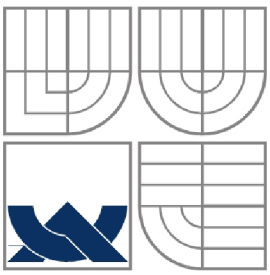
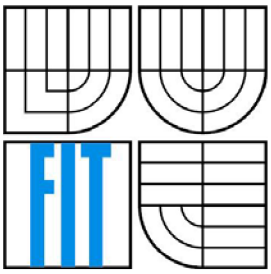


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

OVLÁDÁNÍ POČÍTAČE GESTY

GESTURE BASED HUMAN-COMPUTER INTERFACE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. LUKÁŠ JAROŇ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL ŠPANĚL, Ph.D.

BRNO 2012

Abstrakt

V této diplomové práci jsou popsány možnosti a principy mnou navrženého uživatelského rozhraní, s jehož pomocí lze ovládat počítač gesty. V práci jsou obecně popsány přístupy pro ovládání gesty a detailněji je vysvětlena implementace zvolené detekce rukou a prstů pomocí hloubkové mapy načítané ze senzoru Kinect. Implementace se též věnuje rozpoznávání gest pomocí skrytých Markovových modelů. Pro demonstrační účely je navržena a implementována aplikace pro prohlížení fotografií, jež využívá vyvinutého uživatelského rozhraní. Práce se též zaměřuje na testování kvality a správnosti vyhodnocování pro zvolený rozpoznávač gest.

Klíčová slova

detekce rukou, rozpoznávání gest ruky, skryté Markovovy modely, Kinect, OpenNI, OpenCV

Abstract

This master's thesis describes possibilities and principles of gesture-based computer interface. The work describes general approaches for gesture control. It also deals with implementation of the selected detection method of the hands and fingers using depth maps loaded form Kinect sensor. The implementation also deals with gesture recognition using hidden Markov models. For demonstration purposes there is also described implementation of a simple photo viewer that uses developed gesture-based computer interface. The work also focuses on quality testing and accuracy evaluation for selected gesture recognizer.

Keywords

hands detection, hand gesture recognition, hidden Markov Models, Kinect, OpenNI, OpenCV.

Citace

Jaroň Lukáš: Ovládání počítače gesty, diplomová práce, Brno, FIT VUT v Brně, 2012

Ovládání počítače gesty

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Michala Španěla, Ph.D.. Další informace mi poskytli Ing. Michal Hradiš a Ing. Pavel Žák.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Lukáš Jaroň
23. 5. 2012

Poděkování

Chtěl bych poděkovat především mému vedoucímu Ing. Michalu Španělovi, Ph.D., za pomoc, za poskytnutí nápadů a vedení při práci. Dále bych chtěl poděkovat Ing. Michalovi Hradišovi za dlouhodobé zapůjčení senzoru Kinect a Ing. Pavlu Žákovi za rady ohledně použití senzoru Kinect.

Obsah

Seznam obrázků

1 Úvod

Cílem této diplomové práce je navrhnout a implementovat knihovnu pro detekci ruky, prstů a gest pomocí Kinectu a pomocí této knihovny vytvořit aplikaci pro prohlížení fotografií, která se bude ovládat gesty rukou. Ovládání počítače gesty mě dlouhodobě zajímá, mám v ní i praktické zkušenosti ze své bakalářské práce a vidím v ní do budoucna velký potenciál.

Tato práce se zabývá vytvořením jiného způsobu ovládání počítače než klávesnicí a myší. Z hlediska počítačového vidění lze tento problém řešit několika způsoby. Já jsem si vybral detekci a parametrizaci ruky z hloubkové mapy načtené ze senzoru Kinect od společnosti Microsoft. A za pomocí předem natrénovaných HMM (Hidden Markuv Models), provést rozpoznávání gest ruky, které se použijí pro ovládání počítače.

V kapitole 2 je popsaná základní teorie, která bude použita pro rozpoznávání gest a to především skryté Markovovy modely.

V kapitole 3 je stručný popis již existujících používaných rozhraní pro ovládání aplikací gesty nebo pohledem. Tyto aplikace se rozdělují na dvě části, na ovládaní zařízení pohledem a poté na ovládaní gesty.

Kapitola 4 se zabývá specifikaci senzoru Kinect a stručným popis knihovny OpenNI, která umožňuje načítání informaci z Kinectu a umožňuje využití pokročilých funkcí jako je detekce uživatelů a jejich skeletonizace.

V kapitole 5 se dočtete o návrhu detekce rukou a gest, o návrhu ukázkové aplikaci demonstrující ovládání gesty a o propojení aplikace s rozhraním. Je zde popsán základní koncept GUI a jaké funkce bude podporovat.

V kapitole 6 se dozvíte, jakým způsobem lze ovládat ukázkovou aplikaci pomocí rukou a gest. Jsou zde uvedena gesta, která aplikaci ovládají a také jak lze ovládat aplikaci pomocí rukou.

V kapitole 7 je samotná realizace rozhraní ovládaná gesty, kde je popsáno zpracování hloubkové mapy, detekce rukou, detekce zdvižených prstů, způsob přípravy gest na rozpoznávání a samotné rozpoznávaní gest. Dále pak jaké nástroje a prostředky byly využity k realizaci a jak je aplikace propojena s ovládáním gesty a rukou.

V kapitole 8 se dozvíte, jak bylo testováno úspěšnost rozpoznávání gest a jak byly vyhodnocovány výsledky. A na konci se dozvíte stručný popis, jak by se dal mnou zvoleny rozpoznávač gest dále vylepšovat.

V kapitole 9 je závěr, kde je zhodnocení dosažených výsledků v diplomové práci, zvýrazněné úspěchy a nedostatky navrženého rozhraní pro ovládání počítače gesty a jak by bylo možné práci rozšířit a vylepšit.

2 Teorie pro rozpoznávání gest

Způsob, jakým je provedena v této práci detekce rukou a gest, je popsán v kapitole 7. V této kapitole je teoretický úvod do problematiky rozpoznávání gest ruky, zaměřený především na skryté Markovovy modely dále jen HMM, které budu využity pro rozpoznávání gest nalezené ruky nebo rukou. Je zde uvedena formální definice HMM jejich trénování a rozpoznávání. Ke konci této kapitoly je uveden jednoduchý příklad a zmínka o diskrétní kosínově transformaci, jenž je využita v HMM.

2.1 Skryté Markovovy modely (angl. Hidden Markov Models - HMM)

Zde se pokusím přiblížit základní teorii skrytých Markovových modelů nastudovanou z dokumentů **[1]** a **[3]**. HMM používám v této práci pro klasifikaci gest ruky. Tuto teorii spolu s obrázky jsem čerpal ze své bakalářko práce **[15]**.

Markovovy procesy jsou podobné konečnému automatu s přechodovou pravděpodobností, kde ke každému stavu je přiřazeno určité pozorování tak, že pokud nastane nějaké pozorování, můžeme ho s přesností přiřadit k určitému stavu. Skryté Markovovy modely jsou rozšířením Markovovských procesů, kde nemůžeme přímo říci, že dané pozorování patří k nějakému stavu, ale v každém stavu může nastat jakékoliv z daných pozorování s určitou konečnou pravděpodobností, takže pokud je nějaká vstupní sekvence pozorování, nemůžeme přesně říci, jaké stavy této sekvenci budou odpovídat. Pro toto pozorování je proto sekvence „skrytá“. Můžeme ale vypočítat pravděpodobnost, s jakou model patří k této sekvenci pozorovaní tak, že je nalezena sekvence stavu, která se nejlépe shoduje s daným pozorováním. Takto dokážeme vypočítat, že určité pozorování s určitou pravděpodobností patří k určitému modelu. Pokud vytvoříme více modelů, můžeme vypočítat pravděpodobnost, s jakou se pozorování shoduje s každým modelem a poté vybrat model, který se shoduje s největší pravděpodobností.

HMM je velmi populární statistický nástroj, který se v počítačové technice řadí mezi statistické rozpoznávání a strojové učení. HMM se používají na rozpoznávání sekvenčních dat o různé proměnné délce, pro rozpoznávání řeči, ručního psaní, podpisu nebo lidského obličeje a mají mnohé další uplatnění. Já HMM v této práci budu používat pro rozeznávání gest ruky, kdy jsou jednotlivá gesta převedena na pozorování a dále jsou tyto gesta natrénovány do HMM. Jakmile vytvořím všechny modely, reprezentující všechny třídy gest, začnu pomocí HMM rozpoznávat, o jaké gesto ruky se jedná. Pokud je zařízením detekováno gesto ruky, je vypočítána pravděpodobnost

pro všechny modely a z těchto pravděpodobností je vybrána ta s nejvyšší shodou. Podle vyhodnocené třídy gest, bude provedena předem definovaná činnost, jako vstup do systému

2.1.1 Formální definice HMM

Zde uvedu formální definici, která je z většiny citací z **[1]** a některé části byly upraveny podle **[2]** a **[3]**.

HMM model **λ:**

$$\lambda = (A, B, \pi), \text{ kde} \tag{1}$$

N je počet všech stavů v modelu
M je počet všech pozorování v modelu. Pokud je pozorování nepřetržité, potom **M** je nekonečné
S je konečná množina všech stavů, a **V** je konečná množina všech pozorování:

$$S = (s_1, s_2, \cdots, s_N), \tag{2}$$

$$V = (v_1, v_2, \cdots, v_M), \text{ kde} \tag{3}$$

Q je sekvencí stavu v čase o délce **T,** kde aktuální stav v čase **t** je $\mathbf{q_t}$ a k nim odpovídající pozorování **O,** kde aktuální pozorování v čase **t** je $\mathbf{o_t}$:

$$Q = q_1, q_2, \cdots, q_T \quad , \tag{4}$$

$$O = o_1, o_2, \cdots, o_T \text{ , kde} \tag{5}$$

A je přechodové pole, kde jsou ukládány přechodové pravděpodobnosti mezi stavem **j** a **i**, kde **i** je stav, ze kterého se přechází a **j** je stav, kam se jde. Poznámka: Přechodové pravděpodobnosti mezi stavy jsou nezávislé na čase:

$$A = [a_{ij}], \; a_{ij} = P(q_t = s_j \mid q_{t-1} = s_i), \; 1 \le i, j \le N \tag{6}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{7}$$

B je pole pozorování, kde jsou uloženy jednotlivé pravděpodobnosti pozorování **k**, vycházející ze stavu **j,** tyto pravděpodobnosti jsou časově neměnné:

$$B = [b_j(k)], \; b_j(k) = P(o_t = v_k \mid q_t = s_j), \; 1 \le j \le N, \; 1 \le k \le M \tag{8}$$

$$\sum_{k=1}^{M} b_j(k) = 1 \tag{9}$$

π je pole pravděpodobností počátečních stavů

$$\pi = [\pi_i], \; \pi_i = P(q_1 = s_i) \tag{10}$$

Dva předpoklady pro vytvoření modelu:

První předpoklad - zvaný Markovova vlastnost, kdy aktuální stav závisí pouze na předchozím stavu - to představuje paměť modelu:

$$P\left(q_t \mid q_1^{t-1}\right) = P\left(q_t \mid q_{t-1}\right) \tag{11}$$

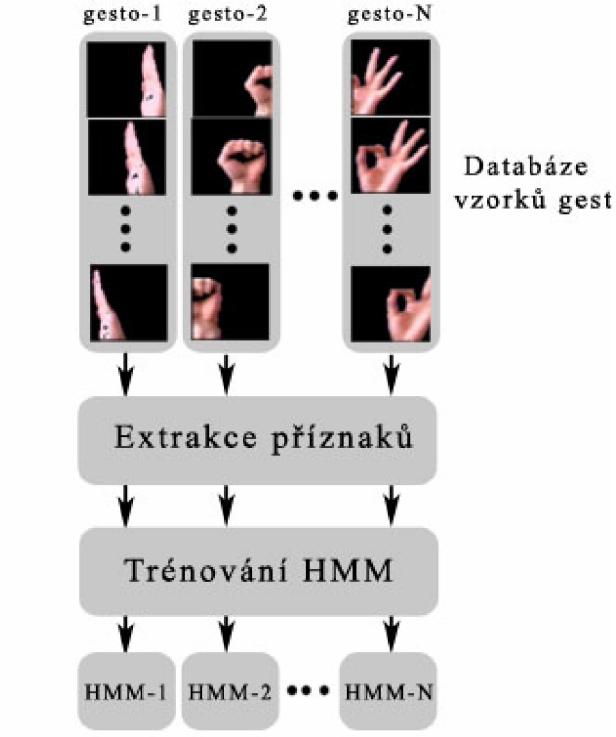
Druhý předpoklad - výstupní pozorování v čase **t** je závislé pouze na současném stavu a je nezávislé na předchozích pozorováních nebo stavech:

$$P\left(o_t \mid o_1^{t-1}, q_1^t\right) = P\left(o_t \mid q_t\right) \tag{12}$$

2.1.2 Trénování HMM

Zde uvedu pouze obecný postup při trénování HMM. Pokud chceme něco trénovat, musíme mít nejdříve co - např.: Databázi slov pro rozpoznávání řeči nebo vzorky snímků tváří pro rozpoznávání obličejů atp. Jelikož člověk nedokáže, aby byl obličej či ruka stejně vzdáleny od kamery nebo říci stejné slovo stejně rychle či se stejnou intonací, musí být v těchto databázích pro každou třídu (např.: slov, obličejů, gest rukou, atd.) dostatek vzorků pro vytvoření HMM. Tím při pozorování správně určíme, o jakou třídu vzorků se jedná. Dále z těchto vzorků musíme vyextrahovat informace a převést je na pozorování, které by co nejlépe říkalo, co se na daném vzorku nachází. Na základě těchto pozorování, ze všech vzorků dané třídy, se provádí trénování HMM, kde se odvodí parametry modelů.

Zde uvedu příklad obecného trénovaní HMM, který můžete vidět na Obrázek 1.



Obrázek 1: Příklad obecného trénování HMM.

2.1.3 Rozpoznávání

Hlavním principem rozpoznávání je objevit skrytou posloupnost stavu, která by co nepravděpodobněji odpovídala příchozí neznámé sekvenci pozorování (např.: slovo, obličej, gesto ruky, atp.). Dále pak objevit pravděpodobnost, s jakou se model shoduje s tímto neznámým pozorováním. Pro nalezení jedné nejlepší posloupnosti stavu pro dané pozorování se používá Viterbiho algoritmus. Tento algoritmus se snaží nalézt největší přechodovou pravděpodobnost v každém kroku na základě nejlepší nalezené shody s pozorováním. Nejdříve musíme definovat **[1]**:

$$\delta_t(i) = \max_{q_1,q_2,\cdots,q_{t-1}} P(q_1, q_2, \cdots, q_t = s_i, o_1, o_2, \cdots, o_t \mid \lambda) \tag{13}$$

jako pravděpodobnost nejlepší nalezené shody s daným pozorováním.

