Software pro určování standardizované velikosti postavy s využitím sensoru Microsoft Kinect

Standard Clothing Size Identification Software using Kinect Sensor

Bc. Tomáš Zaoral

Diplomová práce
2013

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2012/2013

ZADÁNÍ DIPLOMOVÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Tomáš Zaoral**
Osobní číslo:	**A11453**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Počítačové a komunikační systémy**
Forma studia:	**prezenční**
Téma práce:	**Software pro určování standardizované velikosti postavy s využitím sensoru Microsoft Kinect**

Zásady pro vypracování:

1. **Zpracujte literární rešerši na téma práce s Microsoft Kinect.**
2. **Analyzujte možnosti určování standardizované velikosti postavy s využitím sensoru Microsoft Kinect.**
3. **Navrthněte vhodný způsob měření a vyhodnocení velikosti postavy.**
4. **Vytvořte ukázkovou aplikaci.**
5. **Demostrujte výsledky a formulujte závěr.**

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **PIALORSI, Paolo a Marco RUSSO. Microsoft LINQ: kompletní průvodce programátora. Vyd. 1. Brno: Computer Press, 2009, 615 s. ISBN 978-80-251-2735-3.**
2. **NAGEL, Christian. Professional C 4 and .Net 4. Indianapolis, IN: Wiley Pub., c2010, 1474 p. ISBN 04-705-0225-8.**
3. **WATSON, Ben. C 4.0: řešení praktických programátorských úloh. Vyd. 1. Brno: Zoner Press, 2010, 656 s. Encyklopedie Zoner Press. ISBN 978-80-7413-094-6.**
4. **BORENSTEIN, Greg. Making things see: 3D vision with kinect, processing, Arduino, and MakerBot. Sebastopol,: OReilly, c2012, xviii, 416 s. ISBN 978-1-4493-0707-3.**
5. **ASHLEY, Jarrett Webb and James. Beginning kinect programming with the microsoft kinect SDK. New York: Apress. ISBN 978-143-0241-041.**

Vedoucí diplomové práce: **Ing. Erik Král**
Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce: **26. února 2013**

Termín odevzdání diplomové práce: **31. května 2013**

Ve Zlíně dne 26. února 2013

L.S.

prof. Ing. Vladimír Vašek, CSc.
děkan

prof. Ing. Karel Vlček, CSc.
ředitel ústavu

ABSTRAKT

Diplomová práce se zabývá určováním standardizované velikosti oblečení lidské postavy s využitím senzoru Microsoft Kinect. Obsahem práce je seznámení se se senzorem Microsoft Kinect, analýza jeho možností při určování velikosti lidské postavy a návrh vhodného způsobu měření lidské postavy. Hlavní výstup práce je ukázková aplikace, která využívá všechny aspekty senzoru Microsoft Kinect a která provede vyhodnocení velikosti postavy měřeného člověka.

Klíčová slova: senzor Kinect, postava, měření, ukázková aplikace

ABSTRACT

Diploma thesis deals with standard clothing size identifications using Microsoft Kinect sensor. Aim of this thesis is to introduce the Microsoft Kinect sensor, the analysis of its options in determining the size of the human figure and design a suitable method for measuring human figure. The main thesis output is a sample application that uses all aspects of the Microsoft Kinect sensor, which will access the magnitude of the measured human figures.

Keywords: Kinect sensor, figure, measuring, sample application

Rád bych poděkoval vedoucímu diplomové práce, Ing. et Ing. Eriku Královi, za odborné vedení, cenné rady a věcné připomínky během řešení této práce a také své rodině a přátelům za podporu.

Motto

„Non schoale sed vitae discimus. - Neučíme se pro školu, ale pro život.“

Seneca

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně …….

podpis diplomanta

OBSAH

ÚVOD

V moderní době jako je ta dnešní, kdy jsou počítače nedílnou součástí každodenního života, stále vznikají nové způsoby jejich ovládání jako například dotykové obrazovky. Technologie jde ale ještě dále a díky zařízením jako je senzor Microsoft Kinect se vašeho zařízení nemusíte ani dotýkat, abyste jej mohli ovládat. Senzor Microsoft Kinect byl prvně dostupný pouze pro herní konzoli Xbox 360 jako doplněk ovládání. Původní určení bylo čistě pro ovládání her. Jak je patrné dnes, využití senzoru Microsoft Kinect nekončí v herním průmyslu, ale lze jej využít i v jiných odvětvích, například v oděvním průmyslu.

Tato práce se zabývá vytvořením ukázkové aplikace za využití senzoru Microsoft Kinect, která bude schopna změřit velikost lidské postavy a její míry. Výstupem ukázkové aplikace bude charakteristika postavy s doporučením vhodných velikostí oblečení. Uplatnění takového typu aplikace můžeme nalézt například v běžném kamenném obchodě nebo v internetovém obchodě zabývajícím se prodejem oblečení.

V teoretické části této práce je popsána historie a struktura senzoru Microsoft Kinect. Dále jsou zde popsány možnosti a omezení senzoru a způsob jeho ovládání. Poslední kapitola teoretické části je zaměřena na Kinect SDK a jeho hlavní prostředky, kterými lze přistupovat k datům získaných z jednotlivých kamer senzoru Microsoft Kinect.

V praktické části v úvodu rekapituluji veškeré dostupné prostředky Kinect SDK pro měření lidské postavy a navrhuji vhodnou metodiku pro toto měření, která je využita v ukázkové aplikaci. Následuje popis programové části, ve kterém jsou na ukázkách zdrojových kódu vysvětleny způsoby, jakými jsem řešil programovou část aplikace. Rovněž se zde nachází popis obsluhy aplikace a ukázka uživatelského prostředí ukázkové aplikace. V závěru praktické části se nachází vyhodnocení měření postav pomocí ukázkové aplikace, ve kterém lze vidět, s jakou přesností probíhá měření u různých typů postav a zda doporučené velikosti oblečení odpovídají skutečnosti.

I. TEORETICKÁ ČÁST

1 SENZOR MICROSOFT KINECT

V roce 2007 Bill Gates, předseda představenstva a zakladatel společnosti Microsoft, pojednává na konferenci D5 o myšlence ovládání her, založené na snímání lidského těla pomocí kamery, kde lidé mohou používat reálné objekty, jako jsou tenisové rakety nebo baseballové pálky ke kontrolování toho, co se stane na obrazovce zařízení. Později v roce 2009 Microsoft na konferenci E3 oznámil, že pracuje na projektu nesoucí kódové označení „Project Natal“. Součástí oznámení byla i první ukázka technologie v akci, která zahrnovala konzoli Xbox 360 a hraní Ricochetu. O rok později „Project Natal“ dostal název „Kinect“ a v listopadu 2010 byl oficiálně vypuštěn. [1] Název Kinect je inspirován slovem “kinetic”, což znamená být v pohybu a slovem “connect”, což v tomto kontextu znamená spojení se s přáteli.[2]

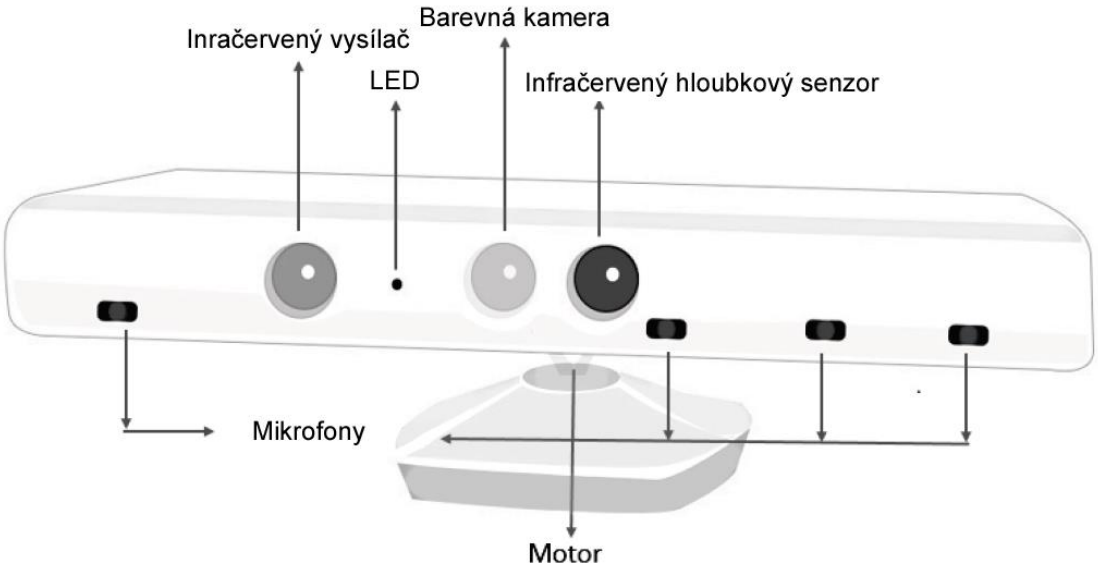
Ve skutečnosti senzor Kinect není produkt vyvinutý pouze společností Microsoft. Za jeho zrodem stojí také společnost PrimeSense, která úzce spolupracovala s Microsoftem a která má na svědomí hardwarovou část zařízení. [3, s. 6]

Historie zařízení Microsoft Kinect se však datuje daleko před dobu jeho koncipování. Samotná myšlenka Kinectu má kořeny v přelomu 21. století, kdy lidé přemýšleli a snili o uživatelském rozhraní založené na ovládání pomocí gest a hlasu. V roce 2002 světově známý hollywoodský režisér Steven Spielberg natočil film Minority Report a právě tento film dal podnět k futuristickému uživatelskému rozhraní. Kinect se stal podobnou revoluci v technologii, jako byla například premiéra osobního počítače nebo internetu v minulém století. [3, s. 1] Taková technologie, která byla dříve dostupná pouze armádě nebo tajným organizacím, nyní připadla do rukou běžným lidem. Kinect byl původně určen pro ovládání her spolu s herní konzolí Xbox 360. Ovšem výhod senzoru Kinect se dnes začíná využívat i v ostatních odvětvích než je herní průmysl. A to díky tomu, že Microsoft vydal pomůcky, potřebné pro práci senzoru Kinect s osobním počítačem. Vydáním pomůcek jako Kinect for Windows SDK otevřelo cestu mnoha vývojářům a inovativním nápadům.

1.1 Struktura senzoru

Senzor Kinect tvoří 4 důležité části, které jsou potřeba pro správnou funkčnost. Těmito částmi jsou barevná kamera, infračervený hloubkový senzor, infračervený vysílač a

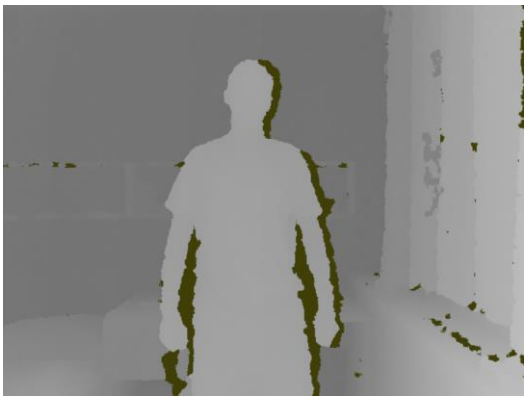
čtveřice mikrofonů. Na čelní straně senzoru Kinect se dále nachází LED dioda a celé tělo senzoru Kinect je umístěno na motorizované základně (Obr. 1).



Obr. 1. Popis jednotlivých částí senzoru Kinect [5, s. 9]

1.1.1 Infračervený vysílač a hloubkový senzor

Infračervený vysílač a hloubkový senzor spolu vzájemné spolupracují a vytvářejí tzv. hloubkový obraz (Obr. 2). Data hloubkového obrazu tvoří jednotlivé obrazové body, tzv. pixely, které nabývají hodnoty 0 až 255. Hodnota se počítá na základně výpočtu vzdálenosti promítaných infračervených vzorů od senzoru Kinect. Tyto vzory produkuje infračervený vysílač a vzdálenost se vyhodnotí hloubkovým senzorem. Maximální podporované rozlišení je 640 x 480 obrazových bodů.



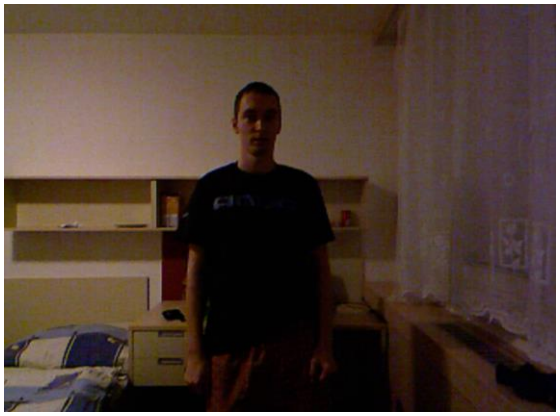
Obr. 2. Hloubkový obraz

1.1.2 LED indikátor

LED dioda je umístěna mezi infračerveným vysílačem a barevnou kamerou. Jejím úkolem je dávat informaci o stavu, v jakém se zařízení nachází. Pokud LED dioda svítí zeleně, znamená to, že zařízení je připravené k použití. V případě, že LED dioda problikává červeně, zařízení není připravené nebo potřebuje připojit k externímu zdroji napájení, aby mohlo plně fungovat.[5, s. 15]

1.1.3 Barevná kamera

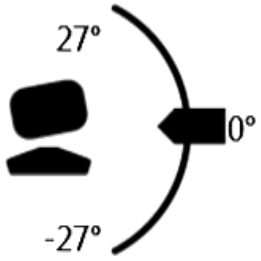
Účelem barevné kamery je sbírání RGB dat. Tyto data jsou následně zpracována a přenášena na výstupní zařízení ve formě posloupnosti po sobě jdoucích statických obrazů. Barevná kamera podporuje až 30 obrazů za sekundu při rozlišení 640x480 pixelů a 12 obrazů za sekundu při maximálním rozlišení 1280x960 pixelů.[5, s. 10]



Obr. 3. Barevný obraz

1.1.4 Motor

Tělo senzoru Kinect je spojeno se základnou malým motorem. Motor může naklonit tělo senzoru Kinect vertikálně o 27 stupňů nahoru nebo dolů. [5, s. 14] Tato schopnost může být velice užitečná při programování aplikací. Například u aplikace, která je založena na sledování osoby a vyžaduje, aby sledovaná osoba byla v zorném poli senzoru Kinect, může se využít motoru k naklonění těla senzoru a přizpůsobení se pozici sledované osoby.



Obr. 4. Úhel náklonu
senzoru pomocí motoru

1.1.5 Mikrofony

Na spodní straně senzoru Kinect je čtveřice odlišných mikrofonů. Jeden mikrofon se nachází v levé části a zbylé tři jsou rovnoměrně rozloženy v pravé části. Hlavní výhodou této sady mikrofonů je, že zachycení a rozpoznání lidského hlasu se provádí efektivněji a s větším potlačením šumu a ozvěny. Účelem těchto mikrofonů není pouhé zachycení zvuku, ale slouží i pro lokalizování směru, odkud zvuková vlna přišla. Lokalizace směru spočívá v porovnání zachyceného stejného zvuku z každého ze čtyř mikrofonů a následném zjištění, který mikrofon zachytil zvuk nejsilněji. [5, s. 15]

Tato technika se využívá i k soustředění všech mikrofonů na jeden určitý směr. Proto jsou tyto mikrofony důležitou součástí celého zařízení Microsoft Kinect.

1.2 Možnosti a omezení senzoru

Žádné zařízení není dokonalé a u senzoru Kinect tomu není jinak. Aby vše správně fungovalo, musí být zachovány určité podmínky. V této kapitole jsou popsány veškeré hlediska, která se musí brát v potaz při práci se senzorem Kinect. Mezi tyto hlediska patří – pozorovací úhly, rozsah vzdálenosti, detekce a sledování a rozsah zvuku.

