Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Andrej Černivec

Brezdotično upravljanje aplikacij s senzorjem Kinect

diplomsko delo

univerzitetni strokovni študijski program prve stopnje računalništvo in informatika

Mentor: doc. dr. Matija Marolt

Ljubljana 2014

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

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

Št. naloge: 00102 / 2013
Datum: 12.4.2013

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

Kandidat: **ANDREJ ČERNIVEC**

Naslov: **BREZDOTIČNO UPRAVLJANJE APLIKACIJ S SENZORJEM KINECT**
TOUCHLESS APPLICATION CONTROL WITH KINECT SENSOR

Vrsta naloge: Diplomsko delo univerzitetnega študija prve stopnje

Tematika naloge:

V okviru diplomske z naloge izdelajte rešitev, ki omogoča brezdotično upravljanje aplikacij s pomočjo senzorja Kinect. Preučite trenutno stanje na tem področju in zasnujte rešitev za upravljanje poljubnih aplikacij s prilagoditvami za uporabo v predstavitvenih kioskih. Rešitev naj podpira več ukazov, ki se podajajo s kombinacijo leve in desne roke, omogoča pa naj tudi zaznavanje menjave uporabnikov. Preizkusite jo z različnimi uporabniki v različnih okoljih in prikazovalnih napravah.

Mentor:

doc. dr. Matija Marolt

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Andrej Černivec, z vpisno številko **63070060**, sem avtor diplomskega dela z naslovom:

Brezdotično upravljanje aplikacij s senzorjem Kinect

S svojim podpisom zagotavljam, da:

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

V Ljubljani, dne 11. februarja 2014 Podpis avtorja:

Kazalo

Povzetek

Cilj te diplomske naloge je bil razviti aplikacijo, preko katere bi lahko z uporabo senzorja za detekcijo gibanja upravljali poljubno aplikacijo na operacijskem sistem Windows. Kot senzor za zajem gibanja je bil uporabljen Microsoft Kinect. Za razvoj je bil uporabljen programski jezik C#, ogrodje Microsoft .NET, za komunikacijo s senzorjem Kinect je bila uporabljena knjižnica Kinect for Windows SDK, za implementacijo dodatnih funkcij pa še knjižnici Coding4fun Kinect Toolkit in InputSimulator. Za razvojno okolje sem uporabil Microsoft Visual Studio 2012. Aplikacija omogoča brezdotično upravljanje aplikacij preko več ukazov, ki se podajajo s kombinacijo leve in desne roke. Možno je izbiranje in klikanje, povečevanje ali pomanjševanje vsebine ter premikanje po vsebini. Aplikacija tudi zazna, ko se uporabnik zamenja, ter sproži zagon druge t. i. uvodne aplikacije.

Abstract

The aim of the thesis was to develop an application that would allow us to control any application on Windows operating system using a motion capture sensor. Sensor used for motion capture was Microsoft Kinect. Programing language used for developing was C#, the framework used was Microsoft .NET and the library used to communicate with the Kinect sensor was Kinect for Windows SDK. Libraries Coding4fun Kinect Toolkit and InputSimulator were used for the implementation of some additional functions. The tool used to develop the application was Microsoft Visual Studio 2012. Application enables touchless controlling of other applications through multiple commands, which are achieved using the combination of left and right hand. Possible commands are choose and click, zoom in and zoom out and scroll in all directions. Application also detects when the user switches and runs a secondary so called intro application.

Poglavje 1

Uvod

Skozi zgodovino se je način kontroliranja računalniških sistemov oziroma način komuniciranja uporabnika z računalnikom vseskozi spreminjal. Sprva je bilo vnašanje podatkov omejeno le na tipkovnico, nato je vse močno poenostavila miška. Za uporabo miške se je seveda moral spremeniti tudi način podajanja ukazov, za kar je poskrbel razvoj grafičnega vmesnika, ki je osnova še danes. Za pojavom miške, ki je še danes na prvem mestu, so se začele pojavljati tudi ostale tehnologije. Prva, ki ji je uspelo narediti preboj in se v zadnjem času množično uporablja na vseh vrstah naprav, je zaslon na dotik. Zaslon na dotik je s svojo prilagodljivostjo nadomestil tako tipkovnico kot miško, ponovno pa je bilo potrebno za uspešno uporabo malo prilagoditi grafični vmesnik.

Čeprav so ekrani na dotik doživeli uspeh, pa niso primerni za vse situacije, saj mora biti uporabik ves čas fizično prisoten ob ekranu. V zadnjih letih se tako razvija vse več tehnologij, ki omogočajo brezdotično upravljanje računalnika. Uporabnik lahko torej podaja ukaze zgolj s premikanjem svojega telesa, brez dotikanja kakršne koli naprave. Sprejemanje ukazov preko gibanja telesa je ponavadi omejeno le na gibanje dlani, lahko pa se preko vnaprej določenih gest poljubno uporablja tudi ostale dele telesa. Način podajanja ukazov, brez da bi se uporabnik dotikal naprave, se izkaže še posebej

dobro pri velikih ekranih, saj lahko uporabnik sistem upravlja z razdalje.

Fokus te diplomske naloge je uporaba senzorja Microsoft Kinect za izdelavo aplikacije, ki bo omogočala uporabo senzorja za nadzor računalniškega sistema. Ukazi se podajajo s premikom rok in vnaprej določenimi gestami, ki so za lažjo uporabo in hitro učenje uporabnikov čim bolj preproste. Ker je vsakemu sistemu za lažjo uporabo pametno prilagoditi tudi grafični vmesnik, se bo aplikacija uporabljala v kombinaciji z t. i. kiosk aplikacijo, ki uporabniku v obliki, prilagojeni za večje ekrane, prikazuje želeno vsebino.

V prvem delu diplomske naloge so na kratko opisane tehnologije in orodja, uporabljena pri razvoju aplikacije. Podan je pregled tehnologij s področja naravnega uporabniškega vmesnika in obstoječe rešitve za upravljanje računalnika preko senzorja Kinect. V drugem delu diplomske naloge je predstavljen sam razvoj aplikacije ter njene funkcionalnosti. Predstavljeni so rezultati testiranja aplikacije v praksi, na koncu pa so podane še možnosti za nadaljno izboljšavo aplikacije.

Poglavje 2

Tehnologije in orodja

Za razvoj aplikacije za brezdotično upravljanje računalnika na operacijskem sistemu Windows so bile uporabljene sledeče tehnologije in orodja:

- Kinect
- Microsoft Visual Studio
- C#
- Microsoft .NET
- Microsoft Kinect SDK
- Microsoft Kinect Toolkit
- Coding4fun Kinect Toolkit
- InputSimulator

2.1 Senzor Kinect

Kinect senzor je senzor gibanja in zvoka, ki ga je razvil Microsoft [23]. Ob izdaji prve verzije je bil njegov osnovni namen dopolniti izkušnjo Microsoftove uspešne igralne konzole Xbox 360. S pomočjo senzorja je mogoče igrati igre

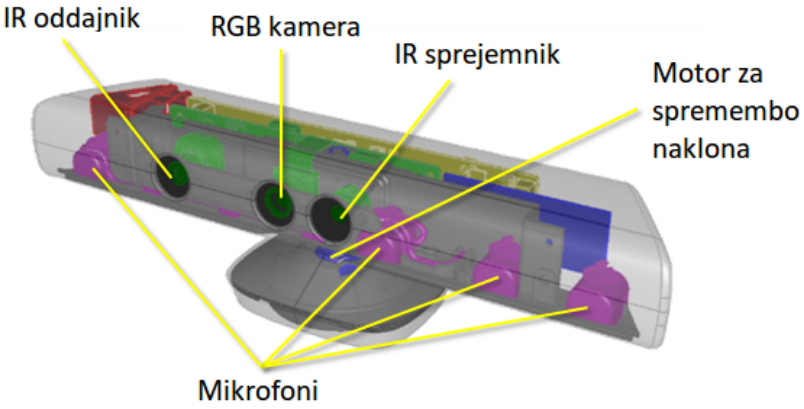
brez potrebe po dodatnem kontrolerju, saj je mogoče ukaze podajati kar z gibanjem telesa, izvajanjem gest in s pomočjo glasovnih ukazov. Rešitve na področju zajema gibanja so razvila tudi konkurenčna podjetja, npr. Nintendo s svojim Wii Remote za konzolo Wii in pa Sony s Playstation Move za svojo konzolo Playstation 3, vendar se vse tehnologije razlikujejo od samega senzorja Kinect.

