice to precision. One way to go about this problem is to leverage space partitioning.

Space partitioning is a geometrical term describing the process of dividing a space into two or more disjoint subsets - partitions. In other words, space partitioning divides space into non-overlapping regions. Depending on the task at hand, such procedures can create

Algorithm 3: The policy iteration algorithm

Input: θ, iteration threshold
Output: π, a policy

for s in S **do**
  $V(s) \leftarrow 0$
  $\pi(s) \leftarrow randomAction$
end for
while True **do**
  $\Delta \leftarrow 0$
  while $\Delta < \theta 0$ **do**
    for s in S **do**
      $v_{tmp} \leftarrow V(s)$
      $V(s) \leftarrow \max_a \sum_{s',r'} p_{s,s',a}(r_{s'} + \gamma V(s'))$
      $\Delta \leftarrow max(\Delta, |v_{tmp} - V(s)|$
    end for
  end while
  $policyStable \leftarrow$ True
  for s in S **do**
    $\pi_{tmp} \leftarrow \pi(s)$
    $\pi(s) \leftarrow \arg\max_a \sum_{s',r'} p_{s,s',a}(r_{s'} + \gamma V(s'))$
    if $\pi_{tmp}! = \pi(s)$ **then**
      $policyStable \leftarrow$ False
    end if
  end for
  if $policyStable$ **then**
    $return\ \pi$
  end if
end while

convex sets representing the space as trees [70] and can operate in multidimensional spaces creating point clouds [71]. For our application, we want to partition the space to create a graph of nodes human position in a two-dimensional space can be associated with. In order to solve this task, we select the Voronoi Diagrams.

2.4.1 *Generalized Voronoi Diagrams*

A Voronoi diagram partitions a plane into regions based on the distance to predefined points. The idea is that for each predefined point a corresponding region consisting of all the points closer to that point than to any other predefined point is found. More formally, the definition of the Voronoi region for a point $p_i \in P$ is given by:

$$V(p_i) = \{\vec{x} \mid ||\vec{x} - \vec{x}_i|| \leq ||\vec{x} - \vec{x}_j||, \forall_j \ni i \neq j\}, \tag{2.5}$$

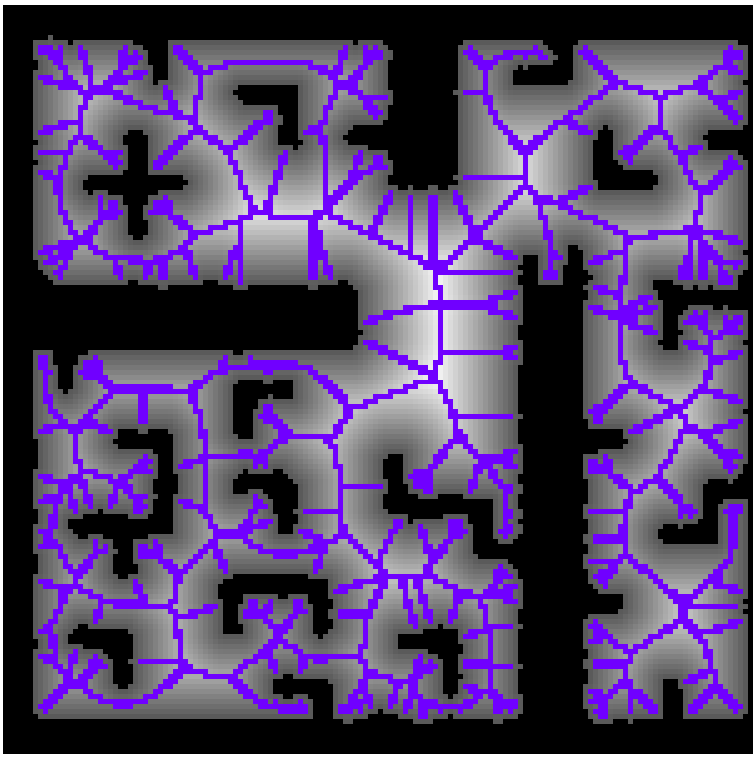


Figure 2.6: The example of a Generalized Voronoi Diagram rendered on a map. Blue lines denote the edges of the diagram and constitute points in free space most distant from the obstacle.

where $||\cdot||$ is usually the Euclidean distance [72]. We call the set given by $V = \{V(p_1), V(p_2), \ldots V(p_n)\}$ the Voronoi diagram of P. The question is, how to construct a Voronoi diagram in practice? The generalized Voronoi diagram (GVD) is a discrete form of the Voronoi diagram defined as the set of points in free space to which the two closest obstacles have the same distance [73]. In mobile robotics applications, the GVD can be constructed from an occupancy grid map of the environment which is usually obtained by mapping the environment with a mobile robot [14] or by parsing the existing floorplans. In this thesis, we will use the latter approach, since we have warehouse plans at our disposal. The example of generated GVD on a floor plan can be seen in Fig.2.6.

2.5PROBABILISTIC DECISION-MAKING METHODS

In this section, we will discuss the probabilistic decision-making methods used in the scope of this thesis. Let us first discuss what labeling a model as *probabilistic* actually means. One of the main challenges in modern artificial intelligence and robotics applications is the presence of uncertainty. This uncertainty manifests itself as the unpredictable environment, unreliable sensors, and models that are inherently inaccurate [14]. Contrary to relying on a single best guess of environment representation, probabilistic algorithms represent perceived information as probability distributions over the entire realm of possible - the probabilistic space.

Our work follows up on the Bayesian Theory of Mind (BToM) framework described in [74], where authors introduced a model for estimating hungry students' desires to eat at a particular food truck by observing its movement. Therein, authors argue that machines

lack the Theory of Mind – intuitive concept humans have about other people's mental state, and propose to emulate it by an intention recognition model based on Partially Observable Markov Decision Processes (POMDPs). Though impressive, the BToM model does not predict the possibility of a student's change of mind and does not ensure real-time operation in changing environments which is crucial in an integrated warehouse problem. Many models addressing the problem of human intention recognition successfully emulate human social intelligence using Markov decision processes. Examples of such models can be found in [4], where authors proposed a framework for estimating pedestrian intention to cross the road, and in [75], where authors proposed a framework for gesture recognition and robot-assisted coffee serving. There are also works from the gaming industry perspective, proposing methods for improving the non-playable character's assisting efficiency [76, 77]. A driver intention recognition problem was approached in [78] and intention estimation based on gaze data was introduced in [9]. Both of those approaches use learning methods for training the models which have been criticized by [79] emphasizing the drawback of using motion pattern learning techniques for trajectory estimation or intention recognition. Authors assert that such techniques operate offline and imply that at least one example of every possible motion pattern is contained in the learning data set which does not hold in practice. They propose using growing hidden Markov models (GHMM) for predicting human motion, a problem that we consider dual to the human intention estimation in the warehouse domain. GHMMs can be described as time-evolving HMMs with continuous observation variables where model structure and parameters are updated every time a new observation sequence is available. That kind of approach can be applied to the human intention recognition problem because it enables adding new goals during the experiment as well as an elegant framework for learning the model parameters online.

Assistive technology such as smart homes [80] and semi-autonomous wheelchairs [81] benefit also from precise human intention recognition. In [81], authors propose a POMDP-driven algorithm for wheelchair control taking into account the uncertainty of user's inputs because of, e.g., unsteady hands. The chair predicts the user's intention and autonomously enacts the intention with only minimal corrective input from the user. The authors also suggest that humans usually focus on moving from one spatial location to another, i.e., hallway to the kitchen, without worrying about the optimality of exact steps that come in between. In the present chapter, we build our model using similar assumptions about human spatial understanding. In [5] Anh and Pereira offer a thorough review of the human intention recognition area emphasizing its potential applications in decision-making theory. In recent years, human action prediction applications ranged from robotized warehouses [82, 83] to sedentary object-picking domain [84, 9, 85] and full-body motions [86, 87, 88]. State-of-the-art human action prediction frameworks are based on Markov models [89], inverse optimal control [90] or conditional random fields [91], which try to learn moving patterns with the respect to pertaining goals, usually assuming nearly-optimal human behavior in the observed sequences. In [92] the authors propose a hybrid deep neural network model for human action recognition using action bank features leveraging the fusion of homogeneous convolutional neural network (CNN) classifier. Input features are diversified and the authors propose varying the initialization of the weights of the neural network to ensure classifier diversity. Another approach based on the Long Short-Term

Memory networks (LSTMs) is proposed in [93] where the authors craft a two-stream attention-based architecture for action recognition in videos. They suggest that such an approach resolves the visual attention ignoring problem by using a correlation network layer that can identify the information loss on each timestamp for the entire video. Furthermore, in [94] authors leverage a bidirectional LSTM to learn the long-term dependencies, and use the attention mechanism to boost the performance and extract the additional high-level selective action-related patterns and cues. The convolutional LSTMS are used in [95] to handle the long-duration sequential features with different temporal context information and are compared to the fully connected LSTM. In [93] the authors propose an end-to-end two-stream attention-based LSTM network for human action recognition that selectively focuses on the effective features of the original input image. The concept of utilizing shared weights for neural networks was brought by de Ridder et al. in [96] with a focus on the feature extraction problem. This approach has gained traction in transfer learning [97] and physics simulation applications [98]. Regarding collaborative environments, the state-of-the-art models infer human actions by measuring different cues captured by wearable (eye gaze [99, 9, 8] or even heart rate and electroencephalography [100]) or non-wearable sensors. The use of non-wearable sensors such as motion capture systems or RGB cameras enables the model to capture crucial cues such as gestures [101], emotion [102], skeletal movement [103] or estimate eye gaze [11]. In works [8, 9, 10, 85, 104] authors have indicated that the eye gaze is a powerful predictor of human action. A good overview of human action prediction methods and their categorization by the type of problem formulation can be seen in [105]. Several works embed the eye gaze feature into human action prediction models using machine learning models such as support vector machine [9] or recurrent neural networks (RNNs) [10]. In the human collaborative scenario, the authors of [9] tested their algorithm relying on verbal instructions as additional features for their model and the actions form a sequence, In [85] the authors calculate the similarity between the hypothetical gaze points on the objects and the actual gaze points and use the nearest neighbor algorithm to classify the intended object. To the best of our knowledge, there does not exist a method that couples the human action prediction model with the directly measured eye gaze and human joint positions in a dynamic, changing environment. For example, in [9] the authors rely on gaze adding verbal commands in the feature space. In [85] the scenario is static and the subject sits while picking the objects who are always visible to the subject. Furthermore, in [105], the multiple-model estimator is leveraged for intention prediction, but the inputs to this model are extracted from a camera using convolutional networks and prior values that are not applicable in the dynamic collaborative domain.

This section introduces several probabilistic frameworks, namely Hidden Markov Model, Multilayer Perceptron, Recurrent Neural Networks, and Long Short-Term Memory Networks with an emphasis on their deployment as decision-making methods. These frameworks can be considered a mathematical attempt to reconstruct human decision-making and capture subtle cues that make human behavior. By putting the observed behavior and movement into a mathematical context it is possible to reverse-engineer the human way of thinking and thus make attempt at predicting the next movement, behavior, or end goal. Firstly, we will introduce the Hidden Markov Model, a Markov Decision Process extension that takes advantage of hidden states. It follows intuitions that not all states can be observed

directly, but need to be extrapolated from the observed behavior or emissions. As more information becomes available it is possible to reconstruct what these hidden states were and if we set up a problem in a way these hidden states match our hypotheses on human intentions and desires, the attempt to predict the following movement can be made. Furthermore, we will briefly expand on the mathematical background of Multilayer Perceptron, also known as a fully connected feedforward artificial neural network (ANN). This will lay the foundation to state-of-the-art recurrent frameworks, Recurrent Neural Networks, and Long Short-Term Memory Networks architectures. We will cover all aspects of these frameworks, from their formulation to training and testing, introducing statistical analysis along the way. Finally, we will define the signal correlation and discuss it as a part of the feature selection process.

2.5.1 *Hidden Markov Model*

The MDP model can be too restrictive to be applicable to many problems of interest [106] because it assumes that all states are fully observable. The hidden Markov model (HMM) is an MDP extension including the case where the observation is a probabilistic function of the state, i.e., the resulting model is a doubly embedded stochastic process with an underlying process that is not observable (it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observation. The overview of the HMM framework can be seen in Fig.2.7.

In general, when using HMMs we are interested in solving one of the following three problems. First, given an existing HMM and an observed sequence, we want to know the probability that the HMM could generate such a sequence (the scoring problem). Second, we want to know the optimal state sequence that the HMM would use to generate the sequence of such observations (the alignment problem). Third, given a large amount of data, we want to find the structure and parameters of the HMM that best account for the data (the training problem). In the scope of this thesis, we focus on the optimal state sequence problem, i.e., the alignment problem, for human movement with the aim of estimating their desires, intentions, and actions.

HMMs are especially known for their application in temporal pattern recognition such as speech, handwriting, gesture recognition [106] and force analysis [107]. There are several important implications that need to be taken into consideration when designing an HMM. It is necessary to carefully design state space and possible actions that lead to transition between them and this step heavily depends on the problem at hand. The next step is to select appropriate probabilities of transition between states and emission probabilities for each observation. If there is not enough data available, these numbers can be hand-picked or chosen by surveying experts and performing advanced analysis on obtained results. Alternatively, these weights can be learned from data utilizing the Baum–Welch algorithm [108], a special case of the expectation-maximization algorithm [109]. The algorithm leverages the synergy of the forward procedure which assigns probabilities of observing certain observations while being in a certain state and the backward procedure which calculates the probability of ending of partial sequence. The final step in the Baum-Welch algorithm is the update step where Bayes' theorem is used for calculating the new parameters of the HMM.

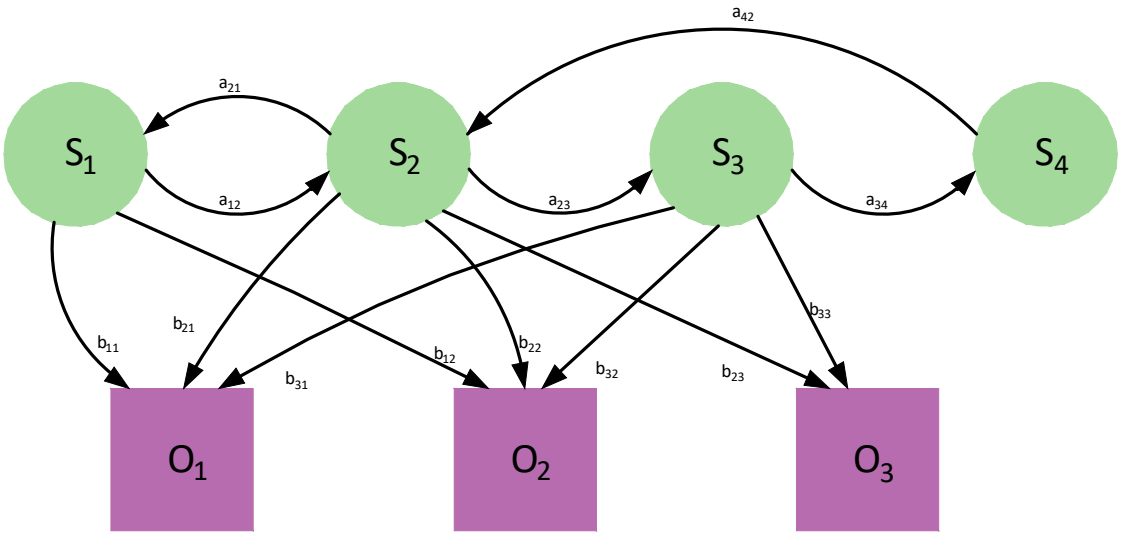


Figure 2.7: The Hidden Markov Model is an MDP framework extension where states are not directly observable. The observations O and emission probabilities B are used to reconstruct the most probable sequence of states S and transitions A.

These steps are repeated iteratively until the algorithm converges meaning the difference between two consecutive steps is lower than the predefined threshold.

When it comes to solving the HMM, the most commonly used algorithm is the Viterbi algorithm [110] - a recursive optimal solution to the problem of estimating the state sequence of a discrete-time finite-state (hidden) Markov process. It relies on principles of dynamic programming for obtaining the maximum a posteriori probability estimate of the most likely sequence of hidden states. Finding the most probable hidden state sequence solves the alignment problem of the HMM and yields the best guess of human underlying desires. The Viterbi algorithm relies on known HMM parameters and structure, a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{B})$, where $\mathcal{S}$, denotes states and $\mathcal{A}$, denotes action transition probability, which was labeled with $\mathcal{T}$ in the MDP framework. We use $\mathcal{O}$, to label a set of possible observations, and $\mathcal{B}$ is the emission matrix that gives probabilities for observing any given observation in any given state. Finally, the Viterbi algorithm solves for the most probable sequence of states, X, given observed sequence Y and initial guess Π. The Viterbi algorithm pseudocode is given in Algorithm4.

2.5.2 *Artificial Neural Networks*

Neural networks, also called Artificial Neural Networks (ANNs), are computational model that aims to mimic the functionality of biological neural networks commonly found in the brains of living beings. Inspired by the architecture of the human neural network, they are based on a collection of connected nodes called neurons. Each node in the network has assigned weights, real numbers, that multiply incoming signals whose sum is passed to the

Algorithm 4: The Viterbi algorithm

Input: $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{B}, \Pi, Y)$, HMM parameters
Output: X, most probable state sequence

$K \leftarrow length(S)$
$T \leftarrow length(Y)$
for state index i = 0, 1, ..., K **do**
  $T_1[i, 1] \leftarrow \Pi_i \cdot B_{iY_0}$
  $T_2[i, 1] \leftarrow 0$
end for
for observation j = 1, 2, ..., T **do**
  for state index i = 0, 1, ..., K **do**
    $T_1[i, j] \leftarrow \max_k(T_1[k, j-1] \cdot A_{ki} * B_{iY_j})$
    $T_2[i, j] \leftarrow \arg\max_k(T_1[k, j-1] \cdot A_{ki} * B_{iY_j})$
  end for
end for
$z_T \leftarrow \arg\max_k(T_1[k, T])$
$X_T \leftarrow S_{z_T}$
for observation j = T, $T-1$, ..., 1 **do**
  $z_{j-1} \leftarrow T_2[z_j, j])$
  $X_{j-1} \leftarrow S_{z_{j-1}}$
end for
$return\ X$

activation function thus creating the output of a node, as in:

$$o = f(\mathbf{x} \cdot \mathbf{w}) = f(\sum x_i w_i) \tag{2.6}$$

where o denotes neuron's output, $\mathbf{x}$ is the input vector, $\mathbf{w}$ is a vector containing neuron weights and f is the activation function commonly selected to be non-linear, e.g. rectified linear unit (ReLU) [111], scaled exponential linear unit (SELU) [112], hyperbolic tangent and binary step. The most typical way of organizing an ANN is by forming layers and literature distinguishes between three types of layers. The first layer is called the input layer and its function is to handle or aggregate the incoming signal. The last layer is called the output layer and is responsible for producing the final output of the model. All the layers between the input and output layers are called hidden layers and their structure and number are arbitrary and decided with respect to the task at hand. This structure, without any cyclic elements, is commonly called a feedforward neural network. The special case of a feedforward neural network where all nodes from one layer are connected to all nodes in the next layer is called Multilayer Perceptron (MLP).

The ANNs were introduced in the early 1940s by McCulloch and Pitts [113] who laid the mathematical foundation of the model even though the hardware capabilities were lacking at that time. With the improvement of processing units in the 1980s and utilizing graphical processing units in 2010 the ANNs surged as one of the most used models for solving prob-

lems in the broad scientific spectrum, ranging from control theory [114], protein structure prediction [115] to object detection [116] and motion prediction [117]. The cornerstone of the ANN model is a weight-tuning procedure, commonly called training of the network or a learning process. Generally, the aim of network training is to obtain parameters of each neuron that maximize the utility of the entire network with the respect to a given task. Training is performed on training data using learning algorithms that try to minimize the loss function that tries to mimic the error of the model. While it is common to split data into fixed train and test sets to showcase the value of an ANN, the obtained results can be biased towards the splitting point [118]. Because of that, it is common to use cross-validation, specifically the k-fold cross-validation process which is defined as a resampling method that uses different portions of the data to test and train a model on each of k iterations [119]. By comparing the results of the network trained on these portions, one can ascertain the statistical validity of results and gain better insight into the network's performance [120].

As the field of neural networks has seen rapid expansion in recent years, numerous modalities have been invented and deployed for solving complex tasks. The most dominant class of neural networks in the literature are Convolutional neural networks (CNNs) which adaptively learn spatial hierarchies of features, most often in computer vision applications [121]. The computer vision space has made use of Generative adversarial networks [122] where two networks are trained simultaneously with opposite goals. In the unsupervised learning domain, the autoencoder model [123] has gained traction for its ability to reduce the dimensionality of input features, commonly known as feature selection or extraction. The principle of the autoencoder model is to couple two elements, an encoder that is an MLP with each layer having fewer neurons than the previous one and a decoder that serves as the inverse of the encoder increasing number of neurons with each layer. The intuition behind such an approach is that the layer with the lowest number of neurons, called code, will contain only the crucial information thus weeding out noise and insignificant data.

We brought up the use of MLP models in computer vision, feature selection, and other domains. However, the main focus of this thesis is action and motion prediction which mandates information propagation with each time step. For this task, we select Recurrent Neural Networks (RNN) and Long Short-Term Memory Networks (LSTM) that will be presented in the following subsections.

2.5.3 *Recurrent Neural Networks*

An RNN, introduced by Rumelhart et al. [124], is a neural network that consists of a hidden state $\boldsymbol{h}_t$ which is connected to its output $\boldsymbol{y}_t$ as well as previous hidden state $\boldsymbol{h}_{t-1}$. This property enables it to capture a temporal dynamic behavior of the process and propagate the information through time because its output depends both on the input at a given time step as well as on the hidden state at the previous time step. Formally, a basic RNN can be described by:

$$\begin{aligned} \boldsymbol{h}_t &= \sigma(\boldsymbol{U}_h \boldsymbol{h}_{t-1} + \boldsymbol{W}_h \boldsymbol{x}_t + \boldsymbol{b}_h) \\ \boldsymbol{y}_t &= \sigma(\boldsymbol{W}_y \boldsymbol{h}_t + \boldsymbol{b}_y) \end{aligned} \tag{2.7}$$

where σ is the sigmoid, the most commonly used activation function, and $\boldsymbol{x}_t$ is the network input. The next hidden state is calculated using hidden weights $\boldsymbol{U}_h$ and input weights $\boldsymbol{W}_h$

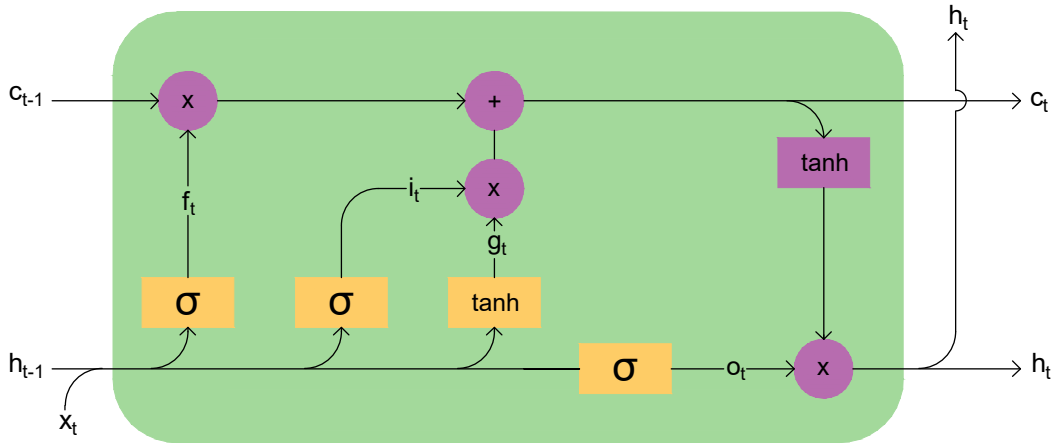


Figure 2.8: A Recurrent Neural Network architecture. The output at each time step is calculated using network weights, the current input, and previous hidden state - a memory unit. The hidden state is then updated for the next time step.

while output weights $\boldsymbol{W}_y$ are used for calculating the output. The model also incorporates hidden layer and output bias - $\boldsymbol{b}_h$ and $\boldsymbol{b}_y$. RNN networks have been successfully used in a plethora of time-series prediction problems, including action sequence prediction [86], machine translation [125] and outlier detection [126]. The architecture of the RNN is shown in Figure2.8. The main deficiency of the RNN model is the vanishing and exploding gradient problems for longer sequences because the hidden state is updated with multiplication for each observation. This phenomenon leads to either loss of learning ability or a requirement for a large number of data making the RNN model unfeasible for some applications. Also, since the hidden state is updated rather simply, it is hard to encode the temporal aspect of dependencies. For example, in the translation domain it is important to know what is the subject of the sentence, and introducing the new subject overrides the old one. As there is no way to forget the old knowledge in the RNN architecture, it is unfit for problems that have both long and short-term dependencies.

2.5.4 *Long Short-Term Memory Networks*

The LSTM networks, introduced by Hochreiter et al. [127], is a derivative of RNN networks that supplements the RNN's hidden state with the more advanced cell state controlled by gates, formally:

$$\begin{aligned}
\boldsymbol{f}_t &= \sigma(\boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{b}_f) \\
\boldsymbol{i}_t &= \sigma(\boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{b}_i) \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{b}_o) \\
\boldsymbol{g}_t &= \tanh(\boldsymbol{U}_g \boldsymbol{h}_{t-1} + \boldsymbol{W}_g \boldsymbol{x}_t + \boldsymbol{b}_g) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \circ \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \circ \boldsymbol{g}_t \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \circ \sigma(\boldsymbol{c}_t) \\
\boldsymbol{y}_t &= \sigma(\boldsymbol{W}_y \boldsymbol{h}_t + \boldsymbol{b}_y)
\end{aligned} \tag{2.8}$$

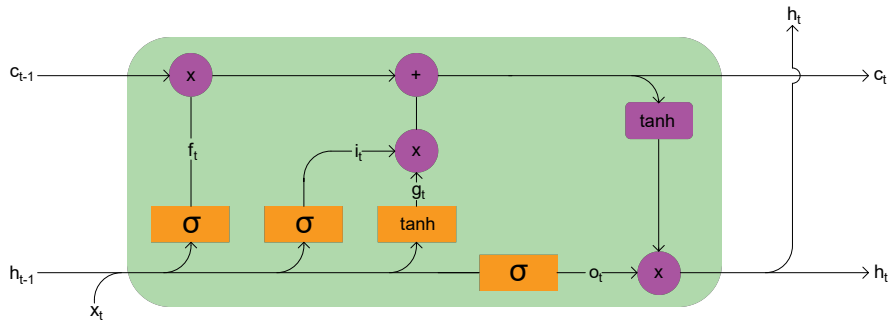


Figure 2.9: A Long Short-Term Memory Network unit. Component-wise operations are colored purple and network layers are colored yellow. The hidden state at each time step is used to calculate a current network output.

where tanh is the hyperbolic tangent function and the operator $\circ$ is the Hadamard product. The cornerstone of an LSTM model is the *cell state* $\boldsymbol{c}_t$ that propagates information through time. Every iteration of the LSTM network first forgets irrelevant information in $\boldsymbol{c}_t$ using the *forget gate* $\boldsymbol{f_t}$, and then adds new information with the *input gate* $\boldsymbol{i_t}$. The $\boldsymbol{c}_t$ is then used for future iterations of the LSTM network as well as for updating current *hidden state* $\boldsymbol{h}_t$ using the *output gate* $\boldsymbol{o_t}$. One can find examples in the literature where the activation function for the hidden state update is changed to a hyperbolic tangent [128]. Finally, the output of any given iteration is calculated in the same manner as in the RNN model, by multiplying the hidden state with the corresponding network weights. LSTM networks have seen applications in similar problems as RNNs, such as pedestrian trajectory prediction [129], and we have selected them as the backbone network in the scope of this thesis due to their ability to capture long-term dependencies.

2.5.5 *Correlation Signal Analysis*

So far in this section, we have highlighted probabilistic decision-making methods used for human action in motion prediction in the scope of this thesis. We have also put emphasis on the importance of real-time operation these models and frameworks have to satisfy. Both RNN and LSTM are deep methods, meaning they can employ multiple layers in their architecture, and the number of parameters increases linearly with respect to the input dimension [130]. The parameter number directly influences the capability of the network where too few parameters risk inability to capture all complex cues and patterns from data, called underfitting. On the other hand, too many parameters on a limited amount of data can lead to the overfitting of the network [131] that manifests as a great performance on train data but subpar performance on the test data. Having this in mind, a common approach in the literature is to try to reduce the dimensionality of input data to reduce the number of network parameters while retaining accuracy. This process is called feature selection if the set of reduced features is a subset of original features, while feature extraction is the process through a set of new features is created [132].

Earlier in this chapter, we introduced autoencoders as a data-driven MLP solution for feature extraction. Another data-driven feature extraction method that gained traction in the literature is principal component analysis (PCA) [133] which relies on decomposing

feature space using N orthogonal axes called components. In the scope of this thesis, apart from an autoencoder-inspired solution, we have employed a signal processing-based feature selection method relying on the correlation of input signals [134]. The correlation of two time-series signals is defined as Pearson coefficient [135] calculated with:

$$\rho_{X,Y} = \frac{cov(X, Y)}{\sigma_X \sigma_Y} \tag{2.9}$$

Where cov denotes covariance calculated with:

$$cov(X, Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)] \tag{2.10}$$

with μ being mean values, σ standard deviations of signals and $\mathbb{E}$ is the expected value. The intuition behind the use of correlation for feature selection is that signals with high correlation have high mutual information and can be represented with only one signal from the correlating group of signals discarding redundant ones. This approach was first proposed by Hall et al. in [134] as used in [136] where authors leveraged the Pearson coefficient to select features for daily activities in smart homes and in [137] where authors introduced a fast correlation-based filter solution.

2.6 Summary

In this chapter, we have laid out the foundation for understanding the contributions of this thesis. We have discussed sensors in the field of robotics with a focus on observing and tracking human position and orientation in shared spaces. For that task, we have selected AR, VR, and motion capture whose principles of operation were presented. Thereafter, we shifted to the mathematical background of utilized models and frameworks. First, we dived into path planning methods, discussing A$*$ and D$*$ graph search algorithms, their properties, and their use in future work. We put emphasis on optimality, completeness, and real-time operation and discussed the GVD method for free space partitioning. Furthermore, the MDP framework was introduced and iterative methods for solving it were presented. Afterward, several probabilistic decision-making methods were introduced, beginning with the HMM framework and the Viterbi algorithm. We described neural networks and gave a brief overview of crucial concepts such as layers, training, testing, and cross-validation. Since the task at hand is human action and motion prediction, we scrutinized state-of-the-art temporal neural network models, the RNN and LSTM. Finally, special emphasis was put on feature selection methods, namely autoencoder architecture and correlation.

3

Human action prediction in integrated warehouse systems

This chapter deals with the problem of human action prediction in integrated robotized warehouses, in the literature commonly known as human intention recognition problem, and we will refer to it as such throughout the chapter. Specifically, we will put the emphasis on ascertaining the final goal of human movement. The first step in our solution is creating an integrated warehouse simulator with three potential goals. The uncertainty in the human movement was modeled with a Markov decision process with special care in state reward calculation based on graph search. This approach was modified for a real warehouse environment in order to account for mobile robots that change the floor plan thus influencing optimal paths towards each goal. A real-time human action validation algorithm based on generalized Voronoi diagram space partitioning and graph search is proposed. The data is collected using augmented reality glasses and within a virtual reality warehouse twin. Also, a larger virtual reality warehouse is modeled to showcase the scalability of the proposed method. The results of human action validation are processed by a human intention estimation algorithm based on a hidden Markov model. This model is carefully crafted to account for uncertainty and the presence of an unknown alternative goal. Finally, at the end of the chapter, the proposed framework is thoroughly tested and the results are interpreted.

3.1 Introduction

Theory of Mind (ToM) is the intuitive grasp that humans have of their own and other people's mental states, how they are structured, how they relate to the world, and how they cause behavior [74]. Human beings understand that others have certain desires and that those desires guide them to act using the means most likely to achieve them. However, *explanation by rationalization* reasoning, which ToM assumes, is highly contextually dependent, and translating such a causal behavior model to machines is not an easy task. Having that in mind, we limit the proposed model to the problem of estimating the intention of a human worker in a highly flexible robotized warehouse. The worker can perform tasks such as maintaining the robots or picking the items from the racks containing goods. We assume that there is a finite number of possible goal locations which are usually in front of the *interesting* racks and that at least one goal is known before the start of the experiment.

Furthermore, we also assume that the position and orientation of the worker are measured. This can be achieved by, e.g., augmented reality glasses, other types of wearable sensors, or specialty vests equipped with vision sensors as developed in the scope of the project SafeLog [138]. Dennett [6] has proposed that social reasoning abilities rely on *intentional stance*, i.e., the assumption that agents have beliefs about the world and their situation in it and will behave rationally to achieve their desires. We argue that, in the warehouse domain, rationally behaving with respect to the desired goal manifests as moving towards that goal's location, and that since our agents are workers, trained professionals, they are highly likely to behave rationally within this context.

In order to determine the most likely goal the worker is moving to, we need to apply a complete and globally optimal path planning algorithm and compare the worker's motion online with the algorithm output (details are discussed later). We assert that a worker following approximately a globally optimal path is a reasonable assumption since the worker is acquainted with the warehouse layout and will plan its motion in accordance with it. Given that, we find that for the problem at hand, where action uncertainties are reduced, frameworks such as POMDPs used in [74] are not necessary. Moreover, the planning algorithm must be able to quickly replan the path with the appearance of moving obstacles such as mobile robots carrying the racks. Having that in mind, in this chapter we will present a novel human intention recognition solution based on HMM. Firstly, we will introduce our intention recognition warehouse simulator in Section3.2where we crafted a novel heuristics for the A* path planning search in order to get rewards for each state. These states are evaluated as part of the proposed MDP framework generating worker action validation. Alternatively, for our real and VR warehouse, in Section3.3partitioned the space using GVD and associated worker's position with visible nodes of the graph. Using these nodes and the D^* graph search, we seamlessly assess is the worker moving towards each of the predefined goals. Both of these action validation methods are used as a front-end solution for our back end, the HMM decision-making process introduced in Section3.4that serves as the intention estimator. Finally, the proposed architecture is validated for simulated, real, and VR warehouses in Section3.5. Finally, the Section3.6summarizes the chapter.

3.2INTENTION RECOGNITION IN SIMULATED ENVIRONMENT

This section presents a front-end solution for the human intention recognition problem. We will describe the intention recognition simulator created for this purpose and explain details of how human actions and movements are observed and interpreted. Finally, we will lay out our method for human action validation thus quantifying is the human going towards any predefined goals. Because in the integrated warehouse environment the worker's position and orientation need to be precisely estimated, for the rest of this Section we assume that these quantities are readily available. Furthermore, we would like to emphasize that most warehouse worker duties, such as sorting and placing materials or items on racks, unpacking, and packing, include a lot of motion which is considered to be inherently stochastic. Therefore, we model the worker's perception model $P(O|S)$, with O and S representing observation and state, respectively, as deterministic, and the action model $P(S'|S, A)$, with A representing action, as stochastic. A paradigm that encompasses uncertainty in an agent's

motion and is suitable for the problem at hand is MDPs [67].