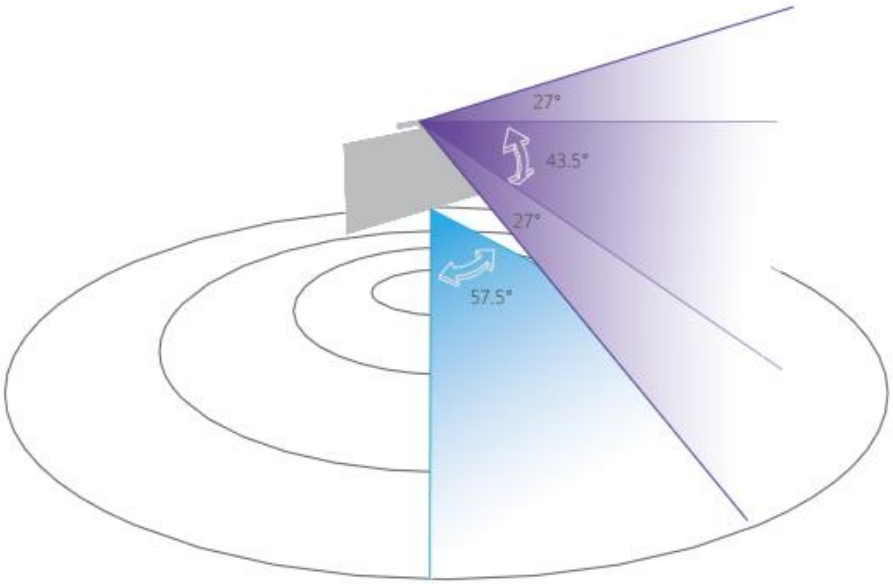
1.2.1 Pozorovací úhly

Očima senzoru Kinect jsou barevná kamera a hloubkový senzor. Tak jako člověk vidí věci před sebou v určitém zorném poli, je limitován zorným úhlem, tak i senzor Kinect má vlastní pozorovací úhly. Pozorovací úhly jsou:

- horizontální: 57,5 stupně,

- vertikální: 43,5 stupně s rozsahem sklonu 27 stupně nahoru nebo dolů

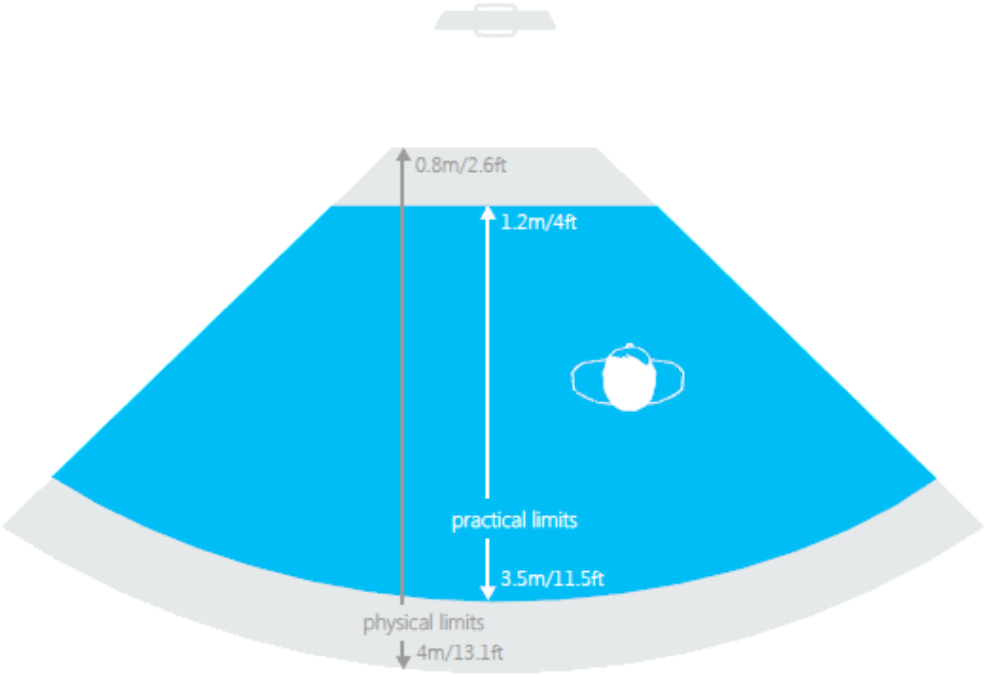
Nachází-li se pozorovaný objekt mimo tyto pozorovací úhly, nebude zaznamenán.



Obr. 5. Pozorovací úhly senzoru Kinect [6]

1.2.2 Rozsah vzdálenosti

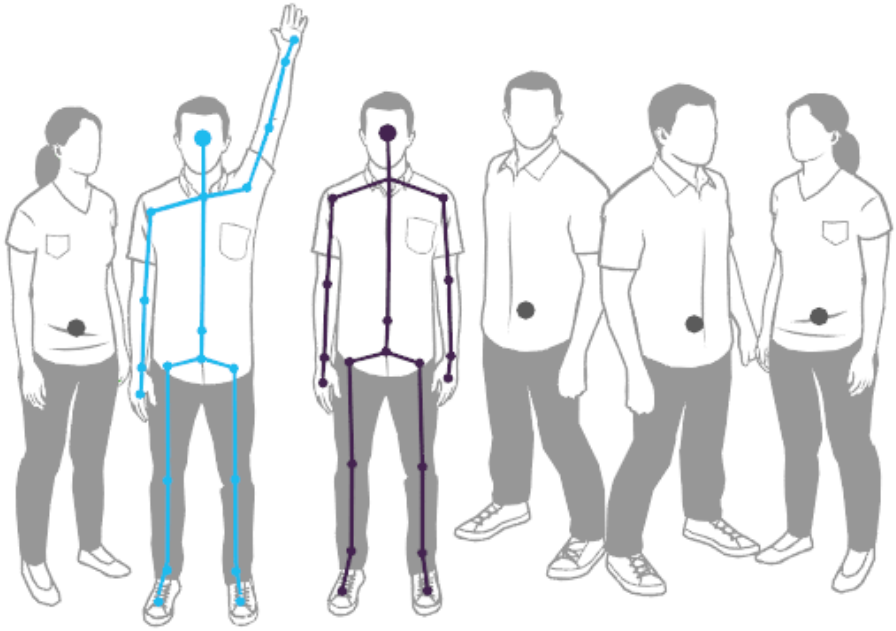
Rozsah vzdálenosti u senzoru Kinect je další z řady důležitých parametrů ovlivňující jeho správnou funkčnost. Tak jako musí být dodrženy správné pozorovací úhly, musí být dodržena i určitá vzdálenost sledovaného objektu. Fyzická vzdálenost měřeného objektu od senzoru Kinect je 0,8 až 4 metry, což je samotná hranice. Proto se používá praktická vzdálenost, která je 1,2 až 3,5 metru. Bude-li sledovaný objekt mimo tyto vzdálenosti, nebude zaznamenán. [6] Grafické znázornění lze vidět na obr. 6.



Obr. 6. Rozsah vzdálenosti senzoru Kinect [6]

1.2.3 Detekce a sledování

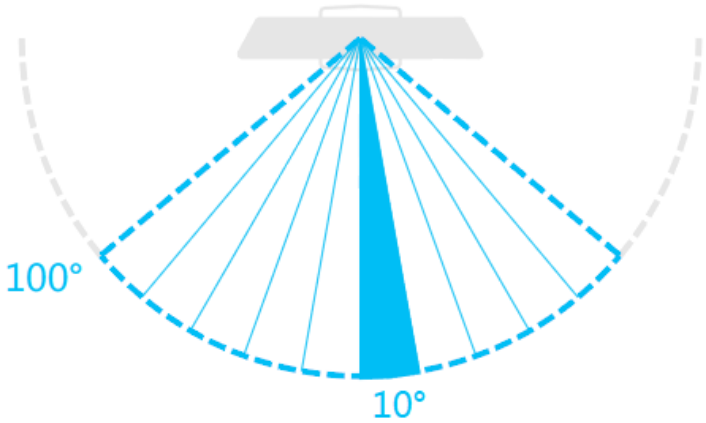
Pojem detekce obecně znamená zjišťování nebo odhalování něčeho skrytého. Mluvíme-li o detekci ve spojení se senzorem Kinect, úkolem detekce je zjistit, zda se nachází nějaký objekt v zorném poli senzoru. Senzor Kinect je schopný detekovat až 6 lidí a z těchto 6 detekovaných lidí dokáže dvě osoby sledovat.[6] Jinak řečeno, provádět interakci pomocí senzoru mohou maximálně 2 osoby. Může se to zdát málo, ale v současné době, kdy je senzor Kinect využíván převážně v herním průmyslu, interakce s dvěma lidmi postačuje. Díky schopnosti detekce by mohl senzor Kinect najít uplatnění jako bezpečnostní kamera, kde by nahrával video pouze v případech, že se někdo pohybuje před senzorem a ušetřil by tak spoustu místa pro data.



Obr. 7. Senzor Kinect dokáže detekovat až 6 lidí [6]

1.2.4 Rozsah zvuku

Mikrofony senzoru Kinect jsou schopny zaznamenat zvuk do úhlu 100 stupňů před samotným senzorem. Přitom se tyto mikrofony dokážou zaměřit a sledovat zvuk na 10 stupních ve zmiňovaném 100 stupňovém rozsahu. [6] Pokud se před senzorem Kinect nachází více zdrojů zvuku, například dvě osoby, sledován bude ten, kdo bude hlasitější. Mikrofony tedy sledují nejhlasitější zdroj zvuku.



Obr. 8. Úhly pro zaznamenání zvuku senzoru

Kinect [6]

1.3 Způsoby ovládání

Prvotní záměr vytvoření senzoru Kinect a jedna z věcí, ve které vyniká a ve které je tak výjimečný a přelomový, je jeho způsob ovládání, nebo spíše ovládání pomocí něj. Tradičně uživatelský vstup přichází z klávesnice nebo myši. Mezi uživatelem a počítačem dochází k interakci právě těmito vstupními zařízeními., který převádí data do počítače, ten je následně zpracuje a vytvoří uživatelský přívětivý vizuální výstup. Je běžné, že každé grafické uživatelské prostředí používá kurzor, který ovládáme myší. S příchodem senzoru Kinect přicházejí i nové způsoby ovládání počítače, kde myš je nahrazena lidskou rukou. Senzor Kinect dokáže sledovat pozici ruky, přesněji pozici dlaně, sledované osoby a na základě této pozice provádět interakci s obrazovkou. Speciálními případy pak je ovládání pomocí gest a hlasu.

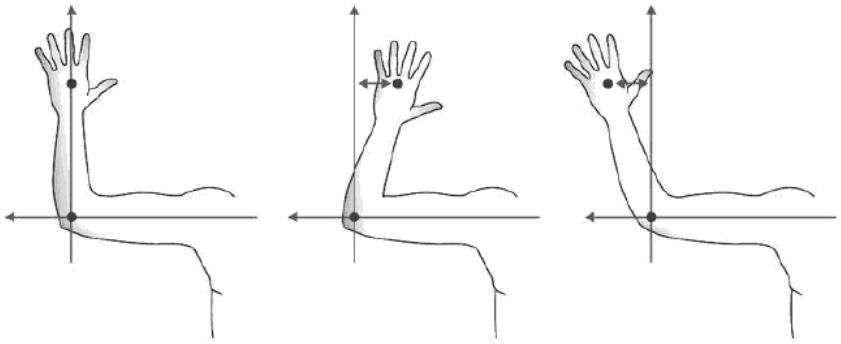
1.3.1 Gesta

V gestickém rozhraní, čistá gesta, pózy a sledování můžou být kombinovány k vytvoření interakčních stylů. Pro senzor Kinect je v současné době osm běžně používaných gest: mávání, tlačítko hover, magnetické tlačítko, tlačítko stisknutí, magnetický slide, univerzální pauza, vertikální rolování a swipe. [4, s. 172]

1.3.1.1 Mávání

Obecně, když někomu máváte rukou, znamená to pozdrav nebo loučení. Použitím gesta mávání u senzoru Kinect dáváte najevo, že jste připraveni a chcete začít používat aplikaci. [4, s. 177]

Mávání je jednoduchý pohyb a rozpoznání tohoto pohybu spočívá v zachycení směru, ve kterém se ruka vyskytuje od neutrální pozice (Obr. 9, vlevo), což je pozice, kdy se ruka nachází v pravém úhlu s ramenem, směrem nahoru. Pokud se ruka pohybuje několikrát sem a tam, zleva doprava, je to vyhodnoceno jako mávání. Důležité je, aby se tento pohyb opakoval několikrát za sebou, jinak gesto nebude považováno za úplné.



Obr. 9. Gesto mávání u senzoru Kinect [4, s. 177]

1.3.1.2 Tlačítko hover

Když kurzor přechází přes tlačítko, sledovaná osoba oznamuje, že chce dané tlačítko vybrat. K vybrání tlačítka dojde tehdy, zůstane-li kurzor nad tlačítkem pár sekund. Jedná se o tzv. hover efekt, který se běžně používá například u webových stránek, s tím rozdílem, že zde kurzor ovládáte pomocí vlastní ruky, bez myši.

1.3.1.3 Magnetické tlačítko

Magnetické tlačítko je nepatrné vylepšení hover tlačítka. Vylepšení spočívá v automatickém zakotvení kurzoru na centrální část tlačítka při přejetí kurzorem přes dané tlačítko. [4, s. 204] Odtud vznikl název magnetické.

1.3.1.4 Tlačítko stisknutí

Tlačítko stisknutí je náhrada za kliknutí na myši. Stisknutí spočívá v zatlačení ruky ve vzduchu vpřed, směrem od sledované osoby.

1.3.1.5 Magnetický slide

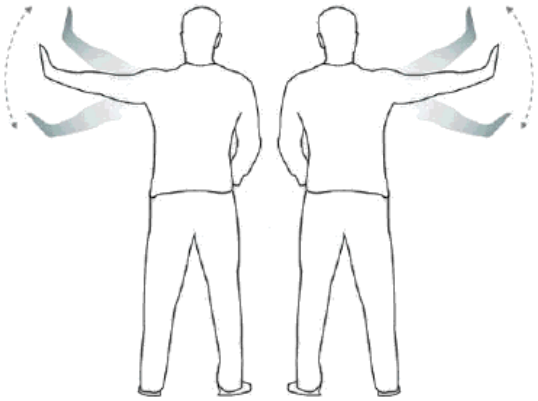
Magnetický slide je kombinací magnetického tlačítka a swipu. Na rozdíl od magnetického tlačítka, které vyžaduje kurzor několik sekund nad tlačítkem, u magnetického slidu stačí pouhé vstoupení do oblasti tlačítka. [4, s. 214]

1.3.1.6 Univerzální pauza

Gesto doporučované společností Microsoft, také známe jako únikové gesto. Tohoto gesta je dosaženo položením levé ruky pod úhlem 45 stupňů od těla. [4, s. 219]

1.3.1.7 Vertikální rolování

Běžně se používá v situaci, kdy se obsah nevleze na celou plochu obrazovku, což vyžaduje posunutí plochy. U konzole Xbox 360 se vertikální rolování používá například v menu, pro vybírání položek. Vertikální rolování spočívá ve snižování a zvyšování výšky ruky, která se nachází ve vodorovné pozici, směrem od těla.



Obr. 10. Gesto vertikální rolování
u senzoru Kinect [4, s. 218]

1.3.1.8 Swipe

Swipe je čisté gesto jako mávání, laicky řečeno se jedná o „mávnutí rukou“. Detekování gesta swipe vyžaduje konstantní sledování pohybu ruky sledované osoby a udržování její předchozí pozice. [4, s. 210] Toto gesto se používá pro přechod mezi snímky. Rozlišuje se swipe doleva a swipe doprava. Typický příklad využití je přepínání snímků prezentace, kde gesto swipe doprava posune prezentaci na další snímek a gesto swipe doleva na snímek předešlý.

