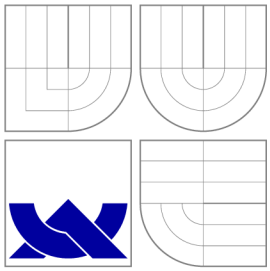
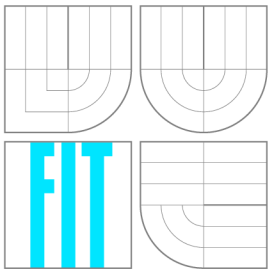


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

3D ŠACHY S VYUŽITÍM LEAP MOTION

3D CHESS WITH LEAP MOTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VÁCLAV SMEJKAL

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. ALENA PAVELKOVÁ

BRNO 2014

Abstrakt

Tato bakalářská práce se zabývá pohybovým ovladačem Leap Motion. Výstupem práce je šachová 3D hra, která nabízí ovládání pomocí tohoto zařízení. Teoretická část popisuje historii podobných vstupních zařízení, Leap Motion a grafický nástroj OpenSceneGraph, který je použit pro vytvoření 3D scény hry. Praktická část práce potom popisuje návrh a implementaci aplikace a vyhodnocení použitelnosti ovládání aplikace na základě testování.

Abstract

This Bachelor thesis deals with Leap Motion controller. The goal of this project is 3D chess game which can be controlled by Leap Motion device. Theoretical part of this thesis describes history of similar input devices, Leap Motion and OpenSceneGraph graphics toolkit which is used to create 3D scene. Then, practical part describes design and implementation of application and evaluation of usability of application control based on testing.

Klíčová slova

Leap Motion, pohybové ovládání, šachy, 3D grafika, počítačové hry, OpenSceneGraph, OSG, Qt

Keywords

Leap Motion, motion control, chess, 3D graphics, computer games, OpenSceneGraph, OSG, Qt

Citace

Václav Smejkal: 3D šachy s využitím Leap Motion, bakalářská práce, Brno, FIT VUT v Brně, 2014

3D šachy s využitím Leap Motion

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením paní Ing. Aleny Pavelkové. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Václav Smejkal
20. května 2014

Poděkování

Chtěl bych poděkovat své vedoucí bakalářské práce Ing. Aleně Pavelkové za odborné vedení, za pomoc a rady při zpracování této práce.

Obsah

Kapitola 1

Úvod

Způsoby interakce člověka s počítačem prošly za posledních pár desítek let dlouhým a pestrým vývojem. Kromě klasických zařízení, které dennodenně používáme, jako je klávesnice či myš, dnes také existují modernější druhy interakce. Poslední dobou jsou populární dotykové displeje a pomalu se začíná prosazovat i bezdotykové ovládání pohybem končetin v prostoru. Právě tento druh interakce člověka s počítačem se snaží prosadit poměrně nové zařízení Leap Motion, které dokáže velmi detailně snímat lidské ruce a rozpoznávat specifická gesta.

Cílem této bakalářské práce je navrhnout a implementovat aplikaci, která nabízí rozhraní ovladatelné pomocí senzoru Leap Motion, a pokusit se tak ověřit využitelnost tohoto zařízení. Dále je cílem vyhodnotit použitelnost tohoto rozhraní formou testování na uživatelích. Obsahem aplikace je šachová hra, přičemž šachovnice je zobrazena ve trojrozměrné scéně. Důvodem výběru tohoto typu aplikace je přirozená možnost mapování virtuálního 3D prostoru šachovnice na reálný prostor nad zařízením Leap Motion. Toto mapování by tak mělo poskytnout přirozenější přístup ke hraní šachové partie.

Technická zpráva se skládá z několika hlavních částí. V kapitole 2 je shrnuta historie a vývoj předchozích existujících řešení pro interakci člověka s počítačem, zejména těch ve formě pohybového ovládání. Kapitola 3 nabízí bližší pohled na zařízení Leap Motion, které je v této práci využito, mimo jiné na jeho technologii, systémovou architekturu a aplikační programové rozhraní. Teoretickou část zprávy pak zakončuje kapitola 4 o grafickém nástroji OpenSceneGraph, pomocí kterého je vytvořena 3D scéna. Praktickou část zastupuje kapitola 5, která pojednává o detailech návrhu a implementace samotné aplikace. V poslední kapitole 6 je popsán průběh testování uživatelského rozhraní na uživatelích a jsou shrnuty výsledky tohoto testování zejména z pohledu použitelnosti rozhraní.

Kapitola 2

Interakce člověka s počítačem

Interakce člověka s počítačem je umožněna vstupními a výstupními zařízeními. Úlohu výstupního zařízení již dlouhou dobu plní displeje různých druhů, lišících se zejména technologií zobrazování. Díky těmto výstupním prvkům člověk při interakci s počítačem dostává důležitou zpětnou vazbu. Z této vazby potom získává zkušenosti a na základě těchto zkušeností se učí používat vstupní zařízení. Právě vstupním zařízením se bude věnovat tato kapitola.

2.1 Klasická vstupní zařízení

Když pomineme éru děrných štítků, hlavním vstupním prvkem počítače byla vždy nějaká forma klávesnice, jejíž vývoj vycházel z klasického psacího stroje. Ta již v ustálené formě setrvává u našich počítačů dodnes.

V 60. letech minulého století Douglas Engelbart vynalezl další důležitou periferii, a to počítačovou myš, která ve své první podobě obsahovala dva kovové kotoučky pro udávání změn pozice v obou osách a červené tlačítko pro klikání. Patent poté převzala firma Xerox a následně firma Apple, až díky které se myš stala populární. Historicky existovalo několik druhů myší: kuličková, optická a laserová, přičemž dnes se již používají pouze poslední dvě varianty [13].

Je taktéž důležité zmínit ještě jedno zařízení, které dnes již také můžeme považovat za klasické. S nárůstem počtu přenosných počítačů — notebooků — roste také použití tzv. touchpadu. Tato ploška využívající elektrické kapacity dokáže nahradit počítačovou myš, ale často postrádá potřebnou ergonomii.

2.2 Dotykové displeje

Dá se říci, že touchpad, který byl zmíněn v předchozí kapitole, je pomyslným mostem k dalšímu způsobu ovládání počítače, a to dotykovému displeji. Ten také nejčastěji používá kapacitní technologii rozpoznání pozice prstu uživatele. Rozdíl je v přímočařejší interakci s dotykovými displeji, kde se stačí přímo dotknout ovládacího prvku a není zde potřeba nejprve posunout kurzor na požadované místo jako u touchpadu.

2.3 Herní zařízení

S ohledem na segment počítačových her vzniklo vedle klasických vstupních zařízení také nepřeberné množství herních ovladačů, kterým se často říká gamepad. Ty typicky obsahují

kromě tlačítek různé analogové páčky apod., ve snaze herních tvůrců o co nejrealističtější a nejpříjemnější zážitek ze hry.

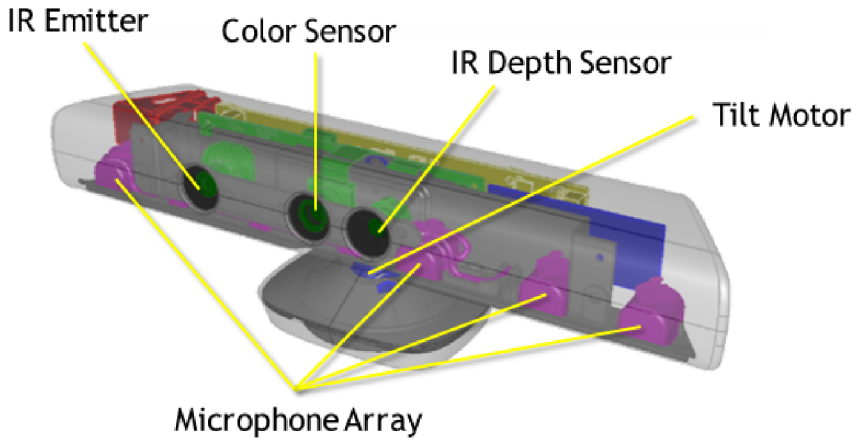
2.4 Pohybové ovládání

Jelikož je tato práce zaměřena na implementaci hry ovládané pomocí pohybového ovládání, následující část textu bude zaměřena na tento druh interakce.

De facto existují dva rozdílné přístupy k pohybovému ovládání. U prvního je vyžadováno, aby uživatel držel v rukou fyzický ovladač a prováděl s ním vhodná gesta. V takovém případě většinou ovladač obsahuje akcelerometr, který dokáže detekovat orientaci ovladače a také zrychlení při pohybu v prostoru. V druhém případě, pro mnohé přirozenějším, je uživatel oproštěn od jakýchkoliv ovladačů a interakci uskutečňuje pouze za pomoci svého těla, zejména rukama. Technicky je toto možné za účasti vhodného typu kamery a algoritmů počítačového vidění, které je schopné detekovat a sledovat tělo resp. ruku uživatele, nebo dokonce několika uživatelů zároveň. To je výsadou zejména ovladače Kinect od firmy Microsoft, která jej nabízí ke svým herním konzolím.

2.4.1 Microsoft Kinect

Popularita Kinectu v posledních letech velmi vzrostla. Toto zařízení primárně slouží k rozpoznávání a sledování těla uživatele a jeho gest za účelem ovládání různých aplikací, většinou her. K tomu slouží hloubkový senzor v podobě infračervené kamery, který tak není ovlivněn světelnými podmínkami, barvou kůže, oblečením ani barvou pozadí za uživatelem. Nicméně v základním provedení není Kinect schopen rozpoznávání na úrovni prstů lidské ruky, takže například poskytuje data pouze o vzájemné orientaci rukou a paží [6]. Na obrázku 2.1 jsou znázorněny jeho hlavní komponenty.



Obrázek 2.1: Hlavní komponenty zařízení Kinect (Zdroj: http://msdn.microsoft.com/en-us/library/jj131033.aspx).

Protože je Microsoft Kinect v současnosti zřejmě nejznámějším prostředkem pro pohybové ovládání bez nutnosti uživatele mít na sobě jakýkoliv ovladač, bude v další sekci stručně uvedena historie podobných zařízení ve spojitosti s vývojem směrem ke Kinectu.

2.4.2 Stručný historický přehled pohybového ovládání

Zatímco firma Microsoft získala za svoje zařízení Kinect všeobecné uznání veřejnosti, rozhodně není Kinect prvním zařízením, které se pokouší o interakci „holýma rukama“. Následuje tedy stručný přehled, jak se podobná — především herní — zařízení vyvíjela posledních 20 let [3].

Je nutné zmínit, že technologie rozpoznávání obrazu nebyla před 20 lety ani zdaleka na takové úrovni jako dnes, tudíž většina pokusů o ovládání bez nutnosti mít v rukou jakýkoliv ovladač skončila ne příliš použitelnými výrobky.

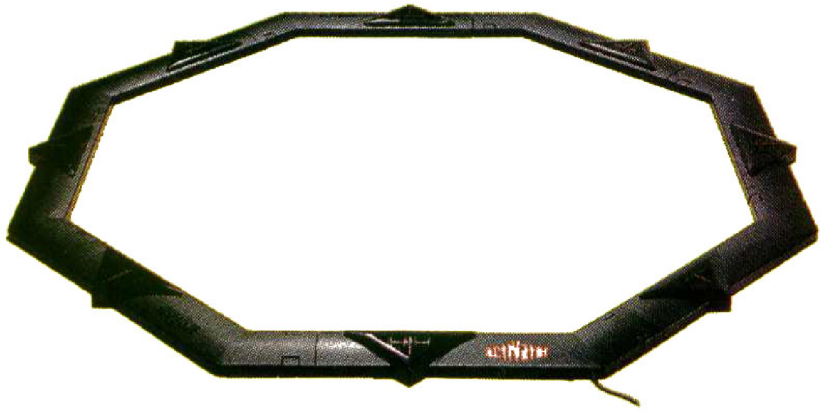
Jedním z prvních takovýchto produktů byl U-Force (na obrázku 2.2) od společnosti Broderbund, která jej vytvořila na konci 80. let pro zábavní systém Nintendo Entertainment System. Dá se říci, že to byl prapředek Kinectu a je mu až dodnes asi nejblíže. Zařízení vypadalo jako těžkopádný notebook, také se tak otevíralo, a pomocí infračervených senzorů detekovalo pohyby rukou ve svém zorném poli. Jak se lze domnívat, v roce 1989 nebyl tento způsob snímání na takové úrovni jako dnes a ve výsledku U-Force často selhávalo v rozpoznávání pohybů a gest.