The MDP framework is based on the idea that transiting to state S yields an immediate reward R. Desirable states, such as warehouse items worker needs to pick up, have high immediate reward values, while undesirable states, such as warehouse parts that are of no immediate interest, have low immediate reward values. The rational worker will always take actions that will lead to the highest total expected reward and that value needs to be calculated for each state. Before approaching that problem, we define the MDP framework

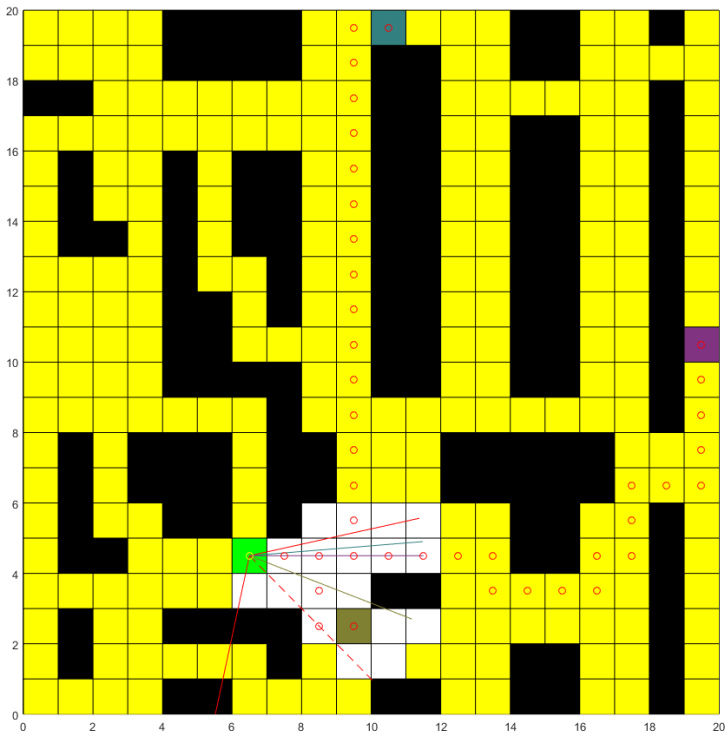


Figure 3.1: Agent (green tile) in simulation environment with three potential agent's goals (colored tiles). Unoccupied space is labeled with yellow tiles and occupied space (i.e. warehouse racks) is labeled with black tiles. The optimal path to each goal is shown with red dots and red solid lines denoting the agent's vision field. Visible tiles are colored white. Red dashed line denotes the agent's current orientation and colored lines indicate the direction of the average orientation of the visible optimal path to each goal calculated using (3.2).

applicable to the warehouse domain. In order to accomplish that, we have placed the worker (later referred to as an agent) in a simulated 2D warehouse environment shown in Figure3.1. The environment is constructed using MATLAB® GUI development environment without predefined physical interpretation of the map tiles and the map size is chosen arbitrarily to be 20×20. There are three potential goals and the shortest path to each goal is calculated.

3.2.1 *Modified A-star heuristics*

There are many off-the-shelf graph search algorithms we can use to find the optimal path to each goal, such as Dijkstra's algorithm and A*. However, if there exist multiple optimal paths to the goal, there are no predefined rules which one to select and the selection depends on the implementation details.

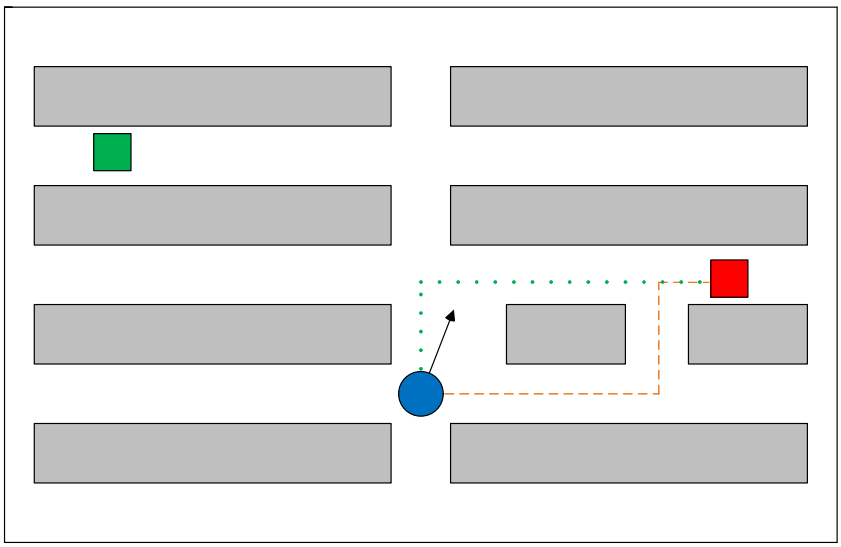
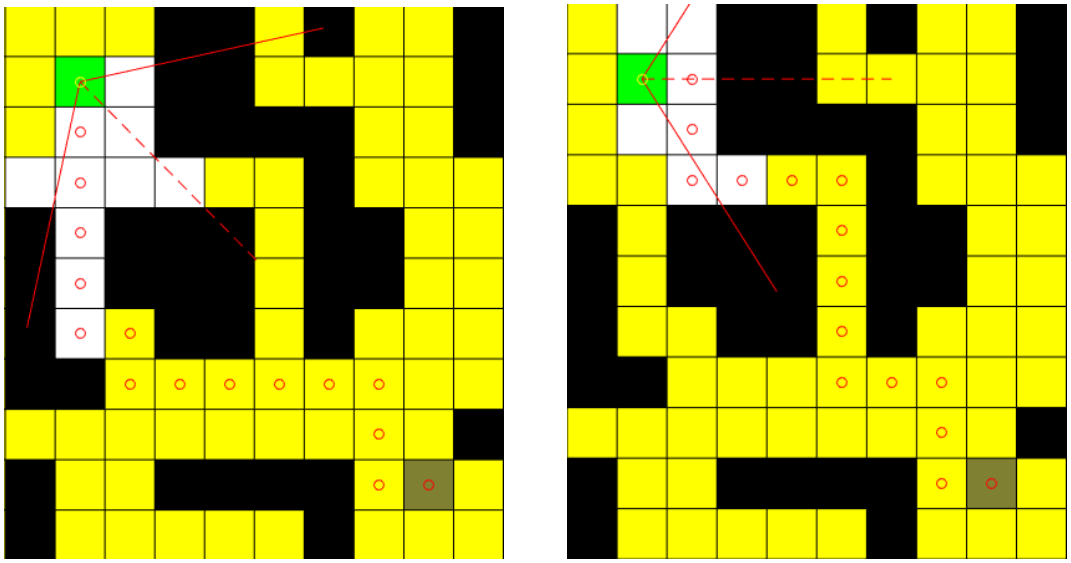


Figure 3.2: Warehouse worker's (blue circle) shortest path to the red goal is ambiguous because both orange dashed and green dotted paths are optimal. The black arrow denotes worker's orientation.

Consider the example scenario with the warehouse worker in Figure3.2. It is intuitive that the rational worker will tend to follow the green path because the orange path would require the worker to either take the additional action of turning or unnatural walking by not looking forward. Having this in mind, we modify the commonly used Manhattan A*



(a) (b)

Figure 3.3: The proposed A* modification yields different optimal paths with the agent's orientation change.

heuristics ensuring that the algorithm always selects the optimal path the agent currently sees the most. This has been done by introducing the heuristic matrix H using the Manhattan

distance (L_1) heuristics as follows:

$$H_{x,y} = \begin{cases} |x_g - x| + |y_g - y| - \epsilon, & \text{if the agent sees tile (x,y)} \\ |x_g - x| + |y_g - y|, & \text{otherwise} \end{cases} \tag{3.1}$$

where ϵ is a small value. Subtracting a small value from the visible tiles directs the search in their direction and does not disrupt the heuristic's admissibility. The cost of each movement is also modified in a the similar way by subtracting a small value ϵ from the base movement cost of 1, if the tile is visible. An example of the modified A* search algorithm results can be seen in Figure3.3. The average orientation of the visible path to each goal is defined as follows:

$$\theta_{goal} = \begin{cases} \text{atan2}\Big(\sum_{n=1}^{N} \sin(\theta_n), \sum_{n=1}^{N} \cos(\theta_n)\Big), & \text{if } N > 0 \\ \theta_a + \pi, & \text{otherwise} \end{cases} \tag{3.2}$$

where N is the number of visible optimal path tiles, θ_a is agent's orientation and θ_n are relative orientations of each visible optimal path tile (x, y) with respect to the agent (x_a, y_a):

$$\theta_n = \text{atan2}(y - y_a, x - x_a). \tag{3.3}$$

3.2.2 *Action validation*

We propose a mathematical model for validating an agent's actions based on the assumption that the rational agent tends to (i) move towards the goal it desires most by taking the shortest possible path, and (ii) orients in a way to minimize the difference between its orientation and the most desirable goal's average orientation of the visible optimal path calculated in (3.2). The proposed model goal is to assign a large value to the actions compatible with the mentioned assumptions, and small values to the actions deviating from them. These values will be used to develop the agent's intention recognition algorithm in the sequel. We can notice that the introduced validation problem is actually a path planning optimization problem. A perfectly rational agent will always choose the action with the greatest value and consequently move towards the goal. We approach the agent's action values calculation by introducing the agent's action validation MDP framework. We assume that the agent's position and orientation are fully observable and create the MDP state space S as:

$$S_{x,y,k} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}. \tag{3.4}$$

The agent's orientation space Θ must be discrete because the MDP framework assumes a finite number of states. We have modeled Θ to include orientations divisible with $\frac{\pi}{4}$ and it can be arbitrary expanded:

$$\Theta = \{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{7\pi}{4}\}. \tag{3.5}$$

The action space A includes actions 'Up', 'Down', 'Left', 'Right', 'Turn Clockwise', 'Turn Counterclockwise' and 'Stay', labeled in order as follows:

$$A = (\wedge, \vee, <, >, R, L, S). \tag{3.6}$$

It has already been stated that the agent's actions are fully observable but stochastic. In order to capture the stochastic nature of the agent's movement, we define the transition matrix T of agent's movement:

$$T = \begin{bmatrix} 1-2\epsilon & 0 & \epsilon & \epsilon & 0 & 0 & 0 \\ 0 & 1-2\epsilon & \epsilon & \epsilon & 0 & 0 & 0 \\ \epsilon & \epsilon & 1-2\epsilon & 0 & 0 & 0 & 0 \\ \epsilon & \epsilon & 0 & 1-2\epsilon & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1-\epsilon & 0 & \epsilon \\ 0 & 0 & 0 & 0 & 0 & 1-\epsilon & \epsilon \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

where element T_{ij} denotes realization probability of the action A_j, if the wanted action is A_i. Moving actions have small probability 2ϵ of resulting in lateral movement, and turning actions have small probability ϵ of failing. The value of the constant ϵ is 0.1 which we obtained experimentally. If the agent's action cannot be completed, because of the occupied space blocking the way, the column responding to the impossible action is added to the last column and is set to zero vector afterward. We define three hypotheses, H_i, $i = 1 \ldots 3$, one for each goal state as follows: *"Agent wants to go to the goal i and other potential goals are treated as unoccupied tiles"*. The immediate reward values R for each hypothesis and state are calculated as follows:

$$R_{i,S'} = \begin{cases} \pi, & \text{if } S\text{'} \text{ is the goal state according to the } H_i \\ -(\epsilon + |\theta_i - \theta_a|), & \text{otherwise} \end{cases} \tag{3.8}$$

where ϵ is a small number and $|\theta_i - \theta_a|$ represents the absolute difference between the average orientation of the visible path to the goal i and the agent's orientation. Note that we have taken the angle periodicity into account while calculating the angle difference in (3.8). The goal state is rewarded and other states are punished proportionally to the orientation difference. If the agent does not see the path to the goal i, the reward is set to the lowest value, $-\pi$ which is derived from (3.3). One of the most commonly used algorithms for solving the MPD optimal policy problem is the value iteration algorithm [4], which assigns a calculated value to each state. The optimal policy is derived by choosing the actions with the largest expected value gain. The value iteration algorithm iteratively solves the Bellman's equation [68] for each hypothesis H_i:

$$V_{j+1}(H_i, S) = \max_a \{ \sum_{S'} P_{S,S'} (R_{H_i,S'} + \gamma V_j(H_i, S')) \} \tag{3.9}$$

where S is the current state, S' adjacent state, and $P_{S,S'}$ element of the row T_a in transition matrix T which would cause transitioning from state S to S'. The algorithm stops once the criteria:

$$\sum_{i,k} ||V_j(H_i, S_k) - V_{j-1}(H_i, S_k)|| < \eta \tag{3.10}$$

is met, where the threshold η is set to 0.01. State values, if the goal state is the dark yellow (southern) goal and agent's orientation of $\frac{3\pi}{2}$, is shown in Figure3.4. The agent's behavior consistent with the hypothesis H_i is defined as follows.

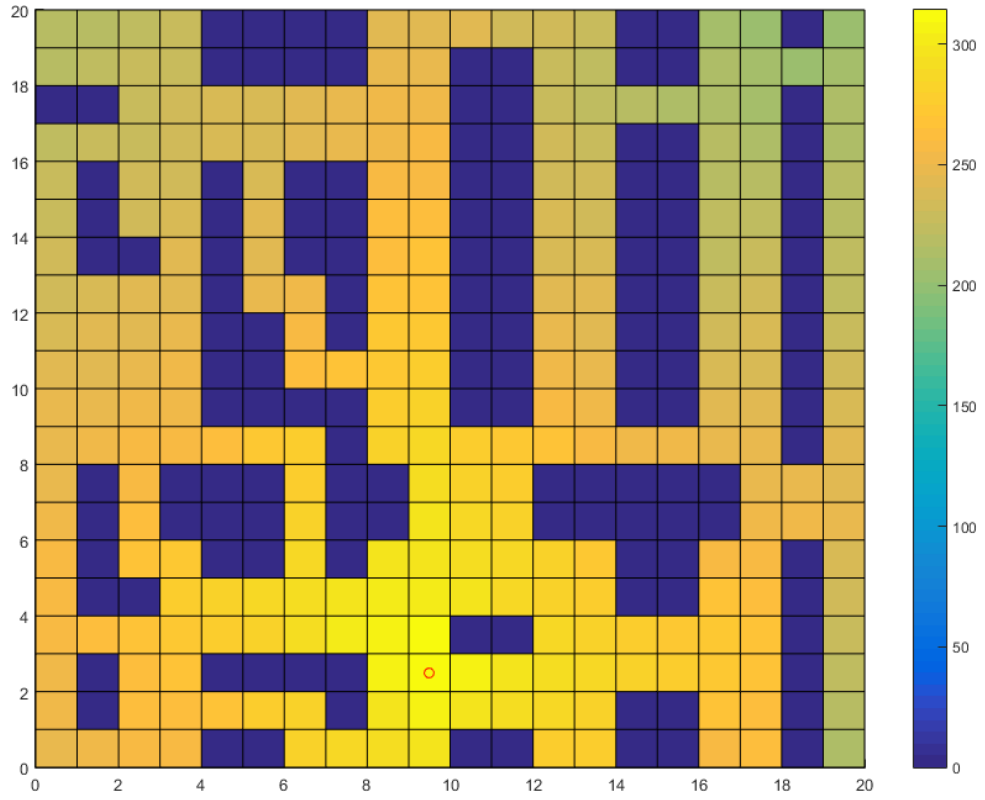


Figure 3.4: State values for the agent's orientation of $\frac{3\pi}{2}$ if the goal state is the southern goal labeled with the red circle.

Definition 3.1. *(Consistent behavior) If the agent is in the state S and takes the action a under the hypothesis H_i, with the expected value gain greater or equal than the expected value gain of the action "Stay", its behavior is considered consistent with the hypothesis H_i. Otherwise, its behavior is considered inconsistent with the hypothesis H_i.*

Behavior consistency is an important factor in determining an agent's rationality, which will be further discussed in the next section. While calculating the immediate rewards and state values has $\mathcal{O}(n^4)$ complexity and can be time-consuming, it can be done offline, before the simulation start. Optimal action, $\Pi^*(H_i, S)$, for each state is the action that maximizes expected value gain and, on the other hand, the worst possible action, $\bar{\Pi}^*(H_i, S)$, is the action that minimizes expected value gain.

3.3 Warehouse worker action validation

Planning using MDP solvers [67] discussed in the previous section and in [139] lacks the ability to quickly replan, since the optimal policy for efficient intention recognition must be computed offline for realistic warehouses. Given that, for the larger and dynamic warehouse, we choose to use the D* algorithm [66] for finding the globally optimal path to the goals. However, having in mind that the modern warehouses are growing in size we aim to reduce the complexity of mentioned search problem. One approach to alleviating the complexity is to reduce the precision of the warehouse occupancy grid representation, thus reducing the D* algorithm search space; however, we assert that it could jeopardize the proposed human

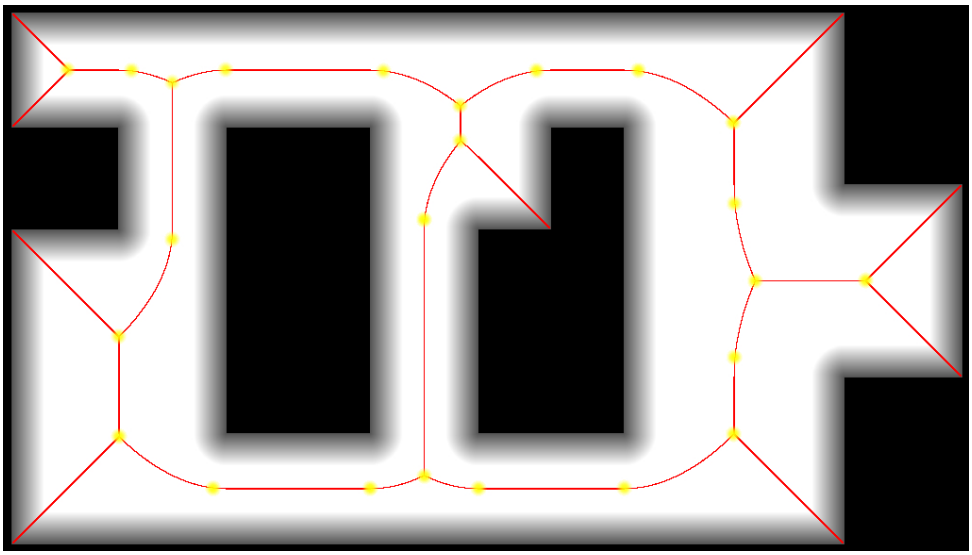


Figure 3.5: Generalized Voronoi Diagram (red) of Swisslog's warehouse in Ettlingen with highlighted graph's nodes (yellow). Untraversable parts of the warehouse such as walls or racks are denoted with black color. Shades of grey denote the distance of the traversable part of the warehouse from the obstacles. The distance is used for GVD generation.

intention recognition performance and cannot be applied to an arbitrary large warehouse, thus directly impacting the proposed algorithm's ability to generalize.

To solve the aforementioned challenges, we propose to use generalized Voronoi diagrams (GVDs) for reducing the search space without losing valuable precision for intention recognition. The motivation for using GVDs in our work is manifold. First, partitioning the plane using GVDs allows us to limit the search space on the Voronoi graph nodes, thus greatly reducing the search time. Second, moving along the edges of a Voronoi graph ensures the greatest possible clearance when passing between obstacles. This property resembles assumed human path planning in a warehouse application because human beings are generally not prone to walking in the proximity of warehouse racks (note that in our example robots can also pass under the racks). Finally, moving obstacles, such as mobile robots, in the flexible warehouse systems can obstruct a worker's path necessitating replanning of the optimal path towards that goal. The replanning can easily be achieved using graph search algorithms by discarding the edge of the Voronoi graph the robot is currently occupying. Using such algorithms on the whole occupancy grid would not be possible in real-time because of the search space's size.

On the other hand, we can imagine a scenario where new possible passages could appear. This could happen when a mobile robot takes a rack and frees up space in the middle of the rack block. In order to handle that event, a new GVD would have to be generated from the occupancy grid map of the warehouse. This could impede the online application of the proposed approach; however, in order for that to happen, since usually, one rack block has two columns of racks, multiple racks close to each other would have to be carried away at similar time intervals. Furthermore, we assert that workers in robotized warehouses would generally not be allowed to use such passages for safety reasons. Having that in mind, from the methodological perspective of the approach, we propose to use the D* algorithm on

GVD nodes for worker path planning. In the sequel, we describe the proposed human intention recognition algorithm in detail and divide it into three parts: the creation of the warehouse GVD (offline), worker action validation, and worker intention estimation (both online).

We have already emphasized the necessity of performing human intention recognition online. In order to ensure online operation, we need to perform time-consuming parts of the algorithm before the start of the experiment. First, we generate the GVD [140] of the warehouse using its floorplan. An example of such a graph generated on Swisslog's warehouse using an occupancy grid with a cell size of 5 mm can be seen in Figure3.5. After generating GVD, we select all of the graph nodes for further processing. It is worth noting that it is possible to insert additional nodes at arbitrary locations, but erasing generated ones except dead ends is not allowed because it would impede the graph's connectivity. Also, at least one goal node must be added before the experiment starts, since it is not possible to emulate ToM inference without having any hypotheses about possible goal locations. In the scope of this work, we add only the goal nodes and discard the dead ends creating a node-set in which we denote $\mathcal{N}$. With the obtained node-set, we run the D* algorithm to find the optimal path between every two nodes and save all the relative distances in a matrix $\mathbf{F}$, where element $\mathbf{F}_{i,j}$ denotes distance in pixels between nodes i and j. This might not be the optimal approach to finding relative distances between the nodes, but this part is performed offline and done only once before the start of the experiment.

3.3.1 *Worker action validation*

During the online phase, we monitor workers' position and orientation provided by the Microsoft HoloLens augmented reality device as well as the positions of mobile robots (provided by the warehouse fleet management system). Mobile robots are treated as moving obstacles with a radius $r = 1\,\text{m}$, and the worker's wearable device shows the positions of nearby robots. If the robot is located on a GVD edge, we cut that edge from the graph and update the relative distance matrix $\mathbf{F}$ using the D* algorithm. The final objective of

Algorithm 5: Human intention recognition

```
while True do
    if Worker moved or turned significantly then
        d ← Modulated distance to every goal
        for Proximate positions and orientations do
            D_i ← Modulated distance to every goal
        end for
        Update intention estimation(d, Di)
    end if
end while
```

the human intention algorithm is to estimate towards which goal the worker is currently going by observing worker motion and comparing it to the optimal path to each of the goals. With each worker's position and orientation information update, we check if (i)

the worker position mapped to the occupancy grid floor plan has changed or if (ii) the worker orientation has changed more than $\frac{\pi}{8}$ from the last intention estimation update. If any of these conditions are met, we perform an intention estimation update by associating observed worker position and orientation with each of the nodes in set $\mathcal{N}$ using vector $\mathbf{c}$ as follows:

$$c_i = \begin{cases} 0, & \text{an unobstructed straight line exists} \\ & \text{between the worker and the } i\text{-th node} \\ \mathcal{G}(d_i, \sigma^2) \cdot \Phi(\theta_i), & \text{otherwise.} \end{cases} \tag{3.11}$$

In (3.11) $\mathcal{G}$ is a Gaussian function with variance $\sigma^2 = 0.005$, which we obtained experimentally, and d_i is distance between worker position and i-th node's location. Bell-shaped functions such as Gaussians have been used for navigating in continuous spaces [141], which motivated us to choose them as a proximity measure. They are smooth and monotonically decreasing functions of distance and have a non-zero value on the entire domain, allowing an intuitive association of the worker's position with every visible node. We also assert that every worker's position will always be associated with at least one node because of GVD's space covering properties. The only exception to the aforementioned claim is if the worker is trapped by mobile robots but we argue that the warehouse management system must never allow such an event to occur for obvious safety concerns. Furthermore, we also modulate the Gaussian with the following triangular function:

$$\Phi(\theta_i) = \frac{\pi - |\theta_i|}{\pi^2}, \tag{3.12}$$

where $\theta_i \in [-\pi, \pi]$ is the difference between worker orientation and the angle at which the worker sees i-th node. It amplifies the association with those nodes the worker is oriented at since we assume that the worker will look at the path it is planning to take [139]. We also need to define the isolation matrix $\mathbf{I}^{n \times g}$, where n is the total number of nodes and g is the number of goal nodes:

$$\mathbf{I}_{i,j} = \begin{cases} 1, & i = n - g + j \\ 0 & \text{otherwise.} \end{cases} \tag{3.13}$$

We normalize the association vector $\mathbf{c}$ and obtain modulated approximate distance vector $\mathbf{d}$ by multiplying it further with the distance matrix $\mathbf{F}$ and isolation matrix $\mathbf{I}$:

$$\mathbf{d} = \mathbf{cFI}. \tag{3.14}$$

Each element of vector $\mathbf{d}$ represents modulated measure of distance to the respective goal. In order to find out if the worker is moving towards or away from the goal, we compare this distance to alternative worker positions and orientations. We take the location l' at which the worker was prior to the last intention estimation update and calculate the difference r between that position and the current worker's position l. Then, we generate set of $m = 16$ equidistant points p on a circle around l' with the radius r which is shown in Figure3.6. For each point $p_i \in p$ and potential worker orientation $\theta \in \{-\pi, -\frac{3\pi}{4}, \dots, \pi\}$ we repeat the calculation of vectors $\mathbf{c}$ and $\mathbf{d}$ and append the result to potential modulated distances matrix $\mathbf{D}$. Computing the matrix $\mathbf{D}$ enables us to validate the worker's motion with respect to states

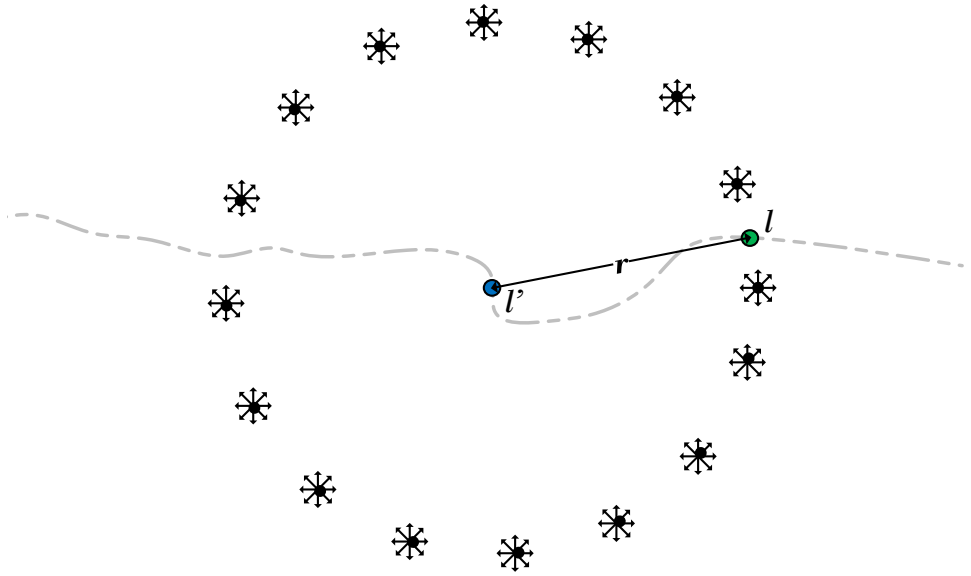


Figure 3.6: Illustration of generating additional points p with orientations θ. We validate the worker's motion by comparing the vector $\mathbf{c}$ of the worker's position and orientation at location l with vectors $\mathbf{c}$ of the newly generated points.

it *could be in*, rather than to the state it *had been to*. The proposed algorithm pseudocode can be seen in Algorithm5.

We use the distance vector $\mathbf{d}$ and the distance matrix $\mathbf{D}$ to validate worker's actions, and we introduce the motion validation vector $\mathbf{v}$, which is computed as follows:

$$\mathbf{v} = \frac{\max\limits_{1 \le i \le n} \mathbf{D}_{ij} - \mathbf{d}}{\max\limits_{1 \le i \le n} \mathbf{D}_{ij} - \min\limits_{1 \le i \le n} \mathbf{D}_{ij}}. \tag{3.15}$$

If the worker is moving towards the goal, the corresponding value of $\mathbf{v}$ will be close to unity, and if it is moving away from that goal, the corresponding value will gravitate to zero. One could argue that $\mathbf{v}$ may be interpreted as the estimate of worker intention because it expresses a measure of approaching the goal. However, elements of $\mathbf{v}$ are very sensitive to sensor noise and they would need to be filtered if one wanted to draw inferences about worker intentions directly from them. Additionally, even though observing the current value of $\mathbf{v}$ does indeed carry crucial information for estimating worker intention, it is not sufficient since it lacks a history of past values of $\mathbf{v}$. Consider the following simple example depicted in Figure3.7, where a warehouse worker moved past the goal labeled by the red square, and further advanced towards the goal labeled with the green square. As soon as the worker turned right, the value of $\mathbf{v}$ related to the green, as well as red, goal started to grow, since the distance between the worker and that goal started to drop. However, since the worker previously failed to turn towards the red goal, its intention estimation for that goal should have remained low. This example also demonstrates why there is a need to measure the distance of a traversable path as the approaching measure, instead of simply having, e.g., the Euclidean distance as elements of vector $\mathbf{v}$. Before further calculations, we apply a discrete first-order low-pass filter on $\mathbf{v}$ in order to reduce the noise influence.

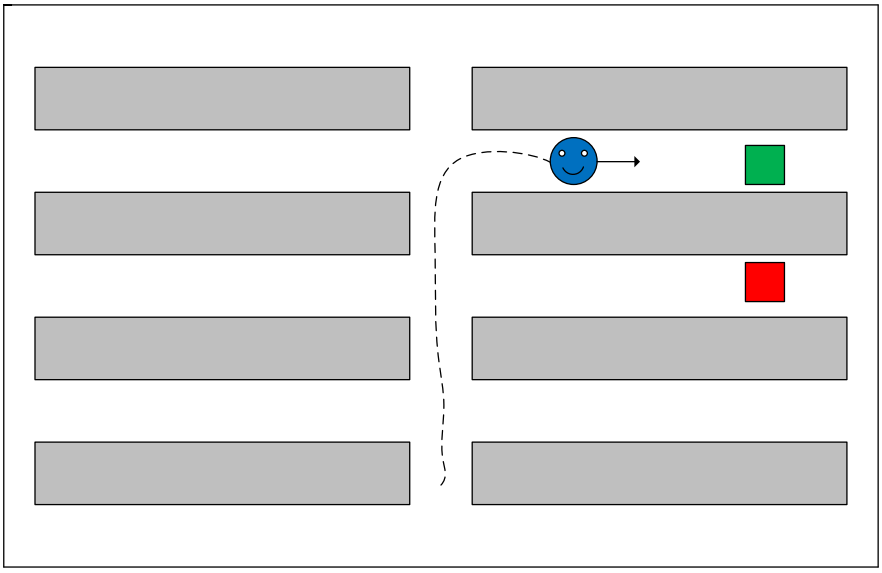


Figure 3.7: The warehouse worker (blue circle) has previously decided not to turn towards the goal labeled with the red box. It is intuitive that the worker desires the green goal more than the red goal because of its action history despite the fact it is now reducing its distance to both goals. The intention estimation algorithm has to take action history into account when estimating the worker's intention.

3.4 HMM BASED INTENTION ESTIMATION

In the previous section, we have introduced two approaches for human action validation. The first one is intended for static warehouse environments and is based on MDP state values obtained via the value iteration algorithm. The second approach is based on GVD space partitioning and planning using D* with the comparison of the current state with the space of possible neighboring states. In the rest of this section, we will consider both of these inputs as motion validation vector **v** and introduce the model for solving the worker's intention estimation problem. While worker's actions, manifested as moving and turning, are fully observable, they depend on the worker's inner states (desires), which cannot be observed and need to be estimated [4]. We propose a framework based on the hidden Markov model for solving the worker's intention estimation problem. They are an MDP extension including the case where the observation (worker's action) is a probabilistic function of the hidden state (worker's desires) which cannot be directly observed. It is important to note that MDP and HMM architecture share a lot of concepts and nomenclature. Because of that, the shared acronyms and names of components will pertain to the introduced HMM model unless stated otherwise. We propose a model with $g + 2$ hidden states shown in Figure3.8and listed in Table3.1. Hidden states G_i describe the worker's intention of going to i-th goal, $G_?$ indicates that the worker prefers multiple goals and the model cannot decide on the exact desire with enough certainty. On the other hand, hidden state G_x indicates that the worker is moving away from all the goal locations. These hidden state models the case of the worker being irrational or a worker desiring a goal we have not yet specified. It is important

Table 3.1: HMM framework components

Symbol	Name	Description
G_i	Goal i	Worker wants to go to Goal i
$G_?$	Unknown goal	Worker's intentions are not certain
G_x	Irrational worker	Worker behaves irrationally

to note that the proposed model cannot distinguish between these two cases. Introduced hidden states enable the human intention recognition model to elegantly save the intention estimation history as probabilities $P(G_i)$. The first building block in this HMM architecture is the transition matrix $\mathbf{T}^{g+2\times g+2}$:

$$\mathbf{T} = \begin{bmatrix} 1-\alpha & 0 & \dots & \alpha & 0 \\ 0 & 1-\alpha & \dots & \alpha & 0 \\ \vdots & & \ddots & & \vdots \\ \beta & \beta & \dots & 1-g\beta-\gamma & \gamma \\ 0 & 0 & \dots & \delta & 1-\delta \end{bmatrix}, \tag{3.16}$$

where the architecture and description of the matrix parameters can be seen in Figure3.8. We have obtained parameters experimentally as follows: $\alpha = 0.5$, $\beta = 0.1$, $\gamma = 0.05$, $\delta = 0.1$.

We use the calculated motion validation vector $\mathbf{v}$ to generate the HMM's emission matrix $\mathbf{B}$. Every time the worker moves or turns significantly, we estimate the worker's intention using the Viterbi algorithm [110], which is often used for solving HMM human intention recognition models [142]. The inputs of the Viterbi algorithm are the hidden states set $S = \{G_1, \dots G_g, G_?, G_x\}$, hidden state transition matrix $\mathbf{T}$, initial state Π, sequence of observations $\mathbf{O}$, and the emission matrix $\mathbf{B}$. The HMM framework generally assumes a discrete set of observations, but since our observation is the validation vector $\mathbf{v}$ with continuous element values, we have decided to modify the input to the Viterbi algorithm by introducing an expandable emission matrix $\mathbf{B}$. While the classic HMM emission matrix $\mathbf{B}^{n\times g}$ links hidden states with discrete observations via fixed conditional probability values, elements of the introduced expandable emission matrix $\mathbf{B}^{k\times g}$, where k is the recorded number of observations, are functions of the observation value. By using this modification of the emission matrix, we additionally simplify the Viterbi algorithm because there is no set of discrete observations it has to iterate through. Once a new validation vector v is calculated, the emission matrix is expanded with the row $\mathbf{B}'$, where the element $\mathbf{B}'_i$ stores the probability of observing $\mathbf{v}$ from hidden state G_i. We also calculate the average of the last m vectors $\mathbf{v}$ and the maximum average value ϕ is selected. It is used as an indicator if the worker is behaving irrationally, i.e., is not moving towards any predefined goal. The value of the hyperparameter m decides how much evidence we want to collect before we allow the algorithm to declare the worker irrational. If the worker has been moving towards at least one goal in the last m iterations ($\phi > 0.5$), we calculate $\mathbf{B}'$ as:

$$B' = \zeta \cdot \begin{bmatrix} \tanh(\mathbf{v}) & \tanh(1-\Delta) & 0 \end{bmatrix}, \tag{3.17}$$

and otherwise as:

$$B' = \zeta \cdot \begin{bmatrix} \mathbf{0}_{1\times g} & \tanh(0.1) & \tanh(1-\phi) \end{bmatrix}, \tag{3.18}$$

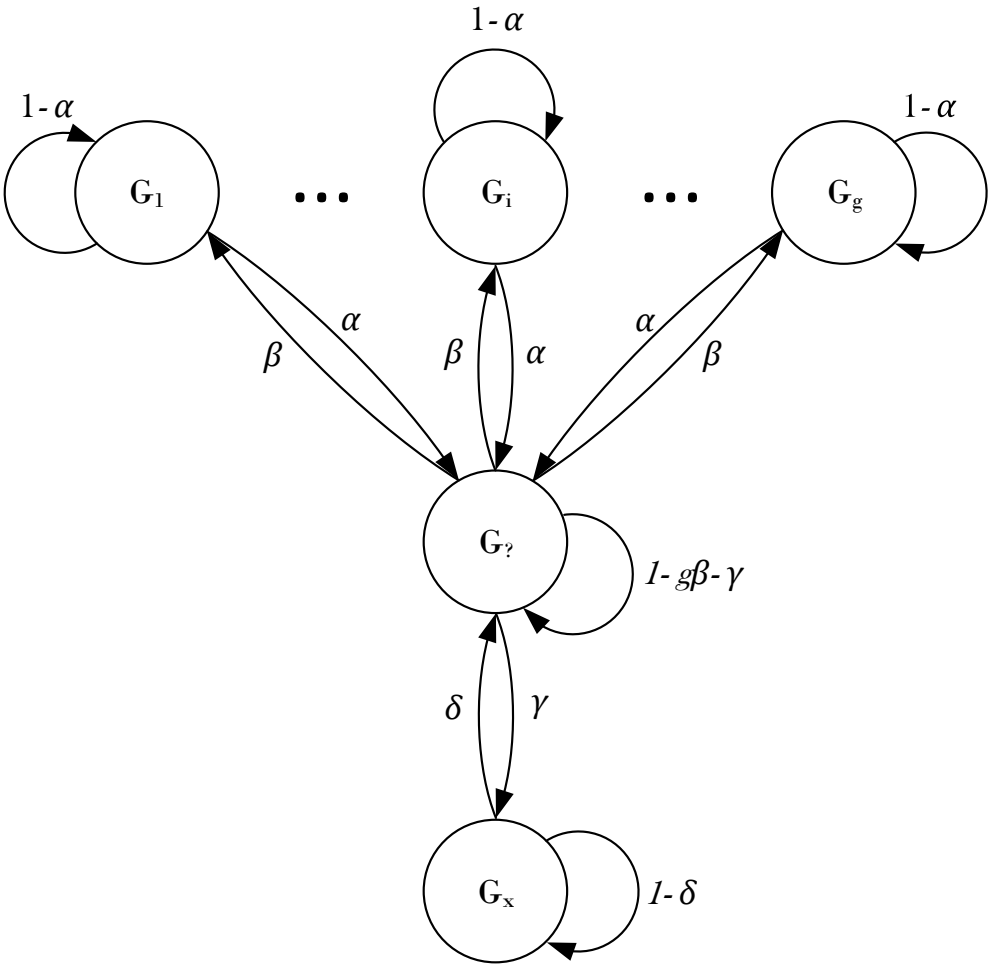


Figure 3.8: HMM architecture used for human intention recognition. Worker's change of mind tendency is captured by the parameter α and the parameter couple β and γ set the threshold for estimating intention for each goal location. Increasing β leads to quicker inference of worker's intentions and increasing γ speeds up the decision making process. Parameter δ captures model's reluctance to return to estimating the other goal probabilities once it estimated that the worker is irrational.

where ζ is a normalizing constant and we calculate Δ as the difference between the largest and second-largest element of $\mathbf{v}$. Using such a way of calculating Δ enables us to simply encode that, in order for our model to decide in favor of any goal location, it has to significantly stand out from other goals. Humans often infer the intentions of others by observing their actions [74], which are generally not optimal with respect to their goals. Nevertheless, we argue that the worker will globally move towards the goal it desires the most, but may locally take suboptimal actions such as looking around or swerving laterally. In order to encode such behavior, we use the hyperbolic tangent function in (3.17) and (3.18) to reduce the difference between the actions that indicate movement towards the goal whilst penalizing other actions approximately equal to the linear function would. Finally, we set the initial

probabilities of worker's intentions as:

$$\Pi = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 \end{bmatrix}, \tag{3.19}$$

indicating that the initial state is $G_?$ and the model does not know which goal the worker desires the most. The Viterbi algorithm outputs the most probable hidden state sequence and the probabilities $P(G_i)$ of each hidden state in each step. These probabilities are the worker's intention estimates.