1.3.2 Hlas

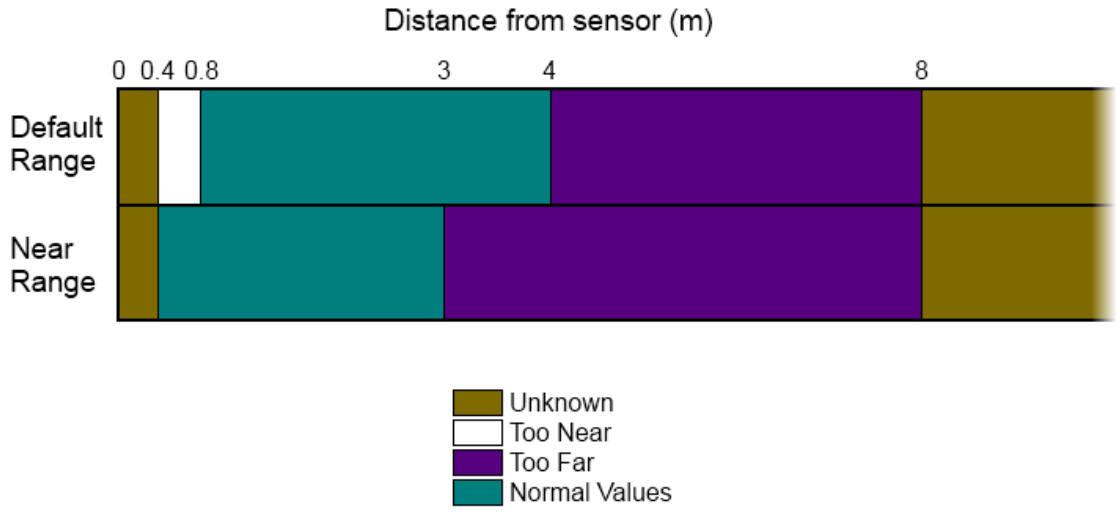
Hlasové příkazy patří mezi další způsob ovládání pomocí senzoru Kinect. Aby bylo možné hlasový příkaz rozpoznat, senzor Kinect využívá základní hlasové rozhraní, které existuje od operačního systému Windows Vista. Toto rozhraní zahrnuje knihovny DirectX Media Object (dále jen „DMO“) a Speech Recognization API (dále jen „SAPI“). Knihovna DMO poskytuje rozhraní pro práci s mikrofony senzoru Kinect a poskytuje funkcionality jako akustické rušení ozvěny, automatické zesílení nebo potlačení šumu, díky kterým zajišťuje lepší kvalitu zvuku. SAPI je jednoduchá vývojová knihovna, která mimo jiné

obsahuje sadu jazykových modelů a také umožňuje vyvíjet vlastní hlasové příkazy. [4, s. 224]

Senzor Kinect dále disponuje vlastním procesem pro zpracování zvuku. Pro rozpoznávání lidského hlasu senzor Kinect aplikuje frekvenční filtr a tím potlačí frekvence, které jsou mimo lidský hlas (80-1100 Hz). [5, s. 218]Tímto vyfiltruje pouze lidský hlas. Samotné rozpoznání spočívá ve dvou fázích. Nejprve je nutné vytvořit tzv. gramatiku, která obsahuje seznam všech slov, které budou použity jako spouštěče. Slova nebo spouštěče by měly být jednoduchá a doporučuje se maximální délka fráze pět slov.

1.4 Rozdíl mezi senzorem Kinect pro Windows a pro Xbox 360

Hlavní účel senzoru Kinect pro Xbox 360 je vylepšení herního zážitku hráčů. Zatímco senzor Kinect pro Windows je primárně určen pro vývojáře a tedy ne pro herní účely. Obě zařízení jsou si podobná v mnoha ohledech. Existuje několik nepatrných rozdílu mezi těmito zařízeními z pohledu vývojáře. Kinect pro Xbox 360 byl vytvořen ke sledování hráčů, kteří jsou nejdále 4 metry od senzoru. Pokud se sledovaný objekt vyskytne ve vzdálenosti menší než 80 centimetrů, senzoru se sledovaný objekt nepodaří zachytit. Tento nedostatek je řešen u senzoru Kinect pro Windows díky novému firmwaru, který povoluje tzv. „blízký režim“ sledování. Použití blízkého režimu pro sledování vylepší sledování objektů až do blízkosti 40 centimetrů od senzoru bez ztráty přesnosti. [5, s. 15 - 16]



Obr. 11. Výchozí a blízký režim senzoru Kinect [7]

2 KINECT SDK A ZPRACOVÁNÍ OBRAZU

Senzor Kinect má vlastní sadu vývojových nástrojů pro usnadnění práce – Kinect for Windows SDK (dál jen „Kinect SDK“). Kinect SDK bylo vydáno 7 měsíců poté, co bylo zařízení senzor Kinect vypuštěno do světa. Tato sada vývojových nástrojů má usnadnit práci programátorům pro platformu Windows. [11, s. 151]

Kinect SDK poskytuje třídy a metody pro zpracování obrazu. Mezi nejdůležitější třídy, které zpracovávají data ze senzorů a tyto data reprezentují v nám přijatelnější formě jsou - ColorImageStream, DepthImageStream a SkeletonImageStream. Práce s datovými proudy je proces o třech krocích. Datový proud musí být nejprve povolen, jakmile je povolen, aplikace vytáhne obrazové data z datového proudu a následně tyto data zpracuje. Poslední dva kroky se opakují stále dokola tak dlouho, dokud jsou obrazová data dostupná. [4, s. 28]

2.1 Zpracování barevného obrazu

Třída *ColorImageStream* získává data z barevné kamery senzoru Kinect. Barevná kamera senzoru Kinect vytváří barevné video jako kterákoliv jiná videokamera nebo webkamera. Tento datový proud je ze všech nejméně složitý. Barevná data jsou v Kinect SDK dostupná ve formátech RGB nebo YUV. Ve výchozím nastavení je obraz ve formátu RGB s rozlišením 640 x 480 obrazových bodů. [4, s. 37] Každý obrazový formát je počítán ze stejných dat získaných barevnou kamerou, proto oba formáty reprezentují stejný obraz.

2.1.1 RGB

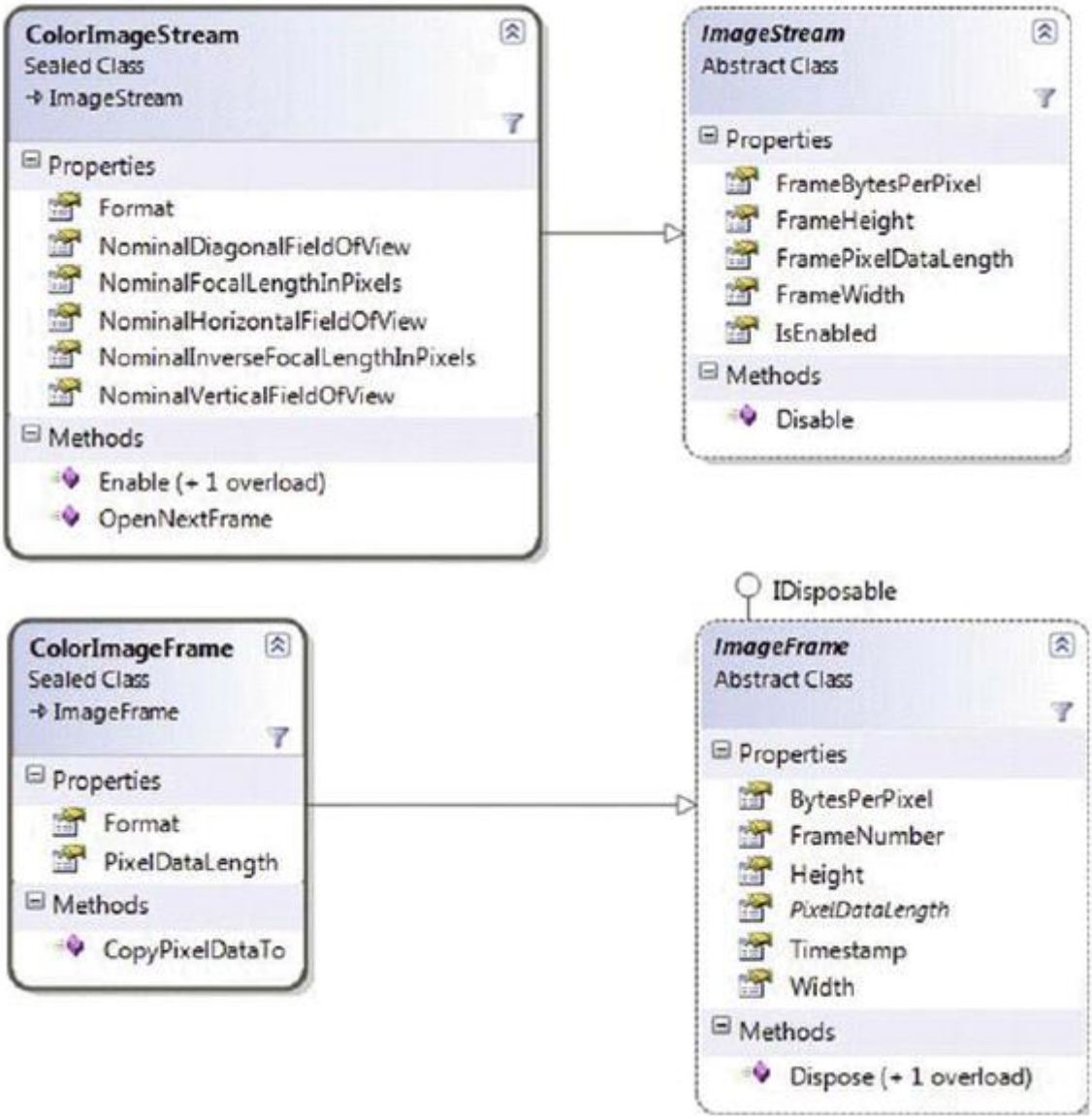
Barevný model RGB má tři barevné složky – červenou (Red), zelenou (Green) a modrou (Blue). Každá složka nese hodnotu 0-255. RGB je aditivní model. To znamená, že jednotlivé složky barev se sčítají a vytvářejí tak světlo větší intenzity. Při plné intenzitě, kdy všechny tři barevné složky mají hodnoty 255, vznikne bílá barva a naopak při nejnižší intenzitě vznikne černá. Tento barevný model je dnes nejpoužívanější a najdete jej v téměř každé elektronické obrazovce.

2.1.2 YUV

YUV vychází z barevného modelu RGB. Y v názvu barevného modelu značí jasovou složku, U a V nesou barevnou informaci a jsou vyhodnoceny jako rozdíl B - Y a R

- Y, tedy rozdíl jasové složky od modré a červené složky modelu RGB. YUV je analogový signál jehož digitální podoba se označuje jako YCbCr. [8]

2.1.3 Objektový model třídy ColorImageStream



Obr. 12. Objektový model třídy ColorImageStream [4, s. 37]

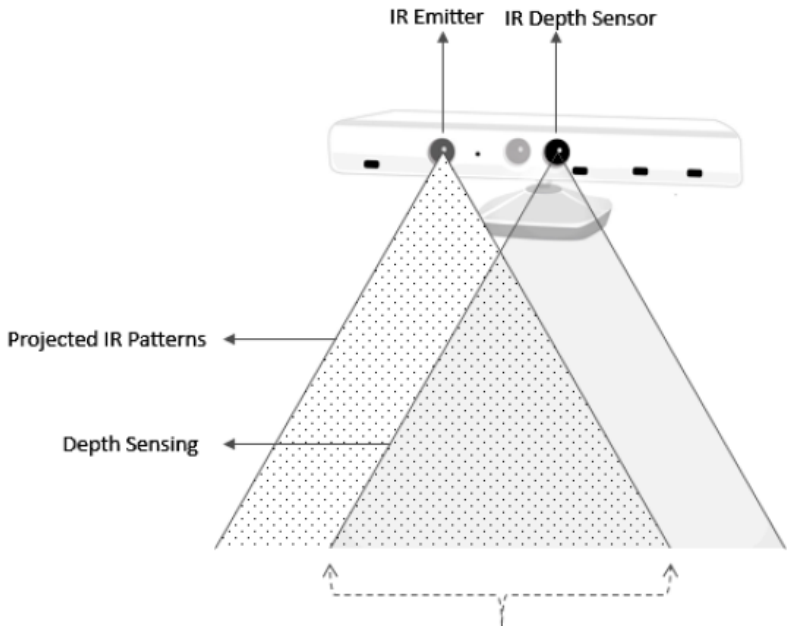
Na obrázku 12 můžete vidět tzv. diagram třídy *ColorImageStream*. Diagram třídy je pomůcka pro vývojáře softwaru při objektovém modelování. Obecně každá třída má určité vlastnosti a metody. Aby bylo možné získat barevná data z datového proudu *ColorImageStream*, je zapotřebí nejprve aktivovat datový proud pomocí metody *Enable()*.

2.2 Zpracování hloubkového obrazu

Senzor Kinect se odlišuje od ostatních zařízení tím, že poskytuje třetí dimenzi. K tomu využívá infračerveného vysílače a infračervené kamery. Kinect SDK neposkytuje přímý přístup k datovému proudu z infračervené kamery, ale tyto data zpracuje a vytvoří z nich hloubkový obraz. Data hloubkového obrazu pochází z třídy *DepthImageFrame*, kterou vytváří *DepthImageStream*.

2.2.1 Měření hloubky

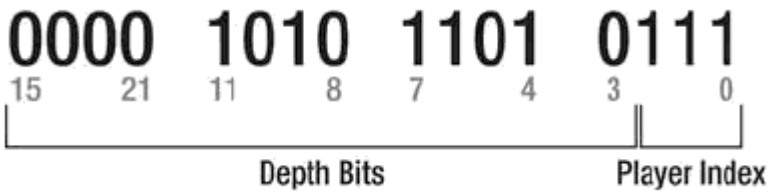
Infračervený vysílač může z venku vypadat jako běžná kamera, ale není tomu tak. Tento vysílač neustále emituje infračervené světlo v pseudonáhodném bodovém vzoru přes všechno, co stojí před vysílačem. Tyto body jsou pro člověka neviditelné, ale je možné zachytit jejich hloubku pomocí infračervené hloubkové kamery. Bodová světla se odráží od různých objektů a infračervená hloubková kamera je od objektu zachytí a převede je do hloubkových informací spočítáním vzdálenosti mezi kamerou a objektem, kde bylo infračervené světlo bodu zachyceno. [5, s. 11]



Obr. 13. Pseudonáhodný bodový vzor [5, s. 11]

2.2.2 Hloubkové informace

Zorné pole infračervené hloubkové kamery má pyramidový tvar. Vzdálenější objekty od kamery mají větší rozsah stran než objekty blíže. To znamená, že šířka a výška rozlišení, například 640 x 480 obrazových bodů, neodpovídá fyzickému umístění v zorném poli kamery. Nicméně hodnota hloubky pro každý obrazový bod se mapuje. Každý obrazový bod je reprezentován 16 bity, jeho hodnota hloubky zabírá jen 13 bitů. [4, s. 52]



Obr. 14. Uspořádání hloubkových bitů [4, s. 52]

V bitu 3 až 15 je uložená hodnota hloubky. Zjištění hloubky vyžaduje bitovou manipulaci. K jejímu získání je zapotřebí posunout bity doprava a odstranit první 3 bity s informacemi o sledovaném objektu označované jako „Player index“. Po odstranění prvních tří bitů se dostaneme k informaci o hloubce sledovaného objektu.

Pokud je hodnota hloubky nulová, obsahuje nuly, znamená to, že senzor Kinect nebyl schopen hloubku zachytit. [4, s. 55] To může být způsobeno tím, že sledovaný objekt je příliš blízko nebo naopak daleko od senzoru Kinect, tzn., že se dostal mimo zorné pole senzoru.