Obrázek 2.2: Pohybový ovladač U-Force (Zdroj: http://nintendo.wikia.com/wiki/U-Force).

Dalším pokusem bylo v roce 1993 herní zařízení Activator od firmy Sega v podobě plastového ringu, sloužícího ke hraní bojových her. Je vidět na obrázku 2.3. Tento osmiúhelník se skládal z několika plastových dílků položených na zemi do kruhu. Ty poté nad sebou snímaly pohyby končetin uživatele a interpretovaly je jako kopy nebo údery ve hře. Stejně jako U-Force zařízení používalo technologii infračervených senzorů a stejně tak nefungovalo moc dobře. Navíc každá strana osmiúhelníku byla mapována na jedno tlačítko běžného ovladače a pohybem končetiny přes tuto stranu se aktivovalo právě jedno přiřazené tlačítko. Bohužel tento přístup přispěl k nepříjemnému a nespolehlivému ovládání.

V roce 2003 firma Sony představila pro svou herní konzoli zařízení EyeToy (na obrázku 2.4), které vypadalo jako webová kamera a de facto již fungovalo podobně jako zařízení Kinect, i když nesklidilo tolik úspěchu. Uživateli umožňovalo vidět sebe samého ve sportovních, tanečních nebo pěveckých hrách. Z tohoto zařízení v posledních letech vychází další produkt od Sony s názvem Playstation Eye, ale u něj již ke hraní uživatel potřebuje držet v rukou speciální ovladače navíc, tudíž toto zařízení zde nebude blíže rozebíráno.

Ovládání pohybem končetin bez nutnosti mít na sobě jakoukoliv formu ovladače tedy prošlo pestrým vývojem a od počátečních neúspěchů, které byly způsobeny nevyspělou



Obrázek 2.3: Zařízení Sega Activator (Zdroj: http://blog.wirebot.com/2011/07/21/what-the-peripheral-sega-activator-2/).

Obrázek 2.4: Kamera Sony EyeToy (Zdroj: http://en.wikipedia.org/wiki/File:EyeToy.png).

technologií infračervených senzorů, a tedy často nesprávně detekovaných gest, se dostáváme k dnes již slušně použitelným nástrojům, jako je Microsoft Kinect nebo nově Leap Motion. O posledně jmenovaném bude pojednávat další samostatná kapitola, jelikož Leap Motion hraje důležitou roli v této práci.

Kapitola 3

Leap Motion

Leap Motion je periferní zařízení, které umožňuje velmi přesné snímání pohybu prstů, rukou a podobně velkých předmětů. Jeho hlavním účelem je poskytnout nový způsob interakce s běžnými počítačovými zařízeními, jako je osobní počítač, laptop, mobilní zařízení a podobně. V konfrontaci s klasickými ovládacími prvky, mezi než patří myš, klávesnice či touchpad, způsobuje Leap Motion malou revoluci a dovoluje ovládat počítačová zařízení pomocí pohybů rukou v prostoru. Jinými slovy, je schopno mapovat pohyby rukou ve 3D prostoru na prvky rozhraní dvourozměrné obrazovky.

3.1 Historie

V roce 2010 byla založena společnost OcuSpec [7]. Ta začala vyvíjet zařízení na bázi pohybového ovládání, které by bylo výkonné, multiplatformní a cenově velmi dostupné. V roce 2011 poté firma získala investici ve výši 1,3 milionu dolarů [12]. Následně se společnost přejmenovala na Leap Motion, Inc. 21. května 2012 představila svůj první produkt, The Leap, který je dnes známý pod jménem Leap Motion. Prezentační videa produktu vzbudila všeobecné nadšení, diskutovala se zejména vysoká přesnost a zároveň malé rozměry zařízení a nízká cena. Často bylo srovnáváno s podobnou periferií Kinect od firmy Microsoft, která je velmi úspěšná. Ta sice nemá tak přesné snímače, avšak na druhou stranu snímá celou postavu člověka. O této periferii již bylo pojednáno v kapitole 2.4.1. V případě Leap Motion tedy šlo o nový druh zařízení snímající prostor nad sebou v mnohem větších detailech.

Ještě na konci roku 2012 firma rozdala 10.000 kusů Leap Motion zaregistrovaným vývojářům, aby mohli prozkoumat tuto novou platformu a přicházet s novými nápady na využití tohoto zařízení [11]. K dispozici také dostali softwarový vývojový nástroj (SDK) a aplikační programové rozhraní (API).

V červnu 2013 byl spuštěn internetový obchod s aplikacemi pro Leap Motion, nazvaný Airspace. Ten má sloužit jako hlavní a jediný zdroj aplikací pro tuto platformu [8] a je blíže popsán v kapitole 3.2.

V polovině roku 2013 pak ovladač Leap Motion vstoupil na trh [2]. Od tohoto milníku také začaly rychle přibývat nové aplikace v obchodě Airspace.

Na konci listopadu 2013 společnost uvolnila první velkou aktualizaci svého softwaru, díky níž má zařízení umět ještě lépe a spolehlivěji snímat ruce a dokonce odhadovat jejich polohu, i když jsou mimo jeho zorné pole [10].

3.2 Webový obchod Airspace

Jak již bylo zmíněno v kapitole 3.1, společnost Leap Motion spustila během roku 2013 internetový obchod s názvem Airspace Store. Ten funguje jako hlavní zdroj aplikací. Dají se zde najít jak placené aplikace, tak bezplatné, kterých je dostatek a často mohou sloužit k rychlému seznámení se zařízením.

Aktuální celkový počet aplikací (květen 2014) je něco málo přes 200. Vývojáři, kteří vytvoří zajímavou aplikaci, mohou zažádat o její zařazení do nabídky obchodu, přičemž aplikace musí projít schvalovacím procesem. Samotné požadavky pro schválení aplikace jsou docela přísné [5], na druhou stranu se těmito opatřeními zajišťuje kvalita a bezpečnost nabízených aplikací.

Stažení aplikace z Airspace Store a následná instalace je možná pouze s nainstalovaným speciálním softwarem, který se stará o běh služby v operačním systému, nabízí ovládací panel s širokou paletou nastavení a přístup k nainstalovaným aplikacím. V neposlední řadě software obsahuje důležité ovladače pro Leap Motion, díky kterým může zařízení fungovat a být spojeno s počítačem. Tento software je dostupný pro všechny běžné operační systémy, tedy Windows (od verze 7), OS X (od verze 10.7) i Linux (Ubuntu od verze 12.04 LTS) [5].

3.3 Konstrukce

Konstrukčně se jedná o malou krabičku o rozměrech 80 x 30 x 11 milimetrů. Váží 45 gramů. Vzhled je strohý až minimalistický, jak lze vidět na obrázku 3.1. Spodní strana je potažena protiskluzovým materiálem a boční strany obepíná hliníkový kryt, který je schopen účinně odvádět teplo, jelikož zařízení se při své činnosti nezanedbatelně zahřívá. Vrchní stranu pak tvoří lesklý plastový kryt, pod nímž se nacházejí hlavní komponenty zařízení. O těch je pojednáno v kapitole 3.4.

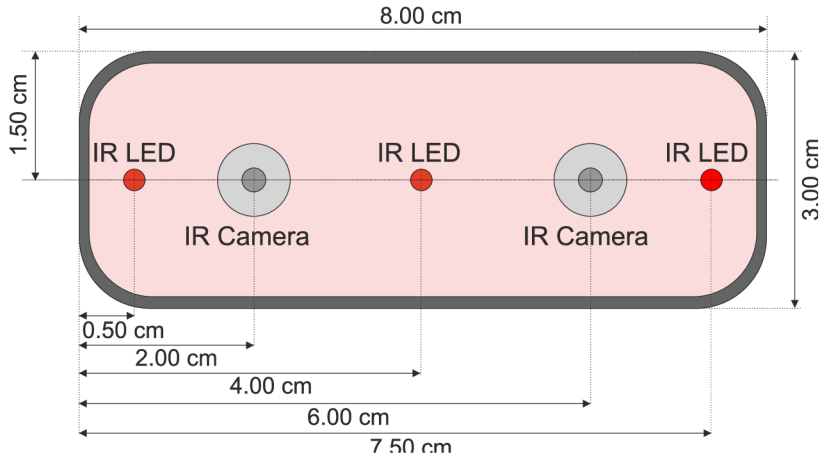
Obrázek 3.1: Zařízení Leap Motion (Zdroj: http://www.laptopmag.com/reviews/accessories/leap-motion-controller.aspx).

Jelikož zařízení komunikuje s počítačem pomocí standardu USB, jsou v dodávaném balení přiloženy dva kabely USB 2.0 s konektory microUSB 3.0 a rozdílnou délkou pro různé možnosti připojení [5]. Skrze kabel je Leap Motion také napájen.

3.4 Technologie a možnosti ovládání

Leap Motion deklaruje, že zařízení má přesnost snímání 0,1 milimetru [5]. To je tedy mnohem více než u již diskutovaného senzoru Kinect od společnosti Microsoft a ostatních bezdotykových produktů. Leap Motion je zhruba 200 krát přesnější [7]. To například znamená rozdíl mezi schopností detekce mávnutí rukou ve vzduchu a schopností vytvořit velmi přesný digitální podpis pomocí prstu ruky nebo pera. Leap Motion má pouze základní nezbytnou funkcionalitu na straně hardwaru, a to vytvářet proud snímků dat a posílat je do počítače. Těchto snímků dokáže vytvořit přes 200 za sekundu [5].

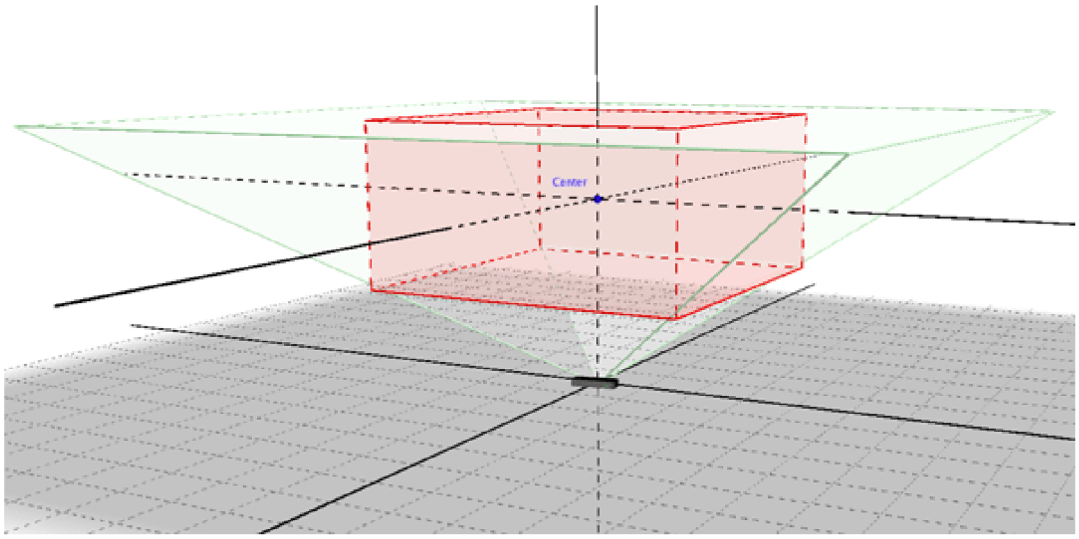
Vzhledem k tomu, že Leap Motion je chráněno patentem, nejsou známy všechny detaily, jak toto zařízení přesně funguje. Nicméně díky těm, kteří jej rozebrali a prozkoumali, je jasné, z jakých základních komponent se Leap Motion skládá. Pozice ruky, kterou senzor dodává, je relativní vzhledem ke středovému bodu zařízení, které je schematicky znázorněno na obrázku 3.2. V tomto bodě je umístěna infračervená LED. Celkově ovladač obsahuje tři tyto infračervené LED a dvě infračervené kamery, které jsou situovány po jedné na každé straně [15].