Do sedaj so bile izdane tri verzije senzorja Kinect. Prva, prikazana na sliki 2.2, je bila uporabljena za potrebe te diplomske naloge. Obstajata še prilagojena verzija prve različice, imenovana Kinect for Windows, ki omogoča zajem gibanja pri manjših razdaljah. Tretja različica senzorja Kinect je izšla skupaj z novo različico igralne konzole Xbox, imela pa naj bi prenovljeno in zmogljivejšo strojno opremo.

Kinect je sestavljen iz sledečih komponent [14]:

- RGB kamera
- IR oddajnik
- IR sprejemnik
- Mikrofoni
- Motor za spremembo naklona

Kot je prikazano na sliki 2.1, ima Kinect v sebi več kot en senzor. Za zaznavanje globinske (3D) slike uporablja infrardeči laser (IR oddajnik) v kombinaciji z monokromatskim CMOS senzorjem (IR sprejemnik), kar mu omogoča zaznavo globinske slike v vseh svetlobnih pogojih. Druga je RGB kamera, ki je zmožna najvišje resolucije 1280x1024 pikslov. Poleg senzorjev za vizualno zaznavanje pa ima Kinect vgrajene še štiri mikrofone, ki služijo pri prepoznavi glasovnih ukazov in uspešno izločajo ambientni zvok. S kombinacijo vseh senzorjev je Kinect sposoben 3D zajema in sledenja celemu telesu, prepoznavanja obrazov in prepoznavanja glasovnih ukazov.



Slika 2.1: Arhitektura senzorja Kinect [14]

Slika 2.2: Senzor Kinect [23]

2.2 Visual Studio 2012

Microsoft Visual Studio je orodje za razvoj programske opreme različnih vrst, npr. igre za konzole, grafične vmesnike, windows forme, WPF aplikacije, spletne strani in aplikacije ter podobno [24]. Razvoj aplikacij je usmerjen predvsem za Microsoft okolja, kot so Windows, Windows Mobile oz. Windows Phone ter igralna konzola Xbox.

Slika 2.3: Visual Studio [24]

Orodje je narejeno na način, da v osnovi ni posebej prilagojeno za noben jezik, temveč se funkcionalnosti dodajo preko storitev oziroma paketov. Nekaj paketov je vključenih že ob sami namestitvi, in sicer paketi za programske jezike C/C++, Visual Basic .NET, C# in F#. Poleg tega se lahko naknadno doda še podpora za druge jezike, npr. Python ali Ruby. V podporo posameznega jezika so vključena številna pomagala, kot so samodejno obarvanje kode, samodejno predlaganje ukazov in sprotno izvajanje kode v ozadju. Slednje omogoča iskanje napak v realnem času. Poleg dobrega urejevalnika kode pa je v Visual Studio vključenih še več stvari, ki so potrebne za razvoj programske opreme. Omogoča namreč vizualno razvijanje grafičnega vmesnika, vključuje pa tudi dober razhroščevalnik, ki je ključnega pomena pri testiranju programske opreme. Zaradi širokega spektra funkcij in velike razširljivosti je to zelo priljubljeno orodje med razvijalci programske opreme in je bilo uporabljeno tudi pri razvoju aplikacije v diplomski nalogi. Uporabljena je bila verzija 2012, zgodovina različic pa je navedena v tabeli 2.1.

2.3 C#

C# je objektno orientiran programski jezik, ki ga je razvilo podjetje Microsoft v sklopu njihove .NET iniciative za čim bolj množično uporabo njihovega ogrodja .NET. Ogrodje .NET je bolj podrobno opisano v poglavju 2.4. V osnovi je programski jezik C#, ki ga je s svojo razvojno ekipo razvil Anders Hejlsberg, mišljen kot preprost, modern, vsestranski objektno usmerjen pro-

Ime	Datum izdaje
`Visual Studio`	april 1995
`Visual Studio 97`	februar 1997
`Visual Studio 6.0`	junij 1998
`Visual Studio .NET`	februar 2002
`Visual Studio .NET 2003`	april 2003
`Visual Studio 2005`	november 2005
`Visual Studio 2008`	november 2007
`Visual Studio 2010`	april 2010
`Visual Studio 2012`	september 2012
`Visual Studio 2013`	oktober 2013

Tabela 2.1: Pregled verzij orodja Visual Studio

gramski jezik [17]. Jezik omogoča hitro razvijanje robustnih in vzdržljivih aplikacij. Ker je jezik objektno orientiran, ga lahko vsak programer s predznanjem katerega od drugih jezikov, kot so C, C++ ali Java zelo hitro obvlada. Da je jezik neka nadgradnja prejšnjih jezikov nam, čeprav na prvi pogled to mogoče ni vidno, sporoča že ime. Pri jeziku C++ nam dva plusa sporočata, da gre iteracijo oziroma novo, nadgrajeno različico programskega jezika C [22]. V primeru C# naj bi se v znaku # skrivali štirje plusi (dva v prvi in dva v drugi vrsti), kar naj bi prav tako pomenilo novo iteracijo oziroma verzijo programskega jezika C++. Ime jezika je bilo v razvojni fazi celo COOL, kar je bila kratica za "C-like Object Oriented Language", torej v prevodu objekto orientiran jezik podoben C-ju.

Čeprav jezik ni bil mišljen kot konkurenca nizkonivojskim jezikom, kot sta C ali zbirni jezik, pa naj bi bila ena od prednosti njegova hitrost. Prav tako pa omogoča veliko možnosti, ki jih ponavadi najdemo pri nizkonivojskih jezikih, npr. direktna kontrola pomnilnika, saj so se razvijalci zavedali, da je v nekaterih primerih to ključnega pomena. Ob pisanju te diplomske naloge

Verzija	Datum izdaje	.NET ogrodje	Visual Studio
`C# 1.0`	Januar 2002	.NET ogrodje 1.0	Visual Studio .NET 2002
`C# 1.2`	April 2003	.NET ogrodje 1.1	Visual Studio .NET 2003
`C# 2.0`	November 2005	.NET ogrodje 2.0	Visual Studio 2005
`C# 3.0`	November 2007	NET ogrodje 2.0 NET ogrodje 3.0 NET ogrodje 3.5	Visual Studio 2008 Visual Studio 2010
`C# 4.0`	April 2010	.NET ogrodje 4.0	Visual Studio 2010
`C# 5.0`	Avgust 2012	.NET ogrodje 4.5	Visual Studio 2012

Tabela 2.2: Razvoj programskega jezika C#

je aktualna verzija C# 5.0, razvoj verzij pa je razviden v tabeli 2.2.

2.4 Microsoft .NET

Microsoft .NET je ogrodje, ki je namenjeno predvsem Microsoft Windows operacijskim sistemom. Obstaja tudi odprtokodna implementacija ogrodja .NET, imenovana Mono [16], ki se lahko uporablja tudi na drugih operacijskih sistemih. Ogrodje je v osnovi sestavljeno iz dveh delov. Prvi del je velika knjižnica, ki omogoča hitro programiranje in kreiranje aplikacij. Knjižnica vsebuje vse za pomoč pri razvoju uporabniškega vmesnika, dostopanja do podatkov, povezovanja z bazami, kriptografije, razvoja spletnih aplikacij, različnih algoritmov in mrežnih povezav [21]. Ogrodje se lahko poljubno kombinira z drugimi knjižnicami in seveda kodo programerja. Drugi del pa CLR (Common Language Runtime), ki skrbi za storitve, kot so varnost, upravljanje s pomnilnikom in delo z izjemami. Programi, ki uporabljajo ogrodje .NET se zaganjajo v programskem, ne v strojnem okolju.