2.2.3 Indexování osob

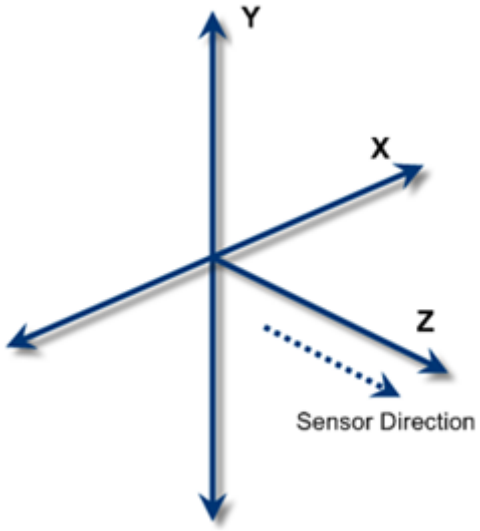
Senzor Kinect detekuje až 6 osob ve svém zorném poli. Aby bylo možné od sebe rozeznat osoby v zorném poli, Kinect SDK každé z osob přiřadí číslo. Toto číslo je uloženo v prvních třech bitech (Obr. 14, Player index) spolu s hloubkovými informacemi daného obrazového bodu. Číslo hráče se tedy uchovává v bitech 0 až 2.

Obrazový bod s nulovou hodnotou *player index* znamená, že na daném obrazovém bodu se nenachází žádná osoba. Jinak jsou osoby číslované od 1 do 6. Nicméně, aktivováním *DepthStreamu se* neaktivuje i sledování osoby. Sledování osoby vyžaduje

aktivování *SkeletonStreamu*. Pouze s aktivním *SkeletonStreamem* se hodnota *player index* zobrazí v hloubkové informaci obrazového bodu. [4, s. 67]

2.3 Zpracování obrazu kostry

Zpracováním hloubkového obrazu z infračerveného vysílače a hloubkové kamery dochází ke zjištění pozic jednotlivých kloubů lidské kostry. Každý kloub je reprezentován v trojrozměrné rovině, souřadnicemi X, Y a Z. Souřadnice X a Y určují umístění kloubu v rovině a vzdálenost od senzoru Kinect je obsažena v souřadnici Z.



Obr. 15. Pravotočivý souřadný systém kloubů kostry [7]

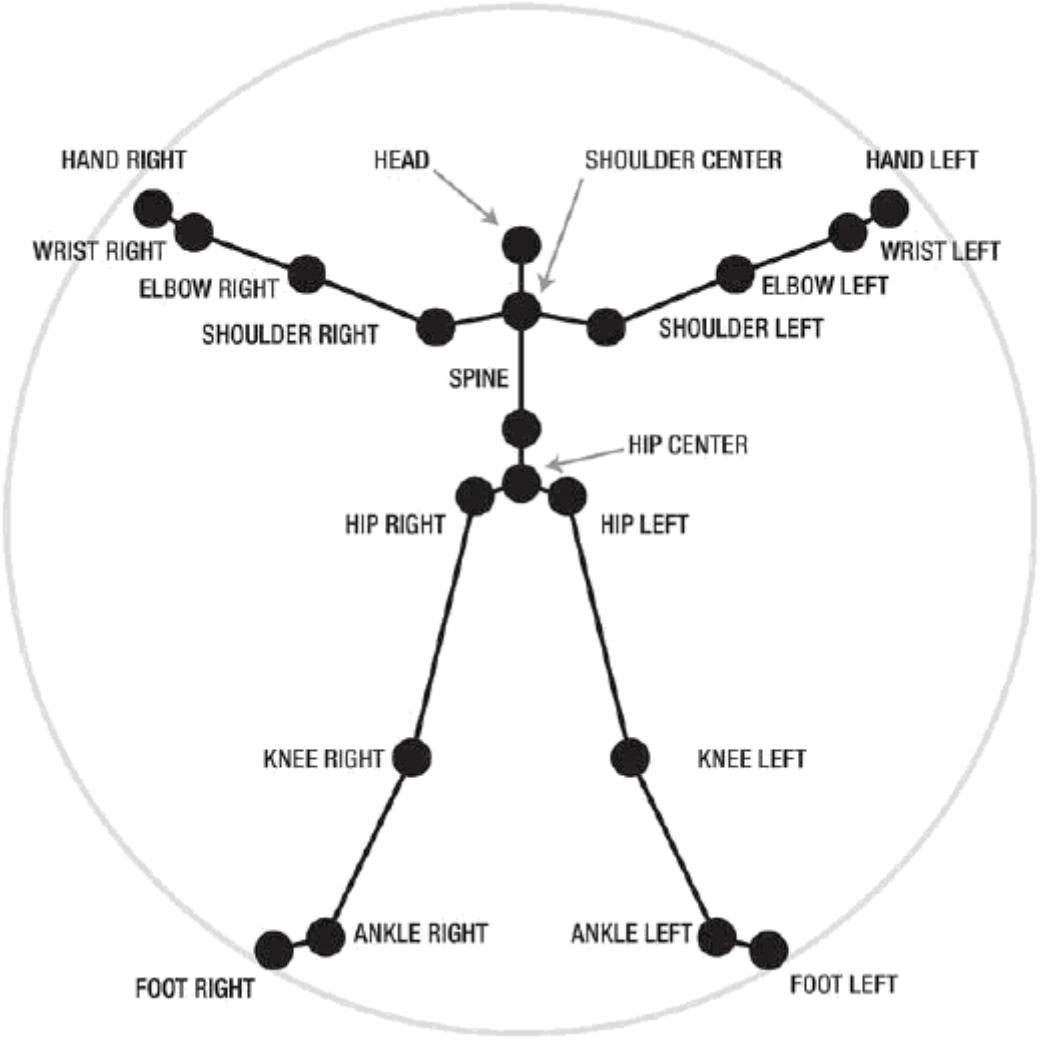
Ve výchozím nastavení se zobrazení kostry na sledované osobě zrcadlí. To znamená, že osoba stojící čelem k senzoru Kinect je považována za dívající se ve směru -Z v pravotočivém souřadném systému. [7]

2.3.1 Klouby

Sledováním kostry člověka se získají tyto body nebo klouby:

- hlava,
- levé a pravé zápěstí,
- levý a pravý loket,

- levé a pravé rameno,
- střed mezi rameny,
- kyčelní kloub,
- levý a pravý bok,
- levé a pravé koleno,
- levý a pravý kotník a
- levé a pravé chodilo.



Obr. 16. Obraz kostry člověka a zobrazením kloubů [4, s. 99]

2.3.2 Stavy sledování kostry

Kinect SDK poskytuje třídu *Skeleton*, která definuje kostru člověka, popisuje pozici kostry samotné a pozici všech dvaceti kloubů. [4, s. 96] Třída Skeleton dále obsahuje vlastnost *SkeletonTrackingState*, která definuje tyto tři stavy:

- NotTracked
  - objekt Skeleton nereprezentuje sledovanou osobu, pozice kostry a všech kloubů obsahují nulové hodnoty
- PositionOnly
  - kostra je detekována, ale není aktivně sledována, pozice kostry obsahuje nenulové hodnoty, pozice kloubů nulové
- Tracked
  - kostra je aktivně sledována, pozice kostry i všech kloubů obsahují nenulové hodnoty

2.3.3 Viditelné části kostry

Některé typy aplikací mohou vyžadovat, aby sledovaná osoba stála v zorném poli senzoru Kinect v určité pozici. Tak aby byly viditelné jen určité části lidské kostry nebo kostra celá. Součástí třídy *Skeleton* u Kinect SDK je pole *ClippedEdges*. Toto pole popisuje, které části kostry jsou mimo zorné pole senzoru Kinect. [4, s. 98] Obsahuje tyto hodnoty:

- bottom
  - sledovaná osoba má jednu nebo více částí kostry nacházející se pod zorným polem senzoru Kinect
- left
  - sledovaná osoba má jednu nebo více částí kostry nacházející se vlevo od zorného pole senzoru Kinect
- right

  - sledovaná osoba má jednu nebo více částí kostry nacházející se vpravo od zorného pole senzoru Kinect

- top
  - sledovaná osoba má jednu nebo více částí kostry nacházející se nad zorným polem senzoru Kinect
- none
  - sledovaná osoba je kompletně v zorném poli senzoru Kinect

2.4 Jednotky měření

Senzor Kinect měří hloubku v milimetrech. Při zpracování hloubkových dat jsou hodnoty v milimetrech. Oproti tomu jednotky vektoru pozice kloubu u kloubového obrazu jsou v metrech. [4, s. 291]

II. PRAKTICKÁ ČÁST

3 NÁVRH METODIKY MĚŘENÍ LIDSKÉ POSTAVY

3.1 Rekapitulace dostupných prostředků pro měření postavy

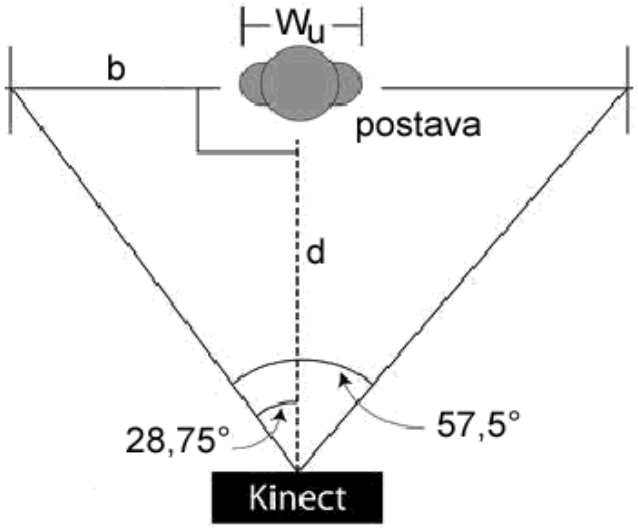
Pro přehled zrekapituluji všechny prostředky, které mám, abych mohl změřit lidskou postavu.

Hloubkový obraz, ze kterého získávám hloubková data a z těchto dat jsem schopen zjistit výšku a šířku sledované osoby. Tato šířka není šířkou ramen nebo šířkou pasu. Jedná se o prostor, který lidská postava vyplňuje v zorném poli senzoru Kinect. To znamená, že pokud budete mít ruce podél těla, šířka bude udávat šířku vašich ramen. Když rozpažíte ruce, šířka bude udávat šířku vašich rozpažených rukou, od konce levé ruky po konec pravé ruky. Pro potřeby měření je tedy nutné, aby sledovaná osoba měla ruce podél těla.

Hlavní prostředek, který budu využívat, je obraz kostry. Kinect SDK generuje celkem 20 kloubů. To mohu využít při zjišťování velikosti oblečení. Vím-li, že pro velikost trička potřebuji znát šířku ramen, jedná se o vzdálenost mezi dvěma klouby, konkrétně mezi levým a pravým ramenním kloubem.

3.2 Určení šířky a výšky postavy z hloubkového obrazu

Známe-li zorné pole senzoru Kinect (viz. kapitola 1.2.1 teoretické části), můžeme dopočítat šířku a výšku sledované osoby (dále jen osoby). K provedení výpočtu potřebujeme znát vzdálenost osoby od senzoru Kinect, dále počet obrazových bodů, které osoba vyplňuje a v poslední řadě rozměr obrazu. Na obr. 17 je vidět grafické znázornění pro výpočet.



Obr. 17. Grafické znázornění výpočtu šířky postavy [4, s. 70]

Odvození vzorce [4, s. 70] pro výpočet šířky postavy:

$$b = d \tan(28{,}75) \tag{1}$$

$$\frac{W_U}{640} = \frac{W_R}{2b} \tag{2}$$

kde d je vzdálenost osoby od senzoru Kinect, b je zbývající vzdálenosti od osoby ke konci rámce obrazu, W_U je šířka osoby v obrazových bodech a W_R je reálná hodnota šířky osoby.

Výsledný vzorec [4, s. 70]:

$$W_R = \frac{2b(W_U)}{640} \tag{3}$$

Obdobným způsobem můžeme odvodit vzorec pro výpočet výšky postavy s tím rozdílem, že změníme pozorovací úhel (místo 57,5° bude 43,5°) a rozměr obrazu:

$$b = d \tan(21{,}75) \tag{4}$$

$$\frac{H_U}{480} = \frac{H_R}{2b} \tag{5}$$

kde H_U zde reprezentuje výšku osoby v obrazových bodech a H_R reálnou hodnotu této výšky.

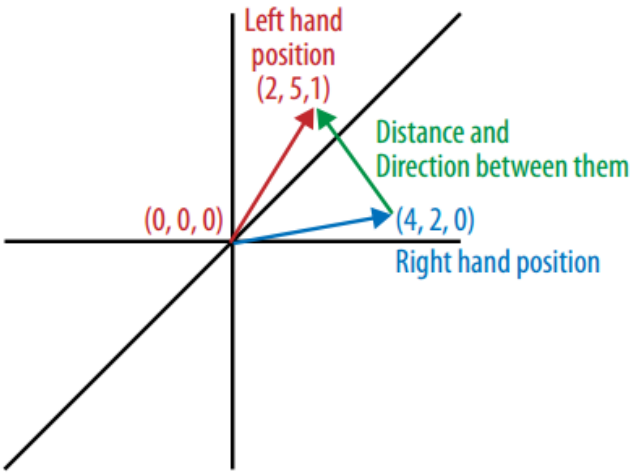
Výsledný vzorec:

$$H_R = \frac{2b(H_U)}{480} \tag{6}$$

Výše uvedené vzorce platí, uvažujeme-li maximální rozlišení hloubkového obrazu 640 x 480 obrazových bodů. Odtud pochází hodnoty 640 a 480 z posledních kroků výpočtů.

3.3 Využití obrazu kostry pro výpočet vzdálenosti kloubů

Je-li každý kloub lidské kostry prostřednictvím Kinect SDK reprezentován v trojrozměrné rovině, můžeme toho využít a vypočítat vzdálenost mezi dvěma klouby.



Obr. 18. Vzdálenost mezi dvěma klouby [3, s. 223]

Využitím Pythagorova teorému získáme rovnici pro výpočet vzdálenosti *d* dvou kloubů v trojrozměrné rovině:

$$d = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2 + (Z_1 - Z_2)^2} \tag{7}$$

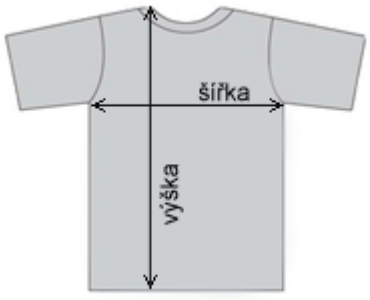
Takto získáme požadovanou vzdálenost dvou kloubů, ve které je započítána i souřadnice Z, tedy vzdálenost kloubů od senzoru. Pro potřeby ukázkové aplikace však souřadnici Z nepotřebujeme. Cílem je zjistit vzdálenost mezi dvěma klouby v dvourozměrné rovině. Vyhneme se tak problému, který by mohl nastat, jestliže by

měřená osoba nestála kolmo k senzoru Kinect. Pak by se lišily souřadnice Z a byly by zahrnuty ve výpočtu. Výsledná vzdálenost by se mohla rozcházet s reálnou hodnotou. Rovnici (7) tedy upravíme takto:

$$d = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \tag{8}$$

Tato metoda je vhodná pro výpočet délky těchto částí oblečení:

- tričko
    - šířka – vzdálenost mezi levým a pravým ramenním kloubem
    - výška - vzdálenost mezi centrálním ramenním kloubem a centrálním kyčelním kloubem



Obr. 19. Parametry pro výpočet velikosti trička [9]

- mikina nebo svetr – délka rukávu jako vzdálenost mezi ramenním kloubem a kloubem na zápěstí, délka mikiny nebo svetru jako vzdálenost mezi centrálním ramenním kloubem a centrálním kyčelním kloubem
- kalhoty – vzdálenost mezi kyčelním kloubem a kloubem u kotníku

4 REALIZACE UKÁZKOVÉ APLIKACE

Pro demonstraci možností a využití senzoru Kinect pro měření lidské postavy byla vytvořena ukázková aplikace. Tato aplikace slouží pouze jako demonstrace využití možností senzoru Kinect a není určena pro praktické využití. Jsem zvyklý psát programy v anglickém jazyce, a proto veškeré názvy proměnných, metod, tříd a také poznámky ke zdrojovému kódu jsou psané v anglickém jazyce. Pro lepší orientaci ve zdrojovém kódu je převážná většina kódu rozčleněna do bloků, tzv. regionů, které jsou logicky pojmenovány.