Obrázek 3.2: Schematický nákres Leap Motion (Zdroj: [15].

Software Leap Motion analyzuje objekty nacházející se v zorném poli zařízení, které si lze představit jako pyramidu obrácenou vzhůru nohama. Účinný rozsah snímání lze stanovit na 25 až 600 milimetrů nad zařízením [4]. Rozpětí od základního bodu je zhruba 150 stupňů [5], tudíž je zde dostatečný prostor pro manipulaci. Oficiální webové stránky dále uvádějí existenci pomyslného kvádru pro interakci, který se nachází v již zmíněné pyramidě. Celá situace je znázorněna na obrázku 3.3. Tato oblast nabízí normalizované souřadnice snímaných rukou, prstů či nástrojů (např. ukazovátko), to znamená, že poskytuje jednodušší mapování pozic do 2D či 3D souřadnicového systému použitého pro vykreslování scény aplikace [5].

Leap Motion podporuje nezávislou identifikaci jednotlivých rukou, prstů a malých předmětů, jako je například tužka. Zároveň je vestavěno rozpoznání specifických gest, konkrétně mávnutí rukou, kroužení prstem, ale také napodobování práce s kurzorem myši. Toto je založeno na pomyslném rozdělení snímaného prostoru na dvě poloviny, bližší a vzdálenější od uživatele. Při pohybu prstem v bližší části se toto pokládá za pohyb kurzoru a na obrazovce by měla být vykreslena odpovídající odezva. Při vniknutí prstu do vzdálenější části se tento pohyb směrem do obrazovky chápe jako zmáčknutí levého tlačítka myši. Obecně je ale podporována myšlenka ovládání pomocí jednoduchých, lehce zapamatovatelných gest



Obrázek 3.3: Prostor pro interakci s Leap Motion (Zdroj: [5]).

namísto toho, aby se pohybové ovládání snažilo nahradit kurzor myši.

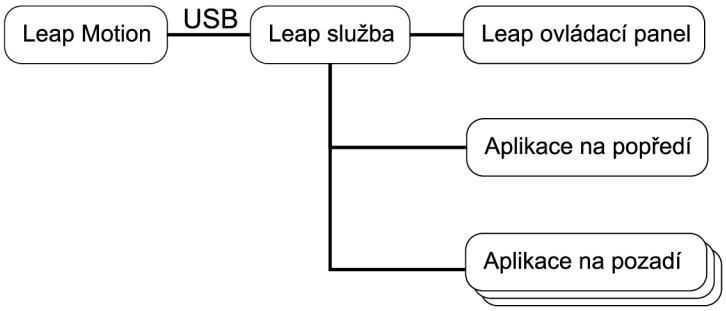
3.5 Systémová architektura

Jak již bylo zmíněno v kapitole 3.2, Leap Motion software běží jako služba v operačním systému počítače. Dále se připojuje k samotnému zařízení přes USB sběrnici. Spuštěnou službu mohou využívat aplikace pro Leap Motion a získávat tak data ze sledování pohybu. Leap Motion SDK (softwarový vývojový nástroj, dále také SDK) nabízí dvě varianty API (aplikační programové rozhraní, dále také API) pro načtení těchto dat. Je to buď API pro nativní rozhraní nebo pro rozhraní za použití webových soketů. Druhá jmenovaná varianta tak dovoluje vyvíjet webové aplikace, například v jazyce JavaScript, ovladatelné pomocí Leap Motion. Jelikož cílem této práce je implementace klasické aplikace, bude zde blíže přiblížena pouze první varianta, tedy architektura a API pro nativní rozhraní.

Nativní aplikační rozhraní je přístupné skrze dynamicky načtenou knihovnu. Ta se připojuje ke spuštěné Leap Motion službě a poskytuje snímaná data aplikacím. Schéma systémové architektury pro nativní rozhraní je na obrázku 3.4. Služba Leap Motion získává data ze zařízení přes USB sběrnici, zpracovává je a posílá běžícím aplikacím, které chtějí využívat toto rozhraní, přičemž standardně jsou doručovány pouze aplikaci na popředí. Nicméně každá aplikace spuštěná na pozadí může o zasílání těchto informací požádat. Leap Motion ovládací panel běží od služby odděleně a dovoluje uživateli nastavit některé parametry zařízení a instalace. V případě, že aplikace na popředí ztratí fokus v rámci operačního systému, služba Leap Motion jí přestane data posílat [5].

3.6 API pro Leap Motion

Aby bylo pro vývojáře co nejsnadnější vytvořit aplikaci, která má pracovat s Leap Motion, jsou dostupné softwarové vývojové nástroje (SDK) a s tím spojené knihovny aplikačního programového rozhraní (API). Dokumentace k API je dostupná na oficiálních webových stránkách Leap Motion.



Obrázek 3.4: Systémová architektura Leap Motion.

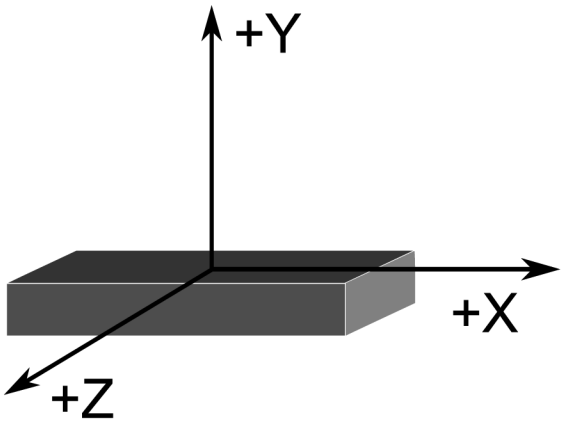
Pro zaregistrované vývojáře je zpřístupněna také přehledná dokumentace a příklady. Dále je možno komunikovat v rámci komunity a sledovat dění na fóru. Existuje tedy dostatek možností, jak začít a postupovat při vývoji aplikací pro Leap Motion.

Aplikace se dají napsat v několika programovacích jazycích. Jedná se o C++, C# v kombinaci s herním enginem Unity, dále je možné programovat v Javě, Objective-C, Javascriptu nebo Pythonu. Každý vývojář si tak může vybrat jazyk podle svých preferencí.

V následujících odstavcích bude uveden stručný přehled API pro nativní rozhraní [5].

3.6.1 Systém souřadnic a jednotky

Leap Motion API používá pravotočivou kartézskou soustavu souřadnic. Počátek soustavy je situován ve středu vrchní strany ovladače. Osy x a z leží v horizontální rovině, přičemž osa x je rovnoběžná s delší stranou zařízení. Osa y je pak vertikální, s kladnými hodnotami směrem vzhůru (v kontrastu s opačným přístupem ve většině počítačových grafických systémů souřadnic). Osa z nakonec směřuje k uživateli. Vše je vidět na obrázku 3.5.



Obrázek 3.5: Systém souřadnic Leap Motion.

API měří fyzikální veličiny v následujících jednotkách:

- vzdálenost v milimetrech,
- čas v mikrosekundách,
- rychlost v milimetrech za sekundu,
- úhel v radiánech.

3.6.2 Informace o snímaném pohybu

Senzor Leap Motion snímá ruce, prsty a nástroje (např. ukazovátko), přičemž kontinuálně poskytuje sadu dat o těchto entitách. Sadě dat se v případě Leap Motion říká rámec, a ten je v API reprezentován objektem třídy `Frame`. Ten kromě již zmíněných entit obsahuje informace o rozpoznaných gestech a také o celkovém pohybu v zorném poli zařízení. Objekt typu `Frame` je tak základním kořenem datového modelu Leap Motion.

Ruce Třída `Frame` v prvé řadě uchovává data o detekovaných rukách. Leap Motion je schopno detekovat i více jak dvě ruce, ale pro optimální přesnost snímání je doporučeno využít maximálně tento počet. Ruku reprezentuje objekt třídy `Hand`. Ten obsahuje informace o její pozici, pohybu, další její charakteristiky a seznam jejích prstů a nástrojů, které jsou v ruce drženy.

Prsty a nástroje Kromě prstů lze pro interakci s Leap Motion použít i různé nástroje, které jsou tvarem podobné prstům, např. tužka nebo ukazovátko. V souvislosti s prsty jsou nástroje klasifikovány jako delší, tenčí a rovnější. Nicméně prsty i nástroje jsou sdruženy do jedné třídy objektů `Pointable`. V případě potřeby rozlišení těchto dvou druhů jsou připraveny třídy speciálně pro prst, třída `Finger`, a pro nástroj, `Tool`. Třída `Finger` pak např. poskytuje atributy `tipPosition`, znamenající pozici konečku prst, a `direction`, udávající směr, kterým prst ukazuje.

Gesta Aby vývojáři nemuseli implementovat rozpoznávání speciálních gest uživatele, nabízí Leap Motion API několik předdefinovaných jednoduchých gest, která automaticky detekuje a podává o nich informace prostřednictvím objektu třídy `Frame`, tedy stejným způsobem jako o základních snímaných entitách. Jsou podporována tato čtyři gesta:

- kroužení prstem,
- mávnutí rukou,
- mávnutí prstem dolů (imitace klávesnicového stisku),
- posunutí prstu vpřed (imitace dotknutí se dotykové obrazovky).

Aby vývojář mohl ve své aplikaci použít některá z těchto gest, musí nejdříve zvlášť každé povolit. To se děje pomocí metody `enableGesture()`, která patří třídě `Controller` reprezentující ovladač Leap Motion.

Pohyby Leap Motion také zaznamenává základní druhy pohybů: změnu měřítka, rotaci a posunutí. Tyto pohyby jsou vypočítávány mezi dvěma datovými rámci (`Frame`). Jejich parametry pak lze získat buďto v kontextu celé scény z objektu `Frame` (rámec), nebo v kontextu jedné ruky z objektu `Hand` (ruka). V tabulce 3.1 je uvedeno, jak Leap Motion pohyby interpretuje v obou kontextech.

V reálné aplikaci může vývojář tyto interpretované pohyby využít například k manipulaci s grafickým objektem.

Typ pohybu	`Frame` **(Rámec)**	`Hand` **(Ruka)**
Změna měřítka	Pohybování předmětů ve scéně k sobě a od sebe. Např. ruce se vzájemně přibližují/oddalují.	Roztažení/semknutí prstů ruky.
Rotace	Rozdílné pohyby předmětů ve scéně. Např. jedna ruka nahoru a druhá dolů.	Natočení jedné ruky.
Posunutí	Průměrná změna pozice předmětů ve scéně. Např. obě ruce se posunují jedním směrem.	Změna pozice jedné ruky.

Tabulka 3.1: Jak Leap Motion interpretuje pohyby předmětů ve scéně.

3.7 Využitelnost a budoucnost

Při pohledu na různé způsoby ovládání a vůbec interakce s počítačovými zařízeními vyvstává otázka, jak je ten či onen způsob použitelný a využitelný. Pokud jde o pohybové ovládání a konkrétně pomocí zařízení Leap Motion, často lze vidět snahu o nahrazení klasického ovládání pomocí myši a klávesnice touto formou. Řešení to jsou ovšem často trochu neohrabaná a ozývají se stížnosti na nepřesnost, nespolehlivost a slabou odezvu. Také se objevují námitky na vedlejší efekt tohoto způsobu ovládání, a tím je únava lidských končetin, které musejí být neustále v napětí.

