Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Andraž Krašček

Vizualizacija in analiza izhodnih podatkov optičnega sistema za sledenje gibanja

magistrsko delo

študijski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Matija Marolt
Somentor: izr. prof. dr. Jaka Sodnik

Ljubljana, 2015

Izjava o avtorstvu magistrskega dela

Spodaj podpisani Andraž Krašček sem avtor magistrskega dela z naslovom:

Vizualizacija in analiza izhodnih podatkov optičnega sistema za sledenje gibanja

S svojim podpisom zagotavljam, da:

- sem magistrsko delo izdelal samostojno pod mentorstvom doc. dr. Matije Marolta in somentorstvom izr. prof. dr. Jake Sodnika,
- so elektronska oblika magistrskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko magistrskega dela,
- soglašam z javno objavo elektronske oblike magistrskega dela v zbirki "Dela FRI".

V Ljubljani, 21. avgust 2015 Podpis avtorja:

Zahvaljujem se mentorju, doc. dr. Matiji Maroltu in somentorju, izr. prof. dr. Jaki Sodniku za vse nasvete in strokovno pomoč, ki sta mi jo nudila v času izdelave tega magistrskega dela. Zahvaljujem se tudi staršem, da sta mi omogočila študij in me pri njem podpirala.

Monika, hvala ti za pomoč in spodbudo takrat, ko sem jo najbolj potreboval.

Hvala tudi bratu, prijateljem, sodelavcem in članom Laboratorija za informacijske tehnologije za njihov čas, ki so ga porabili za reševanje uporabniškega testa.

Kazalo

1 Uvod **1**

2 Zajem gibanja **5**

2.1 Uporaba 5

2.2 Zgodovina sistemom za zajem gibanja 6

2.3 Akustični sistem 7

2.4 Mehanski sistem 8

2.5 Magnetni sistem 10

2.6 Vztrajnostni sistemi 10

2.7 Optični sistem brez označevalnikov 12

2.8 Optični sistem z označevalniki 13

2.9 Qualisys Track Manager (QTM) 21

2.10 Razlogi za razvoj nove rešitve 22

3 Načrtovanje lastne aplikacije **23**

3.1 Zahteve 23

3.2 Uporabljene tehnologije 24

3.3 Načrtovanje uporabniškega vmesnika 28

3.4 Arhitektura aplikacije 31

4 Izvedba **35**

4.1 Analiza podatkov 35

4.2 Prepoznavanje obstoječih modelov 37

5 Uporabniška študija **47**

5.1 Udeleženci 47

5.2 Opis in potek študije 48

5.3 Zastavljene naloge 49

5.4 UEQ 49

5.5 Rezultati 52

6 Zaključek **57**

Slike

2.1 Primer mehanskega sistema (Gypsy MetaMotion). 9
2.2 Kamera Qualisys Oqus. 17
2.3 Plošča z mrežo označevalnikov, uporabljena za linearizacijo. . 19
2.4 Postopek umerjanja sistema. 20
2.5 Grafični prikaz sledenja dveh označevalnikov. 21

3.1 Levo: zakasnitev glede na število vzorcev. Desno: porazdelitev zakasnitev. 26
3.2 Strežniška arhitektura aplikacije. 31

4.1 Postopek izvedenega simuliranega ohlajanja v dveh korakih. . 39
4.2 Kontrolni model evalvacije s šestnajstimi označevalniki in sedemnajstimi povezavami. 42
4.3 Natančnost in priklic za različno število označevalnikov v posnetkih. Vidimo lahko, da natančnost predlagane metode pada ob večjem številu označevalnikov v posnetku. 45

5.1 Uporabljen UEQ vprašalnik. 55
5.2 Povprečni časi v sekundah s stopnjo natančnosti. 56
5.3 Rezultat UEQ vprašalnika po lestvicah. 56

Seznam uporabljenih kratic

kratica	angleško	slovensko
AIM	Automatic Identification of Markers	Samodejno zaznavanje označevalnikov
FPS	Frames Per Second	slik na sekundo
HCI	Human Computer Interaction	interakcija med človekom in računalnikom
QTM	Qualisys Tracker Manager	Qualisys Tracker Manager
RTP	Real Time Protocol	protokol za prenos v realnem času
TOF	Time of Flight	čas preleta
UEQ	User Experience Questionnaire	vprašalnik o uporabniški izkušnji
WS	WebSocket	WebSocket

Povzetek

Magistrsko delo je osredotočeno na zajem in analizo izhodnih podatkov optičnega sistema za sledenje gibanja Qualisys. Predlagamo prototipno aplikacijo, razvito s pomočjo sodobnih spletnih tehnologij, ki bo omogočala povezovanje na oddaljeni optični sistem in zajem podatkov o koordinatah izbranih prostorskih točk v realnem času. Aplikacija omogoča 3D izris zajetih točk na poljubnem številu sočasnih in neodvisnih odjemalcev, preko katerih je mogoče avtonomno in interaktivno manipulirati s prikazano 3D sceno. V magistrskem delu predlagamo konkretne algoritme in pripadajoče interakcijske postopke za povezovanje individualnih točk v kompleksne modele opazovanih oziroma sledilnih objektov, na osnovi katerih je možno analizirati gibanje teh objektov. S pomočjo uporabniške študije smo preverili učinkovitost izvedenih interakcij med človekom in računalnikom (HCI) ter izpostavljamo glavne prednosti in slabosti predlaganih metod v primerjavi z obstoječimi rešitvami, ki so del sistema Qualisys.

Ključne besede: zajem gibanja, analiza gibanja, interakcija človek-računalnik, uporabniška študija

Abstract

Title: Visualization and analysis of captured data from optical motion capture system

This master thesis is focused on acquisition and analysis of motion capture data. We propose a prototype application developed with latest web technologies that enables real time data transfer of selected markers from remote optical system to client's local coordinate system. The application is designed to enable multiple concurrent connections to individual clients. Each client can interactively manipulate with 3D scene. Along side we also propose algorithms and interaction methods for connecting individual markers in complex models of markers. Based on this models users can analyze movements data in 3D space and time. We verify the effectiveness of HCI method and proposed application's user interface with user study.

Keywords: motion capture, motion analyses, human-computer Interaction, user study

Poglavje 1

Uvod

Z napredkom tehnologije na področjih, ki se ukvarjajo s preučevanjem in posnemanjem človeškega ali kakšnega drugega gibanja, se je pojavila potreba po natančnem in hitrem zajemu gibanja v digitalno uporabni obliki. Takšna področja so na primer biomehanika v športu, opazovanje napredka rehabilitacije po nesrečah v medicini, zabavna industrija, itd. To omogočajo sistemi zajema gibanja. Sistem pri zajemu gibanja rezultate prikaže kot gručo zajetih točk v prostoru. Za analizo gibanja je potrebno opazovati večje število točk oz. označevalnikov, ki so nameščeni na različne dele opazovanega telesa ali objekta. Točke, ki označujejo in določajo posamezni rigidni del opazovanega objekta, je potrebno povezati in združiti v t.i. segmente (v primeru človeškega telesa so to roke, noge, prsni koš, glava, itd.), ti pa se nadalje združijo v kompleksen model [1] – Automatic Identification of Markers (AIM). Model AIM je generičen model za predstavitev objektov v prostoru in ne temelji na kinematičnem modelu, ampak samo na povezavah med označevalniki ter njihovimi lastnostmi.

Za potrebe magistrskega dela bomo uporabili profesionalni sistem za sledenje gibanja švedskega podjetja Qualisys, ki je na voljo v Laboratorij za informacijske tehnologije na Fakulteti za elektrotehniko. Sistem je sestavljen iz osmih kamer Oqus3+, velikega števila pasivnih označevalnikov in namenske programske opreme za izračun 3D položajev točk, imenovane QTM. Apli-

kacija omogoča izgradnjo prej omenjenih modelov AIM in preprosto analizo gibanja njihovih posameznih delov. Analiza je zelo omejena in je ni mogoče shraniti ali izvoziti in tako uporabiti pri nadaljnjem delu. Aplikacija QTM teče na enem samem računalniku in ne omogoča sočasnega dela večjega števila uporabnikov. Sistem je zato nemogoče uporabiti kot študijsko orodje za večje število sočasnih uporabnikov, ki bi lahko neodvisno opazovali in analizirali podatke. V okviru magistrskega dela želimo razviti prototipni sistem, ki bo omogočal zajem podatkov o opazovanih objektih v realnem času in njihov interaktivni prikaz ter analizo na večjem številu sočasnih računalnikov. Omogočal bo izgradnjo večjega števila neodvisnih kompleksnih modelov, njihovo shranjevanje in izmenjevanje. Vseboval bo tudi intuitivni uporabniški vmesnik, preko katerega bo uporabnik zgradil nov model s povezovanjem izbranih označevalnikov. Predlagan in implementiran bo tudi algoritem za samodejno zaznavo modela iz novo zajetih podatkov. Obstoječe analitične rešitve in postopki za opis in analizo gibanja 3D objektov v prostoru bodo posplošeni in prilagojeni izhodnim podatkom sistema Qualisys in pripadajočim modelom. V našem primeru se morajo vsi omenjeni postopki izvajati v realnem času, saj želimo sproti prikazovati spreminjajoče se trajektorije gibanja opazovanega modela v prostoru.

Vizualizacija 3D podatkov v spletnih aplikacijah je bila raziskana v različnih študijah. V veliko primerih se študije osredotočajo na izrisovanje poligonskih mrež z uporabo vtičnikov v brskalnikih. [2], [3], [4]. Najpogostejši problemi le-teh so specializirani programski jeziki, različne zahteve posameznega vtičnika, omejena prenosljivost med sistemi in napredna podpora izrisovanja [5]. Danes je proces izrisovanja 3D grafik v brskalnikih poenostavljen in poenoten z uporabo WebGL JavaScript API-ja, ki temelji na OpenGL ES 2.0. Vsi vodilni brskalniki ga že podpirajo ali pa je njegova podpora načrtovana. Raziskave opisujejo uporabo WebGL-a za prikaz in interakcijo s 3D grafiko v brskalniku. Ostali raziskujejo prednosti uporabe spletnih aplikacij za izogibanje podpori, potrebni različnim sistemom. Congote s sodelavci raziskuje

zmogljivost in razširljivost izrisovanja prostorninskih podatkov z WebGL [6]. Leung in Salga razpravljata o tem, kako lahko spletne aplikacije ne samo nadomestijo namizne, ampak tudi ponudijo edinstvene 3D vsebine [7]. Schwartz in sodelavci predlagajo okolje, ki bi temeljilo na WebGL API-ju, za predstavitev 3D objektov, ki bi kar najbolj posnemalo materialni občutek le-teh.

V magistrskem delu bi izpostavili dva izvirna prispevka. Prvi prispevek je zasnova in izvedba celovitega in inovativnega sistema za vizualizacijo podatkov optičnega sistema za sledenje gibanja na spletu s predlogom lastne metodologije za grajenje in analizo modelov kompleksnih merjencev. Pri zasnovi smo v sistem uvedli sledeče novosti in izboljšave na področju sorodnih orodij: uporabniški vmesnik, ki se prilagaja trenutni interakciji uporabnika, razširljivo platformo za dodajanje analiz na modelih (v okviru magistrskega dela predlagamo tri osnovne analize) in metodologijo za hitro grajenje modelov, ki namesto lokacije označevalnikov v prostoru upošteva medsebojno lego označevalnikov in njihovo strukturo.

Drugi prispevek je evalvacija uporabniškega vmesnika s pomočjo uporabniške študije. Poleg izvedbe študije podamo tudi zasnovo uporabniškega testa UEQ in možnosti uporabe ter apliciranja njegovih rezultatov.

Izvirni prispevki dela so bili predhodno objavljeni v naslednjih primerih:

- A. Krašček, K. Stojmenova, S. Tomažič, J. Sodnik. Web based e-learning tool for visualization and analysis of 3D motion capture data. DigitalWorld 2015, str.: 131-137, Lizbona (Portugalska), 2015.

- A. Krašček, J. Sodnik. Qualisys web tracker - a web-based visualization tool for real-time data of an optical tracking system. 4th International Conference on Information Society and Technology, str.: 155-159, Kopaonik (Srbija), 2014.

- A. Krašček, G. Jakus, J. Sodnik. JavaScript aplikacijski programski vmesniki in njihova uporaba v sodobnih odjemalskih aplikacijah. Zbor-

nik dvajsete mednarodne Elektrotehniške in računalniške konference ERK 2011, str.: 47-50, Portorož (Slovenija), 2011.

Zgradba magistrske naloge je sledeča. V drugem poglavju predstavimo zgodovino razvoja ter različne sisteme in tehnike zajema gibanja. Vsak sistem na kratko opišemo in poudarimo njegove prednosti in slabosti. V podpoglavju o optičnih sistemih dodatno opišemo sistem Qualisys in aplikacijo QTM. Sledi poglavje o obstoječih rešitvah za prikaz in analizo podatkov. V tem poglavju tudi analiziramo razloge za načrtovanje nove rešitve, na kratko opišemo uporabljene tehnologije ter predlagamo interakcijske metode uporabnika z aplikacijo in arhitekturo aplikacije. Četrto poglavje je namenjeno opisu izvedbe dveh modulov spletne aplikacije. Opis, potek in rezultati uporabniške študije so predstavljeni v petem poglavju. Sledi še zaključek in uporabljena literatura.

Poglavje 2

Zajem gibanja

Zajem gibanja je proces snemanja položaja in orientacije objekta v računalniško uporabni obliki. Navadno snemamo gibanje človeka, živali ali različnih predmetov.

2.1 Uporaba

Zajem gibanja kot orodje za analizo je bil najprej uporabljen na področju biomehanike. Z razvojem je postal pomemben vir podatkov o gibanju tudi na področju računalniške animacije, izobraževanja, športnega treniranja in nazadnje za računalniške igre in film. Pri filmu je zajem gibanja uporabljen za snemanje igralcev, na gibanje katerih s pripadajočo programsko opremo animirajo različne domišljijske like. Med bolj znanimi primeri so filmi Avatar, Gospodar prstanov, itd.

Na področju računalniški iger se zajem gibanja uporablja za animacijo in simuliranje gibanja likov v igri (npr.: simulacija igralcev nogometa ali borilnih veščin), kot tudi za zajem gibanja igralca, ki igro igra. S sistemi brez označevalnikov lahko z gibanje telesa nadziramo igro, dodatno lahko igralcu tudi simuliramo 3D resničnost. Z različnimi naglavnimi pripomočki lahko igralec dobi občutek virtualne resničnosti.

Pomembno področje uporabe je medicina. Z analizo hoje ali pri razvoju

protez ljudem omogoča boljše okrevanje po poškodbah ali drugih boleznih. [21] V športni medicini športnikom pomaga pri izboljšanju učinkovitosti.