4.1 Použité prostředky při tvorbě

Pro naprogramování ukázkové aplikace jsem zvolil objektově orientovaný programovací jazyk C#, framework .NET a vývojové prostředí Microsoft Visual Studio 2012. Ukázková aplikace používá Kinect for Windows SDK ve verzi 1.6, které poskytuje prostředky pro komunikaci senzoru Kinect a počítače.

4.1.1 Kinect pro Windows SDK 1.7

V době psaní diplomové práce byla vydána nová verze SDK. Konkrétně se jedná o verzi 1.7, která přináší mimo jiné nové prvky interakce se senzorem Kinect, ale hlavně přichází s novou technologií, nazvanou „Kinect Fusion“.

Technologie Kinect Fusion zachytí a zrekonstruuje velmi podrobný 3D model člověka nebo objektu v reálném čase. Tento 3D model se vytvoří ze sekvencí snímků hloubkových dat získaných senzorem Kinect. [10]

Obr. 20. Kinect Fusion[9]

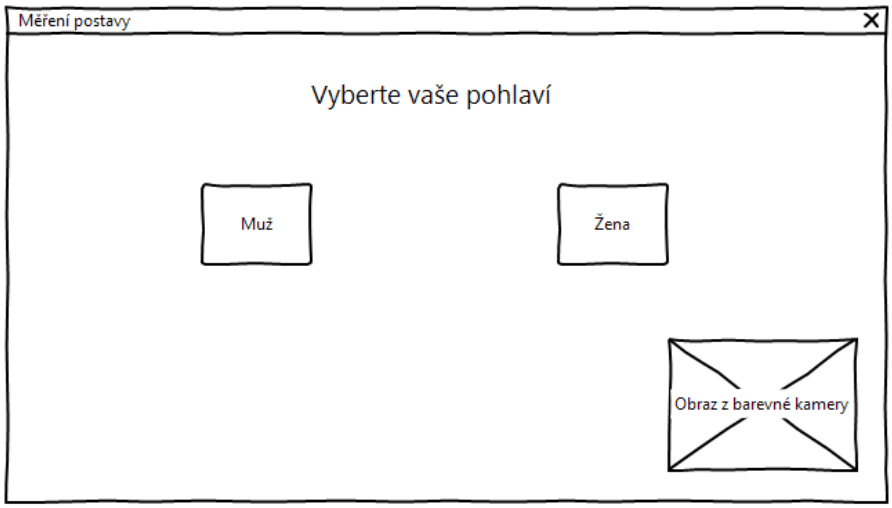
4.2 Systémové a hardwarové požadavky

Podporované operační systémy jsou Windows 7 a Windows 8. Pro správnou funkčnost ukázkové aplikace je nutné mít počítač minimálně s touto konfiguraci:

- 32bitový nebo 64bitový procesor,
- dvou-jádrový procesor s frekvencí 2,66 GHz nebo vyšší,
- port USB 2.0,
- 2 GB operační paměti a samozřejmě
- senzor Microsoft Kinect

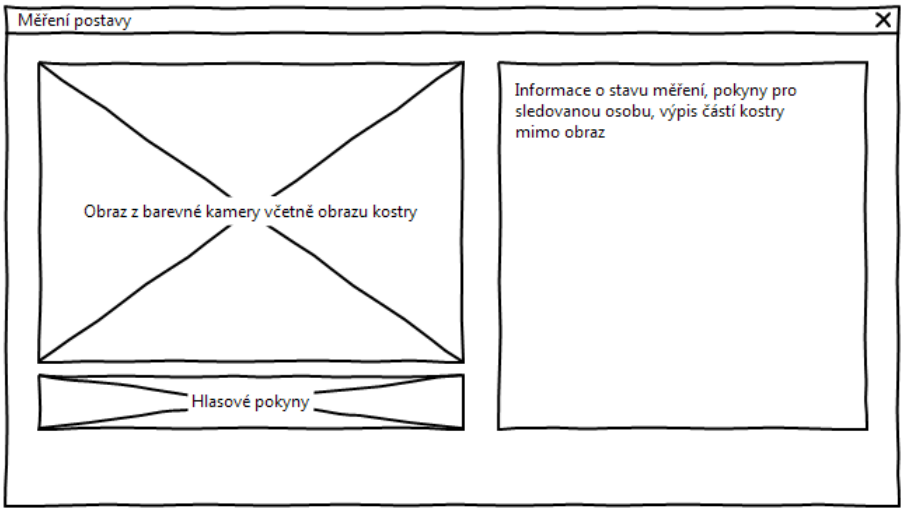
4.3 Návrh uživatelského rozhraní a funkčnost

Před samotnou tvorbou ukázkové aplikace je nutné se zamyslet, jak bude vypadat uživatelské rozhraní. Pro tyto účely jsem vytvořil tzv. wireframe obrazovky, na kterých je vidět grafické zobrazení budoucího uživatelského rozhraní. Tvorba wireframů zvyšuje efektivitu práce, když se nemusíte zabývat uživatelským rozhraním v průběhu tvorby aplikace. Ukázková aplikace bude rozdělena na 3 části: úvodní obrazovka, hlavní obrazovka a obrazovka s výsledky měření. Jak můžeme vidět na obr. 21, úvodní obrazovka je jednoduchá a budou se na ní nacházet 2 tlačítka pro volbu pohlaví uživatele aplikace (dále jen uživatel) a obraz z barevné kamery, aby se uživatel mohl vidět.



Obr. 21. Wireframe úvodní obrazovky

Na hlavní obrazovce se bude odehrávat nejdůležitější část celé ukázkové aplikace. Bude rozdělena na 2 poloviny – levou a pravou. Na levé polovině se bude nacházet ve vrchní části výstup z barevné kamery spolu s vykresleným obrazem kostry uživatele a ve spodní části se budou zobrazovat použité hlasové pokyny. V pravé polovině hlavní obrazovky se budou zobrazovat informace pro uživatele, které bude muset uživatel vykonat, aby mohlo začít měření jeho postavy.



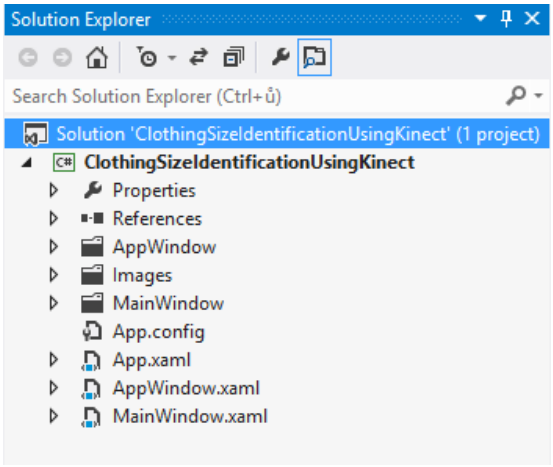
Obr. 22. Wireframe hlavní obrazovky

Jako poslední bude obrazovka s výsledky měření, která se automaticky zobrazí, jakmile skončí měření uživatele. Bude se zde nacházet informace s charakteristikou a popisem postavy uživatele a dále doporučené velikosti oblečení.

Obr. 23. Wireframe obrazovky s výsledky měření

4.4 Programová část aplikace

Na obr. 24 je vidět struktura aplikace v Solution Exploreru Visual Studia, která je logicky rozčleněna. Celá aplikace se dělí na dvě části – detekování senzoru Kinect a vlastní aplikace. Veškeré třídy sloužící pro detekování senzoru Kinect se nachází ve složce *MainWindow*, patřící pod okno *MainWindow.xaml*. Obdobně vlastní aplikace se nachází ve složce *AppWindow* a tato složka patří oknu *AppWindow.xaml*. Pro uložení obrázků je určena složka *Images*. Obsah výše zmíněných složek bude rozebrán později.



Obr. 24. Struktura projektu

Pro umožnění práce se senzorem Kinect bylo zapotřebí přidat referenci na Kinect SDK (knihovna Microsoft.Kinect). Pokud se tak neučiní, nebude možno přistupovat k SDK a používat tak kamery senzoru Kinect. Aplikace používá hlasové ovládání, proto bylo zapotřebí přidat rovněž referenci na Speech SDK (knihovna Microsoft.Speech) pro umožnění používání hlasových příkazů. Poslední přidanou referencí je knihovna Coding4Fun.Kinect.wpf. Tato knihovna obsahuje užitečné třídy pro práci se senzorem Kinect, které neobsahuje Kinect SDK a které ukázková aplikace bude využívat.

Reference byly přidány, ale pro přístup ke Kinect SDK a Speech SDK je potřeba dále přidat direktivu ve všech třídách, ve kterých se bude pracovat s těmito SDK. Na následujícím kódu lze vidět přidání direktiv:

```
1  using Microsoft.Kinect;
2  using Microsoft.Speech;
```

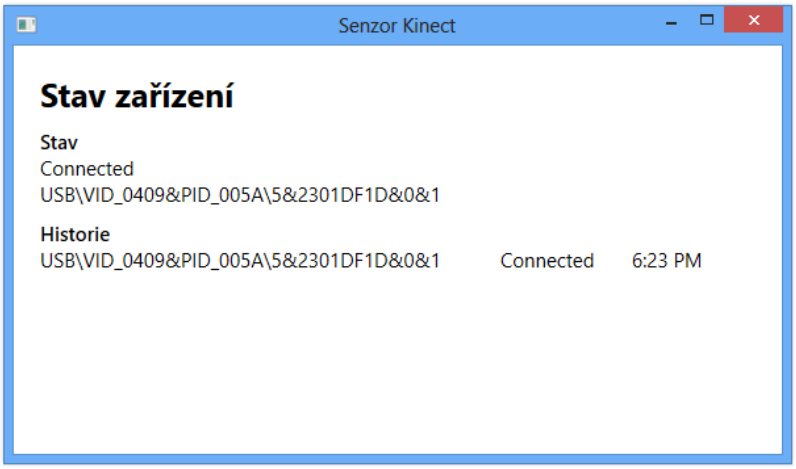
4.4.1 Detekce senzoru Kinect

Pro vytvoření uživatelského rozhraní detekování senzoru Kinect byl použit deklarativní značkovací jazyk XAML(eXtensible Aplication Markup Language), který vychází z jazyku XML. U jazyku XAML se vytváří párový tag objektu <window>obsah</window>, který definuje vlastnosti okna uživatelského rozhraní a jeho obsah. Zdrojový kód:

```
1  <Window x:Class="ClothingSizeIdentificationUsingKinect.MainWindow"
2    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4    xmlns:l="clr-namespace:ClothingSizeIdentificationUsingKinect"
5    Title="Senzor Kinect" Height="350" Width="600"
6    Loaded="WindowLoaded" Closed="WindowClosed">
     …
7  </Window>
```

Na řádku 5 jde vidět, že okno má rozměr 600 x 350 obrazových bodů a je pojmenováno „Senzor Kinect“. Dále na řádku 6 jsou přiřazeny atributy *Loaded* a *Closed*. Atribut *Loaded* zavolá metodu *WindowLoaded* jakmile se spustí aplikace. Obdobně, ale v opačném případě, tedy při ukončení aplikace se zavolá metoda *WindowClosed*.

Bez zapojeného senzoru Kinect by ukázková aplikace nemohla správně fungovat. Proto se jako první věc po spuštění kontroluje, zda je senzor Kinect zapojen a připraven k použití. Pokud tomu tak není, zobrazí se chybová hláška. V takovém případě se nespustí hlavní okno aplikace, ve které následně probíhá měření a vyčkává se, dokud není zapojen senzor Kinect. Pokud je senzor Kinect zapojen, chybová hláška zmizí a objeví se informace o stavu senzoru Kinect a také jeho unikátní identifikační číslo. Jakmile je senzor Kinect inicializován a připraven k použití, nachází se ve stavu připojen, informace se uloží do krátkodobé historie spolu s dobou jeho připojení, jak lze vidět na obr. 25. Informace o stavu zařízení je zobrazena v anglickém jazyce. Je tomu tak proto, že tato informace je čerpána přímo z Kinect SDK, které nepodporuje češtinu.



Obr. 25. Senzor Kinect byl detekován

4.4.2 Použití gest

Zdrojový kód:

```
1  <Window x:Class="ClothingSizeIdentificationUsingKinect.AppWindow"
2   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4   xmlns:Controls="clr-5namespace:Coding4Fun.Kinect.Wpf.Controls;
5   assembly=Coding4Fun.Kinect.Wpf"
```

Na řádku 4 a 5 ukázky zdrojového kódu lze vidět přidání reference na knihovnu Coding4Fun.Kinect.wpf, protože ukázková aplikace využívá gesto tlačítko Hover a tato knihovna obsahuje třídy pro práci s tímto gestem. Pro připomenutí funkčnosti gesta tlačítka Hover, jakmile se kurzor nachází nad tlačítkem, aplikace čeká určitou dobu a po uplynutí této doby aktivuje tlačítko.

```
6  <Controls:HoverButton Margin="0" Padding="0" x:Name="kinectButton" ImageSize="50"
7   ImageSource="/ClothingSizeIdentificationUsingKinect;
8   component/Images/RightHand.png"
9   ActiveImageSource="/ClothingSizeIdentificationUsingKinect;
10  component/Images/RightHand.png"
11  TimeInterval="2000“ Panel.ZIndex="1000" Canvas.Left="0" Canvas.Top="0"
12  Click="kinectButton_click" Visibility="Collapsed"/>
```

Na řádku 11 zdrojového kódu lze vidět hodnotu atributu *TimeInterval* nastavenou na číslo 2000. Toto číslo udává dobu v milisekundách, po kterou bude tlačítko čekat, než se stane aktivní. Tedy 2 sekundy. Hover tlačítko je tedy připraveno. Na řádku 12 vidíme atribut *Visibility*, který je nastaven na *Collapsed*. To znamená, že tlačítko Hover bude po zapnutí aplikace schované a zobrazí se až tehdy, detekuje-li senzor Kinect postavu.

Pro zobrazení pozice ruky uživatele na obrazovce je potřeba tuto pozici vykreslit. K tomu je určen objekt <canvas></canvas> jazyka XAML, který slouží jako plátno a umožňuje explicitní pozicování prvků. [14, s. 381] Následuje ukázka kódu pro vykreslení pozice uživatelovi ruky:

```
13  if (hand.TrackingState == JointTrackingState.NotTracked)
14  {
15   kinectButton.Visibility = Visibility.Collapsed;
16  }
17  else
18  {
19  kinectButton.Visibility = Visibility.Visible;
20  DepthImagePoint point = this.sensor.CoordinateMapper.MapSkeletonPointToDepthPoint
21  (hand.Position, DepthImageFormat.Resolution640x480Fps30);
22  handX = (int)((point.X *
23  LayoutRoot.ActualWidth/this.sensor.DepthStream.FrameWidth) –
24  (kinectButton.ActualWidth / 2.0));
25  handY = (int)((point.Y *
26  LayoutRoot.ActualHeight/this.sensor.DepthStream.FrameHeight) –
27  (kinectButton.ActualHeight / 2.0));
28  Canvas.SetLeft(kinectButton, handX);
29  Canvas.SetTop(kinectButton, handY);
30  }
```

Na řádku 13 ukázky zdrojového kódu lze vidět začátek klauzule *if/else* a podmínku, zda je pozice kloubu ruky sledována. Pokud není, tlačítko Hover zůstane schované. V opačném případě se tlačítko Hover zobrazí na obrazovce na pozici ruky uživatele. Pozice ruky uživatele je zjištěna výpočtem, který lze vidět na řádku 21 až 27. Jak pro souřadnici X, tak pro souřadnici Y. Po vypočtení souřadnic se nastaví pozice tlačítka Hover na obrazovce a to určením vzdálenosti od levého a vrchního okraje plátna. Lze vidět na řádku 28 a 29.