Dobrá využitelnost zařízení může existovat např. ve zdravotnictví, kde bezdotykové ovládání přístrojů umožňuje lépe zachovat sterilní prostředí. Dalším vhodnou oblastí může být manipulace s 3D objekty a jejich vytváření, kde klasické formy ovládání nemohou nikdy poskytnout takovou míru intuitivnosti. Také by se dala představit dobrá použitelnost v situacích, ve kterých člověk nemá k dispozici zobrazovací zařízení v takové velikosti a poloze, aby je mohl ovládat přímo dotyky, ale displej je ve formě, jak jej známe z nové oblasti chytrých brýlí. V tomto případě je obrazovka v bezprostřední blízkosti očí, není ji tedy možné ovládat dotyky, a právě tehdy by se mohlo najít využití pro ovládání takovýchto zařízení gesty rukou v prostoru.

Ředitel společnosti Leap Motion, Michael Buckwald, oznámil, že senzor Leap Motion bude ve zmenšené podobě vestavěn do chytrých telefonů a tabletů. Toto je očekáváno ke konci roku 2014. Již nyní je Leap Motion vestavěn z výroby do notebooku HP Envy 17 od firmy Hewlett Packard. V tomto případě je senzor umístěn vedle touchpadu přístroje a uživatelé tak mohou používat gesta k ovládání počítače bez jakýchkoliv přídavných periferií. Dále je možné, že Leap Motion bude postupem času vestavěn také do televizí a interiérů aut, přičemž v případě posledně zmíněného by se pravděpodobně zvýšila bezpečnost interakce s ovládacími prvky auta [1].

Kapitola 4

Grafický nástroj OpenSceneGraph

Jelikož jedním ze stěžejních bodů této práce je vytvoření trojrozměrné scény šachové partie, musel být pro splnění tohoto účelu zvolen grafický nástroj, jenž nabízí odpovídající prostředky pro implementaci takové scény. Nakonec byl vybrán grafický nástroj OpenSceneGraph (dále také zkráceně OSG). V této kapitole bude blíže popsána jeho obecná charakteristika (4.1), historie (4.3), a také budou zmíněny základní komponenty tohoto nástroje (4.4). V sekci 4.2 budou přiblíženy principy tzv. grafu scény, což je základní datová struktura, na které je OSG postaven.

4.1 Obecná charakteristika

Jedná se o tzv. vykreslovací middleware aplikaci (spojovací vrstva nízkoúrovňových příkazů operačního systému a softwarových aplikací), která zvyšuje úroveň abstrakce a snižuje tak složitost vzhledem k použití nízkoúrovňové grafické knihovny OpenGL, i když na úkor flexibility. Dá se tedy říci, že OpenSceneGraph je jakási nadstavba právě nad knihovnou OpenGL a poskytuje vyšší úroveň programování grafických aplikací. Dále nabízí vysokou modularitu a objektový přístup, což je využito zejména ve správě grafických primitiv, materiálech a dalších datových sadách v uživatelských aplikacích. To má za následek kratší čas potřebný k vývoji aplikací a lepší možnosti dodávání nových funkcionalit (kombinovaných jako moduly a zásuvné moduly).

OpenSceneGraph je systém, který je postaven na teorii grafu scény (blíže v kapitole 4.2). Systém zaznamenává vykreslovací příkazy a data do své vyrovnávací paměti, aby je mohl ve vhodném čase provést. Díky tomu mohou být před samotným vykreslením scény podle grafu scény provedeny různé optimalizace a v případě složitých scén také vícevláknový přístup pro lepší výkon [14].

4.2 Graf scény

Graf scény je obecná datová struktura, která definuje a uchovává prostorové a logické vlastnosti grafické scény. Je vhodným prostředkem pro efektivní správu a vykreslování všech jejích částí. Graf scény je typicky reprezentován acyklickým orientovaným grafem (obrázek 4.1), který obsahuje různé druhy uzlů. Vždy se skládá z kořenového uzlu, skupinových uzlů (ty mohou obsahovat libovolný počet dětí) a listových uzlů, které nemají žádné dětí a představují tak spodní vrstvu grafu. Graf scény běžně nesmí obsahovat cyklus ani osamocený uzel (nemá rodiče ani děti).

Obrázek 4.1: Hierarchie grafu scény.

Jak již bylo zmíněno, skupinový uzel může mít libovolný počet dětí. Vlastnosti tohoto skupinového uzlu jsou pak automaticky přenášeny na všechny jeho děti. Lze tak sdílet informace o uzlu a brát celý podgraf začínající skupinovým uzlem jako jeden.

V grafu scény také může nastat situace, že uzel má více jak jednoho rodiče. V tomto případě lze říci, že je vícenásobně duplikován (případ listového uzlu na obrázku 4.1). To přináší velmi užitečný efekt, a to možnost sdílení grafických objektů v grafu scény.

Koncept grafu scény je kromě nástroje OpenSceneGraph široce využit ve známých moderních grafických aplikacích, jako je AutoCAD, Maya či CorelDraw [14].

4.3 Historie

Projekt OpenSceneGraph byl vytvořen v roce 1998 Donem Burnsem, který pracoval ve společnosti Silicon Graphics Inc. (SGI) a měl tento projekt jako vedlejší činnost. Byl vášnivý rogalista, a tak pro svou potřebu simulovat právě let rogalem vytvořil na platformě Linux aplikační programové rozhraní grafu scény a pojmenoval jej SG (prototyp OSG). SG de facto vycházel z nástroje OpenGL Performer od SGI.

Následně v roce 1999 se Robert Osfield, mimo jiné designový konzultant jednoho výrobce rogal, začal také angažovat v tomto projektu. Zachoval jeho samostatnost a open source licenci a následně projekt zpřístupnil i pod operačním systémem Windows. Poté jej zcela převzal na sebe a přejmenoval jej z SG na dnešní OpenSceneGraph. Společně s tím byl OSG také kompletně přepsán, aby efektivně využíval standardy jazyka C++ a návrhové vzory.

V důsledku zvyšujícího se zájmu o tento projekt na něm Robert Osfield začal pracovat na plný úvazek a v roce 2001 založil společnost *OpenSceneGraph Professional Services*, která měla za úkol dále vyvíjet OSG a poskytovat komerční i bezplatné služby k OSG. Podobným způsobem pokračoval také zakladatel Don Burns, jenž založil firmu *Andes Computer Engineering*, taktéž pro podporu vývoje OSG.

V roce 2003 byla představena knihovna *Producer*, která zajišťovala integraci okenního

systému, a dále důležité knihovny *osgText* a *osgFX* (knihovny blíže popsány v kapitole 4.4). Během roku 2005 pak byl oznámen OSG 1.0 reprezentující první finální verzi OpenSceneGraph.

Druhá verze, OSG 2.0, byla uvolněna v roce 2007. Obsahovala vylepšené jádro, vícenásobnou podporu grafických procesorů a také tři nové knihovny: *osgViewer*, *osgManipulator* a *osgShadow*. Tímto byla nahrazena knihovna *Producer*, která od té doby byla spravována jako nezávislý projekt.

Díky velmi rychlému vývoji v dalších letech přibývaly následující knihovny: *osgWidget* v roce 2008, *osgVolume* a *osgAnimation* v roce 2009 a *osgQt* v roce 2010. V témže roce pak byla uvolněna třetí verze, OSG 3.0, která zahrnovala podporu pro OpenGL 3.0 a OpenGL ES.

Několik set vysoce výkonných aplikací dnes používá právě OpenSceneGraph, zejména pro vykreslování složitých scén a spravování obsáhlých kolekcí dat. Rovněž nemalé množství hlavních vývojářů a podpora od běžných softwarových vývojářů přispívá ke světlé budoucnosti tohoto nástroje [14].

4.4 Komponenty

OpenSceneGraph obsahuje značné množství komponent, díky nimž je vysoce škálovatelný a schopný poskytnout rozšiřující funkcionalitu. Existují čtyři základní knihovny, které tvoří jádro OSG. Kromě nich je možné použít přídavné knihovny, kterým se říká *NodeKits*. Ty nabízí zejména specifičtější funkcionalitu, kterou mohou vývojáři vyžadovat (tato práce je také využívá).

OSG obsahuje čtyři základní knihovny [14]:

- *OpenThreads*: Poskytuje programátorům v jazyce C++ objektově orientované rozhraní pro implementaci vláken. Je to hlavní vláknový model pro OSG.
- *osg*: Obsahuje základní prostředky pro vytváření grafů scény (uzly, geometrické útvary, textury atd.). Pro možnost práce s vektory a maticemi dále obsahuje také nezbytné matematické třídy.
- *osgDB*: Nabízí přídavné moduly pro čtení a zapisování souborů s 2D a 3D grafikou a díky odvozeným třídám také obyčejných datových souborů a standardního vstupu a výstupu. V neposlední řadě umožňuje dynamické načítání částí grafu scény.
- *osgUtil*: Tato knihovna je určena pro podporu vykreslovací části OSG. Prochází graf scény, provádí ořezávání scény v každém snímku a převádí scénu v OSG na seznam příkazů OpenGL.

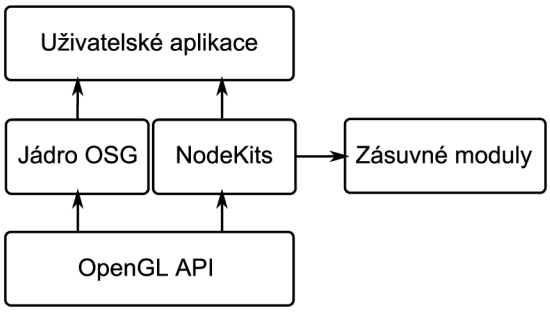
V současnosti jsou dostupné tyto přídavné knihovny *NodeKits* [14]:

- *osgAnimation*: Knihovna poskytuje nástroje pro vytváření animací. Používá generické šablony pro vytváření klíčových snímků a kanálů animací.
- *osgFX*: Nabízí prostředí pro tvorbu speciálních efektů v 3D prostoru.
- *osgGA*: Tato knihovna je prostředníkem mezi OSG a grafickým uživatelským rozhraním (GUI). Pracuje nad běžnými okenními systémy a dovoluje např. zpracovávat události vyvolané periferními zařízeními, jako je myš či klávesnice.

- *osgManipulator*: Rozšiřuje funkcionalitu o interaktivní manipulaci s 3D objekty. Jedná se o posunutí, rotaci a změnu měřítka transformačních uzlů.
- *osgParticle*: Podporuje vykreslování částicových efektů, výbuchů, ohně a kouře.
- *osgShadow*: Poskytuje prostředí pro práci se stíny.
- *osgSim*: Tato knihovna dodává do OSG nástroje pro různé druhy simulace.
- *osgTerrain*: Přidává podporu pro vykreslování terénů pomocí výškového modelu a obrazových dat.
- *osgText*: Pro vykreslování textů ve formě fontů, dostupných z projektu *FreeType*. Je schopná vytvořit 2D či 3D nápisy, a to opět buď ve 3D prostoru nebo na 2D obrazovce.
- *osgViewer*: Tato velmi důležitá knihovna poskytuje sadu tříd pro zobrazení grafů scén v grafických prostředích operačních systémů, jako je např. Win32, X11, Carbon či Cocoa.
- *osgVolume*: Přidává podporu pro techniky objemového vykreslování.
- *osgWidget*: Rozšiřuje funkcionalitu OSG o 2D ovládací prvky (*widgety*), které mohou být zasazeny do 3D aplikací.
- *osgQt*: Tato knihovna integruje do OSG grafů scény prostředí Qt a dovoluje tak zobrazit běžné grafické ovládací prvky, které nabízí právě Qt.