Pretekle verzije so prikazane v tabeli 2.3. Z verzijami se je spreminjala tudi sama zgradba ogrodja, oziroma so mu bile dodane nove komponente, kar je

Verzija	Datum izdaje	Vključeno v
1.0	13.2.2002	n/a
1.1	24.4.2003	Windows Server 2003
2.0	7.11.2005	Windows Server 2003 R2
3.0	6.11.2006	Windows Vista Windows Server 2008
3.5	19.11.2007	Windows 7 Windows Server 2008 R2
4.0	12.4.20010	n/a
4.5	15.8.2012	Windows 8 Windows Server 2012
4.5.1	17.10.2013	Windows 8.1 Windows Server 2012 R2

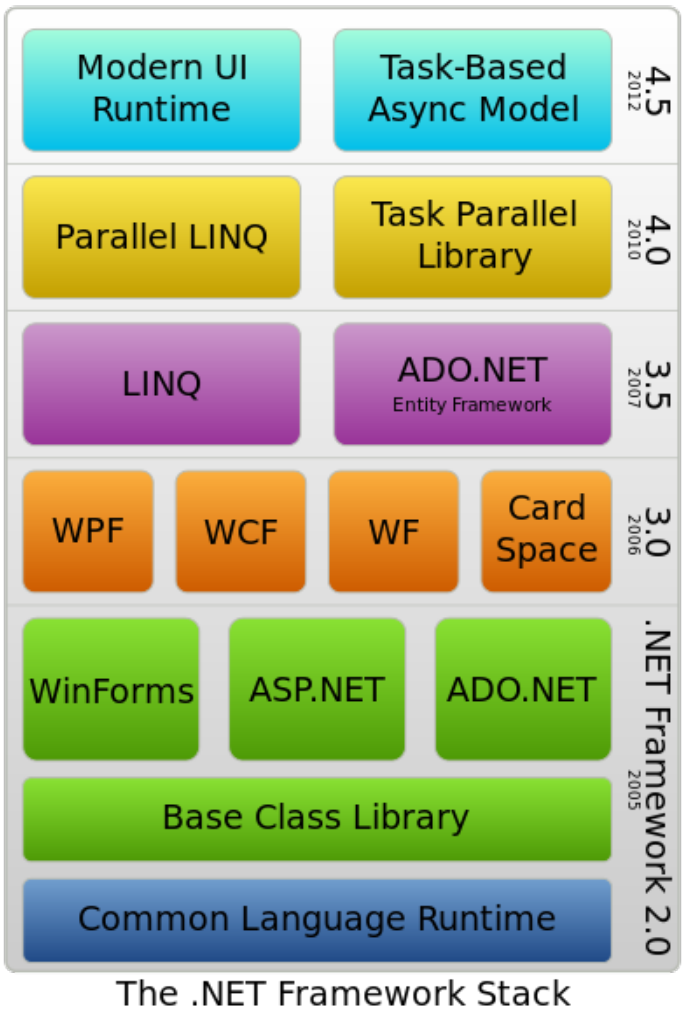
Tabela 2.3: Razvoj Microsoft .NET

vidno na sliki 2.4.

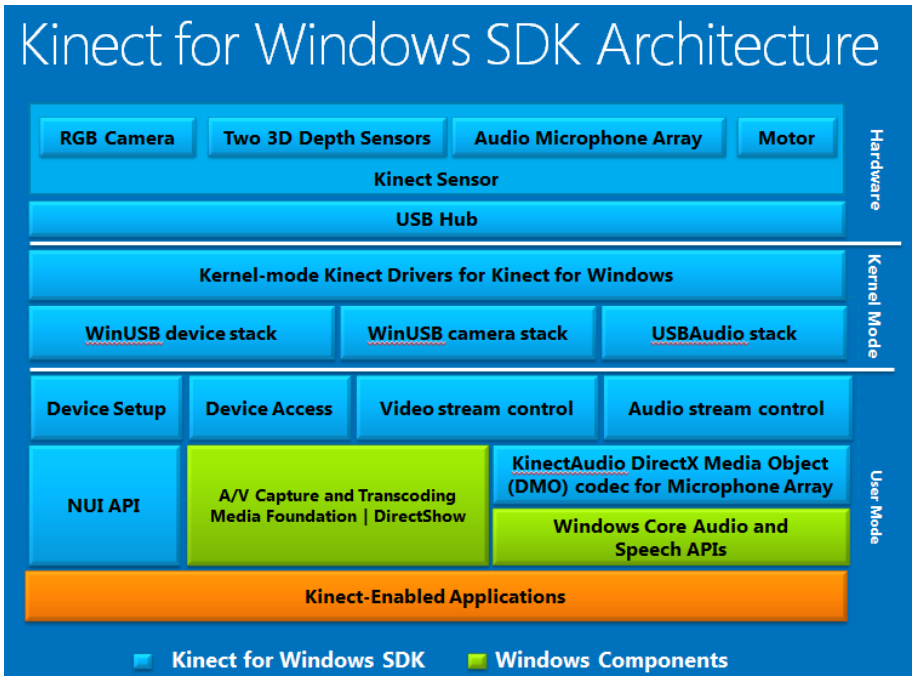
2.5 Kinect for Windows SDK 1.8

Kinect for Windows SDK (prej znan kot Microsoft Kinect SDK) je uradni paket, izdan s strani Microsofta, za razvoj aplikacij s senzorjem Kinect [13] na operacijskem sistemu Windows. Izdan je bil leta 2011, vanj pa je vključeno vse potrebno za hiter razvoj aplikacij. Kinect for Windows SDK vsebuje vse potrebne gonilnike, obširno knjižnico in jasne primere, s pomočjo katerih programerji lahko začnejo razvijati aplikacije zelo hitro. Od februarja 2012 obstaja tudi komercialna verzija paketa, ki podjetjem omogoča prodajo svojih razvitih izdelkov. Podrobnejša arhitektura paketa Microsoft Kinect SDK je prikazana na sliki 2.5.

Poleg uradnega ogrodja Kinect for Windows pa obstajajo tudi druge od-



Slika 2.4: Zgradba ogrodja .NET glede na verzijo [21]



Slika 2.5: Zgradba paketa Microsoft Kinect SDK

prtokodne rešitve, ki za razliko od paketa Kinect for Windows delujejo tudi na drugih operacijskih sistemih. Primera takih rešitev sta OpenNI ali pa OpenKinect.

2.6 Kinect for Windows Developer Toolkit 1.8

Kinect for Windows Developer Toolkit je zbirka vnaprej pripravljenih programov, ki služijo učenju in lažjemu ter hitrejšemu razvijanju aplikacija za uporabo s senzorjem Kinect [12]. Vsak program v zbirki prikazuje eno izmed funkcij senzorja Kinect, s pomočjo priložene kode pa se je možno naučiti pravilne implementacije različnih funkcij v lastni aplikaciji. Deli kode v zbirki so podani na tak način, da jih lahko takoj uporabimo v lastnih aplikacijah in s tem pospešimo razvoj, hkrati pa poskrbimo za pravilno implementacijo. Z

dodajanjem novih možnosti senzorju Kinect se povečuje in posodablja tudi zbirka Kinect for Windows Developer Toolkit. Za razvoj naše aplikacije je bila uporabljena verzija 1.8, ki se ujema s knjižnico Kinect for Windows SDK verzije 1.8.

2.7 Coding4fun Kinect Toolkit

Coding4fun je spletna stran, namenjena spremljanju novih in zanimivh projektov na področju strojne in programske opreme [7]. Namenjena je osebam, ki se s tem področjem ne ukvarjajo le službeno, temveč tudi v svojem prostem času. Poleg spremljanja zanimivih projektov pa spletna stran Coding4fun tudi sama ustvarja projekte na različnih področjih.

