ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
Katedra kybernetiky

DIPLOMOVÁ PRÁCE

Využití senzoru Kinect pro komerční prezentaci produktů

2013

Vypracoval: Bc. Pavel Bártek
Vedoucí práce: Ing. Jiří Hlinka

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne:10.5.2013 .....

Podpis autora práce

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Bc. Pavel Bártek

Studijní program: Otevřená informatika (magisterský)

Obor: Počítačové vidění a digitální obraz

Název tématu: Využití senzoru Kinect pro komerční prezentaci produktů

Pokyny pro vypracování:

Vytvořte systém pro bezdotykové prohlížení a nákup produktů. Aplikace bude detekovat přítomnost osoby a reagovat na její pohyby rukou. Po vykonání předem definovaných gest provede aplikace příslušné akce. K ovládání bude využito snímání obrazu senzorem Microsoft Kinect. Gesta bude aplikace identifikovat z obrazových dat poskytovaných senzorem s využitím původní metody navržené studentem. Systém bude umožňovat zákazníkům procházet katalog produktů, zobrazovat detaily a varianty, a vytvářet nákupní košík s vybranými výrobky.

1. Proveďte rešerši podobných aplikací a kriticky je zhodnoťte.
2. Funkčnost vytvořené aplikace vhodným způsobem demonstrujte a diskutujte dosažené vlastnosti včetně vyhodnocení úspěšnosti rozpoznání gest. Vyzvedněte vlastní přínos, zejména v identifikaci gest. Proveďte srovnání s metodami, které standardně nabízí SDK.
3. Použité metody a vytvořenou aplikaci zdokumentujte standardním způsobem.
4. Jako implementační jazyk použijte C#.

Seznam odborné literatury:

Davison, Andrew: Kinect Open Source Programming Secrets - 27th September 2012, http://fivedots.coe.psu.ac.th/~ad/kinect/index.html

Vedoucí diplomové práce: Ing. Jiří Hlinka

Platnost zadání: do konce letního semestru 2012/2013

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

L.S.

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 5. 11. 2012

Anotace

Tato práce se zaměřuje na použití senzoru Kinect v praktické aplikaci interaktivní nabídky zboží. Snímá uživatele a umožňuje mu procházet nabídku položek. Jako implementační jazyk byl zvolen C# a je použito Kinect for Windows SDK. Není využit Skeleton Stream z SDK, detekce uživatele je prováděna vlastním algoritmem.

Summary

This thesis focuses on using Kinect in the practical application of interactive shopping. It scan the user and allows him to navigate the menu items. C# was choosen as the implementation language and the Kinect for Windows SDK is used. Application does not use Skeleton Stream from SDK but use its own algorithm.

Obsah

1 Úvod **1**

2 Senzor Microsoft Kinect **2**

2.1 Firma Microsoft 2

2.2 Microsoft na konzolovém trhu 3

2.3 Kinect 4

2.4 Hardware Kinektu 5

2.4.1 Barevná kamera 6

2.4.2 IR projektor a IR kamera 6

2.4.3 Mikrofonní pole 9

2.4.4 Ovládání náklonu 9

3 Kinect API **10**

3.1 OpenKinect 10

3.2 CL NUI 10

3.3 OpenNI 11

3.4 Microsoft's Kinect for Windows SDK 11

3.4.1 Licence 11

3.4.2 Použití Kinect For Windows SDK 12

3.4.3 Color Stream 12

3.4.4 Depth Stream 13

3.4.5 Skeleton Stream 13

3.4.6 Použití více senzorů najednou 16

3.4.7 Kalibrace senzoru 17

3.4.8 Kalibrace tohoto konkrétního senzoru 20

4 Použití Kinectu **23**
4.1 Digitální výloha 23
4.1.1 Obdobné systémy 24

5 Navržené řešení **26**
5.1 Databáze 27
5.2 Administrativní aplikace 28
5.3 Prezentační aplikace 30
5.3.1 Součásti aplikace 31
5.3.2 Třída Kinect Control Class 31
5.3.3 Hlavní okno 32
5.3.4 Třída DatabaseConnection 32
5.3.5 Třída Nastavení 33
5.3.6 Třída Setting Windows 34
5.3.7 Sledování ruky 34
5.4 Ovládání 36
5.5 Porovnání s SDK 38
5.6 Vylepšení 38

6 Závěr **40**

Seznam obrázků **41**

Literatura **42**

Kapitola 1

Úvod

Interakce mezi člověkem a počítačem se postupně s rozvojem výpočetní techniky značně proměnila, a děrné štítky byly nahrazeny klávesnicí a myší tak, jak je známe dnes. Tento vývoj se ovšem u současných zařízení nezastavil a jedna z revolučních technologií přišla v podobě senzoru Microsoft Kinect. Ten získal ihned po svém uvedení obrovskou popularitu a je zapsán v Guinessově knize rekordů jako nejprodávanější elektronické zařízení vůbec. Kinect, původně určený k ovládání her na konzoli Xbox 360, je na první pohled odlišný od dnešních vstupních zařízení. Místo tlačítek obsahuje kamerový systém, který umožňuje rozpoznat polohu částí těla uživatele a využít ji k ovládání počítače. Otevírá tak novou kapitolu v interakci člověk-počítač (HCI) tím, že umožňuje samostatně ovládat počítač bez použití myši, klávesnice a dotyku vůbec. Výstižně a s nadšením to popisuje citát James L. McQuiveyho ze společnosti Forrester: *"Kinect is to the next decade what the operating system was to the 1980s, what the mouse was to the 1990s, and what the Internet has been ever since. It is the thing that will change everything"*. Předznamenává příchod éry, ve které budou lidé moci ovládat libovolné přístroje v domácnosti nebo na pracovišti jen pohybem ruky nebo jiné části těla, bez nutnosti nacházet se v jejich bezprostřední blízkosti a používat speciální hardware.

Cílem této práce je využít možnosti senzoru Kinect a vytvořit systém pro komerční prezentaci produktů. Tento systém rozpozná uživatele a umožní mu procházet nabídku zboží podobně, jako je zvyklý z internetových obchodů, ale tentokrát pomocí pohybu rukou.

Kapitola 2

Senzor Microsoft Kinect

2.1 Firma Microsoft

Jméno Microsoft je jedním z nejznámějších jmen v IT světě. Firma byla založena 4. dubna 1975 Billem Gatesem a Paulem Allenem v USA. Jejich první produkt byl interpreter jazyka BASIC pro počítač Altair 8800. Ještě v roce 1975 dosáhla firma obratu 1 milionu dolarů. Je zajímavé, že jméno Microsoft vlastně popisuje prvotní zaměření firmy, je to zkratka ze slov *Microcomputer* a *Software*. Následovala tvorba operačního systému, vycházejícího z Unixu, pojmenovaného Xenix a prvního textového procesoru pod jménem "Multi-Tool Word". Největší úspěch přišel s operačním systémem DOS, který firma dodávala IBM i dalším výrobcům klonů IBM PC. Úspěšná obchodní strategie udělala ze zakladatelů firmy jedny z nejbohatších lidí světa.

Dnes nabízí Microsoft značné množstvím produktů pro osobní i firemní použití. Nejznámější jsou operační systémy, které procházely dlouhým vývojem už od dob DOSu, který vyústil v samostatnou větev NT, na které jsou založeny dnešní Microsoft systémy, například stále populární Windows XP i nejnovější Windows 8. V době psaní této práce tvoří podíl systému Windows podle průměru odhadů 79,88%. Značně rozšířený je také kancelářský balík Microsoft Office, který obsahu tabulkový a textový procesor, e-mailový klient a program pro tvorbu prezentací. Ve firemní sféře nabízí serverové verze operačních systémů, Exchange server pro provoz a správu elektronické pošty a další.

2.2 Microsoft na konzolovém trhu

Kromě nesporných úspěchů ve výše zmíněných oblastech se Microsoft rozhodl proniknout i do světa počítačových her. Není to velké překvapení, když podle publikovaných statistik dostihl herní průmysl v obratu i průmysl filmový. Z jiných statistikách, které nebývají uveřejňovány v herních magazínech, nicméně vyplývá, že herní průmysl s tržbami 60 miliard dolarů filmový průmysl s tržbami 77 miliard USD jen dotahuje. Navíc do této statistiky je započítán i herní hardware, zatímco zařízení jako domácí kina a podobné nejsou do čísel za filmový průmysl započítány. Stále se však jedná o značně lukrativní odvětví.

Od roku 1998 vyvíjel Microsoft vlastní konzoli. Podle některých zdrojů bylo hlavním důvodem nabídnout konzoli s rozhraním DirectX a konkurovat s ní právě uváděnému PlayStation 2 od Sony. Za jméno konzole bylo zvoleno neexistující slovo Xbox - zkratka z *Direct X Box*. Na trh byla uváděna od roku 2001 do roku 2002 v různých státech. V té době konzolovému světu dominovaly asijské značky Sony se svým PlayStation 2 a Nintendo s GameCube (všechny se řadí do tzv. šesté generace konzolí). Konzoli se dostalo celosvětově poměrně pozitivního přijetí a do roku 2006 se prodalo více než 24 milionu kusů. Výjimkou bylo Japonsko, které zůstávalo věrnější Japonským značkám.

Obrázek 2.1: Konzole Xbox360

V roce 2005 Microsoft přichází s novou konzolí Xbox 360 2.1. Tato konzole je zařazována do takzvané sedmé generace herních konzolí (seventh console generation). Jedná se o generaci, která je v době psaní stále aktuální. Jejími zástupci jsou tři konzole, kromě Xboxu

360 je to PlayStation 3 od Sony a konzole Wii od Nintenda. Každá z konzolí přinesla technologické vylepšení proti předchozí generaci. U Xboxu 360 a PlayStation šlo o hraní nebo přehrávání filmů na Blue-ray nosičích v HD rozlišení. I když je to pro uživatele vítané vylepšení, jedná se v podstatě jen o vylepšení aktuálních hodnot rozlišení. Nintendo zacílilo svou konzoli jiným směrem a než na hard-core hráče se soustředilo více na širší publikum a přišlo s převratným systémem ovládání, které mělo ke hraní konzolových přivést nové hráče. Ostatní konzole se ovládají zařízením zvaným gamepad, ergonomickým ovladačem drženým obouručně, kde palce slouží ke stisku tlačítek a dlaně s prsty drží zařízení. To je ke konzoli připojeno buď kabelem nebo bezdrátově a navzdory tomu, že je vyráběno množstvím firem, je jeho tvar víceméně stejný. Wii tento systém reformoval svým ovladačem Wii Remote. Jak název napovídá, jedná se čistě o bezdrátové zařízení připojené přes Bluetooth rozhranní. Jeho hlavní výhodou je použití akcelerometru, který umožňuje přenášet pohyby ruky hráče držící tento ovladač na postavu ve hře, které tak provede stejný pohyb. Díky tomu bylo možné ovládat například herní postavičku v tenisové hře reálnými údery a podobné využití se dá najít i u dalších sportovních a jiných her.

Xbox 360 se zapsal do historie konzolí nechvalně tím, že u něj neobvykle často docházelo k selháním hardwaru, díky čemuž byla postižená konzole nepoužitelná. Toto se podle některých odhadů týkalo až jedné třetiny prodaných konzolí. Microsoft na to reagoval prodloužením záruky na nejčastější vadu o jeden rok a tajením přesných údajů o poruchovosti a příčinách. Dnes se nejčastěji spekuluje o použití nevhodné bezolovnaté pájky a přehřívání vnitřního prostoru konzole.