4.4.3 Použití hlasových příkazů

Hlasové příkazy fungují obdobně jako stisknutí klasického tlačítka. Liší pouze v tom, že nemusíte na tlačítko kliknout kurzorem myši, ale stačí říct nadefinovaný hlasový příkaz a událost, která má za úkol obsluhu stisknutí tlačítka se zavolá. Samozřejmě hlasové příkazy nemusí být spojeny pouze s událostmi tlačítek. Mohou být použity v kterékoliv části programu. Následují ukázky zdrojového kódu:

```
1  private SpeechRecognitionEngine speechRecognizer;
```

Na řádku 1 lze vidět inicializaci proměnné pro engine hlasového rozpoznávání. Proměnná je jednoduše pojmenována *speechRecognizer*.

```
2  this.speechRecognizer = this.CreateSpeechRecognizer();
```

Na řádku 2 se přistupuje k této proměnné. Volá se metoda s názvem *CreateSpeechRecognizer(),* ve které jsou nadefinované hlasové příkazy a která vrací objekt typu *SpeechRecognizeEngine*.

```
3  var grammar = new Choices();
4  grammar.Add("man");
```

Na řádku 3 lze vidět vytvoření instance třídy Choices(), což je naše gramatika hlasových příkazů a na řádku 4 je ukázka, jak se takový hlasový příkaz definuje.

```
5  private DispatcherTimer readyTimer;
```

Hlasové příkazy nejsou přístupné hned po spuštění aplikace. Musí se čekat 4 sekundy, dokud nebude senzor Kinect připraven přijímat zvuky. Jako funkci čekání jsem použil časovač *DispatcherTimer()* objektu *Dispatcher*. Inicializace lze vidět na řádku 5.

```
6  if (this.speechRecognizer != null && sensor != null)
7  {
8   this.readyTimer = new DispatcherTimer();
9   this.readyTimer.Tick += this.ReadyTimerTick;
10  this.readyTimer.Interval = new TimeSpan(0, 0, 4);
11  this.readyTimer.Start();
12  Speaker.Content = "Inicializuji audio...";
13 }
```

Na řádku 6 až 13 lze vidět *if* klauzuli, která se vykoná, pokud se v proměnných *speechRecognizer* a *sensor* nenachází nulové hodnoty, tedy hlasové rozpoznání bylo nastaveno a senzor Kinect byl nalezen. Na řádku 8 se vytváří nová instance objektu *DispatcherTimer()*, následně je přiřazena obslužná metoda *ReadyTimerTick()* a nastaven časový interval na 4 sekundy. Jakmile jsou nastavené všechny potřebné parametry, časovač je spuštěn vlastní metodou *Start().* Princip je jednoduchý, po uplynutí doby 4 sekund se vykoná kód, který se nachází v metodě *ReadyTimerTick().* Tento časovač v aplikaci využívám vícekrát.

```
14  private void SreSpeechRecognized(object sender, SpeechRecognizedEventArgs e)
15  {
16    if (e.Result.Confidence < 0.7)
17    {
18       return;
19    }
20    switch (e.Result.Text.ToUpperInvariant())
21    {
22       case "MAN":
23        // kód pro vykonání
24        break;
       .
       .
       .
```

Při použití hlasových příkazů je důležitým parametrem jistota, s jakou má aplikace přijmout hlasový příkaz. Pokud se nastaví na příliš nízkou hodnotu, může docházet k jevu, že aplikace použije jiný hlasový příkaz, než který se po ní žádá. V opačném případě, při nastavení příliš vysoké hodnoty jistoty, aplikace příkaz nemusí přijmout. Jako vhodná hodnota pro parametr jistoty bylo zvoleno číslo *0.7*, jak lze vidět na řádku 16 ukázky zdrojového kódu, což odpovídá přibližně 70% jistoty. Na řádcích 16 až 19 lze vidět *if* klauzuli, která se vykoná, pokud je jistota hlasového příkazu menší než *0.7*. V takovém případě se vykonávání kódu v dané metodě ukončí. Jestliže je jistota hlasového příkazu větší než *0.7*, pokračuje v dalším vykonávání kódu v dané metodě. Pro přiřazení kódu pro vykonání pro jednotlivé hlasové příkazy slouží *switch* klauzule, jak lze vidět na řádcích 20 a výše.

4.4.4 Vykreslení barevného obrazu a obrazu kostry

Pro vykreslení barevného obrazu a obrazu kostry je potřeba objektu *KinectSensor*. Bez tohoto objektu by nebylo možné se dostat k datům senzoru Kinect. Následuje ukázka zdrojového kódu:

```
1  private KinectSensor sensor;
```

Na řádku 1 lze vidět vytvoření globální proměnné objektu *KinectSensor* a pojmenované jednoduše *sensor*. Tato proměnná se dále používá pro přístup k veškerým datům senzoru Kinect.

```
2  this.sensor.ColorStream.Enable(ColorImageFormat.RgbResolution640x480Fps30);
3  this.colorPixels = new byte[this.sensor.ColorStream.FramePixelDataLength];
4  this.colorBitmap = new WriteableBitmap(this.sensor.ColorStream.FrameWidth,
5  this.sensor.ColorStream.FrameHeight, 96.0, 96.0, PixelFormats.Bgr32, null);
6  this.ColorImage.Source = this.colorBitmap;
```

Pro přístup k datům barevného obrazu se musel nejprve aktivovat *ColorStream*. V teoretické části je tato třída pojmenována jako *ColorImageStream*. Jedná se ovšem o stejnou třídu, která je v použité verzi Kinect SDK 1.6 pojmenována takto. Jak lze vidět na řádku 2, pro aktivování třídy *ColorStream* slouží metoda *Enable()*. Jako parametr je specifikovaný formát obrazu. Ten byl zvolen RGB s maximálním rozlišením 640 x 480 obrazových bodů při 30 snímcích za sekundu. Na řádku 3 se inicializuje globální proměnná *colorPixels* pro zjištění počtu obrazových bodů. Tato proměnná bude použita později pro vykreslení barevného obrazu. Na řádcích 4 až 5 se vytváří bitmapa s parametry třídy *ColorStream*, jako jsou šířka obrazu, výška obraz, počet bodů na palec a formát obrazu. To vše slouží k inicializaci objektu *Image*, do kterého bude vykreslen barevný obraz. Lze vidět na řádku 6 ukázky zdrojového kódu.

```
7  this.sensor.ColorFrameReady += this.Sensor_ColorFrameReady;
```

Datový proud z třídy *ColorStream* je spuštěn, nyní je potřeba přidat obsluhu události pro každý barevný obraz, který dostaneme z výše zmiňované třídy. Na řádku 7 lze vidět přidání obslužné metody *Sensor_ColorFrameReady* k události *ColorFrameReady* třídy *KinectSensor*. Tato metoda se stará o vykreslení barevného obrazu z dat získaných ze třídy *ColorStream*.

```
8  this.sensor.SkeletonStream.Enable(parameters);
```

Obdobně jako u třídy *ColorStream* je zapotřebí aktivovat datový proud. Třída *SkeletonStream* obstarává data o pozici všech 20 kloubů. Opět se jedná o shodnou třídu, která je pojmenována v teoretické části jako *SkeletonImageStream*. Na řádku 8 lze vidět zavolání metody *Enable()* s parametrem. Tento parametr není povinný, ale je zde proto, aby zpomalil zobrazování pozic kloubů o určitou dobu. Je tomu tak z důvodu pohodlnějšího ovládání gest, kdy pohybujete rukou ze strany na stranu. Na obrazovce se zobrazuje pozice ruky, která kopíruje trajektorii pohybu uživatelovi ruky nikoliv v reálném čase, ale s menším zpožděním.

```
9   var parameters = new TransformSmoothParameters
10  {
11    Smoothing = 0.3f,
12    Correction = 0.0f,
13    Prediction = 0.0f,
14    JitterRadius = 1.0f,
15    MaxDeviationRadius = 0.5f
```

```
16  };
```

Na řádcích 9 až 16 lze vidět nastavení těchto parametrů. Parametry byly nastaveny experimentálním způsobem. Měnil jsem hodnoty, dokud jsem nebyl spokojen.

```
17  this.sensor.SkeletonFrameReady += this.Sensor_SkeletonFrameReady;
18  this.Skeletons=new Skeleton[this.sensor.SkeletonStream.FrameSkeletonArrayLength];
```

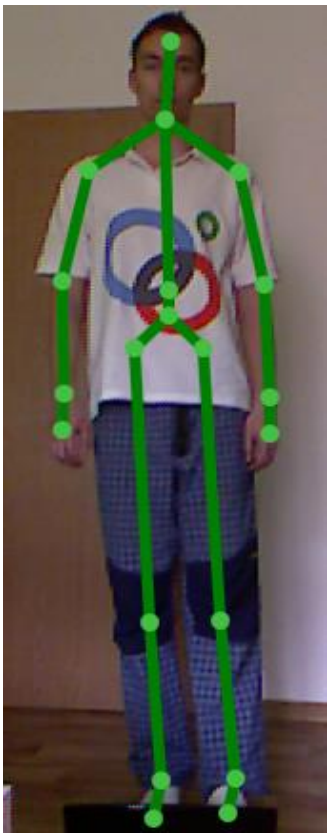
Na řádku 17 lze vidět přidání obsluhy události *SkeletonFrameReady* třídy *KinectSensor*. Obdobně jako je tomu u třídy *ColorStream*. Na řádku 18 se inicializuje globální proměnná *Skeletons*, do které jsou vloženy pozice kloubů všech detekovaných postav v zorném poli senzoru Kinect. K určení pouze jedné hlavní postavy ze všech detekovaných postav slouží metoda *GetPrimarySkeleton()*. Ta určí, která z detekovaných postav se bude sledovat. Vždy se jedná o detekovanou postavu, která je nejblíže senzoru Kinect. Jakmile jsou získána data o pozicích všech kloubů z třídy *SkeletonStream*, je potřeba vykreslit obraz kostry. Pro vykreslení byla použita třída *DrawingContext*. Následuje ukázka zdrojového kódu:

```
19  drawingContext.DrawEllipse(drawBrush,null,
20  this.SkeletonPointToScreen(joint.Position), JointThickness, JointThickness);
```

Na řádku 19 lze vidět použití metody *DrawEllipse()*. Ta slouží k vykreslení bodu na obrazovce. V případě aplikace se jedná o vykreslení kloubů z obrazu kostry. Tato metoda má 5 parametrů – barvu štětce, barvu pera, pozici bodu k vykreslení a poslední 2 parametry jsou poloměry X a Y daného bodu. Pero zde není potřeba, proto se na druhém místě parametrů nachází hodnota *null*. Na řádku 20 lze vidět použití metody *SkeletonPointToScreen()*, která převádí jednotlivé pozice kloubů na body v obrazovce.

```
21  drawingContext.DrawLine(drawPen,this.SkeletonPointToScreen(joint0.Position),
22  this.SkeletonPointToScreen(joint1.Position));
```

Aby byla vykreslená celá kostra a ne jen jednotlivé klouby, pomocí metody *DrawLine()* se spojují 2 klouby a vytváří se tak úsečky. Metoda *DrawLine()* má celkem 3 parametry - barvu pera, pozici prvního bodu (kloubu) a pozici druhého bodu (kloubu). Výsledek, jak vypadá vykreslený obraz kostry postavy lze vidět na obr. 26.



Obr. 26. Vykreslení obrazu kostry

4.4.5 Výpočet parametrů postavy z obrazu kostry

Ve složce *AppWindow* se nachází soubor *SkeletonData.cs*, který obsahuje stejnojmennou třídu a několik metod pracující s obrazem kostry. Mezi tyto metody patří - zjištění vhodné vzdálenosti uživatele od senzoru Kinect, postavení uživatele (čelem, bokem) a výpočty parametrů postavy jako jsou výška, šířka ramen, délka nohy apod. Následuje ukázka zdrojového kódu:

```
1   public double UserHeight()
2   {
3   var head = this.skeleton.Joints[JointType.Head];
4   var shoulderCenter = this.skeleton.Joints[JointType.ShoulderCenter];
5   var spine = this.skeleton.Joints[JointType.Spine];
6   var hipCenter = this.skeleton.Joints[JointType.HipCenter];
7   var hipLeft = this.skeleton.Joints[JointType.HipLeft];
8   var hipRight = this.skeleton.Joints[JointType.HipRight];
9   var kneeLeft = this.skeleton.Joints[JointType.KneeLeft];
10  var kneeRight = this.skeleton.Joints[JointType.KneeRight];
11  var ankleLeft = this.skeleton.Joints[JointType.AnkleLeft];
12  var ankleRight = this.skeleton.Joints[JointType.AnkleRight];
13  var footLeft = this.skeleton.Joints[JointType.FootLeft];
14  var footRight = this.skeleton.Joints[JointType.FootRight];
```

```
15   int legLeftTrackedJoints = CountOfTrackedJoints(hipLeft, kneeLeft, ankleLeft,
16   footLeft);
17   int legRightTrackedJoints = CountOfTrackedJoints(hipRight, kneeRight,
18   ankleRight, footRight);
19
20   double legLength = legLeftTrackedJoints > legRightTrackedJoints ?
21   JointsLength(hipLeft, kneeLeft, ankleLeft, footLeft) : JointsLength(hipRight,
22   kneeRight, ankleRight, footRight);
23
24   return (JointsLength(head, shoulderCenter, spine, hipCenter) + legLength +
25   headDivergence);
26   }
```

Na řádcích 1 až 26 lze vidět jednu z metod, která slouží pro měření. Zde se jedná konkrétně o měření výšky postavy. V konstruktoru třídy *SkeletonData* je předáván obraz kostry – všech 20 kloubů. Pro přístup k jednotlivým kloubům se zavolá kolekce *Joints* s typem kloubů v hranatých závorkách. Jedná se o řádky 3 až 14. Někdy může nastat situace, kdy jedna noha nemá sledované všechny klouby a výsledná výška by pak mohla být nepřesná. Proto je na řádku 15 až 22 pro přesnější měření dalším krokem zjištění, která noha je lépe sledována, protože při měření výšky postavy se zahrnují klouby od hlavy až po patu. Na řádku 24 lze vidět použití metody *JointsLength()*. Zde je ukázka zdrojového kódu této metody:

```
27  public double JointsLength(Joint joint0, Joint joint1)
28  {
29  double jointsLength = Math.Sqrt(
30  Math.Pow(joint0.Position.X - joint1.Position.X, 2) +
31  Math.Pow(joint0.Position.Y - joint1.Position.Y, 2));
32  return jointsLength * 100;
33  }
```

Metoda *JointsLength()* vychází z rovnice (8), kde dochází k výpočtu vzdálenosti mezi dvěma klouby. Jedná se o přepis rovnice do zdrojového kódu. Pokud je metoda volána s více jak dvěma parametry, zavolá se stejnojmenná metoda, která ve *for* cyklu prochází všechny klouby z parametrů a volá zpětně metodu *JointsLength()* z řádku 27 až 33.

Při pohledu na řádek 25 lze vidět konstantu nazvanou *headDivergence*. Tato konstanta je zde z důvodu kompenzace ztráty při měření. Například princip měření výšky uživatele spočívá ve zjištění délky od kloubu nohy po kloub hlavy. Jak lze vidět na obr. 27, mezi vrchní částí hlavy a vykresleným kloubem hlavy je volný prostor. A právě tento volný prostor je zahrnut v konstantě *headDivergence*. Podobně je řešena kompenzace ztrát i u

ostatních parametrů, které jsou měřeny. V důsledku toho je docíleno přesnější měření postavy uživatele.