Viterbiho algoritmus je následující:

1. Inicializace:

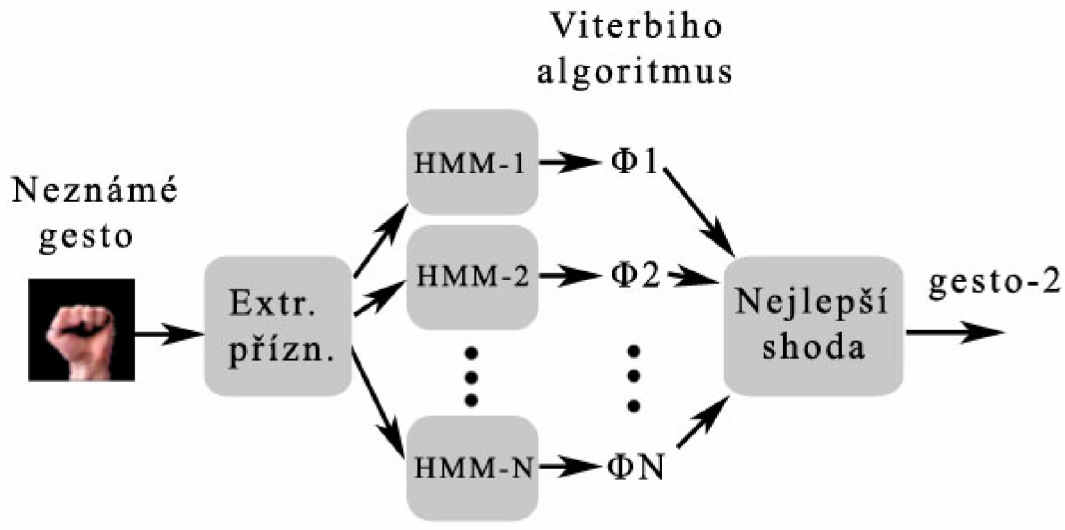
$$\delta_1(i) = \pi_i \cdot b_i(o_1),\ 1 \le i \le N,\ \psi_1(i) = 0 \tag{14}$$

2. Rekurze:

$$\delta_t(i) = \max_{1\le i\le N}\left[\delta_{t-1}(i)\cdot a_{ij}\right]\cdot b_j(o_t),\ 2 \le t \le T,\ 1 \le j \le N \tag{15}$$

$$\psi_t(j) = \arg\max_{1\le i\le N}\left[\delta_{t-1}(i)\cdot a_{ij}\right],\ 2 \le t \le T,\ 1 \le j \le N \tag{16}$$

Rozpoznávání neznámého vzorku (jako je slovo, obličej, gesto atd.) se provádí tak, že se vzorek převede na pozorovací podobu a Viterbiho algoritmem se vypočítá pravděpodobnost, na kolik procent se každý model, který jsme natrénovali, shoduje s daným pozorováním a poté nalezneme nejlepší shodu. Viz Obrázek **2**:

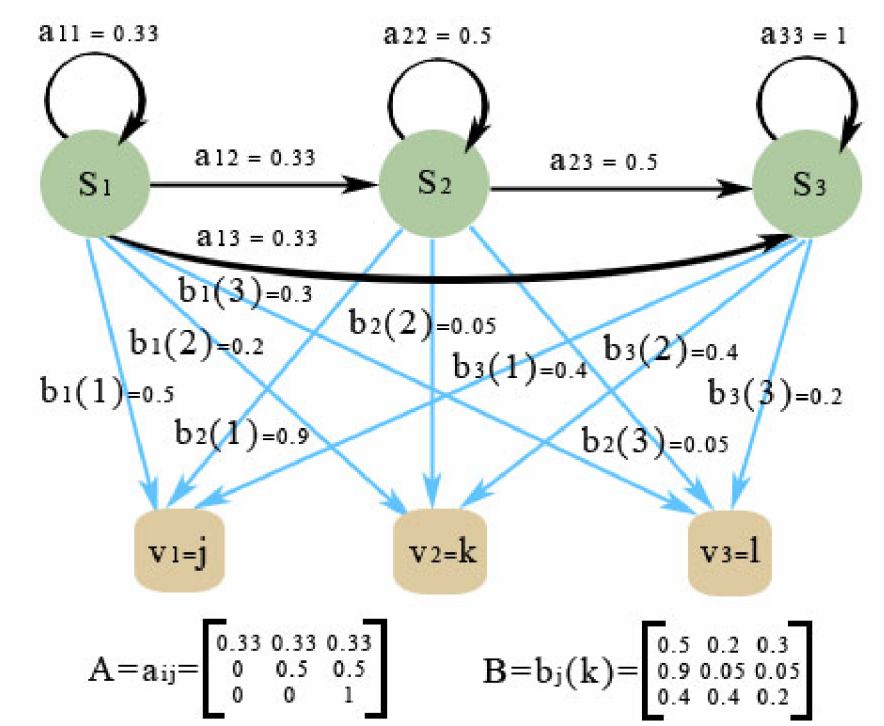


Obrázek 2: Ukázka rozpoznávání.

2.1.4 Jednoduchý Příklad HMM

Toto je velice jednoduchý příklad HMM pouze se třemi stavy. V každém časovém kroku je možné pozorování j, k nebo l. Můžeme pracovat s pravděpodobností, že sekvence pozorování [j, j, k, l] byla vygenerována tímto HMM tak, že hledá největší přechodovou pravděpodobnost stavů, která by se vytvořila z tohoto výstupního pozorování, a výpočetní pravděpodobnost (což je pravděpodobnost každého pozorování vynásobená pravděpodobností přechodů). Nejpravděpodobnější posloupnost přechodů stavů v tomto případě (začneme-li ve stavu 1) je: [1,2,3,3]. Příklad byl převzat ze stránky**[4].**

Jak tento model vypadá, můžete vidět na Obrázek **3**, na kterém jsou zobrazeny všechny hodnoty tohoto modelu.



Obrázek 3: Nákres jednoduchého příkladu HMM, kde jsou pozorovány pouze j, k nebo l.

V tomto příkladě jsou tedy 3 stavy S = (s1, s2, s3) a pouze 3 pozorování V= (v1,v2, v3), kde v1=j, v2=k, v3=l. Sekvence pozorování O = (o1, o2, o3, o4), kde o1=j, o2=j, o3=k, o4=l, těmto pozorováním musíme najít odpovídající sekvenci stavu Q = (q1, q2, q3, q4), kde q1=s1, takže v tomto stavu je o1=j a tomu odpovídá P(O|λ) = b1(1)*a12*b2(1)*a23*b3(2)*a33*b3(3) = 0.5*0.33*0.9*0.5*0.4*1*0.2 = 0.00594 - toto je největší pravděpodobnost, že se tato sekvence pozorování shoduje s tímto modelem.

Pokud bychom vytvořili více modelů, vypočítáme pravděpodobnost, na kolik % se každý model shoduje s pozorováním a vybereme ten, který se shoduje nejvíce.

2.2 Diskrétní kosinová transformace

Diskrétní kosinová transformace (angl. Discrete Cosinus Transform), dále jen DCT, je obdobou diskrétní Furierovovy transformace (DFT), ale vytváří pouze reálné koeficienty. DCT se v praxi používá pro zpracování signálu nebo obrazu, především pro ztrátovou kompresi. Například v obrazových formátech jako je JPEG (angl. Joint Picture Encoding Group), nebo modifikace ve zvukových formátech jako je MP3.

$$X_{k1k2} = \sum_{n1=0}^{N_1-1} \cdot \sum_{n2=0}^{N_2-1} \cdot x_{n1n2} \cdot \cos\left[\frac{\pi}{N_1} \cdot \left(n_1 + \frac{1}{2}\right) \cdot k_1\right] \cdot \cos\left[\frac{\pi}{N_2} \cdot \left(n_2 + \frac{1}{2}\right) \cdot k_2\right] \quad \textbf{(17)}$$

Vzorec 17 ukazuje vypočet 2D DCT-II, převzaty ze stránek wikipedie **[5],** který se obecné používá pro výpočet vícerozměrné 2D diskrétní kosinusové transformaci.

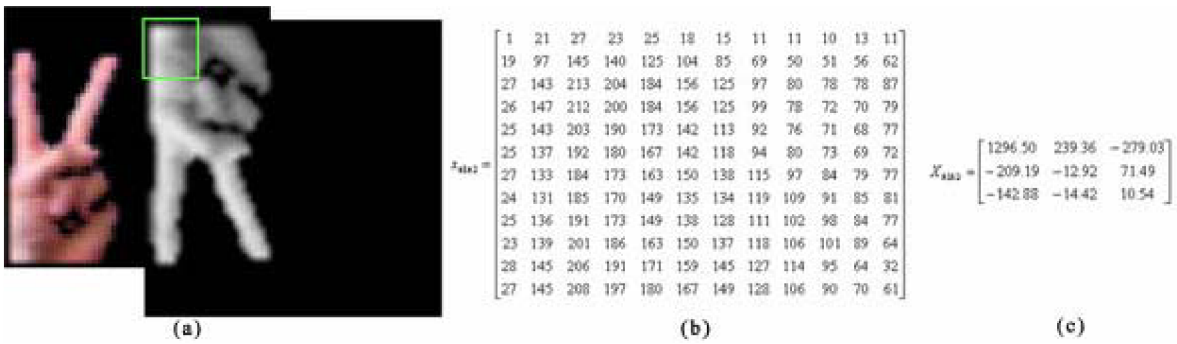
DCT v mé práci je použita pro převedení obrazu na pozorovací strukturu pro práci s HMM. Následující vzorec je odvozen z 2D DCT-II **[5]** a mých pokusů s tímto vzorcem při implementaci, který jsem zjistil při tvorbě mé bakalářské práce **[15]**. Protože většina vzorců je uzpůsobena pro bloky 8x8, ale já zde používám blok 12x12, musel jsem tento vzorec pozměnit tak, aby odpovídal.

$$C_{k1} = \begin{cases} 1 & \text{if } (k_1 == 0 \;\&\&\; k_2 == 0) \\ \frac{1}{\sqrt{2}} & \text{else} \end{cases}$$

$$C_{k2} = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } (k_1 \geq 1 \;\&\&\; k_2 \geq 1) \\ 1 & \text{else} \end{cases} \quad \textbf{(18)}$$

$$X_{k1k2} = C_{k1} \cdot C_{k2} \cdot \frac{1}{12} \cdot \sum_{n1=0}^{12-1} \cdot \sum_{n2=0}^{12-1} \cdot x_{n1n2} \cdot \cos\left[\frac{\pi}{12} \cdot \left(n_1 + \frac{1}{2}\right) \cdot k_1\right] \cdot \cos\left[\frac{\pi}{12} \cdot \left(n_2 + \frac{1}{2}\right) \cdot k_2\right]$$

Pro lepší pochopení uvedu názorný příklad na obrázku, ze kterého se vezme blok 12x12 pixelů a ten se za pomocí DCT převede na 3x3 koeficientů - jak můžete vidět na Obrázek **4**:



Obrázek 4: Ukázka použité DCT pro převod obrazu na data pro HMM.(a) Je vzorek gesta ruky převedený do škály šedi a zeleným čtvercem je vymezena oblast 12x12 pixelů, ze které se bude počítat DCT. (b) Je číselné ohodnocení tohoto bloku. (c) Jsou DCT koeficienty z tohoto bloku.

3 Existující rozhraní pro ovládání gesty

V této kapitole stručně popíši základní vlastnosti již existujících rozhraní pro ovládání aplikací gesty nebo pohledem.

3.1 Příklady ovládání pohledem

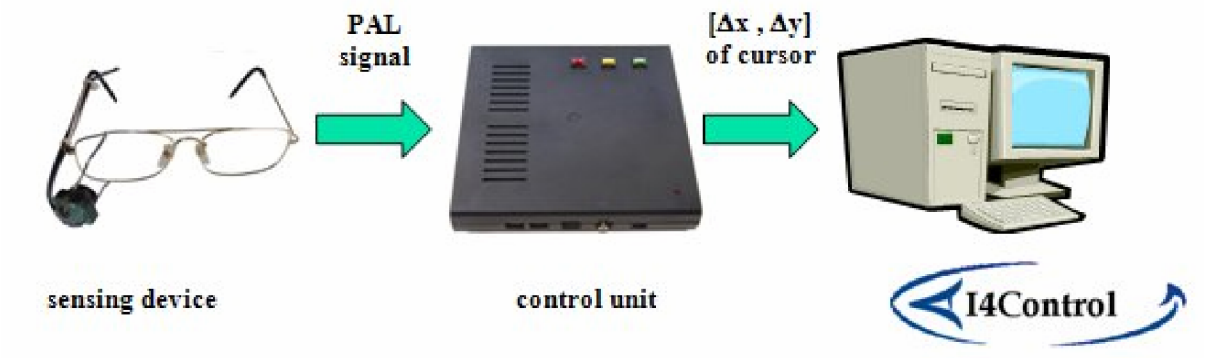
Ovládání aplikací nebo celého počítače pohledem, je technika, kdy se za pomocí kamery snažíme detekovat pozici očí nebo oka, případně spolu s informací o umístění a natočení hlavy v obraze a z těchto informací vypočítat směr pohledu, kam se uživatel s největší pravděpodobností dívá. A když je vypočítán směr pohledu, tak se na jeho základě provede příslušná operace jako např.: posuv kurzoru myší, posuv mezi prezentací pohledem doprava nebo doleva, atd.

Jsou dva základní přístupy z pohledu detekce oka. Sledování oka z bezprostřední vzdálenosti (kamera připevněna k hlavě) nebo sledovaní očí spolu s hlavou ze staticky umístěné kamery, které sebou přináší více možností ovládání, ale za to je výpočetně náročnější, protože musí řešit detekci obličeje a detekci očí. Dva příklady ovládání počítače pohledem naleznete níže:

3.1.1 Systém I4Control®

Představuje novou počítačovou periferii, umožňující bezkontaktní ovládaní osobního počítače za pomocí pohybu oka, který vyvinuli pracovníci a studenti FEL ČVUT katedry kybernetiky. Tento systém nabízí uživateli jednoduchý způsob, jak komunikovat s počítačem prostřednictvím pohybu oka, tím že plně nahrazuje běžnou počítačovou myš.

Základ tvoří malá černobílá kamera, která je pevně uchycena k brýlové obrubě, tak aby snímala pohyb oka uživatele z bezprostřední vzdálenosti a nebránila volnému pohybu hlavy. Kamera snímá aktuální polohu zorničky, systém ji průběžně vyhodnocuje a vysílá počítači pokyny pro pohyb kurzoru na obrazovce. Další částí ovládání je kliknutí nebo dvojklik levého tlačítka myši nahrazeno zavřením oka na určitou dobu. Jak tento systém funguje, můžete vidět na Obrázek 5:

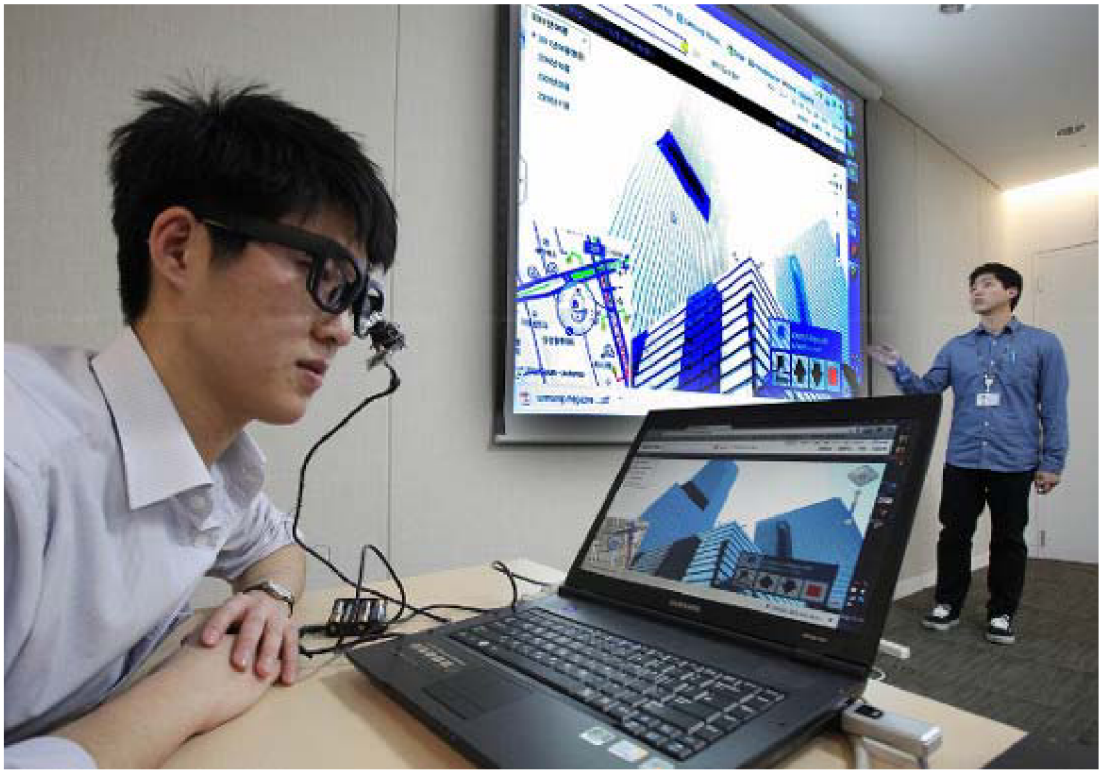


Obrázek 5: Schéma systému I4Control převzato z [9].

Informace o systému I4Control byly čerpány z domovských internetových stránek, **[8]** a **[9]** kde se můžete dočíst více.