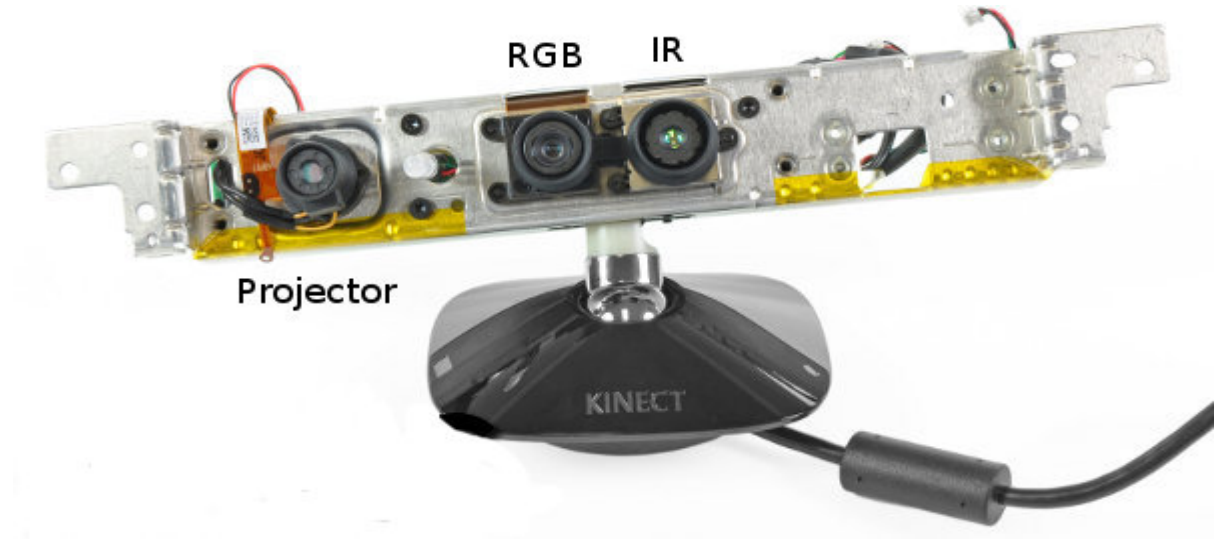
2.3 Kinect

Obrázek 2.2: Senzor Kinect

Díky úspěchu konzole Wii a jejího bezdrátového ovladače začali hledat podobný způsob ovládání i jiné firmy. Microsoft přišel ve spolupráci se společností PrimeSense, s vlastním řešením, vyvíjeným jako projekt Natal, což je jméno Brazilského města a současně termín označující zrození, což může vystihovat Microsoftem očekávaný revoluční dopad této technologie. Místo, aby dal hráčům do rukou nový typ ovladače, doslova jim z rukou ovladač sebral a nabídl jim senzor pojmenovaný nakonec Kinect. Nový systém Microsoftu totiž používá k ovládání celého hráče, konkrétněji jeho tělo. To je snímáno kamerou a hloubkovým senzorem Kinectu a zpracováno do výsledných souřadnic (prostorových) klíčových bodů na těle hráče. Díky tomu může hráč ovládat hru bez použití ovladače a není tak limitován délkou kabelu a podobnými faktory. Naopak limitem se v tomto případě může stát nedostatek místa. Aby senzor podával ideální výkon, je nejvhodnější umístit ho poblíž (ideálně nad nebo pod) zobrazovací zařízení (dnes většinou televizní přijímač) tak, aby k němu byli uživatelé sledující obrazovku otočeni čelem. Tím je zajištěno, že uživatelé při sledování obrazu jsou k senzoru natočení čelně, což usnadňuje detekci. Původní senzor byl označen stříbrnými písmeny Xbox 360, přibližně po roce po uvedení přišel Microsoft s verzí Kinect pro Windows, ke které poskytl vlastní SDK. Tato verze byla označena velkým nápisem Kinect, ale jinak byla totožná s původní verzí pro konzoli. Zařízení je osazeno kabelem s jedním konektorem na konci. Jde o speciální proprietární konektor použitý u Xboxu 360, označovaný AUX. V podstatě se jedná o rozšíření USB 2.0 sběrnice o další 2 vodiče pro přenos +12V napájecího napětí. Díky tomu není možné senzor provozovat jen pomocí napájení z běžného USB portu (maximální napětí je +5V a proud je omezen na 500mA pro USB 2.0 a 900mA u USB 3.0). Tento problém vyřešil Microsoft přídavným zdrojem, který se zapojuje mezi AUX konektor Kinectu a PC s běžnými USB porty a dodává potřebné napájení.

2.4 Hardware Kinektu

Samotný senzor má plastové tělo umístěné na plastovém stojánku, který zajišťuje stabilitu a obsahuje motor, kterým je možné senzor natáčet podle vodorovné osy. Jiné natáčení není možné, takže senzor není schopen uživatele sledovat při extrémním pohybu do stran. Tělo senzoru tvoří podlouhlý, na kratších stranách zkosený kvádr, ve kterém jsou umístěny vlastní zařízení. Jde o IR projektor, který promítá mřížku, která je snímána IR CMOS kamerou a určuje vzdálenost. Dále je Kinect osazen RGB kamerou, která snímá barevný obraz v rozlišení 1080 * 1024. Pro doplnění je v senzoru ještě pole mikrofonů, které umožňuje



Obrázek 2.3: Kinect po demontáži krytu

určovat směr zdroje zvuku (například může sloužit pro identifikaci hráče, který promluvil).

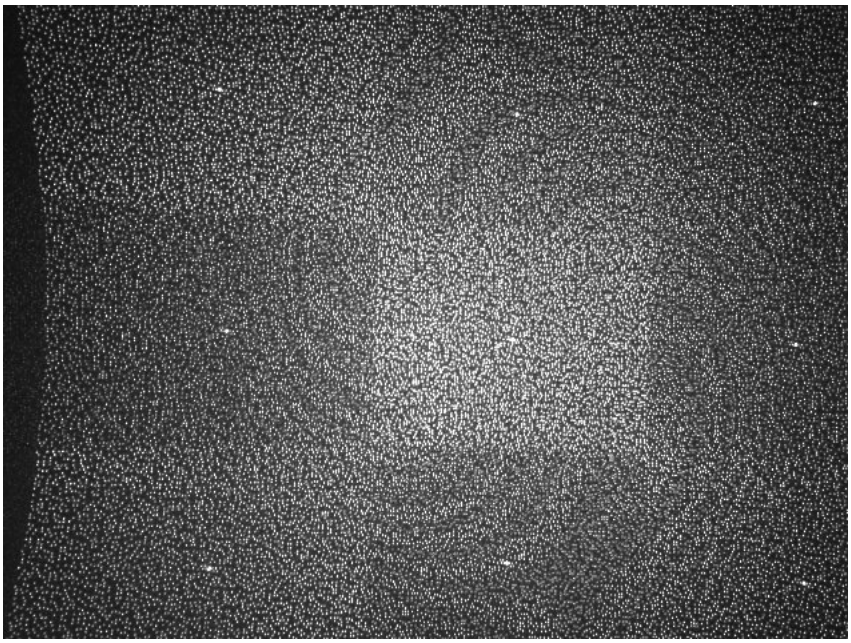
2.4.1 Barevná kamera

Barevná CMOS kamera se už stává standardem v mnoha zařízeních, jako jsou mobilní telefony, tablety a podobně. Kamera v Kinectu neslouží přímo pro výpočet vzdáleností a co se obrazové kvality týče, je spíše průměrné až podprůměrné kvality. Rozlišení videa je maximálně 1280x1024 pixelů, zorné úhly kamery jsou 63° horizontálně a 50° vertikálně.

2.4.2 IR projektor a IR kamera

IR podsystém je klíčový pro vznik hloubkového obrazu, který usnadňuje zpracování obrazu a identifikaci postav hráčů. Skládá se z IR projektoru, který promítá mřížku z teček v IR spektru a monochrom1atickou CMOS kamerou citlivou jen na IR pásmo. Kamera má rozlišení 1280 na 1024 pixelů zorné úhly 57° horizontálně a 45° vertikálně, tedy méně, než nabízí RGB kamera. Ohnisková vzdálenost je 6,1mm.

Přesný způsob vzniku hloubkového obrazu zamotal hlavu mnoha technologickým nadšencům a ještě dnes rezonují mnoha internetovými diskuzemi překroucené informace. Samotný princip fungování není pro využití senzoru podstatný a programátor se na něj může dívat jako na blackbox poskytující hloubkový obraz v daném rozlišení. V principu se hloubkový obraz získává triangulací. Triangulace je způsob výpočtu polohy bodu v prostoru, když známe

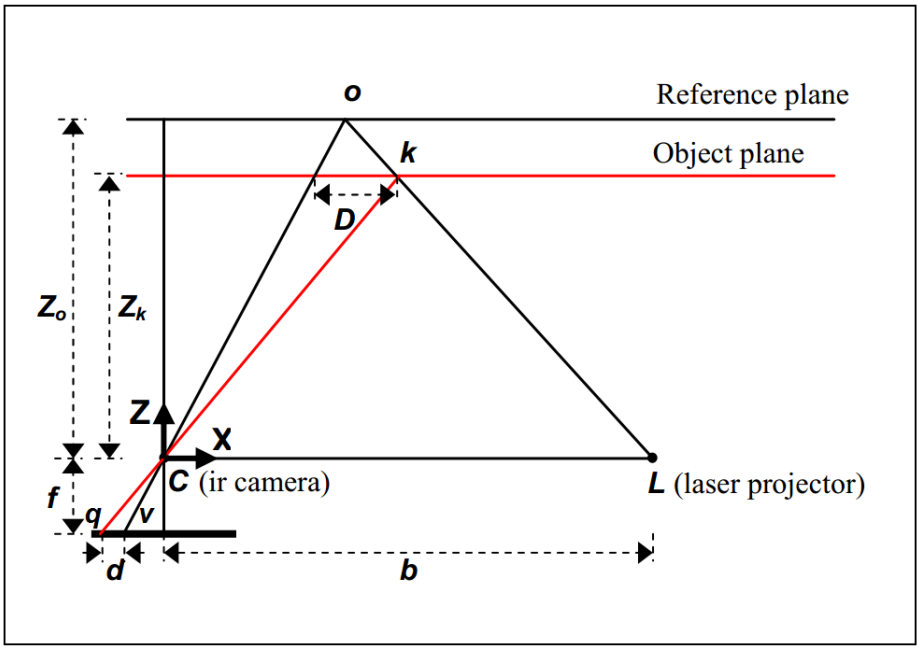


Obrázek 2.4: Obraz promítaný IR projektorem

jeho obraz ve 2 kamerách a parametry těchto kamer. V tomto případě je jedna z kamer nahrazena IR projektorem, druhá je pak standardní IR kamera. IR projektor promítá obrazec složený z teček tak, jak je zobrazeno na obrázku 2.4. Tečky nejsou rozprostřeny náhodně, ale tvoří stálý vzorek vytvořený z IR laseru na difrakční mřížce. Každý Kinect promítá mírně odlišný IR obrazec a na tento svůj obrazec je při výrobě uložen v paměti senzoru.