4.5 Architektura

Komponenty zmíněné v předchozí kapitole hrají významnou roli v architektuře OSG. Ta je zobrazena na obrázku 4.2. Nejspodnější vrstvu představuje aplikační programové rozhraní OpenGL, na kterém je OpenSceneGraph postaven. Jádro OSG pak tvoří již zmíněné čtyři základní knihovny (*OpenThreads*, *osg*, *osgDB* a *osgUtil*) a přídavné knihovny *NodeKits*. Tato část je pak doplněna zásuvnými moduly, které mohou přinášet podporu např. pro práci s více formáty souborů. Vrchní část architektury nakonec tvoří uživatelské aplikace [14].



Obrázek 4.2: Architektura nástroje OpenSceneGraph.

Kapitola 5

Návrh a implementace aplikace

Jak bylo zmíněno v úvodní kapitole 1, výstupem práce má být počítačový program představující hru šachů ve 3D. Jako hlavní cíl je pak možnost ovládání šachových figurek pomocí periferního zařízení Leap Motion (kapitola 3).

Před samotnou fází implementace této aplikace byla provedena analýza problému. Ta spočívala v rozpoznání stěžejních bodů vývoje, a tedy získání poznatků, co vše je potřeba udělat. Na základě těchto poznatků byl poté proveden návrh podoby a chování klíčových částí aplikace a následně byly tyto části implementovány. Pro samotnou implementaci byl zvolen programovací jazyk C++.

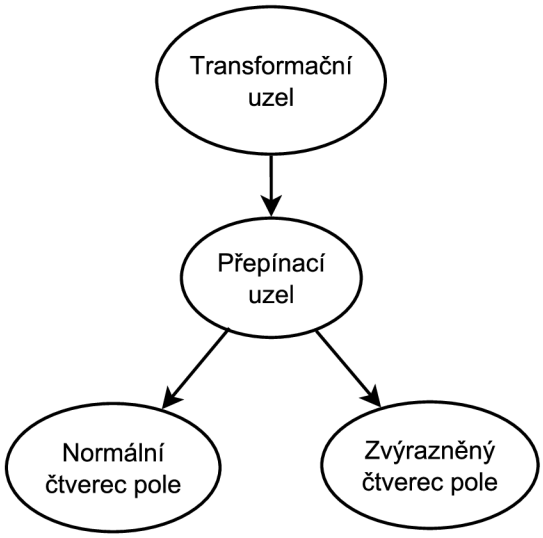
5.1 3D scéna

Jedním z prvních bodů práce bylo vytvoření 3D scény, která představuje hlavní prvek v grafickém uživatelském rozhraní aplikace. Ve scéně má být zobrazena šachová partie, tedy určitá podoba šachovnice a různé hrací figurky, to vše podle pravidel šachů [9]. 3D scéna musí být navíc schopna poskytovat odpovídající odezvu na vstup uživatele, tj. ovládání šachové partie ze vstupních zařízení počítače. Bylo rozhodnuto, že hra bude kromě zařízení Leap Motion ovladatelná také pomocí klávesnice. To má zajistit, že i v případě poruchy nebo výpadku ovladače Leap Motion lze aplikaci řídit i jinou formou.

5.1.1 Použití grafu scény

Jako prostředek pro vytvoření vizuální stránky 3D scény byl zvolen grafický nástroj OpenSceneGraph (více o tomto nástroji v kapitole 4). Všechny části scény jsou v tomto případě ukládány do grafu scény. Využity jsou jak listové uzly pro koncové grafické objekty (těmi jsou zejména modely figurek, čtverce polí a texty jako popisky šachovnice), tak skupinové uzly. Jedním z druhů skupinového uzlu je transformační uzel (třída `osg::MatrixTransform`). Ten dovoluje provádět geometrické transformace nad objekty, které jsou jeho dítětem. Lze tak jednoduše uskutečnit posunutí, rotaci či změnu měřítka nad všemi jeho dětmi zároveň. V grafu scény je dále hojně využit přepínací uzel (třída `osg::Switch`), který je schopen přepínat viditelnost svých dětí. Díky tomu je možné za běhu aplikace přepínat např. mezi obyčejným a zvýrazněným čtvercem pole (více o zvýrazňování polí v sekci 5.2.1). Část takového grafu scény je vidět na obrázku 5.1. V tomto grafu jsou spojeny obě výše zmíněné vlastnosti a lze je využívat za běhu programu.

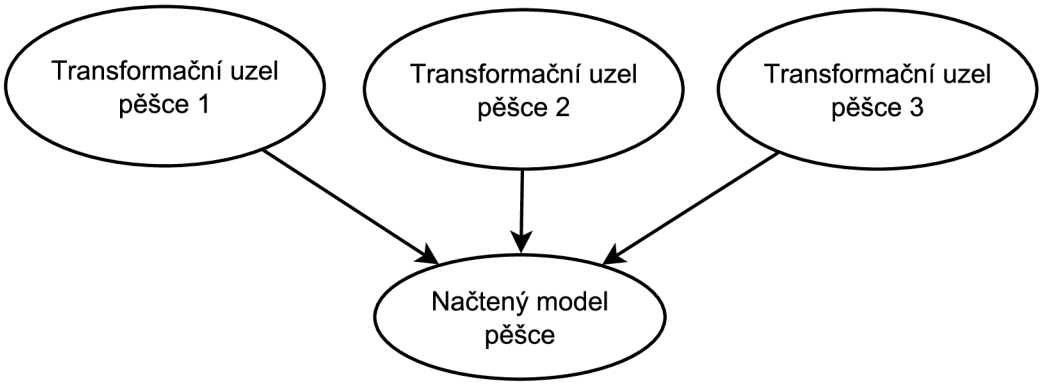
Tento postup je stejným způsobem použit i pro manipulaci s figurkami, které se musí během hry pohybovat a jsou také při uchopení příslušně označeny.



Obrázek 5.1: Část grafu scény manipulující se čtvercem pole.

5.1.2 Sdílení grafických objektů

O možnosti sdílení grafických objektů v grafu scény již bylo pojednáno v kapitole 4.2. Na obrázku 5.2 je vidět konkrétní použití této možnosti v implementaci aplikace. Jde o sdílení modelu jednoho druhu šachové figurky, pěšce. Uzel s modelem je napojen na několik rodičů, transformačních uzlů, které jakoby duplikují pohled na tento model. Ve výsledku je tak figurka ve scéně vidět několikrát a díky transformačním uzlům lze s každou instancí modelu pracovat nezávisle. Hlavní výhodou tohoto přístupu je pak značná úspora operační paměti při běhu programu.

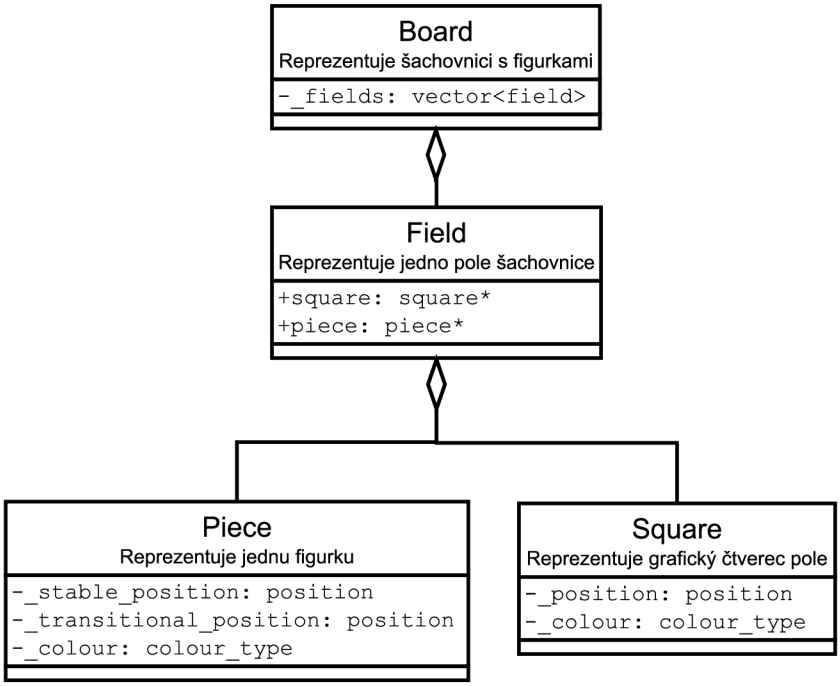


Obrázek 5.2: Část grafu scény demonstrující sdílení modelu.

5.1.3 Šachovnice

Objektový návrh Z hlediska objektového návrhu je celá scéna, tedy lze říci šachovnice s figurkami, reprezentována třídou `Board`. Ta v sobě ukládá všechna šachovnicová pole (vždy je jich 64, tedy 8krát 8), která jsou představována třídou `Field`. Každé pole pak obsahuje odkaz na čtverec reprezentující toto pole (třída `Square`) a na figurku, která toto pole zrovna okupuje (třída `Piece`). Jelikož ne na každém poli je umístěna figurka, může tento odkaz

nabývat hodnoty `NULL`, tedy že nikam neukazuje. Zmíněná struktura tříd je zobrazena na obrázku 5.3, jedná se o zjednodušený diagram.



Obrázek 5.3: Zjednodušený diagram tříd tvořících šachovnici.

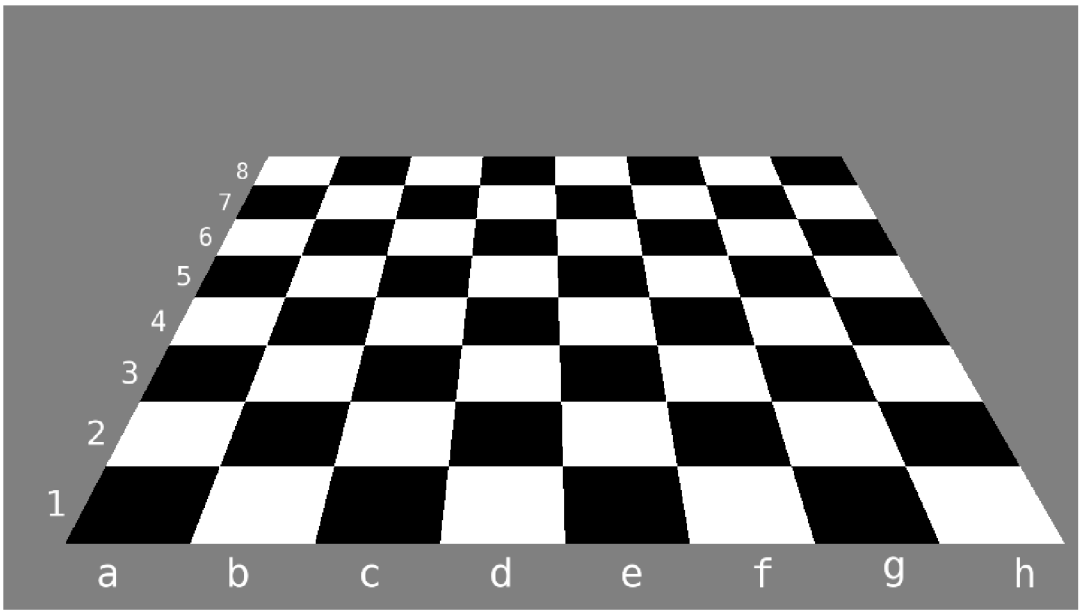
Grafická podoba Šachovnice, jež je umístěna ve scéně, je složena z 64 bílých a černých polí (čtverec 8krát 8), která se střídají, přičemž v levém spodním rohu je pole barvy černé. Každé pole je reprezentováno samostatným geometrickým útvarem, čtvercem. Za účelem správného umístění je pole přesunuto na svou pozici pomocí základní operace posunutí (počátek souřadného systému je ve středu šachovnice). Z výše uvedeného vyplývá, že šachovnice je de facto 2D objekt. Aby vypadala jako objekt v trojrozměrném prostředí, je pohledová matice kamery (*view matrix*) nastavena tak, aby se uživatel díval na šachovnici z přirozeného pohledu, tedy jako by ji měl položenu před sebou. Na obrázku 5.4 je vidět výstup tohoto návrhu.