Obr. 27. Detail vykreslení kostry hlavy

4.4.6 Výpočet šířky postavy z hloubkového obrazu

Protože aplikace využívá pro měření jak obraz kostry, tak hloubkový obraz, je potřeba na základě dat získaných z hloubkového senzoru vypočítat šířku, kterou zabírá postava v zorném poli senzoru Kinect. Pro přístup k datům z hloubkového obrazu je potřeba aktivovat *DepthStream*. Následuje ukázka zdrojového kódu:

```
1  this.sensor.DepthStream.Enable();
2  this.sensor.DepthFrameReady += this.Sensor_DepthFrameReady;
```

Na řádku 1 lze vidět aktivaci třídy *DepthStream*. Na řádku 2 následuje přidání obsluhy události *DepthFrameReady* třídy *KinectSensor*. V této události dochází ke zpracování každého snímku hloubkového obrazu.

Ve složce *AppWindow* se nachází soubor *DepthData.cs*, který obsahuje stejnojmennou třídu. Tato třída obsahuje metody pro výpočet výšky, šířky a vzdálenosti lidské postavy od senzoru Kinect. Zde je ukázka zdrojového kódu metody pro výpočet šířky lidské postavy:

```
3  public double RealWidthMillimetrs
4  {
5  get
6  {
7    double b = this.Depth * HorizontalTanA;
8    return this.PixelWidth * 2 * b / this.FrameWidth;
9  }
```

Na řádku 7 lze vidět přepis rovnice (1) do zdrojového kódu. V proměnné *HorizontalTanA* se nachází výpočet pro tangus úhlu 28,75°. Na řádku 8 se nachází konečný

výpočet pro šířku, kterou zabírá lidská postava v zorném poli senzoru Kinect. Jedná se o přepis rovnice (3) do zdrojového kódu. Data z hloubkového obrazu jsou udávána v milimetrech, proto je ve výpočtu zahrnut i převod na centimetry. Výslednou metoda, která vrací šířku postavy v zorném poli senzoru Kinect v centimetrech jsem pojmenoval jednoduše *UserWidth()*. Následuje ukázka zdrojového kódu:

```
10  public double UserWidth()
11  {
12  return this.RealWidthMillimetrs * MillimetrsPerCentimer;
13  }
```

Při výpočtech parametrů z obrazu kostry jsou zahrnuty i konstanty pro kompenzaci ztráty při měření. U výpočtu z hloubkového obrazu tyto konstanty nejsou potřeba, protože naměřené a vypočtené hodnoty jsou vždy od jednoho okraje lidské siluety pro okraj druhý a zde nedochází k žádné ztrátě.

4.4.7 Přiřazení velikosti oblečení

Ve složce *AppWindow* se nachází soubor *ClothingSize.cs*, který obsahuje stejnojmennou abstraktní třídu, ze kterých dědí třídy *MaleSize* a *FemaleSize*. Metody pro přiřazení velikosti oblečení jsou pro obě pohlaví stejná, liší je pouze jinými hodnotami parametrů v podmínkách pro přiřazení velikosti daného oblečení. Následuje ukázka zdrojového kódu metody pro přiřazení velikosti trička pro mužské pohlaví:

```
1  public override string TshirtSize(double width, double length)
2  {
3    if (width > 54 || length > 66) {
4      return "XL";
5    }
6    else if (width > 50 || length > 64) {
7      return "L";
8    }
9    else if (width > 50 || length > 60) {
10     return "M";
11   }
12   else if (width < 50 || length < 60) {
13     return "S";
14   }
15   else {
16     return "N/A";
17   }
18 }
```

Jak lze vidět na řádcích 1 až 18, metoda pro přiřazení velikosti trička má 2 vstupní parametry, kterými je šířka ramen a délka trička. Délka trička je brána jako vzdálenost mezi centrálním ramenním kloubem a centrálním kyčelním kloubem. Stejnojmenná metoda

pro ženské pohlaví se tedy liší pouze hodnotami v řádcích 3, 6, 9 a 12. Podobným způsobem jsou přiřazeny velikosti kalhot.

4.4.8 Odhad hmotnosti uživatele

Pro přesné určení hmotnosti uživatele by bylo zapotřebí vytvořit 3D model postavy, který nemám. Mohu však na základě vlastní postavy, která má standartní hmotnost 82 kg při výšce 194 cm (což odpovídá hodnotě 21,79 BMI – optimální váha [15]) a šířce v pase 36 cm, odhadnout přibližnou hmotnost uživatele. Následuje ukázka zdrojového kódu:

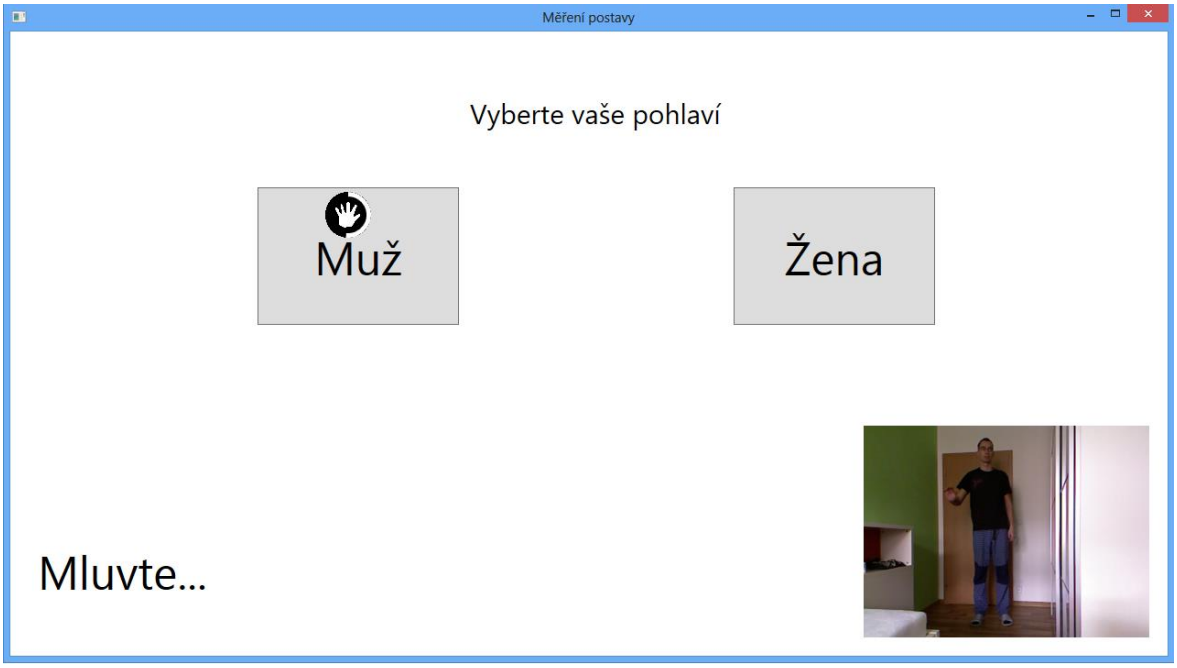
```
1  public string BodyWeight(double height, double width)
2  {
3    string weight = "";
4    if (height > 195)
5    { … }
6    else if (height > 185 && height < 195)
7    {
8      if (width < 37) {
9          weight = "80";
10     }
11     else if (width < 38) {
12         weight = "90";
13     }
14     else {
15         weight = "100 a více";
16     }
17     return weight;
    .
    .
```

Na řádcích 1 až 17 lze vidět částečný úsek metody *BodyWeight()* pro odhad hmotnosti uživatele. Tato metoda má 2 vstupní parametry. Parametr *height* reprezentuje výšku postavy a parametr *width* šířku postavy v pase. Na základě těchto dvou parametrů jsem schopný určit přibližný odhad hmotnosti uživatele. Na řádku 6 ukázky zdrojového kódu lze vidět v podmínce rozmezí výšky uživatele 185 až 195 cm. Za předpokladu, že moje šířka v pase je přibližně 36 cm při hmotnosti 82kg, následují podmínky na řádcích 8 až 16, ve kterých s rostoucí šířkou uživatele v pase přidávám 10kg na hmotnosti.

Pokud je výška postavy uživatele menší než 155 cm, jeho hmotnost je určena jako méně než 40 kg. V opačném případě pokud je výška uživatele větší než 195 cm a jeho šířka v pase větší jak 39 cm, hmotnost je určena jako 110 a více kilogramů. Toto jsou minimální a maximální stanovené hodnoty, které mohou být jako výsledek odhadu hmotnosti uživatele.

4.5 Obsluha aplikace

Obsluha ukázkové aplikace je jednoduchá. Stačí, aby uživatel zapnul aplikaci a postavil se před senzor Kinect. Následně stačí se řídit pokyny na obrazovce. Aplikace kombinuje dva způsoby ovládání - prostřednictvím gest a pomocí hlasových příkazů. Na úvodní obrazovce se nachází dvě tlačítka, která má uživatel na výběr. Tyto tlačítka jsou určena k ovládání pomocí gest – pohybem ruky v prostoru. Jakmile uživatel zvedne jednu ze svých rukou, aplikace ruku rozpozná a na obrazovce se zobrazí obrázek ruky, náhrada za kurzor myši, kterou lze vidět na obr. 27 nad tlačítkem „Muž“. Obrázek ruky je dvojího typu – levá nebo pravá ruka. Je to kvůli tomu, aby uživatel viděl, kterou ruku aplikace používá. Sledovanou rukou bude vždy ta, která je blíže senzoru Kinect. Tlačítka jde také aktivovat klasickým kurzorem myši nebo hlasovými příkazy, viz. kapitola 4.5.1.

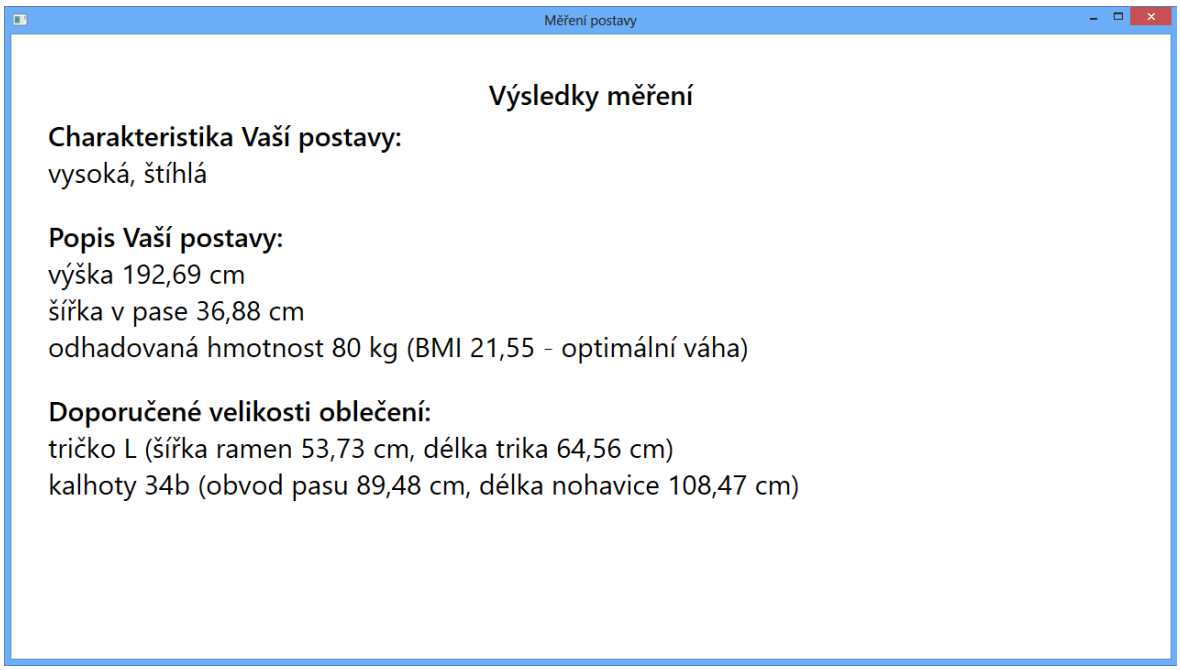


Obr. 28. Uživatelské rozhraní úvodní obrazovky

Po výběru pohlaví se uživatel dostane k hlavní obrazovce, kde se nachází jádro aplikace – měření. Měření postavy se dělí na 2 fáze. V první fázi je vyžadováno, aby uživatel stál čelem senzoru Kinect ve vzdálenosti 2,7 až 2,9 metrů. K druhé fázi měření se je možné dostat po úspěšném skončení první fáze a je v ní vyžadování, aby uživatel stál bokem k senzoru Kinect. Měření začne počínaje správným odstupem uživatele od senzoru Kinect a následně jeho hlasovým pokynem. Každá fáze měření má vlastní hlasový příkaz.

Obr. 29 Uživatelské rozhraní hlavní obrazovky

Po úspěšném ukončení měření se během 3 sekund zobrazí okno s výsledky, které byly na základě měření zpracovány.



Obr. 30. Uživatelské rozhraní obrazovky s výsledky

4.5.1 Hlasové příkazy

Díky hlasovému rozpoznávání mohu aplikaci naučit hlasové příkazy, tzv. gramatiku. Protože doposud nebyla zavedena podpora češtiny v rozpoznávání hlasu, všechny hlasové příkazy jsou v anglickém jazyce. To ovšem ničemu nevadí, protože hlasových příkazů není velké množství a jsou lehce zapamatovatelné. Mezi použité hlasové příkazy patří:

- man
    - volba pohlaví, pokud je uživatel muž
- women
    - volba pohlaví, pokud je uživatel žena
- first
    - spuštění první fáze měření postavy
- second
    - spuštění druhé fáze měření postavy
- restart
    - nové měření postavy
- stop
    - násilné ukončení měření postavy
- close
    - ukončení celé aplikace

4.6 Vhodné podmínky pro aplikaci

Pro správnou funkčnost ukázkové aplikace je potřeba dodržet několik podmínek. Mezi tyto podmínky patří:

- dostatečně velká místnost,
- vhodné oblečení a
- jeden uživatel v zorném poli senzoru Kinect.

Dále se doporučuje umístění senzoru Kinect přibližně ve výšce pasu člověka, ideálně na stole. Místnost by měla mít délku alespoň 3 metry, jelikož aplikace vyžaduje, aby uživatel stál ve vzdálenosti 2,7 až 2,9 metru od senzoru Kinect. U vhodného oblečení je na mysli například tričko a kalhoty. Jde o to vyvarovat se rozevlátému oblečení, které by mohlo způsobit nepřesné měření. Dále také pokud stojí v zorném poli senzoru Kinect více osob, může dojít ke zmatení aplikace a v důsledku toho k chybě měření. Světelné podmínky by neměly ovlivnit funkčnost aplikace, protože pro měření se využívá hloubkové kamery senzoru Kinect a ta využívá technologii infračerveného záření, které je nezávislé na světelných podmínkách.

5 VYHODNOCENÍ MĚŘENÍ POSTAV

Pro ověření přesnosti měření ukázkové aplikace bylo provedeno 5 testů. V těchto testech bylo opakovaně změřeno dvěma metodami pět různých postav (čtyři mužské a jedna ženská postava). Přitom byla snaha o měření různorodých typů postav, aby bylo znatelné, s jakou přesností ukázková aplikace dokáže změřit lidskou postavu a zda doporučené velikosti oblečení odpovídají skutečnosti. Postavy byly změřeny standartní metodou pomocí krejčovského metru, zváženy osobní váhou a výsledky porovnány s navrženou metodou pro měření lidské postavy pomocí senzoru Kinect.

Ve většině případů byly navrženou metodou pro měření lidské postavy vyhodnoceny velikosti oblečení správně a lze usoudit, že tato metoda je úspěšná. Méně úspěšný je odhad hmotností, který se u atypických postav s větší hmotností rozchází s reálnou hodnotou hmotnosti měřené osoby.