3.1.2 Samsung EyeCAN

Společnost Samsung plánuje do konce roku 2012 vyrazit do prodeje s produktem EyeCAN, který umožní ovládání počítače očima. Cílovou skupinou, pro který je EyeCAN vyvíjen, jsou motoricky handicapovaní uživatelé, kteří mají problémy s použitím konvenčních periferií, jakou jsou myš a klávesnice. EyeCAN by měly být brýle snímající polohu oka uživatele, kterou převede jako ovládání kurzoru myší. Podobně jako tomu bylo u systému I4Control, ale značným rozdílem by měla byt cena, která by se měla pohybovat pod hranici 50$ (přibližně 1000 Kč), zatímco systém I4Control se prodává okolo 40000 Kč. Tato výhoda by mohla povzbudit i jiné uživatele, kteří by chtěli ovládat kurzor myši například ve hrách, netradičním způsobem. Na Obrázek **6** můžete vidět uživatele používající EyeCAN.



Obrázek 6: Samsung EyeCAN převzato z **[19]**.

Informace o Samsung EyeCAN z internetového článku „Samsung EyeCAN: ovládání počítače očima“**[19]**.

3.2 Příklady ovládání gesty

Pokus vytvořit kvalitní uživatelská rozhraní ovládaná gesty se po celém světě snaží mnoho výzkumných skupin a programátorů. A to možností ovládaní počítače nebo jiného zařízení od mimik obličeje až po gesta celým tělem. Je mnoho přístupů s různou úspěšností, ale zatím není nějaké univerzálnější řešení. Určité problémy nutné řešit pro detekci gest, jsou popsány v podkapitole 5.1.

Nyní se pokusím popsat některé základní principy pro gesta s rukou. Ruku lze vyhledat i z webové kamery, ale to většinou za ideálních podmínek osvětlení a je nutno řešit spoustu dílčích problémů. Aby bylo možné ruku nalézt, musí se provést detekce kůže, která v ideálních případech nalezne pouze místa, kde se kůže skutečné nachází, ale to se většinou neděje. Pro lepší vlastnosti se ještě před detekcí kůže provádí odfiltrování pozadí. Poté je potřeba v obraze najít pozici hlavy a rukou, které je možné využít v pozdější detekci gest. Poté je provedena detekce gest buď na základě sledování pozic ruky v čase a rozpoznání obrazců, které vznikly pohybem ruky a nebo detekce přemístěním ruky do určité oblasti a podobně. Tato metoda je však velmi náchylná na osvětlení, a tak ji lze málo využít, i když existují přístupy, které tuto problematiku částečně řeší. Další možností je detekce speciální rukavice.

V současné době se objevují přístupy detekce uživatele a rukou pomocí hloubkové mapy vytvořeny 3D kamerami nebo specializovanými senzory, jako je například Kinect, o kterém se můžete dozvědět více v podkapitole 4.1. Za pomoci těchto snímačů hloubky lze získat 3 rozměrnou informaci o snímané scéně, která poskytuje více informací a možností pro zpracování než data z 2D kamery. Navíc již existuje software pro rozpoznávání gest. Při použití senzoru Kinect je s pomocí gest a pohybu těla ovládána celá herní konzole Xbox 360. Už nějakou dobu se programátoři snaží toto ovládaní co nejlépe přenést i na počítače a využívat jeho vlastností. Některé příklady takového ovládání:

3.2.1 Jako v Minority Report

Ke konci roku 2008, představila firma Mgestyk s technologii umožňující ovládání počítače prostřednictvím gestikulace rukou. Tato technologie měla uživatelům přiblížit možnosti ovládaní počítače na dálku gesty, jako ve slavném scifi filmu Stevena Spielberga Minority Report. Gesta jsou rozeznávána pomoci 3D kamery a speciálního softwaru, který umí spolupracovat s většinou aplikací na operačním systému Windows.

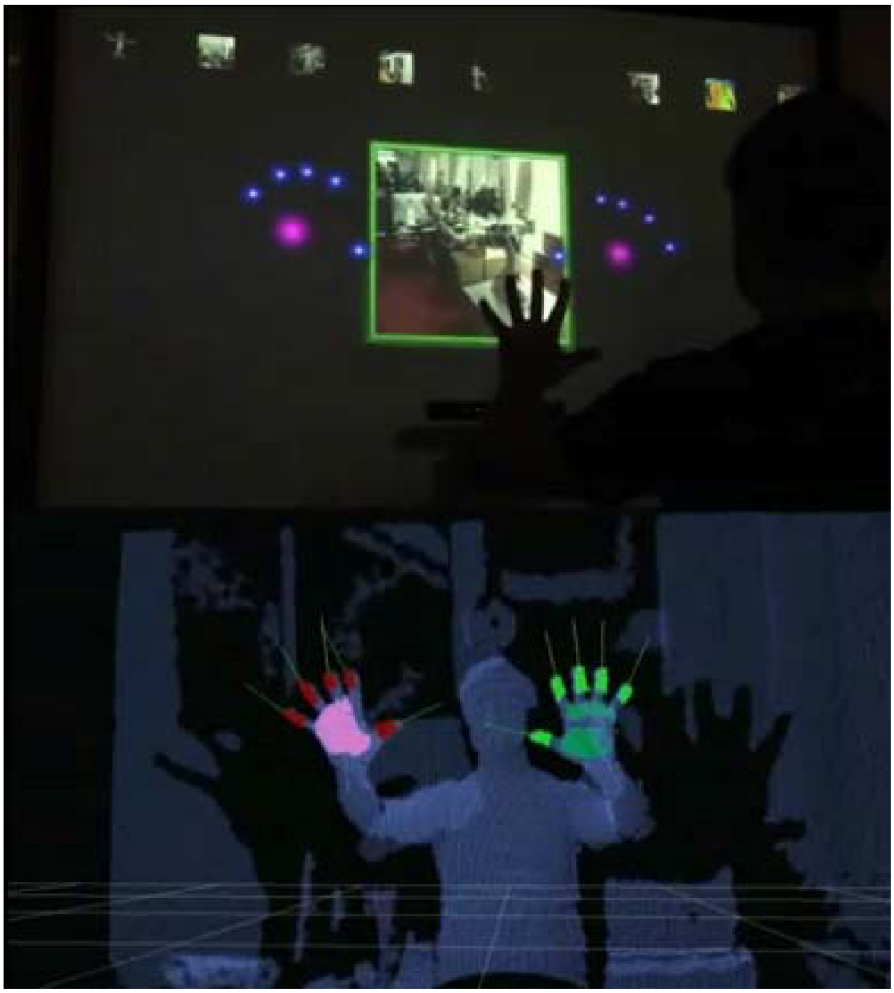
Pro ovládaní programů většinou postačí jednoduché pohyby jedné ruky, v některých případech se pro ovládáni použiji i dvě. Společnost se rozhodla svůj produkt prezentovat především pro ovládání počítačových her, jako jsou akční střílečky, automobilové závody, nebo dokonce i složitější letecké simulátory. V prezentacích se objevily možnosti ovládaní i běžných aplikací, jako je webový prohlížeč, prohlížeče fotografií, manipulace s Gogole Earth, MS paint, MS solitaire a další.

Obrázek 7: Ukázka možností ovládání PC gesty od Mgastyk převzato z **[17]**. 1–Halo: Combat Evolved, 2-Need for Speed 5 (multiplayer), 3-MS flight Simulator 2004, 4-Guitar Hero 3, 5-Need for Speed 5 (in the dark), 6-Web Browning (engadget.com), 7-Google Earth a MS Paint.

Ačkoliv je tento koncept velmi dobrý, firma Mgastyk byla malá a neměla dostatek prostředků pro masovou marketingovou kampaň pro zavedení na běžný spotřebitelský trh a tak se tento produkt moc nerozšířil. Informace o ovládání gesty od Mgestyck byli čerpány z internetového článku „Jako v Minority Report – ovládání PC gesty“ **[18]**, kde se můžete dozvědět více podrobností a zhlédnout i prezentační video.

3.2.2 Detekce rukou s Kinectem

MIT CSAIL (The Massachusetts Institute of Technology's Computer Science and Artificial Inteligence Laboratory) přišla s videozáznamem ukazující způsob detekce rukou za použiti hloubkového senzoru Kinect. Za použití knihoven libfreenect pro práci s Kinectem v linuxu. Grafické rozhraní s detekcí rukou byly napsány na MIT, propojený s open source robotics package 'ROS', který vyvinul Willow Garage. Software pro detekci rukou představuje možnosti Point Cloud Library (PCL), část ROS, kterou MIT pomohlo optimalizovat. Tato detekce rukou je schopna rozlišit ruce a prsty v cloudu (oblaku) větším než 60000 bodů při rychlosti 30 snímků za sekundu, což umožňuje přirozenou interakci v reálném čase. Detekce rukou je předvedena na ovládání prohlížeče fotografií, s kterým uživatel manipuluje pouze za použití rukou a jejich gest.



Obrázek 8: Ukázka z videa detekce rukou s Kinectu převzato z **[20]**.

Informace o Detekci rukou z Kinektu jsem čerpal z informací o videozáznamu na **[20]**, kde můžete tento záznam zhlédnout.

4 Kinect a knihovna OpenNI

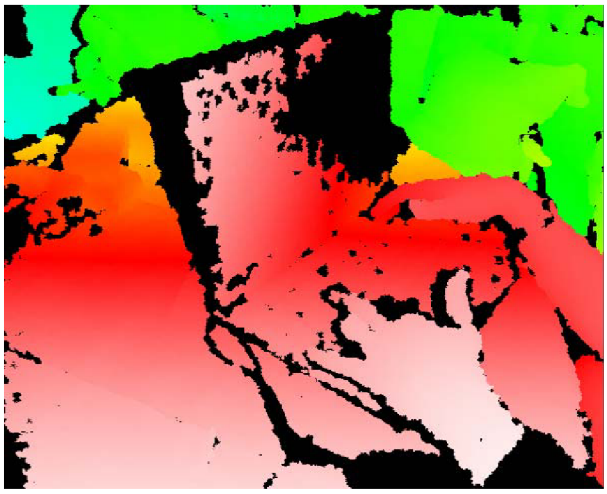
Tato kapitola se zabývá specifikací senzoru Kinect, z kterého se načítá hloubková mapa, tedy pseudo 3D zobrazení prostoru, ve které se provádí detekce rukou. A stručný popis knihovny OpenNI, která umožňuje načítání informací z Kinectu a umožňuje využití i pokročilých funkcí jako je detekce uživatelů a jejich skeletonizace. Více detailů se dozvíte níže.

4.1 Senzor Kinect

V této podkapitole popíši co to Kinect je a provedu jeho detailnější specifikaci. Tyto informace byly čerpány z Wikipedie **[14]**, kde se může dozvědět více informací.

Kinect, původně známý pod kódovým označením „Projekt Natal“, je pohybový senzor, pro herní konzoly Xbox 360 společnosti Microsoft. Toto zařízení je založeno na snímání uživatele za pomoci projektoru, infračervené kamery a speciálního hardwaru. Projektor do prostoru vysílá velké množství infračervených paprsků, infračervená kamera prostor s paprsky snímá a hardware informace převede na hloubkovou mapu a tím předá informace jak je který bod v obraze vzdálen od zařízení v milimetrech. Hloubková mapa je zpracována v Xboxu 360 speciálním softwarem, který umožňuje uživatelům ovládat a pracovat se zařízením za pomocí pohybu těla, aniž by se musel dotýkat nějakého herního ovladače. A tak je celý Box 360 ovládán za pomocí přirozeného a intuitivního uživatelského rozhraní a to vlastním tělem prováděním gest a za pomocí mluvených příkazů. Na Obrázek 9 můžete vidět infračervený snímek ukazující rozložení laserových bodů z projektoru Kinectu pro výpočet hloubkové mapy. Na Obrázek 10, je barevné zobrazení hloubové mapy vypočítané pro předchozí obrázek, od bíle barvy, která označuje nejbližší body až po barvu modrou označující největší vzdálenost. Černě jsou pak označeny body, které jsou zastíněny nebo odraženy od lesklých povrchů a senzor je nemohl rozeznat.

Obrázek 9: Infračervený snímek ukazující rozložení laserových bodů z Kinectu, převzato z **[14]**.



Obrázek 10: Barevné znázornění hloubkové mapy, převzato z **[14]**.

Hloubkový senzor Kinectu není žádnou novinkou, ale společnost Microsoft jako jedna z prvních sestavila toto zařízení tak, aby se jeho cena pohybovala přibližně kolem dvou set dolarů namísto přístrojům poskytujícím hloubkovou mapu v té době, pohybující se cenově v tisících až desetitisících dolarů. A to především kvůli masové výrobě, která umožnila jeho rozšíření do celého světa. Prodej Kinectu byl zahájen 4. listopadu 2010 v severní Americe a během šedesáti dní bylo po celém světě prodáno osm milionů kusů této konzole, tím se Kinect zapsal do Guinessovy knihy světových rekordů jako „nejrychleji prodávaný elektrický spotřebič“ a tento rekord si stále drží. Do 9. března 2011 bylo již prodáno přes deset milionů kusů této herní konzole.

Nyní popíši základní parametry přístroje. Na Obrázek 11, je znázorněné schéma konzole Kinect, spolu se samotným přístrojem, tak i s popisky umístění jednotlivých částí, které podrobněji popíši dále. Konzole Kinect je připevněna k malé základně s motorem, která umožňuje náklon přístroje v horizontálním směru, a je navržena tak, aby byla umístěna podélně pod nebo nad zobrazovacím zařízením (televizor nebo monitor). Přístroj je vybaven RGB kamerou, hloubkovým senzorem (skládající se z laserového projektoru a snímače), LED signalizace a směrových mikrofonů, které jsou umístěné po krajích.

Obrázek 11: Schéma konzole Kinect.

Hloubkový senzor se skládá z infračerveného laserového projektoru v kombinaci s monochromatickým CMOS senzorem, který snímá video s 3D daty za jakýchkoliv světelných podmínek s výjimkou přímého slunečního světla, protože v sobě obsahuje infračervené záření, které senzor má. Společnost Microsoft neuvolnila přesné specifikace přístroje. Za pomoci zpětného inženýrství bylo zjištěno, že video výstup ze senzorů Kinectu posilá až 30 snímků za sekundu. Video z RGB kamery využívá osmi bitový přenos s rozlišením VGA (640x480 pixelů) s Bayerovým barevným filtrem. Monochromatický senzor pro snímání hloubky přenáší video taktéž v rozlišení VGA, ale s jedenácti bitovou informací o hloubce, která poskytuje až dva na jedenáctou úrovni citlivosti tedy v rozmezí 0 až 2048. Hloubkový snímač má praktické omezení vzdálenosti v rozmezí 1,2 až 3,5 metrů od přístroje, při použití softwaru Xboxu 360. Oblast, ve které se tedy musí uživatel nacházet pro ovládání je zhruba šest metrů čtverečních, i když čidlo se dá aproximací rozšířit na rozsah vzdálenosti přibližně 0,7 až 6 metrů. Hloubkový senzor má zorné pole 43° vertikálně a 57° horizontálně, zatímco motorový pivot je schopen k tomu naklánět snímač až o 27° směrem nahoru i dolu. Horizontální pole hloubkového senzoru je při minimální pozorovací vzdálenosti 87 centimetrů a vertikální pole kolem 63 centimetrů, což znamená, že senzor zvládá rozlišit více než 1,3 milimetru na jeden pixel. Dalším příslušenstvím konzole, jsou čtyři směrové mikrofony umístěny po stranách, které pro kanál zpracovávají šestnácti bitové audio na vzorkovací frekvenci 16 kHz.