Worker tasks are not always predefined in the beginning and can appear or cease during workers' stay on the shop floor. We have taken such events into consideration and made it possible to add or remove goals during the experiment. If the goal has to be removed, e.g., because another worker took that job over or the task was canceled, we simply discard that goal from our calculations and add it to the unknown goal intention estimation. If the goal has to be added, we do this by setting its intention estimation to $\max(\min(P(G_i), 0.1)$ with $0 < i < g$ and by expanding the **I** and **T** matrices. We would like to emphasize that it is not possible to add an arbitrary number of goals, because of the fixed values of parameters in **T** matrix. The maximum number of goals this model can handle is limited because elements of transition matrix **T** must be greater than 0. The only value of matrix **T** that depends on the goal number is the probability of staying in $G_?$ state and equals to $1 - g\beta - \gamma$. We calculate the maximum number of goals by applying the positivity condition to that expression:

$$g_{max} = \left\lfloor \frac{1-\gamma}{\beta} \right\rfloor, \tag{3.20}$$

which for our current value of parameters amounts to 9 goals. However, after detailed testing, we have concluded that the proposed model does not perform well if there are more than 5 goals. The main difficulty here is that if there are more potential goals, workers' actions are not discriminatory enough and there is not enough evidence that the worker desires one goal more than the other goals. Because of that, the model estimates $G_?$ as the most probable state throughout the experiment. While we argue that this still is accurate worker's intention estimation, it is hardly useful to a supervisory system that is taking our algorithm's estimations as input.

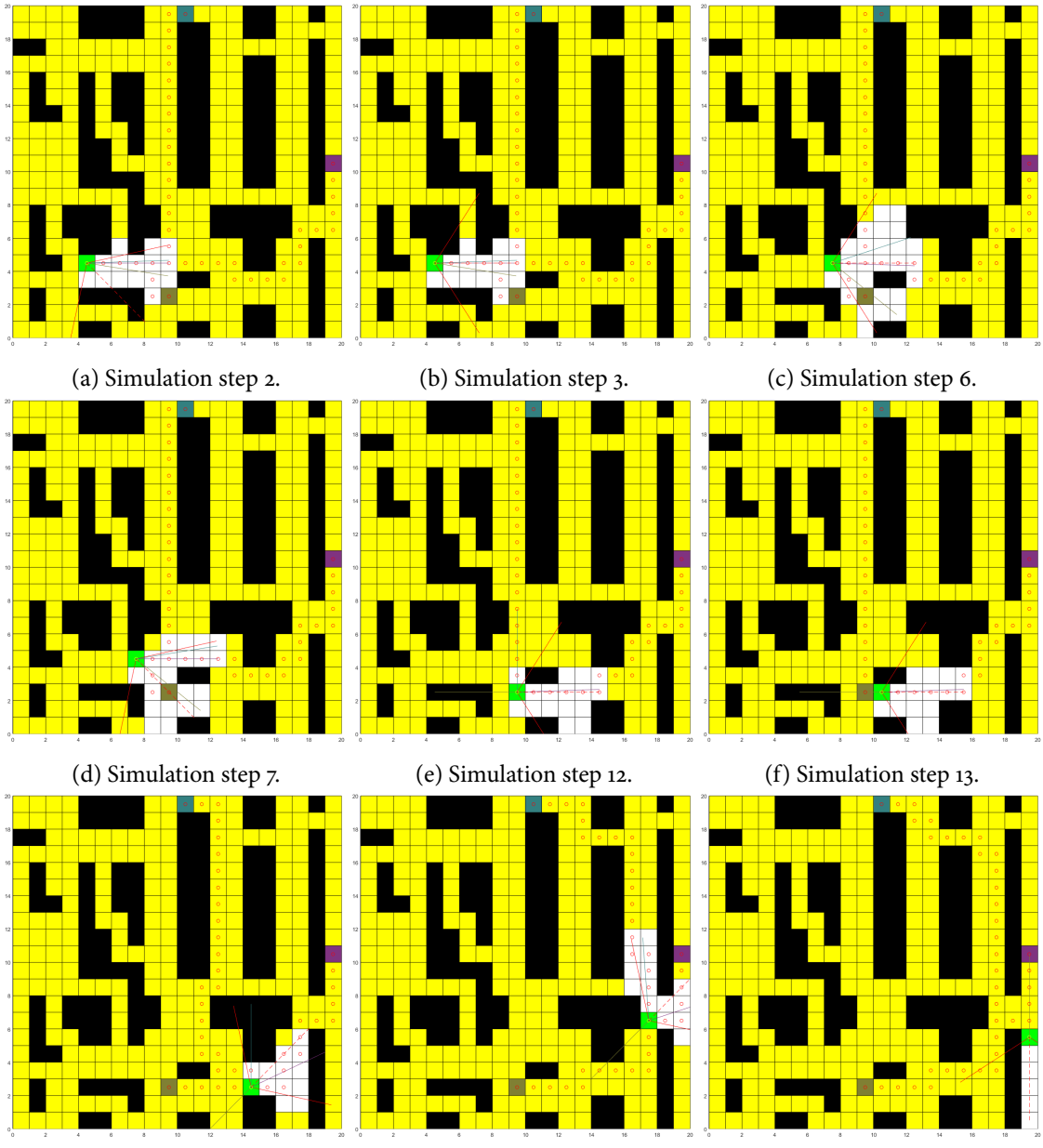
3.5 Experimental Results

This section presents the simulation and experiment results of the proposed framework. Firstly, we will discuss the results for our proposed simulated environment and show that the MDP human action validation approach successfully captures cues such as moving towards the goal of looking at the goal's direction. Then, we will exhaustively comment on multiple experiments conducted in the real warehouses and a larger virtual reality warehouse. The Hololens localization will be showcased and some special cases, such as goal addition during the experiment will be discussed.

3.5.1 *Simulation Results*

In the previous sections, we have introduced the MDP and HMM frameworks for modeling human action validation and intention recognition. We have obtained the model parameters

empirically and conducted multiple simulations evaluating the proposed human intention recognition algorithm. The proposed algorithm is tested in a scenario, where the most important simulation steps are shown in Figure3.9and the corresponding desire estimates



(a) Simulation step 2. (b) Simulation step 3. (c) Simulation step 6.

(d) Simulation step 7. (e) Simulation step 12. (f) Simulation step 13.

(g) Simulation step 18. (h) Simulation step 25. (i) Simulation step 31.

Figure 3.9: Representative simulation steps (best viewed in color) of human intention estimation in simulated environment.

are shown in Figure3.10. The starting position is $(x_1, y_1, \theta_1) = (5, 5, \frac{3\pi}{2})$. The agent behaves consistently with all the hypotheses and proceeds to the state $(x_6, y_6, \theta_6) = (8, 5, 0)$. Because of the mentioned hypothesis consistency, the desire estimates for all of the goal states increase. The actions from simulation step 7 to step 12 are consistent only with the hypothesis H_3 which manifests as the steep rise of the $P(G_3)$ and fall of probabilities related to other goal hypotheses. In step 13, action "Stay" is the only action consistent with the hypothesis H_3 and because the agent chooses the action "Right", the $P(G_3)$ instantly falls towards the zero,

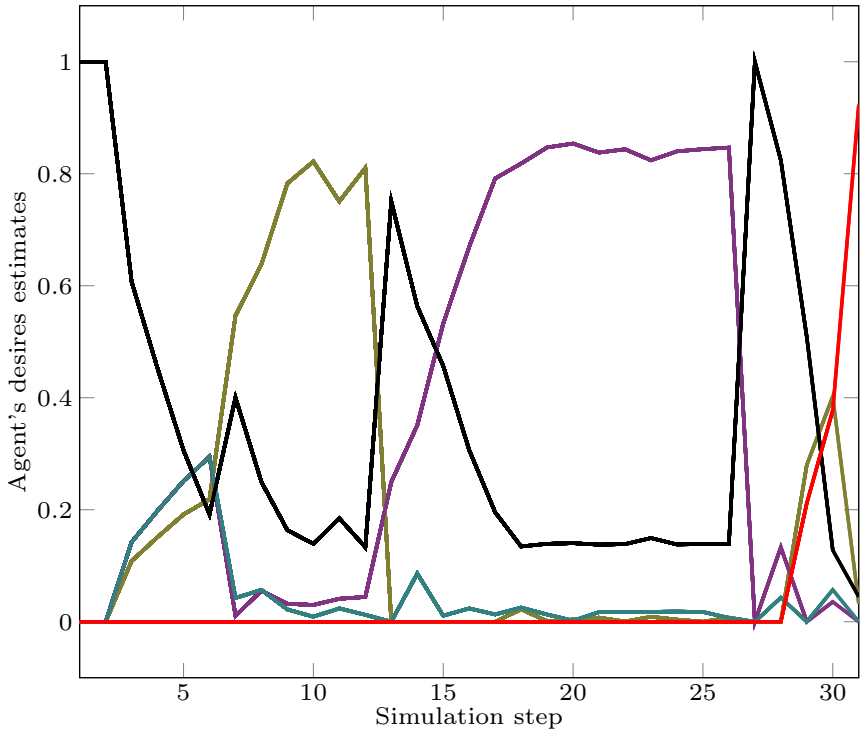


Figure 3.10: Hidden state (desires) probabilities. Probabilities of the goal states are colored according to the goal tile's color. The unknown goal state probability is colored black and irrational agent state probability is colored red.

and $P(G_?)$ and $P(G_2)$ rise. While it might seem obvious that the agent now actually wants to go to Goal 2, it has previously chosen actions inconsistent with that hypothesis and the model initially gives a greater probability value to the desire $G_?$ than to G_2. The next few steps are consistent with the hypothesis H_2 and the $P(G_2)$ rises until the simulation step 18, when it enters a steady state of approximately 0.85. The goal desires will never obtain value of 1 because the element B'_4 is never zero, thus allowing agent's change of mind. In the state $(x_{25}, y_{25}, \theta_{25}) = (16, 7, \frac{\pi}{4})$ agent can decide to go to the Goal 1 or Goal 2. However, it chooses to take the turn towards the dead end in the simulation step 31. The proposed model recognizes that this behavior is inconsistent with all of the hypotheses and the $P(G_x)$ steeply rises to value slightly smaller than 1, declaring the agent irrational.

In this section, we demonstrate the proposed human intention estimation algorithm performance and discuss the results. We have conducted experiments in both an industrial setup and in a larger virtual reality rendered warehouse. Before we analyze the results, we discuss the method for evaluating the proposed model.

To the best of our knowledge, a recognized criterion or method for evaluating human intention estimation models does not exist. Models such as those proposed in [74] rely on people's judgments to evaluate results, while models which use learning methods have well-defined start and end points of the experiments, and the ground truth information is readily available or evident from the experiment's endpoint. While the worker's intention at the end of the experiment is unambiguous, it is unclear how to empirically determine intentions *during* the experiment because of the possibility of the worker's change of mind. Also, we obtained the parameters given in (3.16) experimentally after thorough testing, and

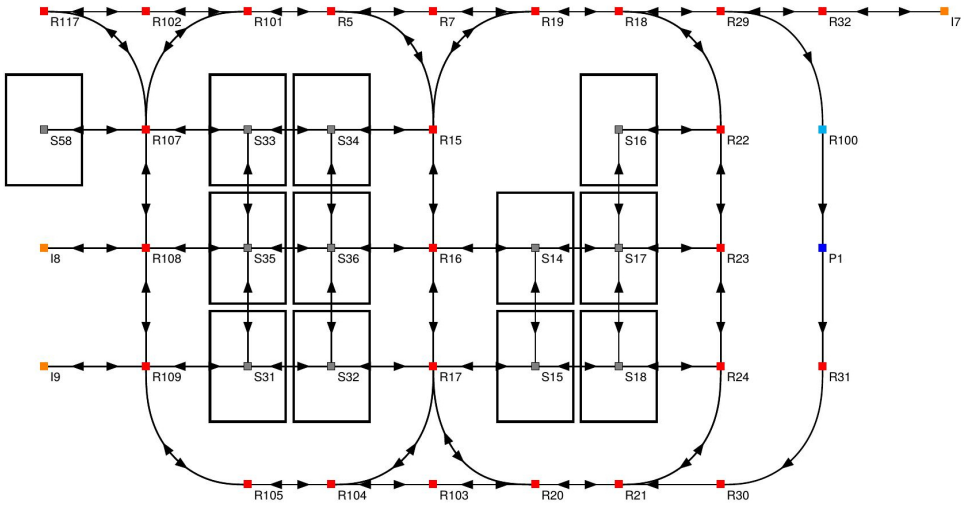


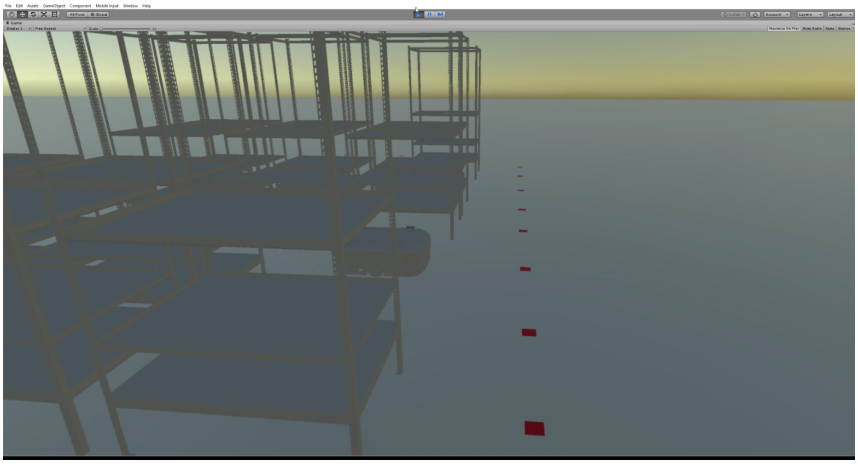
Figure 3.11: The layout of the laboratory warehouse used for AR-based experiments. Mobile robots move over ground nodes labeled with letters R (traversable path for warehouse workers) and S (under the racks which robots can pick up and move). Nodes P and R100 are used as picking stations and queue nodes but are treated the same as other R nodes in our experimental setup.

they were selected in a way to produce consistent and semantically interpretable results on different datasets. We do not claim that those parameters are in any way optimal, but we assert that applying some of the well-known algorithms for learning HMMs, such as expectation-maximization, is not feasible in our case, since an unbiased labeled dataset is unavailable. Furthermore, we are deterred from using people's judgments as ground truth for the same reason. Since we want estimates of the proposed human intention estimation algorithm to be useful to a supervisory system, we insisted that if the worker is moving towards more than one goal, the intention for those goals should be higher than for any other goals; but, it should be lower than estimation for $G_?$. If the worker moves to a single goal, intention for that goal should be the highest, while if the worker does not move towards any goal for a predefined amount of time, the model should declare it as irrational. We have chosen that time to be 1 second. Using such interpretation of intentions can be encoded in a mobile robot fleet management system, which can then, e.g., reroute mobile robots away from the goal worker is moving to.

3.5.2 *Augmented reality experiments*

The augmented reality experiments were conducted in the laboratory warehouse of Swisslog and consisted of twelve racks and two robots. The layout of the warehouse is shown in Figure3.11. Although of a smaller scale than commercial warehouses, it nevertheless enables conducting experiments in a realistic environment. As previously mentioned, for the AR experiments the Microsoft Hololens augmented reality glasses were used. Since our intention estimation algorithm assumes that the position and orientation of the worker are known, we used Hololens' proprietary algorithm to localize the person while walking inside

(a) First-person view from the 2 Megapixel Hololens camera



(b) First-person view from HTC Vive

Figure 3.12: Comparison between the real-world experiment and the experiments in VR. Both experiments use exactly the same setup.

the warehouse. Since no external tracker was used, we first wanted to verify the localization accuracy of the Hololens inside a warehouse-like environment. The experiment was conducted by a person first starting from a predefined ground node (R32 in Figure3.11, all the ground nodes are manually marked and unique) and then walking from ground node to ground node. This is similar to the way robots follow a path since they use ground nodes for navigation. In this experiment, all AR interactions were disabled and the person navigated the warehouse autonomously the only exception was a holographic sphere positioned at the Hololens' location every 100 frames. This allowed us to conduct a qualitative analysis of the localization shown in Figure3.13. The results showed that the localization was indeed robust with only a few centimeter deviations, at most, from the straight-line paths between nodes. A first-person view from the Microsoft Hololens is shown in Figure3.12a.

To test the proposed algorithm, two experiments were conducted (besides the localization experiment described previously). In both experiments the starting point was node R32 and featured two robots; one stationary and positioned at R15, while the other was mobile. In the first experiment shown in Figure3.14, the worker has three potential goals located in front of racks near nodes on R117, R108, and R17. The worker initially starts moving toward

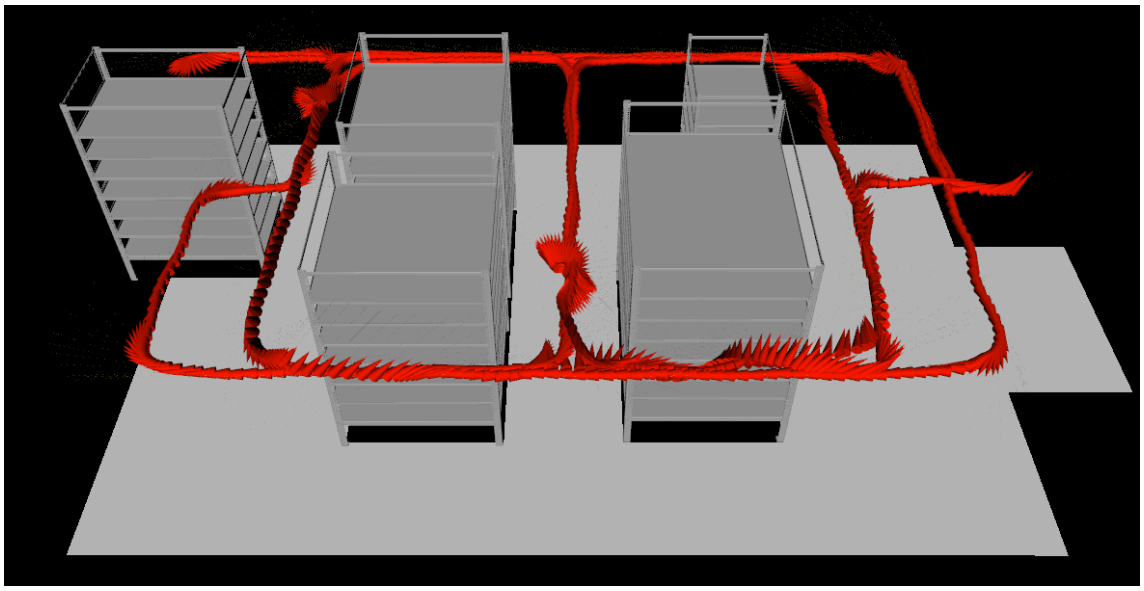


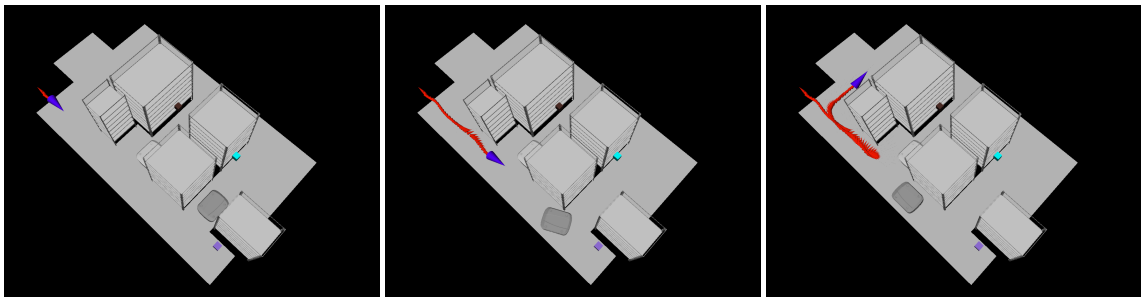
Figure 3.13: Conducted experiment showcasing the precision of the Hololens' localization. We have created a warehouse model in *rviz* - Robot Operating System's 3D visualization tool, which will be further used to demonstrate proposed algorithm's results.

node R117, but the mobile robot moves forward from node R107 and turns right towards node R101, thus blocking the worker's initially intended path towards the R117. The worker then turns around and follows the only remaining path towards R117 which is also the best path for the other two goals. Once the worker reached R103 and continued to R104, the proposed algorithm detects that it has failed to turn towards the R17 goal and lowers its intention estimation value. The worker further continues to goal R109 but turns around before reaching it. After turning, the only goal the worker could be going to is R17, but the worker continues past it returning to node P. When the worker passed the goal R17, the model recognized there are no goals it could be going to and declared the worker irrational. The estimated intentions of the proposed algorithm can be seen in Figure3.15a. The second experiment included a worker moving towards R117 and changing its mind to go to R17 simultaneously with a mobile robot blocking the best path towards that goal and eventually returning to R117. The main idea of the second experiment was to showcase the algorithm's flexibility in scenarios where the worker changes their mind often. Given that, we only show the second experiment in the accompanying video[1].

3.5.3 *Virtual Reality Setup*

We built a virtual reality framework for rapid prototyping of applications for flexible robotized warehouses, a tool that might also evolve into a training framework in the future. The main motivation behind a virtual framework, i.e., a virtual reality digital twin of a warehouse, is that commercial automated warehouses are often unavailable for experiments, but they can be simulated, thus avoiding the warehouse downtime and financial losses, yet enabling testing in full scale to identify potential problems and obtain realistic user experience. Furthermore, this approach also enables us to do tests with multiple users more

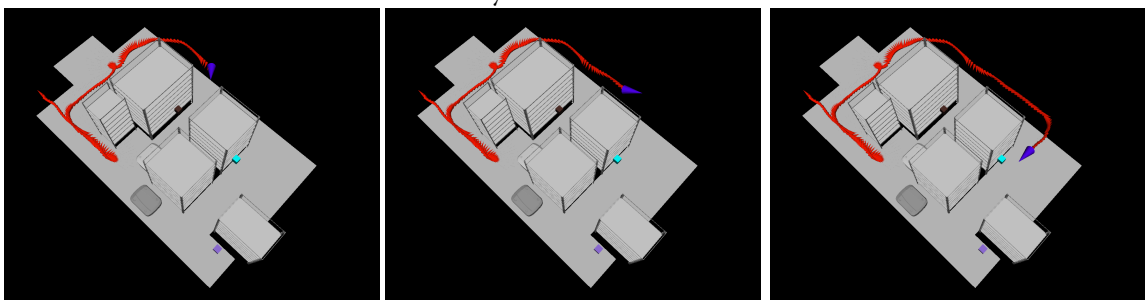
[1] https://youtu.be/SDD-v-pHov4



(a) Initial position.

(b) The worker moves towards the purple goal (R117) but the mobile robot obstructs the intended path towards it. Because the worker has not turned around immediately, the model warns that the worker is behaving irrationally.

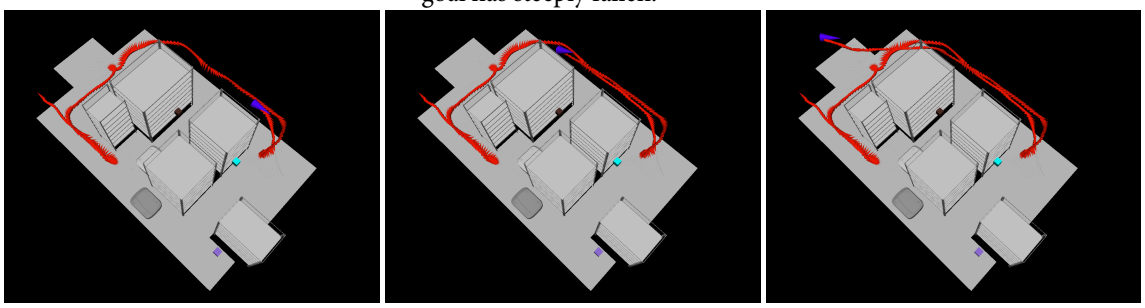
(c) The worker turns and follows the path consistent with going to all three goals.



(d) The worker is on a crossroad. If it turns towards the brown goal it is obvious that it is the goal wants to reach.

(e) However, the worker has decided not to advance towards the brown goal and to continue towards the cyan and purple goals instead. Because of that the intention estimation for the brown goal has steeply fallen.

(f) The worker stopped near the cyan goal and started turning around in place.



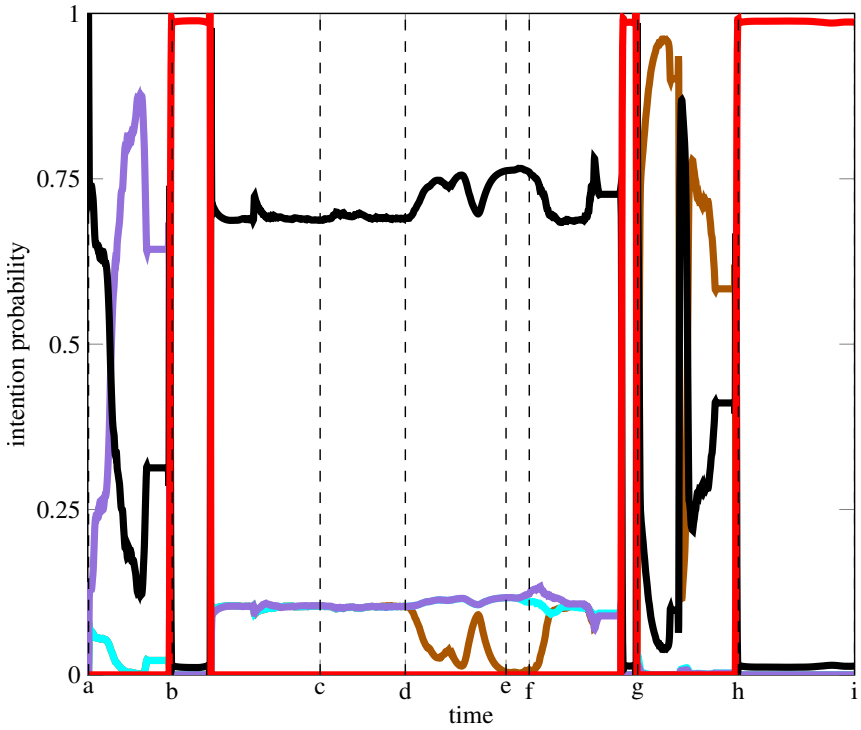
(g) The worker has taken its AR device off and is moving towards the brown goal.

(h) Because the worker passed brown goal and is moving backwards, the model estimates its behavior to be irrational.

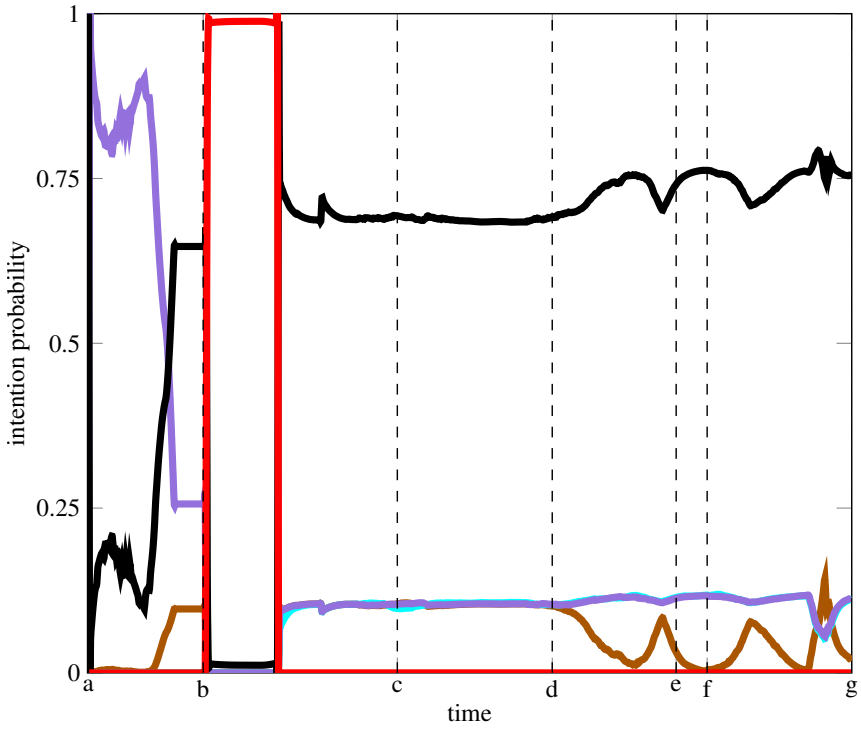
(i) End of the experiment.

Figure 3.14: Key moments of the industrial experiment setup in the laboratory warehouse visualized in *rviz*. Goal nodes are labeled with cubes as follows: R117 purple, R17 brown and R109 cyan.

freely and achieve the best possible interaction modalities. Note that besides testing user or worker behavior inside the warehouse, the virtual setup also serves for testing augmented



(a) Algorithm's output in real world AR experiment.



(b) Algorithm's output in VR experiment.

Figure 3.15: Comparison of the proposed algorithm's output between the real-world experiment and the experiment recreated in VR. Intention estimations for the three goal locations are labeled with respect to their color in Figure3.14(brown, cyan, and purple), the unknown goal state is labeled black and the irrational worker state is labeled red. One can see that the results in this scenario are similar for both AR and VR technology.

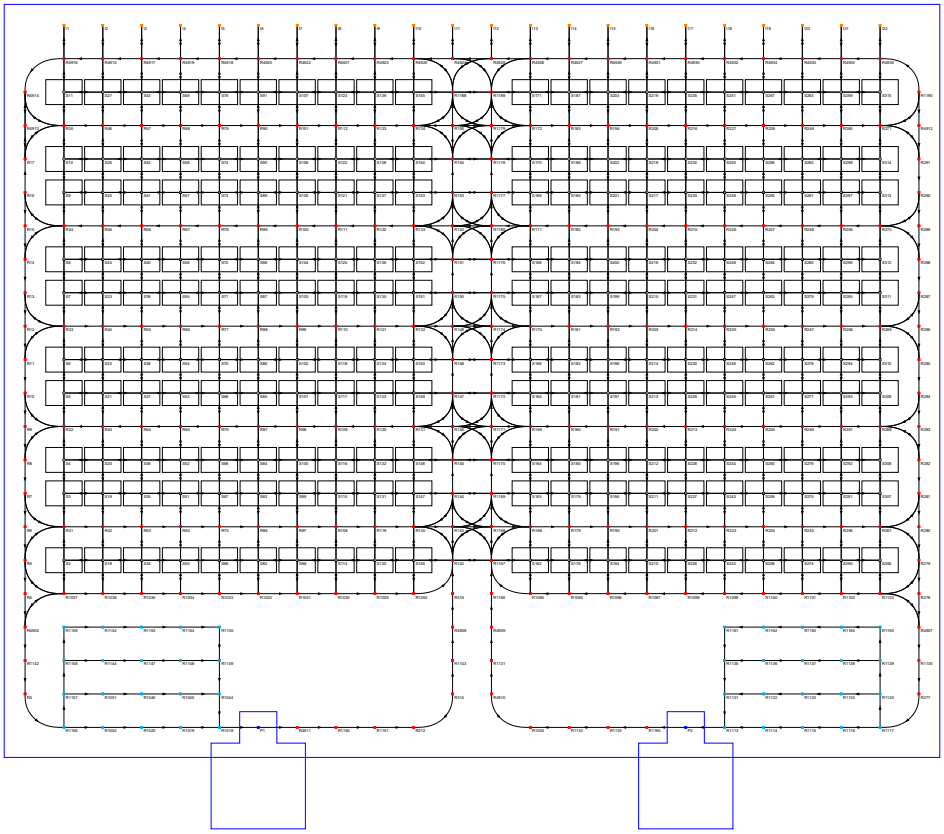
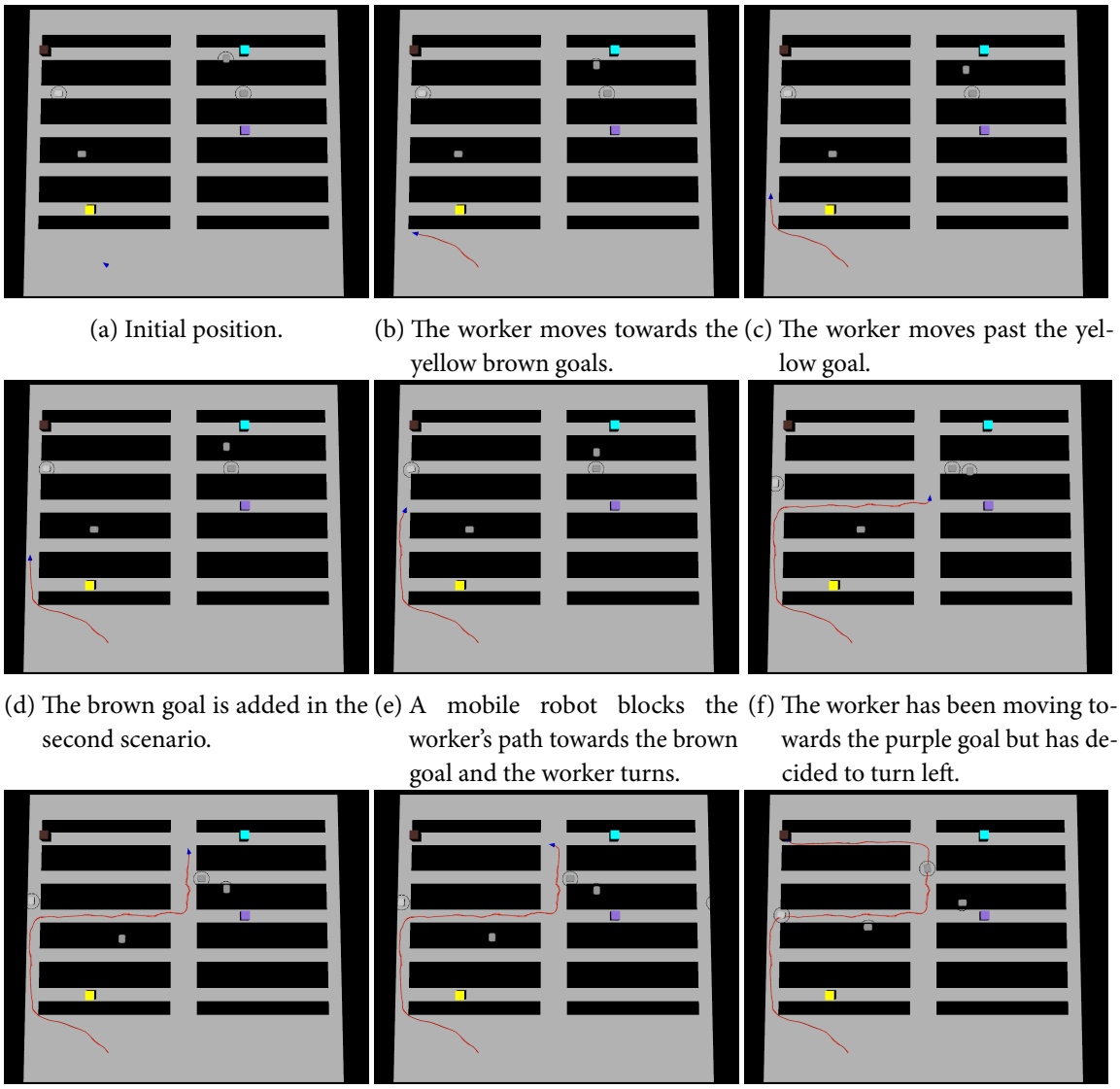


Figure 3.16: The larger warehouse layout used for VR experiments. The layout shows movable racks, picking points and ground nodes that are connected via graph edges robots can move on. However, in present chapter we do not leverage this information for human intention recognition.

reality applications for warehouse workers wearing glasses such as the *Microsoft Hololens*. Concretely, the application was developed in *Unity3D* and is used with the *HTC Vive* headset. An example of a user view within the virtual reality warehouse is shown in Figure3.12b.