Promítaný obrazec je snímám IR kamerou a z poloh teček je určena vzdálenost. Způsob výpočtu je znázorněn na obrázku 2.5. Pokud promítněme obraz z IR projektoru na referenční rovinu a nasnímáme kamerou, získáme obraz pro tuto vzdálenost referenční roviny. Při posunu této roviny ve směru osy z, tedy od nebo ke kameře se polohy konkrétních teček mění. Pokud jsme schopni detekovat konkrétní IR tečky nebo jejich skupiny, můžeme zjistit, jaký je jejich posun oproti vzdálenosti referenční roviny, která je známá. Na obrázku 2.5 je bod o ve vzdálenosti referenční roviny zobrazen v obrazové rovině v bodě q. Pokud dojde k posunutí této roviny, vidí kamera tento shluk IR teček (v obrázku bod k) v obrazové rovině jako bod v. Zjištěnou rozdíl poloh těchto bodů je označeno jako vzdálenost d. Pomocí podobnosti trojúhelníků můžeme formulovat rovnice 2.1 a 2.2.

$$\frac{D}{b} = \frac{Z_0 - Z_k}{Z_0} \tag{2.1}$$



Obrázek 2.5: Schéma promítání a snímání IR obrazu [Převzato z [3]]

$$\frac{d}{f} = \frac{D}{Z_k} \tag{2.2}$$

Pokud z rovnice 2.1 vyjádříme D a použijeme v rovnici 2.2, dostaneme 2.3.

$$Z_k = \frac{Z_0}{1 + \frac{dZ_0}{fb}} \tag{2.3}$$

Tato rovnice říká, že můžeme spočítat vzdálenost Z_k při znalosti těchto parametrů:

Z_0	vzdálenost referenční roviny při kalibraci
f	ohnisková vzdálenost IR kamery
b	baseline - vzdálenost mezi kamerami
d	rozdíl poloh obrazů bodu v obrazové rovině

Protože hodnoty Z_0, f a b jsou získány při kalibraci senzoru, jedinou neznámou je hodnota d, která určuje rozdíl obrazů aktuálního bodu a bodu ve vzdálenosti Z_0. Z této hodnoty lze určit vzdálenost bodu od IR kamery v ose z, tj. směrem od kamery. Další dvě souřadnice jsou dopočítány podle rovnic 2.4 a 2.5. Kde x_k jsou souřadnice bodu v obrazové

rovině, x_0 je souřadnice centra obrazu a δx je zkreslení čočky, které představuje parametr, který lze získat kalibrací.

$$X_k = \frac{Z_k}{f}(x_k - x_0 + \delta x) \tag{2.4}$$

$$Y_k = \frac{Z_k}{f}(y_k - y_0 + \delta y) \tag{2.5}$$

Výstupní hodnota vzdálenosti je normalizována do 11 bitů, dovolovala by tedy popsat 2048 hodnot hloubky, ale protože je jeden bit vyhrazen k označení pixelů, u kterých nebyla hloubka změřena, je pro vyjádření hloubky prakticky k dispozici 10 bitů, tedy 1024 hodnot. Hodnota vzdálenosti není převedena do 10 bitů lineárně a umožňuje dosáhnout vyššího rozlišení u kratších vzdáleností. Výstupní hloubkový obrázek je generován s frekvencí 30Hz, tedy každých $33,3ms$.

2.4.3 Mikrofonní pole

Senzor je osazen čtyřmi mikrofony umístěnými na spodní straně zařízení v jedné přímce, každý se vzorkovací frekvencí 16kHz a 24bitovým ADC převodníkem. Tři z mikrofonů jsou umístěny na jednom okraji senzoru, čtvrtý na protilehlém. Toto umístění umožňuje určovat směr, ze kterého zvuk přichází a tím například určit, který z hráčů promluvil. Směr je určen úhlem měřeným od osy z (hloubkové osa) a pohybuje s v rozmezí -50° až +50°.

2.4.4 Ovládání náklonu

Součástí podstavy senzoru je i mechanismus umožňující naklápění horní části senzoru s kamerami a mikrofony. Toto naklopení je omezeno na ±27° od svislé osy, díky zornému úhlu kamer je pro sledování uživatele v předpokládané vzdálenosti dostatečné. Microsoft v dokumentaci k senzoru nicméně varuje, že není vhodné používat ovládání náklonu příliš často a doporučuje používat ho jen k nastavení při inicializaci prováděné po spuštění. Součástí senzoru je navíc akcelerometr, který umožňuje určit aktuální odchylku od svislého směru.

Kapitola 3

Kinect API

Brzy po vydání Kinectu v roce 2010 začala vznikat ovladače a software umožňující jeho využití i na jiných platformách než Xbox 360, hlavně na Windows, Linuxu a počítačích Apple. Sám Microsoft se držel zpátky a své vlastní SDK (Software Development Kit - balík softwaru určený k vývoji dalšího softwaru) nabídl až následující rok. V současnosti jsou pro použití Kinectu na systému Windows nejpoužívanější tyto SDK:

1. OpenKinect
2. CL NUI
3. OpenNI
4. Microsoft's Kinect for Windows SDK

3.1 OpenKinect

3.2 CL NUI

Vyvinuto společností Code Laboratories v roce 2010, poslední verze k dispozici je z 10.12.2010. Lze stáhnout ze stránek společnosti. Umožňuje vývoj v jazycích C# a C/C++ pod operačními systémy Windows XP, Vista a 7. Je k dispozici zdarma. Podporuje jak obrazová data z barevné a hloubkové kamery, tak nahrávání zvuku z mikrofonů, signalizační LED diody a ovládání motoru.

3.3 OpenNI

Open NI framework je produktem společnosti PrimeSense, která se podílela i na vývoji Kinectu. Je zaměřen na více širší škálu zařízení, než jen Kinect (PrimeSense prodává vlastní senzory Carmine, ve dvou provedeních pro blízké a standardní vzdálenosti). Open NI nabízí široké možnosti využití díky své konstrukci a nabídce tzv. middlewarů, tedy programů, které běží mezi uživatelskou aplikací a OpenNI a přidávají další funkce, například detekci polohy uživatele, detekci kostry, detekci ruky.

3.4 Microsoft's Kinect for Windows SDK

Microsoft uveřejnil první verzi vlastního SDK sedm měsíců po uvedení Kinectu (4.11.2010 - 7.6.2011). Jeho hlavní výhodou je pohodlná instalace a spolupráce s vývojovým prostředím Visual Studio (taktéž od Microsoftu). Mezi nevýhody patří podpora pouze pro systém Win 7 a novější, podporován tak není stále rozšířený systém Windows XP, jehož rozšíření ale bude s průběhem času velmi pravděpodobně klesat. SDK je možno využít s jazyky Visual Basic, C, C++ a C#, pro některé vývojáře může být překážkou absence podpory jazyka Java. Od té doby bylo vydáno několik aktualizací, v současnosti byla uveřejněna verze 1.7. Nové verze přidávají zajímavé nové možnosti, nejedná se tedy jen o opravy chyb a minoritní změny. Například v nové verzi byla přidána možnost číst data z akcelerometru senzoru a určovat tak jeho pohyb nebo zpřístupnění obrazu z IR kamery. V nových verzích je také přidána spolupráce s Matlabem a OpenCV. Nové verze SDK oficiálně podporuje upravenou verzi senzorů, nazvaných "Kinect For Windows Sensor", která je otestovaná a funkční s novým SDK. Původní a tuto verzi lze rozeznat podle nápisu Xbox 360 na přední straně původní verze, nový senzor má na stejném místě napsáno Kinect.

3.4.1 Licence

Licence je důležitou součástí dodávaného softwaru a je přiložena obvykle ve formě instalačního okna s celou licencí, kterou musí uživatel odsouhlasit, aby bylo možné software nainstalovat (nutno poznamenat, že pro většinu uživatelů místo čtení licence obnáší tento krok jen posunutí posuvníku a odklepnutí tlačítka pro pokračování). Součástí licence pro Kinect For Windows SDK je několik důležitých podmínek, které by měl znát každý uživatel tohoto systému:

1. Použití senzoru určeného pro Xbox360 je povoleno jen pro testovací a vývojové účely.

2. Uživatelé finální aplikace nemají licenci k použití senzoru Kinect for Xbox 360 a nesmí být k takovému chování přímo ani nepřímo pobízeni.

3. No High Risk Use - licence zakazuje použití senzoru a softwaru v aplikacích, kde může selhání vést k vážnému zranění nebo smrti osob, nebo závažnému poškození životního prostředí.

4. Licence povoluje vývoj a prodej komerčních aplikací vytvořených v tomto SDK zákazníkům používajícím Kinect for Windows senzor.

   Licence tedy povoluje vývoj, prodej nebo bezplatnou distribuci vytvořeného softwaru, pokud budou uživatelé používat senzor určený pro Windows.

Velkou výhodou je možnost stažení značného množství pomocných aplikací, které jsou dodávány jako Kinect For Windows Developer Kit (obsahuje například Kinect Studio), příkladů a další podpora pro vývojáře.

3.4.2 Použití Kinect For Windows SDK

Po stažení je nutné Kinect for Windows SDK nainstalovat klasickým spuštěním instalačního souboru, instalaci a integraci do Visual Studia instalační program provede sám. Pro použití je také třeba mít Visual Studio, dostatečná je volně stažitelná Express verze, která podporuje vývoj v jazycích C#, C++ i Visual Basic. Po spuštění Visual Studia a založení nového projektu (v jazyce C#) je nutné z GAC (Global Assembly Cahe) pro tento projekt přidat referenci na Microsoft.Kinect.dll. Poté stačí v programu použít direktivu *using* pro snadnější přístup do Kinect namespace: *using Microsoft.Kinect*.

Hlavním prostředkem komunikace se senzorem je použití streamů, což jsou v tomto případě proudy dat ze senzoru do PC. Aplikace si u senzoru může aktivovat tyto streamy:

1. Color Stream [4]

2. Depth Stream [5]

3. Skeleton Stream [6]

3.4.3 Color Stream

Představuje stream generovaný z klasické barevné kamery senzoru. Je možné nastavit rozlišení 1280x960 při snímkové frekvenci 12fps nebo 640x480 se snímkovou frekvencí 30

snímků za sekundu v barevném modelu RGB nebo YUV (v tom případě je k dispozici jen rozlišení 640x480). Před použitím tohoto streamu je nutné ho aktivovat zavoláním funkce ColorStream.Enable();, jejímž jediným argumentem je požadované rozlišení a frekvence (k dispozici po výčtovém typu *ColorImageFormat*). O přijetí nového snímku je aplikace informována pomocí události *ColorFrameReady*. Po jejím spuštění je spuštěna funkce přihlášená k odběru této události. Jí je předán argumentem typu *ColorImageFrameReadyEventArgs*, který obsahuje pole dat nasnímaných senzorem. Pomocí jeho metody *OpenColorImageFrame* jsou data překopírována do instance třídy *ColorImageFrame* a odtud jsou dále zkopírovány do pole hodnot typu byte určeného pro jejich uložení a další zpracování.

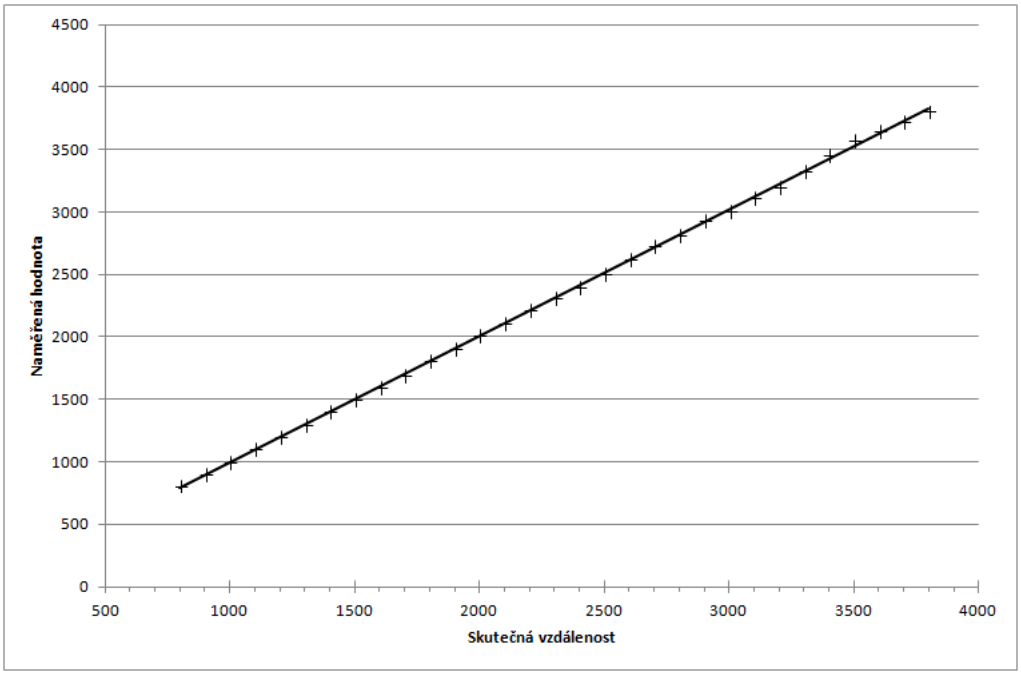
3.4.4 Depth Stream

Informace získané IR kamerou jsou přenášeny tímto streamem. Hloubkové data jsou přenášena jako 11 bitové číslo, jeden bit je vyhrazen pro identifikaci pixelů, u kterých se nepodařilo hloubku zjistit. Další 3 bity jsou určeny pro identifikaci pozic jednotlivých hráčů. Obsahují tedy číslo od 1 do 6 pro identifikaci konkrétní osoby, nebo 0 pro pole, kde se hráči nenacházejí. Způsob práce s tímto streamem je velmi podobný Color Streamu. Stream je nutno nejprve ho aktivovat zavoláním funkce *DepthStream.Enable()*, jejímž argumentem může být rozlišení, které je požadováno. To lze volit z těchto možností: 80x60, 320x240 nebo 640x480, všechny při 30 snímcích za sekundu. Systém přenosu dat ze senzoru je stejný jako v předchozím případě: událost DepthFrameReady slouží k přijetí dat a uložení do pole, tentokrát díky většímu možnému rozsahu hodnot jde o pole typu short (16 bitů).