2.2 Zgodovina sistemom za zajem gibanja

Animacija se je začela v začetku 20. stoletja, ko je Winsor McCay skiciral lik na večje število strani z majhnimi razlikami med vsako skico. Skice je vzorčil s konstantno frekvenco in s tem ustvaril iluzijo premikanja lika. [8] Do razvoja računalnikov se proces animiranja ni kaj veliko spremenil. Animatorji so morali animirati vsak časovni okvir posamično, kar je časovno zelo zamudno delo. Z animacija s ključnimi točkami se je ta proces poenostavil, saj animatorjem ni bilo več potrebno animirati vmesnih časovnih okvirjev. [9] Kljub animaciji s ključnimi točkami pa je bilo nekatere gibe še vedno skoraj nemogoče animirati. Primer takega gibanja je animacija človeške hoje, ki je zelo kompleksna zaradi skupnega usklajenega delovanja večjega števila sklepov telesa. Pojavili so se prvi poskusi zajema gibanja s pomočjo filmskih kamer. Leta 1915 je Max Fleischer izumil postopek "rotoskopiranje". Rotoskopiranje je postopek, pri katerem gibanje igralca najprej zajamemo s filmsko kamero. Posamezne slike filma so prenesene na steklene plošče, s katere animator preriše obris igralca in s tem ustvari posamezen časovni okvir animacije. Z željo po nadaljnjih pohitritvah animiranja so se začeli razvijati sistemi, ki jih še danes uporabljamo za zajem gibanja. Ti sistemi zajamejo gibanje v resničnem svetu ter podatke o gibanju prenesejo v 3D model v virtualnem svetu. Najprej so se razvili mehanski sistemi, ki pa so bili zelo okorni in so igralca zelo ovirali pri prostem premikanju, predvsem zaradi velikega števila žic, ki so povezovale senzorje. [10] Razpon sistemov za zajem gibanja se je še povečal in danes so na voljo poleg mehanskih tudi optični, akustični in magnetni sistemi.

2.3 Akustični sistem

Akustični sistem navadno uporablja ultrazvok za zaznavo razpona. Ultrazvok ima frekvenco nad $20kHz$ in je neslišen za človeško uho.

Par oddajnik - sprejemnik zadošča za meritev oddaljenosti merjene točke od fiksne točke. Brez dodatnih informacij ta podatek opiše sfero, kjer se lahko nahaja merjena točka glede na fiksno. Z dodatnim sprejemnikom lahko to področje zmanjšamo na krog, presek med dvema sferama. Tretji sprejemnik dodatno zmanjša področje nahajanja merjene točke na dve točki v prostoru. Eno od njiju lahko navadno zavržemo. Za določitev položaja in orientacije oddajnika v prostoru torej potrebujemo vsaj tri sprejemnike. Na opazovani objekt je lahko razporejeno večje število oddajnikov, ki se zaporedno sprožajo in ustvarijo značilen niz frekvenc.

Akustični sistemi navadno uporabljajo eno od dveh tehnik za določanje položaja in orientacije točk. Te so:

- **Čas preleta (TOF)**
  Pri metodi čas preleta izmerimo čas t, ki ga potrebuje ultrazvočni impulz, da prepotuje od oddajnika k sprejemniku. Sistem ve, kdaj je oddajnik oddal signal in kdaj je sprejemnik signal prejel, iz razlike pa lahko izračuna čas t. Tega pomnožimo s hitrostjo zvoka in dobimo oddaljenost točke, d. $d = c * t$. Iz dobljene oddaljenosti lahko ocenimo absolutni položaj točke glede na sprejemnike. [20] Pomanjkljivost te metode je zakasnitev in nizka frekvenca posodabljanja.

- **Fazna koherenca**
  Metoda meri fazno razliko med zvočnim valom pri oddajniku in sprejemniku ter izračuna spremembo oddaljenosti. Vse signale lahko predstavimo kot vsoto sinusnih funkcij v obliki $A\cos(\omega t - \phi)$, pri tem je ϕ fazni premik in A amplituda signala. Iz faznega zamika med oddanim in sprejetim signalom z znano valovno dolžino ali frekvenco lahko izračunamo oddaljenost.[17]

V obeh metodah se hitrost zvoka uporablja za pretvorbo časa v oddaljenost. Pomanjkljivost obeh metod je zakasnitev, ki nastane zaradi počasnega potovanja signala od oddajnika do sprejemnika. Hitrost zvoka je $331 m/s$ pri $0° C$, s spreminjanjem temperature in zračnega pritiska se hitrost zvoka spreminja. Zaradi tega je ne moremo določiti kot konstanto. V idealnih plinih lahko hitrost zvoka c posplošimo

$$c = \sqrt{\frac{\gamma RT}{M}} \tag{2.1}$$

Pri tem je γ adiabatni eksponent, R splošna plinska konstanta, M molska masa plina in T temperatura. Ker so ostali parametri konstanta za dano vrsto plina, vidimo, da je hitrost zvoka v plinu odvisna le od njegove temperature.

Akustični sistemi imajo omejen delovni prostor, oddajniki morajo biti v vidnem polju sprejemnikov. Na te sisteme vpliva temperatura okolja in spremembe pritiska ter oviranje ultrazvočnega signala ob prisotnosti ljudi.

2.4 Mehanski sistem

Mehanski sistem sestavljajo potenciometri in drsniki, ki nameščeni na izbrane sklepe prikazujejo njihovo stanje. Ti sistemi imajo nekaj prednosti pred ostalimi. Ker je njihovo delovanje podobno stop-motion tehniki v animiranju, je ta sistem zelo priljubljen pri filmski industriji. Njihova prednost je tudi neobčutljivost na magnetne učinke in motnje iz okolja, poleg tega pa skoraj ne potrebujejo umerjanja, zaradi česar je delo z njimi zelo hitro in produktivno. Poglavitna slabost je v njihovi okornosti pri uporabi. [18]

Mehanski sistemi merijo kote sklepov in dolžine med sklepi. S poznavanjem enega položaja lahko ostale položaje izračunamo iz relativnih položajev sklepov glede na prvi položaj. Uporabljajo se za meritve vseh delov telesa. Poznamo različne izvedbe, od oblek za celotno telo, ki merijo položaje vseh glavnih sklepov, do mehanskih rok in rokavic, ki merijo položaj roke ali prstov.

Poznamo fiksne mehanske sisteme in sisteme, ki so pritrjeni samo na

uporabnika. Prvi so pritrjeni na fiksni lokaciji v prostoru in na ta način omejujejo uporabnika pri izvedbi zajema gibanja, pri slednjih pa uporabnik ni prostorsko omejen, lahko pa takšne obleke zaradi svoje teže in velikosti ovirajo gibanje uporabnika.

Rotacije in dolžine med sklepi merimo s potenciometri, upogibnimi senzorji in prestavami. Potenciometer je naprava, ki pretvori rotacijo ali premik v napetost. Upogibni senzor je tanek trak iz plastične mase, katere upornost se spremeni z upogibanjem. Z večjim upogibom se upornost povečuje. Kot alternativo na kratkih razdaljah lahko uporabimo optično vlakno. Pri optičnem vlaknu merimo zmanjševanje jakosti svetlobe pri večjem upogibu vlakna.

Slika 2.1: Primer mehanskega sistema (Gypsy MetaMotion).

2.5 Magnetni sistem

Magnetni sistemi uporabljajo več sprejemnikov, razporejenih po različnih točkah na opazovanem objektu. Z uporabo magnetnega polja določijo oddaljenost in orientacijo oddajnika glede na sprejemnik. Magnetna polja so lahko nizko frekvenčna izmenične napetosti ali enosmerna, ki oddajajo magnetne pulze.

Magnetni sistem za zaznavanje gibanja je sestavljen iz oddajnega in sprejemnega navitja. Za zaznavo ene dimenzije (npr.: x os) potrebujemo en sprejemnik, orientiran v tej smeri. Ko skozi oddajnik steče električni tok, ustvari magnetno polje. Magnetno polje v navitju sprejemnika, orientiranega v isti smeri kot oddajnik, sproži najvišjo napetost. Zaznana napetost omogoča izračun oddaljenosti oddajnika od sprejemnika in poravnanosti osi med njima. Tri ločena navitja, postavljena ortogonalno eno na drugo, potrebujemo za sprožitev in izmero magnetnega polja v oseh x, y, z. Magnetno polje na treh navitjih oddajnika je sproženo zaporedno, na vsakem navitju posebej. Vsak signal izmerijo vsa tri sprejemnikova navitja. Celoten cikel tako zajema tri meritve za tri oddana magnetna polja. Iz devetih vrednosti je mogoče izračunati položaj in orientacijo oddajnika glede na sprejemnik. Sprejet signal se zmanjšuje s kvadratom glede na oddaljenost in kosinusom osi navitja in smeri magnetnega polja. Moč prejetega signala se primerja s predhodno prejetimi signali za izračun oddaljenosti in moči sproženih signalov v oseh x, y, z se primerjajo med seboj za izračun orientacije.

Magnetni sistemi spadajo med cenejše sisteme za zajem gibanja, zaradi nizke vzorčne frekvence pa so uporabni le za enostavne zajeme gibanja. Poleg nizke frekvence je opazovani objekt omejen tudi z velikim številom žic, ki jih takšen sistem zahteva.[19]

2.6 Vztrajnostni sistemi

Vztrajnostni sistemi uporabljajo pospeškometre in žiroskope za merjenje pospeška položaja in spremembo orientacije objekta. Senzorji so pasivni in za

svoje delovanje ne potrebujejo oddajnikov kot akustični ali optični sistemi. Izvajanja meritve tako niso omejena s prostorom ali drugimi vplivi okolja (npr.: hrup, viri infrardeče svetlobe). Idealno je, da vsak označevalnik vsebuje ortogonalne trojice, za 3D položaj (x, y, z) in 3D orientacijo (ang.: $roll$, $pitch$, yaw).

2.6.1 Pospeškometer

Pospeškometer meri silo, ki deluje na določeno maso, saj ne more direktno meriti dejanskega pospeška. Masa je pritrjena na dušeno vzmet, ta pa na ohišje pospeškometra. Ko na pospeškometer ne deluje nobena sila, vzmet miruje in ne pride do premika. Ko na pospeškometer deluje sila, zaradi vztrajnosti masa zaostane in tako pride do premika. Premik je proporcionalen pospešku, ki je deloval na ohišje pospeškometra.

Prvi pretvornik pretvori pospešek v premik, r, po enačbi:

$$r = \iint a dt^2 \tag{2.2}$$

Drugi pretvornik premik r spremeni v uporaben signal. Tipično so pretvorniki potenciometri ali piezo električne komponente.

2.6.2 Žiroskop

Žiroskop je naprava, ki ponazarja in izrablja načelo ohranitve vrtilne količine v fiziki. Če deluje sila na vrtečo se maso, bo njena os vrtenja precesirala pravokotno na svojo os in os, v kateri deluje sila. Pri precesiji os vrtenja ne miruje, ampak se giblje tako, da opisuje plašč stožca. Masa, ki se vrti zelo hitro, ima velik kotni moment, ki se močno upira spremembi smeri. Če je vrteča se masa vpeta v kardansko obešenje, lahko ta pojav uporabimo za meritve sprememb v orientaciji. Kot, za katerega se nagne žiroskop znotraj ohišja, je mera kotne hitrosti. Za meritev sprememb 3D orientacije potrebujemo tri žiroskope, ki dajo informacijo o treh kotnih hitrostih.

2.7 Optični sistem brez označevalnikov

S hitrim razvojem na področju računalniškega vida se razvijajo tudi sistemi za zajem gibanja, ki za svoje delovanje ne potrebujejo označevalnikov. Iz posnetih video vsebin ali videa, zajetega v realnem času, algoritmi poskušajo določiti obrise človeškega telesa ali drugih predmetov. Postopek zajema gibanja je v celoti izveden s programsko opremo. S tem se izničijo vse fizične omejitve, ki so jih predstavljale žice, posebne obleke, označevalniki, pojavile pa so se računske omejitve strojne opreme in z njo povezana cena takšnih sistemov. Primer takšnega sistema je Microsoft Kinect.[10]

2.8 Optični sistem z označevalniki

V preteklosti so optični sistemi z označevalniki (v nadaljevanju: optični sistem) zaradi njihovih prednosti (hitri, natančni, brezžični, poceni) predstavljali alternativo pogosto uporabljenim magnetnim sistemom. Trenutno se uporablja veliko število različnih algoritmov in strojne opreme v optičnih sistemih za zajem gibanja, ki so na voljo na trgu ali se razvijajo v raziskovalnih laboratorijih. Optične sisteme lahko razdelimo v kategorije glede na različne parametre.

Prvi parameter je postavitev sistema. Optični senzorji so lahko postavljeni v okolje, kjer poteka zajem, in usmerjeni proti objektu, ki ga opazujemo. Takšen način imenujemo outside-in. Pri obratni postavitvi, ki se imenuje inside-out, pa so optični senzorji pritrjeni na opazovani objekt in zajemajo spremembe okolja, ko se objekt premika.

Outside-in način se uporablja v zajemu gibanja človeka, posebej kadar želimo zajeti skupno delovanje več delov telesa. Na ta način zajemamo tudi geste obraza, rok, dlani, nog. Tako se lahko opazovani objekt neovirano giblje, njegovo gibanje pa ni ovirano z žicami in okornimi senzorji. Za uporabo, pri kateri opazujemo samo toge objekte, je primernejši način zajema inside-out. Takšni primeri so uporaba v razširjeni resničnosti, kjer uporabnik nosi vso opremo na sebi tudi za potrebo prikaza vizualizacije. Meriti moramo samo položaj glave v prostoru, za katero predpostavimo, da je toga. Dodatna prednost inside-out načina je njegova neomejenost s prostorom. Način postavitve senzorjev ne vpliva na natančnost zajema, saj je le-ta odvisna od kvalitete leče, ločljivosti senzorja in velikosti slike projekcije togega objekta.

Naslednji parameter, ki vpliva na zajem gibanja z optičnim sistemom, je osvetlitev. Navadno prostor, v katerem poteka zajem, vpliva na omejitve glede osvetlitve. V prostoru se lahko nahajajo ekrani, projektorji, dodatna spremenljivka je sončna svetloba. Z uporabo algoritmov za obdelavo slik lahko izločimo probleme, ki jih povzročajo različni načini osvetlitve [11].

Pogosto se uporablja naravna dnevna osvetlitev, vendar pri tem ni mogoče

zaznati vseh značilnosti, ki se razkrijejo ob uporabi dinamične osvetlitve. Druga možnost je uporaba osvetlitve, ki ima nižjo ali višjo frekvenco, kot jo ljudje lahko zaznamo. Prednost tega načina je neodvisnost takšnega sistema od osvetlitve okolja. Sistem lahko deluje tudi v pogojih z majhno osvetlitvijo, slabše pa deluje v pogojih z veliko sončne svetlobe. Dodatna možnost za osvetlitev je uporaba strukturirane svetlobe, ki se projicira na površino opazovanega objekta. Na takšen način lahko lažje zaznamo in zajamemo značilnosti površine. Za takšno postavitev potrebujemo projektor in kamero, ki sta fiksno postavljena in umerjena med seboj. 3D točke lahko določimo na površini glede na umerjene podatke in postopek triangulacije. Postopek je širše uporabljen za namene 3D rekonstrukcije in ne še toliko v zajemu gibanja.