Eno izmed teh področij je tudi razvoj programske opreme za senzor Microsoft Kinect. Pri Coding4fun so napisali knjižnico, ki skupaj z uporabo Microsoftovega SDK omogoča še hitrejši razvoj aplikacij. Velik del ponavljajočih se nalog je mogoče realizirati s preprostimi ukazi. Ker Coding4fun spada pod Microsoftovo okrilje, knjižnico Coding4fun pri programiranju uporablja tudi Microsoft. Svetujejo in uporabljajo jo celo v njihovih spletnih video tečajih.

2.8 InputSimulator

InputSimulator je majhna odprtokodna knjižnica [6], napisana v programskem jeziku C#, ki ima le eno nalogo. Ta naloga je programsko simuliranje pritiska posamezne tipke na tipkovnici ali miški. Njen namen je s preprostimi ukazi realizirati uporabo *Interop* metode, ki pri programskem jeziku C# služi za podajanje nizkonivojskih ukazov operacijskemu sistemu Windows.

Poglavje 3

Brezdotično upravljanje in NUI

Brezdotično upravljanje računalnikov je koncept, ki je v računalništvu prisoten že dolga leta. Povezuje se ga predvsem s pojmom NUI (natural user interface), kar pomeni vmesnik, ki je uporabniku naraven. Pri uveljavljenih metodah podajanja ukazov preko tipkovnice, miške ali katere druge naprave, je potrebno uporabnika vedno najprej naučiti, kako se napravo uporablja. Potrebno mu je podati možne ukaze in mu pokazati, kaj ti ukazi pomenijo. Smisel naravnega uporabniškega vmesnika pa je, da simulira našo interakcijo z zunanjim svetom v vsakdanjem življenju in je tako z vidika uporabnika bolj logičen in preprost. Temu se da s pomočjo brezdotičnega upravljanja še najbolj približati. Vzemimo za primer, da želi uporabnik kocko, ki jo vidi na ekranu, premakniti desno. Z uporabo tipkovnice bi moral napisati nek ukaz, z uporabo miške bi moral biti seznanjen s konceptom kazalca in miškinih tipk, pri brezdotičnem upravljanju pa lahko preprosto z roko zamahne v desno, kot bi to storil v realnem svetu. Seveda je interakcija daleč od resničnega sveta, vendar se izkušnja z leti in novimi tehnologijami izboljšuje.

Velik preboj na tem področju je naredil Microsoft z izdajo svojega senzorja Kinect za igralne konzole. Pred tem se je tovrstna tehnologija uporabljala samo na specifičnih področjih, s senzorjem Kinect pa se je vse skupaj približalo navadnemu uporabniku. Kinect pri igrah dobro predstavlja koncept

naravne uporabniške izkušnje. Če želi uporabnik brcniti žogo, mora dejansko brcniti z nogo. Če želi skočiti, mora skočiti tudi v resničnem svetu. To seveda omogoča, da uporabnik brez kakršnega koli učenja upravlja sistem, s katerim se spoznava prvič. Nekaj primerov naravnih uporabniških vmesnikov, ki se trenutno že uporabljajo, je opisanih v poglavju 3.1.

3.1 Trenutne brezdotične tehnologije

Na tržišču je trenutno veliko število različnih rešitev, ki pa se razlikujejo predvsem po uporabljeni tehnologiji za sledenje gibom, natančnosti in pa sledenju različnim delom telesa. Spodaj je naštetih le nekaj primerov uporabe brezdotične tehnologije.

Kinect

Senzor za zaznavanje gibanja, ki za zaznavo uporablja kombinacijo IR kamere in VGA kamere. Sposoben je slediti celotnemu telesu, zaznavati globinsko razdaljo in geste. Bolj podrobno je opisan v poglavju 2.1.

Leap Motion

Senzor, ki s pomočjo IR LED tehnologije in kamere prepoznava gibanje prstov uporabnika [15]. Namesti se ga pred tipkovnico, roke pa morajo biti za uporabo pozicionirane nad senzorjem.

Elliptic Labs

Tehnologija uporablja ultrazvok za spremljanje gibov uporabnika in zaznavanje ukazov [10]. Idealna postavitev naj bi vsebovala 6 zvočnikov in 8 mikrofonov, vendar naj bi v osnovi delovala že z navadnimi zvočniki in mikrofonom.

Tobii EyeX

Senzor, ki se ga namesti na dno ekrana, s pomočjo IR tehnolgije spremlja gibanje oči in tako izniči uporabo miške [20]. Uporabnik s pogledom na ekran lahko upravlja s kurzorjem.

EyeSight

EyeSight tehnologija ne potrebuje posebnih senzorjev, saj deluje že z uporabo navadne 2D kamere, ki je vgrajena v skoraj vsak prenosnik ali pametni telefon [11]. Programska oprema zna iz videa prepoznati gibanje roke in ukaze, ki jih uporabnik z roko podaja.

3.2 Pregled obstoječih rešitev NUI s senzorjem Kinect

Na področju kontroliranja računalniških sistemov z uporabo senzorja Kinect se pojavlja veliko število rešitev. Med vsemi se največkrat pojavlja rešitev, ki za kontrolo računalnika simulira uporabo miške. Način podajanja ukazov je različen od aplikacije do aplikacije. Nekatere uporabljajo eno roko, nekatere dve, omogočajo vse od preprostega klika do prekapljanja strani v brskalniku. Raznoraznih rešitev na tem področju je trenutno veliko, v nadaljevanju je opisanih le nekaj najbolj znanih.

Kinect Magic Cursor

Kinect Magic Cursor je odprtokodna aplikacija, s pomočjo katere lahko upravljamo miško preko senzorja Kinect [8]. Aplikacijo je razvil in na svojem blogu objavil David Renton, ki je predavatelj predmeta Razvoj iger na fakulteti Reid Kerr. Z aplikacijo lahko z uporabo desne roke vodimo miškin kurzor, z uporabo leve roke pa izvedemo klik. Klik se izvede preko geste pritiskanja ali preko geste stiskanja roke v pest. Za razvoj aplikacije je bil uporabljen programski jezik C# in ogrodje XNA4, deluje pa na operacijskih sistemih Windows 7 in Windows 8.

Kinect Mouse Cursor

Kinect Mouse Cursor je odprtokodna aplikacija, ki demonstrira uporabo senzorja Kinect za nadzor miške [4]. Desno roka omogoča vodenje kurzorja, klik pa izvedemo z dvigom leve roke. Aplikacija omogoča

tudi način za levičarje, kjer se ukaza zamenjata. Koda je na voljo v programskih jezikih C# in Visual Basic, za razvoj pa je bil uporabljen Microsoft Kinect SDK (zdaj imenovan Kinect for Windows SDK).

Easy Kinect Mouse Controller for Windows

Easy Kinect Mouse Controller for Windows je bolj kot aplikacija mišljen kot primer, iz katerega se je mogoče naučiti uporabiti senzor Kinect za upravljanje miškinega kurzorja [9]. Temu primerno je avtor Walt Smith zraven podal tudi dokumentacijo, ki programerja po korakih vodi skozi kodo. Pri aplikaciji z desno roko premikamo miškin kurzor, če se za dalj časa ustavimo na enem mestu, pa se izvede klik. Hitrost miške in zamik klika sta nastavljiva. Aplikacija je razvita kot WPF aplikacija z uporabo jezika C# in ogrodja Kinect for Windows SDK. Delovanje aplikacije se lahko opiše v treh fazah:

- Aplikacija prestreže podatke o skeletonu, ki jih podaja Kinect in zazna desno roko.
- Izračuna pozicijo roke relativno na ekran računalnika.
- Preda pozicijo operacijskemu sistemu Windows preko Microsoft API za kontrolo miške.

Kinect Toolbox

Kinect Toolbox ni končna aplikacija, temveč skupek primerov kode, ki omogočajo hitrejši razvoj aplikacij [5]. Tukaj je omenjena, ker poleg zaznavanja gest, poz uporabnika, zaznavanja audio ukazov in še veliko drugih metod ponuja tudi sledeči metodi:

- *MouseController*: metoda omogoča, da preko senzorja Kinect upravljamo miškin kurzor.
- *MouseImpostor*: metoda zamenja obliko miškinega kurzorja.