Takto obdržená data je třeba bitově posunout vpravo a zbavit se tak bitů určených k indikaci osoby. Po této operaci dostáváme čísla, která odpovídají vzdálenosti bodů obrazu od senzoru, v milimetrech. Některé zdroje uvádějí nelineární průběh závislosti vrácené hodnoty na vzdálenosti objektu. Toto se týká jiných SDK, hodnota vrácená v tomto případě odpovídá vzdálenosti, viz graf 3.1.

3.4.5 Skeleton Stream

Pomyslný vrcholem funkcí senzoru představuje Skeleton Stream. Jde o proud dat, ve kterých už byly s použitím zbylých dvou streamů nalezeny osoby a určeny polohy části jejich těl. Lze tedy přímo vyčíst například polohu ruky, nebo vzájemnou polohu rukou. Zbylé dva streamy nemusejí být pro použití skeleton streamu programátorem zapnuty, budou zapnuty a nastaveny automaticky.



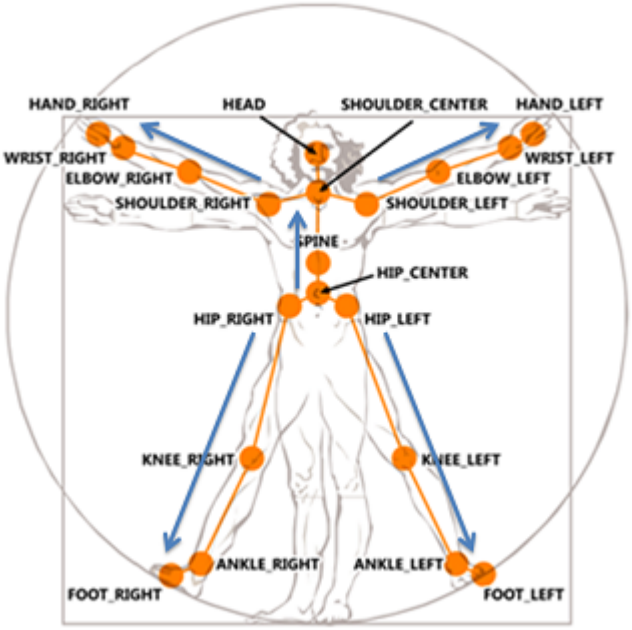
Obrázek 3.1: Závislost změřené vzdálenosti na skutečné vzdálenosti objektu od senzoru

Použití tohoto streamu je se mírně liší od obou předchozích. Společné mají oznámení příchodu nových dat vyvoláním události a předání dat argumentem (tentokrát typu *SkeletonFrameReadyEventArgs*), další kroky jsou odlišné - zatímco Depth a Color Stream vracejí pole typu byte, které odpovídají jasu pixelů v obrázku (případně jasu subpixelů u RGB zápisu), vrací Skeleton Stream až 6 tzv. skeletonů - instancí třídy Skeleton. Každý z nich představuje obraz jednoho hráče. Tento obraz může být jednoho ze dvou typů (jsou určeny vlastností *TrackingState*):

1. tracked - označované také jako aktivní sledování. Pokud je skeleton tohoto typu, je v této instanci využita kolekce tzv. Jointů: souřadnic jednotlivých částí lidského těla. Těchto souřadnic je celkem 20 a slouží k detailnímu popisu celé postavy uživatele. Jejich polohu na lidském těle zachycuje obrázek 3.2. Každý z nich obsahuje typ jointu (například Hlava, Pravý Loket, Pravé Rameno atd.) a jeho souřadnice - ty jsou trojrozměrné, takže je k dispozici i vzdálenost od senzoru. Takto je senzor schopen popsat až dva uživatele. V případě, že jich je před senzorem přítomno více, jsou ostatní také detekováni (až do počtu 6), ale jen jako "position only". Dvě detailně popisované uživatelské postavy jsou ty, které se v zorném poli senzoru objeví nejdříve. Aplikace má možnost změnit uživatele, kteří budou takto sledováni pomocí metody

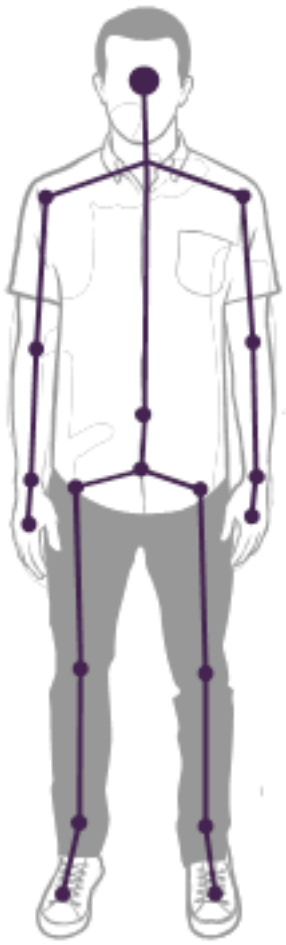
ChooseSkeletons třídy SkeletonStream. Metoda má 0 až 2 parametry, které určují ID skeletonů, které budou sledovány. V případě verze s 0 parametry je výběr uživatele vrácen senzoru. Pokud uživatel, jehož skeleton je nastaven jako tracked, opustí zorné pole senzoru a opět se do něj vrátí, je pravděpodobné, že "výsada"detailního sledování mezitím přešla na jiného uživatele.

2. position only - senzor je schopen detailně sledovat maximálně dvě postavy. U ostatních postav, až do počtu 6, je vrácena poloha postavy v obraze, tedy souřadnice jednoho bodu. Ta je jako v předchozím případě trojrozměrná. Její Z složku (tedy vzdálenost od senzoru) lze využít například pro určení uživatelů, kteří jsou nejblíže senzoru a ty pak detailně sledovat předáním jejích ID funkci *ChooseSkeletons(int ID1, int ID2)*.



Obrázek 3.2: Jointy - body, které senzor detekuje na postavě uživatele [Zdroj Microsoft.com]

Právě Skeleton Stream představuje jednu z nejužitečnějších funkcí Kinectu. Díky automatické identifikaci uživatele a nalezení souřadnic jednotlivých bodů na těle velmi zjednodušuje vytváření aplikací. Vývojáři nemusí implementovat vlastní metody pro rozpoznání obrazu, ale mohou se soustředit na vývoj vlastní aplikace.



Obrázek 3.3: Kresba kostry z bodů Skeleton Streamu [Zdroj Microsoft.com]

3.4.6 Použití více senzorů najednou

Teoreticky nebrání současnému připojení a běhu více senzorů k jednomu počítači žádné omezení. Senzor se připojuje přes USB port, které se staly standardním vybavením dnešních počítačů a každá základní deska nabízí tyto porty minimálně 2, velmi často až 6. Počet připojených senzorů lze pak zjistit pomocí vlastnosti *Count* kolekce *KinectSensors* a k jednotlivým senzorům pak přistupovat pomocí indexů. Problémy může způsobit jen Skeleton Stream, který lze snímat maximálně ze 4 senzorů najednou.

Při použití více senzorů snímajících jednu scénu se však mohou vyskytnout problémy s přesností snímání hloubkových dat způsobené interferencí IR obrazu promítaného IR projektorem jednoho senzoru s IR projektory ostatních senzorů. To může způsobit problémy

i u snímaného Skeleton Streamu.

3.4.7 Kalibrace senzoru

Aby bylo možné obrazy nasnímané z obou kamer (RGB a hloubkové) spojit a kromě barvy mít pro objekty ve scéně i hloubku, je nutné provést nejprve kalibraci obou kamer. Kalibrace je proces, kterým získáme parametry kamery, které můžeme později použít k transformaci obrazu z jedné z kamer na kameru druhou (a samozřejmě také obráceně). Pro kalibraci předpokládáme chování kamery odpovídající modelů dírkové kamery (Pinhole camera model).

Model dírkové kamery

Model dírkové kamery je užitečné zjednodušení, které předpokládá, že světelný paprsek se šíří po přímce z objektu (bodu ve scéně) přes bod nazývaný střed kamery prochází obrazovou rovinou. Paprsky neprocházející bodem C jsou ignorovány. Nejedná se však o pouhý teoretický model, ale vychází z reálných zařízení, které je možné jednoduše sestrojit proražením malé dírky například ve stěně krabice, na jejíž protější vnitřní straně (obrazové rovinně) se bude promítat převrácený obraz. Tento jednoduchý přístroj využívají fotografičtí nadšenci i v dnešní době a projevy tohoto principu lze pozorovat i v přírodě.

Obrazová rovina (image plane) je plocha, na které vzniká obraz.

Z modelu dírkové kamery můžeme odvodit vztah pro zobrazení bodu X z 3D prostoru do obrazové roviny.

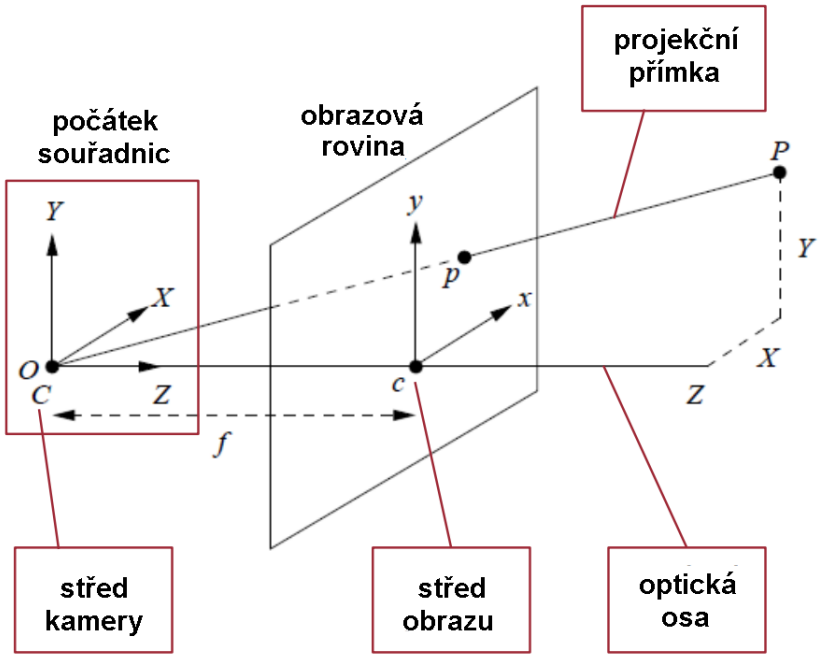
Tento model nám umožňuje matematicky vyjádřit činnost kamery jako zobrazení

$$X \in A^3 \rightarrow x \in A^2 \tag{3.1}$$

kde X představuje souřadnice bodu v prostoru a x představuje souřadnice obrazu tohoto bodu v obrazové rovině.