The warehouse layout is planned using Swisslog proprietary network planner, from which an XML file is exported. We have developed an XML file parser which together with an available CAD model builds a VR warehouse from scratch. The fleet of robots is presently controlled by parsing a series of JSON messages, which are the output of a path planner [143].

Currently, two AR interaction scenarios have been implemented for prototyping augmented reality applications: path visualization for worker navigation and rack object picking assistance. The virtual setup emulates the Microsoft Hololens: all of the *holographic objects* are only visible through a narrow field of view (FoV), which according to specifications ranges from 30◦ to 35◦ horizontally and 17.5◦ to 17.82◦ vertically (we selected the lower bounds). The interaction pointer uses raycasting to position itself with respect to objects. The potentials of the proposed system are numerous. For example, such a system allows us to test if a certain interaction modality fails or is not as informative because of the low FoV in a realistic environment. It also allows us to inject localization errors to determine at which point various interaction modalities become unusable. However, in the present solution,



(a) Initial position.

(b) The worker moves towards the yellow brown goals.

(c) The worker moves past the yellow goal.

(d) The brown goal is added in the second scenario.

(e) A mobile robot blocks the worker's path towards the brown goal and the worker turns.

(f) The worker has been moving towards the purple goal but has decided to turn left.

(g) The worker is on a crossroad. If it turns towards the brown goal it is obvious that it is the goal it wants to achieve.

(h) The worker has turned towards the brown goal.

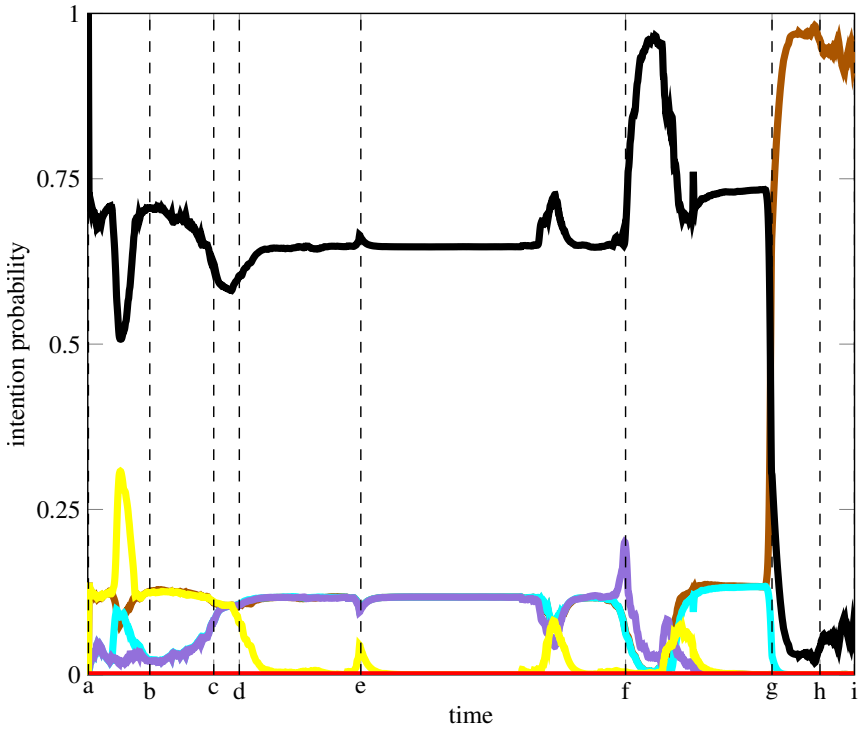
(i) End of the experiment.

Figure 3.17: Key moments of the virtual reality generated scenario using larger warehouse visualized in *rviz*. Goal nodes are labeled with cubes and goal labeled with brown cube is being added during the experiment.

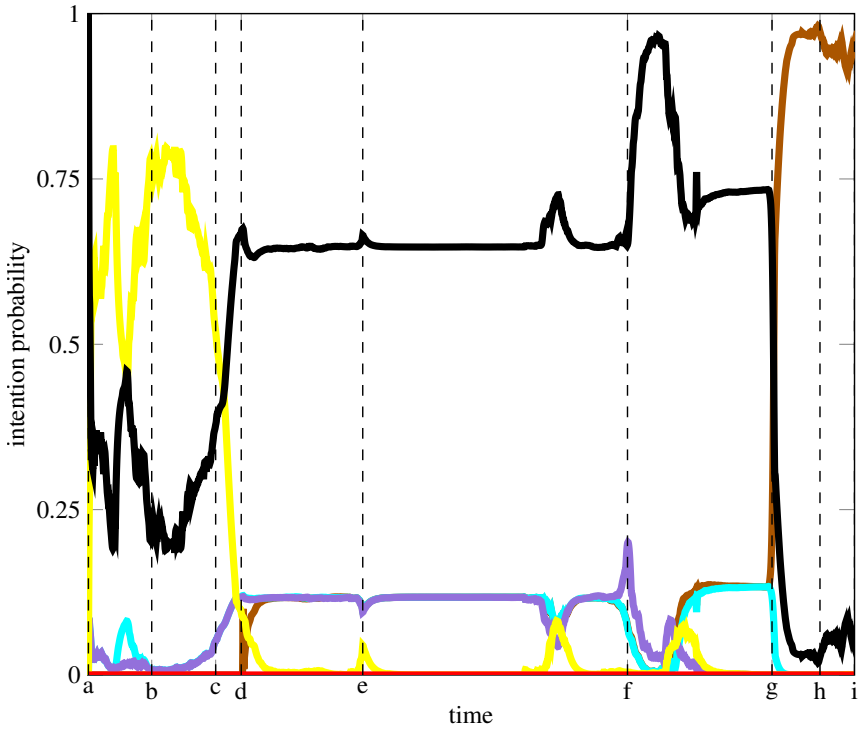
the virtual framework is used primarily as a controlled simulated warehouse environment for conducting experiments for worker intention estimation.

3.5.4 *Virtual reality experiments*

Due to the unavailability of a full-scale commercial warehouse for testing, we conducted larger-scale tests in VR, using the system described in Section3.5.3. The tracking method of the HTC Vive has an RMS error of 1.9 mm, which offers very accurate tracking for a realistic VR experience for warehouse localization and worker behavior purposes. We tracked the position of the user, as well as of each robot. An example of a first-person view from the

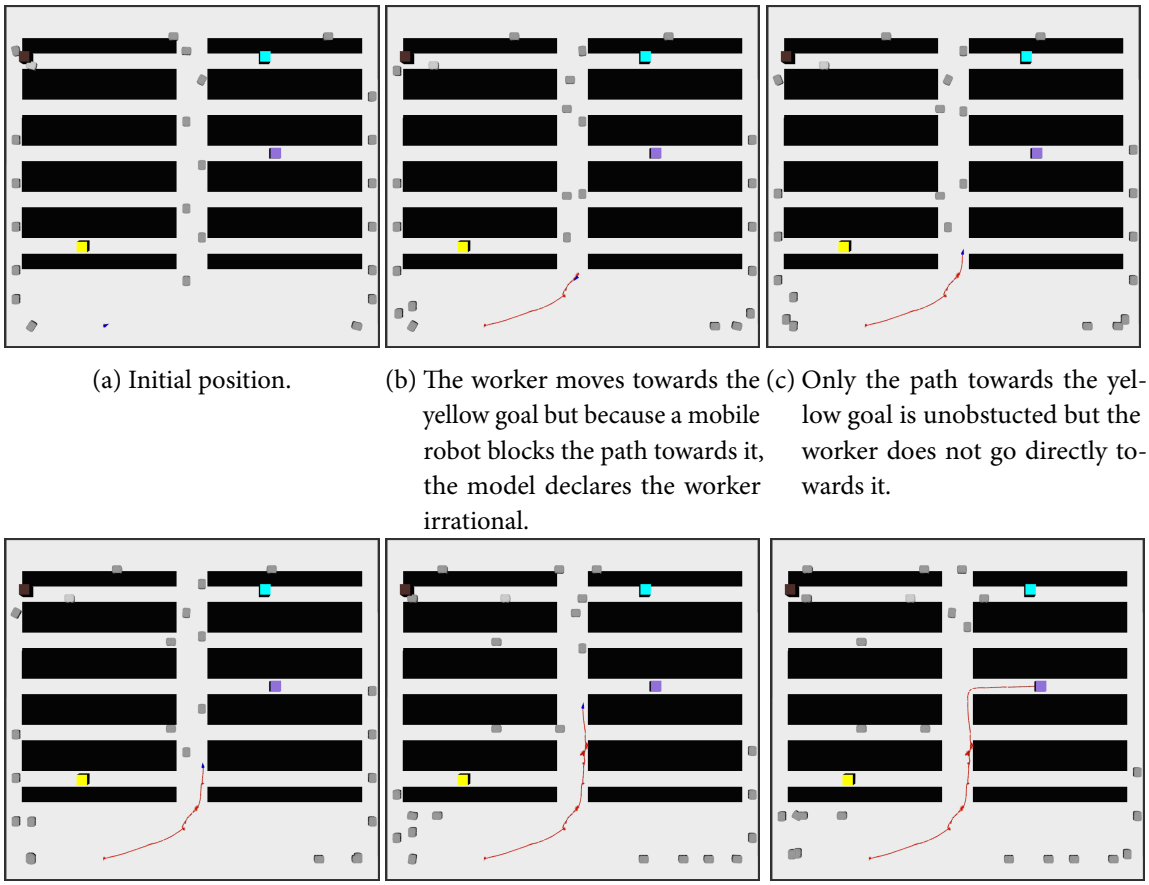


(a) Algorithm's output with brown goal being known through the whole experiment.



(b) Algorithm's output with brown goal being added during the experiment.

Figure 3.18: Comparison of the proposed algorithm's output in the larger VR warehouse between two different scenarios. Intention estimations for three locations are labeled with respect to their color in Figure3.17(brown, cyan, and purple), the unknown goal state is labeled black and the irrational worker state is labeled red. One can notice that the algorithm's outputs are different only on the segment where the brown goal has not been introduced in3.18b.



(a) Initial position.

(b) The worker moves towards the yellow goal but because a mobile robot blocks the path towards it, the model declares the worker irrational.

(c) Only the path towards the yellow goal is unobstucted but the worker does not go directly towards it.

(d) The mobile robot blocks the worker's advancement. Because the worker turns and moves while waiting for the robot to move, the model switches estimation between the yellow goal and the unknown goal state.

(e) Mobile robot moved and the worker is going to the purple goal.

(f) The worker is at the purple goal's location which ends the experiment.

Figure 3.19: Key moments of the virtual reality generated scenario using a larger warehouse with 24 mobile robots visualized in *rviz*. Goal nodes are labeled with colored cubes.

Hololens used in the laboratory warehouse and HTC Vive in a VR digital twin of the same warehouse can be seen in Figure3.12.

We repeated the first AR experiment in the constructed VR environment to demonstrate the applicability of VR for human intention estimation. The only difference between the scenarios was that we did not reproduce the last part of the AR experiment when the worker took off the AR device to simulate a type of irrational behavior. The results are shown in Figure3.15b, where we can see that the biggest difference between real-world and VR experiments is at the beginning of the experiment. Those differences are caused by the fact that the VR experiment starts with the worker going directly towards the goal, whereas at the beginning of the real word experiment the worker is still putting on the AR device. Because they can be interpreted as semantically similar to AR results, we conclude that testing intention estimation in VR can produce credible results.

We then proceeded to conduct experiments in a larger VR warehouse (layout can be

seen in Figure3.16) with four robots running on preprogrammed paths. These paths were generated by the path planner and read from a text file containing JSON messages. We considered two scenarios for this experiment: one with four initially known goals and one with three initially known goals with one goal being added during the experiment. Visualization of the key moments can be seen in Figure3.17and the proposed algorithm output is shown in Fig3.18. The worker initially starts moving towards yellow and brown goals. If the brown goal is known, the model cannot decide which goal the worker desires more and stays in the unknown goal state (Figure3.18a). However, if the brown goal has not yet been added, the worker moves only to the yellow goal, and the probability for that goal steeply rises (Figure3.18b). The worker then continues past the yellow goal which causes its intention estimation to fall. Shortly after this event, the brown goal is added in the second scenario. Once the mobile robot blocks the shortest path towards the brown goal, the worker turns right. Because it is now also moving towards the cyan and purple goal, the model cannot decide which goal the worker desires the most. The worker continues moving towards the crossroad and hesitates with turning toward brown and cyan goals which manifest as a spike of estimation for the purple goal. However, since the worker eventually turns toward brown and cyan goals, the purple goal's estimation steeply falls. The worker then continues moving towards the brown goal and the model recognizes its intention.

In the end, we have conducted another experiment on the large-scale VR test warehouse with 24 mobile robots. The robots were placed in groups of eight on the far left, middle, and far right vertical corridors, equally spaced vertically. Each robot then selected one of the reachable adjacent nodes at random and continued selecting nodes in such a manner, except that it is not allowed to return to the node visited in the previous step. If another robot already selected the same node in the same time step, it stops and waits for it to pass. This works well at emulating a fleet of robots moving around the warehouse without collisions. The goal of this experiment was to show the proposed algorithm's scalability with the respect to the number of mobile robots. We emphasize that mobile robot trajectories were not taking worker position into account. Given that, in some scenes, the worker is moving in the proximity of a mobile robot carrying a warehouse rack, which would not meet safety requirements in a realistic flexible robotized warehouse. Nevertheless, we use this scenario solely for illustration purposes in order to demonstrate the proposed algorithm's scalability. The key moments of the experiment are shown in Figure3.19and the result of intention estimation is shown in Figure3.20.

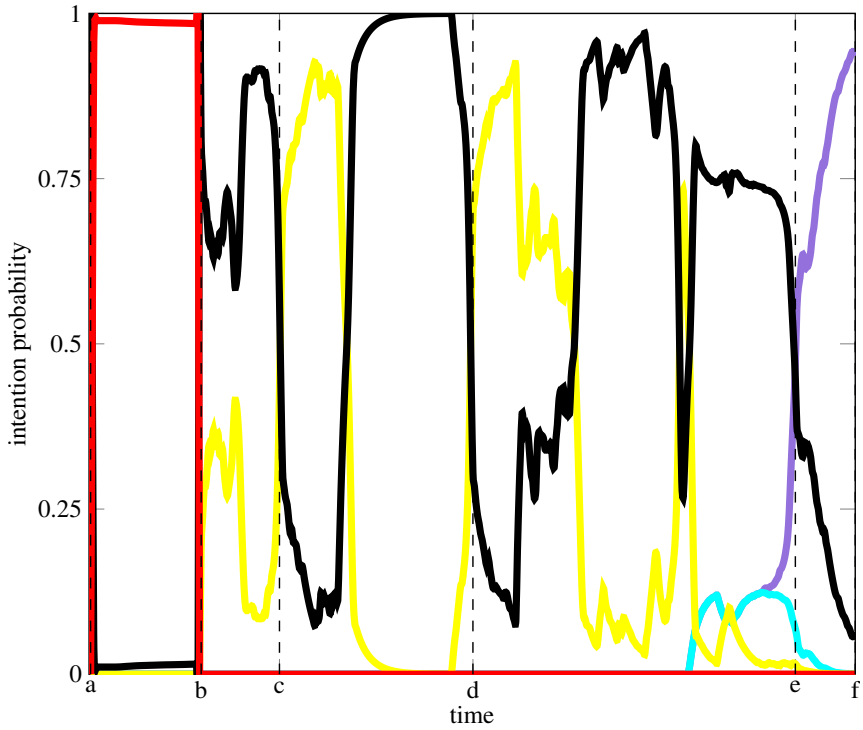


Figure 3.20: Proposed algorithm's output in VR scenario with 24 mobile robots. Intention estimations for goal locations are labeled with respect to their color in Figure3.19, the unknown goal state is labeled black and the irrational worker state is labeled red.

3.6 Summary

In this chapter, we have proposed a real-time human intention estimation algorithm capable of handling a dynamic environment. Our goal was to estimate the intention of a human worker inside of a robotized warehouse whose layout can change due to robots blocking paths. We assumed that the worker position and orientation are measured and that the warehouse layout and robot positions are readily available. The worker has a set of potential goal locations that can be defined before or added during the experiment. The task of the proposed algorithm is to precisely estimate workers' desires for each goal. Given that, we evaluated worker actions with the respect to a modulated optimal path. This path was generated using a hidden Markov model for the static simulated environment and for the real and virtual reality warehouse we used the generalized Voronoi diagram coupled with the D* algorithm. Then we used the resulting motion validation as observations of the hidden Markov model framework to estimate the final probabilities of worker intentions. We have carried out multiple experiments in a simulated, real-world industrial setup using augmented reality glasses and in virtual reality generated warehouses in order to demonstrate the scalability of the algorithm. Results corroborate that the proposed framework estimates warehouse workers' desires precisely and within reasonable expectations.

4

Human motion prediction for human-aware planning

This chapter presents a novel solution to human motion prediction for human-aware planning in the integrated robotized warehouse. One of the major challenges in such a system is planning paths and trajectories for all agents, humans, and robots while retaining operational efficiency and safety. Firstly, we define what is expected human behavior with the respect to designated tasks and propose a method for detecting deviations - the movement not consistent with expected human behavior. An extension of the previously proposed task planner is introduced, with an emphasis on human-aware planning. We describe how the safety levels influence the planning process and which constraints we have taken into account. Afterward, the proposed human intention recognition method is employed for motion planning if the deviation is detected. The set of most probable human trajectories is calculated and forwarded to the human-aware planner that makes decisions on rerouting robots in the warehouse. Finally, the proposed framework is tested with one and three humans in the warehouse and with a varying number of robots. We have tracked the total number of deliveries, responsiveness of the system, number of replanning, and human-robot encounters in the warehouse.

4.1 Introduction

A simulated example of a typical automated flexible warehouse can be seen in Fig.4.1. It consists of large storage racks, packed with goods that are carried by a fleet of mobile robots to designated picking stations, where humans pick goods, pack them, and forward them further for shipment. One of the main problems warehouse management currently faces is that most issues inside the warehouse shopfloor, e.g., a robot malfunction or goods falling on the floor storage racks, require human attention. Furthermore, sometimes it is more efficient for a human worker to carry out the picking of goods if their distribution in the warehouse is too disparate. Given that, there are many times a human intervention in the robotized warehouse is needed, and if this is not carried out in a planner manner it can lead to interruption of the whole warehouse operation, ultimately exacerbating the issue we were trying to solve in the first place. However, human workers do not always behave in a deterministic and prescribed fashion which can affect the carefully orchestrated coordination devised by the robot fleet management system (FMS); hence, to adapt to such

perturbations, a human intention recognition (HIR) system is needed.

The main task of the FMS, which carries out all the robot planning tasks, is to find trajectories between a pair of nodes in the warehouse while taking into account the plans of other robots. The problem of trajectory planning and motion coordination is a well studied problem [144], and since then many approaches have been introduced often based on the classical single-robot planning approaches [145, 146, 60, 147] that provide completeness or even optimality; however, they are not practical for warehouse environments due to their large computational complexity. To counter this issue, another category of sub-optimal planning algorithms has been introduced [148, 149, 150, 151, 152], among which the context-aware route planning (CARP) algorithm, in particular, provides good quality solutions in a warehouse environment with low path computation time and is easily extendable with warehouse specific constraints. The main drawback of the original algorithm was its reliance on the ordering of the agents to be planned. Several heuristic approaches have been introduced in [153] to improve the properties of the algorithm. However, in the warehouse environment, the tasks are not known all at the same time but are given sequentially.

As stated earlier, to ensure warehouse operation efficiency, FMS needs to be complemented with HIR, thus effectively enabling a *human aware planner* (HAP). Given that, when a deviation has happened, HAP should be notified and assisted with the estimation of paths the human might follow – this becomes possible if worker intentions are accurately recognized. We define deviation as a prolonged human movement not consistent with expected behavior with respect to the given goal. The HIR module we select is based on the method proposed in Chapter3and is based on the Bayesian Theory of Mind approach utilizing generalized Voronoi diagrams, a D* graph search, and hidden Markov model architecture. The proposed method observes human behavior and, should they deviate from the assigned path, queries the HIR module for the human goal estimation. These estimates are coupled with expected human velocity to provide the HAP with a set of most probable human trajectories. This enables handling deviations locally, without the necessity to cease the operations of all the robots during human presence inside the warehouse. The HAP reacts by moving the robots out of the human's way or simply stopping the robots. Even though we assume that robots are equipped with a safety system, human-aware planning can reduce the chance of robots driving close to humans thus lowering the number of times the safety system is triggered. In conjunction, this leads to the more efficient operation of the warehouse and hypothetically less stress on the human workers (note that load-carrying robots can weigh up to 1000 kg). We have benchmarked the proposed HIR framework with other methods and recorded an increase in a number of human deliveries by 207%, an increase in total deliveries by 28%, and a reduction of human-robot encounters by 91%.

This chapter is organized as follows. We first introduce the system architecture in Section4.2, laying out details of communication between FMS, HAP, and HIR modules, highlighting the planning method for robots in the warehouse. In Section4.3we describe a novel human deviation detection algorithm, summarize the principle of the HIR module and give detail about trajectory calculation from intention estimates. This information is processed by the HAP for a timely reaction which might include rerouting or stopping robots in the warehouse. In Section4.4we exhaustively test the proposed framework with varying numbers of robots and humans in the warehouse. We have tracked the total

number of deliveries, responsiveness of the system, number of replanning, and human-robot encounters in the warehouse and discussed the results. Finally, the Section4.5summarizes the chapter.

4.2THE FLEET MANAGEMENT SYSTEM

In this section, we will introduce the key FMS components needed for understanding the context of HAP operation. Firstly, we will break down the simulated warehouse environment where humans and robots coexist and execute numerous tasks. We will continue with an in-depth explanation of the relations between FMS, HAP, and HIR modules with a focus on the information they exchange. Afterward, the multi-robot route planning will be presented and we will define conflicting states and safety regions. Finally, we will introduce the details of the planning algorithm for each of the robots in the warehouse.

4.2.1 *The warehouse simulator*

The simulated warehouse shown in Figure4.1is organized into connected nodes and resembles faithfully the software architecture used by an FMS of a true robotized warehouse system. Each node can either be occupied by a robot, a storage rack, or a human. Unloaded robots can move across all free nodes, while loaded robots cannot enter a node already containing a storage rack. As incoming orders arrive, the robot fleet management system coordinates all the robots so that storage racks containing ordered goods are delivered to picking stations, then returned to a free storage node (not necessarily the same as the starting one), and the robots that are idle are sent to the charging nodes. All this needs to be carried out in an efficient manner ensuring continuous operation of the whole warehouse. This planning task is quite challenging since the number of robots can range from 50 to 800, and it gets an additional layer of complexity by having to account for human workers in the area. Note that we assume that robots are equipped with a safety system ensuring that the robots will stop if they come to a range that is too close to the human worker. The multi-robot warehouse simulator was originally presented in [154], while for the current solution we have extended the simulator with the ability to include human worker plan deviation.

4.2.2 *Submodules overview*

The proposed system architecture is shown in Figure4.2. FMS is in charge of planning paths for all the agents in the warehouse and that includes multiple robots and humans. The HIR module serves as part of the FMS with the task of assisting HAP with the uncertainties that human behavior might cause. The HIR has at its disposal information about the position of all the agents in the warehouse, as well as the planned paths of human workers. While the robots are controlled by FMS and always follow the paths given to them, human workers can deviate from their paths for a number of reasons. Because of that, the supervising system can not assume that workers will always follow the path that the FMS gives to them. The job of the HIR module is to detect human deviations from the planned path and estimated the predicted trajectories, based on which the HAP will produce an updated plan for all

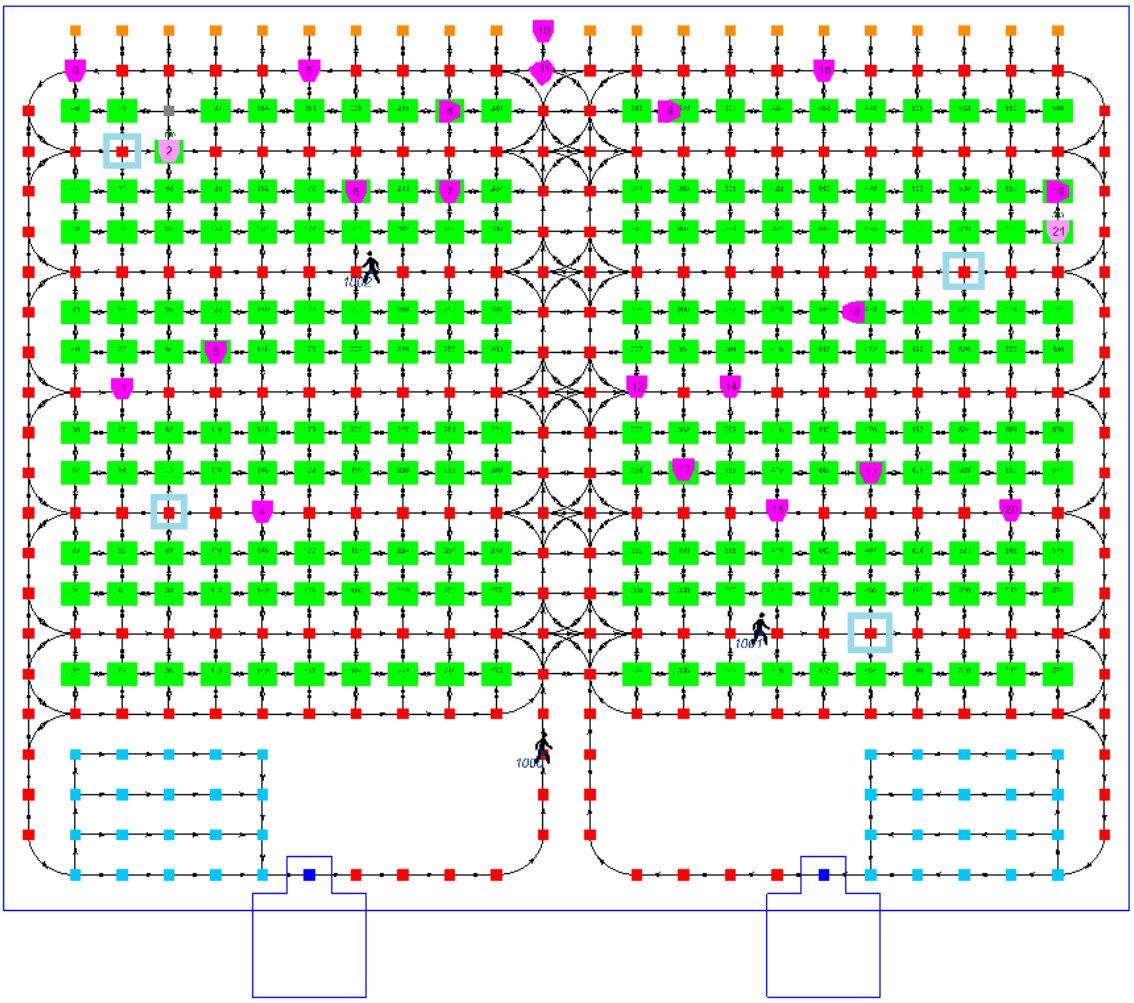


Figure 4.1: The developed flexible warehouse simulator. Mobile robots (pink) do a predefined set of tasks such as carrying storage racks (green) while moving on the ground nodes (red). Three human workers move freely between the storage racks picking goods at specified warehouse locations or doing maintenance work.

the agents. To efficiently assist the HAP, HIR should ascertain (i) if a worker deviated from the original path and (ii) where that particular worker is going to. Given that, the output of the HIR module is a logical flag indicating if there is a worker deviation, followed by a set of probable paths. The HAP then reacts with the method described in Section4.3. In the sequel, we describe each of the proposed system's components in detail.

4.2.3 *Multi-robot route planning*

In this subsection, we leverage the planning method proposed in [154] that is based on the CARP algorithm [148]. The original algorithm structures its map as a resource graph, where each resource has a corresponding timeline. This timeline consists of free and occupied time windows, that indicates whether the resource is available in a given time interval or already taken by a different agent. For each agent that needs planning, the algorithm then finds a free time window on the corresponding resource to the start node of the agent and uses a modified A* algorithm to find the shortest path to the free time window on

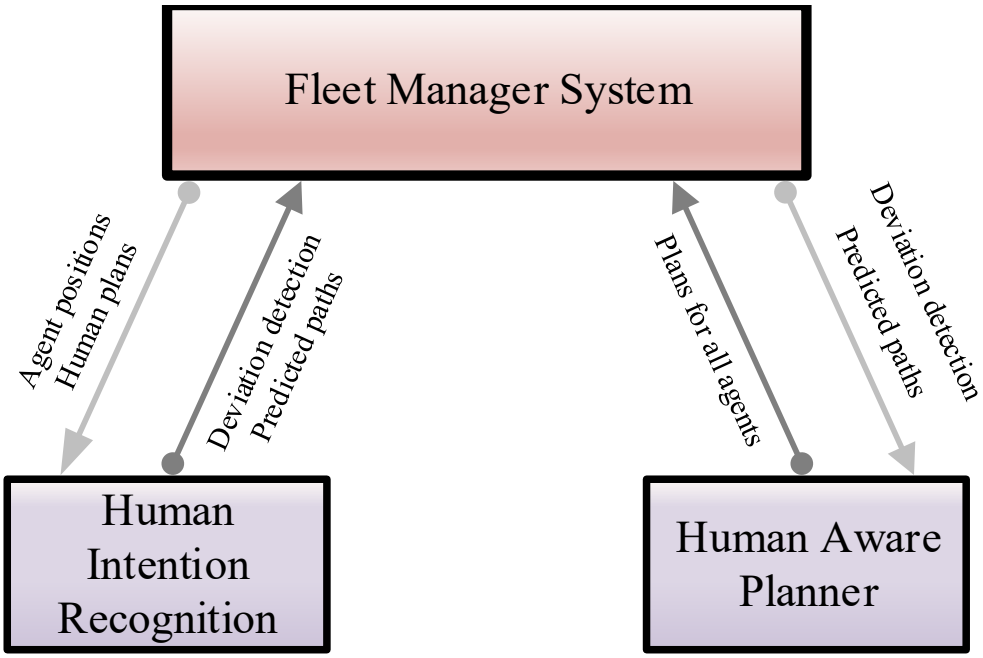


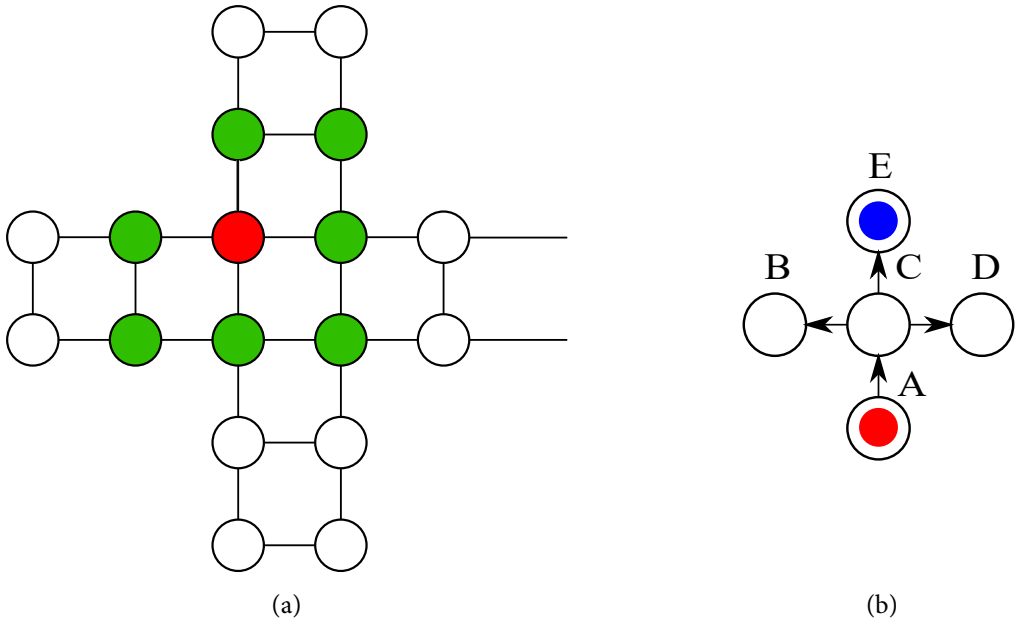
Figure 4.2: Architecture of the proposed system. HIR and HAP are implemented as separate threads of the FMS. While the FMS is responsible for entire warehouse management, we have highlighted only the data flow which is described in the scope of this thesis.

the corresponding goal resource through expansion to the neighboring overlapping time windows. The advantage of the used approach is that it takes into account only a handful of the most influencing agents. The algorithm aims to generate a trajectory for an agent a_k while assuming that trajectories for k-1 agents are already planned. This can lead to modifying those planned trajectories to accommodate the new agent. The main idea is that the algorithm iteratively builds a set of agents whose trajectories affect the optimal trajectory of agent a_k the most and replan this set of agents as well as agent a_k in ordering that yields a solution with the best global cost. Another type of constraint is the inclusion of the safety regions around the robot that differ in their radius and the interaction with robots. There are three different safety regions defined, but only two of them are considered for planning:

- Safety region 1: The robot stops its operation if it enters this radius
- Safety region 2: Planner avoids planning robot in this region
- Safety region 3: The robot must decrease its speed.

As the reader can notice, the safety regions 1 and 2 are identical for the planning algorithm, because the planner needs to avoid planning robots into them, and therefore the algorithm considers only regions 2 and 3.

To accommodate the constraints, the planning algorithm adds an additional timeline to each resource on the resource graph. This timeline is referred to as *conflicting*, with the main timeline referred to as *physical*. The *physical* timeline indicates when the resource is being physically occupied by an agent, while the *conflicting* timeline indicates whether any resource that the current resource conflicts with are being physically occupied. An example of conflicting states can be found in Figure4.3a, while an example of planning with



(a) (b)

Figure 4.3: (a) An example of conflicting states. As long as the robot is standing on the *Red* node, no other robot can stand on any of the *Green* nodes. (b) An example of planning with conflicting states. *Blue* robot wants to go to node *B*, and *Red* robot on node *E*. The node *C* is in conflict with nodes *B*, *D* and *E*. This means that when *Blue* robot attempts to plan to node *B*, the planning fails because even though the *physical* timeline of node *C* is free, the *conflicting* timeline of node *C* shows that it is occupied by the *Red* robot.

conflicting states is shown in Figure4.3b. If these timelines were merged into one, the result would be a timeline where the free time windows are windows that the agent can move into without violating any constraints.

The safety regions for humans are handled similarly, i.e., by keeping timelines *Safety region 2* and *Safety region 3* for each resource and human present in the warehouse, corresponding to time windows that the human plan occupies. Furthermore, the *Safety region 2* timeline is added for each robot as well. The *Safety region 2* is used by merging the *physical* and *conflicting* timelines to obtain the final occupancy timeline of a resource during robot planning. *Safety region 3* occupancy timeline is used to check if the resource intersects the *Safety region 3* during the computation of the time it takes the robot to cross a given resource.