ZÁVĚR

Cílem této diplomové práce bylo vytvořit ukázkovou aplikaci, která bude schopna za využití senzoru Microsoft Kinect změřit lidskou postavu a pro tuto postavu následně určit standardizovanou velikost oblečení.

Vytvořená aplikace je postavena na platformě .NET a využívá Kinect for Windows SDK pro přístup k jednotlivým senzorům zařízení Microsoft Kinect. Při tvorbě aplikace byla snaha dle naměřených údajů uživatele nejen vyhodnotit vhodnou velikost pro jeho postavu, ale také jeho postavu charakterizovat. Pro měření lidské postavy jsem použil obě navrhované metody – obraz kostry a hloubkový obraz. Problém nastal u přesnosti, s jakou aplikace dokázala změřit lidskou postavu. Ta nebyla tak přesná, jak bych si představoval, a proto, abych dosáhl větší přesnosti při měření, jsem experimentoval s různými typy postav, od menších postav po vysoké postavy, dle kterých jsem následně kalibroval naměřené hodnoty tak, aby co nejvíce odpovídaly těm reálným. Každý měřený parametr je navíc průměrován ze vzorků přesahujících 150 hodnot. Tento postup se mi jevil jako nejvhodnější a zároveň jsem tak mohl otestovat aplikaci v praktických podmínkách. Většina měření postav proběhla korektně s minimální chybou, což dokazuje efektivitu navržené metody pro měření lidské postavy pomocí senzoru Kinect.

Podařilo se mi úspěšně dosáhnout cíle mého zadání. Vytvořenou aplikací jsem ověřil schopnost využití senzoru Microsoft Kinect nejen v zábavním průmyslu pro ovládání her, tedy jako pohybový ovladač, ale také jako měřicí přístroj pro měření lidské postavy. Práce s tímto zařízením je zajímává, stejně tak jako jeho možnosti využití.

ZÁVĚR V ANGLIČTINĚ

The aim of this diploma thesis was to create a sample application that will be capable of using the Microsoft Kinect sensor to measure the human figure and for this figure determine the standardized dress size.

Created application is built on the platform. NET and uses the Kinect for Windows SDK to access all the sensors of Microsoft Kinect device. When creating the application was an attempt by the measured data of the user not only to evaluate the appropriate size for his figure, but also his figure describe. For the measurement of human body I use both the proposed method - the skeleton image and the depth image. The problem occurred in the accuracy with which the application was able to measure the human figure. It was not as accurate as I would like, therefore, to achieve greater accuracy in measurement, I experimented with different types of figures, minor figures from the tall, by which I then calibrate the measured values to match those of the most real. Each measured parameter is also averaging of samples exceeding 150 values. This procedure seemed to me most while I was able to test the application in practical terms. Most measurements of figures were performed correctly with minimal error, which proves the effectiveness of the proposed method for measuring human body using Kinect sensor.

I managed to successfully accomplish objective of my task. With created application I verified the ability to use Microsoft Kinect sensor not only for entertainment industry - as a motion controller, but also as a measuring device for measuring human body. Work with this device is interesting, as well as its possibilities.

SEZNAM POUŽITÉ LITERATURY

[1] Microsoft – CNET News: *Timeline - Look back at Kinect´s history,* [online]. ©2013 [cit. 2013-02-10]. Dostupné z: http://news.cnet.com/8301-10805_3-20035039-75.html

[2] HowStuffWorks: *How Microsoft Kinect Works* [online]. ©1998-2013 [cit. 2013-02-10]. Dostupné z: http://www.howstuffworks.com/microsoft-kinect.htm

[3] BORENSTEIN, Greg. *Making things see: 3D vision with kinect, processing, Arduino, and MakerBot*. Sebastopol,: O'Reilly, c2012, xviii, 416 s. ISBN 978-1-4493-0707-3.

[4] WEBB, Jarrett a James ASHLEY. *Beginning kinect programming with the microsoft kinect SDK*. New York: Apress, 2012, xvii, 306 s. Expert's voice in Microsoft. ISBN 14-302-4104-7.

[5] ABHIJIT, Jana. *Kinect for Windows SDK Programming Guide.* Birmingham: Packt Publishing, 2012, 366 s. ISBN 978-1-84969-238-0.

[6] Kinect for Windows: *Human interface guidelines v1.5*, [online]. ©2012 [cit. 2013-02-16]. Dostupné z: http://go.microsoft.com/fwlink/?LinkID=247735

[7] MSDN – the Microsoft Developer Network: *Coordinate spaces* [online]. ©2012 [cit. 2013-02-22]. Dostupné z: http://msdn.microsoft.com/en-us/library/hh973078.aspx

[8] PC magazine encyclopedia: *Definition of YUV* [online]. ©1981-2013 [cit. 2013-02-28]. Dostupné z: http://www.pcmag.com/encyclopedia_term/0,1237,t=YUV&i=55165,00.asp

[9] Tabulkavelikosti.cz: *Tabulka velikostí triček* [online]. ©2012 [cit. 2013-02-23]. Dostupné z: http://www.tabulkavelikosti.cz/tricka

[10] Kinect for Windows Blog: *Kinect Fusion demonstrated at Microsoft research* [online]. ©2013 [cit. 2013-04-07]. Dostupné z: http://blogs.msdn.com/b/kinectforwindows/archive/2013/03/06/kinect-fusion-demonstrated-at-microsoft-research-techfest-coming-soon-to-sdk.aspx

[11] KEAN, Sean, Jonathan C. HALL a Phoenix PERRY. *Meet the Kinect, An introduction to programming natural user interfaces*. New York: Apress, 2011, 205 s. ISBN 978-1-4302-3888-1

[12] WATSON, Ben. *C# 4.0: řešení praktických programátorských úloh*. Vyd. 1. Brno: Zoner Press, 2010, 656 s. Encyklopedie Zoner Press. ISBN 978-80-7413-094-6.

[13] PIALORSI, Paolo a Marco RUSSO. *Microsoft LINQ: kompletní průvodce programátora*. Vyd. 1. Brno: Computer Press, 2009, 615 s. ISBN 978-80-251-2735-3.

[14] NAGEL, Christian. *Professional C# 4 and .Net 4*. Indianapolis, IN: Wiley Pub., c2010, 1474 p. ISBN 04-705-0225-8.

[15] MTE: *Výpočet BMI – index tělesné hmotnosti* [online]. ©2013 [cit. 2013-05-05]. Dostupné z: http://www.mte.cz/bmi.php

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

RGB	Red Green Blue
DMO	DirectX Media Object.
SAPI	Speech recognization API.
SDK	Software development kit
USB	Universal Serial Bus
XAML	eXtensible Aplication Markup Language
XML	eXtensible Markup Language
BMI	Body Mass Index

SEZNAM OBRÁZKŮ

SEZNAM PŘÍLOH

PŘÍLOHA P I: MĚŘENÍ POSTAV.

MĚŘENÍ Č.	Standartní metoda (krejčovský metr, váha)										Navržená metoda (senzor Kinect)									
	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Velikost trička**	**Vel. kalhot**	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Vel. trička**	**Vel. kalhot**
1	194	36	54	67	92	109	81	Opt. váha	XL	34	191,4	36,8	54,4	64,3	91,1	109,8	80	Opt. váha	XL	34
2	194	35	54	67	92	108	81	Opt. váha	XL	34	194,2	36,6	54,5	65,3	90,6	109,8	80	Opt. váha	XL	34
3	194	36	54	67	92	109	81	Opt. váha	XL	34	193,9	36,4	53,2	64,3	90,9	110,4	80	Opt. váha	L	34
4	194	36	54	68	92	109	81	Opt. váha	XL	34	194,2	36,5	54,1	65,1	91,4	110,3	80	Opt. váha	XL	34
5	193	36	55	67	92	109	81	Opt. váha	XL	34	194,3	36,2	53,3	65,2	91,0	110,0	80	Opt. váha	L	34
6	194	35	54	67	92	110	81	Opt. váha	XL	34	193,8	36,8	54,5	64,9	91,8	110,3	80	Opt. váha	XL	34
7	195	36	54	67	92	109	81	Opt. váha	XL	34	194,6	36,7	54,3	65,5	90,8	110,7	80	Opt. váha	XL	34
8	194	36	54	68	92	109	81	Opt. váha	XL	34	194,5	36,6	54,3	65,2	90,9	110,2	80	Opt. váha	XL	34
9	193	36	54	67	92	109	81	Opt. váha	XL	34	193,3	36,7	54,7	65,2	90,9	109,5	80	Opt. váha	XL	34
10	194	36	55	68	92	108	81	Opt. váha	XL	34	194,9	36,6	54,4	65,9	90,8	109,6	80	Opt. váha	XL	34

MĚŘENÍ Č.	Standartní metoda (krejčovský metr, váha)										Navržená metoda (senzor Kinect)									
	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Velikost trička**	**Velikost kalhot**	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Velikost trička**	**Velikost kalhot**
1	187	38	56	63	112	106	116	obezita	XL	36	185,5	38,1	55,4	61,2	99,9	106,0	100	nadváha	XL	36
2	187	38	56	63	112	107	116	obezita	XL	36	186,9	38,2	55,2	61,8	110,3	105,8	100	nadváha	XL	36
3	187	38	56	62	113	106	117	obezita	XL	36	186,6	38,5	55,2	61,4	110,8	106,1	100	nadváha	XL	36
4	186	39	57	62	112	106	116	obezita	XL	36	187,2	38,2	55,6	61,5	110,1	106,3	100	nadváha	XL	36
5	187	38	56	63	112	106	116	obezita	XL	36	187,0	38,8	55,5	61,5	99,8	106,3	100	nadváha	XL	36
6	187	38	56	62	112	107	116	obezita	XL	36	186,8	38,6	55,7	61,7	110,5	106,5	100	nadváha	XL	36
7	187	38	57	62	113	107	116	obezita	XL	36	186,8	38,7	55,3	61,8	110,7	105,9	100	nadváha	XL	36
8	186	39	56	63	112	106	116	obezita	XL	36	187,1	38,9	55,8	61,3	110,5	106,4	100	nadváha	XL	36
9	187	38	56	62	112	106	116	obezita	XL	36	186,9	38,7	55,4	61,4	110,8	106,1	100	nadváha	XL	36
10	187	38	57	63	112	106	116	obezita	XL	36	187,2	38,5	55,7	61,8	110,3	106,7	100	nadváha	XL	36

MĚŘENÍ Č.	Standartní metoda (krejčovský metr, váha)										Navržená metoda (senzor Kinect)									
	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Vel. trička**	**Vel. kalhot**	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Vel. trička**	**Vel. kalhot**
1	162	35	53	62	93	85	76	nadváha	XL	36	162,6	34,9	52,2	59,2	92,4	84,9	60	Opt. váha	XL	36
2	162	35	52	62	93	85	76	nadváha	XL	36	161,0	35,2	52,6	58,9	92,8	85,5	70	nadváha	XL	36
3	163	35	52	63	93	85	76	nadváha	XL	36	162,2	35,0	52,1	58,6	92,1	84,8	70	nadváha	XL	36
4	162	35	52	62	93	86	76	nadváha	XL	36	162,6	35,1	52,3	58,2	92,3	86,0	70	nadváha	XL	36
5	162	35	52	62	93	85	76	nadváha	XL	36	161,5	34,8	51,8	59,1	92,4	85,6	60	Opt. váha	XL	36
6	162	36	53	62	93	85	76	nadváha	XL	36	162,4	35,3	52,0	58,6	92,6	85,9	70	nadváha	XL	36
7	162	35	52	62	93	85	76	nadváha	XL	36	162,6	35,3	51,9	58,3	92,8	84,8	70	nadváha	XL	36
8	163	36	52	63	92	85	76	nadváha	XL	36	162,1	35,8	52,8	58,9	92,1	85,2	70	nadváha	XL	36
9	162	36	52	62	93	85	76	nadváha	XL	36	162,3	35,9	52,5	59,0	92,5	85,2	70	nadváha	XL	36
10	162	35	52	62	93	86	76	nadváha	XL	36	162,0	35,2	52,1	58,2	92,6	85,4	70	nadváha	XL	36

MĚŘENÍ Č.	Standartní metoda (krejčovský metr, váha)										Navržená metoda (senzor Kinect)									
	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Vel. trička**	**Vel. kalhot**	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka nohavice [cm]	Hmotnost [kg]	**BMI**	**Vel. trička**	**Vel. kalhot**
1	179	36	54	59	90	97	82	nadvá ha	XL	34	176,2	37,5	55,2	57,3	92,9	96,8	90	nadvá ha	XL	34
2	179	37	54	58	90	97	82	nadvá ha	XL	34	177,1	37,6	55,0	57,8	92,1	97,1	90	nadvá ha	XL	34
3	179	36	55	59	90	97	82	nadvá ha	XL	34	176,8	37,2	55,5	57,5	92,3	96,7	90	nadvá ha	XL	34
4	179	36	54	59	90	97	82	nadvá ha	XL	34	176,4	37,5	55,3	58,1	91,8	96,2	90	nadvá ha	XL	34
5	178	36	54	59	90	96	82	nadvá ha	XL	34	176,9	37,5	55,6	57,6	92,4	96,3	90	nadvá ha	XL	34
6	179	37	54	59	91	97	82	nadvá ha	XL	34	176,8	37,4	55,1	57,1	92,5	97,2	90	nadvá ha	XL	34
7	179	36	54	58	90	97	82	nadvá ha	XL	34	177,2	37,6	55,0	57,1	92,0	96,3	90	nadvá ha	XL	34
8	179	36	54	59	90	97	82	nadvá ha	XL	34	176,5	37,2	55,4	58,0	91,7	96,7	90	nadvá ha	XL	34
9	179	36	54	59	90	97	82	nadvá ha	XL	34	176,2	37,4	55,2	57,6	91,9	96,5	90	nadvá ha	XL	34
10	179	36	55	59	90	97	82	nadvá ha	XL	34	176,4	37,2	55,3	57,8	92,1	96,9	90	nadvá ha	XL	34

MĚŘENÍ Č.	Standartní metoda (krejčovský metr, váha)										Navržená metoda (senzor Kinect)									
	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka noha vice [cm]	Hmotnost [kg]	**BMI**	**Vel. trička**	**Vel. kalhot**	Výška postavy [cm]	Šířka v pase [cm]	Šířka ramen [cm]	Délka pro tričko [cm]	Obvod pasu [cm]	Délka noha vice [cm]	Hmotnost [kg]	**BMI**	**Vel. trička**	**Vel. kalhot**
1	195	42	56	65	111	109	119	obezita	XL	36	192,6	39,0	56,8	63,4	105,0	109,2	110	nadváha	XL	36
2	194	43	56	65	111	109	119	obezita	XL	36	193,2	39,5	55,9	64,0	105,5	109,5	110	nadváha	XL	36
3	195	42	56	65	112	108	119	obezita	XL	36	193,8	39,1	56,3	63,8	105,7	109,1	110	nadváha	XL	36
4	195	42	56	65	111	109	119	obezita	XL	36	192,7	39,6	56,7	64,2	105,1	109,7	110	nadváha	XL	36
5	195	42	56	64	111	109	119	obezita	XL	36	193,3	39,8	56,1	64,6	105,9	108,9	110	nadváha	XL	36
6	194	42	56	65	111	109	119	obezita	XL	36	193,5	39,1	56,5	64,0	105,3	109,2	110	nadváha	XL	36
7	195	42	57	65	112	108	119	obezita	XL	36	193,0	39,0	56,8	63,7	105,9	109,4	110	nadváha	XL	36
8	195	42	56	65	111	109	119	obezita	XL	36	193,2	39,5	56,0	64,2	105,4	109,0	110	nadváha	XL	36
9	195	42	56	64	111	108	119	obezita	XL	36	192,9	39,4	56,7	64,5	105,5	108,8	110	nadváha	XL	36
10	195	43	57	65	111	109	119	obezita	XL	36	193,5	39,5	55,8	64,1	105,5	109,5	110	nadváha	XL	36