Winect

Winect je razvilo podjetje Xiora studios in aplikacijo ponudilo kot brezplačen prenos [25]. Za razliko od večine ostalih aplikacij za kontrolo

miške preko senzorja Kinect pa Winect ne uporablja Microsoftovega SDK, temveč OpenNI. Glavna razlika je v tem, da lahko Winect prepozna prste na roki in s tem omogoča, da ukaze podajamo kar preko različnih položajev prstov. Omogoča ukaze:

- Levi miškin klik
- Desni miškin klik
- Dvojni klik
- Sredni miškin klik
- Stran naprej (brskalnik Chrome)
- Stran nazaj (brskalnik Chrome)
- Povečava (brskalnik Chrome)
- Pomanjšava (brskalnik Chrome)

Čeprav naštete rešitve omogočajo veliko ukazov, pa za namene te diplomske naloge niso bile primerne. Poleg osnovnih ukazov, kot je klik, jim v večini primerov manjkajo ukazi kot je npr. premikanje po vsebini ali pa povečevanje in pomanjševanje vsebine. Poleg vsega pa niti ena izmed aplikacij ni prilagojena za upravljanje kiosk aplikacije, kjer so pogoji zahtevnejši. Aplikacije nimajo naprednega izbiranja in "zaklepanja"delovanja na enega uporabnika, niso prilagojene upravljanju kiosk aplikacij in niso razvite na način, ki bi omogočal preprosto implementacijo funkcionalnosti za boljše delo v zahtevnejših okoljih.

Poglavje 4

Aplikacija za brezdotično upravljanje

4.1 Ideja

Na info točkah, konferencah, sejmih in podobnih dogodkih želijo podjetja podati informacije, predstaviti idejo ali produkt na čim bolj zanimiv način. Nova tehnologija jim je bila pri tem vedno v pomoč, saj je vedno ponujala nove možnosti za podajanje informacij. Kjub temu pa je bila komunikacija skoraj vedno enosmerna, saj so uporabniki dobili informacije, ki so bile ob danem času na voljo, ne pa informacije, ki so jih želeli in potrebovali.

Cilj te diplomske naloge je bilo razviti aplikacijo, ki bo omogočala uporabnikom, da na enostaven in naraven način podajajo ukaze, s katerimi pridejo do informacij, ki jih želijo. Ker je trenutno najbolj naraven uporabniški vmesnik brezdotično upravljanje, je bil uporabljen tak način podajanja ukazov. Osnovna postavitev naj bi vsebovala sledeče elemente:

Večji zaslon

Zaslon služi za prikazovanje grafičnega vmesnika, preko katerega uporabniki pridejo do želenega podatka, ki se prikaže na istem zaslonu.

Kinect senzor

Senzor je nameščen pod ali nad ekranom in služi za beleženje gibov uporabnika, ki se nato pretvorijo v ukaze.

Računalnik

Računalnik z operacijskim sistemom Windows, na katerega sta priklopljena ekran in senzor Kinect. Na računalniku je nameščena kiosk aplikacija, ki skrbi za grafični vmesnik in pa aplikacija za detekcijo ukazov, ki je bila razvita v sklopu te diplomske naloge.

4.2 Naloge aplikacije

Cilj aplikacije je preko senzorja Kinect zaznavati ukaze, podane s strani uporabnika in jih pretvoriti v obliko, ki bo razumljiva operacijskemu sistemu Windows. Ukazi, ki jih mora aplikacija prepoznati so:

- Klik
- Povečava vsebine na zaslonu
- Zmanjšanje vsebine na zaslonu
- Premikanje po vsebini (scroll)

Aplikacija se bo uporabljala na lokacijah, kjer se bo izmenjevalo več ljudi, zato mora ustrezati določenim pogojem. Aplikacija se mora "zakleniti"le na enega uporabnika ter ignorirati ostale, da ne pride do motenj pri podajanju ukazov. Ukazi morajo biti preprosti, da jih uporabniki lahko hitro osvojijo. Ker aplikacija ni prirejena za uporabo s točno določeno kiosk aplikacijo in ker ni cilj, da bi morala kiosk aplikacija na kakršen koli način posebej prilagajati svoje delovanje, mora biti zato razvita aplikacija za brezdotično upravljanje popolnoma neodvisna in kompatibilna z vsemi aplikacijami.

4.3 Prepoznavanje uporabnika

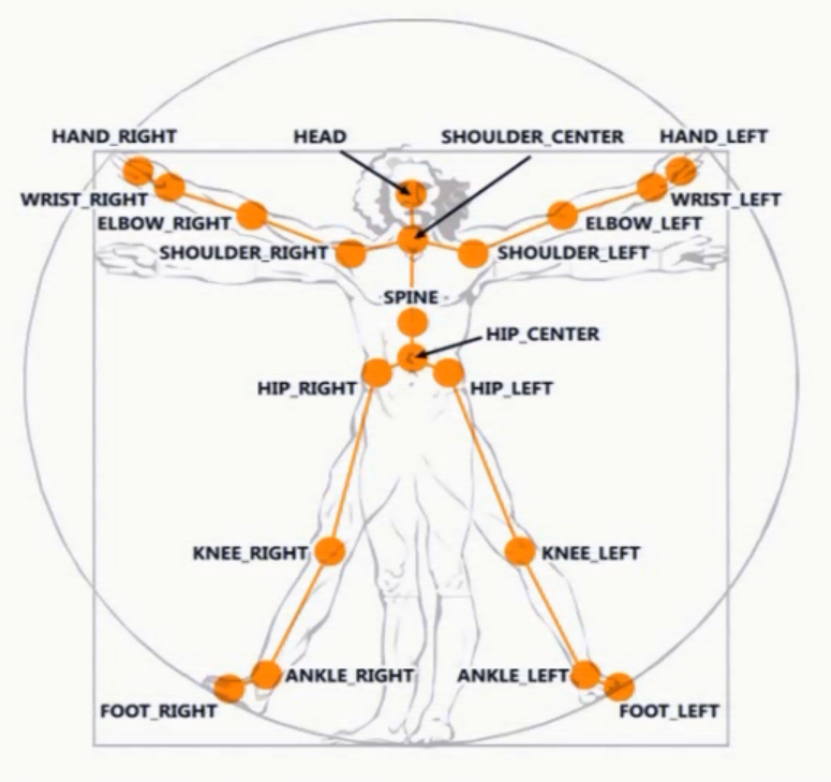
Senzor Kinect prve generacije lahko v svojem območju prepozna do 6 ljudi, spremljanje gibanja pa se lahko naenkrat izvaja na dveh osebah. Pri vsaki osebi se lahko sledi več posameznim točkam na telesu, ki so prikazane na sliki 4.1. Kinect spremlja gibanje točk na telesu pri prvih dveh osebah, ki jih najde v svojem polju zaznavanja, pri ostalih osebah pa zazna le prisotnost. Za prepoznavo osebe Kinect ne potrebuje celotnega telesa oziroma uspešnega sledenja vsem točkam telesa, dovolj je že, da iz oblike prepozna človeško telo.

Za delovanje aplikacije je pomembno, da Kinect prepozna in sledi samo eni osebi. Ta oseba mora biti tista, ki trenutno upravlja z aplikacijo. Vsako sledenje drugim osebam bi pomenilo napačno prepoznavanje ukazov, in s tem nedelovanje aplikacije. Ker je pričakovano, da bodo poleg osebe, ki z aplikacijo trenutno upravlja, v vidnem polju senzorja tudi druge osebe, ki bodo opazovale dogajanje, je bilo potrebno preprečiti nenamerno predajanje kontrole.

Vsaki osebi, ki pride v vidno polje, Kinect dodeli svojo unikatno ID številko. Aplikacija na podlagi te ID številke dodeljuje kontrolo. Kontrola je dodeljena prvi osebi, ki pride v vidno polje senzorja, in ne preide na drugo osebo, dokler je prva oseba aktivna. Ko se oseba odloči, da želi predati kontrolo, mora zapustiti vidno polje senzorja. Če je v vidnem polju že druga oseba, se kontrola takoj preda, v nasprotnem primeru pa aplikacija počaka, da se pojavi nova oseba.