Jelikož se při hraní šachů střídá bílá a černá strana a aplikace nabízí jenom jeden pohled v jednom časovém okamžiku, bylo rozhodnuto, že se šachovnice bude po každém odehraném tahu otáčet pro hráče druhé barvy. To se děje pomocí animace rotačního pohybu celé šachovnice.

Nedílnou součástí šachovnice jsou také popisky podél jejích hran. Přední a zadní hrana je označena písmeny *a* až *h*, levá a pravá pak číslicemi od *1* do *8*. To usnadňuje orientaci hráče na šachovnici a formální zápis průběhu partie. Popisky jsou navíc nastaveny takovým způsobem, že při otáčení šachovnice dochází k jejich pootáčení směrem ke kameře. Díky tomu se zamezí zrcadlovému zobrazení popisek, což by mohlo být pro uživatele matoucí.

5.1.4 Hrací figurky

Na šachovnici jsou rozmístěny podle pravidel hrací figurky [9]. Ty jsou několika druhů a liší se možnostmi svých pohybů. Lze je stručně shrnout do následujícího seznamu:



Obrázek 5.4: Návrh šachovnice v nástroji OpenSceneGraph.

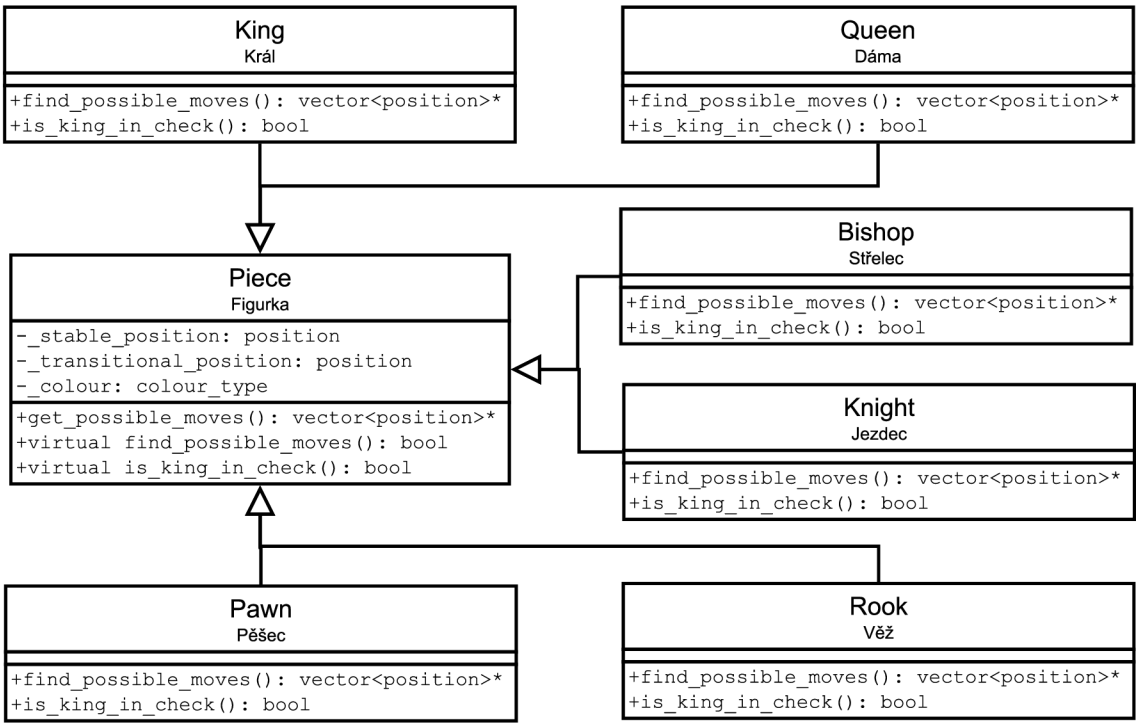
- Král: Může se přesouvat o jedno pole jakýmkoliv směrem. V případě, že je ohrožen figurkou druhé barvy, je v tzv. šachu. Pokud z tohoto ohrožení nelze „utéct“, hra končí vítězstvím druhé barvy.
- Dáma: Smí se pohybovat po řádcích, sloupcích a diagonálách.
- Střelec: Smí se přesouvat po diagonálách.
- Jezdec: Může skákat ve tvaru písmena „L“.
- Věž: Může se pohybovat po řádcích a sloupcích.
- Pěšec: Svým prvním tahem smí poskočit o dvě pole dopředu, jinak jenom o jedno. Brát může o jedno pole dopředu po diagonále.

Ve výčtu byly zanedbány některé detaily pravidel a kromě základních tahů existuje ještě několik speciálních, např. *braní mimochodem* (u pěšců), *rošáda* (král a věž) a *proměna pěšce*. Jelikož jsou pravidla šachů víceméně všeobecně známou záležitostí, nebudou zde ohledně teorie pravidel uvedeny další podrobnosti.

Objektový návrh Jak už bylo zmíněno v sekci o objektovém návrhu šachovnice (5.1.3), pro objektovou reprezentaci hrací figurky existuje třída `Piece`, která mimo jiné uchovává údaje o stabilní a přechodové poloze figurky. Přechodová poloha znamená pozici, na které se grafická podoba figurky v daný moment nachází, kdežto stabilní poloha je pozice figurky ve validním ustáleném postavení před započetím tahu.

Z důvodu existence již zmíněných několika druhů figurek bylo nutné navrhnout způsob, jak zařídit dodržování pravidel šachů, která jsou specifická pro každý jednotlivý typ. Na obrázku 5.5 je zjednodušeně vyobrazen návrh řešení tohoto požadavku. Existuje jedna základní třída `Piece` reprezentující figurku a z této třídy jsou pomocí dědičnosti odvozeny třídy pro každý druh figurky. Základní třída obsahuje virtuální metody pro zjištění všech

možných tahů dané figurky a jestli daná figurka ohrožuje krále opačné barvy. Tyto virtuální metody jsou pak implementovány v každé odvozené třídě a ve výsledku stačí nad libovolnou figurkou zavolat stejnou metodu a navráceny jsou možné tahy pro konkrétní figurku, to vše při zachování specifického chování každého typu figurky. Je tak využita jedna ze základních výhod objektového návrhu, a to polymorfismus.



Obrázek 5.5: Diagram tříd vyjadřující hierarchii dědičnosti pro specifické typy figurek.

Grafická podoba Za účelem zobrazení figurek na šachovnici bylo potřeba získat grafický 3D model každého druhu figurky. V aplikaci jsou použity modely stažené z webové stránky *TF3DM*[1]. Pro potřeby aplikace byly pozměněny barvy textur modelů tak, aby lépe odpovídaly bílé a černé barvě.

Při ovládání šachové partie je důležitá schopnost figurek pohybovat se odpovídajícím způsobem po šachovnici, případně zvednout se při uchopení a položit se při dokončení tahu. Toho je dosaženo pomocí geometrických transformací nad geometrickými objekty, v případě použití OpenSceneGraph pomocí transformačních uzlů nad uzly reprezentujícími modely figurek.

5.1.5 Využití nástroje OpenSceneGraph

Grafický nástroj OpenSceneGraph byl použit ve verzi 3.0.1. Jak je psáno v kapitole 4.4, OSG obsahuje množství komponent v podobě externích knihoven. Při implementaci aplikace byly využity následující knihovny OSG:

- *osg*: Nejobecnější knihovna byla použita pro vytváření geometrických transformací,

[1]Zdroj 3D modelů figurek: `http://tf3dm.com/3d-model/another-chess-60360.html`

textů ve scéně, zdrojů světla, geometrických útvarů, přepínacích uzlů a dalších základních elementů.

- *osgDB*: Využita pro načtení modelů ze souborového systému.
- *osgUtil*: Knihovna použita pro celkovou optimalizaci grafu scény.
- *osgFX*: Tato knihovna použita pro vytvoření efektu vyznačení siluety figurky.
- *osgGA*: Využita pro zpracovávání externích událostí z periferního zařízení (klávesnice).
- *osgText*: Knihovna použita pro vytvoření popisků k okrajům šachovnice.
- *osgViewer*: Tato knihovna využita pro vytvoření prohlížeče 3D scény šachové partie.
- *osgQt*: Použita pro integraci s prostředím Qt, vytvoření grafického okna Qt.

5.2 Ovládání šachové partie

V kapitole 5.1 již bylo zmíněno, že partii je možné ovládat dvěma způsoby, klávesnicí a zařízením Leap Motion. Tato práce má primárně za cíl co nejlépe využít schopnosti Leap Motion a vyhodnotit použitelnost takového uživatelského rozhraní.

5.2.1 Zvýraznění polí šachovnice

Základním prvkem pro zpětnou vazbu uživateli při ovládání aplikace je zvýrazňování polí šachovnice. To se mírně liší pro ovládání klávesnicí a pomocí Leap Motion. Rozdíly budou popsány v následujících sekcích. Mají ale také společné chování. Při zaměřování pole na šachovnici se odpovídající čtverec zbarví do žluta a při zvednutí figurky pak do zelena (kromě toho se figurka fyzicky posune směrem vzhůru nad šachovnici a také se zvýrazní). Uživatel tak vždy může lehce poznat, nad kterým polem se aktuálně nachází a jestli je konkrétní figurka vybrána.

5.2.2 Ovládání pomocí klávesnice

Lze říci, že ovládání pomocí klávesnice je u této aplikace sekundárním typem interakce. Ve fázi návrhu bylo rozhodnuto, že pohyb po šachovnici, ať už s figurkou či bez ní, bude uskutečňován kurzorovými šipkami, protože se dá mimo jiné očekávat, že uživatel bude jako první klávesy zkoušet právě tyto. Zvednutí a položení figurky se dá provést klávesou *Enter*, což by také mělo být pro uživatele poměrně intuitivní.

Co se týče zvýrazňování polí (kapitola 5.2.1), na začátku hry je implicitně zvoleno jedno pole a hráč se může posouvat po šachovnici na libovolná pole pomocí kurzorových šipek. Aktuální pole je vždy zvýrazněno.

Při pokusu uživatele o upuštění figurky na nedovolené pole se tah stornuje a hráč musí táhnout znovu.

5.2.3 Ovládání pomocí Leap Motion

Jak se píše v oficiálních instrukcích pro vývojáře využívající Leap Motion [5], každá taková aplikace by měla zajišťovat vhodnou a okamžitou odezvu uživateli, aby v každém momentu

viděl, jak se projevuje to či ono gesto. To má mimo jiné zamezit případné narůstající frustraci z ovládání aplikace v případě, že by často docházelo k nežádoucí interpretaci gest.