Optični sistemi se razlikujejo v številu uporabljenih kamer. Uporaba večjega števila kamer še ne pomeni tudi uporabe stereoskopskega vida [12]. S premikom kamere dobimo premik scene in s tem iluzijo stereoskopskega vida. Uporaba dveh kamer ima prednost v postavitvah, kjer se objekti premikajo. Z dvema kamerama imamo v vsakem trenutku na voljo oceno 3D položaja, če je postavitev kamer statična in so znani podatki umerjanja kamer. Monoskopski vid potrebuje za oceno položaja vsaj štiri statične točke in znane točke ravnine.

Prednosti optičnih sistemov so visoka frekvenca vzorčenja v visoki ločljivosti, kar omogoča tudi zajem hitrih gibov (npr.: borilne veščine) ali zelo majhnih natančnih premikov (npr.: mimika obraza), poleg tega pa omogočajo opazovanemu objektu relativno prosto gibanje v prostoru brez žic ali senzorjev, ki bi jih omejevali. Pri boljših sistemih je lahko tudi opazovani prostor precej velik (npr.: filmski studio), kamere pa niso občutljive na zunanje motnje (npr.: sonce, reflektorji) in se jih uporablja tudi na prostem. Proizvajalci navadno skupaj s sistemi ponujajo tudi sofisticirano programsko opremo. Zaradi vseh naštetih prednosti so ti sistemi najdražji na trgu.

2.8.1 Označevalniki

Optični sistemi za zajem gibanja uporabljajo dva tipa označevalnikov: aktivne in pasivne označevalnike.

Aktivni označevalniki so napajane infrardeče LED diode. Infrardeče LED diode se uporabljajo za zmanjševanje vpliva svetlobe iz okolja na izvedbo meritve. Pri aktivnih označevalnikih se uporablja tudi posebna sekvenca proženja. Ne prižgejo se vsi hkrati, temveč eden za drugim, vrstni red pa je poznan sprejemniku in lahko točno identificira posamezen označevalnik.

Pasivni označevalniki niso napajani, ampak so narejeni iz odsevnih materialov in samo odbijajo svetlobo, ki jo oddajajo sprejemniki ali drugi svetlobni elementi v prostoru.

Pri obeh tipih uporabljenih označevalnikov sprejemnik zajema opazovani objekt in iz zajetih kotov lahko preračunamo položaj in orientacijo objekta.

2.8.2 Senzorji

Detektorji so lahko preproste video ali charge-coupled device (CCD) kamere (navadno za pasivne označevalnike) ali foto diode z bočnim efektom (navadno za aktivne označevalnike), ki podajo lokacijo središča svetlobe v ravnini. Video in CCD kamere zahtevajo dodatno tehnologijo za izmero središča, medtem ko foto diode podajo električni tok, proporcionalen središčni legi svetlobe.

Foto diode z bočnim efektom (LEPD)

LEPD meri padec svetlobnega žarka v eni dimenziji. Na foto občutljivi površini sta na vsakem koncu pritrjena senzorja. Svetlobni žarek na površini sproži elektrone, ki tečejo na obe strani k senzorjema. Tok na vsakem senzorju je odvisen od oddaljenosti središča vpada svetlobnega žarka na foto občutljivo površino. Če je središče vpada svetlobnega žarka na sredini površine, je izmerjen tok na obeh senzorjih enak. Dvodimenzionalen LEPD dobimo tako, da združimo dve LEPD diodi, eno pod kotom 90 stopinj

glede na drugo. LEPD diode lahko uporabljamo za izmero intenzivnosti in položaja svetlobnega žarka.

LEPD zagotavljajo hitrejši čas odgovora in večjo ločljivost položaja vpada svetlobe kot CCD senzor. Z diodo je tudi lažje izračunati središče kot pri CCD senzorju, za razliko od CCD senzorja pa diode omogočajo uporabno meritev tudi, ko je vpadna svetloba zamegljena oziroma ni izostrena. [13]

Quad-cells

Quad-cells so sestavljene iz štirih foto občutljivih celic. Ko svetlobni žarek vpade na quad-cell, ta ustvari električni tok v vsaki od štirih celic sorazmerno osvetljenosti posamezne celice. Če svetlobni žarek pade točno v center celic, bo električni tok vseh celic enak. Če je svetlobni žarek premajhen in pade med celice ali če je prevelik in prekrije celotno površino celic, ne bomo dobili uporabne informacije o poziciji žarka. [14]

Iz napetosti posameznih celic lahko izračunamo x in y koordinate izvora svetlobnega žarka.

$$\begin{aligned} x &= \frac{(i_1 + i_2) - (i_3 + i_4)}{i_1 + i_2 + i_3 + i_4} \\ y &= \frac{(i_1 + i_4) - (i_2 + i_3)}{i_1 + i_2 + i_3 + i_4} \end{aligned} \tag{2.3}$$

CCD senzor

CCD senzor je eno ali dvodimenzionalno polje celic, občutljivih na svetlobo. Ko svetlobni žarek vpade na CCD celico, se ustvarijo elektroni in vsaka celica nakopiči število elektronov sorazmerno količini vpadne svetlobe na to celico za določen čas mirovanja. Število elektronov, nakopičenih v celici, predstavlja osvetljenost slikovnega elementa. Celica brez elektronov predstavlja črn, neosvetljen slikovni element. Zbirka slikovnih elementov predstavlja digitalno sliko. Polje vrednosti osvetlitev lahko analiziramo in tako poiščemo najbolj osvetljeno celico. [15]

Od časa mirovanja je odvisno, koliko časa bo celica hranila nakopičene elektrone. Čas mora biti dovolj dolg, da zagotovimo zadostno razmerje med signalom in šumom med iskanjem središča vpada svetlobe, hkrati pa mora biti čas mirovanja čim manjši, saj je od njega odvisna hitrost posodobitve meritve.

2.8.3 Qualisys optični sistem

Za potrebe magistrskega dela bomo uporabili profesionalni sistem za zajem gibanja Qualisys. Sestavlja ga osem kamer za snemanje z visoko hitrostjo zajema, komplet pasivnih označevalnikov in pripadajoča programska oprema Qualisys Track Manager (QTM).

Kamere Oqus 3+ omogočajo snemanje 500 slik na sekundo z 1,3 MP v normalnem načinu delovanja in 1750 slik na sekundo z 0.3MP v hitrem načinu delovanja. Kamere so med seboj povezane z ethernet povezavo, kot zadnji v vrsti je povezan računalnik s programsko opremo. Okoli kamere so razporejene infra rdeče LED diode, ki oddajajo svetlobo za pasivne označevalnike. Sistem prav tako deluje z aktivnimi označevalniki. Med delovanjem LED diode utripajo z določeno frekvenco.

Slika 2.2: Kamera Qualisys Oqus.

Poleg podjetja Qualisys optične sisteme za zajem gibanja razvijajo tudi druga podjetja, med njimi bi izpostavili Vicon, Ariel system, CODA, Elite-Plus, Motion Analysis, Peak. Richards [22] primerja navedene sisteme.

2.8.4 Rekonstrukcija 3D položaja s sistemom Qualisys

Za rekonstrukcijo iz 2D slik v 3D točke sta potrebni vsaj dve kameri in poznavanje njunih notranjih in zunanjih parametrov.

Notranji parametri so:

- Goriščna razdalja leče
- Koordinate središča slike
- Radialni in tangencialni parametri popačenja leče.

Zunanji parametri so:

- Položaj goriščne točke v globalnem koordinatnem sistemu
- Orientacija senzorja kamere v globalnem koordinatnem sistemu.

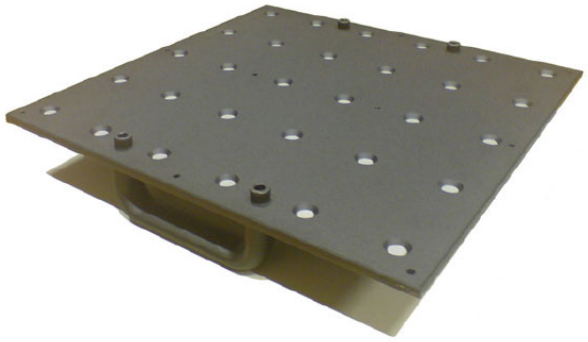
Ko so našteti parametri znani, lahko iz 2D slike točko pretvorimo v žarek, ki ima izhodišče v goriščni točki kamere in gre skozi 3D koordinato označevalnika. Z uporabo dveh kamer lahko natančno določimo presek žarkov za posamezen označevalnik.

Opisani postopek je razdeljen na podrobnejše naloge. Iskanje notranjih parametrov kamere je linearizacija, iskanje zunanjih parametrov kamere se imenuje umerjanje, sledenje pa je naloga ugotavljanja, kateri označevalnik na 2D sliki naj se uporabi za triangulacijo.

Linearizacija

Podatke, potrebne za linearizacijo zberemo tako, da pred kamero držimo ploščo (na sliki 2.3) z v mrežo postavljenimi označevalniki. Ploščo posna-

memo v različnih postavitvah pred kamero (različni koti, oddaljenost od kamere). Za izvedbo linearizacije položaji in pot premikanja plošče niso potrebni, pomembno je število zajetih slik, ki mora biti čim večje. Za vsako sliko se oceni položaj plošče in glede na ocenjene položaje se optimizira notranje parametre kamere, da kar čimbolj ustrezajo vsem slikam.



Slika 2.3: Plošča z mrežo označevalnikov, uporabljena za linearizacijo.

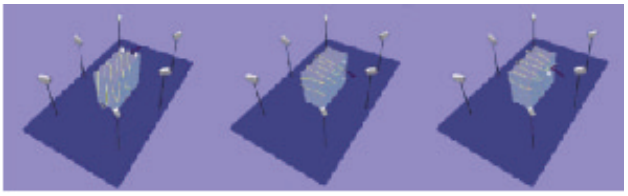
Qualisys kamere uporabljajo algoritem za linearizacijo, opisan v [23]. Algoritem poteka v štirih korakih:

- Določitev značilnih točk na slikah (označevalniki v mreži)
- Ocena notranjih in zunanjih parametrov z uporabo $closed - form$ rešitve
- Ocena radialnega popačenja z rešitvijo metode najmanjših kvadratov
- Izboljšava vseh parametrov z minimizacijo.

Umerjanje

QTM lahko umerimo na različne načine. Najbolj pogosta metoda je uporaba palice, ki ima obliko črke T in s katero označimo prostor, ki naj ga kamere pokrivajo. Palica ima nameščen po en označevalnik na vsakem koncu, pri metodi pa se uporabi tudi palica v obliki črke L, ki predstavlja koordinatno izhodišče opazovanega prostora in njegovo orientacijo.

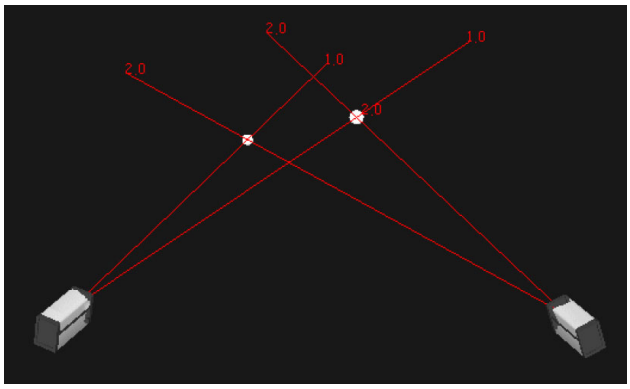
Ko s palico mahamo po prostoru, ustvarimo ogromno 2D koordinat, ne poznamo pa njihovih 3D koordinat. Zunanje parametre in 3D koordinate označevalnikov na palici v vseh zajetih časovnih okvirjih poiščemo s postopkom "popravek svežnja"(ang. bundle adjustment). Pri tem postopku QTM naredi globalno nelinearno optimizacijo in tako poišče zunanje parametre vseh kamer in nepoznanih 3D koordinat naenkrat. Podroben opis uporabljene metode je v [24].



Slika 2.4: Postopek umerjanja sistema.

Sledenje

Določanje, katere 2D označevalnike oziroma njihove žarke uporabiti za triangulacijo 3D označevalnika, je zahtevno. Upoštevati je potrebno težave, kot so celotna in delna zakritost označevalnika in navidezna združitev dveh označevalnikov. Naivni pristop bi pomenil izračun vseh presekov žarkov, s katerim bi dobili večje število označevalnikov, kot jih dejansko opazujemo. Da ne bi prišlo do napačnih presekov, QTM uporablja metodo, ki sledi označevalnikom, to je sledenje. Pri tej metodi QTM poskuša ustvariti krivuljo posameznega označevalnika v času, ki sledi dejanskemu označevalniku na opazovanem objektu. Iz obstoječe krivulje lahko QTM predvidi položaj označevalnika v naslednjem časovnem okvirju. Tako lahko žarke, ki se približajo predvidenemu položaju označevalnika, uporabi v triangulaciji 3D položaja. Podjetje Qualisys ne razkriva podrobne izvedbe metode, razkrivajo samo še, da zna njihov algoritem rešiti težave s skritimi in združenimi označevalniki v realnem času.



Slika 2.5: Grafični prikaz sledenja dveh označevalnikov.

2.9 Qualisys Track Manager (QTM)

QTM je namizna aplikacija, podprta na operacijskem sistemu Windows, ki je s kamerami povezana preko Ethernet povezave. Od kamer prejema zajete podatke o 2D položaju označevalnika in iz podatkov vseh kamer izračuna 3D položaj označevalnika v prostoru. Program omogoča nadzor nad nastavitvami kamer (število slik na sekundo, kvaliteto slike, itd.), njihovo kalibracijo. Če kamera to omogoča, tudi sprejema in prikazuje barvno sliko trenutnega snemanja.

QTM prikazuje 3D pozicijo vsakega označevalnika v kartezičnem koordinatnem sistemu z barvno kroglo. Posamezni označevalnik ali skupino leteh je mogoče poimenovati in povezati v model. Par dveh označevalnikov lahko povežemo s črto, imenovano kost (zaradi toge strukture prispodoba iz človeškega telesa). Pri zajemu gibanja človeka kosti v QTM-u ustrezajo polažaju kosti v telesu. Iz množice označevalnikov, povezanih s kostmi, ustvarimo model, ki ga lahko shranimo za uporabo pri prihodnjih zajemih gibanja. Tako shranjen model se imenuje Automatic Identification of Markers (AIM). [25]

Z uporabo modela AIM uporabnik pridobi veliko časa pri ponavljajočih se podobnih zajemih gibanja. Ob začetku zajema uporabnik na prejete pozicije označevalnikov naloži model AIM, ki naj bi ustrezal razporeditvi

označevalnikov na opazovanem objektu. QTM nato poskuša ugotoviti, kateri pari označevalnikov so povezanimi s kostmi iz modela. Tehnologija delovanja modela AIM in aplikacija le-tega na množico označevalnikov je poslovna skrivnost podjetja Qualisys. Iz analize shranjenih podatkov modela AIM ugotavljamo, da QTM poveže kosti iz modela AIM na podlagi primerjave odstopanj med razdaljami posameznih parov označevalnikov v času.