4.4 Predstavitvena aplikacija

Vsakič, ko senzor Kinect prepozna novega uporabnika po postopku, opisanem v poglavju 4.3, se zažene predstavitvena aplikacija. Namen predstavitvene



Slika 4.1: Točke gibanja, ki jih spremlja Kinect [19]

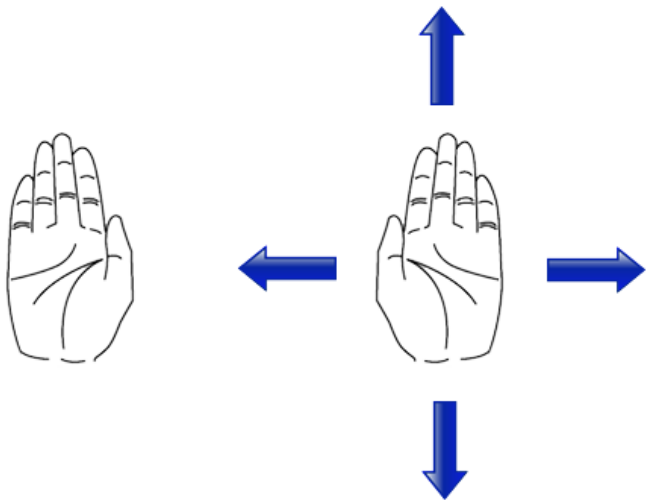
aplikacije je, da uporabnika seznani z naravnim vmesnikom. Na pozdravnem ekranu uporabniku ponudi možnost, da se na primeru nauči uporabljati katerega koli izmed možnih ukazov. Po končanem učenju ga odpelje na prvo stran kiosk aplikacije. Uporabniki, ki vmesnik že poznajo, lahko preskočijo učenje in takoj začnejo z brskanjem po kiosk aplikaciji.

4.5 Premikanje kurzorja in klik

Za upravljanje katere koli aplikacije sta dve glavni možnosti, ki morata biti na voljo. To sta izbira opcije in klik za potrditev opcije. Ker je bil en izmed ciljev aplikacije kompatibilnost z vsemi ostalimi aplikacijami na sistemu Windows, je bilo potrebno podajanje ukazov sistemu implementirati na način, ki bi ga razumele vse verzije sistema. Ker je glavni pripomoček za premikanje in izbiranje pri delu z računalnikom miška, se ukazi, podani preko senzorja Kinect translirajo v premik miške.

Pri uporabniku se vedno spremlja gibanje obeh rok. Spremlja se položaj leve in desne roke, podana kot koordinate v dvodimenzionalnem koordinatnem sistemu, ter stanje dlani. Stanje dlani je lahko ali odprto ali zaprto. Zaradi pogleda od spredaj se lahko odprta roka, ki je vodoravno usmerjena neposredno naprej proti senzorju Kinect, smatra kot zaprta roka, saj dlan in prsti s strani senzorja niso vidni.

Za premikanje kurzorja morata biti tako leva kot desna roka v odprtem položaju, kot je prikazano na sliki 4.2. Vsak premik desne roke se nato pretvori v premik kurzorja na ekranu. Ker je lahko površina na ekranu velika, bi moral uporabnik pri neposrednem pretvarjanju premika roke v premik kurzorja opravljati velike gibe, zato je bilo potrebno gibe uporabnika preslikati v drugačnem razmerju. Pri testiranju se je izkazalo, da je razmerje približno $3 : 1$ primeren kompromis med natančnostjo in udobno uporabo. Vsak uporabnikov premik se torej pomnoži s 3. Če bi se pomnožil večkrat, bi se s tem



Slika 4.2: Položaj rok za premikanje kurzorja

zmanjšala natančnost, saj senzor ne zaznava dobro zelo majhnih premikov in tako izbira opcij na ekranu postane zelo težavna.

Sam klik se izvede v dveh fazah. Ko uporabnik zapre dlan, kot je prikazano na sliki 4.3, se operacijskemu sistemu poda ukaz za pritisk levega miškinega gumba. Ko uporabnik dlan ponovno odpre, se operacijskemu sistemu poda ukaz za izpustitev levega miškinega gumba. Klik v dveh fazah omogoča dodatne akcije, ki sicer niso bistvene za potrebe te diplomske naloge, vendar so uporabne pri splošnem kontroliranju računalnika. Taki akciji sta npr. povleci in spusti (drag and drop) ali pa izbiranje vsebine, npr. teksta. Funkcija povleci in spusti omogoča tudi premikanje elementov na ekranu, kot so okna aplikacij ali ikon na namizju.

4.6 Premikanje po vsebini (scroll)

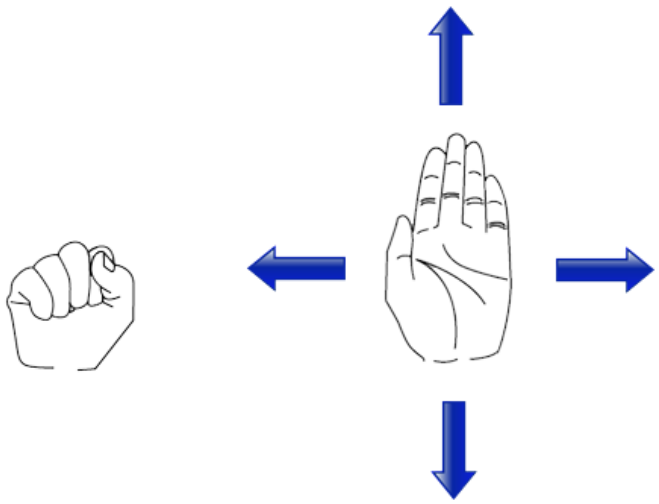
Ker se aplikacija lahko uporablja tudi za brskanje po vsebinah, ki so večje od dimenzij ekrana za prikazovanje, je bilo potrebno implementirati premikanje

Slika 4.3: Položaj rok za klik

po vsebini. Težavo bi lahko preprosto rešili na nivoju aplikacije za prikazovanje vsebine z drsniki, vendar je tako premikanje zamudno in težavno. Boljša rešitev premikanja je s pomočjo uporabe gest.

V tem primeru je pomikanje realizirano s pomočjo uporabe obeh rok. Kadar je leva roka v zaprtem stanju, desna pa v odprtem, aplikacija preide v način premikanja po vsebini. Položaj rok za premikanje po vsebini je prikazan na sliki 4.4. Premikanje deluje tako, da se ob trenutku spremembe stanja leve roke iz odprte v zaprto pozicija desne roke zamrzne in vzame kot referenčna točka. Desno roko nato premaknemo v želeno smer in s tem se začnemo premikati po vsebini. Premikanje po vsebini je odvisno od razdalje med trenutno pozicijo desne roke in referenčno točko, ki smo si jo zabeležili ob začetku premikanja. Torej, če je razdalja med pozicijo roke in referenčno točko večja, večja je hitrost premikanja po vsebini in obratno.

Ta gesta se operacijskemu sistemu poda kot klik sredinskega miškinega gumba oziroma koleščka ter premik miške. Premikanje deluje v vse smeri (gor, dol, levo, desno) in se obnaša enako, kot če bi se premikali z miško. Taka vrsta premikanja je standardna v kar nekaj Windows aplikacijah, predvsem



Slika 4.4: Položaj rok za premikanje po vsebini

pa deluje v vseh internetnih brskalnikih. Ko se stanje leve roke spremeni iz zaprtega v odprto, aplikacija izstopi iz načina premikanja.