4.2.4 *Planning For Robots*

The proposed planner differentiates between five different states where a robot can be *i)* going to pick up its rack assigned by the job, *ii)* taking the rack to the start of the queue before the goal picking station, *iii)* in the picking station queue, *iv)* heading back to return it to its position or *v)* heading back to its charging station. In addition to these plan states, the robot also has five internal states: *Idle* - The robot is in a charging station, *Busy* - The robot has assigned a job, currently working on its completion, *Free* - No job assigned, returning to a charging station, *Interrupted* - The robot has been interrupted and needs to be replanned and *Failed* - The robot failed to find a plan.

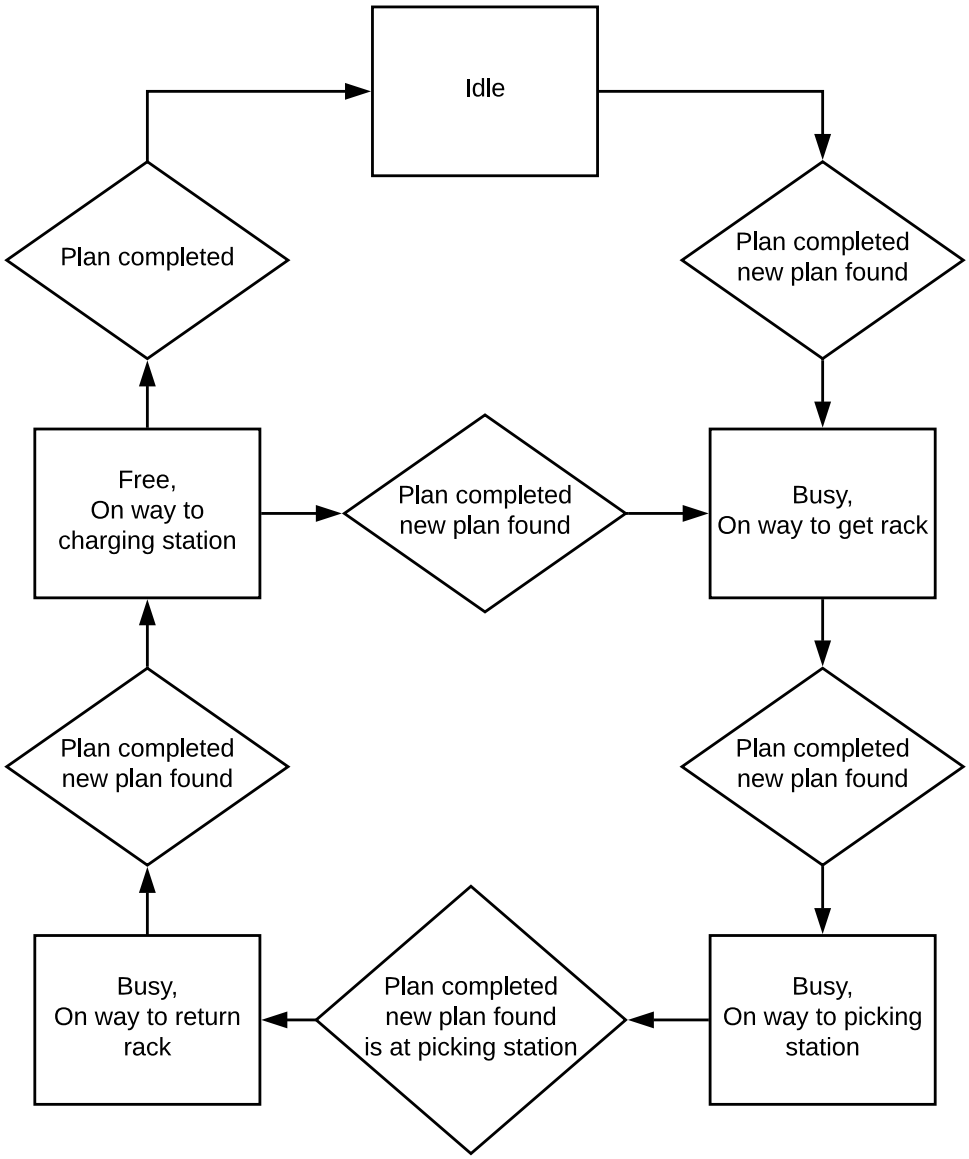


Figure 4.4: The main state diagram of the robot planning.

The detailed diagram of the standard robot operation without human interruption can be seen in Figure4.4which can be described as follows: At the start of the program, each robot starts as being *Idle*. Once a job arrives, the planner decides which robot should be assigned this newly arrived job. If a robot is chosen for the given job, he is assigned a plan to the desired storage unit (rack) and becomes *Busy*. Once the robot arrives there and picks the rack up, it signals that its current job is complete. At this stage, the fleet manager attempts to find the next part of the robot plan. Once the plan is found the robot is set to be in a state that indicates it is on the way to the picking station. Once the robot gets to the picking station node through the queue (note that the robot is not planned directly to the picking station but to the start of the queue, where he is assigned further movement depending on the push and pop operations), it signals that its current plan is complete, and the planner attempts to assign a new plan for it that leads back to the original storage unit; If successful, the robot transitions to the next state that indicates that it's returning the rack. Once the robot gets there and puts the rack down, it is marked as being *FREE* and assigned plan back to its charging station. In his state robot can be assigned a new job immediately. However, if

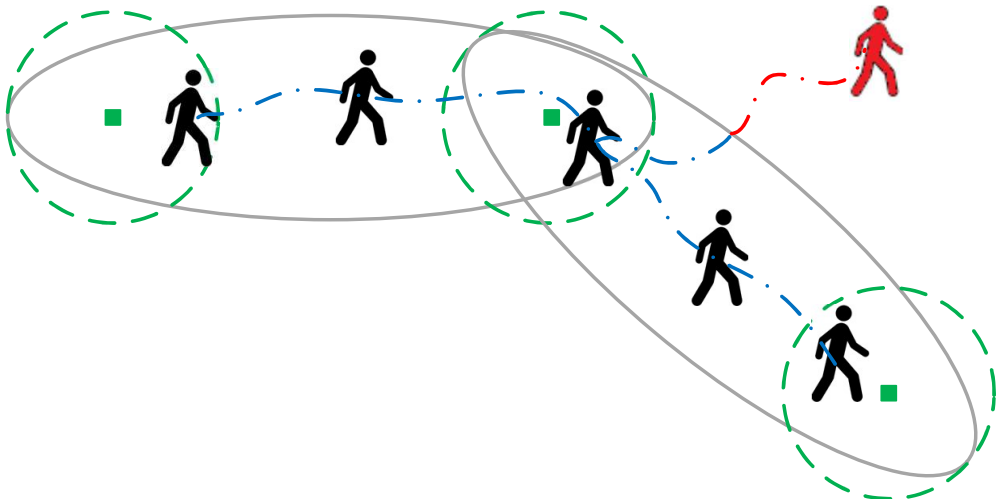


Figure 4.5: The allowed deviation area is depicted by a grey ellipse defined by the current node and next node as focal point (green). Once the worker enters the green circle surrounding the next node, it becomes worker's current node and its successor in the path sequence becomes worker's next node. The example of human path which follows the plan is given with blue color while the path deviating from the plan is red.

the robot arrives back at the charging station it transitions to the original *IDLE* state.

4.3 Human motion prediction

This section introduces the human motion prediction algorithm based on the HMM framework introduced in the previous chapter. We start by discussing human deviations in the warehouse domain and our method for detecting them. After the deviation is detected, the HIR module calculates the probability distributions over pertaining goals and optimal paths towards them. These paths are then used by the HAP to make decisions about rerouting or stopping robots in the way of a human.

4.3.1 *Human Deviation Detection*

Each worker that enters the warehouse has a predefined path that consists of a sequence of ground nodes shown in Figure4.1. We designate the first node of the worker path as the *current node*, and the second node of the path as the *next node*. Every time the distance between the worker and the next node in the path is less than $r = 0.25$ m, that node becomes worker's *current node* and its successor in the path sequence becomes worker's *next node*, until the end of the path is reached. Human beings usually do not walk in a perfectly straight line [155], but swing laterally while moving forward. Given that, we allow deviation from the path defined by the *allowed deviation area* that is described by an ellipse having focal points in the current node and next node, while the major axis is defined as the Euclidean norm between the focal points increased by $2r$. The allowed deviation area is shown in Figure4.5 and the worker is considered to be deviating from the path if it has been outside of the area for at least 4 consecutive cycles.

4.3.2 *Human Path Prediction*

If a worker is detected to be deviating from its predefined path, it is necessary to estimate its future path. In this section, we propose a human path prediction method relying on the method proposed in the previous chapter, where we proposed a human intention estimation method based on hidden Markov model (HMM) motion validation. We will highlight it here with a focus on the changes made for the path prediction problem.

We assume that there is a finite number of possible goal locations, which are usually in front of the storage racks of interest. For current experiments, we selected four auxiliary goals, each one in a different corner of the warehouse and their locations are labeled with turquoise rectangles shown in Figure4.1. It is important to emphasize that the worker is not required to go to the predefined goals, but they do serve as starting points for the proposed algorithm. Also, the last node of the human path provided by the planner is also considered a goal location.

Before the simulation starts, we calculate the distance between all the ground nodes using the D* algorithm [66] and save it in a distance matrix **F**. In case of a robot blocking the edge between two nodes during the experiment, we discard that edge from the graph and recalculate the distance matrix **F**. Because we use road nodes, the search space is reduced significantly and the recalculation can be made in *3.456* ms[1] for 228 nodes and 348 edges of the warehouse road graph. For comparison, if we used a grid map representation with the precision of 10 cm, the recalculation would be done on approximately 2×10^7 nodes and 1.5×10^8 edges which would make the recalculation time larger than one minute, thus rendering it too long for real-time application.

Each time a worker makes a significant displacement, we update its predefined goals intention estimate using a scaled-down version of the algorithm proposed in [82]. We associate the position of the worker with the observable nodes by forming a so-called *association vector* **c**. The closer the human is to the node, the larger the value of the vector **c**. By multiplying **c** and **F**, and by isolating the goal nodes, we obtain a modulated distance vector **d** of dimension g, where g is the number of goals. We also calculate the alternative association vector **c'** of the positions the worker might have gone to if it moved the same distance from the last observation; we also calculate the corresponding modulated distance vector **d'**. By comparing values of the vector **d** with values of each **d'** that we collect in matrix **D**, we calculate the observation vector **o** via element-wise division:

$$\mathbf{o} = \frac{\max\limits_{1 \le i \le n} \mathbf{D}_{ij} - \mathbf{d}}{\max\limits_{1 \le i \le n} \mathbf{D}_{ij} - \min\limits_{1 \le i \le n} \mathbf{D}_{ij}}. \tag{4.1}$$

If the worker is moving towards a goal, the corresponding value of **o** will be close to unity, and if it is moving away from that goal, the corresponding value will gravitate to zero. We record the observation history and process it with an HMM with $g + 1$ states, one for each goal and one for the last node of the human's predefined path. We define the HMM's transition matrix **T** with $\alpha = 0.823$ on the diagonal and $\frac{1-\alpha}{g}$ otherwise. We have obtained this parameter by learning from the recorded data with workers moving in the simulated warehouse without robots and minimizing the average displacement error [156]. Using this

[1] Configuration used for testing: Intel®Core™i7-7700HQ CPU @ 2.80GHz×8 with 15,5 GiB memory

formulation of **T** we allow the worker to change their mind about going towards any of the goals during the experiment. Finally, we set the initial probabilities of worker's intentions to g^{-1} for each goal indicating that all of the goals are equally probable. During the experiment we use the Viterbi algorithm [110] to output probabilities of the worker going to each goal, which we consider as intention estimations.

After recording the probabilities of each goal we query if the probability of the worker going to the last node of the human's path is high enough by comparing it to the largest of the probabilities. If their difference is less than the threshold of $0.25g$, we assume that the worker still might be going to the original goal and we report it to the HAP. Otherwise, we find all goals with the probability higher than the threshold of $0.8g^{-1}$ and using the D* algorithm on warehouse nodes shown in Figure4.1, we find the shortest path towards these goals on the road nodes. These paths are then reported to the HAP.

4.3.3 *Human Aware Planner*

As mentioned in the previous sections, once the human deviates from his planned path, the HIR determined paths to all predefined goals whose probabilities exceed a given threshold. The procedure can be seen in Algorithm6. Once the planner registers paths from the

Algorithm 6: Planner reaction to human deviation

Input: S – a set of predicted paths
Input: R – a resource graph
Input: O – time occupancy of R

```
InterruptAllRobots()
T ← GetLongestCommonSegment(S)
s ← T.begin()
t ← T.end()
S ← Set().insert(T)
P ← PlanUsingNodes(s,t,R,O,S)
if P.empty() then
    P ← Plan(s,t,R, O)
    if !P.empty() then
        O.update(P)
        RedirectHuman(P)
        SendPlanToHIR(P)
    else
        TellHumanToStop()
else
    O.update(P)
    SendPlanToHIR(P)
```

HIR module, it interrupts all robots and finds the longest common path segment of the obtained paths (lines 1-2). This longest common path segment is then processed by the

planner. Notice that if the planner took into account all of the paths, it would possibly block a large portion of the warehouse. Once the segment that the planner will use for the human replanning is known, the system takes the first and last node as start s and goal t locations respectively (lines 3-4). The planner then attempts to find a plan from the start location s to the goal location t using the nodes that were present in the planning segment $\mathcal{T}$ (lines 5-6). If found, the system updates the human path (lines 15-17). If no such path is found, it means that some robot must be standing either directly on the segment of nodes or in a conflicting position. The planner attempts to find a plan to the goal location t in such a situation but without the use of any specific nodes (lines 7-8). If this planning succeeds, the system tells the human that he deviated from his original path and informs him of the new plan (lines 9-12). However, if the planning is not successful, the system tells the human to stop immediately (lines 13-14).

We assume that the worker is equipped with a system, such as a hand-held screen or augmented reality glasses, that can navigate the human through the warehouse. The planning is done in a similar manner as the robot planning; however, human always takes precedence over robots in the planning process. When the human planning starts, all the robots are interrupted. Once all the robots have stopped, the planner attempts to find a path to the human goal destination, while considering stopped robots as obstacles and taking into consideration the *Safety region 2* region where robots should not enter. If such a path is found, it is returned by the system for the human to follow, and the system automatically replans the interrupted robots, while taking the human plan into account. However, if such a path does not exist the system attempts to move the robots out of the way by first planning the human to the goal node, while not taking any of the robots into consideration. The robots then attempt to plan their paths to the closest possible node to their current goal, which is not in conflict with the human path. If the paths for all robots are found, it means that the evasive maneuver is possible and all the paths for humans and robots are returned. If none of these approaches succeed, the system indicates that the planning was unsuccessful. Moreover, to take into account the variance of human velocity, the system also plans the path for the human while taking into account the minimum and maximum velocity. Each resource along the human path is taken for a time interval w_i^r that starts at an entry time t_{entry}^{fast}, the time that would take the human to get to the goal if walking at maximum speed, and ends at the time t_{exit}^{slow}, the time it would take the human to leave if walking at minimum speed. Each time window i for all resources r in the path sequence is then $w_i^r = \langle t_{entry}^{fast}, t_{exit}^{slow} \rangle$.

4.4 Experimental Results

In this section, we will discuss the results of our experimental results. We record the observation history and process it with an HMM model and the Viterbi algorithm outputting probabilities of the worker going to each goal, which we consider as intention estimations. Using the D* algorithm on warehouse nodes in Figure 4.1 we find the shortest path towards each goal. Then, we predict the worker's future motion with respect to goals by interpolating the D* paths using the assumption of constant velocity. Finally, after obtaining the path towards each goal, we find the expected path by weighting each path with the intention

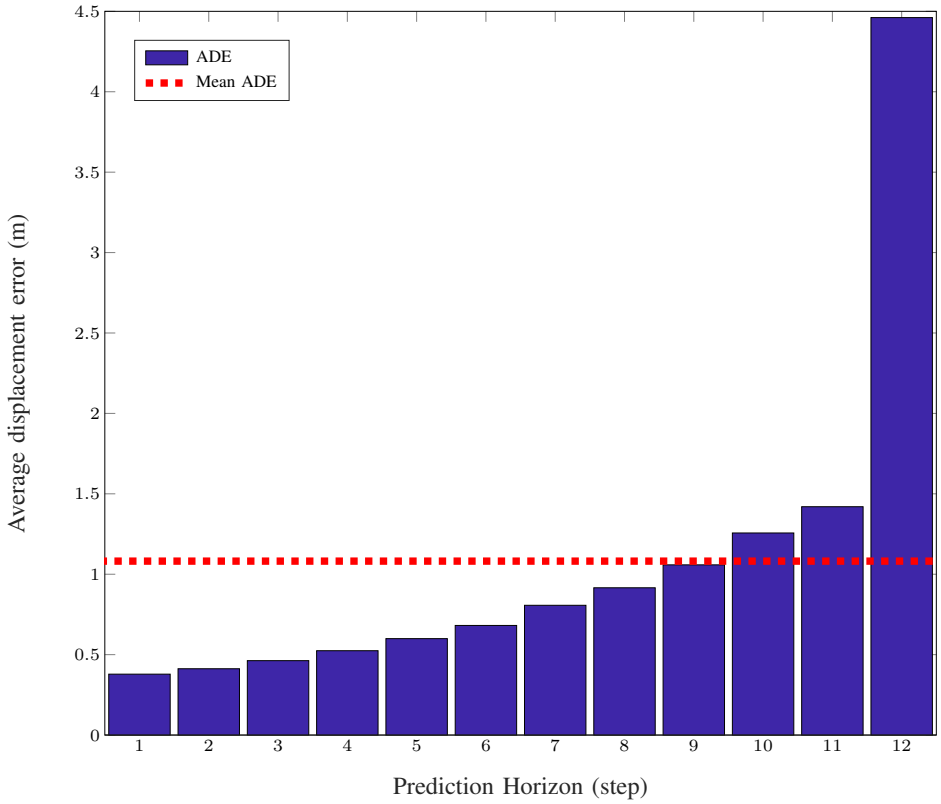


Figure 4.6: Human motion prediction results. We observe past 8 worker locations and predict the next 12 as in [157].

probability towards that goal.

Before testing the entire system, we ran multiple simulations with one human in the warehouse without any deviations. We disabled sending of the planned path to the HIR and observed the predicted trajectory for the next 12 simulation steps. The predicted path is calculated as the probability-weighted average of all paths, fit on the traversable space. The results of the proposed algorithm can be seen in Figure4.6, where we show that the proposed model yields 1.084 m average displacement error (ADE) with respect to the prediction horizon. The results can be compared with linear interpolation which yields 1.1186 m ADE but often predicts untraversable path or unreachable positions because of autonomous robots blocking the path.

To further demonstrate the integrated system functionality, we have designed an experimental setup with several delivering scenarios. All workers had a set of assignments, *e.g.* picking or maintenance, that needed to be completed during the experiment. We measured the average number of deliveries and human-robot encounters for cases when *i)* humans deviate and the planner reacts without the HIR module (NHIR) and replanning is done only when a human enters *Safety region 1*, *ii)* humans deviate and planner reacts using simple HIR module (SHIR) and *iii)* humans deviate and planner reacts using proposed HIR module (PHIR). The SHIR module outputs human path prediction on warehouse nodes with minimal change in heading assuming constant velocity.

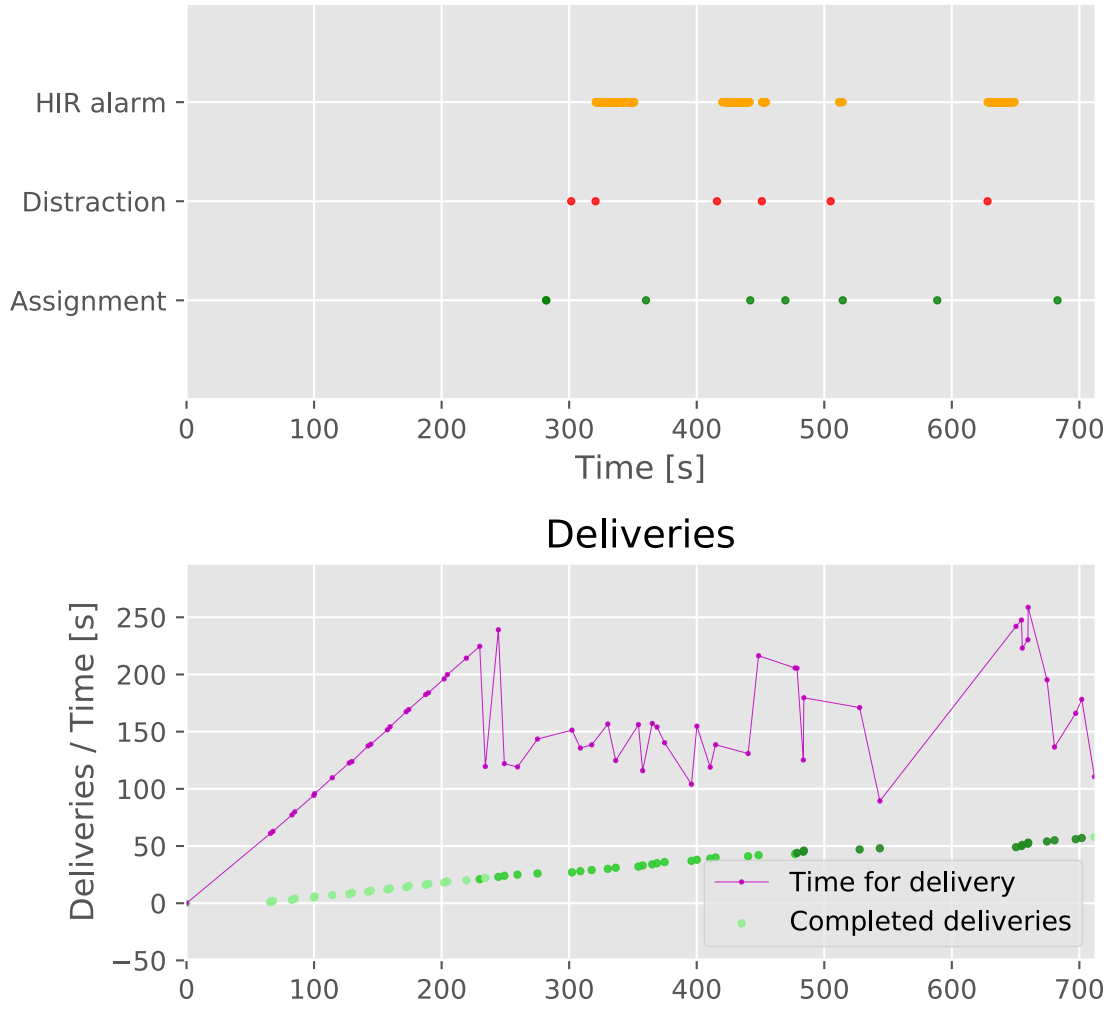


Figure 4.7: Performance of the system with proposed HIR in one of the experimental setups. On top figure one can see times at which human is given assignment (green), when it deviated (red), as well moments in which proposed method signals deviation and sends path prediction to the planner. On the bottom figure we show all the completed deliveries in time with the plot of needed time for each delivery.

We have given humans assignments and simulated deviation multiple times during each experiment with random locations which were not known to the HIR module. The HIR module reacts promptly with alarm and path prediction which is then handled by the HAP, which can be seen in Figure4.7. Once the path is accepted by the HAP and human starts following it, the HIR alarm is turned off.

We have conducted two experimental scenarios, the first with a single human worker and the second with three human workers. The results of ten experiments for each scenario lasting 750 s with unique job sets for all agents can be seen in Table4.1. Specifically, for the single human scenario we have achieved an increase of 28% in human deliveries and 4% in total deliveries, while for the three humans scenario, we have achieved an increase of 18% in robot deliveries, 207% in human deliveries and 29% in total deliveries. The example run showing time needed for each delivery for the SHIR and HIR is shown in Figure4.8 where one can see reduced average as well as maximum time of all deliveries. Results suggest that correct prediction of human intention can improve warehouse throughput

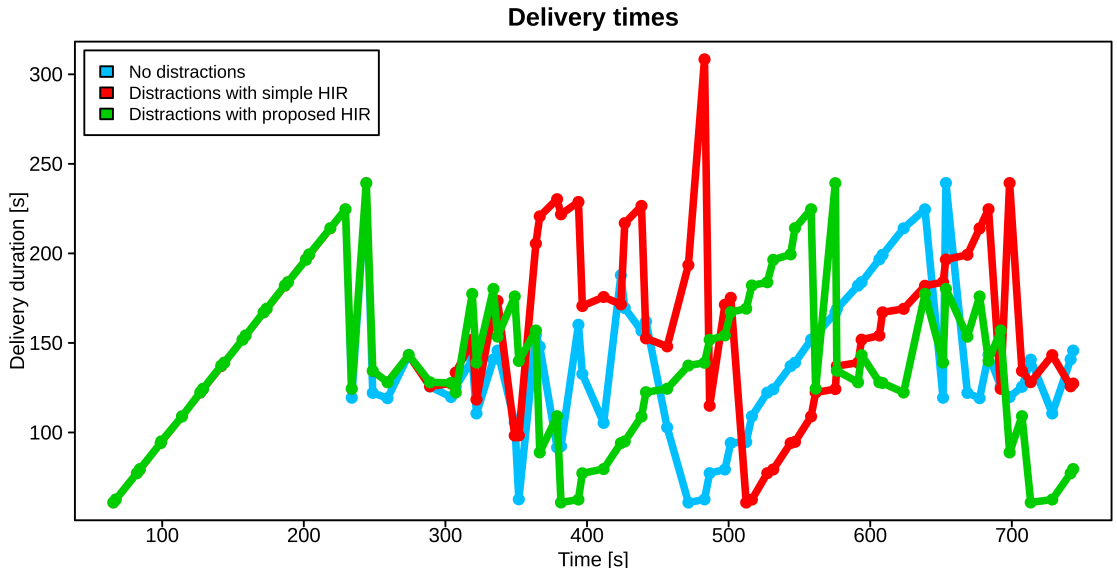


Figure 4.8: Test results. The time needed for each delivery is labeled with filled circle. The human enters the warehouse at $t = 250$ seconds and deviates at $t = 300, 320, 420, 450, 500, 625$ seconds experiment time.

when integrated with a HAP, especially in cases when there are multiple humans operating in the warehouse at the same time. Having in mind the crucial role human workers can have in integrated warehouses, we assert that this result is the main contribution of the current method. Furthermore, an interesting side-effect of the proposed method is the reduction in the number of human-robot encounters. By reducing human-robot encounters, the system

	NHIR	SHIR	PHIR
One Human			
Robot Deliveries	57.9	53.2	**58.1**
Human Deliveries	9.4	8.8	**12.0**
Total Deliveries	67.3	62.0	**70.1**
Human-Robot Encounters / min	5.40	**0.39**	0.49
Three Humans			
Robot Deliveries	48.5	31.7	**57.4**
Human Deliveries	3.0	4.4	**9.2**
Total Deliveries	51.5	36.1	**66.6**
Human-Robot Encounters / min	0.13	0.13	0.13

Table 4.1: Average experimental results for ten experiments lasting 750 seconds.

can hypothetically reduce the discomfort and stress of human workers, since each close encounter with the robot triggers a robot safety stop (loaded warehouse robots can weigh close to 1000 kg). For the single human scenario, the number of encounters was reduced by 91%, while for the three humans scenario this number remained unchanged. The average replanning number during each experiment was also reduced from 10.5 for SHIR to 3.1 for the proposed method. It would be interesting for future work to investigate the behavior of the HIR enhanced HAP with respect to the number of human-robot encounters and the

increasing number of workers in the warehouse.

4.5 SUMMARY

In this chapter, we have presented a human motion prediction method in service of human-aware planning. Human aware planning is planning routes for robots in an integrated warehouse that takes possible future human actions into account. Because of that, we leveraged the human intention recognition module introduced in the previous chapter to find human intention estimates. These estimates were coupled with optimal paths towards each goal and average human moving speed to find motion prediction. We gave an overview of the Fleet Management System modules used for accomplishing this task and introduced details of the Human Aware Planner. We focused on planning routes for robots using a suboptimal planner but accounting for safety levels and regions. The concept of conflicting states governing planner resource allocation was introduced. As the replanning is only done once human deviation is detected, we proposed an algorithm for revealing such behavior. Finally, we have tested the proposed motion prediction algorithm directly by comparing it with an alternative method. We have also compared the performance of the entire system with and without the proposed human-aware planner showing that the proposed solution increases the number of deliveries, and reduces the number of replanning and human-robot encounters.

5

Human action prediction in object picking tasks

This chapter introduces the problem of human action prediction based on their kinematic model with the emphasis on application in the object picking domain. Our main objective is timely and precise prediction of the final goal of human movement. In order to accomplish that, we have relied on probabilistic decision-making methos that use data to tune their parameters. Namely, the models of our choice are the Recurrent Neural Networks (RNNs) and their advanced version Long Short-Term Memory Networks (LSTMs). These models interpret the observed input sequence and yield predictions based on the task at hand. One of the main challenges for deploying such models is necessity of abundant amount of data they are trained on. The recent recording of the MoGaze [10] dataset enabled the human action prediction research community to take advantage of the multiple segments of labeled purposeful human movement. The human movement is captured with a motion capture system using specialized body suits that map the markers to the kinematic model of a human skeleton. These observations are coupled with the measurements from eye gaze capturing device giving plenty of cues for inferring human actions and intentions. Because of that, the human action prediction method proposed in this chapter is based on learning from present data. In this chapter, we have decided to use RNNs and LSTMs for yielding human action predictions as they encompass state-of-the-art time-series prediction models. These models rely on large number of parameters to predict the final goal of human movement. Having that in mind, the proposed method incorporates two feature dimensionality reduction methods which decrease the number of input parameters for the network architectures. One of these methods is based on feature selection via correlation and individual merit while the second one is a feature extraction method inspired by the autoencoder architecture. We have tested the proposed framework on the MoGaze dataset using area under curve, mean squared error and execution time as quality measures. In order to demonstrate the generality of the proposed method, we have recorded the SubMotion, a simpler dataset that consists of hand position and head orientation recordings. The proposed method was exhaustively tesed on both datasets using statistically significant cross-validation and we managed to consistently beat the baselines. Furthermore, we have shown that our method runs fast enough and can use a gaze estimation model for situations where it can not be measured or it would be impractical.

5.1 INTRODUCTION

With the robots becoming more capable and sophisticated, we are witnessing a growth in their presence and integration in private and professional human environments. Nowadays, such environments, besides cohabitation, often include close human-robot collaboration and interaction, yielding novel challenges concerning system efficiency and human safety. While robots are fully controllable, human behavior, on the other hand, although nearly optimal with respect to the task, is inherently stochastic. For example, imagine a healthcare worker treating a patient or a manufacturing shop floor worker assembling products in an agile production system. Their goals are well defined, but the execution and sometimes the environment are not completely controlled. While carrying out the task, the healthcare worker needs to adapt to the responses of the patient, while the worker on a manufacturing shopfloor might change the order of the task execution for justified reasons. We argue that robots in human's proximity should be aware of such changes and react accordingly. Having that in mind, one of the main challenges in collaborative environments is to capture the uncertainty and nuances of human behavior. Supervisory systems try to overcome these challenges by taking advantage of the plethora of methods that revolve around human trajectory prediction, safety regions assertion and action/goal prediction [158, 159, 160, 161, 92].

The problems of human action prediction and intention recognition have come under the spotlight of the research community in recent years. They serve as independent modules or are integrated into the human motion prediction either explicitly [162, 163] or implicitly [164]. The advantages of embedding human intentions implicitly in the model lie in the fact that those models can be trained jointly with the higher-level system and are validated straightforwardly through its performance. The higher-level system could be a fleet management system [82] that tries to reroute the robots out of a human's path and is evaluated by the warehouse deliveries, the number of rerouting, and collision number or a human trajectory prediction model [165] evaluated with the root mean square error of the predicted trajectory. On the other hand, explicitly estimating human actions enables the model to be crafted or trained independently of the higher-level system. In practice, this means that training the action prediction module can be done without the robots operating thus cutting costs. These models can also be interpreted more easily [90], allowing the higher-level system to have semantic meaning and reasoning of performed actions.

In the last few years, multiple datasets concerning motion and action prediction have become publicly available but, to the best of our knowledge, none of them couple these two problems. Examples of purely motion prediction datasets are: ETH [156], KITTI [166] and UCY [167]. We encourage the reader to examine Table 2 in [168] for a detailed listing of the datasets and their descriptions. These datasets, alongside methods trained and evaluated on them [169], offer enough diverse data to train and test human motion prediction models focused on answering the question *"Where is a human going to be during the next N steps?"*, but they are not adequately labeled with the context which would help to answer *"What is (the goal of) the observed human motion?*". On the other hand, datasets tailored for models focused on the second question, like the CMU's motion capture database [170], HumanEva [171] and G3D [172] excel in action diversity, but they are focused on distinguishing be-

tween different actions (jumping, catching, throwing), do not incorporate complicated motion patterns, and usually are not long enough for a long or mid-term human motion prediction problem. The MoGaze [10] dataset positions itself as an excellent blend of the aforementioned datasets because all the recorded motions have a labeled purpose (an object picking). Its subset has already been used by the authors for human motion prediction problems based on RNN networks and trajectory optimization [87, 173]. Therein, they used the Euclidean distance of the right hand to each object as an action prediction signal, improving their original motion prediction result. They also introduced the problem of graspability, which focuses on the exact wrist position at the moment of grasping, and placeability, defined as a probability distribution over possible place locations on a surface the carried object could be placed on. Mentioned models are not evaluated explicitly, but the authors compared a higher-level human motion prediction model's error for different graspability and placeability models thus validating them implicitly.

In this chapter, we propose a novel human action prediction model based on shared-weight LSTM networks. The novelty of the current method lies in the introduction of (i) two feature dimensionality reduction methods, (ii) a novel shared-weight approach to action prediction (iii) a new gaze estimation algorithm, and (iv) creation of a novel dataset that validated our approach as a general method for human action recognition. Similarly to related work, our model relies on the positions and orientations of human joints, recorded by a motion capture system, and on eye gaze captured using a wearable device. In order to reduce the model complexity, we perform feature selection through correlation as well as feature extraction introducing a multilayer perceptron inspired by the autoencoder architecture. The selected features are fed to the ensemble of LSTMs that perform classification and infer the goal of a human. Since eye gaze might not always be available in a real-world scenario, we introduce a neural network-based gaze estimation that serves as an additional input to the proposed method and shows promising results. We have tested our approach on the publicly available MoGaze [10] dataset and published the code with a sample pretrained network. Additionally, we present SubMotion – a simpler dataset that includes six subjects, two female and four male, in object-reaching scenarios similar to the MoGaze. Our dataset records only the head orientation and hand position – a setup that could be easily applied in a real-world application without adding to worker's discomfort or costs. We compared the accuracy of the proposed model with alternatives such as RNN network, fully connected LSTM network, and the strongest individual signal predictors (baselines), based on area under the curve (AUC) score of the predicted goal accuracy and mean squared error (MSE) of the predicted goal location. Our model outperformed all of the baselines and alternative methods in MSE distance on both datasets and had better accuracy on the MoGaze dataset.

This chapter is organized as follows. In Section5.2we describe novel human action prediction method based on shared-weight LSTMs and data from human kinematic chain. This data is sampled from a motion capture system with subjects wearing specialized suits, and using a eye gaze capturing device. We also present a feature selection method based on correlation and a data-driven feature extraction method. Also, we briefly introduce a gaze estimation method. In Section5.3we lay out details of a novel recorded SubMotion dataset. The Section5.4gives overview of the results obtained by thorough testing of the proposed

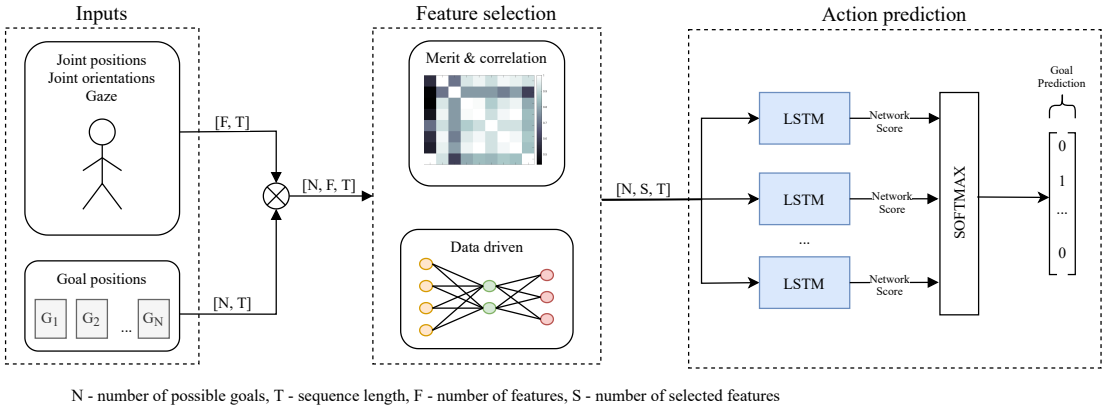


Figure 5.1: Pipeline of the proposed method. Square brackets denote the dimension of the corresponding tensor.

framework. Finally, the Section5.5summarizes the chapter.