2.10 Razlogi za razvoj nove rešitve

QTM je kompleksno orodje z velikim naborom ukazov in funkcij. Kompleksnost se prenese tudi na zasnovo uporabniškega vmesnika, ki je zapleten in nepregleden. Hiter pogled ponuja samo osnovne funkcije, kompleksnost se začne kazati s poglabljanjem v posamezne menije orodja. QTM je prvotno namenjen nadzoru kamer in ne analizi zajetih podatkov. Vsakršno analiziranje podatkov se izkaže za zahtevno in časovno potratno dejanje. Dodatna slabost je, da QTM omogoča le omejen izvoz podatkov, katere bi raziskovalec rad uporabil v lastnih analitičnih aplikacijah.

Kot namizna aplikacija je QTM na voljo samo enemu uporabniku istočasno. Ne omogoča deljenja zajetih podatkov z drugimi zainteresiranimi v procesu (npr.: raziskovalci na projektu, študentje na predavanju). Podatke, katere je mogoče izvoziti iz QTM-a, se lahko deli z drugimi samo preko običajnih metod, kot so na primer izmenljivi mediji, e-mail, FTP. V najboljšem primeru lahko prejmenik prejme zajete podatke tik po prenehanju zajema, nikakor pa ne more pregledovati zajetih podatkov istočasno kot uporabnik QTM-a. S tem se podaljšuje čas in strošek zajema, saj lahko drugi vpleteni sporočijo napake ali izboljšave pri zajemu šele naknadno.

Poglavje 3

Načrtovanje lastne aplikacije za vizualizacijo in analizo gibanja

3.1 Zahteve

Glede na razloge iz prejšnjega poglavja smo določili zahteve, ki jih mora izpolnjevati nova rešitev. Zahteve bomo ločili na tehnične zahteve in zahteve uporabniškega vmesnika. Tehnične zahteve bomo objektivno ovrednotili glede na parametre, kot so: hitrost prenosa podatkov med aplikacijo QTM in našim sistemom, skupne zakasnitve in odzivni čas, frekvenca izrisovanja 3D scene, vpliv števila opazovanih točk na hitrost delovanja. Zahteve uporabniškega vmesnika bomo vrednotili s pomočjo uporabniške študije na osnovi subjektivnih kriterijev (mnenja posameznikov) in standardiziranega testa UEQ. Glede na končno vrednotenje razvite rešitve bomo lahko potrdili, ali smo uspeli razviti inovativnejšo in uporabniku bolj prijazno rešitev od obstoječe.

Tehnične zahteve:

- Hkratno pretakanje zajetih podatkov na večjem številu naprav
- Izrisovanje zajetih podatkov na odjemalcu s frekvenco 60 slik na sekundo (velja za naprave s strojno podporo izrisovanja)

- Možnost deljenja shranjenih modelov med aktivnimi odjemalci.

Zahteve uporabniškega vmesnika:

- Porabljen krajši čas za rešitev zastavljene naloge v primerjavi z obstoječo rešitvijo.
- Boljši rezultat UEQ testa, predvsem v kategoriji novost in privlačnost uporabniškega vmesnika.

3.2 Uporabljene tehnologije

3.2.1 QTM Real Time Protocol (RTP)

RTP omogoča prenos obdelanih podatkov iz QTM-a preko TCP ali UDP povezave. Prenos lahko poteka v realnem času ali na zahtevo iz shranjenih posnetkov. Struktura protokola je jasno definirana in omogoča osnovne funkcije, kot so samodejna zaznava vhodnih vrat, spreminjanje nastavitev kamer, upravljanje z QTM, pretakanje, sporočanje napak, itd.

Večina RTP ukazov je v sinhroni besedilni obliki (plain text). Ukaz, ki vrne trenutno uporabljeno verzijo aplikacije QTM, na primer pošlje `QTMVersion` strežniku, ki odgovori `QTM version is 2.3`.

Nekateri ukazi niso sinhroni in v besedilni obliki, ampak so tok binarnih vrednosti, imenovanih podatkovni okvirji (ang.: data frames). Vsak podatkovni okvir ima zaglavje, iz katerega je mogoče razbrati velikost okvirja in tip podatkov. Takšen ukaz je na primer `GetCurrentFrame`. Njegova uporaba je prikazana spodaj.

```
GetCurrentFrame All | ([2D] [2DLin] [3D] [3DRes]
[3DNoLabels] [3DNoLabelsRes] [Analog] [Force] [6D]
[6DRes] [6DEuler] [6DEulerRes])
```

Parametri ukaza določajo, katere komponente zajema gibanja naj vsebuje zahtevani podatkovni okvir. Podrobnosti parametrov so opisane v tabeli 3.1. V našem delu uporabljamo prenos preko TCP povezave v 3D strukturi z

oznakami označevalnikov, od načina uporabe pa je odvisno, ali se prenos izvaja v realnem času ali na zahtevo.

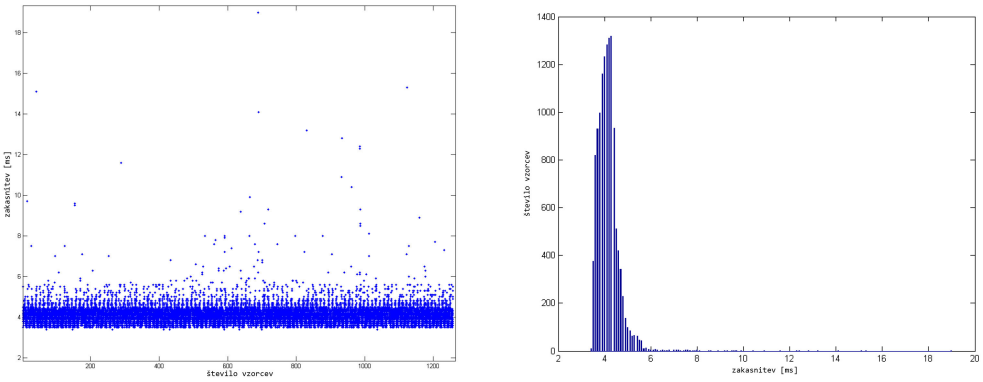
Tip ID	Ime	Opis
2	3DnoLabels	3D podatki neimenovanih označevalnikov
10	3DnoLabelsRes	3D podatki z ostanki neimenovanih označevalnikov
3	Analog	Analogni podatki iz ene ali več naprav
4	Force	Podatki ene ali več plošč za zaznavanje sile
5	6D	6D pozicija in matrika vrtenja
11	6DRes	6D pozicija in matrika vrtenja z ostanki
6	6DEuler	6D pozicija in Eulerjevi koti
12	6DEulerRes	6D pozicija in Eulerjevi koti z ostanki
7	2D	2D pozicija označevalnikov na eni kameri
8	2DLin	Linearni 2D podatki

Tabela 3.1: Tipi komponent, ki jih omogoča RTP. [26]

Zakasnitve v realnem času

Arhitektura sistema Qualisys je prilagojena za nizke zakasnitve pri prenosu od kamere preko RTP. Zakasnitev je neodvisna od števila zajetih slik na sekundo, lahko pa se poveča z zahtevnostjo 3D rekonstrukcije. Na zahtevnost vpliva število kamer in število označevalnikov.

Meritev zakasnitve je bila izvedena na sistemu z 8 kamerami in 12 označevalniki. Izmerjen je bil čas od zajema slike na kameri do trenutka, ko so podatki na voljo za prenos preko RTP. Prenos preko RTP ni upoštevan, saj je ta čas odvisen od več faktorjev (zakasnitve omrežja, oddaljenost, hitrost prenosa). Rezultati meritev in porazdelitev zakasnitev glede na število vzorcev so prikazani na sliki 3.1. Izmerjena je bila zakasnitev 4,2 $\pm$ 0,6 ms. Ta zakasnitev velja za sisteme do 16 kamer in 100 označevalnikov. [27]



Slika 3.1: Levo: zakasnitev glede na število vzorcev. Desno: porazdelitev zakasnitev.

3.2.2 WebSocket API

Svetovni splet je bil zasnovan okrog protokola HTTP, ki sledi modelu poslane zahteve in odgovora strežnika. Zaradi omejitev tega modela in želje po bolj dinamičnem spletu se je razvila asinhrona XHR zahteva, ki je eden izmed gradnikov AJAX tehnologije. Asinhrona XHR zahteva omogoča sprožitev zahteve na strežnik brez osvežitve celotne strani. Prednosti takšne zahteve so hitrost in manjša obremenitev podatkovne povezave. Kljub prednostim pa še vedno ostaja omejitev, da je možna komunikacija med odjemalcem in strežnikom samo ob zahtevi odjemalca.

WebSocket API omogoča stalno dvosmerno (ang.: full-duplex) komunikacijo med odjemalcem in strežnikom. Po začetnem rokovanju lahko strežnik potisne podatke odjemalcu. Odjemalcu se sproži dogodek s podatki (odvisno od izvedbe odjemalca), katere lahko uporabi ali spusti mimo. Je del HTML5 specifikacije. Za razliko od asinhrone XHR zahteve ne temelji na protokolu HTTP, ampak na lastnem protokolu WS. Razlika med protokoloma je v velikosti zaglavja posameznega paketa. Protokol WS uporablja zaglavlja, velika med 48 biti do največ 112 bitov. Velikost je odvisna od velikosti vsebine paketa. Maksimalna velikost paketa je odvisna od izvedbe protokola. [28]

3.2.3 WebGL

WebGL je aplikacijski programski vmesnik, ki omogoča podporo OpenGL ES 2.0 v spletnih brskalnikih. Izveden je kot 3D kontekst znotraj HTML-ja, izpostavljen je le njegov Document Object Model (DOM) vmesnik. Uporablja OpenGL-ov jezik senčenja (ang.: shading language) GLSL ES. HTML element se obnaša enako kot ostali elementi. Idealen je za 3D dinamične aplikacije v JavaScript programskem jeziku. Podpirajo ga že vsi širše uporabljeni spletni brskalniki, tudi na mobilnih napravah. [29]

Izris enega okvirja v WebGL-u je lahko računsko zahteven in lahko začasno ustavi izvajanje JavaScript kroga dogodkov (ang.: event loop). Izvedba animacije mora biti zato izvedena s klicem metode `requestAnimationFrame`. Metoda bo izvedla izris okvirja ob prvem prostem časovnem intervalu, s tem pa ne bo blokirala izvajanja ostalih skript in uporabniškega vmesnika. Metoda tudi ustavi izvajanje izrisa okvirjev, če uporabnik ni osredotočen na stran (stran je minimizirana).

3.2.4 Three.js

V aplikaciji smo uporabili ogrodje Three.js, ki v visoko nivojske objekte in metode združuje nižje nivojsko logiko WebGL-a.

Nekatere izmed funkcionalnosti, ki jih omogoča Three.js, so:

- Proces izrisovanja je skrit v abstraktni metodi in podpira tudi izrisovanje brez WebGL podpore (izrisovanje se izvaja na CPU namesto GPU)
- Osnovni gradniki v 3D grafiki (materiali, mreže, luči, kamera) so objekti, ki se medsebojno povezujejo v sceno
- Vgrajene metode za interakcijo s sceno (raycasting)
- Omogoča uvoz modelov iz programov, kot so Blender in 3ds Max. [30]

Za uporabo ogrodja v aplikaciji smo se odločili zaradi hitrejšega delovnega procesa, ki ga le-ta omogoča, pri izbiri ogrodja Three.js pa je prevladala njegova razširjenost med razvijalci. Prednost projekta, ki ga podpira velika skupnost, je hitro odpravljanje napak in pomoč ob posameznih problemih.

3.2.5 Node.js

Node.js je platforma, zgrajena na osnovi Googlovega JavaScript izvajalnega okolja V8, za enostavno gradnjo hitrih, skalabilnih omrežnih aplikacij. Node.js uporablja dogodkovno gnan vhodno izhodni model brez blokiranja (ang.: event-driven non-blocking I/O model), ki ga odlikuje lahkotnost in učinkovitost. Primeren je za podatkovno usmerjene aplikacije v realnem času, ki se izvajajo porazdeljeno po napravah. [31]

Node.js je ob pravilni rabi hitrejši od nekaterih širše uporabljenih strežniških skladov (npr.: Apache, Nginx, PHP). Našteti skladi imajo za razliko od Node.js, kjer je strežnik izveden v izvajalnem okolju, ločen strežnik in izvajalno okolje programskega jezika.

Node.js izvaja programske ukaze v istem procesu v eni niti, imenovani dogodkovni krog (ang.: event loop). V vsakem krogu se izvede en dogodek. Zaradi takšnega načina delovanja mora biti aplikacija napisana asinhrono, brez čakanja na odgovore vhodno/izhodnih vmesnikov. Ko prejme odgovor vmesnika, ga postavi na zadnje mesto v čakalni vrsti. Med sinhronim čakanjem morajo vsi dogodki čakati v vrsti, dokler le-ta ne prejme odgovora in se zaključi. Na takšen način lahko Node.js rokuje z velim številom istočasnih povezav.

3.3 Načrtovanje uporabniškega vmesnika

3.3.1 Uporabniška izkušnja

Koncept uporabniške izkušnje združuje vidike, kot so uspešnost, učinkovitost, estetika, privlačnost za uporabo. Uspešnost in učinkovitost spadajo med

pragmatične vidike [32], ostale pa uvršamo med hedonske vidike kvalitete izkušnje. [33]

Raziskave kažejo [34], da znajo proizvode, ki imajo visoko stopnjo kvalitet, kot so estetika in privlačnost (hedonske), uporabniki lažje uporabljati.

Podrobneje bo vrednotenju aplikacije med pragmatične in hedonske vidike posvečeno poglavje o uporabniški študiji.

3.3.2 Interakcija s 3D prostorom

Pretežni del uporabniškega vmesnika predstavlja 3D prostor, v katerem je vedno prikazano izhodišče koordinatnega sistema s puščicami v smeri osi x, y, in z. Uporabniki so navajeni iz pogosto uporabljenih aplikacij (npr.: Google Maps, Google Earth), da s 3D prostorom upravljajo z miško. Klik miške in poteg v eno smer predstavljata zasuk okoli osi, uporaba manjšega kolesca na miški pa približevanje in oddaljevanje prostora. Za podporo uporabnikom brez miške oz. mišk, ki kakorkoli ne omogočajo navedenih akcij, smo dodali podporo interakcije s 3D prostorom preko tipkovnice. Smerne tipke predstavljajo zasuk osi, tipki *page up* in *page down* pa približevanje in oddaljevanje od centra 3D prostora. Uporabnik lahko 3D prostor tudi premakne, to stori s preklopom načina interakcije v aplikaciji. Klik in poteg miške se potem namesto kot rotacija, obnaša kot premik koordinatnega izhodišča.

V 3D prostoru bodo predstavljeni zajeti označevalniki. Zaradi lažje miselne preslikave iz aplikacije na zajet objekt so lahko označevalniki obarvani. Določena barva lahko predstavlja skupino označevalnikov (npr.: roke rdeče barve, noge zelene barve). S klikom na označevalnik ga uporabnik označi, s ponovnim klikom na označevalnik odznači iz trenutnega izbora. Možen je izbor večjega števila označevalnikov skupaj tako, da uporabnik hkrati drži tipko *ctrl* in klikne na označevalnik. Princip izbire večjega števila označevalnikov je izposojen iz brskalnika operacijskega sistema, saj je uporabniku že poznan.