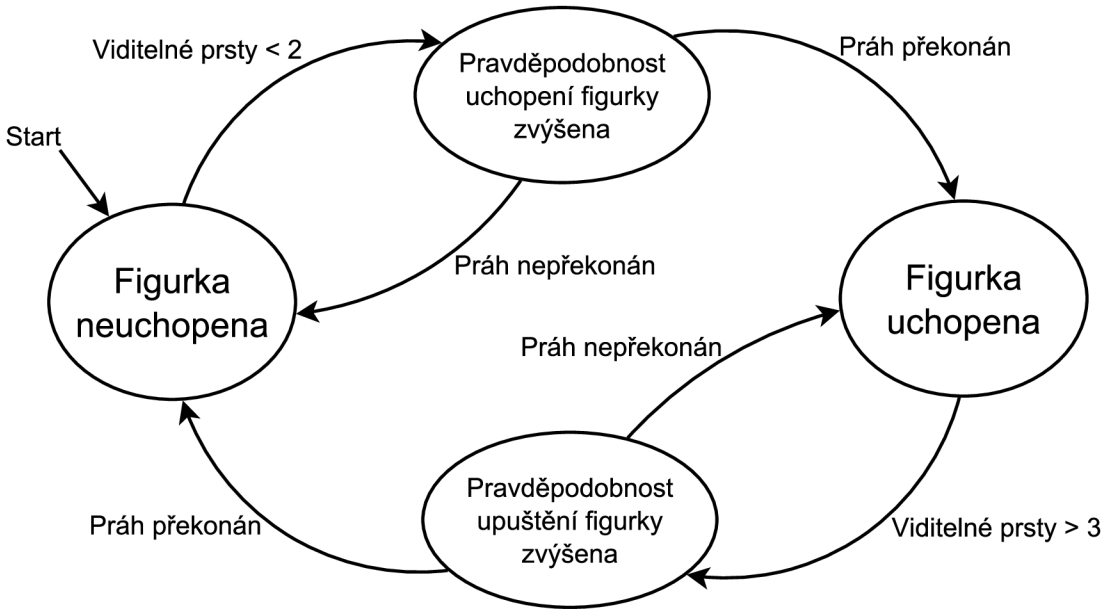
Na rozdíl od ovládání klávesnicí se při pohybu nad šachovnicí zvýrazňují jenom ta pole, na kterých stojí figurky barvy, která je na tahu. Když má tedy uživatel ruku nad polem, kde toto neplatí, tak je označeno nejbližší takové pole, kde je podmínka splněna. Podobné chování pak nastává při zvednuté figurce. Při upuštění figurky se pak figurka položí na nejbližší povolené pole.

Kromě zvýrazňování polí nabízí tento způsob ovládání také přesný kurzor, který udává, kde nad šachovnicí se ruka hráče nachází (pozice je brána z údaje o poloze dlaně z datového rámce Leap Motion). Kurzor má podobu částečně průhledného červeného kruhu a velmi usnadňuje orientaci uživatele ve hře. Pokud zařízení Leap Motion ztratí ruku hráče ze svého dohledu, kurzor zmizí, případná zvednutá figurka je okamžitě položena na původní pole a tah je stornován.

Za účelem integrace ovládání pomocí senzoru Leap Motion bylo použito aplikační programové rozhraní poskytované výrobcem tohoto zařízení ve verzi 1.0.9.

Rozpoznávání gest Efektivní rozpoznávání gest uživatele je jednou ze stěžejních částí práce. Bylo potřeba zavést dvě gesta: pro uchopení figurky a upuštění figurky. Bylo shledáno, že API pro Leap Motion (kapitola 3.6) nenabízí žádná vhodná gesta pro tento účel. Musely být tedy využity základní prostředky Leap Motion API, které aplikaci posílá proud datových rámců s informacemi o snímaných objektech.

Od uživatele je vyžadováno, aby měl při pohybu nad šachovnicí prsty ruky rozevřené, díky čemuž senzor Leap Motion může detekovat tyto prsty. Předchozí požadavek má jednoduchý důvod. Gesto pro uchopení figurky je totiž určeno jako sevření ruky v pěst. V tom případě Leap Motion přestane prsty u rukou „vidět“ a gesto tak může být rozpoznáno. Gesto upuštění figurky je pak definováno jako inverzní k předchozímu, tedy rozevření dlaně a prstů ruky.



Obrázek 5.6: Konečný automat reprezentující proces rozpoznávání gest.

Na obrázku 5.6 je schéma konečného automatu, který byl navržen pro běh rozpoznávače

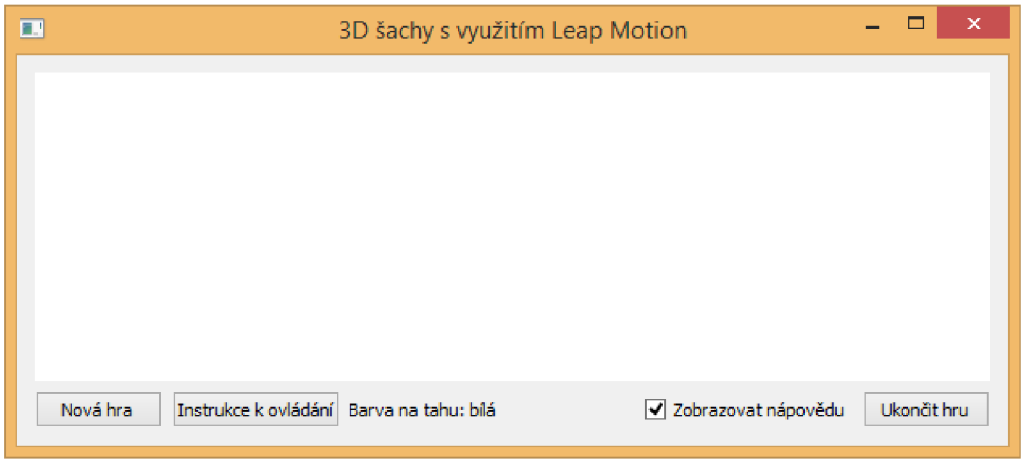
zmíněných gest. Jsou zde dva hlavní stavy. Po spuštění je automat ve stavu, kdy není uchopena žádná figurka. Druhý hlavní stav pak vyjadřuje uchopenou figurku. Pokud je v zorném poli senzoru Leap Motion ruka s méně jak dvěma viditelnými prsty (jeden viditelný prst je tedy tolerován), a to po určitou stabilizační dobu (na obrázku jako *práh*), je vyhodnoceno gesto uchopení figurky a jsou provedeny odpovídající akce. Opačným směrem automat funguje analogickým způsobem. Opět se bere v úvahu tolerance (více jak čtyři viditelné prsty), protože senzor často posílá nepřesné informace o počtu prstů, nejčastěji právě o jeden.

5.2.4 Systém nápovědy

Lze předpokládat, že někteří uživatelé umí pravidla šachů lépe a jiní ne tak dobře. I proto byl v aplikaci vytvořen systém nápovědy, který při uchopení figurky zvýrazní všechna pole, na která lze s danou figurkou táhnout. Nápověda je implicitně zapnutá, přičemž uživatel ji může kdykoliv pomocí zaškrtávacího políčka zakázat, příp. opětovně povolit.

5.3 Grafické uživatelské rozhraní

Poté, co byla vytvořena 3D scéna pomocí nástroje OpenSceneGraph, přišla řada na vytvoření grafického uživatelského rozhraní (dále také GUI). Bylo nejprve nutné zvolit vhodný prostředek, který nabízí dostatečnou podporu pro tento účel. Vzhledem k tomu, že OpenSceneGraph obsahuje knihovnu *osgQt* (kapitola 4.4), která integruje do OSG grafů scény prostředí Qt, byl vybrán právě tento nástroj, a to ve verzi 4.8.5.

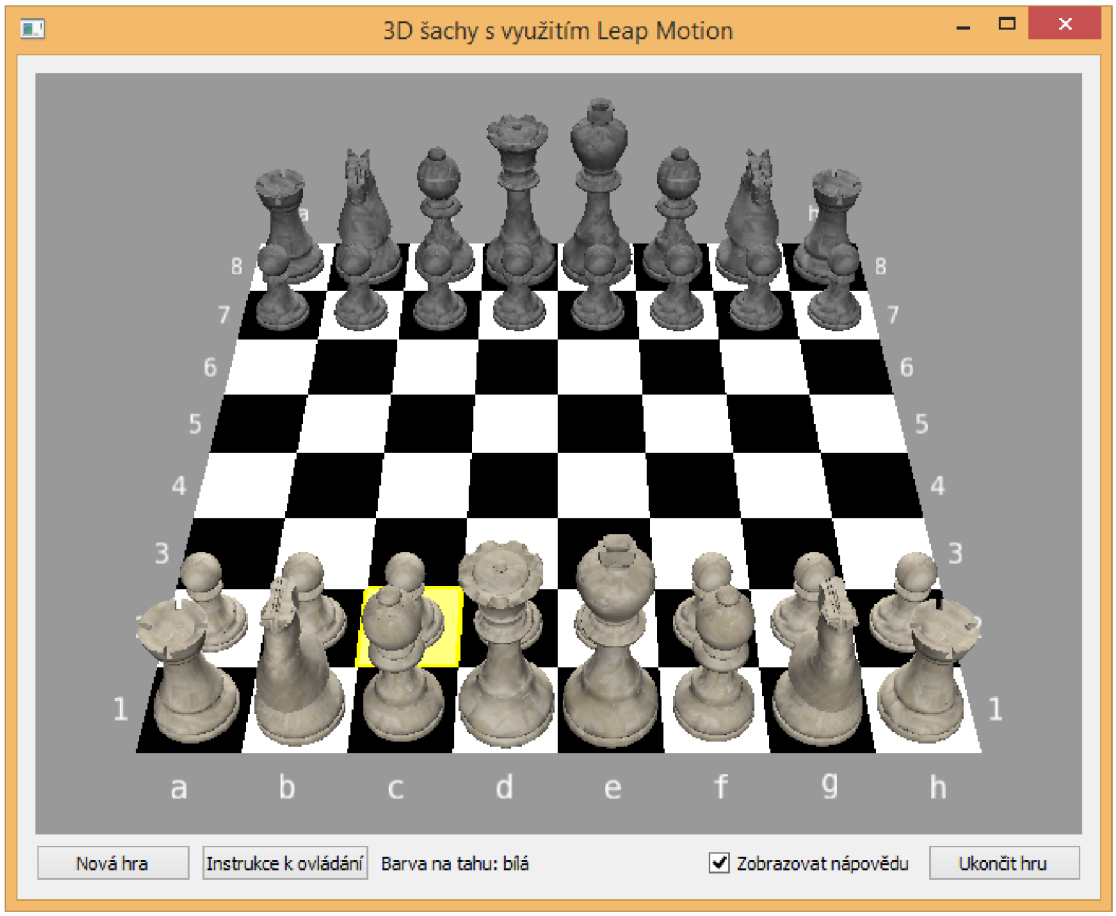


Obrázek 5.7: Návrh grafického uživatelského rozhraní.

Na obrázku 5.7 je návrh GUI v prostředí Qt. Bílá oblast vyznačuje místo, kde je ve výsledné aplikaci umístěna 3D scéna. Zaujímá tedy nejvíce prostoru a je ve středu okna. Dále rozhraní obsahuje několik tlačítek. Pro spuštění nové hry (první hra je spuštěna automaticky po zapnutí aplikace), ukončení hry (okno aplikace se zavře) a pro zobrazení instrukcí k ovládání aplikace. Nalevo od tlačítka „Ukončit hru“ je pak již zmíněné zaškrtávací políčko, které slouží k zapnutí/vypnutí nápovědy možných tahů hráče. Nakonec je zde ukazatel barvy, která je právě na řadě, a ukazatel, který se objeví v případě ohrožení některého z králů (není na obrázku vidět).

Výsledná podoba aplikace je pak vidět na obrázku 5.8. V centrální oblasti je umístěna scéna šachové partie. GUI je nastaveno takovým způsobem, že se v případě zvět-

šení/zmenšení okna automaticky přizpůsobuje i velikost scény a všech dalších ovládacích prvků. Mimo to se hned po spuštění aplikace založí nová hra a figurky jsou rozmístěny do počátečních pozic.