Matice kamery

Zmíněný model dírkové kamery lze použít k přepočítání souřadnic bodu ze scény na souřadnice, na kterých ho najdeme na výsledném obraze nebo fotografii vytvořené kamerou. Nejednoduší případ je zobrazen na obrázku 3.4. V něm označuje P bod ve scéně se souřadnicemi X, Y, Z, jehož obrazem je bod p v obrazové rovinně se souřadnicemi u, v.



Obrázek 3.4: Výpočet souřadnic bodu X v obrazové rovinně [Zdroj Microsoft.com]

Souřadnice x, y v obrazové rovině můžeme pomocí podobnosti trojúhelníků vyjádřit jako

$$\frac{X}{Z} = \frac{x}{f} \qquad \frac{Y}{Z} = \frac{y}{f} \tag{3.2}$$

$$x = \frac{X}{Z}f \qquad y = \frac{Y}{Z}f \tag{3.3}$$

Výše uvedené vzorce počítají s tím, že počátek souřadnic v obrázku je v centru obrazové roviny (senzoru). Obvyklejší je počítat souřadnice obrázků pomocí souřadné soustavy s počátkem v levém horním rohu. Pokud v této soustavě vyjádříme polohu centra obratové roviny, tedy bodu c, jako vektor o dvou složkách $\vec{c} = (c_x, c_y)$, můžeme rovnice upravit na

$$x = \frac{X}{Z}f + c_x \qquad y = \frac{Y}{Z}f + c_y \tag{3.4}$$

Tyto rovnice je vhodné přepsat do maticového tvaru, označme 3D bod X a jeho obrazem bude bod x a použijeme kalibrační matici kamery K.

$$\tilde{x} = \frac{1}{Z}PX \tag{3.5}$$

$$\tilde{x} = \frac{1}{Z} \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.6}$$

Označíme-li $\frac{1}{Z} = \rho$, dostaneme

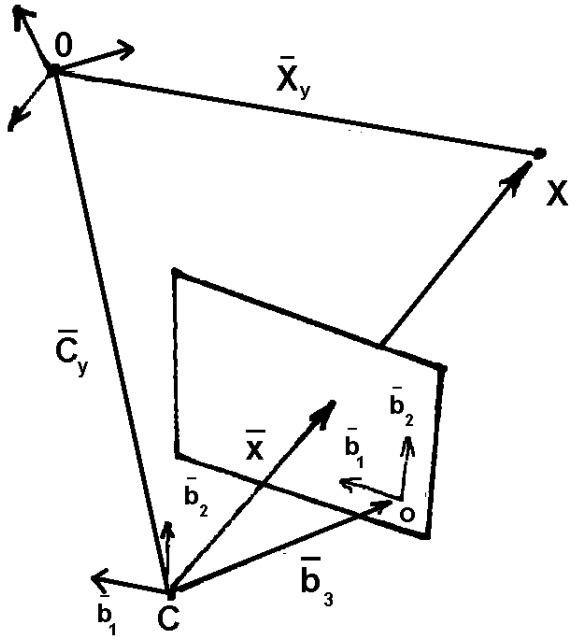
$$\rho\tilde{x} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.7}$$

V rovnici jsme dostali místo očekávaného bodu x bod označený jako $\tilde{x}$. Jedné o bod x vyjádřený v homogenních souřadnicích. Ty jsou pro operace promítání vhodné, protože je umožňují vyjádřit jako násobení matic. Zjednodušeně lze říct, že se jedná o přidaní další dimenze k aktuálním souřadnicím, takže místo (x, z) dostáváme (x, z, w).

Matice, označená jako K, představuje kalibrační matici kamery. Určuje, jak budou body kamerou zobrazeny a obvykle se nemění. Zahrnuje v podstatě jen dvě hodnoty: f, které představuje ohniskovou vzdálenost kamery. Je určena vzdáleností středu kamery C od obrazové roviny (snímacího čipu). Tato vzdálenost se dá změnit použitím jiného objektivu nebo změnou ohniskové vzdálenosti u tzv. zoom objektivů, tedy objektivů, které mají proměnnou ohniskovou vzdálenost, což není případ Kinectu, takže je můžeme považovat za neměnné. Další její složky, c_x a c_y, jsou neměnné. V některých aplikacích se používá matice obsahující další nenulové hodnoty, které odpovídají zkosení pixelů, ale v tomto případě je uvedená matice dostatečná.

Tyto vzorce platí ale jen v případě, že je střed kamery umístěn v počátku souřadné soustavy a osa Z je totožná s optickou osou. Pokud toto neplatí, je nutné použít složitější model, konkrétně transformaci souřadnic mezi dvěma bázemi - první báze je obvykle nazývána souřadnice světe/scény a představuje souřadný systém nezávislý na kameře. Druhým je souřadný systém kamery, který má počátek v bodě C, označovaným jako střed kamery, viz obrázek 3.5

Bod X vyjádříme v homogenních souřadnicích a označíme jej $\tilde{X}$. Díky tomu můžeme



Obrázek 3.5: Projekce

sestavit následující rovnici:

$$\rho\tilde{x} = \begin{bmatrix} KR & | & -KRC \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.8}$$

Matice R, který se v rovnici objevila, představuje rotační matici, která určuje rotaci mezi bázemi. C představuje polohu středu kamery v souřadné soustavě světa.

Parametry kamery se dělí na dvě skupiny, vnitřní a vnější. Vnitřní, nazývané také intrinsické, jsou určeny vlastnostmi kamery a nemění se. Představuje je matice K. Druhou skupinou jsou vnější, extrinsické, které vyjadřují aktuální polohu kamery a její natočení, ty jsou vyjádřeny maticí R a vektorem C. Tyto se mění poměrně často, při každém pohybu kamery, proto je vhodné je vyjádřit samostatně.

3.4.8 Kalibrace tohoto konkrétního senzoru

Matice a hodnoty uvedené výše je nutné nějakým způsobem získat. U profesionálních přístrojů se dá spolehnout na výrobce a jím poskytované informace, jedná se však o drahá

a specializovaná zařízení. V případě použití obvyklých snímacích prostředků je nutné tyto parametry spočítat. Tento proces se nazývá kalibrace kamery (kalibrace může mít i jíné významy, například určení závislosti jasu pixelů na jasu/teplotě objetků, ale v tomto případě máme na mysli geometrickou kalibraci). Ke kalibraci se standardně používají speciální obrazce, nejčastěji tzv. šachovnice, na kterých se pravidělně střídají černá a bílá pole, stejně jako na klasické šachovnici. Na rozdíl od ní může být polí libovolné množství a šachovnice libovolně velká. V dneštní době není problém si tuto šachovnici vytisknout na klasické počítačové tiskárně. Šachovnice je nasnímána, ideálně v různých polohách kalibrovanou kamerou. V těchto snímcích jsou pak. buď ručně nebo automaticky, určeny rohy šachovnice. Po určení rohů lze použít automatický nebo ruční systém pro určení polohy jednotlivých pixelů.

Kalibrací s použítím programu Matlab a "softwaru"Camera Calibration Toolbox for Matlab byly zjištěny tyto vnitřní parametry kamery:

$$K = \begin{bmatrix} 527.97 & 0 & 326.99 \\ 0 & 527.22 & 275.95 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

Kromě nich byly zjištěny tyto koeficienty zkreslení: $K_1 = 0.356060$
$K_2 = -1.50301$
$K_3 = 0.00479$
$K_4 = -0.00085$
$K_5 = 2.31797$

Kde koeficienty s indexy 1 a 2 odpovídají radiálnímu zkreslení a koeficienty s indexy 2 a 3 odpovídají tangenciálnímu zkreslení. Z velikostí koeficientů je vidět, že tangenciální zkreslení je minimálně o dva řády menší než radiální, proto je možné ho zanedbat.

Hodnoty zhruba odpovídají předpokládaným hodnotám získaným kalibrací jiných Kinect senzorů uváděných na internetu. Stejným způsobem je potřeba provést kalibraci hloubkové kamery. Zde nelze použít šachovnici jako v předchozím případě, protože by byla zobrazena jako plocha jedné barvy. Naštěstí pro kalibraci není její rozeznatelnost bezpodmínečně nutná, kalibraci lze provést i s jiným rovinným pravidelným útvarem, ideálně například listem papíru většího formátu, u kterého stačí zjistit souřadnice rohů. Alternativně by bylo možné použít zdroj IR záření a šachovnici vytvořit aplikací dvou barev, z nichž jedna IR

záření použité vlnové délky odráží a druhá nikoli.

$$K_{depth} = \begin{bmatrix} 578.95 & 0 & 320.94 \\ 0 & 572.0417 & 242.66 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

Kapitola 4

Použití Kinectu

Kinect vyvolal po svém uvedení v roce 4.11.2010 vlnu zájmu především mezi hráči videoher, hlavně těmi vybavenými konzolí Xbox 360, která byla jako jediná schopná se senzorem spolupracovat. Protože byly se senzorem vydány hry podporující nové ovládání (nově vyvinuté i upravené starší vhodné tituly) a Microsoft investoval do propagace půl miliardy dolarů, byl zájem hráčské veřejnosti poměrně vysoký. Byl dokonce tak vysoký, že si vysloužil zápis do známé Guinessovy knihy rekordů jako *Nejrychleji prodávané zařízení domácí elektroniky*. Během prvních 25 dní se prodalo 2,5 milionů Kinectů. Téměř okamžitě po vydání nabídla společnost Adafruit Industries odměnu 1000 americkývch dolarů osobě nebo týmu, která vytvoří a publikuje open-source ovladač. Microsoft se proti této výzvě ohradil a pohrozil právními následky, načež Adafruit navýšil odměnu na 2000 dolarů. Vytvořit první open-source ovladač (pro OS Linux) a aplikaci trvalo přibližně dva týdny (v té době činila odměna 3000 dolarů). Microsoft poté ústy svého mluvčího prohlásil, že Kinect byl navrhnut úmyslně nechráněn proti použití s neoriginálním zařízením. Také označili tvorbu komunity nadšenců okolo Kinectu za inspirativní. Postupně vznikla celá řada aplikací od čistě demonstrativních programů, počítačových her přes programy využitelné v průmyslu, robotice, školství, lékařství i jiných oborech.

4.1 Digitální výloha

Digitální výloha je termín používaný pro prezentaci firmy a jejích výrobků (umístěný převážně ve výloze) s pomocí digitální audio a video techniky. Výkladní prostor byl vždy jakýmsi okénkem do obchodu, ve kterém vystavoval prodejce lákavou aranž toho nejlepšího, co nabízel. Dříve se jednalo o statickou ukázku zboží, fotografií, ozdob a nápaditého

osvětlení. Ta měla nevýhodu v nutnosti ruční výměny zboží, která byla časově náročná. Postupně bylo možné upoutat zákazníka a představit mu zboží i pomocí televizní obrazovky, případě projektoru nebo obrazu promítaného na průhledné nebo neprůhledné fólie. Protože je interakce v mnoha oborech považována za lepší, než pasivní sledování, je možné využít Kinect k dalšímu kroku ve vývoji výlohy a umožnit zákazníkovi interakci s výlohou. Tím lze zapojit do hry interaktivitu, která má mnohem větší potenciál zákazníka/uživatele upoutat. Zatímco klasická výloha je buď statická nebo se mění podle nastavení obchodníka, v tomto případě si uživatel může sám rozhodnout, co ho zajímá a objekt svého zájmu si prohlédnout detailněji. To mu umožní vybrat a zobrazit zboží, o které má zájem, zobrazit informace o něm, popřípadě ho rovnou objednat, rezervovat nebo koupit. To lze navíc provádět i v době, kdy je obchod zavřený, navíc je systém v interiéru "za sklem", takže nepřichází do přímého kontaktu s uživatelem a je chráněn proti řádění vandalů.