3.3.3 Ustvarjanje modela

Model je predstavljen kot graf med seboj povezanih označevalnikov. Tako je tudi predstavljen v 3D prostoru. Povezave med označevalniki se imenujejo kosti. V aplikaciji smo poskušali zasnovati postopek ustvarjanja modela, ki naj bo čim bolj enostaven in intuitiven za uporabnika. Posamezno kost uporabnik ustvari tako, da izbere dva še nepovezana označevalnika. Ob izboru se v uporabniškem vmesniku pojavi gumb "ustvari kost". S klikom je kost ustvarjena. Uporabniški vmesnik omogoča tudi ustvarjanje večjega števila kosti naenkrat. Uporabniku je omogočeno tudi brisanje posamezne kosti s klikom nanjo in uporabo tipke "delete". Ko so vse kosti ustvarjene, uporabnik modelu določi ime ter ga shrani za nadaljnje delo. Uporabnik ima pregled nad shranjenimi modeli v navigacijski plošči. Model lahko naloži na označevalnike v 3D prostoru ali izbriše.

3.3.4 Akcije in analize

Uporabnik do vseh akcij in analiz dostopa v navigacijski plošči. Navigacijska plošča je omejena samo na v danem času razpoložljive akcije ali analize. Če ima uporabnik izbran samo en označevalnik, ne more ustvariti kosti, zato mu te akcije ne prikažemo. S skrivanjem akcij in analiz smo hoteli poenostaviti uporabniški vmesnik in s tem pomagati uporabniku hitro najti želeno akcijo. Na ta način smo se izognili tradicionalni razporeditvi v menije in podmenije.

3.3.5 Sodelovanje uporabnikov in sinhronizacija podatkov

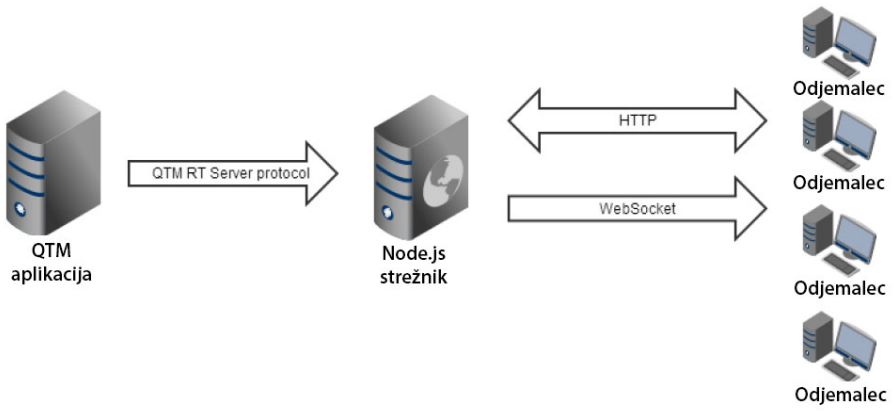
Sodobni delovni procesi temeljijo na sodelovanju vpletenih z različnih delov sveta preko spletnih aplikacij. Ta način dela želimo s predlagano rešitvijo vnesti tudi v analizo gibanja. Primera takšnega dela sta: zajem gibanja se izvede v enem laboratoriju, podatke pa analizira oddaljen raziskovalec v drugem laboratoriju; profesor želi med predavanji omogočiti študentom individualno analizo na obravnavanem modelu, kjer študentje vedno pogosteje

uporabljajo lastne mobilne naprave. V naštetih primerih mora aplikacija ponujati orodja za sodelovanje in deljenje podatkov.

Predlagana rešitev omogoča oddaljenemu raziskovalcu spremljanje zajemanja v živo. Raziskovalec lahko sproti sporoča popravke, s tem se čas zajema precej skrajša. V drugem primeru lahko profesor že ustvarjen model deli s študenti, študentje pa ga lahko uporabijo, popravijo ali ustvarijo nove lastne modele na zajetih podatkih.

3.4 Arhitektura aplikacije

Sistem sestavljajo tri glavne enote: aplikacija QTM, Node.js strežnik in odjemalec. Število aplikacij QTMV magistrskem delu predpostavljamo, da je aplikacija QTM samo ena. in odjemalcev je lahko večje, medtem ko je strežnik en sam. Podatki o poziciji vseh zajetih označevalnikov se prenesejo iz aplikacija QTM na strežnik preko RTP-ja. Strežnik prevede prejete podatke iz binarnega toka podatkov v JavaScript Object Notation (JSON), ki ga razpršeno oddaja vsem trenutno povezanim odjemalcem. Posamezen odjemalec iz prejetih podatkov rekonstruira vizualizacijo 3D scene, kot jo vidi uporabnik aplikacije QTM.



Slika 3.2: Strežniška arhitektura aplikacije.

3.4.1 Strežniška arhitektura

Strežniška Node.js aplikacija ima tri glavne module, ti so: module HTTP, module WebSocket in modul RTP.

Modul HTTP je sestavni del okolja Node.js. Uporabljen je za komunikacijo preko protokola HTTP med strežnikom in odjemalci. V aplikaciji opravlja predvsem nalogo strežnika statičnih vsebin, kot so CSS, JS in glavna index.html datoteka. Vsak odjemalec najprej sproži zahtevo na strežnik preko modula HTTP, vzpostavi se seja, ki je potrebna za razlikovanje med odjemalci ob WebSocket komunikaciji. Ločeno od osnovnega uporabniškega vmesnika preko modula HTTP deluje administratorski vmesnik. V administratorskem vmesniku upravljalec aplikacije QTM določi IP naslov in vrata le-te, hitrost prenosa v okvirjih na sekundo ter požene ali ustavi pretakanje podatkov.

Modul RTP modul je namenski modul za potrebe aplikacije. Zaradi splošnega aplikacijskega vmesnika, značilnega za Node.js module, je lahko uporabljen tudi v drugih aplikacijah, ki uporabljajo okolje Node.js. Modul izvaja osnovne ukaze QTM protokola RT ter razčlenjuje binaren tok podatkov med pretakanjem. Ob administratorjevi zahtevi za povezavo na QTM modul RTP opravi rokovanje s QTM-om ter prenese datoteko XML z vsemi parametri trenutnega zajema gibanja. Izpostavili bi parametre z nazivi označevalnikov in njihovimi barvami, ki se prenesejo k odjemalcem. Datoteka s parametri se lahko prenese tudi, ko pretakanje že poteka, vendar v tem primeru odjemalci ne dobijo popolne informacije o označevalnikih.

Poleg pošiljanja ukazov QTM-u je glavna naloga modula RTP translacija zajetih podatkov iz binarnega toka v odjemalcem primernejšo obliko JSON. JSON je tipični zapis podatkov za izmenjavo z aplikacijami JavaScript. Struktura JSON posnema obliki objekta (ang. object) in polja (ang. array) iz programskega jezika JavaScript, slednjo je zato mogoče hitreje prevesti kot na primer strukturo XML. V vsakem zajetem časovnem okvirju so označevalniki predstavljeni z objektom, v katerem x, y, z ključi predstavljajo odmik označevalnika v podano smer iz središča opazovanega koordinatnega

prostora v milimetrih. Poleg podatkov o poziciji označevalnikov so v objektu časovnega okvirja še nekateri drugi podatki (npr.: tip časovnega okvirja in njegova zaporedna številka). Slednja je pomembna v primeru, ko zaradi značilnosti v prenosu preko protokolnega sklada TCP/IP nekateri paketi izpadejo iz zaporedja in jih je potrebno počakati ali preskočiti.

Modul WebSocket je namensko razvit modul za potrebe aplikacije. Modul temelji na protokolu WebSocket in v njem predvidenem API-ju. Njegovo delovanje je odvisno od predhodno predstavljenih modulov. Preko modula HTTP se v odjemalec prenese odjemalec WebSocket. Spletni brskalniki sicer podpirajo WebSocket API, vendar lasten odjemalec WS razširja osnoven API in olajša ter pohitri delo. Ob začetnem rokovanju strežnik shrani sejo in lahko prosto pošilja podatke odjemalcu. Strežnik ima več možnosti pošiljanja podatkov, lahko pošlje vsem prijavljenim odjemalcem naenkrat ali posameznemu odjemalcu. Po začetem prenosu zajetih podatkov modul deluje v prvem načinu pošiljanja, v razpršenem oddajanju. Oddajanje se zaključi z ustavitvijo prenosa ali ob zahtevi posameznega odjemalca za prekinitev.

Modul WebSocket je ohlapno povezan z modulom RTP, kar pomeni, da ob napaki v slednjem, modul WebSocket še vedno deluje. Z modulom RTP komunicirata preko teme. Ko modul RTP prejme časovni okvir, sproži dogodek na temi, katero spremlja modul WebSocket in časovni okvir posreduje odjemalcem.

3.4.2 Arhitektura na odjemalcu

Aplikacijo na odjemalcu sestavljata dva modula, modul WebSocket in modul WebGL.

Modul WebSocket je preprosta izvedba, ki temelji na WebSocket API-ju. Glavna naloga je vzpostavitev povezave s strežnikom ob začetku inicializacije celotne aplikacije. Povezava je vzpostavljena celoten čas življenja aplikacije. Modul smo poskušali ohraniti čim vitkejši, da ne blokira delovanja modula WebGL. Ob prejetju podatkov JSON s strežnika jih modul samo shrani v lokalno spremenljivko.

Modul WebGL je zadolžen za stalno izrisovanje 3D scene in prikazovanje v HTML5 elementu *canvas*. Scena se izrisuje stalno zaradi stalnih sprememb. Spremembe se lahko zgodijo zaradi dveh dejavnikov. Prvi je sprememba pozicije označevalnikov, ki jo aplikacija prejme preko modula WebSocket od strežnika. Strežnik, razen med pavzo, osvežuje sceno z minimalno frekvenco 40 okvirjev na sekundo. Drugi dejavnik sprememb scene je uporabniški vmesnik. Uporabnik lahko preko uporabniškega vmesnika zavrti sceno okrog dveh osi istočasno in jo lahko približuje in oddaljuje. Oba tipa sprememb lahko programsko zaznamo in spremembe na sceni bi lahko izrisali samo ob dejanski spremembi scene. Na ta način bi bil zahtevan računski čas aplikacije manjši, vendar smo se zaradi odzivnejšega uporabniškega vmesnika in lažje izvedbe odločili za stalno izrisovanje. Modul WebGL uporablja vgrajeno metodo brskalnikov `window.requestAnimationFrame`. Metoda kot argument sprejme povratno metodo (ang. callback method), katera izriše nov okvir scene. Povratna metoda se mora obnašati rekurzivno in po izrisu okvija scene ponovno klicati metodo `window.requestAnimationFrame`. Pri načrtovanju aplikacije je bil eden od pogojev hitrost izrisovanja 60 slik na sekundo. To smo z modulom WebGL dosegli na napravah, ki omogočajo izrisovanje na namenski strojni opremi.

Poglavje 4

Izvedba

V tem poglavju bomo predstavili izvedbo najpomembnejših delov aplikacije, ki predstavljajo doprinos magistrskega dela. To so analiza zajetih 3D podatkov, ki jo bomo predstavili na primeru analize kota in algoritem za predlaganje obstoječih modelov. Vsak del bo predstavljen s kratkim opisom in izsekom iz izvorne kode.

4.1 Analiza podatkov

Analize prejetih podatkov iz sistema za zajem gibanja so pomemben del aplikacije. Njihovo zasnovo smo poskušali zgraditi kar čim bolj modularno. To nam bo omogočalo hitro dodajanje novih vrst analiz v prihodnosti. Vse analize podatkov se izvajajo na odjemalcu, s čemer želimo razbremeniti obremenjenost strežnika, hkrati pa je hitrost izvedbe analize povezana z zmogljivostjo uporabnikove naprave.

Med prejemanjem podatkov s strežnika za prikaz označevalnikov v 3D prostoru odjemalec shranjuje podatke časovnih okvirjev v lokalno shrambo. Shranijo se številka časovnega okvirja in podatki koordinat x, y, z vseh označevalnikov v tem časovnem okvirju.

Glavna funkcija je `getFrames`, ki sprejme dva parametra. Prvi parameter je funkcija, ki zmanjša število označevalnikov v podanem časovnem okvirju

in jih vrne v zahtevani obliki za nadaljnjo obdelavo. V podanem primeru je to funkcija `compare`. Ta izloči tri označevalnike, ki opisujejo želeni kot, in jih vrne kot JavaScript objekt s ključi *a*, *b*, *c*.

Drugi parameter je funkcija, ki prejme tri označevalnike iz funkcije `compare` in izračuna kot med njimi. Dobljeni rezultat shrani v objekt z vsemi podatki za grafično predstavitev analiziranih podatkov. Ta vsebuje še tip analize, na podlagi katerega izriše pravilen graf.

Zaradi modularne zasnove postopka lahko metodo `getFrames` razširimo tako, da prejme dodatna parametra za začetek in konec želenega intervala analize, saj lahko trenutno uporabnik analizira le celoten čas zajema, v prihodnje pa moramo razmisliti tudi o možnosti izvajanja analiz na strežniku. V spodnjem izseku izvorne kode je predstavljena funkcija `getFrames`, kot asinhrona funkcija, ki za vsak časovni okvir doda novo instanco funkcije v JavaScript dogodkovno vrsto.

Izvorna koda analize kota v času

```
function getFrames(compare, function (err, frames) {
  _.each(frames, function (val, key) {
    var orgAB = getOrigin(ray1, vertex);
    var orgBC = getOrigin(vertec, ray2);
    var angle = getVectorsAngle(orgAB, orgBC);
    data.push({x: val.frameNo, y: angle});
  });
  callback(data);
});
```

Pri izvajanju analiz na odjemalcu se pojavi problem, da pri nižji frekvenci prejemanja podatkov časovnih okvirjev nastanejo "luknje"med časovnimi okvirji. To je težko zaznati med vizualnim spremljanjem 3D prostora, ampak lahko privede do napak v končnih podatkih, sploh pri podatkih, ki se navezujejo med seboj v času (npr.: hitrost). Problem se pojavi tudi pri shranjenih

zajemih[1], saj je potrebno počakati do konca prenosa, preden lahko izvedemo celotno analizo.

4.2 Prepoznavanje obstoječih modelov

Za analizo podatkov zajetega gibanja in za lažje delo z zajetimi podatki uporabnik označevalnike lahko poveže v model. Model predstavljajo označevalniki, ki jih je zaznal sistem za zajem gibanja in povezave med njimi, imenovane kosti. V aplikaciji QTM se model imenuje model AIM.

Ker je možnost povezovanja v modele pomembna funkcija aplikacije in podatka o strukturi modela ne prejmemo prek protokola RT, smo želeli uporabniku olajšati delo in mu predlagati, kateri model najbolj ustreza zajeti strukturi označevalnikov.

Za izračun ujemanja shranjenega modela in trenutne strukture označevalnikov smo izbrali metodo simuliranega ohlajanja, s katero zmanjšamo število preverjenih rešitev v primerjavi s pregledom vseh možnih rešitev.