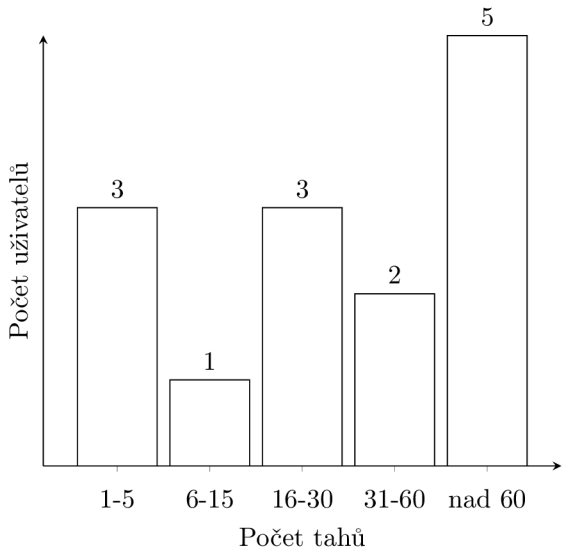
Obrázek 5.8: Výsledná podoba aplikace.

Kapitola 6

Vyhodnocení ovládání aplikace

Ovládání aplikace, přesněji samotné šachové partie, pomocí senzoru Leap Motion je stěžejní částí této práce. Proto bylo po dokončení aplikace provedeno testování použitelnosti rozhraní a na základě jeho výsledků pak vyhodnocení.

Testování spočívalo v oslovení 14 respondentů, kteří bez předchozích zkušeností s ovládáním pomocí Leap Motion usedli k této aplikaci. Nejprve si každý uživatel přečetl instrukce k ovládání, které jsou dostupné skrze jedno tlačítko v GUI. Poté již začal hrát samotnou hru, a to za obě strany, a zkoušel provést několik tahů. Během tohoto bylo měřeno, kolik tahů zhruba potřebuje uživatel udělat, aby získal jistotu v ovládání. Dále byly respondentům kladeny dotazy obecnějšího charakteru. Zejména jaké mají z ovládání pocity, jakou cítí jistotu při ovládání, jestli v tomto typu ovládání vidí nějaký smysl či budoucnost a zda-li si dokáží představit, že by takto běžně ovládali tuto hru anebo by dali přednost klávesnici.



Obrázek 6.1: Histogram vyjadřující počet tahů potřebných k získání jistoty v ovládání.

Na obrázku 6.1 je histogram, který vyjadřuje počet provedených tahů potřebných k získání jistoty v ovládání aplikace. Nejvíce uživatelů potřebovalo pro tento účel odehrát celou partii (v grafu vyznačeno „nad 60“). Většina lidí z této skupiny byla starších 40 let a nebyla tolik sžitá s dnešními technologiemi, tudíž je tento výsledek pochopitelný, ale i tak něco vypovídá zejména o mírné nepřesnosti ovládání. Na druhou stranu 3 respondenti získali jis-

totu velmi rychle (během prvních pěti tahů) a hraní jim nečinilo žádný problém. Stejnému počtu uživatelů to pak trvalo kolem 20 tahů. Zbylé intervaly počtu tahů jsou v menšině. Bylo shledáno, že velký vliv na schopnost uživatelů ovládat aplikaci pomocí Leap Motion měl jejich věk a s tím spojené sžití s aktuálními technologickými trendy. Respondenti okolo 20 let neměli žádný výraznější problém se rychle přizpůsobit a naučit se provádět potřebná gesta.

Jedním z problémů při ovládání bylo množství tahů provedených omylem. To však nejčastěji bylo způsobeno tím, že uživatel po dokončení tahu nechal svou ruku v zorném poli senzoru a sevřel prsty takovým způsobem, že to bylo vyhodnoceno jako gesto zvednutí figurky. Po ukončení tahu je tedy nutné, aby uživatel dával pozor a měl prsty ruky roztažené na znamení, že nechce momentálně žádnou figurku zvolit.

Několik uživatelů si stěžovalo na to, že se musí na ovládání pořád velmi soustředit a také je začíná po několika tazích bolet ruka, tudíž by radši volili klávesnici, která by podle nich byla pohodlnější a jistější a nemuseli by se bát, že provedou nechtěný tah. Na druhou stranu přiznávali, že je to zřejmě záležitost zvyku, stejně jako si kdysi zvykali na počítačovou myš.

Nicméně většina respondentů uvedla, že jim způsob interakce přijde velmi zajímavý a dokáží si představit dobrou budoucí použitelnost takového ovládání aplikací.

Co uživatelé velmi oceňovali bylo to, že je aplikace sama navádí pouze na povolená pole (v případě výběru figurky pouze na ta pole, kde mají svou figurku, a v případě přesunu figurky pouze na pole, kam lze provést validní tah), což celý proces tahu velmi usnadňuje.

V otázce smysluplnosti takového typu ovládání hry se názory respondentů rozcházely. Mírná většina uvedla, že jim tento způsob interakce přijde velmi zajímavý a dokáží si představit dobrou budoucí použitelnost takového ovládání aplikací. Ostatní pak byli více skeptičtí.

Kapitola 7

Závěr

Jako výstup práce byla vytvořena aplikace nabízející hraní 3D šachů s využitím pohybového senzoru Leap Motion, čímž byl splněn hlavní cíl této práce.

Vývoj aplikace byl rozdělen do několika fází s ohledem na zadání bakalářské práce. Nejprve bylo nastudováno aplikační programové rozhraní (API) pro Leap Motion, aby mohlo být efektivně využito toto zařízení. Následně byla nastudována pravidla hry šachy, neboť jsou jejich nedílnou součástí a musela být správně implementována. Dále byl zvolen a prostudován grafický nástroj OpenSceneGraph, pomocí kterého bylo navrženo a vytvořeno 3D prostředí hry. Před samotnou implementací aplikace pak byl navrhnut způsob ovládání pomocí Leap Motion a rozpoznávání gest uživatele.

Po dokončení implementace bylo uskutečněno testování aplikace na 14 uživatelích a na základě jejich odezvy provedeno vyhodnocení uživatelského rozhraní. Z něj mimo jiné vyplývá, že ovládání aplikace je docela použitelné, avšak jej místy sráží problémy jako potřeba více se soustředit či únava ruky. Bylo také shledáno, že jsou mezi zkušenostmi uživatelů s novými technologickými trendy velké rozdíly, a tak někteří získali jistotu při ovládání velmi rychle v řádu jednotek tahů, zatímco nezanedbatelná část potřebovala za tímto účelem odehrát celou šachovou partii.

K aplikaci byla napsána příručka k jejímu ovládání a také bylo vytvořeno krátké video, které prezentuje způsob hraní šachů v této aplikaci.

Z pohledu budoucnosti by funkcionalita aplikace mohla být rozšířena (např. síťová hra, lepší grafická stránka) a pokud by splňovala podmínky webové obchodu Airspace s aplikacemi pro Leap Motion, bylo by vhodné ji zde umístit. Z testování aplikace vyplývá, že by však bylo nutné zpřesnit rozpoznávání gest, aby mohla být aplikace úspěšná u široké veřejnosti.

Literatura

[1] Boxall, A. *Leap Motion's 3D gesture controls coming to smartphones and tablets in 2014* [online]. 2013 [cit. 1. května 2014]. Dostupné na: <http://www.digitaltrends.com/mobile/leap-motion-coming-smartphones-tablets-2014>.

[2] Etherington, D. *Leap Motion Controller Ship Date Delayed Until July 22, Due To A Need For A Larger, Longer Beta Test* [online]. 2013 [cit. 1. května 2014]. Dostupné na: <http://techcrunch.com/2013/04/25/leap-motion-controller-ship-date-delayed-until-july-22-due-to-a-need-for-a-larger-longer-beta-test/>.

[3] Greenwald, W. *Kinectrospective: A Brief History of Controller-Free Gaming* [online]. 2010 [cit. 1. května 2014]. Dostupné na: <http://www.pcmag.com/article2/0,2817,2372058,00.asp>.

[4] Guna, J., Jakus, G., Pogačnik, M. et al. An Analysis of the Precision and Reliability of the Leap Motion Sensor and Its Suitability for Static and Dynamic Tracking. *Sensors*. 2014, roč. 14, č. 2. S. 3702–3720. Dostupné na: <http://www.mdpi.com/1424-8220/14/2/3702>. ISSN 1424-8220.

[5] Leap Motion. *Official website* [online]. 2014 [cit. 1. května 2014]. Dostupné na: <https://www.leapmotion.com/>.

[6] Li, Y. Hand gesture recognition using Kinect. In *Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on*. June 2012. S. 196–199.

[7] Maliarov, M. *Leap Motion: ovládání gesty začíná dávat smysl* [online]. 2013 [cit. 1. května 2014]. Dostupné na: <http://www.mobilmania.cz/clanky/leap-motion-ovladani-gesty-zacina-davat-smysl/sc-3-a-1323502/default.aspx>.

[8] Price, E. *Leap Motion Launches Its Motion-Controlled App Store* [online]. 2013 [cit. 1. května 2014]. Dostupné na: <http://mashable.com/2013/06/24/leap-motion-airspace/>.

[9] Schiller, E. *Official Rules of Chess*. 2. vyd. New York: Cardoza, 2003. 96 s. ISBN 1-58042-092-3.

[10] Statt, N. *How Leap Motion tracks what it can't see* [online]. 2013 [cit. 1. května 2014]. Dostupné na: <http://www.cnet.com/news/how-leap-motion-tracks-what-it-cant-see/>.

[11] Terdiman, D. *Leap Motion giving 10,000 developers free Leaps* [online]. 2012 [cit. 1. května 2014]. Dostupné na: <http://www.cnet.com/news/leap-motion-giving-10000-developers-free-leaps/>.

[12] Tsotsis, A. *OcuSpec Raises 1.3M From Andreessen And Others To Build An Affordable Kinect* [online]. 2011 [cit. 1. května 2014]. Dostupné na: <http://techcrunch.com/2011/06/10/ocuspec-raises-1-3m-from-andreessen-and-others-to-build-an-affordable-kinect/>.

[13] Uhlíř, F. *Design počítačové myši*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2006. 39 s. Bakalářská práce.

[14] Wang, R. a Qian, X. *OpenSceneGraph 3.0: Beginner's Guide*. 1. vyd. Birmingham: Packt Publishing, 2010. 412 s. ISBN 978-1-849512-82-4.

[15] Weichert, F., Bachmann, D., Rudak, B. et al. Analysis of the Accuracy and Robustness of the Leap Motion Controller. *Sensors*. 2013, roč. 13, č. 5. S. 6380–6393. Dostupné na: <http://www.mdpi.com/1424-8220/13/5/6380>. ISSN 1424-8220.

Příloha A

Příručka k ovládání aplikace

Tato příručka vysvětluje možnosti ovládání šachové partie ve vytvořené aplikaci. Šachové figurky lze ovládat dvěma způsoby:

1. Pomocí pohybového senzoru Leap Motion.
2. Pomocí klávesnice.

Leap Motion Pohybem ruky s nataženými prsty v prostoru nad zařízením se dá pohybovat nad šachovnicí a zaměřovat cílovou figurku, přičemž všechny prsty musí zůstat nataženy (obrázek A.2), jinak by mohlo být detekováno gesto uchopení figurky, která se nachází nejblíže. Pro dobrou orientaci je potřeba sledovat červený kurzor.

Uchopení figurky se provede sevřením ruky v pěst (obrázek A.1). Úspěšné uchopení je signalizováno pohybem figurky vzhůru a jejím označením zelenou barvou. Při pohybu s figurkou musí ruka zůstat po celou dobu sevřená, jako by v ní byla figurka držena. V případě pohybu ruky mimo zorné pole Leap Motion se tah stornuje a uživatel dostane novou možnost znovu táhnout.

Položení figurky na cílové místo je možné rozevřením ruky a natažením všech prstů, jak je vidět na obrázku A.2. Tah se dokončí a šachovnice se otočí pro druhého hráče.

Klávesnice Pohyb po šachovnici je uskutečňován směrovými šipkami. Pro uchopení i položení figurky se používá klávesa Enter.

Obrázek A.1: Uchopení figurky

Obrázek A.2: Položení figurky