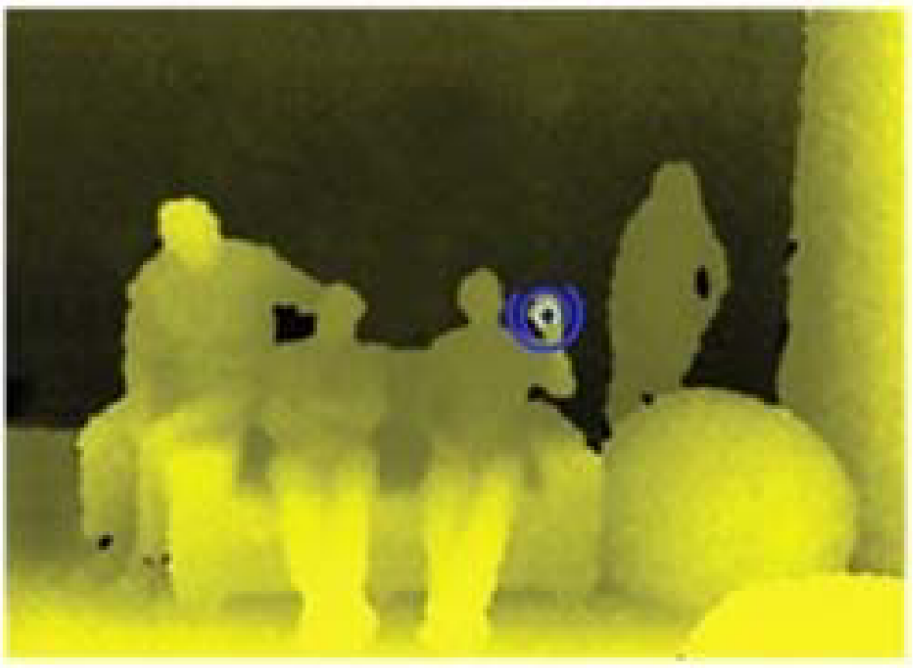
To, co však činí herní konzoly Kinect tak unikátními a chtěnými, není jejich hardware ale software. Ten informace z konzole zpracovává a díky tomu umožňuje ovládání samotného zařízení gesty provedené pohybem těla nebo ovládaní hlasovými příkazy.Softwarové technologie umožňují pokročilé rozpoznávání a detekci gest, rozpoznávání obličejů a rozpoznávání hlasu. Software Kinect pro Xbox 360 je schopen současně sledovat až šest osob, včetně dvou aktivních hráčů. Pro analýzu pohybu se používá extrakce příznaku až dvaceti bodů nalezené z hlavních pohybových kloubů uživatele.

Jakmile tato konzole vyšla, společnost Adafruit nabídla finanční odměnu programátorovy, který vytvoří open-source ovladače pro Kinect do počítače. A již do necelých deseti dnů byla na světě první verze ovladačů. Od té doby vzniká už řada ovladačů, dokonce i společnost Microsoft uvolnila do světa 16. června 2011 nekomerční SDK (Software Development Kit) pro Windows 7. Ale z důvodu využívání operačního systému Windows XP, jsem pro práci s Kinectem použil multiplatformí ovladače OpenNI, které popíši níže.

4.2 Knihovna OpenNI

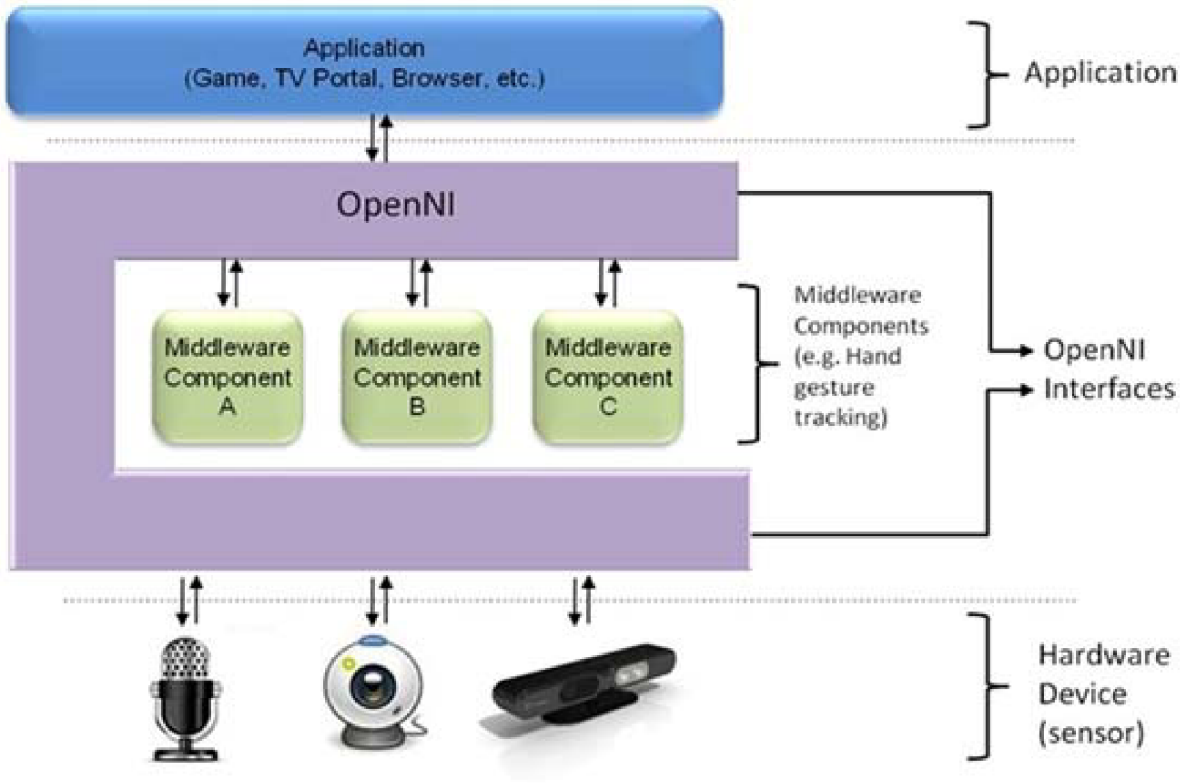
OpenNI (Open Natural Interaction) je vícejazyčný multiplatformní framework, definující API (Application Programming Interface), pro psaní aplikací využívajících přirozené interakce, tedy přirozenou kooperaci mezi člověkem a zařízením, založené na lidských smyslech, zaměřena především na sluch a vidění. Informace k této části byly čerpány z dokumentace OpenNI **[10]**,

kde se můžete dozvědět více podrobností. Hlavním účelem knihovny OpenNI je vytvořit standardní API, které umožňuje komunikaci s vizuálními a akustickými senzory, které vnímají uživatele a jeho okolí. A middlewaru simulující vnímaní vidění a poslechu, což jsou softwarové komponenty, které analyzuje obrazová a zvuková data zaznamenané ve scéně a snaží se tato data pochopit a interpretovat. Příkladem užití je třeba software, který přijímá obrazová data a vrací umístění dlaní zjištěné v obraze. Jak to může vypadat, vidíte na Obrázek 12.



Obrázek 12: Detekce dlaně, převzata z dokumentace OpenNI **[10]**.

Pro pochopení je na Obrázek 13 zobrazen třívrstvý pohled na koncept OpenNI, kde každá vrstva představuje nedílnou součást. Na obrázku nahoře, je vidět aplikační vrstva, představující software, který implementuje přirozenou interakci aplikace nad OpenNI. Střední vrstva, reprezentuje samotné OpenNI, poskytuje komunikační rozhraní, které spolupracuje zároveň se senzory a komponentami middlewaru, který analyzuje data ze snímačů. A nakonec spodní vrstva, kde se nachází hardwarová zařízení, která zachycují zvukové a obrazové prvky scény.



Obrázek 13: Koncept OpenNI, převzatý z **[10]**.

OpenNI umožňuje: Načítání libovolného vizuálního streamu, jako je hloubková mapa, RGB obraz, infračervený obraz atd., a zaznamenávat je nezávisle na vstupním senzoru. Dále ořez videa pro vybrání oblasti zájmu aplikace a navýšení výkonu. Synchronizace dvou rámců, například načítání hloubkové mapy a RGB obrazu tak, aby se překrývaly. Zrcadlení, pokud je snímaný obraz zrcadlově obrácen, tak je generován již správně otočený obraz a díky tomu se nemusí softwarově obracet. Generátor uživatelů, který dokáže rozpoznat uživatele v obraze a dále sledovat jeho pozici. Generátor kostry, který vygeneruje skupinu bodů symbolizující zjednodušenou kostru člověka na uživatelovi a poté je možné sledovat, kde se jaký kloub nachází. A mnoho dalších funkcí.

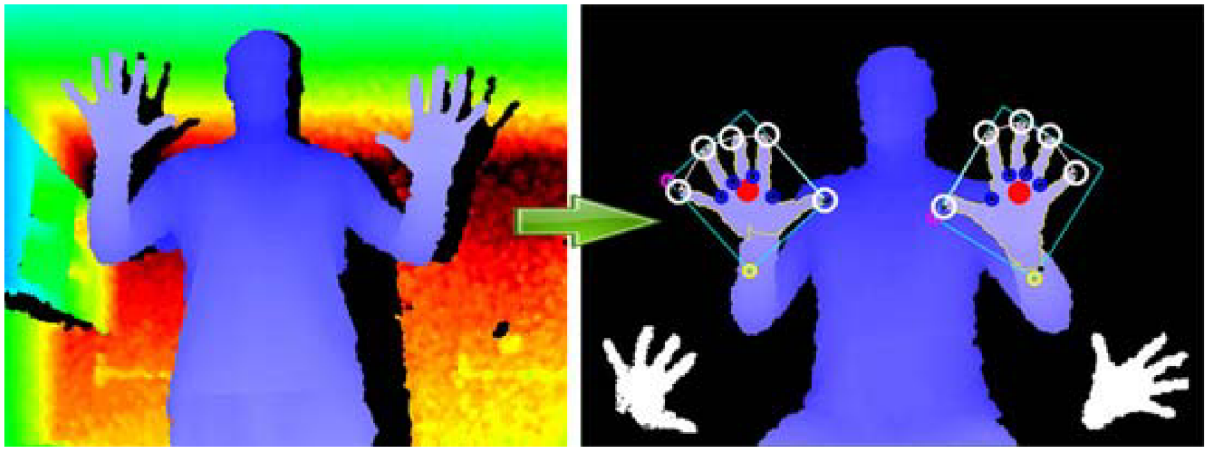
5 Návrh aplikace ovládané gesty

V této kapitole je popsán návrh aplikace ve formě prohlížeče fotografií, demonstrující rozhraní ovládané gesty ruky. Dočtete se jaké problémy tento návrh doprovázely a jak se v průběhu práce pozměňoval. V dalších oddílech této kapitoly se věnuji popisu funkcí a vzhledu grafického uživatelského rozhraní (anglicky Graphical User Interface dále jen GUI), návrhu propojení rozhraní ovládání gesty s ukázkovou aplikací a na závěr návrh detekce rukou a gest.

Obrázek 14: Ukázka vzhledu mnou navrženého prohlížeče fotografií ovládaného gesty.

Návrh detekce rukou a gest popisuje, jakým způsobem získám z hloubkové mapy načtené ze senzoru Kinectu, informace o pozicích rukou a prstů a jakým způsobem detekuji jednotlivá gesta.

Ukázku funkce detekce rukou a gest můžete vidět na Obrázek 15.



Obrázek 15: Převedení hloubkové mapy (vlevo) na informace pro ovládání rukou a gesty (vpravo).

5.1 Motivace pro vytvoření aplikace

Pokud je vytvořen systém pro ovládání počítače, je nutné vytvořit i aplikaci, na které se ukáže jeho funkčnost a možnosti. Já, po poradě s vedoucím mé diplomové práce, jsem se rozhodl vytvořit ukázkovou aplikaci, prohlížeče fotografií, pro zobrazení a správu fotografií, která bude demonstrovat možnosti ovládaná uživatelem, za pomocí rukou a gest snímané ze zařízení Kinect. Tato aplikace by tedy měla ukázat možnosti ovládání počítače nebo jiného zařízení novým a zábavným způsobem.

Ale ještě než se pustíme do návrhu a vytvoření takovéto aplikace, bylo nutné si položit a zodpovědět několik otázek. Např.:

- Co bude aplikace umět a jaké funkce bude podporovat?
- Jaké funkce by byly naopak zbytečné?
- Jaké užitečné vlastnosti prohlížeče fotografií ovládaném na vzdálenost rukou, by se mohly hodit z pohledu uživatelů?
- Kde, kdy a za jakých okolností by se mohla využívat?

Představte si třeba situaci, kdy se vrátíte domů z dovolené nebo zábavy a chcete se sám nebo s rodinou či přáteli podívat na pořízené fotografie. Ve většině případů je budete muset nahrát do počítače a pak si je jednu po druhé za pomoci klávesnice nebo myši prohlédnout. Ale co s tím kdybychom je chtěli ukázat někomu dalšímu? Nebylo by pohodlnější a zábavnější prohlížet si fotografie z pohody vašeho křesla či gauče a ovládat vše na dálku pouhým pohybem ruky? Aplikace by se dala též využít jako reklamní plocha v televizi ve výlohách obchodů. Kde by si zákazník na dálku aniž by se něčeho dotýkal, mohl projít nabídku obchodu, akční ceny nebo informace o jednotlivých produktech atd.

Tato aplikace nemá přinést nový, převratný či úžasný způsob zobrazení fotografií, ale hlavním důvodem pro vytvoření této aplikace je ukázka a možnosti ovládání za pomoci rukou a gest, které není zatím na takové úrovni, aby bylo využito pro celodenní práci ale spíše pro uvolnění, zábavu a trochu větší fyzické námahy, než pouze sedět a pracovat u počítače.

5.2 Návrh GUI pro aplikaci ovládanou gesty

V této podkapitole popíši základní koncept grafického uživatelského rozhraní pro ukázkovou aplikaci prohlížeče fotografií. Funkce a vlastnosti, které bude aplikace podporovat:

- Načtení fotografií ze složky
- Zobrazení načtených fotografií v 3D liště
- Prohlížení fotografií v 3D liště
- Výběr fotografie z 3D lišty a přidání na pracovní plochu
- Odebrání fotografie z pracovní plochy

- Manipulace s fotografiemi na pracovní ploše:
  - Přemisťování pozice vybrané fotografie po pracovní ploše
  - Rotace vybrané fotografie v rozsahu 360° okolo jejího středu
  - Změna zobrazení velikosti fotografie a její zvětšení i zmenšení
  - Zrcadlové otočení fotografie, v horizontálním i vertikálním směru
- Zobrazení pozice bodu pro manipulaci po přiblížení
- Zobrazení pozice dlaní a prstů uživatele pro manipulaci
- Barevné rozlišení pravé a levé ruky
- Zobrazení časovače kliknutí pro bezpečnější manipulaci
- Možnost přepínání mezi třemi módy zobrazení:
  - Pracovní plocha s 3D lištou a tlačítky pro posun po liště
  - Pouze pracovní plocha
  - Zobrazení a prohlížení všech načtených fotografií ze souboru
- Možnost ovládat aplikace:
  - Klávesnicí a myší
  - Pozicí a manipulaci rukou a prstů
  - Gesty

Při tvorbě návrhu bylo důležité popřemýšlet, jaké funkce bude aplikace podporovat, jak bude vypadat její vzhled, aby byl přizpůsobený k metodě ovládání rukou a gesty, a o jejím omezení, které se liší od klasického způsobu ovládání klávesnicí a myší.

Například pokud v případě polohy ruky v hloubkové mapě nalezneme pozici a chceme-li ji přenést do počítače třeba v podobě změny umístění kurzoru myší, nastává problém, protože informace o poloze ruky se berou z hloubkové mapy, která má rozlišení 640x480, ale běžná obrazovka má v dnešní době rozlišení 1920x1200 a vyšší. Z toho důvodu je změna minimální 3x větší oproti klasickému způsobu ovládání myší. Proto tento způsob ovládání není přesný a nehodí se pro důkladnou práci a tak musí být tlačítka a jiné ovládací prvky dostatečně velké, aby se s nimi uživateli dobře a pohodlně pracovalo.

5.2.1 Popis funkcí a vlastností GUI

Zde stručně popíši nejdůležitější funkce a vlastnosti navržené aplikace a zdůrazním body, které se budou muset udělat odlišně než pro klasické ovládání.

Aby bylo možné pracovat s fotografiemi, musí být nejdříve načteny ze složky. V běžných případech postačí otvírací dialog, ale kvůli nepřesnosti nalezení pozice a kvůli rozdílům rozlišení, které jsem popsal výše, by sice práce s otvíracím dialogem byla možná, ale v některých případech nepříjemná a zdlouhavá, což by uživatele při častém provádění znechutilo. Proto bude načítáno 15 fotografií z předpřipravené složky, aby se na nich ukázala manipulace za pomocí rukou a gest.

Po načtení, jsou fotografie převedeny do 3D lišty, která je umístěna na spodní části aplikace. Na této liště se zobrazuje 5 fotografií. Středová fotografie je nejblíže a tím i největší, čím je dále od středu tím více se vzdaluje a opticky zmenšuje. Fotografie v liště může uživatel přesouvat v horizontální rovině a to za pomoci přichycení některé fotografie a tažením do strany rukou nebo myší. Po puštění se animace fotografie srovnají do správné pozice. Též je možné posun ovládat tlačítky ve tvaru šipek umístěných po stranách lišty, nebo gesty a šipkami klávesnice.