4.2.1 Definicija modela

Model sestavljajo označevalniki (vozlišča) in kosti (robovi). Gre torej za neusmerjen graf. Vsakemu robu lahko določimo tudi utež, ki je njegova dolžina.

Posamezna kost naj bi sicer določala fiksen del na opazovanem objektu (od tu tudi njeno poimenovanje), vendar lahko pride do spreminjanja njene dolžine. Razloga za to sta lahko dva: prvi je, da kost ni določena na fiksnem delu opazovanega objekta, drugi pa se pojavi zaradi namestitve označevalnika. Označevalnik je potrebno v večini primerov namestiti na zunanji del sklepa in to privede do manjših napak pri meritvi.

[1]Zajem, ki je shranjen v QTM in se ne izvaja v živo.

4.2.2 Simulirano ohlajanje

Simulirano ohlajanje (ang.: simulated annealing) je ne-deterministična metoda za reševanje optimizacijskih problemov. Uporablja se jo za iskanje globalnega minimuma v prostoru z več lokalnimi minimumi. Metodo sta istočasno razvila S. Kirkpatrick [35] in V. Černy [36].

Ime in zamisel za metodo izvirata iz metalurgije, kjer z nadzorovanim počasnim ohlajanjem taline dosežejo pravilno strukturo kristalov v snovi in s tem manj napak v končnem materialu. Ideja počasnega ohlajanja je prenesena v algoritem simuliranega ohlajanja tako, da algoritem počasi zmanjšuje verjetnost sprejetja slabše rešitve med raziskovanjem prostora. Sprejetje slabše rešitve je poglavitna lastnost hevrističnih metod, saj s tem omogoča širše iskanje optimalnejše rešitve.

Osnovni elementi simuliranega ohlajanja so:

- Končna množica vseh rešitev S
- Funkcija $P()$, ki izračuna verjetnost sprejetja trenutne rešitve
- Funkcija $neighbor()$, ki vrne naključno sosednjo rešitev
- Funkcija $E()$, ki izračuna energijo posamezne rešitve s iz množice S
- Funkcija $temperature()$, ki ob vsaki iteraciji zmanjša temperaturo.

Data: $G_s(V, R)$

Result: $s \in S$

uvodna nastavitev

$s = neighbor()$

$k = 0$

while $T > 1$ **do**

  $T \leftarrow temperature()$

  $s_n \leftarrow neighbor(s)$

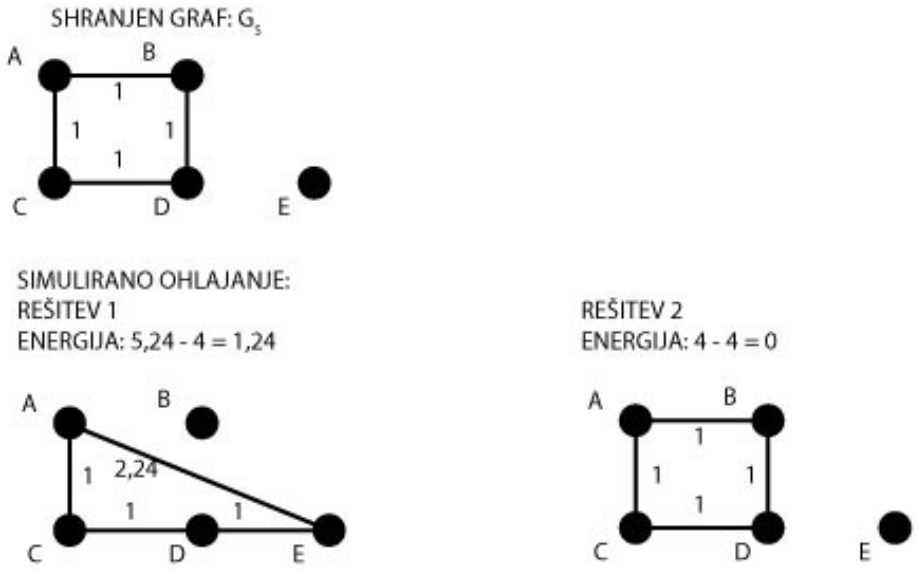
  if $P(E(s), E(s_n), T) > random(0, 1)$ **then**

    $s \leftarrow s_n$

  end

end

Algorithm 1: Simulirano ohlajanje



Slika 4.1: Postopek izvedenega simuliranega ohlajanja v dveh korakih.

4.2.3 Generiranje rešitve

Najpomembnejši del metode je generiranje rešitev. Rešitev generira funkcija *neighbor*(). Funkcija vrne graf z enako strukturo, kot jo ima shranjen graf.

Ob prvem klicu funkcije za vsako vozlišče v_s shranjenega grafa $G_s(V, R)$ funkcija izbere naključno vozlišče v_t trenutnega grafa $G_t(V, R)$ in iz izbranega vozlišča v_t ustvari enako število robov, kot jih ima shranjeno vozlišče v_s. Po končanem postopku dobimo prvo rešitev $s_0; s_0 \in S$;

Ob vsakem nadaljnjem klicu funkcije le-ta ne ustvari nove rešitve s_n po istem postopku kot prej. Na sliki 4.1 vidimo dve možni rešitvi za shranjen graf.

4.2.4 Energija rešitve

V metodi simuliranega ohlajanja manjša energija rešitve predstavlja boljšo rešitev. V podani izvedbi funkcije $E()$ je energija rešitve seštevek vseh uteži robov v rešitvi s_n, primerjanih z utežmi robov grafa $G_s(V, R)$.

$$E(s_n) = |\sum_{r_0; r_n \in R_t}^{r_n} dist(r_n) - \sum_{r_0; r_n \in R_s}^{r_n} dist(r_n)| \tag{4.1}$$

Funkcija $dist()$ vrne dolžino podanega roba. Ker se dolžina robov v času spreminja, se upoštevata najdaljša in najkrajša izmerjena razdalja.

Rezultat funkcije $E()$ je razlika med najkrajšimi in najdaljšimi razdaljami robov grafa $G_s(V, S)$ in rešitve s_n.

4.2.5 Verjetnost sprejetja nove rešitve

Funkcija $P()$ se za razliko od funkcij *neighbor*() in $E()$ navadno ne spreminja glede na različne uporabe metode. V [35] je funkcija $P(e, e', T)$ določena za dva primera. Ko je nova energija e' manjša od obstoječe energije e, vrne 1, v nasprotnem primeru vrne $exp(-(e' - e)/T)$. Rezultat funkcije se v vsaki iteraciji primerja z naključno vrednostjo med 0 in 1. Rešitev z nižjo energijo bo vedno sprejeta kot boljša, medtem ko bodo rešitve s slabšo energijo sprejete

samo takrat, ko bo rezultat $P()$ večji od trenutne naključne vrednosti. Ker na rezultat $P()$ slabše rešitve vpliva tudi parameter temperature T, bodo slabše rešitve sprejete samo ob visokih vrednostih T (visoka temperatura). Ko se vrednost T zmanjšuje (nizke temperature), se iskanje osredotoči samo še na boljše rešitve v ozki okolici trenutne rešitve (lokalni minimum).

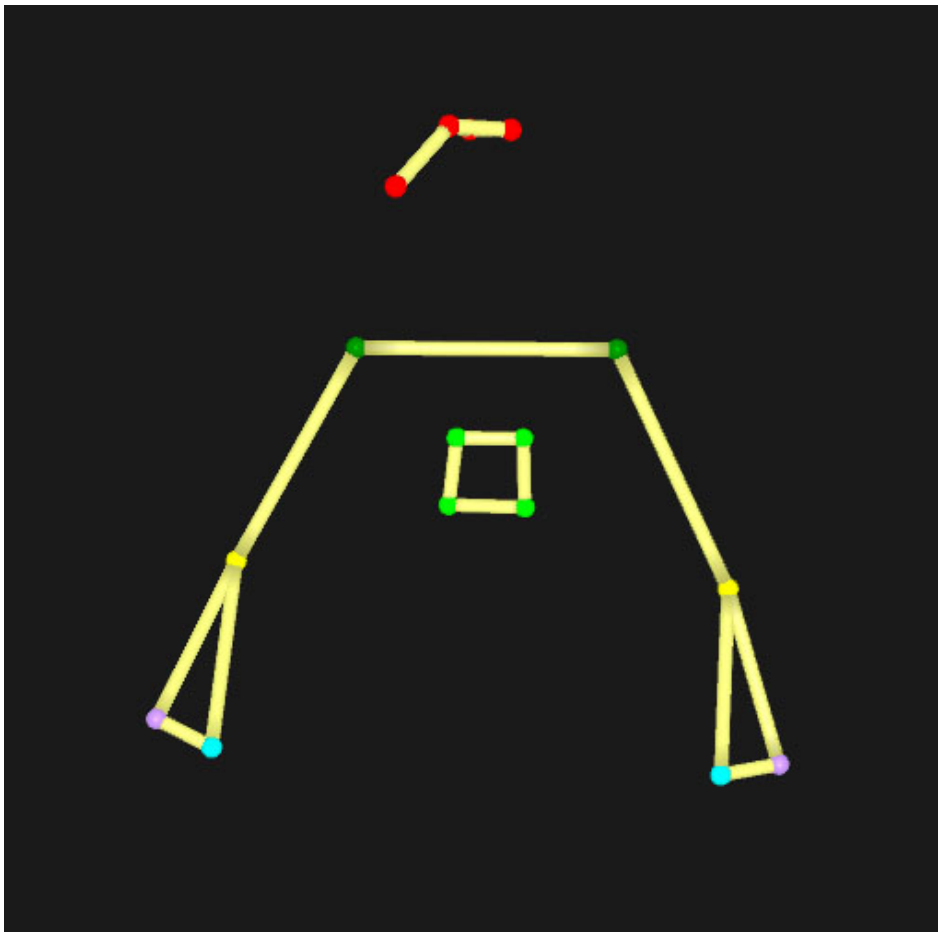
4.2.6 Evalvacija metode v primerjavi z modelom AIM

Z evalvacijo predlagane metode prepoznavanja obstoječih modelov želimo preveriti uspešnost metode v primerjavi za modelom AIM.

Model AIM podjetja Qualisys določi ustreznost modela glede na shranjena gibanja zajetega objekta. Posamezen označevalnik ima določen prostor, v katerem se je gibal, prav tako pa je pomembna lega glede na sosednje označevalnike (kot, oddaljenost do sosednjega označevalnika). [37]

Za uspešno določitev model AIM na trenutne označevalnike QTM potrebuje vsa možna gibanja objekta, medtem ko pri predlaganem postopku izbire modelov na izbiro vpliva samo struktura in razdalje med označevalniki.

Evalvacija bo potekala tako, da bomo obe metodi (predlagano in model AIM) preizkusili na devetih posnetkih zajetega gibanja. V posnetkih se osebe gibljejo v prostoru, vendar je v vsakem posnetku gibanje različno (hodijo v prostoru, se sklanjajo, premikajo roke, itd.) Na osebe je pritrjeno različno število označevalnikov, vsem pa je skupnih 16 označevalnikov, pritrjenih na glavi (4 označevalniki), rokah in ramah (8 označevalnikov) ter hrbtu (4 označevalniki), preostali označevalniki so dodatno razporejeni na prsih, trebuhu ali rokah. Iz skupnih šestnajstih označevalnikov smo sestavili model na kontrolnem posnetku (na sliki 4.2), ki ni vključen v poskusnih devet posnetkov. Iz vsakega posnetka smo uporabili samo odseke, dolge 1000 časovnih okvirjev.



Slika 4.2: Kontrolni model evalvacije s šestnajstimi označevalniki in sedemnajstimi povezavami.

Uspešnost obeh metod bomo ovrednotili na podlagi natančnosti in priklica, kjer:

- natančnost (P) pomeni delež pravilnih povezav med vsemi najdenimi,
- priklic (R) pomeni delež vseh najdenih povezav med vsemi obstoječimi.

$$F = 2\frac{P \cdot R}{P + R} \tag{4.2}$$

Rezultati evalvacije predlagane metode so predstavljeni v tabeli 4.1, rezultati modela AIM pa v tabeli 4.2. Na sliki 4.3 je predstavljen grafični prikaz vrednosti natančnosti in priklica. Iz dobljenih rezultatov je razvidno, da predlagana metoda slabše deluje, ko je poleg označevalnikov, povezanih v model, v posnetku prisotnih še veliko drugih prostih označevalnikov. Na posnetkih, kjer so bili prisotni označevalniki na prsih, je imela predlagana metoda težave z razlikovanjem teh označevalnikov s tistimi na hrbtu (oboje predstavljajo štirje označevalniki, povezani v kvadrat). Pri modelu AIM pa opazimo samo nepopolno natančnost na posnetku, kjer oseba ni stala vzravnano.

Predlagana metoda je imela v vseh primerih popoln priklic, saj poskuša vedno razporediti vse povezave med označevalnike in pri tem izračunava ujemanje shranjenega modela z označevalniki v posnetku, za razliko od modela AIM, ki poveže samo označevalnike, za katere je prepričan, da so pravilni.

Kljub slabšim rezultatom predlagane metode proti modelu AIM smo prepričani, da ima predlagana metoda pomembno prednost, to je, da sama preveri ujemanje vseh shranjenih modelov in uporabniku predlaga najboljšega, medtem ko mora v aplikaciji QTM uporabnik sam izbrati model AIM, ki ga želi aplicirati.

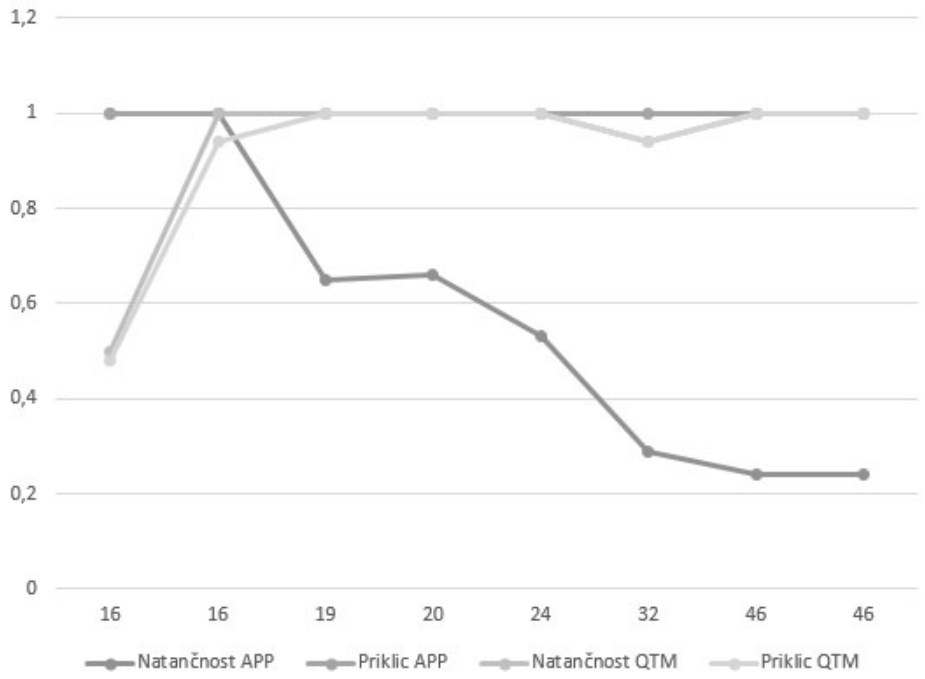
V prihodnje želimo predlagano metodo prepoznavanja obstoječih modelov nadgraditi predvsem tako, da bo prepoznala povezane označevalnike tudi ob večjem številu označevalnikov na posnetku in bo razlikovala med sicer podobnimi strukturami v modelu (npr.: razlika med prsmi in hrbtom).