Protože systém dokáže interaktivně reagovat na své okolí, může detekovat přítomnost osoby například ji přímo oslovit a upoutat tak její pozornost. Také je možné zaujmout zákazníky (především ty mladší) hrou, která může zákazníka jak přitáhnout k prezentaci, tak pobavit. Jako další motivace pro uživatele může být možnost použít potenciální výhru k osobní výhodě, například slevě zboží nebo získání reklamního předmětu.

Použití Kinectu přináší několik specifických podmínek:

1. scéna musí být přiměřeně osvětlená. Příliš tmavá scéna neumožňuje pořídit barevný obraz, pokud je scéna příliš přesvětlena zdroji s podílem IR záření, může to způsobit potíže při měření hloubky, které používá IR laser

2. Je třeba zachovávat vhodnou vzdálenost mezi senzorem a uživatelem. Maximální vzdálenost je 4 metry, minimální přibližně 50 cm.

4.1.1 Obdobné systémy

Systémy digitální výlohy zapojující snímání uživatele a vyhodnocování jeho akci představují poměrně novou technologii, která zatím není v širším měřítku nasazována, ačkoli je použití Kinectu nebo obecně snímání uživatele pro obchodníky slibné. Nicméně je rozumné očekávat, že se v následujících letech rozšíří. Výhodou je také neustále klesající cena televizních a jiných zobrazovacích zařízení, senzoru Kinect a jeho kopií a konkurentů a poměrně aktivní komunita zabývající se vývojem aplikací s využitím těchto senzorů, která by mohla vytvořit veřejně přístupné Open-Source řešení, které by mohl využít téměř každý technicky orientovaný uživatel.

Širší veřejnosti byl podobný systém představen v pořadu Prizma České televize ze dne 28.7.2012, kde byla jedna z reportáží zaměřena na systém digitální výlohy vyvíjený německými vědci. Reportáž nesla název "Nakupovat 24 hodin denně". Svoji vizi v ní představili výzkumníci z Frauenhoferova Institutu. Systém uměl snímat pohyb ruky a podle ní posouvat zboží na obrazovce. Snímání bylo prováděno buď pomocí 4 kamer, které nabízely více možností díky detailnějšímu obrazu, nebo senzorem Kinect. V tomto případě se jednalo o jednodušší verzi systému. Nicméně se stále jedná o experimentální systém.

V současnosti není příliš používané a neexistují standardizované systémy, které ho poskytují. Systémy jsou zatím většinou ve vývoji.

Kapitola 5

Navržené řešení

Navržený systém sleduje jako základní princip jednoduchost. Cílem systému je poskytnout základní možnost ovládání a prohlížení zboží tak, jak je to běžné v internetových obchodech. Veškerá interakce uživatele se systémem je bezdotyková a snímaná senzorem Kinect. Základní schéma systému je na obrázku 5.1.

Obrázek 5.1: Schema systému

Základními komponentami systému jsou senzor Kinect, počítač s příslušným ovládacím programem a monitor poskytující zpětnou vazbu uživateli. Samotný program je rozdělen na dvě samostatné části:

1. Administrativní aplikace

2. Prezenční aplikace

Administrativní aplikace slouží ke správě systému, uložení produktů do databáze, nahrávání obrázků a mazání. Prezenční program je hlavní částí, která komunikuje s uživatelem.

Obě aplikace využívají technologii WPF (Windows Presentation Foundation). Tato technologie byla uvedena (pod názvem Avalon) v .NET Frameworku 3.0. Je určena k vytváření přívětivějšího a bohatšího uživatelského rozhraní v prostření Windows (někdy bývá označováno jako RUI - Rich User Interface). Na rozdíl od běžně používaného grafického subsystému GDI může být část režie WPF zpracována pomocí DirectX přímo na grafické kartě. Celý kód pro popis WPF komponent je psán ve značkovacím jazyce XAML (eXtensible Application Markup Language). Jednou z důležitých vlastností je oddělení grafického návrhu od programové implementace. To mimo jiné umožňuje použít tzv. *Styly* (*Styles*). Ty jsou standardně uloženy v souboru *App.xaml* lze pomocí nich hromadně definovat vzhled GUI elementů a dosáhnou tak jednoduše a rychle změny vzhledu celého programu. Postup je velmi podobný jako při použití kaskádových stylů (CSS) při designu webových stránek.

5.1 Databáze

Systém používá databázové úložiště pro ukládání položek, které jsou zobrazeny. Samotná databázová aplikace není součástí této práce, předpokládá se její samostatná instalace nebo využití již funkčního serveru. Instalaci lze provést na libovolný počítač, ke kterému je možný přístup po síti nebo přímo na počítač, na kterém je spuštěna aplikace. Aplikace spolupracuje s programem Microsoft SQL Server 2008. Systém musí být nastaven a umožňovat připojení uživatelů, na serveru je nutné zapnout autentizaci SQL serverem a nepoužívat výchozí autentizaci účtem ze systému Windows.

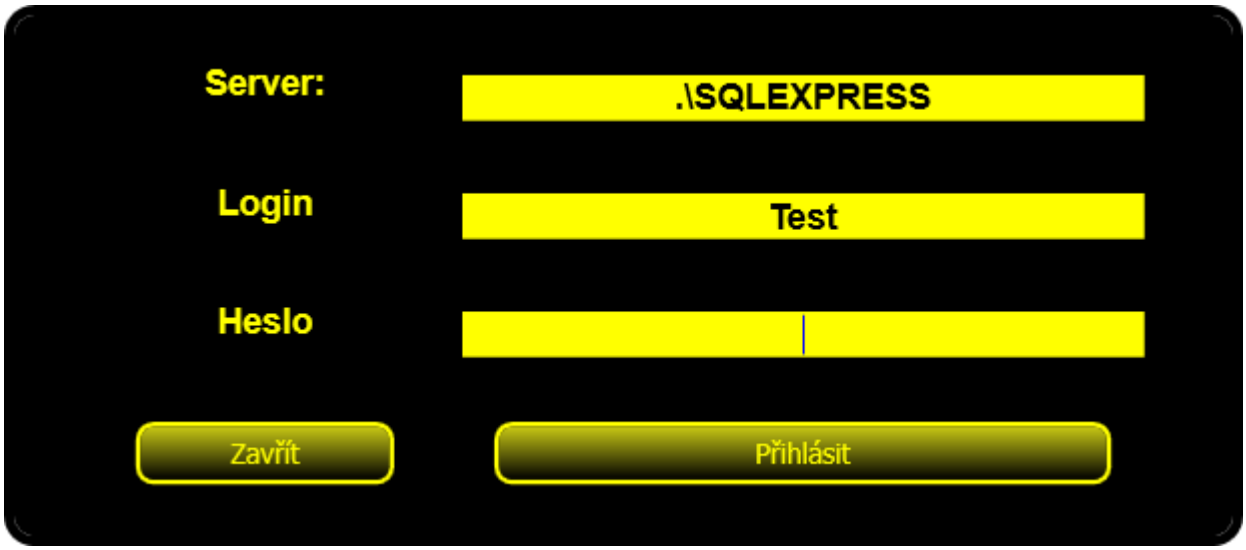
V databázi jsou použity dvě tabulky, jedna z nich ukládá produkty a informace o nich, druhá obsahuje obrázky. Produkty jsou s obrázky propojeny pomocí klíče *Cislo*.

Pro připojovací jsou potřeba následující údaje:

1. Adresa SQL serveru
2. Přihlašovací jméno
3. Heslo

Připojení k databázi je podmínkou pro chod aplikace, proto je při spuštění libovolné z jejích části požadováno přihlášení. Pokud je přihlašovací okno uzavřeno bez zadání údajů, ukončí se i aplikace. Dokud se uživatel úspěšně nepřihlásí, není možné aplikaci používat. Po úspěšném přihlášení je přihlašovací okno uzavřeno a do popředí se dostane spuštěná aplikace.

5.2 Administrativní aplikace



Obrázek 5.2: Přihlašovací okno

Tato aplikace je určena pro nastavení systému. Předpokládané je její využití technickým pracovníkem ideálně po první instalaci systému a při případných úpravách. Slouží k umístění jednotlivých produktů do databáze. Po jejím spuštění je nutné se přihlásit k databázovému serveru. Je proto zobrazeno okno pro zadání přihlašovacích údajů (5.2, kterými jsou adresa serveru, přihlašovací jméno a hesla. Klepnutím na tlačítko přihlásit dojde k přihlášení a okno se zavře (pokud se uživatel rozhodne nepřihlašovat, tlačítkem Zavřít dojde k uzavření okna a ukončení programu). Po úspěšném přihlášení je v horní části okna zobrazeno jméno databázového serveru a uživatelské jméno. Pod nimi je zobrazen seznam položek, při prvním spuštění prázdném.

Každá položka má tyto vlastnosti:

1. Číslo
2. Název

3. Popis
4. Obrázky
5. Rodič (předcházející prvek)

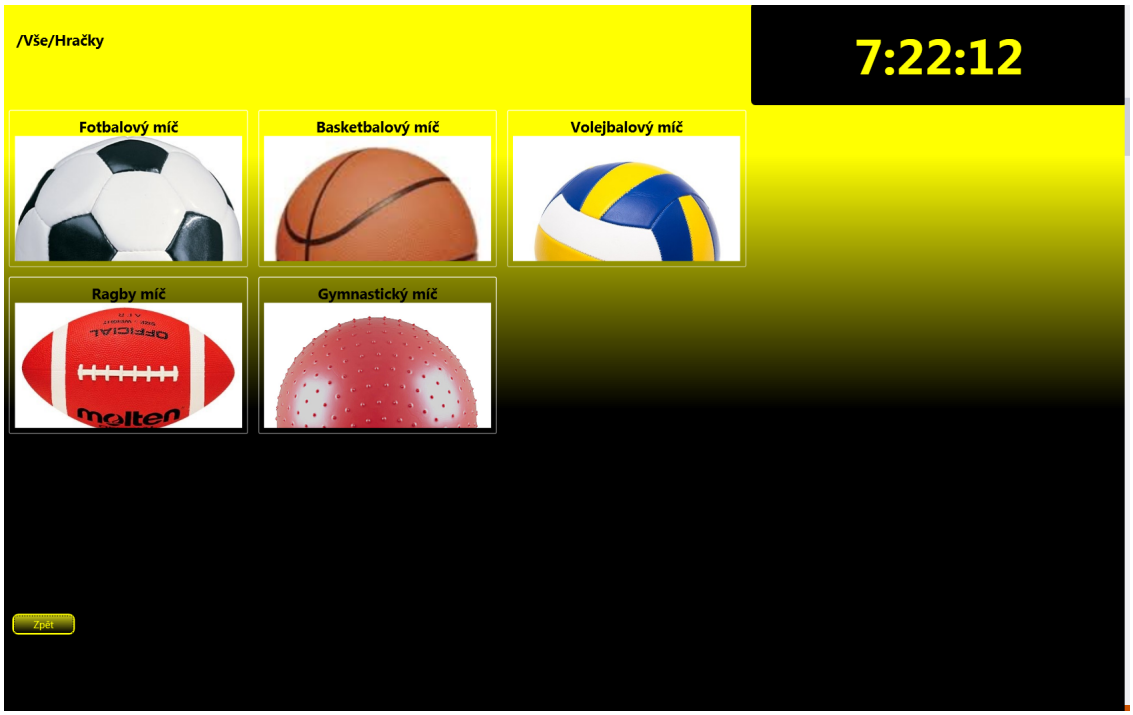
Položky jsou uspořádány do hierarchické struktury, stromu s jedním kořenem, který představuje prvek s hodnotou Rodiče rovnou nule. Každá položka může mít libovolný počet potomků - položek, jejichž je rodičem. U každého takového potomka je pole Rodič nastaveno na Číslo rodiče této položky. Každá nová položka dostane při svém vytvoření přiřazeno unikátní identifikační Číslo, které je neměnné. Na toto číslo se budou odkazovat v poli Rodič všichni potomci této položky. Tím je vytvořena stromová struktura, pro jejíž kompletní načtení stačí postupovat od kořene až do listů. Lze tedy načíst hodnotu Číslo kořene stromu (ten lze najít díky hodnotě 0 v poli rodič). Poté stačí vyhledat položky, jejichž pole rodič je nastaveno na tuto hodnotu a pro každou z nich postupovat stejně jako pro kořenovou položku. Jedná se tedy o rekurzivní algoritmus, který vrátí položky načtené z databáze reprezentované vhodnou strukturou, v tomto případě spojovým seznamem instancí třídy Produkt.