Pro zobrazení možností ovládání pohybem rukou je možné po celé pracovní ploše aplikace fotografie přesouvat, měnit jejich natočení a velikost. Dále přidávat více fotografií a vybírat mezi nimi. Změny jsou pouze pohledové a nemění fotografii. Jakým způsobem se dá toto provést a jaká gesta a pohyby je nutné vykonat, se dočtete v kapitole 6 ovládání gesty a rukou.

Další netradiční funkcí je možnost změny varianty pohledu za pomocí gesta nebo stisku klávesy M, která bude přepínat mezi třemi módy zobrazení:

- Mód 0, při kterém se uživateli zobrazuje hlavní pracovní plocha pro manipulaci s fotografiemi a s 3D lištou pro zobrazení a výběr fotografií.
- Mód 1, při kterém se uživateli zobrazí pouze pracovní plocha, pro lepší práci a manipulaci s vybranými fotografiemi.
- Mód 2, který zobrazí všech 15 fotografií ve třech řadách po pěti, pro lepší náhled a s možností si kteroukoliv fotografii přiblížit. Poté se vrátit na náhled na všechny, nebo se posunout na nejbližší fotografie z okolí.

Ale hlavním rozdílem oproti jiným aplikacím je zobrazení pozice rukou a prstů uživatele načítané z Kinectu a možnost ovládat celou aplikaci pohybem rukou a gesty.

5.2.2 Vzhled GUI

Hlavní pracovní plocha má rozměry 1680x1050 a je proložena vertikální černobílým gradientem pro zjemnění vzhledu. Prvky v aplikaci jsou zobrazovány ve třech vrstvách:

1. Pracovní plocha, zde se zobrazují fotografie, s kterými se později manipuluje. A dále se zde zobrazují kruhy označující umístění bodu pro manipulaci s fotografií - otočení a změny velikosti.

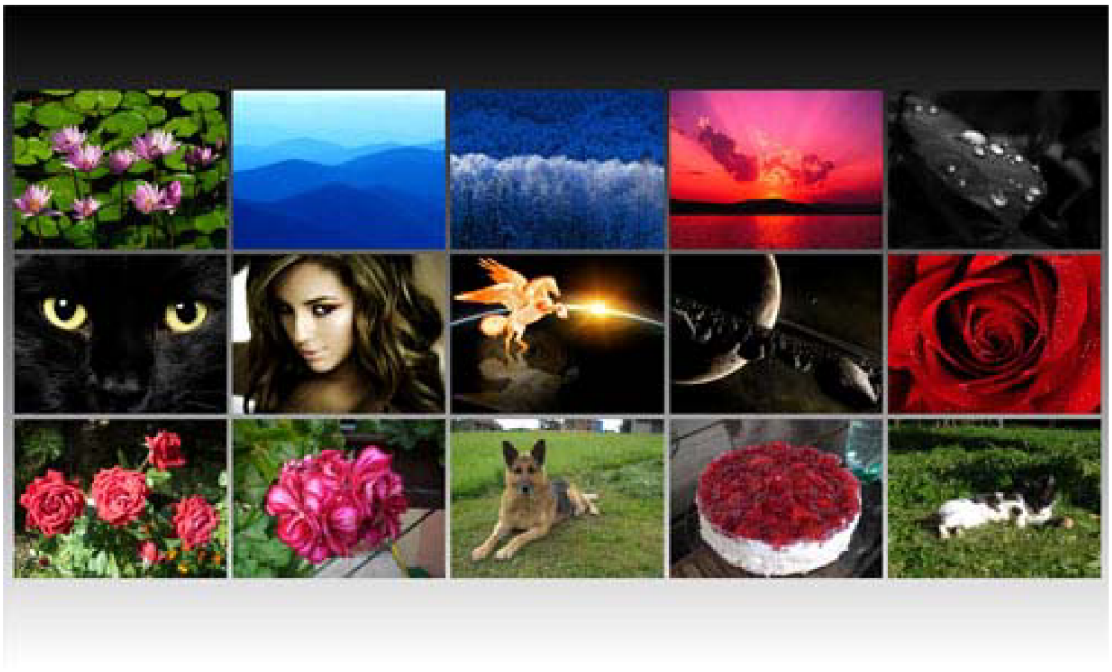
2. V této vrstvě se nachází 3D lišta pro zobrazení načtených fotografií spolu s dvěmi tlačítky po stranách ve tvaru šipek, které lištu ovládají. Lišta spolu s tlačítky je umístěna na spodní části aplikace. V této liště je viditelných 5 fotografií, ale uživatel s nimi může pohybovat a vybírat mezi všemi obrázky z vybrané složky.

3. V poslední nejvyšší vrstvě, která je průhledná, se zobrazuje umístění prstů a dlaní uživatele, ve formě koleček, menší pro prsty a větší pro dlaně. Aby uživatel věděl, kam má intuitivně pohnout rukou, a mohl tak ovládat aplikaci. Přitom barva pro pravou a levou dlaň je rozdílná, aby se rozlišilo ovládání pro levou, nebo pravou ruku, která ovládá pohyb kurzoru myši. Dále se zde zobrazuje kružnice, kterou se začne zvětšovat úhel zobrazení, jakmile je pravá ruka sevřena v pěst, aby po dosažení časového intervalu došlo k simulaci kliknutí tlačítka myši a zamezilo se tak častějším chybám.

Na Obrázek 16 je vidět umístění dvou rukou se zdviženými pěti prsty u každé z nich, které je zobrazeno pomocí fialových koleček, kde velké označuje umístění dlaně a malé umístění prstů.

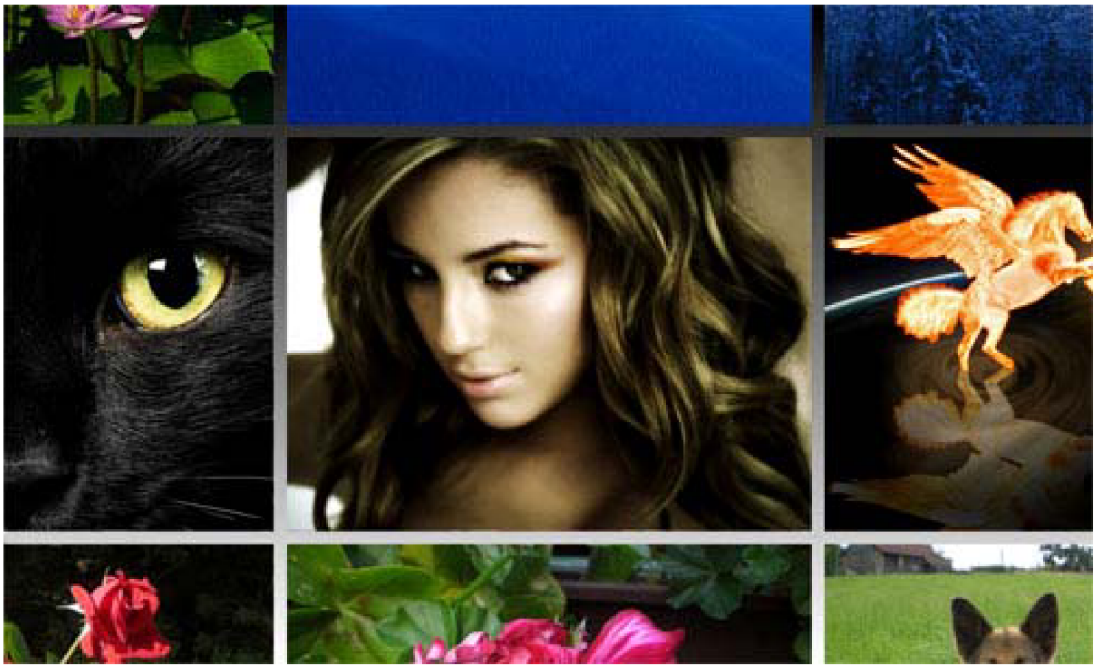
Obrázek 16: Vzhled GUI ukázkové aplikaci Mód 0.

Toto byl popis a zobrazení aplikace v Módu 0. Popis a vzhled módu 1 je prakticky stejný jako u módu 0 až na absenci druhé vrstvy, tedy 3D lišty s tlačítky, které nejsou vidět pro lepší manipulaci s fotografiemi na hlavní ploše. Za zmínku ale stojí vzhled v módu 2, který je odlišný od předchozích. Ruce se zobrazují stejně jako u předchozích módů, ale zobrazení fotografií je jiné. Po přepnutí na tento mód se uživateli zobrazí všech 15 fotografií načtených ze složky a to ve třech řadách po pěti. Toto zobrazení můžete vidět na následujícím Obrázek 17:



Obrázek 17: Vzhled GUI ukázkové aplikace Mód 2 pro všechny fotografie.

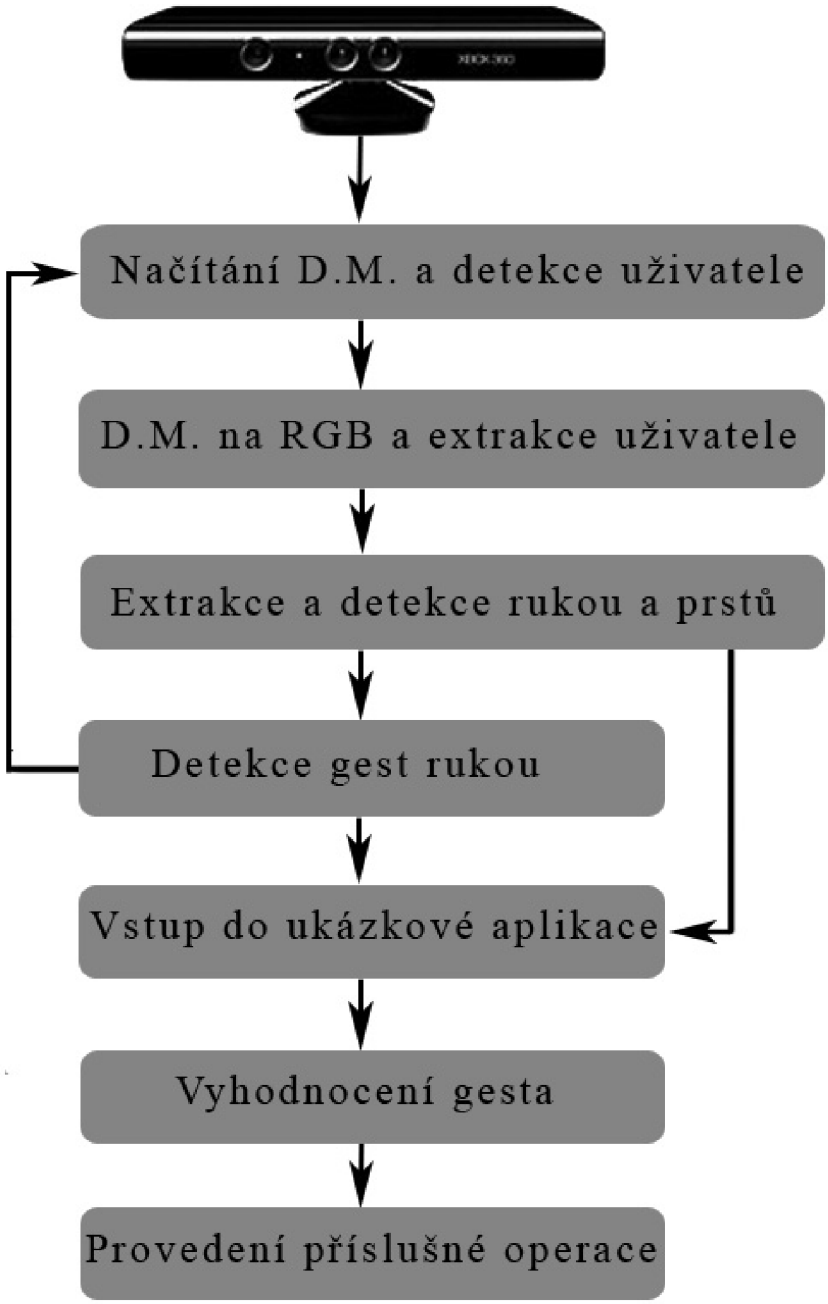
V Módu 2 nelze s fotografiemi manipulovat, lze je pouze přiblížit kliknutím na vybranou fotografii. Po kliku se animací přiblížíte k vybrané fotografii a vidíte jak zvolenou fotografii, tak i její nejbližší okolí, tedy až 8 částečných snímků. Je tedy 8 směrů, kam se kliknutím můžete vydat, tedy po klinutí na vybraný snímek se kamera animací přesune na jeho střed a je možné opět vybírat. Po kliknutí na středový obrázek se kamera přesune animací do původní polohy, kde jsou vidět všechny fotografie. Jak to vypadá, můžete vidět na Obrázek **18** po kliknutí na druhou fotografii zleva ve druhém řádku.



Obrázek 18: Vzhled GUI ukázkové aplikace v Módu 2, při přiblížení k zvolené fotografii.

5.3 Návrh spolupráce rozhraní a aplikace

V této podkapitole budou stručně popsány základní bloky návrhu propojení detekce gest ruky z Kinectu s ukázkovou aplikací. Jak je provedena přesná detekce rukou a prstů a příprava na detekci gest, se můžete dočíst v další kapitole. Na Obrázek 19 je znázorněn Diagram základního konceptu propojení detekce rukou a gest s ukázkovou aplikací prohlížeče fotografií. Pro vysvětlení D.M. znamená Depth Map tedy hloubkovou mapu.



Obrázek 19: Diagram základního konceptu propojení detekce rukou s ukázkovou aplikací.

Načítání D.M. a detekce uživatele:

Za pomocí knihovny OpenNI se načítají hodnoty hloubkové mapy ze zařízení Microsoft Kinect pro zpracování a je provedena detekce uživatele tak, že se za pomocí funkce OpenNI vytvoří maska s hodnotami ID uživatele v místech, kde se vyhodnotilo umístění uživatele.

D.M. na RGB a extrakce uživatele:

V této části se provádí převedení hloubkové mapy na barevnou informaci, aby bylo možné zobrazit, jak vypadá a záznam videa uchovat pro pozdější testování. A je zde provedena extrakce uživatele podle masky, nalezené v předchozím kroku. Musí se najít nejbližší bod uživatele, aby byla provedena extrakce rukou, protože se předpokládá, že uživatel bude mít pro ovládání ruce před sebou.

Extrakce a detekce rukou a prstů:

Z obrazu, kde je již pouze uživatel, se odstraní všechny body, které jsou vzdálenější od určité vzdálenosti od nejbližšího bodu nalezeného v předešlé části. Nalezne se pozice rukou a prstů, která je později předaná ukázkové aplikaci. Vytvoření normalizované podoby tvaru ruky jak velikostně tak natočením, pro pozdější detekci statických gest.

Detekce gest rukou:

Zde se vyhodnotí za pomocí vnitřních podmínek, zdali nenastalo nějaké statické nebo dynamické gesto rukou a za pomocí HMM bude rozpoznáno, o jaké gesto se s největší pravděpodobností jednalo. Tato informace se poté předá jako vstup do ukázkové aplikace, nebo jako vstup do systému. A celý proces se opět opakuje.

Vstup do ukázkové aplikace:

Zde budou přijímány informace z předchozích částí a budou přeformovány pro pozdější zpracování, jako např.: přepočítaní umístění pozic rukou a prstů v závislosti na rozdílných rozlišeních.

Vyhodnocení gesta:

Potom, co jsou informace předpřipraveny, je vyhodnoceno podle pozice rukou a provedeného gesta, jakou činnost a s jakou částí aplikace bude nutné pracovat, nebo předvídat jak akce bude následovat.

Provedení příslušné operace:

Zde se budou provádět veškeré funkce popsané v této kapitole. Na základě vyhodnocených gest a pozic rukou se budou zobrazovat uživateli, který ihned uvidí výsledky jeho ovládání a může na ně intuitivně reagovat. V této části se tedy bude vykreslovat celá aplikace spolu s umístěním rukou a zdvižených prstů v obraze.

5.4 Návrh detekce rukou a gest

V této podkapitole se dočtete jaké kroky jsou potřebné, aby bylo možné detekovat ruce a prsty uživatele z načtené hloubkové mapy ze zařízení Kinect pro mnou navržené ovládání gesty. Kromě detekce rukou a prstů je zde popsaná detekce gest, která dokáže vyhodnocené statické gesto upravit do správného formátu a předat na rozpoznání gest. O tom se podrobněji dočtete až v následující podkapitole. Bližší podrobnosti o detekci rukou naleznete v kapitole implementace.