Tabela 4.1: Rezulati evalvacije predlagane aplikacije.

posnetek	**št. označevalnikov v modelu**	**št. vseh najdenih povezav**	**št. pravilnih povezav**
k1	46	17	4
k2	46	17	4
m1	24	17	9
m2	20	17	11
m3	19	17	11
m4	32	17	5
m5	16	17	17
m6	16	17	17

Tabela 4.2: Rezulati evalvacije aplikacije QTM.

posnetek	**št. označevalnikov v modelu**	**št. vseh najdenih povezav**	**št. pravilnih povezav**
k1	46	17	17
k2	46	17	17
m1	24	17	17
m2	20	17	17
m3	19	17	17
m4	32	16	15
m5	16	8	4
m6	16	16	16



Slika 4.3: Natančnost in priklic za različno število označevalnikov v posnetkih. Vidimo lahko, da natančnost predlagane metode pada ob večjem številu označevalnikov v posnetku.

Poglavje 5

Uporabniška študija

Z namenom preizkusiti predlagano aplikacijo in potrditi njeno uporabnost in intuitivnost smo izvedli uporabniško študijo, v kateri smo primerjali obstoječo aplikacijo, QTM, s predlagano. Aplikaciji sta predstavljali dve neodvisni spremenljivki v uporabniški študiji. Tri odvisne spremenljivke so bile:

- Potreben čas za dokončanje zadane naloge
- Subjektivno vrednotenje aplikacije ocenjeno z UEQ vprašalnikom
- Pripombe, ki so jih podali udeleženci.

5.1 Udeleženci

V uporabniški študiji je sodelovalo 15 udeležencev. Večina udeležencev je imelo izobrazbo naravoslovne smeri (računalništvo - informatika, elektrotehnika, multimedija). Udeleženci so bili stari med 20 in 28 let ($M = 23{,}1$, $SD = 2{,}4$). En udeleženec je med uporabniško študijo uporabljal očala in en udeleženec je uporabljal kontaktne leče. Oba zaradi kratkovidnosti. Ostali udeleženci niso sporočili nobenih težav z vidom. Osem udeležencev je imelo predhodne izkušnje z 3D programsko opremo (uporabljali so aplikacije, kot so Blender, SketchUp, SolidWorks, itd.), noben od njih pa ni imel predhodnih

izkušenj z zajemom gibanja in analizo zajetih podatkov. Naključno so bili razdeljeni v dve skupini, vsaka skupina je uporabila samo eno od aplikacij.

5.2 Opis in potek študije

Uporabniško študijo smo zasnovali kot primerjavo med aplikacijama, kjer vsaka skupina uporabi samo eno aplikacijo. S tem smo se želeli izogniti učinku zaporedne uporabe aplikacij in zmedenosti oziroma predhodnih pričakovanj po uporabi prve aplikacije. Prva skupina, v kateri je bilo osem udeležencev je preizkusila predlagano aplikacijo, medtem ko je druga skupina s sedmimi udeleženci preizkusila obstoječo aplikacijo QTM. Poskus je bil razdeljen v dva dela.

V prvem delu poskusa je vsak udeleženec naloge uporabniške študije opravljal posamično, ob spremljanju vodje poskusa. Vodja je udeležencu pred pričetkom poskusa predstavil, kaj je in kako deluje sistem za zajem gibanja ter namen aplikacije, ki jo bo uporabil. Udeleženci niso dobili navodil za uporabo aplikacij, imeli so pet minut, v katerih so lahko sami spoznali, kaj jim le-ta ponuja. Po preteku petih minut se je poskus začel. Vodja je udeležencu prebral navodilo posamezne naloge in začel meriti čas. Udeleženec razen navodil nalog ni smel uporabiti dodatne pomoči. Ko je vodja ocenil, da je udeleženec končal zadano nalogo, je čas ustavil. Navodila za naslednjo nalogo je udeležencu podal šele po opravljeni predhodni nalogi.

V drugem delu poskusa so morali udeleženci iz obeh skupin izpolniti standardizirani vprašalnik UEQ in v njem oceniti uporabljeno aplikacijo.

V tretjem, zadnjem delu poskusa, jim je vodja poskusa zastavil še vprašanja glede njihove izkušnje z aplikacijo. Udeleženci so lahko podali svoja subjektivna mnenja in spremembe aplikacij. Med izvajanjem nalog si je vodja poskusa zapisoval probleme, ki so jih imeli udeleženci pri uporabi aplikacij med opravljanjem nalog.

Vsi računalniki, na katerih se je izvajala uporabniška študija, so bili povezani v omrežje preko hitre povezave. Dostopni časi in hitrost prenosa so

bili hitri in konstanti pri vseh poskusih, zato to ni vplivalo na porabljen čas, ki so ga udeleženci porabili za opravljanje nalog.

5.3 Zastavljene naloge

V prvem delu uporabniške študije je vsak udeleženec opravil tri različne naloge:

- Ustvari model tako, da označevalnike povežeš s kostmi. (*N1*)
- Shrani model. Ponovno odpri aplikacijo in uporabi shranjen model. (*N2*)
- Izvedi analize na modelu. (*N3*)
  - Analiziraj položaj desnega komolca. (*N3.1*)
  - Analiziraj dolžino nadlahti desne roke. (*N3.2*)
  - Analiziraj kot med nadlahtjo in podlahtjo desne roke. (*N3.3*)
  - Določi točno vrednost prejšnje analize na podanem časovnem okviru. (*N3.4*)
  - Shrani analizirane podatke v datoteko v surovi obliki. (*N3.5*)

5.4 UEQ

Vprašalnik ocenjuje udeležencev občutke, vtise in odnos do testirane aplikacije. Uporabimo ga lahko na dva načina. Prvi primer uporabe vprašalnika je, kadar želimo preveriti, ali obstoječa aplikacija omogoča uporabniku prijazno uporabo, ali pa če želimo zgolj ugotoviti, kako se uporabnik počuti ob njeni uporabi. Drugi način je stalno spremljanje kvalitete uporabniške izkušnje med razvojnim procesom. Z vsako novo verzijo aplikacije dobimo nove rezultate UEQ vprašalnika in tako lahko spremljamo napredek v razvoju. Za potrebe magistrskega dela bomo uporabili drugi pristop, kjer bomo namesto ocene nove verzije aplikacije vrednotili novo aplikacijo.

V nadaljevanju podrobneje opišemo zgradbo in vrednotenje UEQ vprašalnika, strukturo šestih lestvic ter analizo pridobljenih podatkov.

Zgradba in vrednotenje UEQ

Postavke in lestvice so bile določene po analitičnem pristopu. Najprej je bil sestavljen seznam 229 postavk na podlagi večkratnih srečanj strokovnjakov za uporabniške vmesnike. Seznam je bil nato skrčen na 80 postavk. Izločene so bile predvsem sopomenke. Skrčen seznam je bil uporabljen v več študijah, ki so se osredotočale na kvaliteto interaktivnih izdelkov. Zbrani so bili rezultati 153 udeležencev. Končne lestvice in postavke za vsako lestvico so določili s faktorsko analizo.

Zanesljivost in veljavnost (ali lestvice res merijo določeno kvaliteto) je bila preverjena v več študijah. Pregled vseh teh študij je pokazal, da je zanesljivost lestvic dovolj visoka. [38]

Struktura UEQ

Vprašalnik je razdeljen v 6 lestvic s skupaj 26 postavkami:

- Privlačnost: Ali je aplikacija uporabnikom všeč ali ne?
- Učinkovitost: Ali je mogoče aplikacijo uporabljati hitro in učinkovito? Ali je vmesnik urejen in pregleden?
- Jasnost: Ali je razumevanje uporabe aplikacije enostavna? Ali je navajanje na uporabo hitro?
- Zanesljivost: Ali je uporaba vmesnika varna in predvidljiva?
- Spodbuda: Ali je uporaba vmesnika zanimiva?
- Novost: Ali je vmesnik inovativen in kreativen? Ali izgled privlači uporabnika?

Vrstni red postavk na vprašalniku je razporejen naključno, tudi orientacija protipomenk ni vedno enaka (včasih je pozitivna postavka na levi, včasih na desni strani). V uporabniški študiji smo uporabili angleško verzijo vprašalnika, saj bi morali slovenski prevod najprej preizkusiti v ločeni študiji, zato so morali vsi udeleženci imeti vsaj osnovno znanje angleščine. Vprašalnik je prikazan na sliki 5.1.

Uporaba UEQ vprašalnika

Pri uporabi vprašalnika potrebujemo dve skupini udeležencev: uporabnike in vodje. Za uspešno rešen vprašalnik moramo paziti na sledeče:

- Uporabnik mora poznati ozadje in namen raziskave
- Omogočen mora biti osebni stik z vodjo poskusa
- Če isti uporabnik večkrat izpolnjuje vprašalnik, mora med posameznimi poskusi preteči dovolj časa.

Vodja poskusa je uporabniku na voljo za razlago o vprašalniku in podrobnostih poskusa.

Analiza pridobljenih podatkov

V tem poglavju predstavljamo postopek za analizo podatkov UEQ vprašalnika. Analiza rezultatov celotne uporabniške študije sledi v naslednjem poglavju.

Analiza podatkov, zbranih z UEQ vprašalnikom, poteka v treh korakih. Prvi korak je potrditev Cronbach Alpha vrednosti (ali vse postavke merijo isto kvaliteto). Če je vrednost majhna, nekatere postavke v vprašalniku niso bile pravilno razumljene. V takšnem primeru se moramo odločiti, ali bo ta lestvica upoštevana v končnem rezultatu.

V drugem koraku so lestvice ovrednotene od -3 do +3. Vrednosti nad +1 so na uporabnika naredile pozitiven vtis, pri vrednostih pod -1 je imel uporabnik pomisleke glede njihove privlačnosti. Optimalna želena vrednost je +2, saj uporabniki redko izbirajo maksimalne vrednosti.

Tretji korak je neobvezen in možen samo ob primerjavi različnih verzij aplikacij oziroma različnih aplikacij. S primerjavo rezultatov dveh testov hitro vidimo napredek pri razvoju. Pri postavkah z veliko razliko v vrednosti je bil narejen največji napredek. Pri postavkah, ki so enake vrednosti, vendar je njihova vrednost majhna, so kandidati za nadaljnje spremembe.

5.5 Rezultati

V uporabniški študiji smo preverjali tri spremenljivke:

- Čas opravljene naloge
- UEQ vprašalnik
- Subjektivne pripombe udeležencev o aplikaciji.

5.5.1 Čas opravljene naloge

Čas, potreben za dokončanje posamezne naloge, je izmeril vodja poskusa. Čas smo začeli meriti tik po tem, ko je vodja udeležencu prebral navodila naloge. Meritev smo zaključili, ko je bil udeleženec prepričan, da je nalogo uspešno opravil. Slika 5.2 prikazuje čase opravljenih nalog za obe aplikaciji.

Podatke smo statistično obdelali z uporabo analize variance (ANOVA) in Bonferronijevim popravkom. Statistično pomembnost je predstavljala vrednost $p < 0,05$. Normalna porazdelitev podatkov je bila preverjena s Shapiro-Wilk testom.

Predlagana spletna aplikacija ima slabše rezultate pri prvi nalogi (N1), kjer so udeležanci povezovali označevalnike s kostmi in zgradili model. Pri tej nalogi ni bilo nobene statistično pomembne razlike, enako velja tudi za nalogo N2 in nalogo N3.1. V vseh ostalih nalogah je predlagana spletna aplikacija prekosila aplikacijo QTM.

Rezultati po nalogah:

- N1: $F(1, 12) = 3,733, p = 0.077$

- N2: $F(1, 12) = 2,037, p = 0.179$
- N3.1: $(F(1, 12) = 4,505, p = 0,055)$
- N3.2: $F(1, 12) = 20,618, p = 0,001$
- N3.3: $F(1, 12) = 15,826, p = 0,002$
- N3.4: $F(1, 12) = 43,153, p < 0,001$
- N3.5: $F(1, 12) = 14,182, p = 0,003$

Standardna deviacija je manjša in ustrezne stopnje natančnosti so ožje za naloge opravljene v predlagani spletni aplikaciji. Verjamemo, da je to posledica intuitivnega, zanesljivega in robustnega uporabniškega vmesnika.

5.5.2 Rezultati UEQ vprašalnika

Rezultati UEQ vprašalnika se merijo v šestih lestvicah. Srednje vrednosti v vseh šestih lestvicah so višje za predlagano aplikacijo kakor za aplikacijo QTM. Na sliki 5.3 je prikazan rezultat po posameznih lestvicah.

Podatke smo statistično obdelali z uporabo analize variance (ANOVA) in Bonferronijevim popravkom. Statistično pomembnost je predstavljala vrednost $p < 0,05$.

Statistično pomembne razlike smo izmerili samo za privlačnost, jasnost in zanesljivost. Vsi rezultati so:

- Privlačnost: $F(1, 12) = 11,703, p = 0,005$;
- Jasnost: $F(1, 12) = 11,456, p = 0,005$;
- Zanesljivost: $F(1, 12) = 12,928, p = 0,004$;
- Učinkovitost: $F(1, 12) = 2,001, p = 0,183$;
- Spodbuda: $F(1, 12) = 3,123, p = 0,103$;
- Novost: $F(1, 12) = 0,795, p = 0,390$;

5.5.3 Subjektivne opombe udeležencev

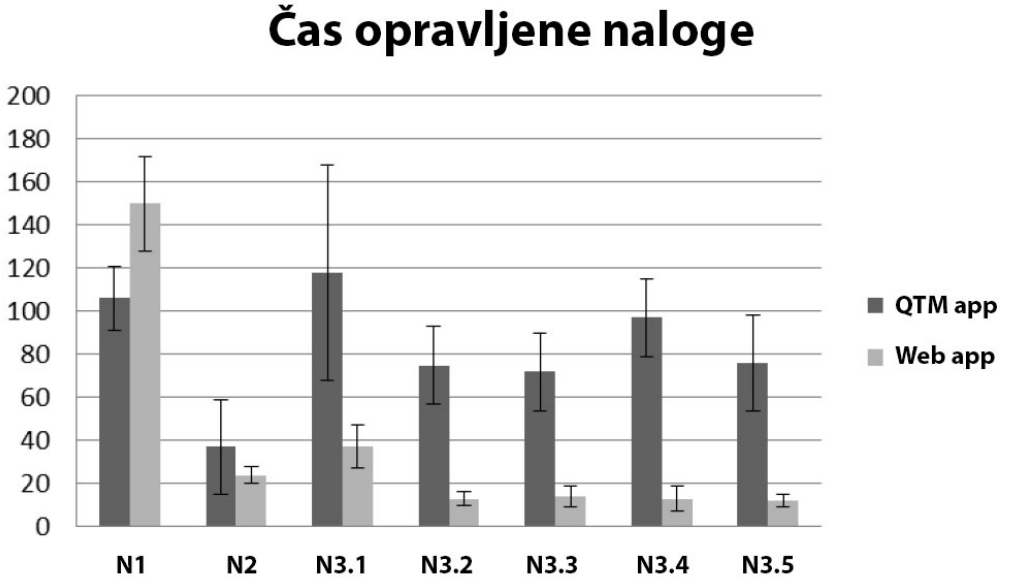
Po vsakem poskusu smo udeleženca vprašali o njegovem mnenju pri uporabi aplikacije. Predvsem so nas zanimali splošni komentarji in pripombe o dopolnitvi aplikacije. Nekaj najpogostejših odgovorov je bilo:

- Označevalniki so premajhni, zato jih težko izbrati.
- Manjkajo bližnjice na tipkovnici za pogoste funkcije.
- Nekateri opisi analiz niso dovolj jasni.
- Dvojni klik bi lahko kadarkoli prikazal izpis vseh možnih akcij.