Obrázek 5.3: Aplikace pro administraci systému

Po načtení je tento seznam zobrazen v hlavní části okna 5.3. Jeden řádek odpovídá jedné položce. Pokud je u položky obrázek, je pro snadnou orientaci zobrazen ještě před jejím popisem. Pokud má položka přiřazeno více obrázků, je zobrazen jen první. Poté následuje název položky a její číslo. Za ním následuje popis. Zobrazení je opět hierarchické, kořenový prvek je zobrazen úplně nahoře a jeho potomci pod ním. Pokud je u prvku tlačítko + (plus), má tento prvek potomky, kteří nejsou zobrazeni a kliknutím na toto tlačítko dojde k jejich zobrazení. Naopak tlačítko - (mínus) umožňuje úspornější zobrazení, které po

kliknutí skryje potomky příslušného prvku. Po označení některého z prvků seznamu je možné provést změnu tohoto prvku kliknutím na tlačítko V. Prvek je možné také odstranit nebo přidat nový. Pro rychlejší ovládání je možné použít tlačítka Rozbalit a Sbalit, které zobrazí, nebo naopak skryje, všechny prvky. Tlačítkem Refresh je možné načíst údaje z databáze znovu.

5.3 Prezentační aplikace



Obrázek 5.4: Vzhled hlavního okna

Tato aplikace je určená pro ovládání programu uživatelem - zákazníkem. Po spuštění je uživatel dotázán na připojení k databázi, ze které aplikace načte potřebná data o položkách, které budou zobrazeny. Po načtení se pokusí zahájit komunikaci se senzorem. Pokud v tuto chvíli není senzor k dispozici nebo se ho nepodaří inicializovat, je uživatel upozorněn chybovou hláškou. Hláška zobrazuje jednu z následujících zpráv:

1. *Kinect nenalezen - připojte Kinect a stisknete tlačítko Znovu. Stiskem tlačítka Cancel aplikaci ukončíte.* - Tato chybová hláška se zobrazí, pokud není k počítači připojen žádný Kinect. K jejímu odstranění postačí připojit senzor do USB portu. Připojení je indikováno LED diodou přímo na senzoru.

2. *Vyskytly se potíže v komunikaci se senzorem - zkontrolujte připojení* - Tato chybová hláška se zobrazí, pokud je senzor připojen, ale není ve stavu, ve kterém by byl schopný pracovat. Nejčastějším problémem je pravděpodobně nepřipojený napájecí zdroj (senzor vyžaduje přídavné napájení - napájení z USB stačí na rozblikání LED diody a na základní komunikaci přes USB sběrnici, ale ne na snímání obrazu. Indikace LED diodou je poněkud nešťastná, protože budí dojem plné funkčnosti senzoru).

Po stisku tlačítka Znovu se bude aplikace znovu pokoušet připojit k senzoru a zobrazí upozornění znovu, pokud nebude senzor připojen a nebude s ním možná komunikace. V případě, že uživatel chce tuto smyčku přerušit, lze použít tlačítko *Cancel*, čímž aplikace zanechá marných pokusů a ukončí se.

Pokud je senzor připojen a je plně funkční, zobrazí se hlavní okno aplikace. V něm je zobrazena aktuální nabídka produktů a aktuální čas. Po spuštění aplikace čeká. Ze stavu čekání přechází v případě detekce pohybu v obraze - v tom případě vyzve uživatele k signalizaci (zamávání) rukou. Tato pohyb rukou je detekován a ruka je sledována a ovládá kurzor.

Ukončení aplikace se provede stiskem tlačítka *Excape* v hlavním okně.

5.3.1 Součásti aplikace

Samotná aplikace se skládá z množství tříd, funkčně nejdůležitějšími z nich jsou tyto:

1. Okno přihlášení k databázi
2. Hlavní Okno
3. Kinect Control Class
4. Database Connection
5. Nastaveni
6. Setting Windows

5.3.2 Třída Kinect Control Class

Tato třída má na starosti komunikaci se senzorem. V konstruktoru dostane jako parametr požadované rozlišení a vnitřní nastavení, a čeká, dokud není zavolána metoda StartSensor. Pokud se připojení k senzoru nepovede, je uživatel upozorněn chybovou hláškou a

pokud nepovolí další opakování, je vyhozena výjimka, která je zpracována ve třídě hlavního okna a ukončí program. Pokud je senzor připojen úspěšně, dojde k zahájení načítání dat. Tato třída obstarává také detekci ruky a její sledování. S hlavním oknem komunikuje pomocí událostí, konkrétně používá událost *NeWCommandEvent*, která předává posluchačů data ve svých argumentech. Ty jsou celkem tři: první je určen výčtovým typem *CommandList* a určuje, o jakou událost se jedná:

- ZmenaPozice - informuje o změně polohy ruky uživatele.
- Prikaz_Zpet - navrácení o úroveň výš v seznamu zboží
- Prikaz_Next - zobrazit další obrázek k vybrané položce
- Prikaz_Previous - zobrazit předchozí obrázek k vybrané položce
- Prikaz_Koupit - vloží aktuální produkt do košíku
- Prikaz_Oznameni - Oznámení, že má uživatel zamávat
- Ping - slouží jako testovací událost, nepředává žádnou informaci

Zbylé dva argumenty jsou typu *int* a určují posunutí kurzoru v ose x a y. Příjemce události je bere v potaz jen, pokud je událost typu *ZmenaPozice*.

5.3.3 Hlavní okno

Tato třída představuje hlavní viditelnou část aplikace. Po spuštění programu zobrazí okno, které je zavřeno až pro ukončení programu a provází tak uživatele po celou dobu. Aby byla zvýšena jeho informační hodnota, je v pravém horním rohu zobrazen aktuální čas. V jeho hlavní části je zobrazen seznam položek tak, jak byly staženy z databáze. Po spuštění se nachází v nejvyšším "patře", a obsahují nejvyšší podkategorie. Okno se stará o správné zobrazení vybrané položky a vykreslení jejích potomků. Zpět v seznamu prvků je možné jít kliknutím na tlačítko *Zpět*. Po stisku klávesy F10 se zobrazí okno s nastavením programu. Také zpracovává události od ostatních částí aplikace, viz obrázek 5.5.

5.3.4 Třída DatabaseConnection

Tato třída slouží ke komunikaci s databází. Její konstruktor je bezparametrový, připojení se prování metodou *ConnectToDB*, jejímž parametry jsou jméno serveru, přihlašovací

Obrázek 5.5: Tok zpráv v aplikaci

jméno a heslo. Mezi nejdůležitější metody patří bezparametrová metoda NactiStrom(), která vrátí strom produktů. Po ukončení práce je nutné provést odpojení od databáze, k čemuž slouží metoda Odpojit(). Tato třída je využita v prezenční i administrativní aplikaci.

5.3.5 Třída Nastavení

Tato třída je použita pro předávání parametrů nastavení. Jedná se o referenční typ umístěný na haldě, takže je možné na jednu instanci odkazovat z více míst programu. Její úprava se provádí v okně nastavení, odkud je po uložení nastavení pomocí událostí zavolána metoda třídy MainWindow, která dále volá metodu třídy KinectControlClass, která nastavení použije.

Obsahuje nastavení pro nejdůležitější parametry detekce (pokud není uvedeno jinak, jedná se o celočíselné proměnné):

- T1 - čas mezi minulým snímkem a vytvořením nového snímku
- T2 - délka intervalu detekce ruky
- ShowWin - promená typu bool, určuje, jestli se zobrazí informační okno s obrazem kamery
- depthMin - spodní mez hloubky
- depthMax - horní mez hloubky
- MaxPortion - proměná typu double, určuje podíl jasu bodu z jasu maxima, kdy je bod ještě přidán do oblasti okolo ruky.
- MaxWinSizeX - Maximální velikost okna okolo nalezené polohy ruky
- MaxWinSizeY - Maximální velikost okna okolo nalezené polohy ruky

- MinWinSizeX - Minimální velikost okna okolo nalezené polohy ruky
- MinWinSizeY - Minimální velikost okna okolo nalezené polohy ruky

5.3.6 Třída Setting Windows

Tato třída umožňuje uživateli zobrazit okno (na obrázku 5.6), ve kterém má možnost nastavit parametry programu (viz 5.3.5). Zobrazí se po stisku klávesy F10 v hlavním okně. Zadaná data jsou potvrzena stiskem tlačítka *Použít* nebo zahozena tlačítkem *Zahodit*. Po potvrzení tlačítkem *Použít* se změny projeví okamžitě.

Obrázek 5.6: Okno pro nastavení parametrů detekce

5.3.7 Sledování ruky

Sledování, označované také jako Tracking nebo Template Matching, představuje jeden z nejrozšířenějších postupů v počítačovém vidění. Má mnoho využití, například v oblasti bezpečnosti, komprese videa, rozšířené reality a nejčastěji se v běžném životě můžeme s jejími výsledky setkat při sledování filmů, ve kterých se používá k vytváření triků. Zabývá se vyhledáním obrázku (vzoru) v jiném obraze. Zpravidla bývá vzor v porovnání s prohledávaným obrazem velmi malý. Cílem je určit pozici vzoru nebo rozhodnout, že se

v obrázku nenachází. Pokud jsou obrázky snímky z videosekvence, je možné v každém následném obrázku předpokládat polohu vzoru blízko pozice v minulém obrázku.

Jedná se o úlohu, kterou můžeme řešit mnoha způsoby:

1. **Transformací do frekvenční oblasti pomocí Fourierovy transformace** - Tento postup použije Rychlou Fourierovu transformaci (FFT) na obrázek, ve kterém hledáme vzor i na tento vzor. Vzniknou tak frekvenční reprezentace obou, které jsou mezi sebou vynásobeny a pomocí zpětné transformace převedeny zpět na prostorový obrázek. V tomto obrázku budou místa shody vzoru s obrazem intenzivnější než okolí, proto stačí najít maximu.

2. **Výpočtem shody v prostorové oblasti**. Tato metoda je intuitivnější a je možné si ji představit jako přikládání hledaného vzoru na místo v prohledávaném obraze a výpočet shody. Toto lze provádět buď v barevném obraze pro každý barevný kanál, nebo převézt obraz na černobílý například známým vzorcem $Y = 0,299R + 0,587G + 0,114B$ [7]. Klíčová je zde funkce, která počítá podobnost vzoru a části obrazů, nad kterým se nachází. Podobnost je reprezentována jako číslo, které určuje míru shody, a hledáme místo, na kterém je maximální (alternativně můžeme místo podobnosti použít rozdílnost a hledat místo, kde bude minimální). Obvykle se používá L1 norma, L2 norma nebo korelace.

3. **KLT tracker** - Algoritmus navržený T. Kanadem a B.Lucasem roku 1981 je iterační, v každé iteraci gradientní metodou přiblíží hledaný vzor blíže jeho skutečné poloze v obraze. Ukončení se provede po dosažení maximálního počtu iterací, nebo pokud je posuv menší než nastavený limit.