5.2THE PROPOSED HUMAN ACTION PREDICTION METHOD

In this section we propose a novel human action prediction model based on shared-weight LSTM networks and feature selection using correlation as well as feature extraction based on the autoencoder architecture. The goal of the proposed model is to ascertain which object in the environment will the human pick next. As we mentioned in the introduction, the creation of the MoGaze dataset with 1435 picking segments including the eye gaze, enabled us to craft a data-driven model for this problem. The segments are labeled with an ID of the object the human is going to pick and serve as ground truth for our framework. We design the proposed action prediction model as a general model for full-body motion that works in real-time and successfully captures relations between input cues and picked objects. Apart from that, we avoid learning specific relations between objects in a dataset. The main reason is that the objects can change their locations during operation and we want our model to handle a varying number of objects in a scene.

Another important aspect that needs to be taken into account by an action prediction model is long-term dependencies since goal inferring cues usually appear much earlier than the actual picking action [9]. For example, imagine a human that intends to pick a specific object from a shelf across the room. Prior to walking to it, they would probably look at that object to ascertain its location and path towards it. While walking, the gaze of the human would not be solely fixed on the object, but could also wander around the scene, especially if there are dynamic obstacles to be negotiated. Given that, a well-designed human action prediction model should take into account the fact that the gaze becomes fixed early in the sequence and can wonder thereafter. In other words, to successfully infer the goal, the model should be able to remember the most important past cue values, e.g., early gaze fixation at the object, as well as capture local tendencies, such as a human approaching the object. To achieve that, we propose multiple LSTM networks with shared weights to serve as the classifier for human action prediction.

However, relying on many inputs adds to the complexity and the network parameter

number, which not only increases the run-time but can also impede the training process by increasing the risk of overfitting. Given that, we further introduce a feature selection method based on signal correlations and individual effectiveness to act as an action prediction cue. To objectively validate our hand-picked selection of features, we also performed feature extraction with a multilayer perceptron (MLP) inspired by the autoencoder architecture and compared the manual selection with a fully data-driven approach.

5.2.1 *Human action prediction framework*

Proposed model fulfills two basic requirements: (i) to be fast enough so that the supervisory system can react in time and (ii) to have good generalization power. To address the latter, we crafted our model so that it can work in a changing environment and handle the addition or removal of objects in the scene. For example, in the MoGaze dataset, the objects are placed on three macro locations: two shelves and a table that do not move during the experiments. If we gave the model distances to all the goals as an input, the model could implicitly learn relations between those macro locations that would not hold should they move during the recording. Also, the number of objects in a scene could change and the transformation of a fully connected LSTM network to accommodate this circumstance would not be a trivial task.

Having that in mind, we decided to approach this problem by training a single classification model and our framework is illustrated in Fig.5.1. For each observed sequence of length T we gather the following input features F: joint positions that are used to calculate Euclidean distances towards each of the N goal positions in the dataset, and gaze and orientation unit vectors that are used to calculate the Euclidean distance between them and the unit vector pointing towards the position of an object. All features are normalized based on the average value in the training set. Each of N sequences is labeled with 1 if it belongs to the object that is eventually going to be picked, otherwise, it is labeled with 0. All the sequences in the training set are aggregated and the dataset is balanced by randomly removing sequences that belong to the "not-a-goal" class. Finally, we train a single LSTM network model for sequence classification with a softmax activation on this data. Note that, during the training, the model does not have access to absolute orientations and positions of the joints or the goals. As a consequence, it learns only if the observed feature sequence (relative to an object) belongs to a pertaining goal or not.

During runtime, we evaluate all selected features for each of the N goals and send them as inputs to N LSTM networks with shared weights (feature selection and extraction are explained in Subsection5.2.2). For the MoGaze dataset, the number of goals was $N = 10$, while for our novel dataset it was $N = 5$. We aggregate outputs of each network via softmax [174] activation function and select the goal whose network has the highest score. This approach enables us to easily add or remove goals if they change during operation which was an important reason behind training only a single LSTM model. Furthermore, by training only a single model that receives relative distances as input and classifies whether that input sequence of features is the pertaining goal or not we remove any contextual environment location information. For example, in the MoGaze dataset, the objects are placed on three macro locations, two shelves, and a table, which do not move during the

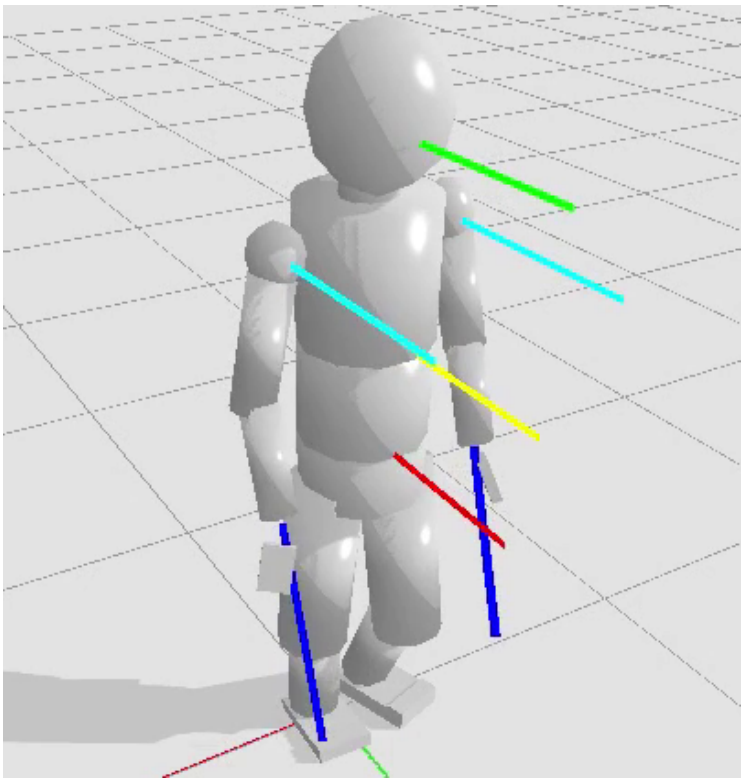


Figure 5.2: Orientations assigned to joints of the human kinematic model. Head, torso, pelvis and shoulders orientation is selected to match forward looking direction while hand orientations are selected to match forearm direction.

experiments. If we give the model, e.g. distances to all the goals as an input, the model could implicitly learn relations between those macro locations that would not hold for other datasets. By utilizing the shared weight concept, we ensure the decision-making process for each goal is the same.

5.2.2 *Feature dimensionality reduction*

For our application, the input features for our model are time series of human joint positions and orientations as well as the eye gaze of the subject. The proposed framework processes these features by numerous matrix additions and multiplications. Each additional input feature adds to the dimensionality of these matrices, thus increasing the number of operations and execution time. Moreover, it could potentially also create the need for increasing the number of hidden dimensions in the network architecture. This is certainly

	Head	Hand	Shoulders	Pelvis	Torso	Gaze	Head	Hand	Shoulders	Pelvis	Torso
Head	1.00	0.93	0.97	0.98	0.97	0.92	1.00	0.91	0.98	0.97	0.96
Hand	0.93	1.00	0.91	0.92	0.93	0.82	0.91	1.00	0.90	0.92	0.92
Shoulders	0.97	0.93	1.00	0.98	0.98	0.82	0.98	0.90	1.00	0.95	0.96
Pelvis	0.98	0.92	0.98	1.00	0.99	0.84	0.97	0.92	0.95	1.00	0.99
Torso	0.97	0.93	0.98	0.99	1.00	0.84	0.96	0.92	0.96	0.99	1.00

Table 5.1: Correlations of selected input features. The Euclidean distances are in the left part of the table, while gaze and orientations are in the right part.

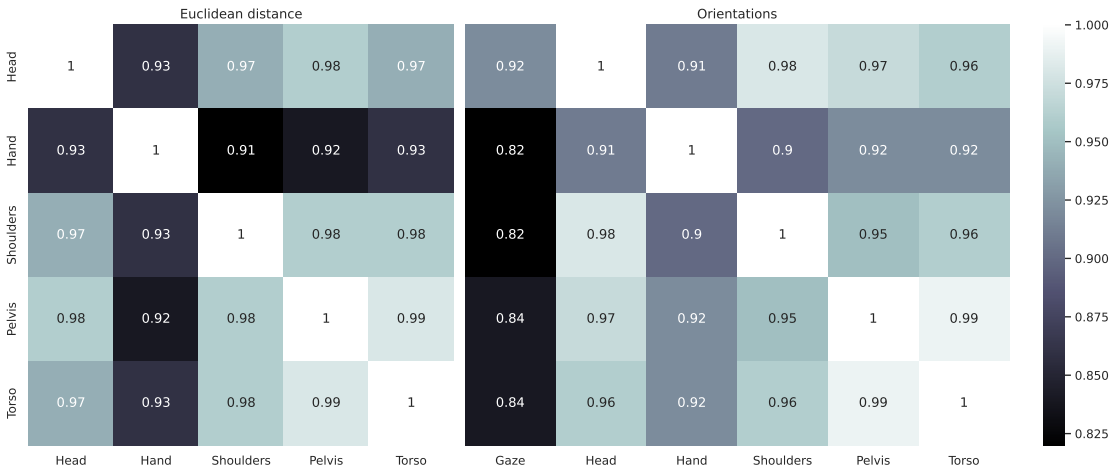


Figure 5.3: Correlations of selected input features. The Euclidean distances are in the left part of the table, while gaze and orientations are in the right part.

an unwanted side effect, not only for previously stated reasons but also due to the limited amount of training data. Having this in mind, we assert that it is important to craft a feature dimensionality reduction method that will indicate which of the recorded joint orientations and positions should be the most relevant inputs to our model. To solve this problem, we took two different approaches.

The first is based on time series analysis - it uses signal correlations to ascertain similarities between features. Our intuition is that features that correlate highly can be substituted by only a single feature from that group. This approach was first proposed by Hall et al. in [134] as used in [134, 136, 137]. In order to choose the most representative feature of the group, we have ranked each feature using the area under curve (AUC) score and selected the highest-ranking feature. The AUC for each feature is calculated for a time span of three seconds (360 frames), as proposed in [10], using an average of accuracy curves on the train set. The accuracy curves are obtained for each feature by checking if the joint is closer to the actual goal than to any other object (Euclidean distance) or if the difference between the orientation vector of a joint and a vector from which that joint sees the object smallest for the goal (orientation distance).

At this point, it is important to clarify the method we used for extracting the joint orientations because the authors give them relative to the humanoid configuration's initial pose, while we need to use them with respect to the world scene. We decided to use a "T-pose" with a human looking towards the x axis as the initial configuration and define all orientations of joints in that pose as $[1, 0, 0]$. This way we ensure that orientations of the head, torso, and pelvis tend to align with the motion direction which we argue is an intuitive way to define orientations of the joints given our application. An example of orientations assigned to joints of interest is shown in Fig.5.2.

Now that we have all set up for feature selection, we calculated correlations between all of the feature distances towards all the goals, and the comparison can be seen in Fig5.3. Euclidean distances of all the joints correlate highly and less so with the hand because of the reaching motion. Orientations of all joints also correlate highly, but less so with the hand orientation. Furthermore, the gaze correlates weakly with all the other features, except

the head indicating that the head orientation could be useful in a gaze estimation problem (when no dedicated gaze tracking equipment is available). The correlation analysis implies that a good subset of features would include the eye gaze, hand position, and orientation of one of the following joints: head, shoulders, pelvis, and torso. We have then proceeded to calculate the AUC score of the proposed input features which we henceforth call baselines since we see each as a potential sole feature for action prediction. Theoretically, a good data-driven model should score better than any single feature, i.e., than any baseline. Finally, the eye gaze scored 155.0, head orientation 71.4, and hand position 83.7, and they were selected as input features for our model. Other baselines scored significantly less than 70. One can notice that we did not analyze joints like toes, knees, and elbows. The main reason is that they correlated poorly with each other and scored very low on the AUC metric which supports our intuition that these joints are of less importance for our application.

The second approach we took is based on autoencoders. Autoencoders are multilayer perceptrons (MLPs) with two main parts: an encoder that maps the inputs to the hidden layer or codes the inputs, and a decoder that reconstructs the input from the hidden layer. If the hidden layer is large enough, the autoencoder can completely recover the input signal at the output. However, in practice, the dimension of the hidden layer is usually much smaller thus forcing the autoencoder to approximate the input by preserving only the most significant information contained within. Because of that, autoencoders are widely used in feature extraction [175,176] and selection [177,178] applications.

We have followed this intuition behind autoencoders and implemented an MLP-based feature extraction. The proposed MLP has an architecture similar to an autoencoder with an input layer that takes all the recorded joint positions and orientations, which is then followed by one hidden layer of a smaller dimension. Finally, the output layer consists of three fully connected neurons, since we wanted to match the number of features used by the correlation-based feature selection method. We tested all commonly used activation functions such as hyperbolic tangent and ReLu [111], and decided to use the sigmoid function as it demonstrated the best performance. Unlike the vanilla autoencoder, our data-driven feature extraction MLP is not trained to match the input data, but is directly connected to the backbone network and trained in an end-to-end fashion. The intuition behind this approach was to enable the training process to refine useful information using data. The extracted input features are composed of a linear combination of all feature input candidates and don't perfectly match any of them. However, by comparing results with the hand-picked features we are able to validate our merit-based approach. The parameter number of the entire model is also reduced because the addition of the fully connected MLP is outweighed by reducing the input dimension of the backbone network.

5.2.3 *Gaze estimation*

Even though the eye gaze has proven to be the most accurate baseline for human action prediction, it might not always be available in real-time practical applications. It requires the user to wear it on their head the whole time, which can be inconvenient and hinder the person's task execution, especially when performing complex tasks. However, the absence of gaze measurements would make our inference with the proposed shared-weight LSTM

networks unviable, since we trained the model to expect gaze in the input along with other motion cues. To alleviate this issue, we propose to estimate the eye gaze from other, more easily obtainable features, such as head orientation and hand position.

Head orientation and hand position can be obtained in real-time from practical wearable sensors, e.g., IMUs mounted on a person's helmet and watch [179, 180]. While the problem of gaze estimation might seem intractable in the general case, due to the human eye gaze presenting an additional degree of freedom compared to the head orientation, we assert that the hand position in collaboration tasks might provide additional information that correlates significantly to the eye gaze. For example, if a person is reaching for an object with their hand, our assumption is that the person will also be looking towards the object in question, thus connecting the gaze to the other motion cues. The proposed estimation procedure relies on having a dataset of human motion while wearing the gaze tracking equipment and then employing a data-driven model to capture the mapping of the subject's head orientation and hand position to their eye gaze. Our gaze estimation model is an MLP that consists of three layers, where the hidden layer is of dimension 10 and the activation function is a rectified linear unit (ReLU). The inputs of our network are hand position and head orientation vectors, while the output of the network is the eye gaze vector. We trained our network using stochastic gradient descent. Once the model is learned, it is utilized during test time to infer the gaze which is then used as an input to the LSTM networks. In practice, this would mean that we can perform a one-time recording of the worker's gaze during collaboration tasks, learn the gaze estimation model using that data, and then perform action prediction during future runs in real-time without requiring the worker to wear the uncomfortable gaze tracking equipment. Potentially, an "average model" could be learned across multiple participants that could generalize well to other people for the same tasks, but this question is out of the scope of current research.

5.3 The novel SubMotion dataset

In order to demonstrate the general application of the proposed algorithm, we have recorded our own dataset which aims to complement the much more comprehensive MoGaze dataset. Unlike the MoGaze dataset, which uses a specialized recording suit and proprietary software to obtain the configuration of the entire human body, our dataset records the positions, and orientations of only two joints: the head and (right) hand. Since it includes only a small subset of human motion features, we dubbed it the SubMotion dataset. Furthermore, the SubMotion setup could be easily embedded in a real-world application without adding to the worker's discomfort. While we have also recorded our data using the OptiTrack system, position of hand and orientation of head could potentially be extracted with wearable sensors in workers' helmets, gloves, or watches. We recorded six times less amount of data than the MoGaze dataset to demonstrate that the proposed algorithm can be trained without the abundant amount of data. This section describes the dataset recording setup and the method we used to obtain the segments we trained and tested our model on.

5.3.1 *Experimental setup*

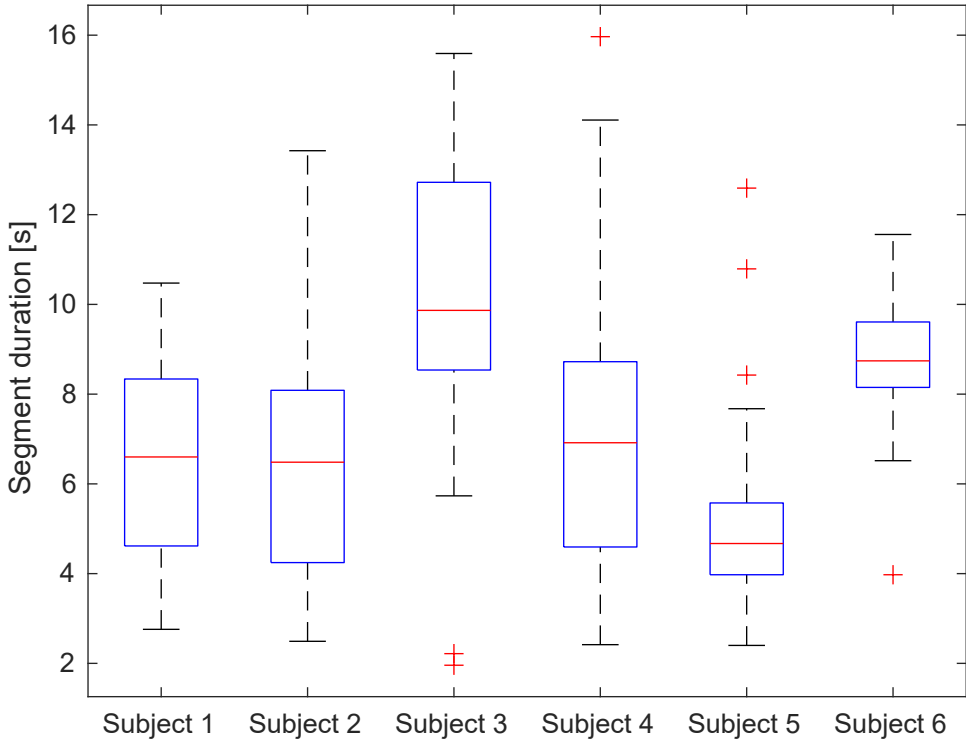
We have used the OptiTrack motion capture system with 12 Flex13 cameras covering the entire workspace. Human participants wore a helmet and a glove with reflecting markers that captured the head and hand locations and orientations. We have chosen a minimal set of wearable equipment which can be easily worn by a worker in a collaborative human-robot scenario without impeding their efficiency or causing discomfort and fatigue. The workspace consisted of three tables on which five objects were placed with an obstacle in the middle. Unlike in the MoGaze dataset, objects in our dataset are static and we don't need to track their position during the recording.

We have recorded the experiment with a total of six subjects, two female and four male, in an object-reaching scenario. Subjects also varied in height, ranging from 155cm to 195cm. Each subject was introduced to the elements of the scene and shown the position of each object. This step was particularly important because the exact positioning of the helmet on the subject's head can vary between subjects. The MoGaze dataset takes advantage of OptiTrack's software for full-body tracking which yields orientation of the head as a property of the obtained human body configuration. In our case, the helmet is defined as a rigid body and its orientation is relative to the orientation the helmet had at the initialization time. Since the head orientation is a crucial input feature for the proposed method, we needed to calibrate its orientation for each subject. We instructed the subject to look at each object at the beginning of the experiment and thus were able to extract reference orientations of the helmet corresponding to each object. This data was used to calculate the helmet's transformation matrix for each subject using MATLAB's ABSOR [181] tool for least-squares estimation of the rotation based on the Horn's [182] quaternion-based algorithm.

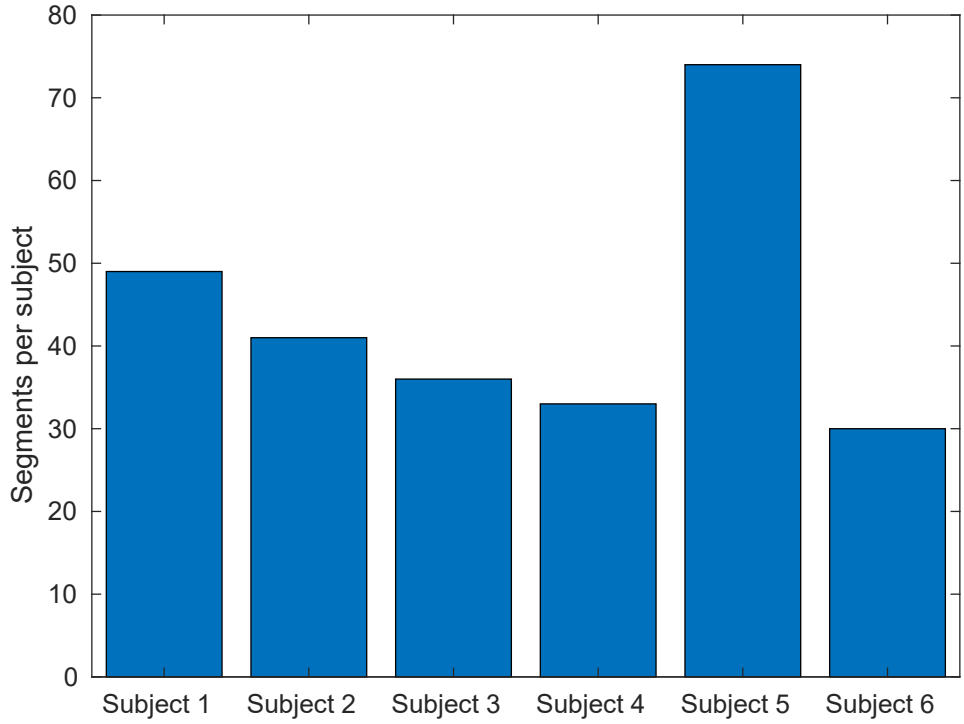
5.3.2 *Dataset Recording*

After the described initialization phase, we recorded two scenarios per subject. In the first scenario, each subject began the recording segment at the same starting point where they waited for the instruction on which object to pick. After the instruction, the subject identified the object, moved to its proximity, picked it, and placed it back on its spot. Subjects were instructed to pick the objects using only the hand that has been recorded. Then they returned to the starting point and waited for the next instruction. We have generated the order of instructions randomly ensuring that each object is picked an equal number of times. In the second scenario, subjects were allowed to walk freely in the scene. Once they decided which object they are going to pick next, they communicated their intention and carried on to execute it as in the first scenario.

We have recorded a total of 30 minutes of data at 120 FPS which is 6 six times fewer than the amount of data present in the MoGaze dataset. The data was split into segments for each subject and the segments were labeled with the object that is eventually going to be picked. The starting point of each segment is when the subject would reach the starting position in the first scenario or when they would communicate the intention in the second scenario. The final point is the moment when the object gets picked. We have analyzed the distribution of segment lengths and the total amount of segments per subject as can be seen in Fig.5.4. We can see that, on average, we recorded more than the three seconds per



(a) Segment durations per subject.



(b) Total amount of segments per subject.

Figure 5.4: The SubMotion dataset analysis. We compensated lower average segment duration of Subject 5 by recording more segments to balance the dataset.

segment for each subject, which is important because the proposed algorithm is evaluated on the last three seconds of each segment. The SubMotion dataset can be made available on request.

5.4 Experimental Results

In this section we present and discuss results of the proposed method on two datasets: the MoGaze and SubMotion. We will discuss multiple network configurations that were tested. In order to compare them, we have used the following three quality measures:
i) Area under Curve: Following our previous work, we continued to use the AUC score as a scalar value representing the accuracy of a model. It is calculated as the average accuracy for each time step in the three-second evaluation window.
ii) Mean Squared Error (MSE) While the AUC score shows in how many frames the proposed method guessed the right goal, it fails to encapsulate how much of the method was when it got the goal wrong. For example, guessing the wrong goal which is 15 cm from the right goal is not the same as guessing the goal which is 1 m away. Having that in mind, we introduce the normalized MSE of the expected goal location for each frame as:

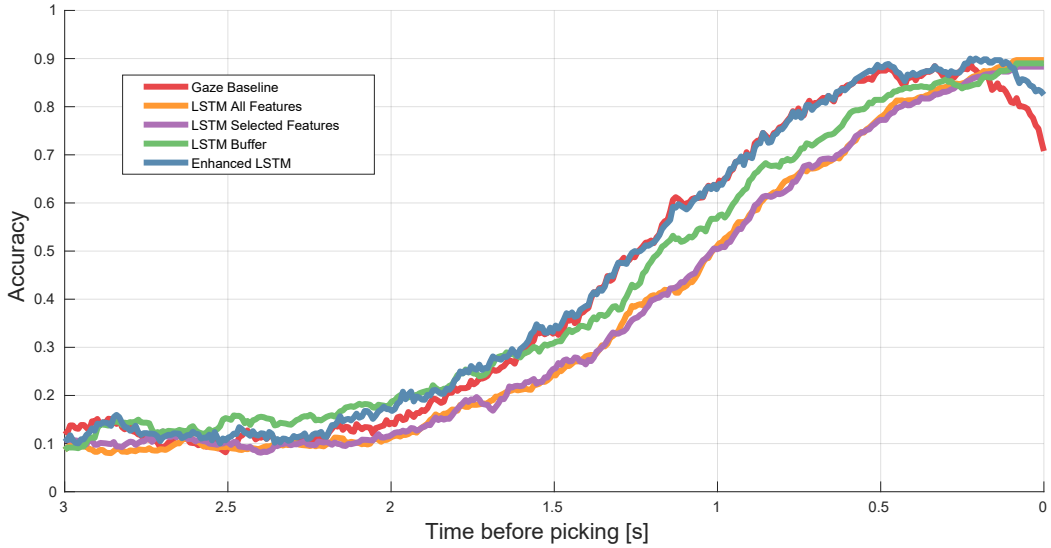
$$\text{MSE} = N\frac{||l_g - \sum_{i=1}^{i=N} p_i l_i||}{\sum_{i=1}^{i=N} ||l_g - l_i||} \tag{5.1}$$

where l_i is the location of i-th object, l_g is the location of the goal object, N is the number of objects and p_i is the probability that the i-th object is the goal and is calculated as the output of the corresponding network divided by the sum of all the network outputs. We use the average distance between objects as the normalization factor.
iii) Execution time: We tracked the average execution time for each of the proposed models to ascertain if they are sufficiently computationally efficient and could enable a potential supervisory system to react accordingly.

We have conducted a series of preliminary tests in Subsection5.4.1to tune in hyperparameters of the proposed framework. Therein we have shown that, on average, the LSTM outperforms the baselines and introduced some implementation details that have improved the result. We have also presented and discussed several examples of the proposed model's inputs and outputs. Finally, the flexibility of the proposed framework is demonstrated by displaying network outputs in the special case of adding and removing goals during the experiment. Further subsections bring the cross-validation method and its results on the MoGaze and newly recorded SubMotion dataset. The gaze estimation method is also thoroughly tested and the results are laid out at the end of this section.

5.4.1 *Preliminary testing*

In this subsection we will describe the preliminary testing we conducted which resulted in selecting some hyperparameters that are used in the rest of this chapter. The MoGaze dataset includes a total of 180 minutes of motion capture data with 1627 pick and place actions [10]. For one participant (participant no. 3) the eye-tracker device did not work so we excluded this session leaving a total of 1435 picking segments. Each group of segments is preceded by the instructions to the participant, e.g. *"Set the table for 2 persons"*, but we do not use this information. Each segment consists of multiple frames before the actual picking happens and a labeled of the object that is eventually picked. Frames in which an object is being carried are discarded. It might be beneficial to additionally discard the parts of segments before the instruction has been given as well as those involving moving of chairs;



	Gaze	LSTM	LSTM Select	LSTM Buff	Enhanced LSTM
AUC	155.02	134.24	134.43	151.98	**159.01**

Figure 5.5: Proposed models' performances. eye gaze has proven to be strongest indicator of human action on this dataset.

however, we decided to leave the dataset intact for easier future comparison. In order to prepare the dataset for model training, we split the data into training and testing datasets. The training dataset consisted of sessions with subjects 1-2 and 4-5 (a total of 853 segments), while the testing dataset contained sessions with subjects 6-7 (a total of 582 segments). We have trained the network using MATLAB with Adam optimization [183] training with 5 epochs and batch size of 5.

First, we trained and tested the proposed network using the whole signal history of all recorded joints' orientations and distances as inputs, including the gaze (we dubbed this version simply as *LSTM*). With the AUC of 134.24, the LSTM model performed worse than the gaze baseline (155.02) during majority of experiments and only succeeded to beat it in the last few moments. On top of that, average prediction time was too slow to work with the aimed frequency of 12 Hz. Therefore, we further tried to improve the results and execution time by reducing the input feature set using correlation and individual merit (dubbed *LSTM Select*). While it slightly improved execution time, the AUC remained unchanged for the LSTM Select model.

In order to reduce the run time, we decided to reduce the complexity by using a buffer to send only the last 20 frames to the network (dubbed *LSTM Buff*). The LSTM Buff model had a satisfying run-time and achieved AUC of 151.98; however, its score was still lower than that of the gaze baseline. As tuning the network hyperparameters mentioned at the beginning of this subsection did not result in any considerable improvements, we decided to further analyze the inputs and the outputs of the proposed model to see where accuracy could be increased and the results are shown in Fig.5.6. For example, in Fig.5.6a), one can see that our model has clearly distinguished which object is the goal after step 250, while Euclidean distance is not the lowest at any point. Also, the gaze signal corresponding with the picked goal has a spike around frame 325. This did not have major effect on the output

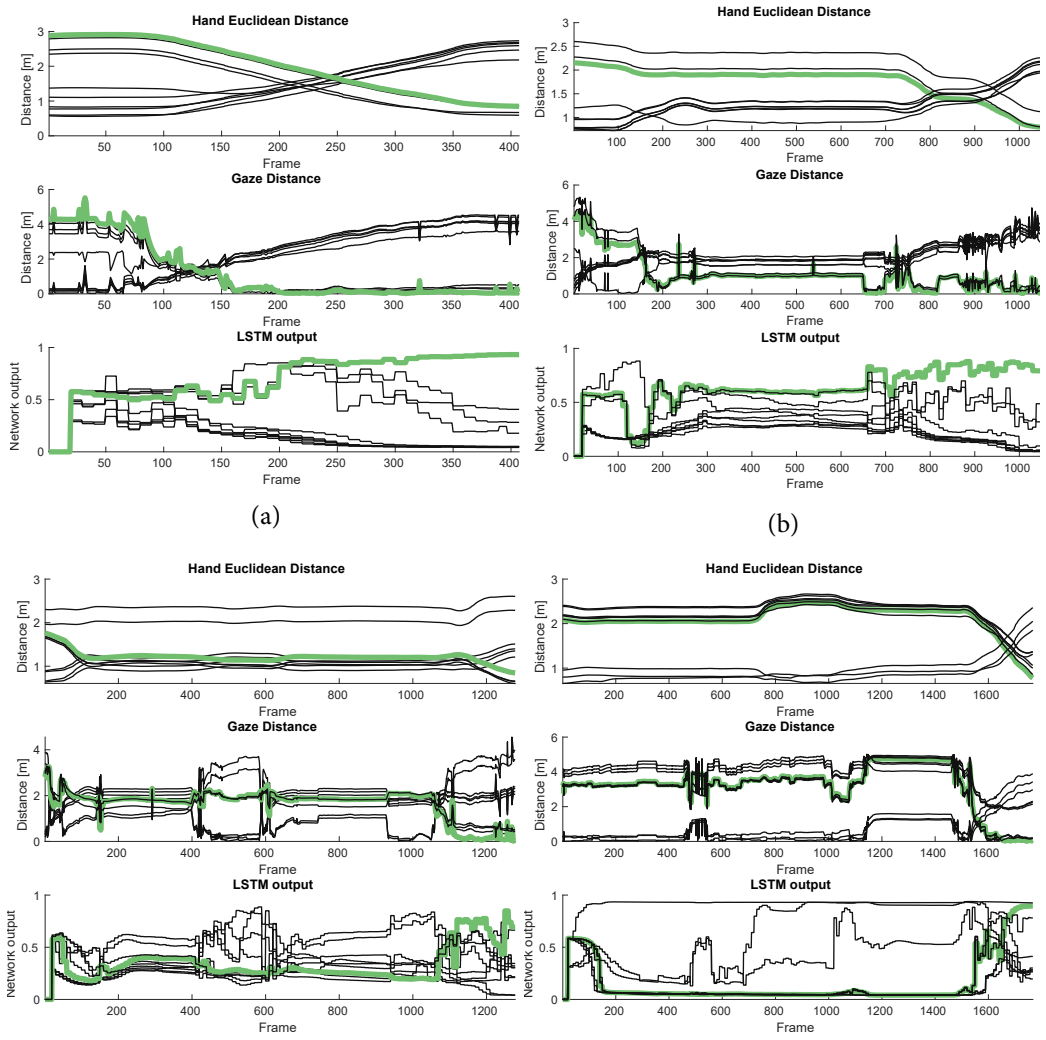


Figure 5.6: Examples of selected model inputs and outputs. The feature corresponding with the object being picked is colored green while the black lines denote features associated with other objects.

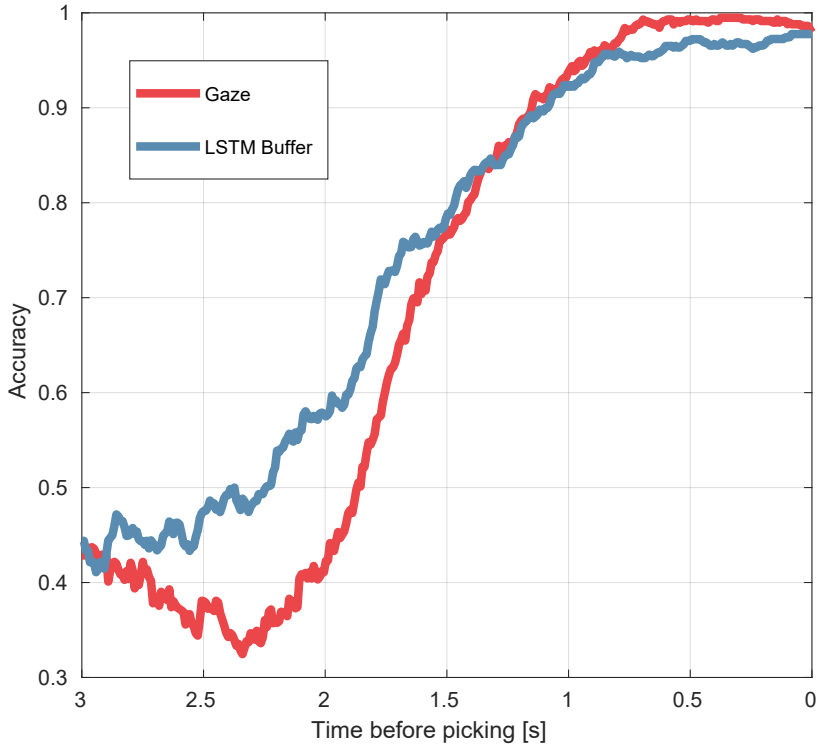
of the proposed model. In Fig.5.6b), the proposed model predicted high probability of the picked object very early but only succeeded to isolate it from nearby objects after step 700. Gaze and Euclidean distance alone could not make such distinction. In contrast, in Fig.5.6c) and Fig.5.6d), proposed model behaved poorly. One possible explanation could be that the subject looked at other objects without moving towards any of the goals, thus not giving the model enough information to conclude the exact goal until the end of the segment.