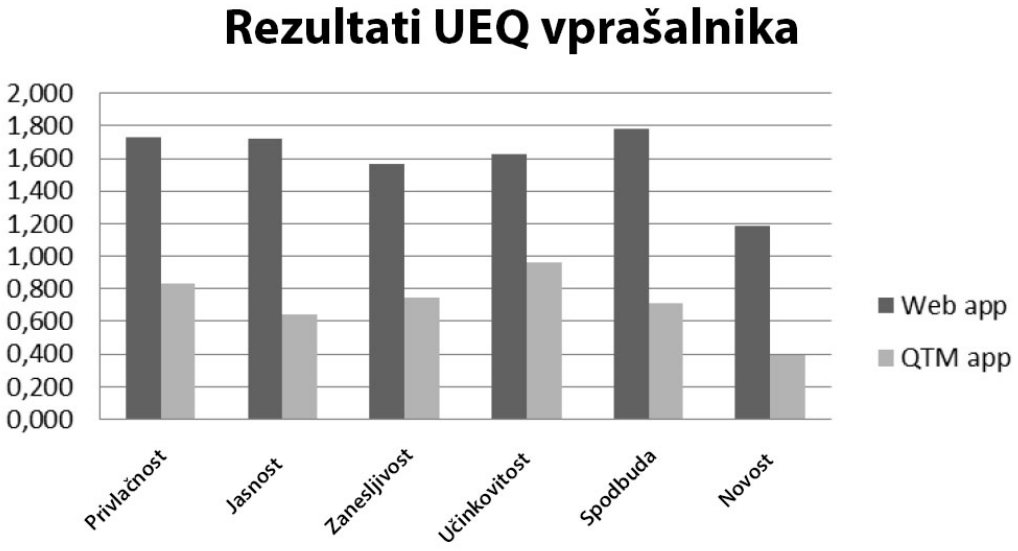
Udeleženci, ki so že uporabljali 3D program v preteklosti, so podali nekaj komentarjev glede interakcije s 3D prostorom. Težave pri njihovem delu so izhajale iz pričakovanj, ki so jih pridobili v predhodnih aplikacijah.

	1	...	7		
annoying	○	...	○	enjoyable	1
not understandable	○	...	○	understandable	2
creative	○	...	○	dull	3
easy to learn	○	...	○	difficult to learn	4
valuable	○	...	○	inferior	5
boring	○	...	○	exciting	6
not interesting	○	...	○	interesting	7
unpredictable	○	...	○	predictable	8
fast	○	...	○	slow	9
inventive	○	...	○	conventional	10
obstructive	○	...	○	supportive	11
good	○	...	○	bad	12
complicated	○	...	○	easy	13
unlikable	○	...	○	pleasing	14
usual	○	...	○	leading edge	15
unpleasant	○	...	○	pleasant	16
secure	○	...	○	not secure	17
motivating	○	...	○	demotivating	18
meets expectations	○	...	○	does not meet expectations	19
inefficient	○	...	○	efficient	20
clear	○	...	○	confusing	21
impractical	○	...	○	practical	22
organized	○	...	○	cluttered	23
attractive	○	...	○	unattractive	24
friendly	○	...	○	unfriendly	25
conservative	○	...	○	innovative	26

Slika 5.1: Uporabljen UEQ vprašalnik.



Slika 5.2: Povprečni časi v sekundah s stopnjo natančnosti.



Slika 5.3: Rezultat UEQ vprašalnika po lestvicah.

Poglavje 6

Zaključek

V magistrskem delu smo dosegli vse zadane cilje. Zasnovali in izvedli smo spletno aplikacijo za vizualizacijo 3D modelov na večjem številu odjemalcev sočasno in možnostjo interakcije s prikazanimi objekti. Uporabnik lahko preko vmesnika iz posredovanih označevalnikov zgradi modele. Predlagali smo algoritem za samodejno zaznavo modela na podlagi medsebojne strukture posameznih označevalnikov. Najpomembnejši cilj in doprinos dela je izvedena uporabniška študija, s katero smo želeli pokazati, da lahko uporabnik izvaja svoje naloge hitreje z ozko usmerjeno aplikacijo kot pa z aplikacijo za splošno rabo.

Pri zasnovi in izvedbi aplikacije nismo imeli večjih težav. Izvedba je zaradi dobre dokumentacije uporabljenih tehnologij in velikega obsega dodatne literature potekala po zadanem načrtu. Pri izvedbi 3D vmesnika v spletni brskalnik smo ugotovili, da kljub hitremu napredku tehnologije veliko uporabnikov interneta ne uporablja starejše verzije brskalnikov, katere še ne podpirajo WebGl-a v celoti. Pri teh uporabnikih je bila obremenitev procesorja zelo visoka tudi pri enostavnih animacijah.

S predlaganim algoritmom za samodejno zaznavo modela iz strukture označevalnikov smo poskušali izboljšati delovanje modelov AIM, vgrajenih v aplikacijo QTM, in ponuditi alternativo algoritmu podjetja Qualisys, čigar delovanje je poslovna skrivnost. Algoritem smo v primerjavi z modeli AIM

izboljšali v količini shranjenih podatkov (zajetih gibanj), ki jih algoritem potrebuje za prepoznavo modela. QTM namreč prepozna samo modele, katerih označevalniki se gibljejo v relativnih območjih že shranjenih gibanj. Predlagan algoritem pa označevalnike predstavi v obliki neusmerjenega grafa in poišče najustreznejše povezave glede na shranjen model (graf).

Z uporabniško študijo smo ugotovili, da je uporabnik za dokončanje nalog s predlagano aplikacijo porabil enako ali manj časa kot z apikacijo QTM. Enako časa so uporabniki potrebovali za sestavo označevalnikov v modele. Takšnega rezultata nismo pričakovali, saj smo v aplikacijo vgradili orodja za hitro dokončanje sestave modela, vendar ta orodja niso bila dovolj izpostavljena in jih uporabniki niso našli in uporabili. Pri ostali nalogah so uporabniki predlagane aplikacije porabili manj časa kot uporabniki aplikacije QTM, saj je predlagana aplikacija zasnovana prav z namenom upravljanja z modeli in izvajanjem analiz.

V uporabniški študiji pa ni bil poglaviten element samo čas, temveč tudi vtis, ki ga je na uporabnike naredila aplikacija. Predlagano aplikacijo so uporabniki na podlagi rezultatov UEQ testa ocenili kot privlačnejšo, uporabniški vmesnik pa kot bolj razumljiv in pregleden, interakcija z njo se jim je zdela bolj zanesljiva. Takšen rezultat odraža čist, intuitiven uporabniški vmesnik, ki se prilagaja trenutnemu početju uporabnika v aplikaciji. Slabši rezultat v kategoriji novost bi pripisali dejstvu, da se je večina uporabnikov prvič srečala s aplikacijami za zajem in analizo gibanja in tudi zasnovi uporabniške študije, saj uporabniki predlagane in aplikacije QTM niso primerjali med seboj.

Prepričani smo, da je predlagana aplikacija primer moderne spletne aplikacije, ki se lahko uporablja samostojno za potrebe e-šolstva in raziskav z uporabo zajema gibanja, hkrati pa je to razširljiva platforma za še bolj specifične namene uporabe. Slednje se že izkazalo v sodelovanju pri projektu Biofeedback Laboratorija za informacijske tehnologije, Fakultete za elektrotehniko Univerze v Ljubljani, kjer predlagana aplikacija deluje kot osnova za vizualno analizo golf udarca.

Literatura

[1] A. G. Kirk, J. F. O'Brien, D. A. Forsyth. "Skeletal Parameter Estimation from Optical Motion Capture Data", v zborniku IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, str. 782-788

[2] P. A. Fayolle, B. Schmitt, A Pasko. "Web-based constructive shape modeling using real distance functions.", IEICE TRANSACTIONS on Information and Systems 88, no. 5, 2005, str. 828-835.

[3] R. Cartwright, V. Adzhiev, A. Pasko, Y. Goto, T. L. Kunii. "Web-based shape modeling with HyperFun.", Computer graphics and applications, IEEE 25, no. 2, 2005, str. 60-69.

[4] Q. Liu, A. Sourin. "Function-defined shape metamorphoses in visual cyberworlds.", The Visual Computer 22, no. 12, 2006, str. 977-990.

[5] J. Behr, P. Eschler, Y. Jung, M. Zöllner. "X3DOM: a DOM-based HTML5/X3D integration model.", v zborniku 14th International Conference on 3D Web Technology, ACM, 2009, str. 127-135.

[6] J. Congote, A. Segura, L. Kabongo, A. Moreno, J. Posada, O. Ruiz. "Interactive visualization of volumetric data with webgl in real-time.", v zborniku k the 16th International Conference on 3D Web Technology, ACM, 2011, str. 137-146.

[7] C. Leung, A. Salga. "Enabling webgl.", v zborniku the 19th international conference on World wide web, ACM, 2010, str. 1369-1370.

[8] J. Canemaker. “Winsor McCay - His Life and Art”, Abbeville, 1987.

[9] T. Igarashi, T. Moscovich, J. F. Hughes. “Spatial Keyframing for Performance-driven Animation”, ACM SIGGRAPH / Eurographics Symposium on Computer Animation, 2005.

[10] P. Nogueira. “Motion Capture Fundamentals: A Critical and Comparative Analysis on Real-World Applications”, v zborniku: 4th International Conference on Information Society and Technology, 2011.

[11] L. Naimark, E. Foxlin. “Circular data matrix fiducial systems and robust image processing for a wearable vision-inertial self-tracker.” v zborniku International Symposium on Mixed and Augmented Reality, IEEE Computer Society, 2002, str. 27.

[12] R. Hartley, A. Zisserman. “Multiple view geometry in computer vision”, Cambridge university press, 2003.

[13] J.F.Wang, R. Azuma, G. Bishop, V. Chi, J. Eyles, H. Fuchs. “Tracking a head-mounted display in a room-sized environment with head-mounted cameras”, v zborniku Orlando’90, International Society for Optics and Photonics, 1990, str. 47-57.

[14] S. Tisa, F. Zappa, S. Cova. ”Monolithic quad-cells for single-photon timing and tracking.”, v zborniku International Congress on Optics and Optoelectronics, 2007, str. 658305-658305.

[15] D. Kim, S. Richards, T. Caudell. “An optical tracker for augmented reality and wearable computers”, v zborniku IEEE VRAIS, IEEE Computer Society Press. 1997, str. 146–151.

[16] A. Krašček, J. Sodnik. “Qualisys web tracker – a web-based visualization tool for real-time data of an optical tracking system”, Vol. 1, 2014, str. 155-159.

[17] I. E. Sutherland. “A head-mounted three dimensional display”, v zborniku fall joint computer conference I, 1968, str. 757-764.

[18] G. Welch, E. Foxlin. “Motion tracking survey”, IEEE Computer graphics and Applications, 2002, str. 24-38.

[19] S. Yabukami, H. Kikuchi, M. Yamaguchi, K. I. Arai, K. Takahashi, A. Itagaki, N. Wako. “Motion capture system of magnetic markers using three-axial magnetic field sensor”, v zborniku: IEEE Transactions on Magnetics, Vol 36, Issue 5, 2013, str. 3646-3648.

[20] A. Sharma, M. Agarwal, A. Sharma, P. Dhuria. “Motion capture process, techniques and applications”, v zborniku: International Journal on Recent and Innovation Trends in Computing and Communication, Vol 1, Issue 4, 2000, str. 251-257.

[21] T. Cloete, C. Scheffer. “Benchmarking of a full-body inertial motion capture system for clinical gait analysis”, v zborniku: 30th Annual International IEEE EMBS Conference, 2008, str. 4579-4582

[22] J. G. Richards. “The measurement of human motion: A comparison of commercially available systems´´, Human Movement Science, 1999, Vol. 18, str. 589-602.

[23] Z. Zhang. “Flexible camera calibration by viewing a plane from unknown orientations.”v zborniku: 7. IEEE International Conference on Computer Vision, vol. 1, IEEE, 1999, str. 666-673.

[24] B. Triggs, P. F. McLauchlan, R. I. Hartley, A. W. Fitzgibbon. “Bundle adjustment—a modern synthesis.”, Vision algorithms: theory and practice, Springer Berlin Heidelberg, 2000, str. 298-372.

[25] A. Krašček, K. Stojmenova, S. Tomažič, J. Sodnik. “Web Based E-learning Tool for Visualization and Analysis of 3D Motion Capture Data”, v zborniku: 7. International Conference on Information, Process and Knowledge Management, 2015.

[26] L. Nilsson. “QTM Real-time Server Protocol Documentation, V1.9”, 2011.

[27] M. Berlander. “Qualisys low latency real-time system”, 2011.

[28] The WebSocket API, dostopno na: http://www.w3.org/TR/websockets/, 4.2.2015

[29] WebGL - OpenGL ES 2.0 for the Web, dostopno na: http://www.khronos.org/webgl/, 8.2.2015

[30] T. Parisi. “WebGL: up and running. ”, O’Reilly Media, Inc.”, 2012.

[31] Node.js, Dostopno na: http://nodejs.org/, 11.2.2015

[32] M. Hassenzahl. “The effect of perceived hedonic quality on product appealingness”, Internatinal Journal of Human-Computer Interaction, 2001, str. 481-499.

[33] J. Preece, Y. Rogers, H. Sharp. “Interaction design: Beyond human-computer interaction”, Wiley, New York , 2002.

[34] M. Kurosu, K. Kashimura. “Apparent usability vs. inherent usability: experimental analysis of the determinants of the apparent usability”, v zborniku: Conference Companion of human factors in computing systems, 1995, str. 292-293.

[35] S. Kirkpatrick. “Optimization by simulated annealing: Quantitative studies.”, Journal of statistical physics 34, no. 5-6, 1984. str. 975-986.

[36] V. Černý. “Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm”, Journal of Optimization Theory and Applications 45, 1985, str. 41-45.

[37] “Qaulisys Track Manager”, 2011.

[38] B. Laugwitz, T. Held, M. Schrepp. “Construction and evaluation of a user experience questionnaire.”, Springer Berlin Heidelberg, 2008.