Mnou navržená detekce rukou a gest musí zvládat tyto funkce:

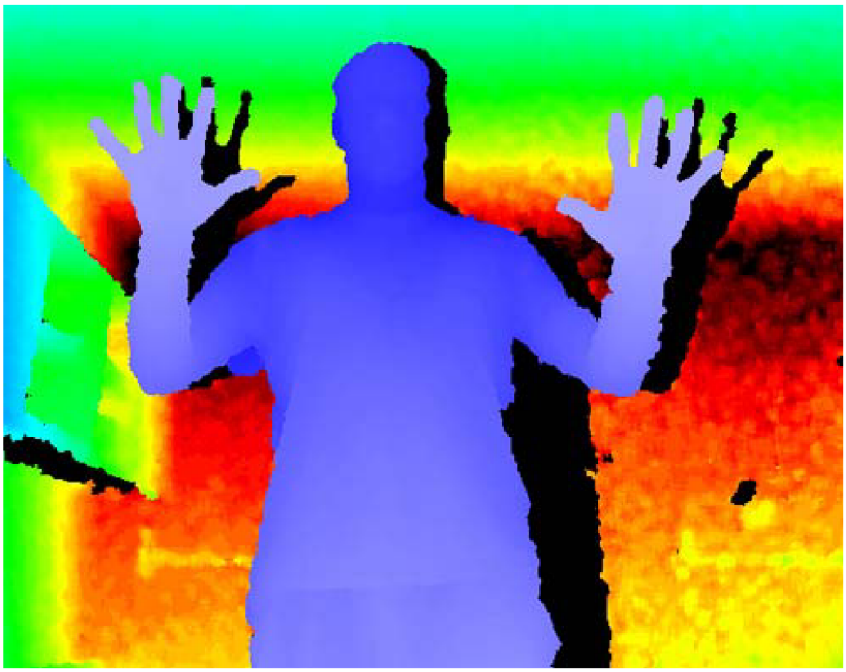
- Práce s Kinectem a načítání hloubkové mapy
- Detekce uživatele
- Detekce rukou a prstů
- Detekce gest

Nyní zde vypíši základní kroky, které je nutné provést pro detekci rukou a gest. Tyto postupy později podrobněji rozvedu s ilustračními obrázky.
Základní kroky:

1. Načtení hloubkové mapy z Kinectu
2. Převedení hloubkové mapy na RGB pro zobrazení
3. Odstranění pozadí a nalezení nejbližšího bodu
4. Odstranění všeho co je dál než nejbližší bod + cca 12 cm
5. Rozdělení na kontury
6. Nalezení pozice a ohraničení rukou
7. Detekce gest
8. Nalezení prstů

V prvním kroku je nutné navázat komunikaci mezi detektorem rukou a Kinectem, aby bylo možné načítat hloubkovou mapu pro pozdější zpracování.

V druhém kroku převedu hloubkovou mapu na barevný RGB obraz pro zobrazení výsledku detekce rukou uživateli. Barevné zobrazení s RGB používám proto, že hloubková mapa nese v každém bodě 11 bitovou informaci o hloubce, což nejde zobrazit, a převedením do 8 bitového obrazu bych ztratil až 5 bitovou informaci o hloubce. Na Obrázek **20**, vidíte stojícího uživatele zabíraného od pasu výše se zdviženýma rukama. Barvy jsou závislé na hloubce, kdy bílá barva je nejbližším bodem, červená barva je nejvzdálenějším bodem a černá barva ukazuje místa, která Kinect nedokázal rozeznat (stíny nebo odrazy).



Obrázek 20: Ukázka barevného RGB obrazu převedeného z hloubkové mapy.

V třetím kroku odfiltruji z obrazu pozadí tak, aby zbyl uživatel, což můžete vidět na Obrázek **21**. To provedu tím způsobem, že porovnávám každý bod obrazu s vygenerovanou maskou uživatelů a tam kde není uživatel, bod začerním, a tam kde je uživatel, hledám nejbližší bod uživatele od Kinectu.

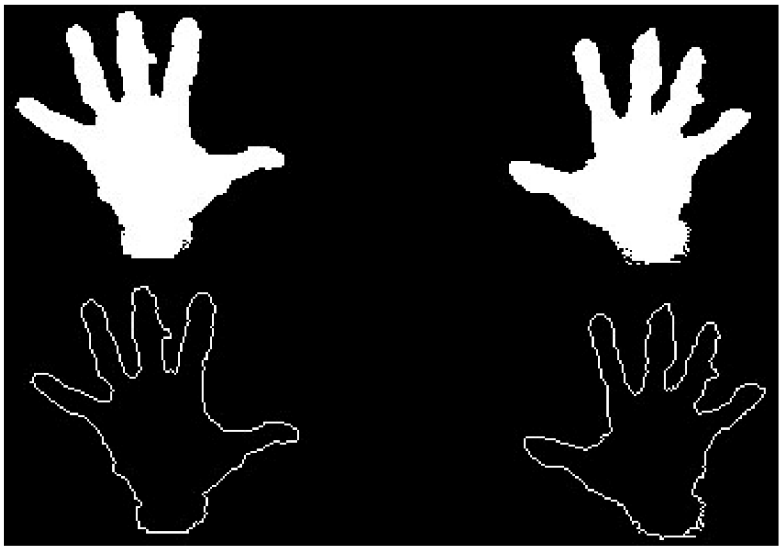
Obrázek 21: Odstranění pozadí.

V kroku 4 vezmu hodnotu nejbližšího bodu uživatele ke Kinectu a přičtu k ní hodnotu120, což má symbolizovat přibližně vzdálenost 12 centimetrů od původního bodu. Dále projdu celý obraz bod po bodu a vytvářím nový černobílý obraz, jehož barvy jsou černé, jestliže je bod v hloubkové

mapě vzdálenější než výše zmiňovaná hodnota, nebo bílé, jestliže se bod vyhodnotí jako úspěšně nalezený. Tímto způsobem vytvořím základ pro detekci rukou, protože předpokládám, že uživatel pro ovládání počítače rukou, musí mít ruce před tělem. Je zde také podmínka, že ruce (dlaně s prsty) musí být alespoň 22 cm před tělem pro ovládání, jinak se nevyhodnocuje, aby se zamezilo chybám a uživatel nemusel mít ruce pořád před tělem a mohl si odpočinout. Ukázku odfiltrování nejbližších bodu ke Kinectu vidíte na Obrázek 22, kde je vlevo zobrazen uživatel a vpravo odfiltrované ruce.

Obrázek 22: Odfiltrovaní nejbližších bodu uživatele od Kinektu.

V kroku 5 se bílé objekty odfiltrované v kroku 4 převedou na kontury, tedy jejich obrysy, jak vidíte na Obrázek 23. Tím začíná hlavní část detekce rukou a gest, kde zpracovávám každou konturu, pokud splňuje požadavky na velikost.



Obrázek 23: Nalezení kontur z obrazu.

Pokud se v kroku pět zjistí, že se nenachází v upraveném obraze žádný objekt, ze kterého by se dala udělat kontura nebo kontura nesplňuje požadavky na velikost, vyhodnocování se ukončí a přejde

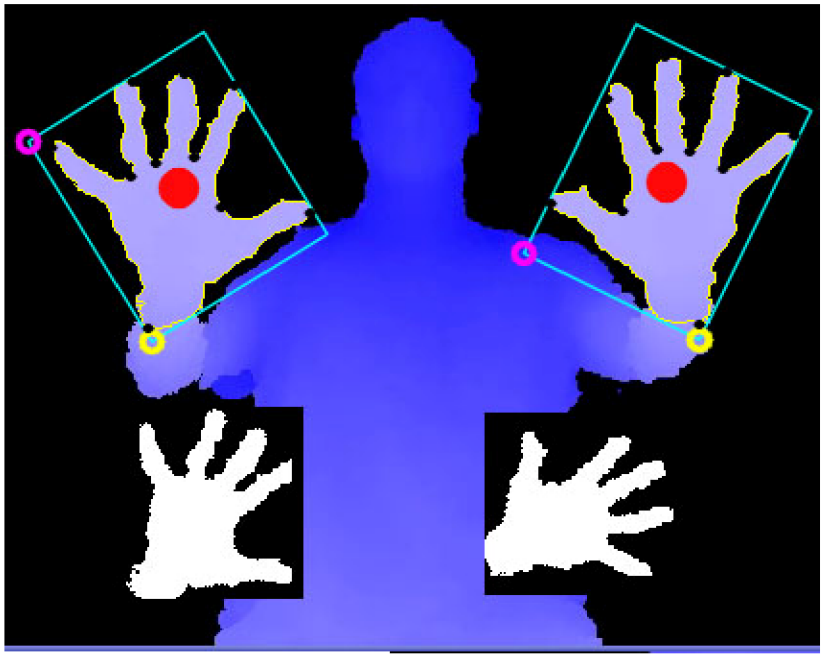
se na krok 1, aby byl vyhodnocen další snímek z Kinectu. Jinak se zpracovává jedna kontura po druhé. A jakmile byly zpracovány všechny kontury, přejde se na krok 1.

V 6 kroku je nutné nalézt ohraničení a umístění každé ruky. To je získáno tak, že se na jednotlivé kontury aplikuje funkce OpenCV pro nalezení nejmenšího ohraničení za pomoci natočeného obdélníku, aby bylo irelevantní menší rozdíly natočení ruky pro pozdější vyhodnocování gest.

Tento obdélník se poté přikreslí do finálního obrazu spolu s červeným kruhem umístěným ve středu obdélníku, který označuje pozici ruky v obraze, kde se tato informace použije pro ovládání kurzoru myši. Tento způsob ohraničení má jednu nevýhodu, a to že se obdélník otáčí pouze od 0° do 90° doprava a poté se přetočí, proto je nutné vytvořit pro trénování sady gest.

V kroku 7 je detektor gest, který se snaží nalézt statická gesta, tedy že je ruka určitou dobu na jednom místě a její tvar se nemění. Detekce gest kontroluje, zda se rozměry a pozice středu obdélníku ohraničující ruku za posledních 30 snímků z hloubkové mapy razantně nezměnili. Pokud byly téměř totožné, je vyhodnoceno gesto a musí se upravit, tedy zpětně natočit a doplnit do čtverce a nakonec normalizovat velikost na 60x60, aby bylo gesto možné poslat na rozpoznávač gest. Pokud ne, rozměry a pozice středu obdélníku pro detekovanou ruku se uloží, což se používá pro nové porovnávání.

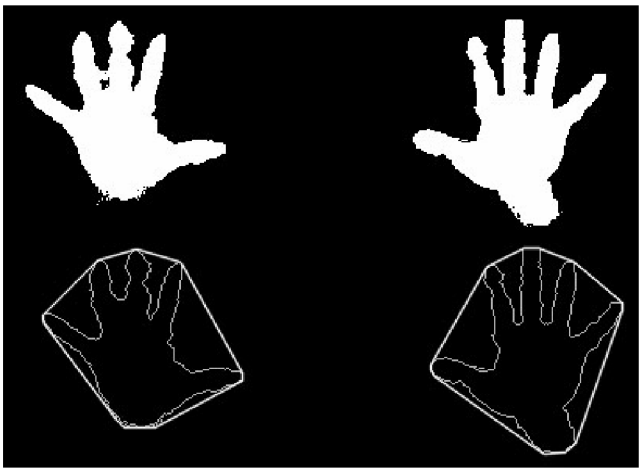
Na Obrázek 24 je ukázka ohraničení rukou a nalezeni jejich pozic spolu s ukázkou gest, která byla detekována a upravena aby byly možné poslat na rozpoznávač gest (dole). Žlutě jsou označeny kontury, červené kolečko symbolizuje umístění ruky v obraze a barvou aqua je označen natočený obdélník, který ruku ohraničuje.



Obrázek 24: Ohraničení a pozice rukou v obraze. A upravené snímky gest pro rozpoznávání.

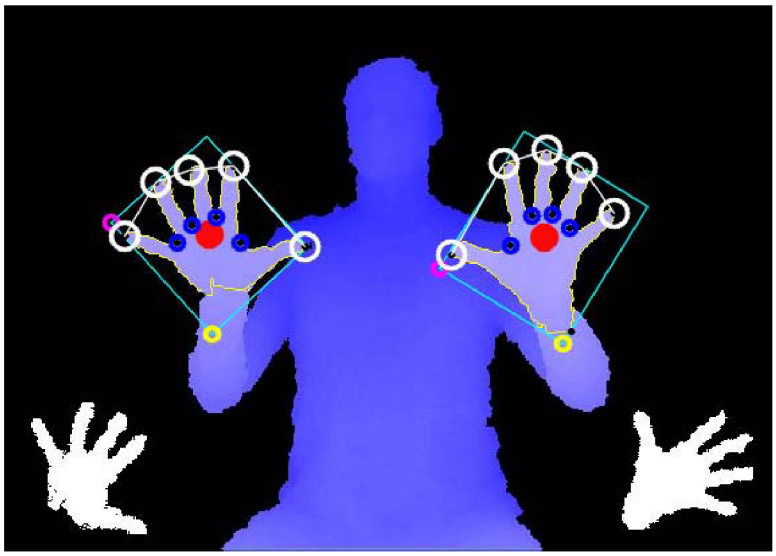
Dále si můžete povšimnout na žluté kontuře černých teček. Tyto body vznikly aproximací křivky kontury pro pozdější nalezení prstů, aby se urychlil výpočet a zamezilo se některý nechtěným chybám.

V kroku 8 se pokouším najit správnou pozici prstů ruky. A to tak, že je nalezena ke kontuře konvexní obálka, která propojuje nejkrajnější body kontury. Na Obrázek 25 je vidět konvexní obálka pro daný obraz.



Obrázek 25: Konvexní obálky z kontury.

Pro nalezení prstu se konvexní obálka nevyhledává pro celou konturu, ale pouze pro body vzniklé aproximací křivky kontury zmíněné výše. To spolu s použitím speciální funkce v OpenCV pro nalezení nejvzdálenějšího bodu kontury od části konvexní obálky pro danou část kontury. Tímto způsobem získáme body, kde se nacházejí prsty, pokud splňují podmínky, že jsou v dostatečné vzdálenosti od sebe a hodnota vzdálenosti od nejvzdálenějšího bodu v kontuře od konvexní obálky je vetší, než přibližná polovina vzdálenosti mezi dvěma prsty.



Obrázek 26: Výsledné detekce rukou, prstů a gest.

Při detekci prstů jsem se inspiroval článkem o rozpoznávání gest ruky **[13]**, kde naleznete jiný způsob detekce ruky z obrazu.

6 Ovládání aplikace gesty a rukou

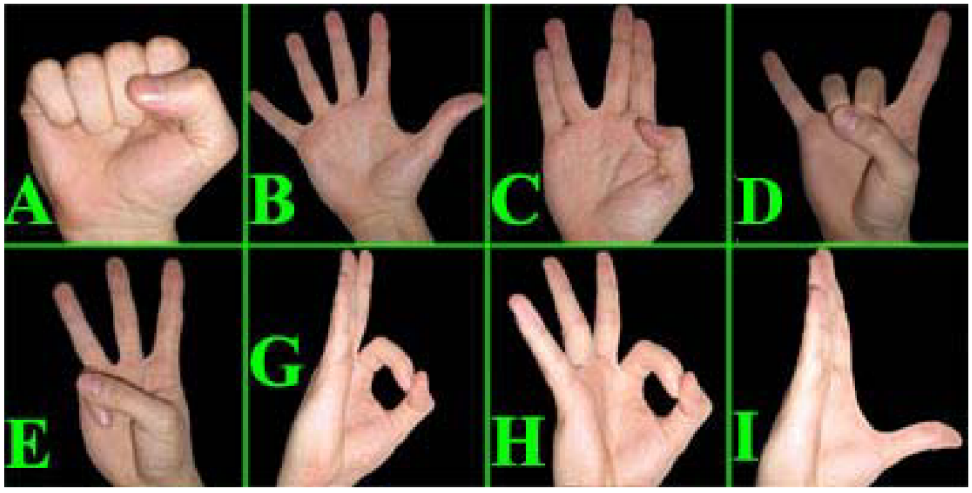
V této podkapitole se dočtete, jakým způsobem jsou ovládány funkce ukázkové aplikace (prohlížeče fotografií). A to především na možnosti ovládání gesty a rukou, jeho výhody a nedostatky a možnosti rozšíření. A způsob ovládaní aplikace pomocí rukou a prstů. A nakonec i možnost ovládání klasickým způsobem pomocí myší a klávesnice

6.1 Ovládání gesty

V této podkapitole vysvětlím princip a druhy gest a také, jak se budou používat pro ovládání funkcí ukázkové aplikace popsané v předchozích podkapitolách.