4.7 Povečevanje in pomanjševanje vsebine

Za optimalno izkušnjo je pomembno, da uporabnik ogled vsebine prilagodi svojim potrebam. Najpomembnejša stvar pri prilagoditvi vmesnika je velikost elementov, ki jih uporabnik vidi na zaslonu. S povečevanjem ali zmanjševanjem elementov na zaslonu lahko uporabnik vpliva na količino vsebine, ki se prikazuje, spreminja izkušnjo brskanja po vsebini (večje elemente je lažje izbrati) in seveda vpliva na samo vidljivost vsebine. Prilagoditev je pomembna že zato, ker se sistem lahko upravlja z različnih razdalj in s tem spreminja vidljivost elementov na ekranu.

Povečevanje in pomanjševanje je v aplikaciji realizirano preko geste, ki uporablja obe roki in je znana pod imenom “pinch-to-zoom”. Položaj in smer rok je prikazana na sliki 4.5. Ideja geste je, da za povečavo stisnjeni roki

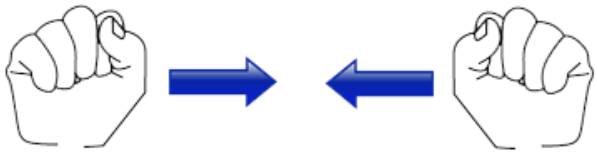
Slika 4.5: Gesta za povečavo elementov na zaslonu

potegnemo narazen, kot bi želeli neko stvar prijeti in raztegniti. Za uporabo geste morata biti obe dlani v odprtem položaju. Ko obe dlani naenkrat zapremo, si aplikacija zapomni začetni točki. Če zaprti dlani oziroma pesti razmaknemo narazen, vsebino na ekranu povečamo. Če pesti približamo, se vsebina na ekranu zmanjša. Ko katerokoli izmed dlani odpremo, se gesta konča. Aplikacija deluje v vsakem primeru, vendar je za preprostost uporabe priporočljivo, da za povečevanje vsebine dlani zapremo, ko sta blizu skupaj. Za zmanjševanje vsebine pa dlani zapremo, ko sta precej narazen.

Da se povečava ali pomanjšava vsebine izvede, mora biti razlika med razdaljo med začetnima točkama (ko zapremo dlani) in končnima točkama dovolj velika. Zaradi prilagodljivosti aplikacije ta razlika ni vnaprej definirana, temveč se izračuna glede na velikost ekrana, ki se uporablja za prikazovanje vsebine. Povečava se torej izvede, kadar je izpolnjena enačba (4.1), pomanjšava pa, kadar je izpolnjena enačba (4.2).

$$sirinaZaslona/5 < koncnaRazdalja - zacetnaRazdalja \quad (4.1)$$

$$0 - (sirinaZaslona/5) > koncnaRazdalja - zacetnaRazdalja \quad (4.2)$$



Slika 4.6: Gesta za pomanjšanje elementov na zaslonu

4.8 Mehanizmi za izboljšavo izkušnje

Zamenjava kurzorja

Standardna oblika kurzorja, ki se danes uporablja na praktično vseh sistemih, je puščica. Čeprav se puščica izkaže za zelo učinkovito in natančno pri normalnem uporabljanju računalnika z miško, pa se pri uporabljanju računalnika z razdalje izkaže za majhno in nepregledno. Za izboljšanje vidljivosti aplikacija omogoča, da za čas uporabe spremenimo videz kurzorja. Ker se aplikacija izvaja v ozadju in nima neposredne povezave s prikazovanjem vsebine, se videz kurzorja spremeni na ravni sistema. Ob zagonu aplikacije se v spomin shrani trenutni videz kurzorja. Če uporabnik v aplikaciji omogoči zamenjavo kurzorja, le ta spremeni vrednost v registru in osveži kurzor. Ob izklopu te opcije ali izhodu iz aplikacije se ponovno nastavi prvotni videz kurzorja.

Časovni zamik ukazov

Ker se vsi ukazi nanašajo na stanje dlani uporabnika, dlani pa sta samo

dve, se lahko ukazi prekrivajo oziroma prehitro izvršijo. Na primer, če hoče uporabnik vsebino povečati, vendar desno dlan zapre malo pred levo, se lahko namesto povečave nenamerno izvede klik. Zato je bil uveden kratek časovni zamik, preden se akcije izvedejo. Aplikacija tako na primer ob zaprtju desne dlani čaka še vnaprej določeno časovno enoto. Če v tem časovnem zamiku zapremo še levo dlan, se izvede povečava ali pomanjšava, če roka ostane odprta pa klik. Prednastavljen časovni zamik je 0.7 s.

Prikazovalnik sledenja rok

Prikazovalnik sledenja rok je namenjen konfiguriranju in testiranju same aplikacije in se ne uporablja pri normalnem delu z aplikacijo. S pomočjo prikazovalnika sledenja rok lahko preverimo, če naš sistem pravilno zazna levo in desno roko, odpiranje in zapiranje posamezne dlani ter odzivnost našega sistema. Prikazovalnik se odpre v ločenem oknu, kjer vsako roko predstavlja krog, ki ob zaprtju ali odprtju dlani spremeni barvo. Kroga se po oknu gibata skladno s premikanjem dlani.

Poglavje 5

Testiranje

Da bi raziskali možnosti za izboljšavo aplikacije, je bilo samo delovanje testirano pri več uporabnikih. Testiranje je potekalo na prenosnem računalniku z operacijskim sistemom Windows 8.1 ter senzorjem Kinect prve generacije. Senzor Kinect je bil priklopljen na prenosnik, sam prenosnik pa na različne vrste TV sprejemnikov, ki so služili kot ekran za prikazovanje. Ker je bilo testiranje opravljeno na več lokacijah in več TV sprejemnikih, so se s tem spreminjali tudi pogoji za delovanje (prostor, osvetlitev). Uporabniki so se najprej spoznali z ukazi preko uvodne aplikacije, kot je prikazana na sliki 5.1, ki se samodejno zažene ob zaznavi novega uporabnika. Nato so uporabnost aplikacije testirali z brskanjem po spletnih straneh. Pri testiranju se je pokazalo več težav, ki bi jih bilo z nadaljnim razvojem aplikacije mogoče izboljšati. Težave so podrobneje opisane v nadaljevanju.

Zaznavanje zaprte dlani

Kinect v svojem paketu Kinect for Windows SDK omogoča zaznavanje odprte oziroma zaprte dlani. Ker preko Kinect for Windows SDK prepoznavanje prstov ni možno, se za stanje roke gleda površina dlani. Pri testiranju se je izkazalo, da je zaznavanje zaprte dlani pri nekaterih uporabnikih slabše. Pomembno je, da ima uporabnik dlan popolnoma odprto in nato popolnoma zaprto, da je sprememba v površini strani čim večja.

Slika 5.1: Prva stran uvodne aplikacije, ki se zažene ob zaznavi novega uporabnika

Povratna informacija

Zaradi časovnega zamika pri izvajanju ukazov, ki je podrobneje opisan v poglavju 4.8, so imeli nekateri uporabniki težave. Čeprav so klik izvedli, se tega niso zavedali, saj niso dobili takojšnje povratne informacije. Vsebina na ekranu pa se je spremenila z zamikom.

Skaliranje premikov

Vsak premik uporabnika se v drugačnem razmerju pretvori v premik kurzorja na ekranu, da lahko uporabnik dela manjše gibe. Kljub temu pa so se različni uporabniki pred senzor postavili na različnih razdaljah. Zaradi razdalje so morali nekateri uporabniki izvajati večje gibe, da so dosegli celotno površino ekrana.

5.1 Možne izboljšave aplikacije

Spreminjanje kurzorja ob akciji

Ker se akcije v aplikaciji izvajajo z zamikom, uporabniki ne dobijo takoj povratne informacije. To bi lahko izboljšali s sprotnim spreminjanjem

kurzorja. Ob izvajanju klika bi se npr. spremenila oblika kurzorja. Tako bi uporabnik dobil potrditev, da je izvedel klik, še preden bi se z zamikom spremenila tudi vsebina na ekranu.