Given the previous analysis, we can see that the gaze baseline acts as the strongest predictor of the object that the human is going to pick, and furthermore, it can distinguish the actual goal among the nearby objects with pinpoint accuracy – something that our network model struggles with. The fact that the gaze is such a strong indicator of action is not surprising; indeed, visual fixation is necessary for object identification and comes after the brain automatically and in parallel gathers basic features, such as colors, shape, and motion [184]. Given that, we have enhanced our model with a few simple conditions. If the

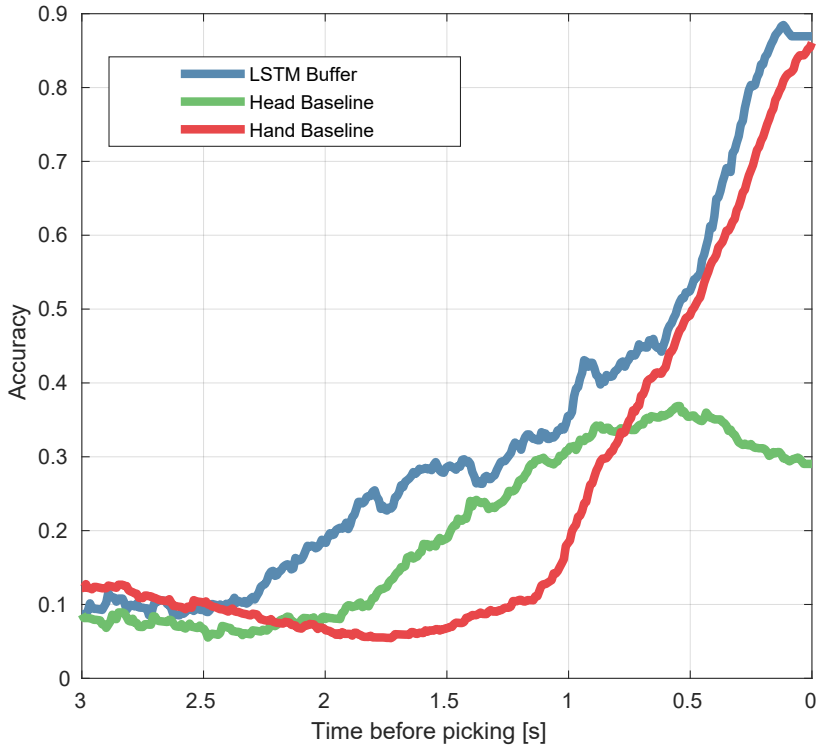
highest LSTM score is larger than the *score threshold*, we consider that object to be the goal. Otherwise, we check if the subject is looking directly at any of the objects, by comparing the minimum gaze distance to the objects with the *gaze threshold*. If the subject is looking at the object, we output that goal, otherwise we select the highest LSTM score. This enhancement led to 159.01 AUC, slightly improving over the gaze baseline by 2.6%. We used the score threshold of 0.49 and gaze threshold of 0.2, which have been selected using grid search on the training set with the AUC as the target function. At the time of writing this thesis, the only method for human action prediction on the MoGaze dataset was published in [10]. The method is based on RNNs but the authors do not provide implementation details rendering a direct comparison difficult. However, by qualitatively comparing the accuracy figures, we can assert that the proposed approach seems to yield more accurate action prediction results.

Although our LSTM models underperformed or showed only a slightly better result than the gaze baseline, we have observed that the LSTMs are better at identifying the macro objects from which the human is going to pick a specific object. Namely, all objects in the Mogaze dataset are placed on three specific macro locations: the table or two shelves. By analyzing the estimated macro location accuracy, the LSTM Buff model scored 266.31 AUC, while the gaze baseline scored 251.31 AUC, as shown in Fig.5.7a). A possible explanation could be that test subjects had to naturally look around while moving towards the goal, preventing them from keeping the gaze fixed on the correct macro location. Our model, on the contrary, managed to successfully leverage the motion cues towards the goal and obtained a higher score in identifying the macro locations.

Since gaze is the most powerful predictor of human action, it begs the question of what kind of performance could be achieved if such a cue was not available? Such a question could be further motivated since presently gaze tracking is done using a cumbersome apparatus that can impede subject's efficiency and comfort in real-life applications. Thus it would be interesting to test what performances could be achieved if the gaze information is not available. We tested the behavior of the proposed LSTM Buff giving only hand and head positions and orientations as input. Because of the reduced number of features, we have increased the number of hidden units to 40 for this model. Our model successfully combined those signals and scored AUC of 118.60 which is considerably higher than the baselines (83.63 and 71.39), as can be seen in Fig.5.7b). One of the main advantages of the proposed framework is its ability to quickly adapt to dynamic and unknown environments. For example, in the collaborative environment, a new interesting item can appear during the operation. Also, some items that have previously been present in the scene can disappear, i.e. they can break, be consumed or become unnecessary. Because of that we have implemented and tested the capability of the proposed framework to handle adding and removing objects (goals) from the scene. Removing the potential goal is done trivially by removing the input connection to the LSTM in Fig.5.1forcing its output to 0. Adding a goal is done in a similar manner, by attaching a new copy of the same LSTM to the action prediction pipeline. The hidden state $\boldsymbol{h}_t$ and cell state $\boldsymbol{c}_t$ of the attached LSTM are inherited from the object closest to the new object at the time of adding. We have also tested initializing the LSTM states with zeros and random values but the proposed method has shown the best results. We have tested adding and removing goals on several experiment runs and the example of one

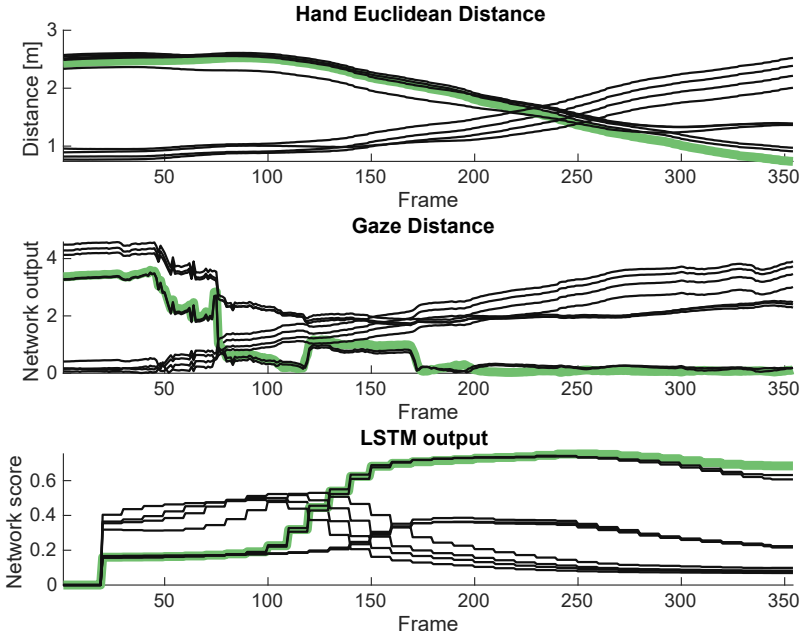


(a) Network performance distinguishing between specific locations.

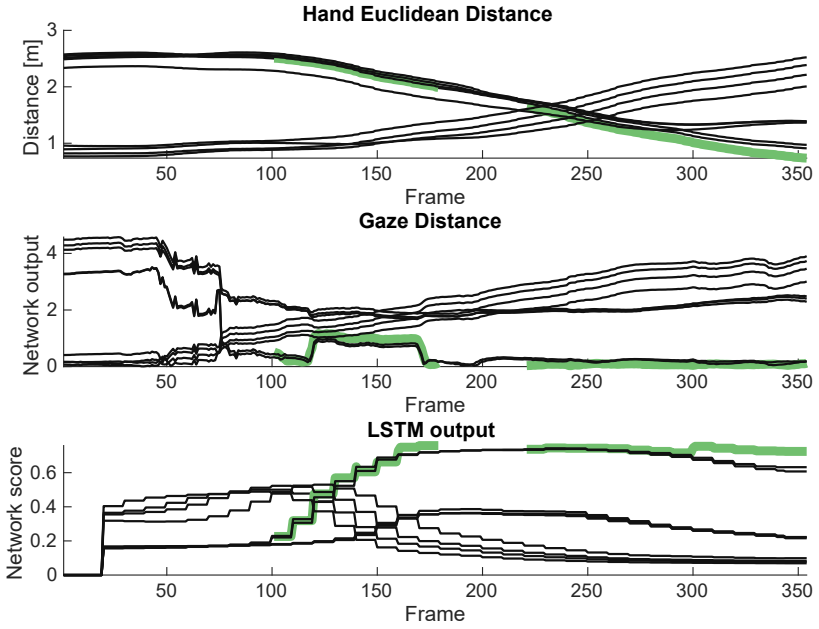


(b) Network performance without eye gaze.

Figure 5.7: The proposed action prediction model has better performance than any baseline if the gaze feature is not available or when one only needs to determine the macro location of the object being picked.



(a)



(b)

Figure 5.8: The goal adding and removing experiment. The top figure shows the output of the network with 10 goals that are not changed during the experiment. The bottom image starts with 9 goals and the eventually picked object is added at frame 100, removed at frame 180 and finally added at frame 220. The network quickly adapted to the addition and removal of the goal and finally succeeded to infer the picked object.

is shown in Fig.5.8.

5.4.2 *Cross-Validation testing*

This subsection lays the foundation of the cross-validation testing conducted for obtaining statistically significant results. Unlike the previous subsection, where we trained a model on data belonging to one half of the subjects and performed testing on the other half, in the current approach we decided to train a unique model for each subject. Such a decision was motivated by observing different motion patterns and data capture quality between subjects, which manifested mostly on the eye gaze. Also, as body proportions, gait and behavior patterns tend to be rather individual, it seams natural to assume that the action prediction models will work better if they are individually trained. our evaluation was performed by k-fold cross-validation in order to demonstrate the statistical significance of our results. In practice, we randomly partitioned the data for each subject in MoGaze dataset into five equally sized subsamples, while the SubMotion dataset was partitioned into three subsamples. Each model was tested on one subsample and training was done on the union of the rest. Such a decision was motivated by observing different motion patterns and data capture quality between subjects, which manifested mostly on the eye gaze. Also, as body proportions, gait and behavior patterns tend to be rather individual, it seems natural to assume that the action prediction models will work better if they are individually trained. our evaluation was performed by k-fold cross-validation in order to demonstrate the statistical significance of our results. In practice, we randomly partitioned the data for each subject in the MoGaze dataset into five equally sized subsamples, while the SubMotion dataset was partitioned into three subsamples. Each model was tested on one subsample and training was done on the union of the rest.

We compared the results of the selected baselines, namely the gaze, head orientation, and hand distance, with the following neural network models:

- LSTMn: denotes multiple LSTM classifiers with shared weights and hidden dimension n. The input for these classifiers are features selected by their individual effectiveness and correlation as in [185]. The best result was achieved for LSTM128.
- MLPn: denotes multiple LSTM classifiers with shared weights and hidden dimension n. The input for these classifiers is all features that first pass through an MLP feature extraction with the hidden dimension of $1-8$. The best result was achieved for MLP128 with a hidden dimension of 2.
- RNNn: denotes multiple RNN classifiers with shared weights and hidden dimension n. The input for these classifiers are features selected by their individual effectiveness and correlation as in [185]. The best result was achieved for RNN128.
- FULLn: denotes an LSTM that takes the selected features for all objects as input and uses a softmax layer to perform classification of the estimated goal. The best result was achieved for FULL32.
- ALLn: denotes an LSTM that takes all available features as input and uses a softmax layer to perform classification of the estimated goal. The best result that was in accordance with the run time constraints was achieved for ALL4.

As one can see, we tested multiple proposed configurations including the multiple RNN and LSTM networks with shared weights as backbones, with and without MLP for feature extraction. The option of using one LSTM with information about the entire scene as an input was explored and we tested for different dimensions of the hidden layer to obtain the best possible result. It is important to note that fully connected action prediction models such as the FULL32 do not have the capability to reduce or expand input dimension. We evaluated baselines by interpreting the inverse of the distance towards each goal as the score at any given time point and treating these scores in the same manner as the network outputs to obtain the prediction. All of the models were trained and tested on the Intel Core i7-7700HQ CPU. The main reason why we decided to use a CPU rather than a graphical processing unit was to show that the proposed model is indeed lightweight and can be easily incorporated into any supervisory system without additional hardware dependencies. The models were implemented in Pytorch and we have made the backbone network publicly available[1]. We trained the model for 100 epochs and used the batch size of 64 for MoGaze and 16 for SubMotion, leveraging Adam [183] optimizer with the learning rate of 0.01. The forementioned parameters were obtained experimentally via exhaustive testing. The following subsections bring the results of testing the proposed framework on MoGaze and SubMotion datasets.

5.4.3 *MoGaze results*

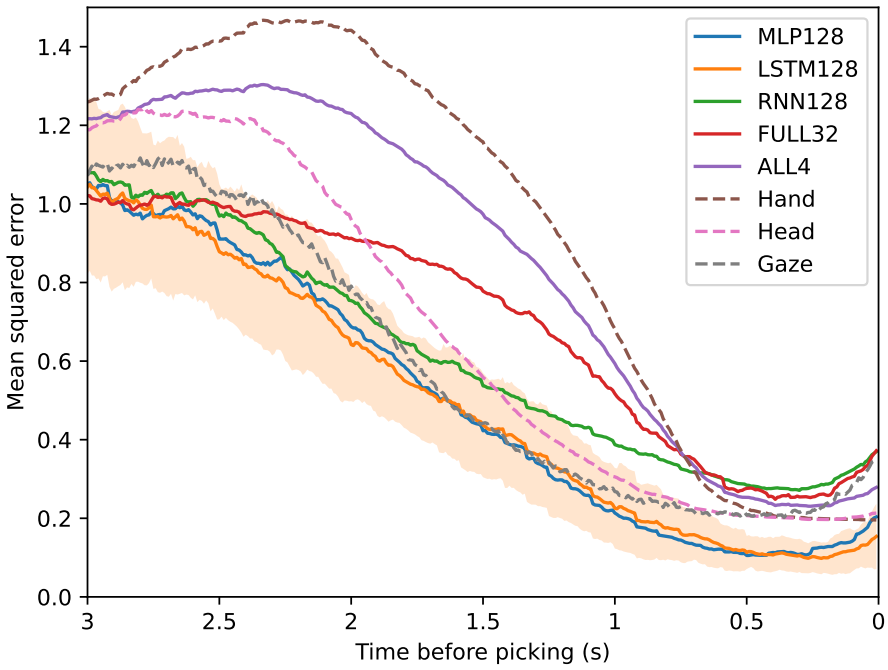
In this subsection we lay out the results of cross-validation on the MoGaze dataset. The results of the 5-fold cross-validation can be seen in the Fig.5.9and Table5.2. The eye gaze has proven to be the best performing baseline with almost double the AUC score compared to the head orientation and hand distance baselines. This result is in accordance with our previous findings which indicated that the gaze baseline acts as the strongest predictor for the object that the human is going to pick, and furthermore, it can distinguish the actual goal among the nearby objects with pinpoint accuracy [185]. The eye gaze also had the smallest MSE but with a smaller margin indicating that subject often fixated their gaze at the goal but was also often browsing around the environment.

	AUC	MSE	Execution Time [ms]
Gaze	168.0	206.3	-
Hand	87.1	339.0	-
Head	96.0	237.8	-
LSTM128	**169.9**	**175.9**	1.7
MLP128	166.6	178.2	2.5
RNN128	149.5	218.2	**0.9**
FULL32	124.4	248.5	2.1
ALL4	102.9	304.6	4.1

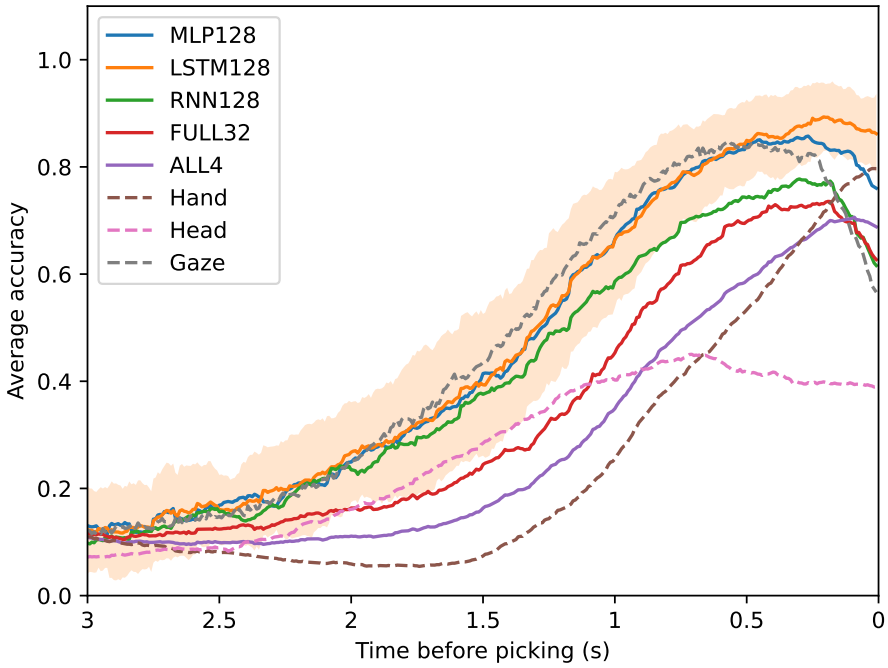
Table 5.2: MoGaze dataset results.

Our proposed model based on the shared-weight LSTM networks outperformed in AUC all the baselines and succeeded to beat the gaze by 1.1%. Following the argument presented

[1] https://github.com/petkovich/ensemble-lstm



(a) Mean squared error of expected goal location before picking the object.



(b) Average accuracy before picking the object.

Figure 5.9: Average values of the MoGaze cross-validation. Additionally, standard deviation of the best performing model (LSTM128) is highlighted.

in [185], we claim that it is hard, if not impossible, to beat the eye gaze significantly in this quality measure. On the other hand, our model had smaller MSE than the gaze baseline by 14.7%. This means that our method, on average, missed the actual goal location by 0.49 times the average distance between all objects while the eye gaze missed it by 0.57. If we

imagine a supervisory system that has to reroute a robot to help the human with executing a task at the goal location, a smaller estimated goal location error could lead to better efficiency of the system. The shared-weight LSTM networks outperformed the shared-weight RNN networks demonstrating LSTM superiority in this time series classification problem. It also achieved a much better result in the full LSTM network, which was expected having in mind that the positions of the objects change during the recording. The use of MLP did not have a positive effect on the result in this case, implying that our feature selection method helped not only to reduce the complexity and execution time of the model, but also to improve the result. Execution times suggest that the proposed framework works at 400 Hz on the CPU which implies it can be seamlessly integrated into the decision-making loop with modern-day sensors. For example, MoCap systems used for recording of both datasets were running at 120 Hz.

5.4.4 *SubMotion results*

We continued with experimental validation of the proposed framework on our SubMotion dataset described in Section5.3. In this case, we compared the network results to the baselines: head orientation and hand distance with models defined as in the previous section using the same abbreviation conventions. Since we have fewer data per subject at our disposal, we decided to reduce the degree of the cross-validation to three. For the same reasons, model complexity has been scaled back and generally, the best performing models had two to four times smaller hidden dimensions than the best corresponding MoGaze model. Also, for this application, the MLP feature extraction had a hidden dimension of 1. The results of the 3-fold cross-validation can be seen in Fig.5.10and Table5.3.

One can notice that the best performing baseline on this dataset is the orientation of the subject's head, similarly to the performance of the gaze on the MoGaze dataset. However, it has proven to be a much stronger cue for action prediction than the same feature in the MoGaze dataset. The exact cause of such a phenomenon is unclear but there are a few possible explanations we would like to mention. Firstly, we calibrate the position of the helmet on each subject individually as described in Subsection5.3.2, while the joint orientations on the MoGaze dataset follow from the predefined human body configuration and are prone to change depending on the exact fit of the marker suit on each subject. Secondly, it is possible that subjects are aware that eye gaze and position of particular joints are measured which can lead to the manifestation of the Hawthorne effect [186] introducing a bias into both datasets. Having that in mind, we proceed carefully with the interpretation

	AUC	MSE	Execution Time [ms]
Hand	129.9	296.2	-
Head	**253.1**	173.5	-
LSTM32	240.9	158.2	1.1
MLP32	240.2	**153.9**	1.2
RNN32	229.0	191.8	**0.7**
FULL16	191.8	197.7	1.9

Table 5.3: SubMotion dataset results.

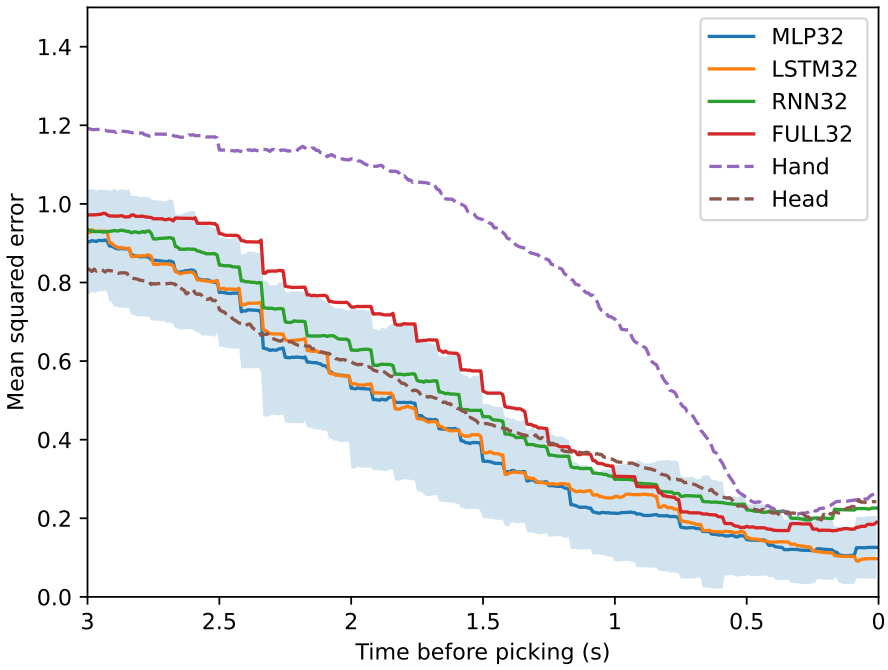
of obtained results.

As the only additional feature for our model, apart from the head orientation, is the Euclidean distance of the right hand from all objects, no method was able to beat the head orientation baseline on the SubMotion dataset. Similar to the MoGaze dataset results, the LSTM model outperformed the RNN model and the shared-weight LSTM networks have performed better than the full LSTM network. On the other hand, the MSE analysis has once again shown the advantages of the proposed model which outperformed the head baseline by 8.8%. The MLP embedding produced an even better result, outperforming the baseline by 11.3%. This is a promising result showing that, although the proposed model is not correct most of the time (lower AUC), it produces smaller errors distance-wise (lower MSE). The full LSTM network performed poorly even though the objects did not move between the segments in this dataset, which additionally justifies the use of the shared-weight method. Compared to the more complicated models used on the MoGaze dataset, execution times were reduced and small enough to ensure real-time operation.

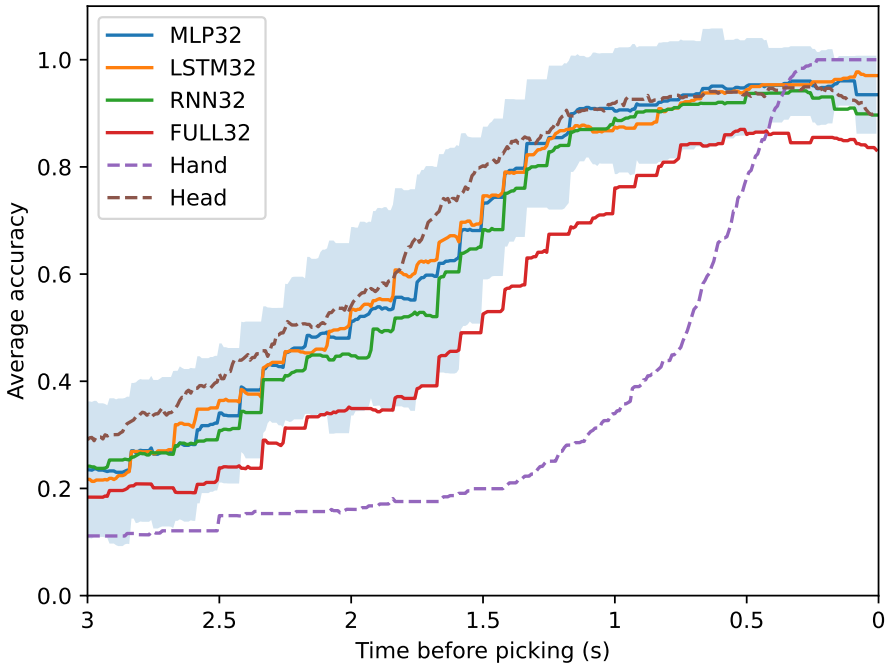
The main motivation for recording the SubMotion dataset was to complement the MoGaze dataset and show that similar results can be achieved using a much smaller and lightweight setup. Furthermore, we wanted to explore the effect of transfer learning approaches by training a single human action prediction model on the MoGaze dataset and testing it on the SubMotion dataset. Unfortunately, we were unable to achieve any sensible result with such an approach which probably follows from the previous argument about differences between the head orientations on these datasets. We tried to leverage gaze in the MoGaze dataset as a comparable signal to the head orientation in the SubMotion dataset but the dynamics of these signals differ a great deal which rendered such an approach invalid.

5.4.5 *Gaze Estimation results*

In this subsection we report the performance of the proposed shared-weight LSTM networks when coupled with the eye gaze estimation procedure proposed in Subsection5.2.3. Evaluating our action prediction LSTM networks model with the estimated gaze, which we dubbed EG-LSTM, required us to partition the MoGaze dataset into three sets. The first set was used for gaze estimation training, meaning that we fed the head orientation and hand position signals as inputs to an MLP proposed in Subsection5.2.3and used the recorded gaze as a supervisory signal during learning. Then we utilized the learned model to estimate the gaze signal on the second and third set from the head orientation and hand position. The second set with the estimated gaze was then used as a training set for the shared-weight LSTM networks, while the third set was used for evaluating the shared-weight LSTM networks with the estimated gaze. This way of partitioning the datasets is transferable to real-world applications. If there exists a pre-recorded dataset that contains gaze measurements, it can be used to train the proposed MLP. Then we can infer the gaze estimates during a person's activity in real-time when the gaze measurement is unavailable and use that data to train and infer with the shared-weight LSTM networks. We compared the average accuracy of EG-LSTM before picking the object with the shared-weight LSTM networks trained with head orientation and hand position as well as the hand, head, and estimated gaze baselines. The results of our analysis are depicted in Fig.5.11. In our evaluation, the EG-LSTM achieved



(a) Mean squared error of the expected goal location before picking the object.



(b) Average accuracy before picking the object.

Figure 5.10: Average values of the SubMotion cross-validation. Additionally, standard deviation of the best performing model (MLP32) is highlighted.

the AUC score of 138.26, outperforming the LSTM trained with head orientation and hand position by 16%. This implies that the estimated gaze signal contained additional information that was utilized in learning the EG-LSTM model to achieve better average accuracy. Its performance matched the ground-truth gaze baseline, having the AUC score within 0.5%,

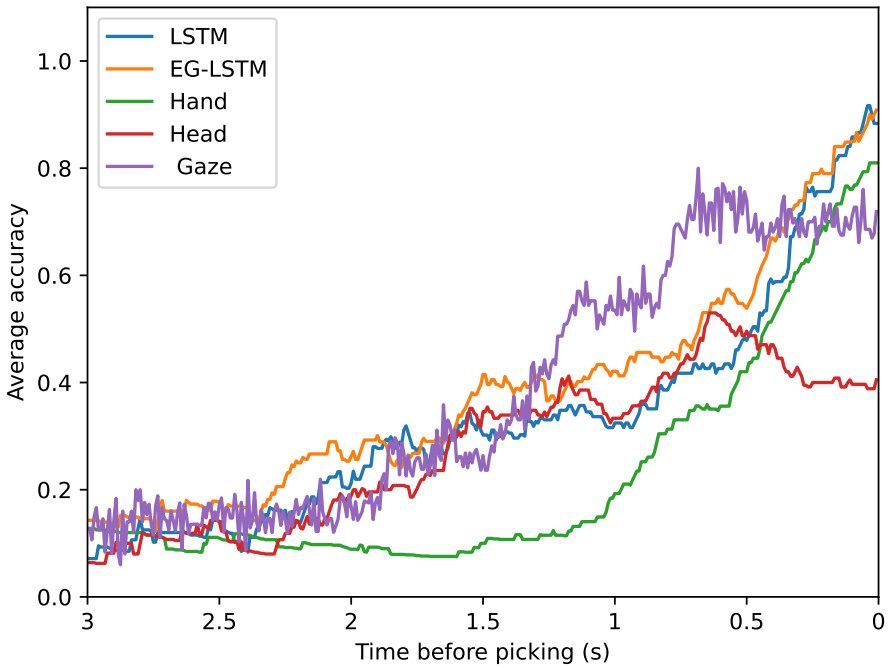


Figure 5.11: Average accuracy before picking the object. The EG-LSTM outperforms the LSTM model without estimated gaze as an input feature.

although the accuracy curves were qualitatively different. Gaze baseline is more accurate at an earlier stage of about 1 s before picking the object, while EG-LSTM was more accurate in the last half of a second before picking. This behavior is consistent with results from earlier sections, where the LSTM relied on the hand position motion cue in close proximity to the goal. Our findings demonstrate that the shared-weight LSTM networks have the potential to work well in specific situations even when human eye gaze measurements are unavailable, which is practical for many real-world applications.

5.5 SUMMARY

In this chapter, we have presented a novel human action prediction framework based on shared-weight LSTMs and feature dimensionality reduction. The input for the proposed method was eye gaze coupled with orientations and positions of joints obtained by recording the kinematic model of a human. Firstly, we have introduced the LSTMs as a potent tool for solving time-series prediction problems, a class of problems to which human action prediction belongs. We continued by laying out the proposed network architecture, giving emphasis on a novel shared-weight approach that enables seamless goal adding and removing while keeping the model complexity lightweight and suitable for a real-time application. Another component that helped reduce the execution time is the proposed feature dimensionality reduction methods. The first feature dimensionality reduction method we introduced was feature selection based on correlation and individual merit. We have grouped features that correlate highly and have chosen the best one from each group as an input to our model. The second method we utilized is a data-driven feature extraction method based on autoencoder architecture. The proposed framework was tested on the MoGaze dataset and our in-house recorded SubMotion dataset. In the SubMotion dataset, we recorded only two joints, the position of the hand and the orientation of the head with the goal of indicating the model's general application. Our testing consisted of showcase examples as well as thorough statistically significant cross-validation where we have successfully beaten baselines on all quality measures. Finally, we have noticed that the eye gaze is the most powerful cue for human motion prediction. Because of that we have implemented and tested the eye gaze estimation method based on MLPs that has shown promising results.

6

Conclusion and Outlook

Recognizing human intentions and subsequently predicting their actions and movement is a task humans execute seamlessly every day. By observing the state of the world, and the actions of others and comparing it to our own decision-making process, we are able to fairly quickly and precisely conclude the motivations and desires of other people. The aim of this thesis is to endow robots, or their supervisory system, with a similar decision-making tool that can mimic the human level of performance, or at least come fairly close to it. As robots become more present in our everyday life, taking into account safety implications brought by human-robot interaction is paramount. In the scope of this thesis, we have decided to focus on industrial collaborative human-robot shared environments. These environments, apart from previously mentioned safety concerns, add to the complexity of human-robot interaction by introducing the efficiency of the entire system as the additional objective of the collaboration.

The thesis started with an overview of complex human-robot environments introducing a distinction between coexistence, collaboration, and cooperation between robots and humans. We continued by laying out the main principles of automated integrated robotized warehouses. These warehouses rely on a fleet of (semi)autonomous robots that carry the racks through the warehouse bringing them to human workers for sorting or inspection. Should a malfunction occur or some other type of maintenance needs to be done, the human worker has to enter the warehouse. As the automated warehouses can be equipped with high-precision measurement equipment, the position of humans can be tracked in real-time. We introduced two sensors utilized in the scope of this thesis. The first one is the motion capture system that can yield positions and orientations of rigid bodies with high precision and in real-time. It is possible to use specialized suits to reconstruct a model human kinematic chain thus mapping rigid bodies to human joints. The second sensor we used is wearable augmented reality glasses that map the environment and give precise location and orientation of the human head. With the constant improvement in this technology, as well as virtual reality, multiple advances in human-robot interaction are made possible. We continued by giving an overview of state-of-the-art human action and motion prediction methods and introduced several mathematical models and frameworks. We discussed path planning and space partitioning, the important tools that enable prediction methods to compare the observed human actions to the optimal behavior with the respect to a goal set. An exhaustive introduction to the probabilistic decision-making models such as the Hidden Markov Model, Recurrent Neural Networks, and Long Short-Term Memory Networks was

given and their strengths and weaknesses have been presented. We gave a comprehensive overview of their use in recent literature related to solving the human action and motion prediction problem.

The first industrial collaborative human-robot shared environment we have put emphasis on is the automated integrated robotized warehouse domain. Unlike robots, human workers in the warehouse are not fully controllable their behavior is stochastic in nature and certain deviations from predefined plans can happen. The warehouse supervisory system, the Fleet Management System, must be able to handle such occurrences and react timely. Having that in mind, we crafted a novel human action prediction method. This method relies on fast planning using the D* algorithm and space partitioning with GVDs. By comparing the observed human behavior with the optimal one obtained by graph search, or alternatively MDP, the model estimates human desires. This kind of reasoning perfectly aligns with the HMM paradigm and we have selected this model for our backbone. The proposed method has been exhaustively tested. Firstly, we created a warehouse simulator demonstrating the effectiveness of the method. We continued by gathering data from an AR device in the real robotized warehouse. As such process is time-consuming and the testing warehouse is rather small, we finalized our tests on much larger VR warehouses with the increased number of robots thus demonstrating the scalability of the proposed method.

The next challenge tackled in the scope of this thesis was human motion prediction. Potential human deviations from the predefined plan can seriously impede warehouse operation. Because of that, we introduced a human-aware planner - a robot route planner expanded with the information on future human motion estimates. For this task, we took advantage of the previously presented human action prediction method. The estimated goal probabilities served as weights for obtaining human trajectory prediction, coupled with the constant velocity assumption. We have also crafted a human deviation detection method that triggers replanning. This approach was tested by calculating the average displacement error of the predicted trajectory. Furthermore, we measured the performance of the entire system and have proven the benefit of the proposed method by observing an increased number of deliveries and a reduced number of unnecessary human-robot encounters.

The third problem we were concerned with was human action prediction based on their kinematic model in collaborative environments. The human kinematic model is reconstructed by observing a human wearing a specialized body suit with infrared markers attached to it. This data is coupled with the recorded eye gaze direction within the MoGaze dataset and presents the input feature set for the proposed method. As the total number of recorded joint positions and orientations is substantial, we started by reducing the feature dimensionality. The first feature dimensionality reduction method we introduced was feature selection based on correlation and individual merit. We have grouped features that correlate highly and have chosen the best one from each group as an input to our model. The second method we utilized is a data-driven feature extraction method based on autoencoder architecture. These features serve as inputs for the proposed action prediction method based on shared-weight LSTMs. The proposed method has successfully beaten the baselines on the MoGaze dataset and on a smaller in-house recorded SubMotion dataset. Furthermore, we have compared its execution with alternative architectures of the network, for example, the RNN and fully connected LSTM. Finally, we have noticed that the eye gaze is the most

powerful cue for human motion prediction. Because of that we have implemented and tested the eye gaze estimation method based on MLPs that has shown promising results.

Naturally, some of the presented methods in the thesis can still be further improved or extended. The human intention prediction in robotized warehouse method heavily relies on the synthetic data and, in some aspects, on manually tuning the parameters of the HMM. This approach could be modified to incorporate fully data-driven learning and parameter tuning. In order to achieve that, access to a fully operational warehouse is necessary. If enough data is collected, more sophisticated state-of-the-art learning-driven frameworks could be employed for precise action and motion prediction. Regarding the action prediction based on the human kinematic model, the proposed method aims to create personalized predictors of human picking actions. This method is crafted with an industrial collaborative environment in mind and a straightforward future work would consist of integrating the current state of the proposed method in such an environment. For example, the proposed human action prediction method can be utilized to timely indicate the potential goal of human movement. This information could be used by a robotic manipulator to do a certain preparatory task with the object human intends to pick, passing it to the human. Such a procedure, coupled with the human kinematic model trajectory prediction method, could potentially lead to increased efficiency of the process at hand.

BIBLIOGRAPHY

[1]Ecommerce Europe. European E-commerce Report 2020, 2022.