Gesta se dají provádět různými způsoby od mimiky obličeje, přes pohyb končetin až po držení celého těla. Ovládání pomocí gest může mít širokou škálu možností a využití, ale ne každé gesto je jednoduché na provedení, nebo na jeho časté opakování a tak je nutné se zamyslet, jaká gesta budou nejvhodnější a uživatelsky přívětivá, když budou prováděny opakovaně. Je zde ještě jeden faktor, který omezuje použití některých gest a to je ten, že vyhodnocování nikdy nebude stoprocentní především, když se bude vybírat ze spousty gest, která si budou velice podobná a lišit se budou pouze v maličkostech. Proto je při návrhu nutné vymyslet co nejjednoznačnější gesta, která nebudou moci být zaměněna za jiná a zároveň budou jednoduchá, aby se s nimi uživateli dobře a jednoduše pracovalo.

V této práci budeme rozlišovat pouze gesta provedena rukou. A to statická gesta, kdy uživatel určitou dobu bude držet ruku v určité poloze a při tom bude naznačovat určité gesto. Po uplynutí této doby se gesto vezme na rozpoznání a vrátí hodnotu gesta, o které se s největší pravděpodobností jedná. Pro ukázku možností ovládání gesty, jsem vytvořil sadu 10 statických gest, pojmenované A až J, které aplikace dokáže rozeznat. Tyto gesta můžete vidět na Obrázek 27:



Obrázek 27: Ukázka ovládacích statických gest.

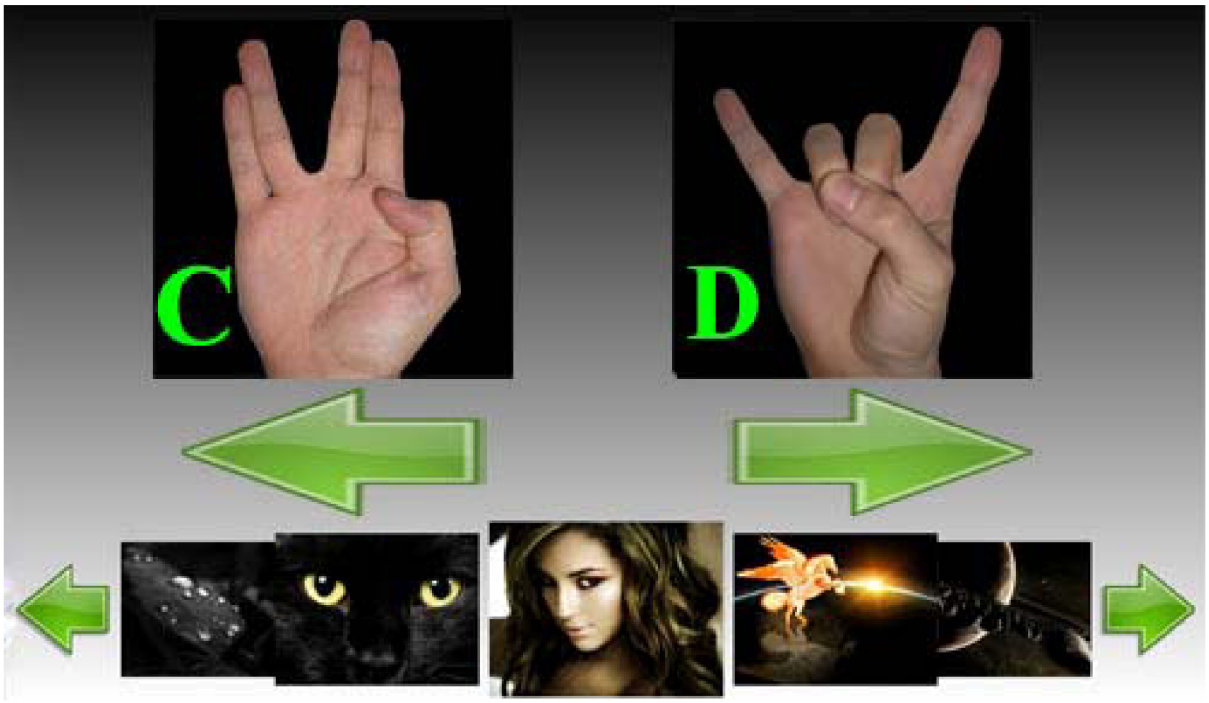
Z těchto gest se prakticky pro ovládání používá šest gest a to C, D, E, G, H a I. Gesta A a B jsou zde přítomná, aby se zamezilo chybám, protože se používají pro odlišný způsob ovládání aplikace pomocí rukou, o kterém bude zmínka později. Gesta F a J byla z rozpoznávání gest odstraněna, kvůli velké chybovosti, kterou způsobovaly při detekci jiných gest. Více podrobností se dočtete v kapitole 8 testování. Navíc gesto J by mohlo být pro dost uživatelů obtížné na provedení.

6.1.1 Ovládání aplikace gesty v módě zobrazení 0 a 1

Výčet gest, která ovládají ukázkovou aplikaci v módě 0:

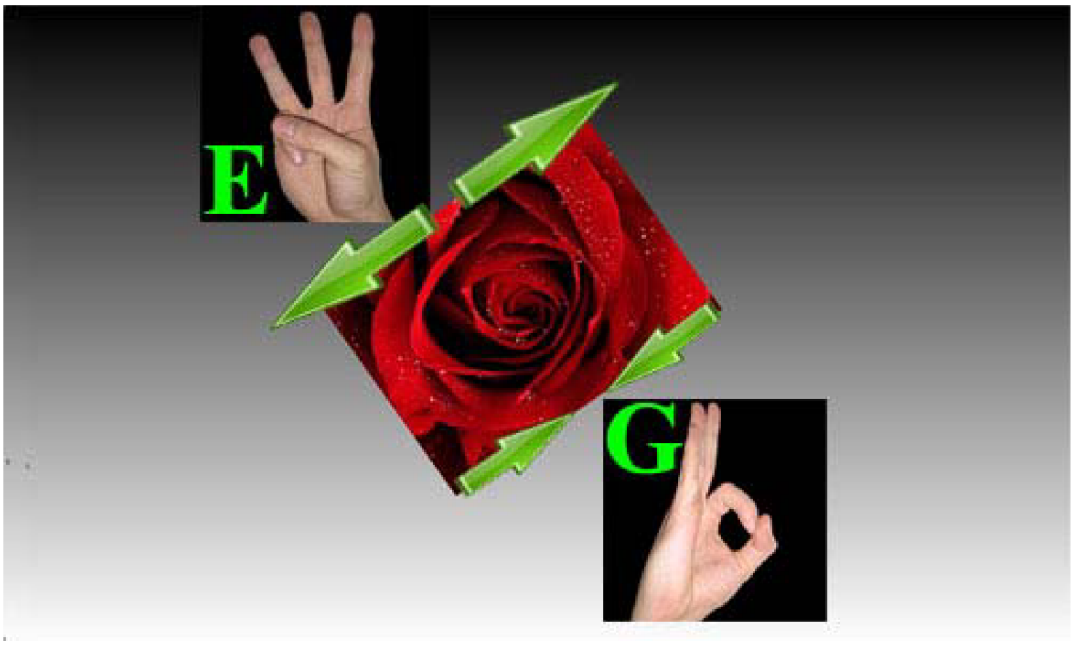
- Gesto C, přesouvá všechny fotografie v 3D liště animaci o jednu pozici doleva
- Gesto D, přesouvá všechny fotografie v 3D liště animaci o jednu pozici doprava
- Gesto E, zvětšuje aktivní fotografii na pracovní ploše o 20 %
- Gesto G, zmenšuje aktivní fotografii na pracovní ploše o 20 %
- Gesto H, přepínání mezi módy zobrazení
- Gesto I, přidá středovou fotografii 3D lišty na hlavní pracovní plochu

Gesta C a D slouží k tomu, aby si mohl uživatel procházet fotografie v 3D liště vždy o jednu pozici doleva nebo doprava. Toto procházení fotografií není pouze pro pohled na fotografie v liště, ale především pro zvolení fotografie, s kterou se má pracovat na hlavní pracovní ploše aplikace. Jak to může vypadat, vidíte na následujícím Obrázek 28, kde je vidět lišta s fotografiemi gesta C a D a kde je prodlouženými šipkami naznačený směr pochybu fotografie po liště při detekování gesta.



Obrázek 28: Nástin ovládání aplikace gesty C a D.

Gesta E a G mění velikost zobrazení aktivní fotografie, tedy fotografie, na kterou bylo naposledy kliknuto nebo která byla naposledy přidaná na pracovní plochu aplikace. Velikost zobrazení fotografie se zvětší nebo zmenší o 20 % původní velikosti. Pro lepší představu je na Obrázek 29 zobrazen nástin akce při provedení gesta E nebo G.

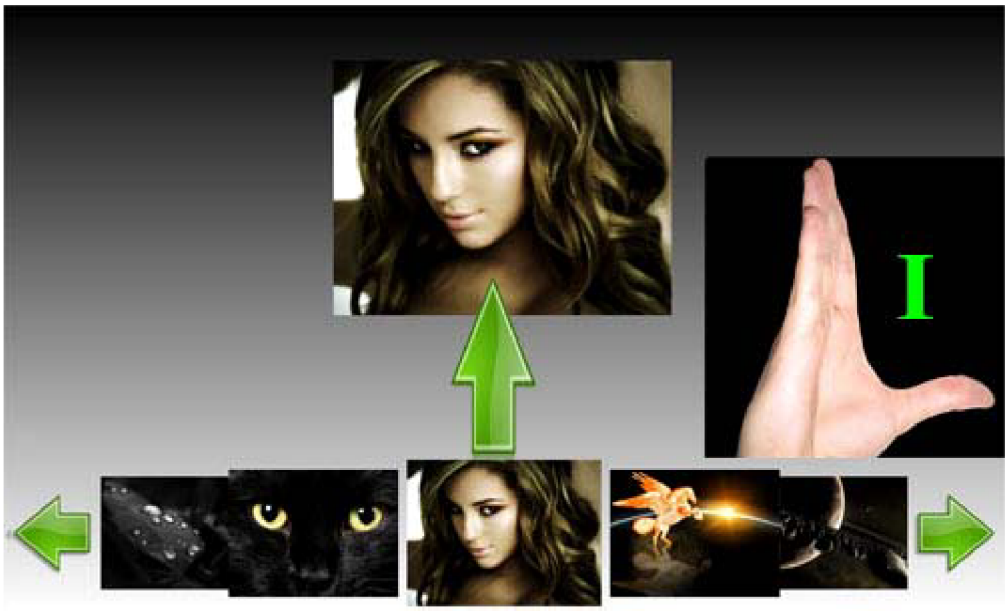


Obrázek 29: Nástin změny velikosti vybrané fotografie pomocí gest E a G.

Gesto H přepíná mezi třemi módy zobrazení fotografií aplikace. Aplikace startuje na módu 0, tedy pracovní plochy s 3D lištou a tlačítky uživatel může přepínat gestem H na mód jedna, kde je pouze pracovní plocha s fotografiemi pro manipulaci a na mód 2, kde je zobrazení všech načtených fotografií najednou a poté zpět na mód 0. Přepínání mezi módy aplikace pomocí gesta H, vidíte na dalším Obrázek 30, kde jsou zobrazeny všechny 3 módy, které aplikace nabízí při použití gesta H.

Obrázek 30: Přepínaní mezi módy zobrazení gestem H.

Pomocí gesta I, uživatel přidá novou fotografii na pracovní plochu pro pozdější manipulaci. Tato fotografie je přidána ze středu 3D lišty. Na Obrázek 31 je zobrazena ukázka přidaní fotografie z 3D lišty na pracovní plochu při rozpoznání gesta I.



Obrázek 31: Ukázka přidání fotografie na pracovní plochu pomocí gesta I.

Toto byly gesta ovládající mód 0, která jsou totožná s módem 1 s tím rozdílem, že práce na 3D liště není vidět, protože je ukryta.

6.1.2 Ovládání aplikace gesty v zobrazení módě 2

Odlišné ovládání je v módě 2. Nyní vypíši výčet gest a jejich ovládání aplikace v módě 2, které později popíši.

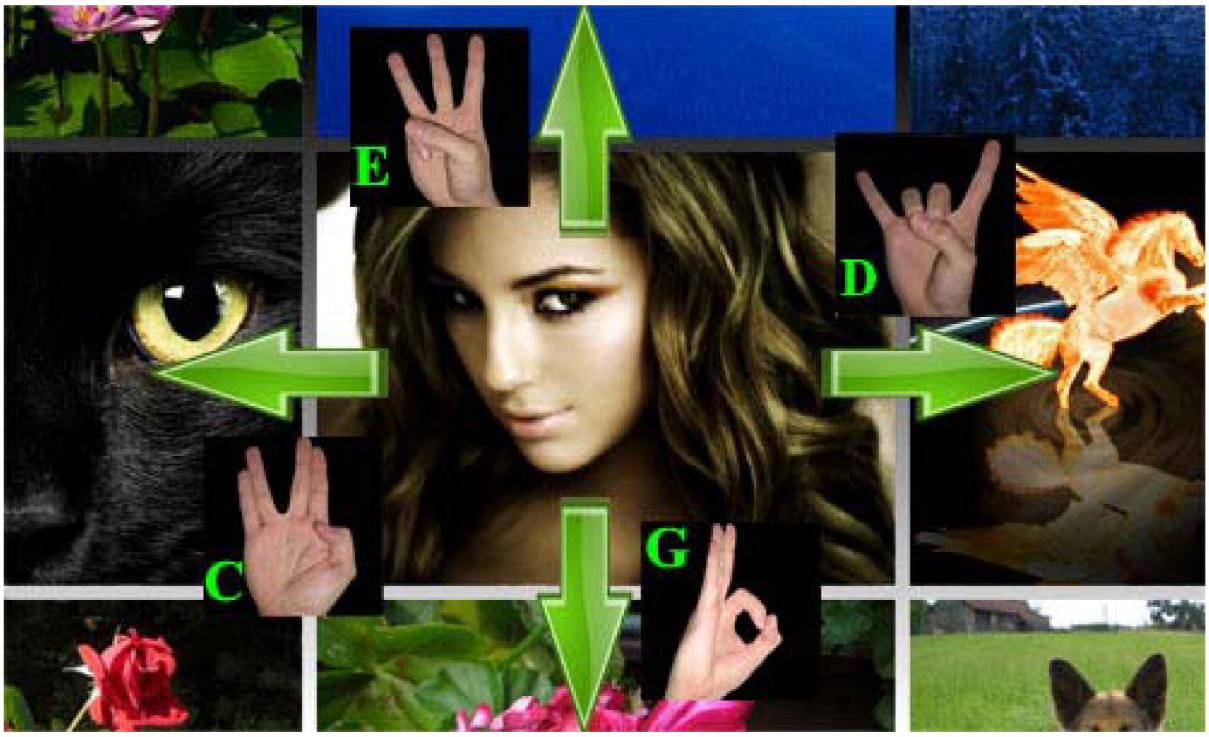
- Gesto C, posun náhledu o jednu pozici vlevo od středu
- Gesto D, posun náhledu o jednu pozici vpravo od středu
- Gesto E, posun náhledu o jednu pozici nahoru od středu
- Gesto G, posun náhledu o jednu pozici dolů od středu
- Gesto H, přepínání mezi módy zobrazení
- Gesto I, přepínání mezi přiblížením na střed či oddálení na všechny fotografie

Gesta C, D, E a G slouží uživateli pro posun náhledu o jednu pozici ve zvoleném směru od středu zobrazení. Posun je proveden pomocí animace a můžou nastat 2 následující možnosti:

- Náhled je přiblížen ke zvolené fotografii
- Náhled je na všechny fotografie

V první možnosti, kde je náhled na zvolenou fotografii a její nejbližší okolí, se za pomocí gest C, D, E a G provede posun od středové fotografie, na kterou byl proveden náhled. A u druhé možnosti,

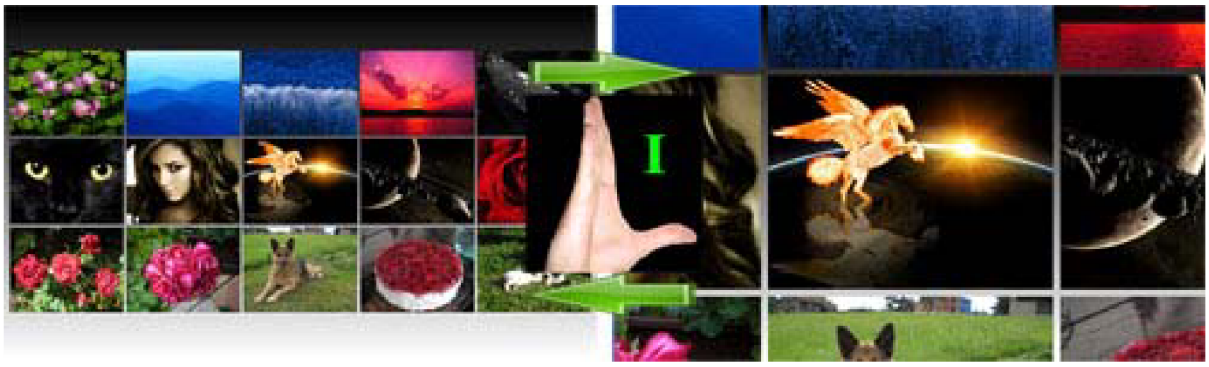
kde je náhled na všechny fotografie se za pomocí gest přiblížíte v daném směru podle gesta od středového obrázku, v našem případě ohnivého pegase, a přesune se pohled na vybranou fotografii. Pro lepší pochopení můžete na Obrázek 32 vidět nástin ovládání náhledu gesty C, D, E a G, jsou zde naznačeny šipky a u nich příslušná gesta, ukazující směr posunu náhledu.