Dodajaje novih gest

Aplikacijo bi bilo mogoče razširiti z dodajanjem novih ukazov, ki bi pripomogli k hitrejšemu brskanju po vsebini. Dva ukaza, ki bi uporabnikom pomagala, sta premik na prejšno stran in premik na naslednjo stran. Ukaza bi se lahko izvajala preko geste, npr. zamah z roko v levo (swipe left) in zamah z roko v desno (swipe right).

Način za levičarje

Aplikaciji bi lahko dodali možnost, da si uporabnik izbere kot glavno roko levo ali desno. Vsi ukazi bi ostali enaki, zamenjati bi bilo potrebno le strani rok pri gestah.

Izboljšanje natančnosti

Čeprav je natančnost že kar visoka, saj se gibi gladijo s pomočjo opcij, vgrajenih v Kinect for Windows SDK, pa bi se jo dalo še izboljšati. Dodalo bi se lahko dodatno glajenje potez in pomagala za izvajanje gest. Ena izmed opcij bi bila npr. dodatna umiritev kurzorja, kadar bi bilo zaznano, da uporabnik izvaja klik.

Različne opcije za klik

Trenutna verzija aplikacije za klik uporablja stisk desne roke v pest, vendar bi se za klik lahko dodale še druge opcije. Ena izmed opcij je navidezni pritisk roke (pomik naprej), klik pa bi se lahko izvedlo tudi preko geste z uporabo obeh rok.

Prilagajanje vsebine glede na razdaljo

Ker omogoča senzor Kinect globinsko zaznavanje, bi se lahko vsebina na ekranu spreminjala glede na oddaljenost uporabnika od senzorja. Na večji razdalji bi se lahko tako vsebina avtomatično povečala, hkrati pa

bi se spremenilo tudi skaliranje gibov uporabnika za boljšo natančnost in uporabniško izkušnjo.

Poglavje 6

Sklep

Cilj diplomske naloge je bilo razviti aplikacijo, s katero bi lahko upravljali računalnik preko gest z uporabo leve in desne roke. Cilj razvoja te aplikacije je bilo med samim razvojem ugotoviti možnosti, prednosti, slabosti in potencialne izboljšave upravljanja računalnika preko senzorja Kinect oziroma na splošno upravljanje računalnika preko brezdotičnega naravnega vmesnika.

Sam razvoj aplikacije je zaradi dobre podpore, ki se je razvila od izdaje senzorja Kinect do izdelave te diplomske naloge potekal brez večjih težav. Strojna oprema je delovala kot pričakovano, vse knjižnice, podrobneje opisane v prejšnjih poglavjih, pa so poskrbele za hitro vzpostavitev osnovnega delovanja. Kinect je brez težav zaznaval osebe in sledil želenim točkam telesa. Prve težave so se pojavile, ko smo želeli prestopiti vnaprej podane možnosti. Težava je bila predvsem v natančnosti samega senzorja, ki je bil zasnovan za komuniciranje s posebej zato prilagojenimi sistemi. Taki sistemi imajo velike gumbe, posebna pomagala za lažjo izbiro, omejene možnosti izbiranja ipd. Cilj naše aplikacije pa je bilo natančnost izboljšati do te mere, da lahko preko naravnega vmesnika naše ukaze spremenimo v ukaze miške in tipkovnice.

Izkazalo se je, da se natančnost z nekaterimi metodami sicer da izboljšati, vendar pa ima vseeno svoje omejitve. S tem je pogojena tudi vrsta uka-

zov, ki jih lahko izvedemo in pa način uporabe aplikacije. Prava uporabnost aplikacije se pokaže šele pri uporabi v kombinaciji z aplikacijo za prikaz prilagojenega vmesnika, ki smo jo v diplomski nalogi poimenovali kiosk aplikacija. Upravljanje kiosk aplikacije je bilo preko brezdotičnega vmesnika dokaj preprosto in naravno. Tudi brskanje po straneh, ki so dovolj povečane in nimajo preveč vsebine z majhnimi povezavami je bilo dokaj uspešno. Uporaba aplikacije za vsakdanja opravila na računalniku pa ni mogoča zaradi premajhne natančnosti in premajhnega nabora ukazov. Aplikacijo bi se z dodatnimi izboljšavami dalo še razširiti ter narediti prijaznejšo uporabnikom. Samo aplikacijo bi lahko ocenili kot uspešno, saj je pokazala, da se z uporabo brezdotičnega upravljanja da doseči naravni uporabniški vmesnik in računalnik upravljati za namen, ki je bil cilj te diplomske naloge.

Literatura

[1] C. Bohak, M. Marolt, *Kinect Web Kiosk Framework*, Human Factors in Computing and Informatics Lecture Notes in Computer Science, št. 7946, str. 785-790, Nemčija: Springer, 2013.

[2] J. Jacko, *Human-Computer Interaction, Novel Interaction Methods and Techniques*, str. 161-169, Nemčija: Springer, 2009.

[3] K. O'Hara, R. Harper, H. Mentis, A. Sellen, A. Taylor, *On the naturalness of touchless: Putting the "interaction" back into NUI*, ACM Trans. Comput.-Hum. Interact., št. 20, str. 1-25, New York: ACM Press, 2013

[4] (2014) CodePlex: Kinect Mouse Cursor. Dostopno na: http://kinectmouse.codeplex.com/

[5] (2014) CodePlex: Kinect Toolbox. Dostopno na: http://kinecttoolbox.codeplex.com/

[6] (2014) CodePlex: Windows Input Simulator. Dostopno na: http://inputsimulator.codeplex.com/

[7] (2014) Coding4fun. Dostopno na: http://channel9.msdn.com/coding4fun/about

[8] (2014) David Renton's educational blog: Kinect Magic Cursor version 1.7 with Gesture support. Dostopno na: http://drenton72.wordpress.com/2013/05/09/kinect-magic-cursor-version-1-7-with-gesture-support/

[9] (2014) Easy Kinect Mouse Controller for Windows. Dostopno na: http://code.msdn.microsoft.com/windowsdesktop/Easy-Kinect-Mouse-09233c52

[10] (2014) Elliptic SDK. Dostopno na: http://www.ellipticlabs.com/?page_id=1805

[11] eyeSight: About the Technology. Dostopno na: http://eyesight-tech.com/technology/

[12] (2014) Kinect for Windows Developer Toolkit 1.8. Dostopno na: http://www.microsoft.com/en-us/download/details.aspx?id=40276

[13] (2014) Kinect for Windows features. Dostopno na: http://www.microsoft.com/en-us/kinectforwindows/discover/features.aspx

[14] (2014) Kinect for Windows Sensor Components and Specifications. Dostopno na: http://msdn.microsoft.com/en-us/library/jj131033.aspx

[15] (2014) Leap Motion. Dostopno na: https://www.leapmotion.com/

[16] (2014) Mono. Dostopno na: http://www.mono-project.com

[17] (2014) MSDN: Introduction to the C# Language and the .NET Framework. Dostopno na: http://msdn.microsoft.com/en-us/library/z1zx9t92.aspx

[18] (2014) MSDN: Overview of the .NET Framework. Dostopno na: http://msdn.microsoft.com/en-us/library/zw4w595w.aspx

[19] (2014) MSDN: Tracking Users with Kinect Skeletal Tracking. Dostopno na: http://msdn.microsoft.com/en-us/library/jj131025.aspx

[20] (2014) Tobii EyeX. Dostopno na:
http://www.tobii.com/en/eye-experience/eyex/#.Ut7Z6xA1iUk

[21] (2014) Wikipedia: .NET Framework. Dostopno na:
http://en.wikipedia.org/wiki/.NET_Framework

[22] (2014) Wikipedia: C Sharp (programming language). Dostopno na:
http://en.wikipedia.org/wiki/C_Sharp_(programming_language)

[23] (2014) Wikipedia: Kinect. Dostopno na:
http://en.wikipedia.org/wiki/Kinect

[24] (2014) Wikipedia: Microsoft Visual Studio. Dostopno na:
http://en.wikipedia.org/wiki/Microsoft_Visual_Studio

[25] (2014) Winect. Dostopno na:
http://ixorastudios.com/portfolio/winect/