V takto vytvořeném obraze je následně provedena detekce ruky uživatele. Uživatel je vyzván k zamávání rukou. Tento postup používá více podobných aplikací. Po zobrazení zprávy začne aplikace sčítat rozdíly mezi hodnotami pixelů v minulém a aktuálním snímku. Rozdíl pro každý pixel je přičten k součtu všech rozdílů pro tento pixel. Vznikne tak pole o stejných rozměrech jako snímaná scéna, ve kterém je na každé pozici součet rozdílů jasů na této pozici mezi minulými k snímky. Z tohoto pole jsou následně vymazány pixely s příliš malou hodnotou a osamělé pixely, které ve svém 8-okolí nemají další nenulový pixel. Zbylé prvky pole jsou kopulativně sečteny (podle rovnice 5.1).

$$i_{(a,b)} = i_{(a,b)} + i_{(a-1,b)} + i_{(a,b-1)} - i_{(a-1,b-1)} \tag{5.1}$$

Takto upravené pole obsahuje nejdůležitější informace na svých okrajích, proto je dále zpracován jen poslední sloupec a řádek. Nejprve je spočítán rozdíl sousedních hodnot a v takto upraveném sloupci/řádku je nalezen maximální prvek. V jeho okolí je určen interval, ve kterém bude tato oblast ještě považována za relevantní. Složením intervalu určeného ze sloupce a řádku vznikne obdélníkové okno, které označuje detekovanou ruku (obrázek 5.8).

Aplikace nevyužívá možnosti, které poskytuje Skeleton Stream, ale vlastní detekci polohy ruky. Základem tohoto postupu je sloučení obrazu z hloubkové a barevné kamery pomocí homografie. Ta je určena maticí H. Ke jejímu výpočtu byla použita funkce *hom_id* v Matlabu (na přiložené CD):

$$K = \begin{bmatrix} 0,93 & 0,0509 & 29,1248 \\ -0,031 & 1,096 & 25,0065 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

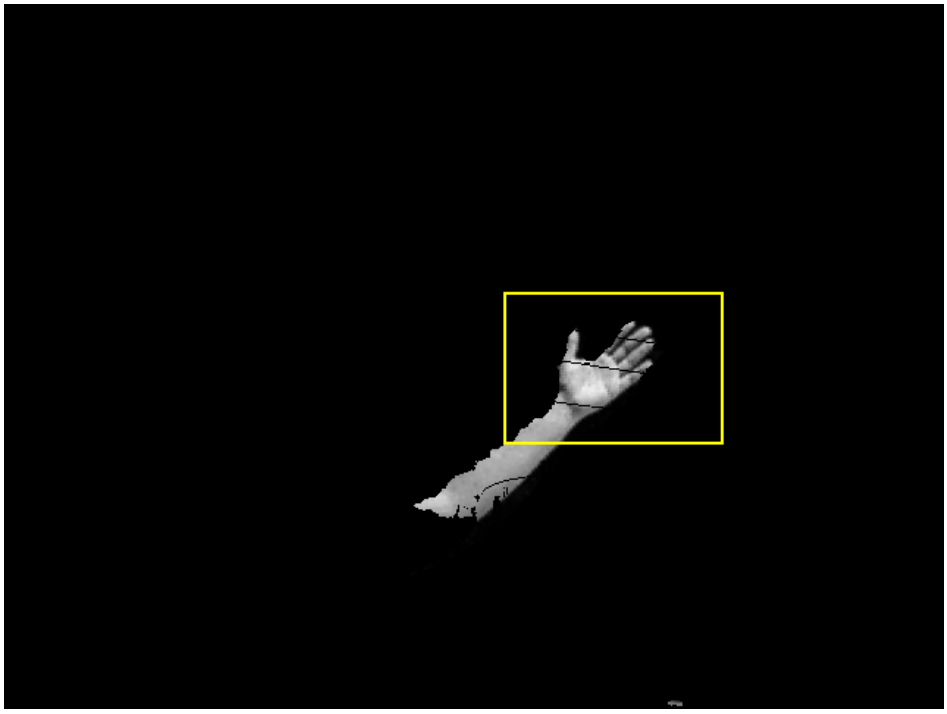
Díky této homografii je možné spojit oba obrazy v jeden. Umožňuje určit přibližnou polohu bodu z jednoho obrazu ve druhém. Toho je využito k "vymaskování"částí obrazu barevného v místech, kde hloubkový obraz přesahuje nastavené hranice vzdálenosti, tedy ty části, které se nacházení příliš daleko nebo naopak blízko. Vhodnou volbou horní a dolní meze lze skrýt části, které jsou nezajímavé a ponechat jen postavu uživatele. Příklad tohoto postupu je na obrázku 5.7.

Samotná homografie není pro každý bod počítána, ale je předpočítána po startu programu pro všechny možné souřadnice pixelů v obrázku.

5.4 Ovládání

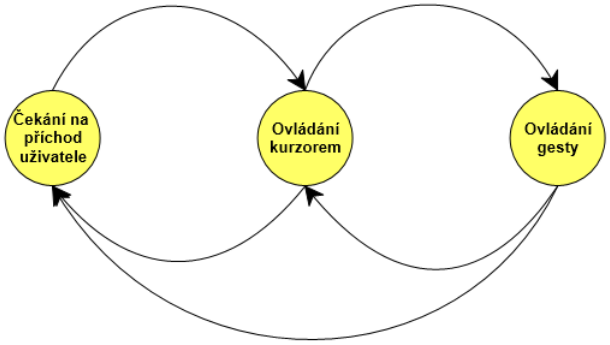
Ovládání aplikace pracuje ve dvou režimech. Oba jsou ovládány pomocí Kinectu, ale každý umožňuje provádět jiné akce. Do prvního režimu se aplikace přepne po detekci uživatele a jeho ruky a umožňuje posouvat kurzor rukou. Potvrzení výběru tlačítka se provede najetím kurzorem na toto tlačítko a ponecháním kurzoru nad ním po určitou dobu. Postupné plynutí této doby je vhodné indikovat buď změnou tlačítka nebo kurzoru [2]. Vhodná je animace, která zobrazuje zbývající dobu do potvrzení akce například pomocí *progress baru* (GUI prvek, ve kterém se posouvá sloupec od nulové ke koncové pozici, slouží obvykle k indikaci zpracované části úlohy). V případě, že uživatel vybere položku, která nemá žádné potomky, tj. nejedná se o kategorii, je zobrazen detail položky. V této chvíli se aplikace přepíná do druhého režimu ovládání, ve kterém není ovládána kurzorem, ale

Obrázek 5.7: Přiklad vyextrahování postavy z obrazu



Obrázek 5.8: Detekce a označení ruky

gesty. Tyto gesta jsou 4:



Obrázek 5.9: Stavový diagram režimů aplikace

1. Zpět na seznam položek - pohybem nahoru
2. Další obrázek - pohybem vpravo
3. Předchozí obrázek - pohybem vlevo
4. Vybrat a umístit zboží do košíku - pohybem dolů

Přechod mezi režimy je zobrazen na obrázku 5.9

5.5 Porovnání s SDK

Pro použitelnost této aplikace je stěžejní rozpoznání uživatele a sledování pohybu jeho ruky. Toto je řešeno vlastními metodami, které využívají jen čtení obrazových dat ze senzoru. Použité SDK nabízí ale navíc ještě Skeleton Stream, který dokáže identifikovat hráče a sledovat 20 bodů na jeho těle, mezi nimiž jsou i ruce. Tento postup se ukazuje rychlejší, spolehlivější a přesnější, než metoda použitá v této práci.

5.6 Vylepšení

V době psaní této práce není program ještě úplně funkční, proto do seznamu vylepšení je možné zahrnout i komponenty, které je třeba přidat, aby byly naplněny očekávání. Jedná se hlavně o:

1. Detekce osoby - protože není tato část implementována, není možné automaticky spustit detekci polohy ruky uživatele

2. Vylepšit sledovací algoritmus. Současná verze je příliš jednoduchá a má velké nároky na výpočetní výkon.

3. Přidat možnost adaptivně měnit vyhledávaný výřez obrázku podle nových dat ze současného obrazu. Po nalezení polohy v novém obraze by bylo vhodné vyhledávaný vzorek mírně upravit tak, aby reflektoval změny, které mohly nastat v průběhu sledování.

4. Košík produktů by mělo být možné v nějaké formě exportovat z aplikace, například zasláním emailu nebo uložením do souboru.

Kapitola 6

Závěr

Závěrem bych rád zdůraznil několik věcí: Senzor Kinect je velmi zajímavé zařízení, které může způsobit revoluci v ovládání výpočetní techniky. Díky své poměrně nízké ceně (v současné době přibližně 2600Kč, což představuje 10% průměrné hrubé mzdy v České Republice) je dostupné téměř každému. Techničtí entuziasté a některé firmy vytvořily jak zajímavé aplikace, tak vlastní Kinect API pro všechny nejpoužívanější operační systémy, čímž značně ulehčili vývoj s použitím senzoru. Samotné Kinect for Windows SDK použité v této práci nabízí také velmi dobré možnosti a díky podpoře Microsoftu je stále aktualizováno na novější verze, takže ho mohu jen doporučit. Naopak použití vlastních algoritmů detekce obrazu není podle mého názoru v obecném případě efektivní v porovnání s výkonem, robustností a jednoduchým použitím algoritmů vestavěných v SDK. Pro speciální aplikace určené do nestandardních podmínek toto platit nemusí.

Díky nepoužití algoritmů nabízených SDK a vývoji vlastního algoritmu, který byl delší, než jsem očekával, nejsou některé části implementovány tak, aby byla aplikace naprosto funkční.

Cílem práce byl vývoj aplikace pro použití v praktickém prostředí, proto je aplikace rozdělena do dvou částí, z nichž jedna slouží k nastavení systému, druhá představuje uživatelskou aplikaci, kterou stačí jen spustit, nastavení není potřeba. Pokud by bylo nutné provést nastavení parametrů, je to i v této aplikaci možné pomocí "skrytého"menu. Ovládání bylo navrženo tak, aby bylo maximálně jednoduché a jasné. Pro uložení prezentovaných položek byla zvolena databáze, která zajišťuje snadnou možnost přístupu po počítačové síti. Aplikace vyžaduje pro svou činnost senzor Kinect.

Seznam obrázků

2.1 Konzole Xbox360 3
2.2 Senzor Kinect 4
2.3 Kinect po demontáži krytu 6
2.4 Obraz promítaný IR projektorem 7
2.5 Schéma promítání a snímání IR obrazu [Zdroj Microsoft.com] 8

3.1 Závislost změřené vzdálenosti na skutečné vzdálenosti objektu od senzoru . 14
3.2 Jointy - body, které senzor detekuje na postavě uživatele [Zdroj Microsoft.com] 15
3.3 Kresba kostry z bodů Skeleton Streamu [Zdroj Microsoft.com] 16
3.4 Výpočet souřadnic bodu X v obrazové rovinně [Převzato z [3]] 18
3.5 Projekce 20

5.1 Schema systému 26
5.2 Přihlašovací okno 28
5.3 Aplikace pro administraci systému 29
5.4 Vzhled hlavního okna 30
5.5 Tok zpráv v aplikaci 33
5.6 Okno pro nastavení parametrů detekce 34
5.7 Přiklad vyextrahování postavy z obrazu 37
5.8 Detekce a označení ruky 37
5.9 Stavový diagram režimů aplikace 38

Literatura

[1] Camera Calibration Toolbox for Matlab, http://www.vision.caltech.edu/bouguetj/calib_doc, 2010

[2] Human Interface Guidelines, http://download.microsoft.com/download/B/0/7/B070724E-52B4-4B1A-BD1B-05CC28D07899/Human_Interface_Guidelines_v1.7.0.pdf, 2013

[3] Khoshelham, Accuracy Analysis of Kinect Depth Data, dostupné z https://wiki.rit.edu/download/attachments/52806003/ls2011_submission_40.pdf

[4] Color Stream, http://msdn.microsoft.com/en-us/library/jj131027.aspx

[5] Depth Stream, http://msdn.microsoft.com/en-us/library/jj131028.aspx

[6] Skeleton Stream, http://msdn.microsoft.com/en-us/library/microsoft.kinect.skeletonstream.aspx

[7] Wikipedia, článek Grayscale, http://en.wikipedia.org/wiki/Grayscale