[2] Raffaello D'Andrea. Guest editorial: A revolution in the warehouse: A retrospective on kiva systems and the grand challenges ahead. *IEEE Transactions on Automation Science and Engineering*, 9(4):638–639, 2012.

[3] Peter R. Wurman, Raffaello D'Andrea, and Mick Mountz. Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses. *AI Magazine*, 29(1):9, 2008.

[4] Tirthankar Bandyopadhyay, Kok Sung Won, Emilio Frazzoli, David Hsu, Wee Sun Lee, and Daniela Rus. Intention-aware motion planning. In *Springer Tracts in Advanced Robotics*, volume 86, pages 475–491, 2013.

[5] The Anh Han and Luís Moniz Pereira. State-of-the-art of Intention Recognition and Its Use in Decision Making. *AI Communications*, 26(2):237–246, 2013.

[6]Daniel Clement Dennett. *The intentional stance*. MIT press, 1989.

[7] Andrea Bonci, Pangcheng David Cen Cheng, Marina Indri, Giacomo Nabissi, and Fiorella Sibona. Human-robot perception in industrial environments: A survey. *Sensors*, 21(5):1571, 2021.

[8] Thomas Bader, Matthias Vogelgesang, and Edmund Klaus. Multimodal integration of natural gaze behavior for intention recognition during object manipulation. *Proceedings of the 2009 international conference on Multimodal interfaces - ICMI-MLMI '09*, page 199, 2009.

[9] Chien-Ming Huang, Sean Andrist, Allison Sauppé, and Bilge Mutlu. Using gaze patterns to predict task intent in collaboration. *Frontiers in Psychology*, 6(July):1–12, 2015.

[10] Philipp Kratzer, Simon Bihlmaier, Niteesh Balachandra Midlagajni, Rohit Prakash, Marc Toussaint, and Jim Mainprice. Mogaze: A dataset of full-body motions that includes workspace geometry and eye-gaze. *IEEE Robotics and Automation Letters (RAL)*, 2020.

[11] Harish Chaandar Ravichandar, Avnish Kumar, and Ashwin Dani. Gaze and motion information fusion for human intention inference. *International Journal of Intelligent Robotics and Applications*, 2(2):136–148, 2018.

[12] Samuel Dupond. A thorough review on the current advance of neural network structures. *Annual Reviews in Control*, 14:200–230, 2019.

[13] MultiMedia LLC. Definition of robot noun from the oxford advanced american dictionary, 2022.

[14] Sebastian Thrun. Learning Occupancy Grid Maps with Forward Sensor Models. *Autonomous Robots*, 15(2):111–127, 2003.

[15] Sunghyun Cho, Yasuyuki Matsushita, and Seungyong Lee. Removing non-uniform motion blur from images. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[16] Maryamsadat Rasoulidanesh, Srishti Yadav, Sachini Herath, Yasaman Vaghei, and Shahram Payandeh. Deep attention models for human tracking using rgbd. *Sensors*, 19(4):750, 2019.

[17] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[18] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995.

[19] Maria Isabel Ribeiro. Kalman and extended kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics*, 43:46, 2004.

[20] Tom Williams, Daniel Szafir, Tathagata Chakraborti, and Heni Ben Amor. Virtual, Augmented, and Mixed Reality for Human-Robot Interaction. In *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '18, pages 403–404, New York, NY, USA, 2018. ACM.

[21] Ivan E Sutherland. A Head-mounted Three Dimensional Display. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, AFIPS '68 (Fall, part I), pages 757–764, New York, NY, USA, 1968. ACM.

[22] Carolina Cruz-Neira, Daniel J Sandin, Thomas A DeFanti, Robert V Kenyon, and John C Hart. The CAVE: Audio Visual Experience Automatic Virtual Environment. *Commun. ACM*, 35(6):64–72, 1992.

[23] J I Lipton, A J Fay, and D Rus. Baxter's Homunculus: Virtual Reality Spaces for Teleoperation in Manufacturing. *IEEE Robotics and Automation Letters*, 3(1):179–186, 2018.

[24] Jarosław Jankowski and Andrzej Grabowski. Usability Evaluation of VR Interface for Mobile Robot Teleoperation. *International Journal of Human-Computer Interaction*, 31(12):882–889, 2015.

[25] Jordan Allspaw, Jonathan Roche, Nicholas Lemiesz, Michael Yannuzzi, and Holly A Yanco. Teleoperating a Humanoid Robot With Virtual Reality. In *Proceedings of the 1st International Workshop on Virtual, Augmented, and Mixed Reality for HRI (VAM-HRI)*, 2018.

[26] J Zhang, E Langbehn, D Krupke, N Katzakis, and F Steinicke. Detection Thresholds for Rotation and Translation Gains in 360; Video-Based Telepresence Systems. *IEEE Transactions on Visualization and Computer Graphics*, 24(4):1671–1680, 2018.

[27] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Ken Goldberg, and Pieter Abbeel. Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation. *CoRR*, abs/1710.0, 2017.

[28] T S Mujber, T Szecsi, and M S J Hashmi. Virtual Reality Applications in Manufacturing Process Simulation. *Journal of Materials Processing Technology*, 155-156:1834–1838, 2004.

[29] S H Choi and A M M Chan. A Virtual Prototyping System for Rapid Product Development. *Computer-Aided Design*, 36(5):401–412, 2004.

[30] Nirit Gavish, Teresa Gutiérrez, Sabine Webel, Jorge Rodriguez, Matteo Peveri, Uli Bockholt, and Franco Tecchia. Evaluating Virtual Reality and Augmented Reality Training for Industrial Maintenance and Assembly Tasks. *Interactive Learning Environments*, 23(6):778–798, 2015.

[31] Neal E Seymour, Anthony G Gallagher, Sanziana A Roman, Michael K O'brien, Vipin K Bansal, Dana K Andersen, and Richard M Satava. Virtual Reality Training Improves Operating Room Performance: Results of a Randomized, Double-Blinded Study. *Annals of surgery*, 236(4):458, 2002.

[32] Rafael Sacks, Amotz Perlman, and Ronen Barak. Construction Safety Training Using Immersive Virtual Reality. *Construction Management and Economics*, 31(9):1005–1017, 2013.

[33] A P Squelch. Virtual Reality for Mine Safety Training in South Africa. *Journal of the Southern African Institute of Mining and Metallurgy*, 101(4):209–216, 2001.

[34] Abhishek Seth, Judy M Vance, and James H Oliver. Virtual Reality for Assembly Methods Prototyping: a Review. *Virtual Reality*, 15(1):5–20, 2011.

[35] Louis B Rosenberg. The Use of Virtual Fixtures as Perceptual Overlays to Enhance Operator Performance in Remote Environments. Technical report, Stanford Univ Ca Center for Design Research, 1992.

[36] P Milgram, S Zhai, D Drascic, and J Grodski. Applications of Augmented Reality for Human-robot Communication. In *Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on*, volume 3, pages 1467–1472 vol.3, 1993.

[37] Rainer Bischoff and Arif Kazi. Perspectives on Augmented Reality Based Human-robot Interaction With Industrial Robots. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3226–3231. IEEE, 2004.

[38] Andre Gaschler, Maximilian Springer, Markus Rickert, and Alois Knoll. Intuitive Robot Tasks With Augmented Reality and Virtual Obstacles. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 6026–6031. IEEE, 2014.

[39] R T Chadalavada, H Andreasson, R Krug, and A J Lilienthal. That's on My Mind! Robot to Human Intention Communication Through On-board Projection on Shared Floor Space. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–6, 2015.

[40] S Omidshafiei, A A Agha-Mohammadi, Y F Chen, N K Ure, S Y Liu, B T Lopez, R Surati, J P How, and J Vian. Measurable Augmented Reality for Prototyping Cyberphysical Systems: A Robotics Platform to Aid the Hardware Prototyping and Performance Testing of Algorithms. *IEEE Control Systems*, 36(6):65–87, 2016.

[41] Carlos Mateo, Alberto Brunete, Ernesto Gambao, and Miguel Hernando. Hammer: An Android Based Application For End-user Industrial Robot Programming. In *Mechatronic and Embedded Systems and Applications (MESA), 2014 IEEE/ASME 10th International Conference on*, pages 1–6. IEEE, 2014.

[42] C Cadena, L Carlone, H Carrillo, Y Latif, D Scaramuzza, J Neira, I Reid, and J J Leonard. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[43] Dennis Sprute, Klaus Tönnies, and Matthias König. Virtual borders: Accurate definition of a mobile robot's workspace using a rgb-d google tango tablet. *arXiv preprint arXiv:1709.00954*, 2017.

[44] Björn Schwerdtfeger, Rupert Reif, Willibald A Günthner, and Gudrun Klinker. Pick-by-vision: There is Something to Pick at the End of the Augmented Tunnel. *Virtual Reality*, 15(2):213–223, 2011.

[45] J Guhl, S Tung, and J Kruger. Concept and Architecture for Programming Industrial Robots Using Augmented Reality With Mobile Devices Like Microsoft HoloLens. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4, 2017.

[46] Thomas Kirks, Jana Jost, and Benedikt Mättig. Ar for optimization of processes in intralogistics. 06 2017.

[47] K Khoshelham, H Tran, and D Acharya. Indoor mapping eyewear: geometric evaluation of spatial mapping capability of hololens. 2019.

[48] Patrick Hübner, Kate Clintworth, Qingyi Liu, Martin Weinmann, and Sven Wursthorn. Evaluation of hololens tracking and depth sensing for indoor mapping applications. *Sensors*, 20(4):1021, 2020.

[49] Jason Jerald, Peter Giokaris, Danny Woodall, Arno Hartbolt, Anish Chandak, and Sebastien Kuntz. Developing virtual reality applications with unity. In *2014 IEEE Virtual Reality (VR)*, pages 1–3. IEEE, 2014.

[50] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y Ng, et al. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[51] Joshua S Furtado, Hugh HT Liu, Gilbert Lai, Herve Lacheray, and Jason Desouza-Coelho. Comparative analysis of optitrack motion capture systems. In *Advances in Motion Sensing and Control for Robotic Applications*, pages 15–31. Springer, 2019.

[52] Teunis Cloete and Cornie Scheffer. Benchmarking of a full-body inertial motion capture system for clinical gait analysis. In *2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4579–4582. IEEE, 2008.

[53] François Rocca, Thierry Ravet, and Joëlle Tilmanne. Humaface: Human to machine facial animation. *Laboratorie de Theorie des circuits et Traitement du Signal (TCTS), Universite de Mons (UMONS), Belgique, QPSR*, 2012.

[54] A Carrera, SR Ahmadzadeh, A Ajoudani, P Kormushev, M Carreras, and DG Caldwell. Towards autonomous robotic valve turning. *Cybernetics and Information Technologies*, 12(3):17–26, 2012.

[55] Matthew Field, David Stirling, Fazel Naghdy, and Zengxi Pan. Motion capture in robotics review. In *2009 IEEE international conference on control and automation*, pages 1697–1702. IEEE, 2009.

[56] Yiwei Wang, John F Doherty, and Robert E Van Dyck. Moving object tracking in video. In *Proceedings 29th Applied Imagery Pattern Recognition Workshop*, pages 95–101. IEEE, 2000.

[57] Shipra Ojha and Sachin Sakhare. Image processing techniques for object tracking in video surveillance-a survey. In *2015 International Conference on Pervasive Computing (ICPC)*, pages 1–6. IEEE, 2015.

[58] Kinjal A Joshi and Darshak G Thakore. A survey on moving object detection and tracking in video surveillance system. *International Journal of Soft Computing and Engineering*, 2(3):44–48, 2012.

[59] Baoye Song, Zidong Wang, Lei Zou, Lin Xu, and Fuad E Alsaadi. A new approach to smooth global path planning of mobile robots with kinematic constraints. *International Journal of Machine Learning and Cybernetics*, 10(1):107–119, 2019.

[60] Steven M. Lavalle. Rapidly-Exploring Random Trees: A New Tool for Path Planning. *Technical Report (Computer Science Deptartment, Iowa State University)*, 11, 1998.

[61] Steven M LaValle et al. Rapidly-exploring random trees: A new tool for path planning. 1998.

[62] Magda Osman. Controlling uncertainty: a review of human behavior in complex dynamic environments. *Psychological bulletin*, 136(1):65, 2010.

[63] Joe E Heimlich and Nicole M Ardoin. Understanding behavior to understand behavior change: A literature review. *Environmental education research*, 14(3):215–237, 2008.

[64] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[65] Edsger W Dijkstra et al. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[66] Anthony Stentz. Optimal and Efficient Path Planning for Partially Known Environments. *Intelligent Unmanned Ground Vehicles*, (May):203–220, 1997.

[67]Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. 1999.

[68] Richard Bellman. A Markovian decision process. *Journal Of Mathematics And Mechanics*, 6:679–684, 1957.

[69]Ronald A Howard. Dynamic programming and markov processes. 1960.

[70] R Schumacher. *Study for applying computer-generated images to visual simulation*, volume 69. Air Force Human Resources Laboratory, Air Force Systems Command, 1969.

[71] Kun Zhou, Qiming Hou, Rui Wang, and Baining Guo. Real-time kd-tree construction on graphics hardware. *ACM Transactions on Graphics (TOG)*, 27(5):1–11, 2008.

[72] Franz Aurenhammer. Voronoi Diagrams-a Survey of a Fundamental Geometric Data Structure. *ACM Computing Surveys*, 23(3):345–405, 1991.

[73] Howie Choset and Joel Burdick. Sensor-Based Exploration: The Hierarchical Generalized Voronoi Graph. *The International Journal of Robotics Research*, 19(2):96–125, 2000.

[74] Chris L. Baker and Joshua B. Tenenbaum. Modeling Human Plan Recognition using Bayesian Theory of Mind. *Plan, Activity, and Intent Recognition*, pages 1–24, 2014.

[75] Hsien-i Lin and Wei-kai Chen. Human Intention Recognition using Markov Decision Processes. *CACS International Automatic Control Conference*, (Cacs):340–343, 2014.

[76] Truong-Huy Dinh Nguyen, David Hsu, Wee-Sun Lee, Tze-Yun Leong, Leslie Pack Kaelbling, Tomas Lozano-Perez, and Andrew Haydn Grant. CAPIR: Collaborative Action Planning with Intention Recognition. pages 61–66, 2012.

[77] Alan Fern and P. Tadepalli. A computational decision theory for interactive assistants. *Advances in Neural Information Processing Systems 23 (NIPS)*, pages 577–585, 2011.

[78] L Jin, H Hou, and Y Jiang. Driver Intention Recognition Based on Continuous Hidden Markov Model. *Transportation, Mechanical, and ...*, 4(3):9–10, 2011.

[79] Dizan Vasquez, Thierry Fraichard, and Christian Laugier. Growing Hidden Markov Models: An Incremental Tool for Learning and Predicting Human and Vehicle Motion. *The International Journal of Robotics Research*, 28(11-12):1486–1506, 2009.

[80] J Rafferty, C D Nugent, J Liu, and L Chen. From Activity Recognition to Intention Recognition for Assisted Living Within Smart Homes. *IEEE Transactions on Human-Machine Systems*, 47(3):368–379, 2017.

[81] Tarek Taha, Jaime Valls Miró, and Gamini Dissanayake. POMDP-based Long-term User Intention Prediction for Wheelchair Navigation. *Proceedings - IEEE International Conference on Robotics and Automation*, (May):3920–3925, 2008.

[82] Tomislav Petković, David Puljiz, Ivan Marković, and Björn Hein. Human intention estimation based on hidden markov model motion validation for safe flexible robotized warehouses. *Robotics and Computer-Integrated Manufacturing*, 57:182–196, 2019.

[83] Tomislav Petković, Jakub Hvězda, Tomáš Rybeckỳ, Ivan Marković, Miroslav Kulich, Libor Přeučil, and Ivan Petrović. Human intention recognition for human aware planning in integrated warehouse systems. pages 1–6, 2020.

[84] Paul Schydlo, Mirko Rakovic, Lorenzo Jamone, and José Santos-Victor. Anticipation in human-robot cooperation: A recurrent neural network approach for multiple action sequences prediction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–6. IEEE, 2018.

[85] Lei Shi, Cosmin Copot, and Steve Vanlanduit. What are you looking at? detecting human intention in gaze based human-robot interaction. *arXiv preprint arXiv:1909.07953*, 2019.

[86] Yanghao Li, Cuiling Lan, Junliang Xing, Wenjun Zeng, Chunfeng Yuan, and Jiaying Liu. Online human action detection using joint classification-regression recurrent neural networks. In *European Conference on Computer Vision*, pages 203–220. Springer, 2016.

[87] Philipp Kratzer, Niteesh Balachandra Midlagajni, Marc Toussaint, and Jim Mainprice. Anticipating human intention for full-body motion prediction in object grasping and placing tasks. In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1157–1163. IEEE, 2020.

[88] Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2117–2126, 2017.

[89] Richard Kelley, Alireza Tavakkoli, Christopher King, Monica Nicolescu, Mircea Nicolescu, and George Bebis. Understanding human intentions via hidden markov models in autonomous mobile robots. *Proceedings of the 3rd international conference on Human robot interaction - HRI '08*, page 367, 2008.

[90] Jim Mainprice, Rafi Hayne, and Dmitry Berenson. Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 885–892. IEEE, 2015.

[91] Sy Bor Wang, Ariadna Quattoni, L-P Morency, David Demirdjian, and Trevor Darrell. Hidden conditional random fields for gesture recognition. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1521–1527. IEEE, 2006.

[92] Earnest Paul Ijjina and Chalavadi Krishna Mohan. Hybrid deep neural network model for human action recognition. *Applied soft computing*, 46:936–952, 2016.

[93] Cheng Dai, Xingang Liu, and Jinfeng Lai. Human action recognition using two-stream attention based lstm networks. *Applied soft computing*, 86:105820, 2020.

[94] Khan Muhammad, Amin Ullah, Ali Shariq Imran, Muhammad Sajjad, Mustafa Servet Kiran, Giovanna Sannino, Victor Hugo C de Albuquerque, et al. Human action recognition using attention based lstm network with dilated cnn features. *Future Generation Computer Systems*, 125:820–830, 2021.

[95] Zufan Zhang, Zongming Lv, Chenquan Gan, and Qingyi Zhu. Human action recognition using convolutional lstm and fully-connected lstm with different attentions. *Neurocomputing*, 410:304–316, 2020.

[96] Dick de Ridder, Aarnoud Hoekstra, and Robert PW Duin. Feature extraction in shared weights neural networks. In *Proceedings of the Second Annual Conference of the Advanced School for Computing and imaging, ASCI*, pages 289–294. Citeseer, 1996.

[97] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 41(4):801–814, 2018.

[98] Mingjie Liu and John R Kitchin. Singlenn: modified behler–parrinello neural network with shared weights for atomistic simulations with transferability. *The Journal of Physical Chemistry C*, 124(32):17811–17818, 2020.

[99] Lei Shi, Cosmin Copot, and Steve Vanlanduit. Gazeemd: Detecting visual intention in gaze-based human-robot interaction. *Robotics*, 10(2):68, 2021.

[100] Achim Buerkle, William Eaton, Niels Lohse, Thomas Bamber, and Pedro Ferreira. Eeg based arm movement intention recognition towards enhanced safety in symbiotic human-robot collaboration. *Robotics and Computer-Integrated Manufacturing*, 70:102137, 2021.

[101] Shuo Jiang, Qinghua Gao, Huaiyang Liu, and Peter B Shull. A novel, co-located emg-fmg-sensing wearable armband for hand gesture recognition. *Sensors and Actuators A: Physical*, 301:111738, 2020.

[102] Mikel Val-Calvo, José Ramón Álvarez-Sánchez, José Manuel Ferrández-Vicente, and Eduardo Fernández. Affective robot story-telling human-robot interaction: exploratory real-time emotion estimation analysis using facial expressions and physiological signals. *IEEE Access*, 8:134051–134066, 2020.

[103] Daniil Osokin. Real-time 2d multi-person pose estimation on cpu: Lightweight openpose. *arXiv preprint arXiv:1811.12004*, 2018.

[104] Daniel Trombetta, Ghananeel Rotithor, Iman Salehi, and Ashwin P Dani. Variable structure human intention estimator with mobility and vision constraints as model selection criteria. *Mechatronics*, 76:102570, 2021.

[105] Ashwin P Dani, Iman Salehi, Ghananeel Rotithor, Daniel Trombetta, and Harish Ravichandar. Human-in-the-loop robot control for human-robot collaboration: Human intention estimation and safe trajectory tracking control for collaborative tasks. *IEEE Control Systems Magazine*, 40(6):29–56, 2020.

[106] Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[107] C.S. Chen, Y Xu, and Jie Yang. Human action learning via Hidden Markov Model. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 27(1):34–44, 1997.

[108] Kyoung Ho Choi and Jenq-Neng Hwang. Baum-welch hidden markov model inversion for reliable audio-to-visual conversion. In *1999 IEEE Third Workshop on Multimedia Signal Processing (Cat. No. 99TH8451)*, pages 175–180. IEEE, 1999.

[109] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[110] G. David Jr. Forney. The Viterbi Algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[111] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[112] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in neural information processing systems*, 30, 2017.

[113] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[114] Abdul Afram, Farrokh Janabi-Sharifi, Alan S Fung, and Kaamran Raahemifar. Artificial neural network (ann) based model predictive control (mpc) and optimization of hvac systems: A state of the art review and case study of a residential hvac system. *Energy and Buildings*, 141:96–113, 2017.

[115] Ning Qian and Terrence J Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *Journal of molecular biology*, 202(4):865–884, 1988.

[116] MA Rashidan, YM Mustafah, SBA Hamid, NA Zainuddin, and NNA Aziz. Detection of different classes moving object in public surveillance using artificial neural network (ann). In *2014 International Conference on Computer and Communication Engineering*, pages 240–242. IEEE, 2014.

[117] Mohammad Bataineh, Timothy Marler, Karim Abdel-Malek, and Jasbir Arora. Neural network for dynamic human motion prediction. *Expert Systems with Applications*, 48:26–34, 2016.

[118] Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, 7, 1994.

[119] Seymour Gkisser. *Predictive inference: an introduction*. Chapman and Hall/CRC, 2017.

[120] Anil K Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.

[121]Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018.

[122] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

[123] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

[124] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[125] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[126] Graham Williams, Rohan Baxter, Hongxing He, Simon Hawkins, and Lifang Gu. A comparative study of rnn for outlier detection in data mining. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 709–712. IEEE, 2002.

[127] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[128] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.

[129] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971, 2016.

[130] Nils Reimers and Iryna Gurevych. Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. *arXiv preprint arXiv:1707.06799*, 2017.

[131] Amir Ghasemian, Homa Hosseinmardi, and Aaron Clauset. Evaluating overfit and underfit in models of network community structure. *IEEE Transactions on Knowledge and Data Engineering*, 32(9):1722–1735, 2019.

[132] Shigeo Abe. Feature selection and extraction. In *Support vector machines for pattern classification*, pages 331–341. Springer, 2010.

[133] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[134] Mark Andrew Hall et al. Correlation-based feature selection for machine learning. 1999.

[135] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer, 2009.

[136] Yaqing Liu, Yong Mu, Keyu Chen, Yiming Li, and Jinghuan Guo. Daily activity feature selection in smart homes based on pearson correlation coefficient. *Neural Processing Letters*, 51(2):1771–1787, 2020.

[137] Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 856–863, 2003.

[138]SafeLog. Safelog-project, 2018.

[139] Tomislav Petković, Ivan Marković, and Ivan Petrović. Human Intention Recognition in Flexible Robotized Warehouses Based on Markov Decision Processes. In *ROBOT 2017: Third Iberian Robotics Conference*, volume 694, 2018.

[140] Boris Lau, Christoph Sprunk, and Wolfram Burgard. Efficient grid-based spatial representations for robot navigation in dynamic environments. *Robotics and Autonomous Systems*, 61(10):1116–1130, 2013.

[141] Kristian Hengster-Movrić, Stjepan Bogdan, and Ivica Draganjac. Multi-agent Formation Control Based on Bell-shaped Potential Functions. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 58(2):165–189, 2010.

[142] Chun Zhu Chun Zhu, Qi Cheng Qi Cheng, and Weihua Sheng Weihua Sheng. Human intention recognition in Smart Assisted Living Systems using a Hierarchical Hidden Markov Model. *IEEE International Conference on Automation Science and Engineering*, pages 253–258, 2008.

[143] Miroslav Kulich and Roman Sushkov. Speed-up of Self-Organizing Networks for Routing Problems in a Polygonal Domain. (688117):5–6.

[144] L. E. Parker. Path Planning and Motion Coordination in Multiple Mobile Robot Teams. *Encyclopedia of Complexity and System Science*, 2009.

[145] M. R.K. Ryan. Exploiting Subgraph Structure in Multi-Robot Path Planning. *Journal of Artificial Intelligence Research*, pages 497–542, 2008.

[146]Jean-Claude Latombe. Robot Motion Planning. July 1991.

[147] Mike Phillips and Maxim Likhachev. Sipp: Safe interval path planning for dynamic environments. In *2011 IEEE International Conference on Robotics and Automation*, pages 5628–5635. IEEE, 2011.

[148] Adriaan W. ter Mors, Cees Witteveen, Jonne Zutt, and Fernando A. Kuipers. Context-aware route planning. In Jürgen Dix and Cees Witteveen, editors, *Multiagent System Technologies*, pages 138–149, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[149] Dan Halperin Kiril Solovey, Oren Salzman. Finding a needle in an exponential haystack: Discrete rrt for exploration of implicit roadmaps in multi-robot motion planning. *Algorithmic Foundations of Robotics XI*, pages 591–607, 2014.

[150] M Peasgood, C M Clark, and J McPhee. A Complete and Scalable Strategy for Coordinating Multiple Robots Within Roadmaps. *Robotics, IEEE Transactions on*, 24(2):283–292, April 2008.

[151] Michal Cap, Peter Novak, Alexander Kleiner, Martin Selecky, and Michal Pechoucek. Prioritized Planning Algorithms for Trajectory Coordination of Multiple Mobile Robots. *IEEE Transactions on Automation Science and Engineering*, Special Is, 2015.

[152] Jakub Hvězda, Miroslav Kulich, and Libor Přeučil. Improved discrete rrt for coordinated multi-robot planning. *arXiv:1901.07363*, 2019.

[153] A. ter Mors. Evaluating heuristics for prioritizing context-aware route planning agents. In *2011 International Conference on Networking, Sensing and Control*, pages 127–132, April 2011.

[154] Jakub Hvězda, Tomáš Rybeckỳ, Miroslav Kulich, and Libor Přeučil. Context-aware route planning for automated warehouses. In *International Conference on Intelligent Transportation Systems (ITSC)*, pages 2955–2960, 2018.

[155] Catherine E Bauby and Arthur D Kuo. Active control of lateral balance in human walking. *Journal of biomechanics*, 33(11):1433–1440, 2000.

[156] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *IEEE International Conference on Computer Vision (ICCV)*, pages 261–268, 2009.

[157] Stefan Becker, Ronny Hug, Wolfgang Hübner, and Michael Arens. An evaluation of trajectory prediction approaches and notes on the TrajNet benchmark. *arXiv:1805.07663*, 2018.

[158] Murilo Varges da Silva and Aparecido Nilceu Marana. Human action recognition in videos based on spatiotemporal features and bag-of-poses. *Applied Soft Computing*, 95:106513, 2020.

[159] Antti Hietanen, Roel Pieters, Minna Lanz, Jyrki Latokartano, and Joni-Kristian Kämäräinen. Ar-based interaction for human-robot collaborative manufacturing. *Robotics and Computer-Integrated Manufacturing*, 63:101891, 2020.

[160] Terrin Babu Pulikottil, Stefania Pellegrinelli, and Nicola Pedrocchi. A software tool for human-robot shared-workspace collaboration with task precedence constraints. *Robotics and Computer-Integrated Manufacturing*, 67:102051, 2021.

[161] Leiyue Yao, Wei Yang, and Wei Huang. A data augmentation method for human action recognition using dense joint motion images. *Applied Soft Computing*, 97:106713, 2020.

[162] Ruikun Luo and Dmitry Berenson. A framework for unsupervised online human reaching motion recognition and early prediction. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2426–2433. IEEE, 2015.

[163] Hao Ding, Gunther Reißig, Kurniawan Wijaya, Dino Bortot, Klaus Bengler, and Olaf Stursberg. Human arm motion modeling and long-term prediction for safe and efficient human-robot-interaction. In *2011 IEEE International Conference on Robotics and Automation*, pages 5875–5880. IEEE, 2011.

[164] Qinghua Li, Zhao Zhang, Yue You, Yaqi Mu, and Chao Feng. Data driven models for human motion prediction in human-robot collaboration. *IEEE Access*, 8:227690–227702, 2020.

[165] Tomislav Petković, Jakub Hvězda, Tomáš Rybeckỳ, Ivan Marković, Miroslav Kulich, Libor Přeučil, and Ivan Petrović. Human motion prediction framework for safe flexible robotized warehouses. In *IEEE International Conference on Robotics and Automation (ICRA2019), Long-term Human Motion Prediction Workshop*, 2019.

[166] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[167] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007.

[168] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras. Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, 39(8):895–935, 2020.

[169] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Multi-level motion attention for human motion prediction. *International Journal of Computer Vision*, pages 1–23, 2021.

[170]Carnegie mellon motion capture database, http://mocap.cs.cmu.edu.

[171] Leonid Sigal and Michael J Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. *Brown Univertsity TR*, 120(2), 2006.

[172] Victoria Bloom, Dimitrios Makris, and Vasileios Argyriou. G3d: A gaming action dataset and real time action recognition evaluation framework. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2012.

[173] Philipp Kratzer, Marc Toussaint, and Jim Mainprice. Prediction of human full-body movements with motion optimization and recurrent neural networks. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[174] John Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. *Advances in neural information processing systems*, 2, 1989.

[175] Jie Chen, ZhongCheng Wu, and Jun Zhang. Driver identification based on hidden feature extraction by using adaptive nonnegativity-constrained autoencoder. *Applied Soft Computing*, 74:1–9, 2019.

[176] Qinxue Meng, Daniel Catchpoole, David Skillicom, and Paul J Kennedy. Relational autoencoder for feature extraction. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 364–371. IEEE, 2017.

[177] Kai Han, Yunhe Wang, Chao Zhang, Chao Li, and Chao Xu. Autoencoder inspired unsupervised feature selection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2941–2945. IEEE, 2018.

[178] Mahmood Yousefi-Azar, Vijay Varadharajan, Len Hamey, and Uday Tupakula. Autoencoder-based feature learning for cyber security applications. In *2017 International joint conference on neural networks (IJCNN)*, pages 3854–3861. IEEE, 2017.

[179] Yang Cai, Sean Hackett, and Florian Alber. Interactive indoor localization on helmet. In *International Conference on Applied Human Factors and Ergonomics*, pages 544–551. Springer, 2020.

[180] Wenchuan Wei, Keiko Kurita, Jilong Kuang, and Alex Gao. Real-time 3d arm motion tracking using the 6-axis imu sensor of a smartwatch. In *2021 IEEE 17th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, pages 1–4. IEEE, 2021.

[181] MATLAB Central File Exchange Matt J. Absolute orientation - horn's method, 2021.

[182] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *Josa a*, 4(4):629–642, 1987.

[183] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[184] A. M. Treisman and G. Gelade. A Feature-integration Theory of Attention. *Cognitive psychology*, 12(1):97–136, 1980.

[185] Tomislav Petković, Luka Petrović, Ivan Marković, and Ivan Petrović. Ensemble of lstms and feature selection for human action prediction. In *International Conference on Intelligent Autonomous Systems*, pages 429–441. Springer, 2022.

[186] Philip Sedgwick and Greenwood. Understanding the hawthorne effect. *British Medical Journal*, 351, 2015.

CURRICULUM VITAE

Tomislav Petković was born in Zadar, the Republic of Croatia in 1993. He received his BSc and MSc degrees in electrical engineering and information technology from the University of Zagreb, Faculty of Electrical Engineering and Computing (UNIZG-FER) in 2015 and 2017, respectively. As an undergraduate student, he received Josip Lončar Dean's Award for outstanding achievements in the second year of studies, while he was continuously awarded the City of Zadar Scholarship (2012-2017) for excellence. During his master's program, he was awarded Rector's award for the project titled *Decentralized control of the multi-agent robotic system*. Upon finishing his studies, he was employed as a research assistant at the Department of Control and Computer Engineering (ZARI) at FER, Zagreb, in June 2017. He has worked on several international and domestic scientific projects. In a collaboration with the industry partner KONČAR – Institut za elektrotehniku d.d., he worked on the SafeLog project: Safe human-robot interaction in logistic applications for highly flexible warehouses (2017-2021). His primary role was the development of a human intention recognition module as part of the human aware planner. Furthermore, he participated in L4MS project: Logistics for Manufacturing SMEs, where his role was development of robot localization algorithms (2018-2019). During his PhD studies, he spent several months in collaboration at a foreign university. His stay at Karlsruhe Institute of Technology, Germany (2017 and 2018) was sponsored by DAAD. His main research interests within the area of autonomous systems and mobile robotics revolve around human action prediction and human path prediction. He is an author or co-author of two papers published in peer-reviewed journals and four papers presented at international conferences. The full list of publications is given below.

LIST OF PUBLICATIONS

Journal Publications:

1. T. Petković, D. Puljiz, I. Marković and B. Hein. Human intention estimation based on hidden Markov model motion validation for safe flexible robotized warehouses. *Robotics and Computer-Integrated Manufacturing*, 57, 182-196, 2019.

2. T. Petković, L. Petrović, I. Marković, I. Petrović. Human action prediction in collaborative environments based on shared-weight LSTMs with feature dimensionality reduction. *Applied Soft Computing*, 126, 2022.

Conference Publications:

1. T. Petković, I. Marković, and I. Petrović. Human Intention Recognition in Flexible Robotized Warehouses based on Markov Decision Processes. In *Third Iberian Robotics Conference (ROBOT), Seville, Spain, 2017.*

2. M. Seder, L. Petrović, J. Peršić, G. Popović, T. Petković, A. Šelek, B. Bićanić, I. Cvišić, D. Josić, I. Marković, and I. Petrović. Open Platform Based Mobile Robot Control for Automation in Manufacturing Logistics. *IFAC Workshop on Robot Control, Daejeon, South Korea, 2019.*

3. T. Petković, J. Hvězda, T. Rybecký, I. Marković, M. Kulich, L. Přeučil, and I. Petrović. Human Intention Recognition for Human Aware Planning in Integrated Warehouse Systems, *28th Mediterranean Conference on Control and Automation (MED), 2020.*

4. T. Petković, L. Petrović, I. Marković and I. Petrović. Ensemble of LSTMs and feature selection for human action prediction. *16th International Conference on Intelligent Autonomous System, Singapore, 2021.*

ŽIVOTOPIS

Tomislav Petković rođen je u Zadru, Republika Hrvatska, 1993. godine. Zvanje prvostupnika, odnosno magistra elektrotehnike i informacijske tehnologije Sveučilišta u Zagrebu, Fakulteta elektrotehnike i računarstva (UNIZG-FER) stekao je 2015. te 2017. godine. Tijekom preddiplomskog studija, nagrađen je Dekanovom nagradom Josip Lončar za izvanredan uspjeh u drugoj godini studija. Tijekom diplomskog studija, nagrađen je Rektorovom nagradom za projekt *Decentralizirano upravljanje multiagentskim robotskim sustavom*. Kroz svoje obrazovanje, više puta je nagrađen Stipendijom Grada Zadra (2012-2017) za izvrsnost. Po završetku svojih studija, zaposlen je kao znanstveni suradnik na Zavodu za automatiku i računalno inženjerstvo (ZARI) pri UNIZG-FER-u, Zagreb, od lipnja 2017. Radio je na nekoliko domaćih i međunarodnih znanstvenih projekata. U suradnji s partnerom iz industrije KONČAR – Institut za elektrotehniku d.d., radio je na projektu SafeLog: *Sigurna interakcija robota i ljudi u logističkoj primjeni za visoko fleksibilna skladišta* (2017-2021). Glavne uloge na tom projektu su uključivale razvoj metode za predviđanje namjere čovjeka za o čovjeku svjesno planiranje. Uz to, sudjelovao je i u projektu L4MS: Logistics for Manufacturing SMEs, gdje se bavio lokalizacijom robota (2018-2019). Tijekom doktorskog studija, proveo je nekoliko mjeseci surađujući na stranom sveučilištu. Njegov boravak na Karlsruhe institutu za tehnologiju, Njemačka (2017 i 2018) financirala je zaklada DAAD. Njegovi glavni istraživački interesi u području autonomnih sustava i mobilne robotike nalaze se u području prepoznavanje namjere čovjeka i predviđanje putanje čovjeka. Autor je ili suautor dva znanstvena rada u časopisima i četiri rada prezentiranih na međunarodnim konferencijama.