Obrázek 32: Nástin ovládání změny pozice náhledu pomocí gest C, D, E a G.

Gesto H, přepíná mezi módy zobrazení stejným způsobem jako v módě 0 popsaném výše.

Gesto I v módě 2, přepíná mezi přiblížením náhledu na středovou fotografii (ohnivého pegase), nebo oddálení na náhled na všechny načtené fotografie ze složky. Pro lepší představu se podívejte na Obrázek 33, kde je zobrazeno jakým způsobem se přepíná náhled za pomocí gesta I.



Obrázek 33: Nástin přepínání mezi přiblížením náhledu na střed a oddálení na všechny fotografie gestem I.

Je možné vytvořit mnoho gest a mnoho způsobů ovládání celého počítače těmito gesty, ale pro jejich ukázku si myslím, že stačí výše zmíněné způsoby ovládání pomocí gest.

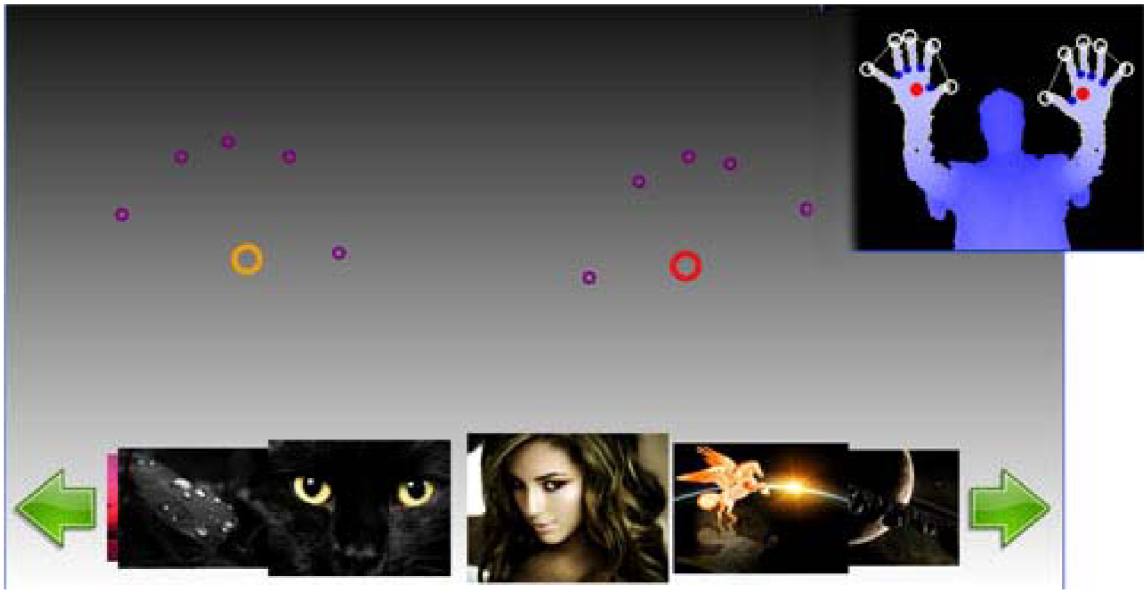
6.2 Ovládání aplikace pomocí rukou

Hlavním ovládacím prvkem ukázkové aplikace je umístění rukou v obraze a detekce, jestli jsou jedna nebo obě ruce uzavřeny v pěst či nikoliv. Pokud ano, začne se provádět příslušná operace podle toho, na jakém místě se ruka v té době nachází. Tímto způsobem se ovládá kurzor a simuluje se kliknutí na jednotlivá tlačítka myšky, stejně jako by to uživatel udělal použitím myšky nebo prstem na tabletu.

Ovládaní aplikace pomocí pozice rukou a prstů je preciznější, kreativnější a intuitivnější než ovládání aplikace gesty, i když na druhou stranu i detekce uzavření ruky v pěst je detekce gesta, kterou aplikace využívá k ovládání.

Aby bylo možné aplikaci ovládat pomocí rukou jednoduše a intuitivně, je potřeba uživateli zobrazit pozice rukou a prstů v aplikaci. Pro zobrazení pozic rukou a prstů používám různobarevné kruhy. Všechny kruhy označující pozici prstů mají fialovou barvu a stejnou velikost. Kruhy označující pozici dlaní jsou větší než velikost kruhů prstů a jsou barevně odlišné, podle pozice rukou. Pozice dlaně nejvíce vlevo má oranžovou barvu a pozice dlaně nejvíce vpravo má červenou barvu. Tímto způsobem lze odlišit pravou a levou ruku. Pozice dlaně nejvíce vpravo ovládá pozici kurzoru myši. Jestliže je detekovaná pouze jedna ruka, vyhodnotí se jako pozice dlaně nejvíce vpravo.

Jak to vypadá, vidíte na následujícím Obrázek **34**, kde je zobrazena aplikace spolu s pozicemi rukou a prstů. V pravém horním rohu vidíte uživatele, u kterého byli detekovány zdvižené dvě ruce s deseti prsty.



Obrázek 34: Ukázka zobrazení pozic rukou a prstů v aplikaci.

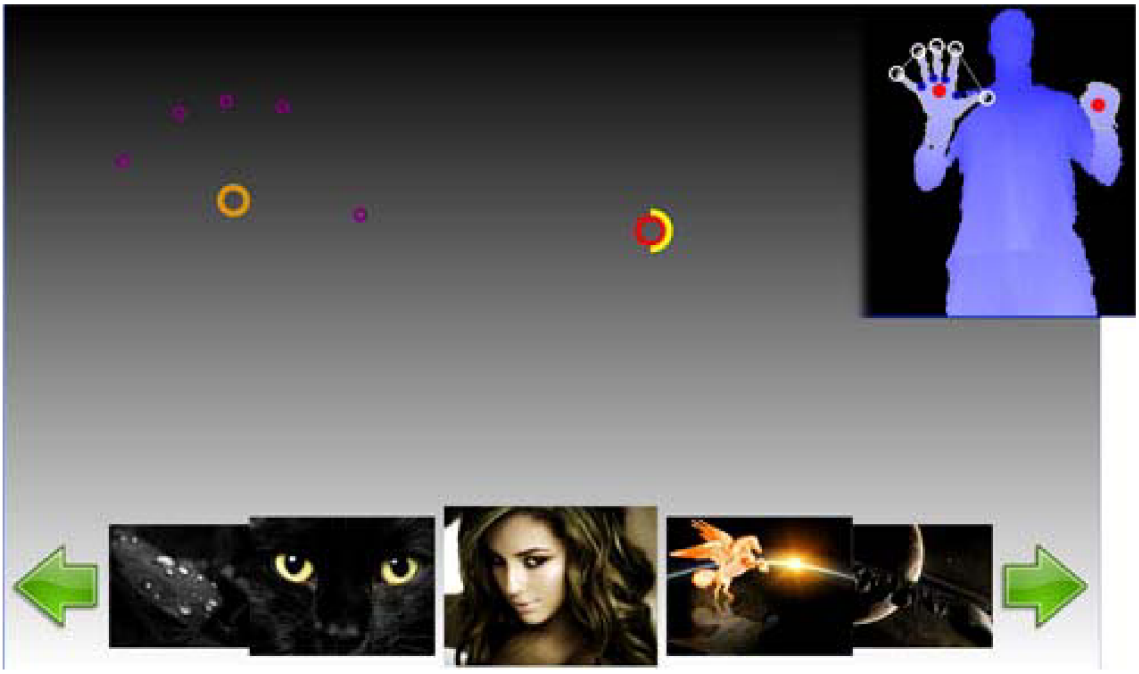
Když už máme nalezenou pozici dlaně pravé ruky, označenou červeným kruhem, která ovládá kurzor myši, potřebujeme detekovat, kdy se má provést simulace kliknutí tlačítka myši. To je provedeno tak, že když je detekováno, že uživatel má zavřenou pěst ruky v pravé části (nejsou vidět žádné prsty), provede se simulace kliknutí levého tlačítka myši.

Aby se zamezilo častým chybám, neprovede se tato simulace okamžitě, ale až po uplynutí doby přibližně jedné sekundy, po kterou je detekováno, že byla pravá ruka zavřena v pěst. To se zobrazuje uživateli ve formě žlutého kruhu, který se rozvijí okolo červeného kruhu označující pozici pravé dlaně. Tento žlutý kruh není hned zpočátku celý, ale v průběhu času, co je detekována zavřená pravá pěst, se rozvijí od nultého stupně začínajícího v přirovnání k hodinám na 12 hodině a během jedné sekundy se rozvijí po směru hodinových ručiček, dokud není celá.

Pokud je po dobu jedné sekundy detekováno, že je pravá pěst sevřena, tak červený kruh následně zezelená a provede se simulace kliknutí levého tlačítka myši, což zaznamená aplikace a provede příslušnou operaci spojenou s pozicí kurzoru myšky. Kruh je zelený a simuluje držení levého tlačítka myši, dokud je pravá ruka uživatele sevřena v pěst. Jakmile je detekováno, že uživatel zvedl více jak dva prsty (kvůli zamezení nechtěných chyb), simulace držení levého tlačítka myši skončí a tlačítko se uvolní, na což reaguje aplikace ukončením operace souvislé s předchozím kliknutím myši.

Ale pokud je v průběhu jedné sekundy od doby, co bylo poprvé detekováno, že uživatel sevřel pravou pěst, je zjištěno, že uživatel zdvihl více jak 2 prsty pravé ruky, žluté rozvíjející se kolečko zmizí a objeví se, až když je opětovně detekováno, že uživatel sevřel pravou pěst.

Na dalším Obrázek 35 vidíte ukázku, jak vypadá rozpoznání pravé ruky v pěst pro simulaci stisku levého tlačítka myši. Je zde zobrazena levá ruka s pěti prsty a pravá ruka zavřená v pěst, kterou vidíte jako červeny kruh kolem kterého se rozvijí žlutý kruh, ten signalizuje, kolik času zbývá, než se provede simulace stisku levého tlačítka myši. Je vidět, že žlutý kruh se rozvinul jenom do poloviny, uběhla proto jen polovina času. A v pravém horním rohu je vidět snímek z Kinectu, ukazující co v tu chvíli děla uživatel.



Obrázek 35: Ukázka rozpoznání pravé ruky v pěst, pro simulaci stisku levého tlačítka myši.

Ale ne pro vše se hodí simulace pouze kliknutí levého tlačítka myši, a tak využívám i pravého tlačítka. Detekce i průběh, že se má provést simulace stisku pravého tlačítka myši, probíhá totožně s předchozím příkladem, s kontrolou zdali ve chvíli, kde po uplynutí jedné sekundy, kdy uživatel držel pravou ruku v pěst, zjištěno jestli je i levá ruka sevřena v pěst (nejsou detekovány prsty na levé ruce). Pokud ano, provede se simulace stisku pravého tlačítka myši, v opačném případě se provede simulace stisku levého tlačítka myši, stejně jako když je detekována pouze jedna ruka.

Tímto způsobem lze tedy ovládat kurzor myši a její pravé a levé tlačítko.

6.2.1 Ovládání aplikace pomocí rukou v módu zobrazení 0 a 1

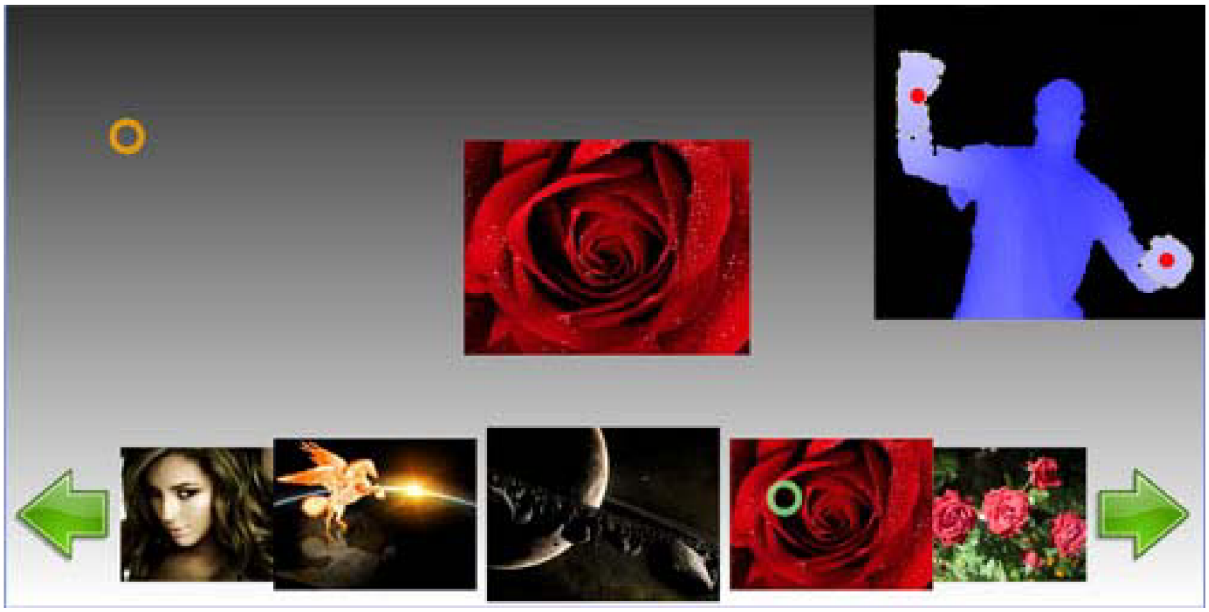
Dále stručně popíši, co a jakým způsobem lze ovládat aplikací pomoci rukou a detekce prstů.
Rukou lze v módě 0 ovládat následující funkce:

- Posuv fotografií v 3D liště
- Výběr fotografie z 3D lišty na pracovní plochu
- Kliknutí na tlačítka pro posun jedné fotografie po 3D liště
- Změna pozice fotografie na pracovní ploše
- Změna velikosti zobrazení vybrané fotografie
- Zrcadlové otočení fotografie ve vertikálním či horizontálním směru.
- Změna natočení fotografie
- Odebrání fotografie z pracovní plochy

Posuv fotografií v 3D liště provede uživatel tak, že najede rukou na zvolený obrázek na liště a sevře pěst, tím se posuv aktivuje a uživatel posuvem ruky doprava či doleva posouvá fotografie. Jakmile uživatel rozevře pěst, fotografie se animací srovnají na správnou pozici.

Pokud chce uživatel vybrat některou ze zobrazených fotografií v 3D liště na pracovní plochu, musí najet pravou rukou na vybranou fotografii, dát dlaň v pěst a zároveň musí mít i levou ruku uzavřenou v pěst (nezávisí to na pozici levé ruky, jen aby u ní nebyly detekovány prsty).

Na následujícím Obrázek **36** vidíte výběr fotografie s růži z 3D lišty na pracovní plochu. Zelené kolečko označuje pozice pravé ruky v pěst a oranžové kolečko označuje levou ruku v pěst. V pravém horním rohu je snímek uživatele z Kinectu při této operaci. Na pracovní ploše je zobrazená vybraná fotografie.

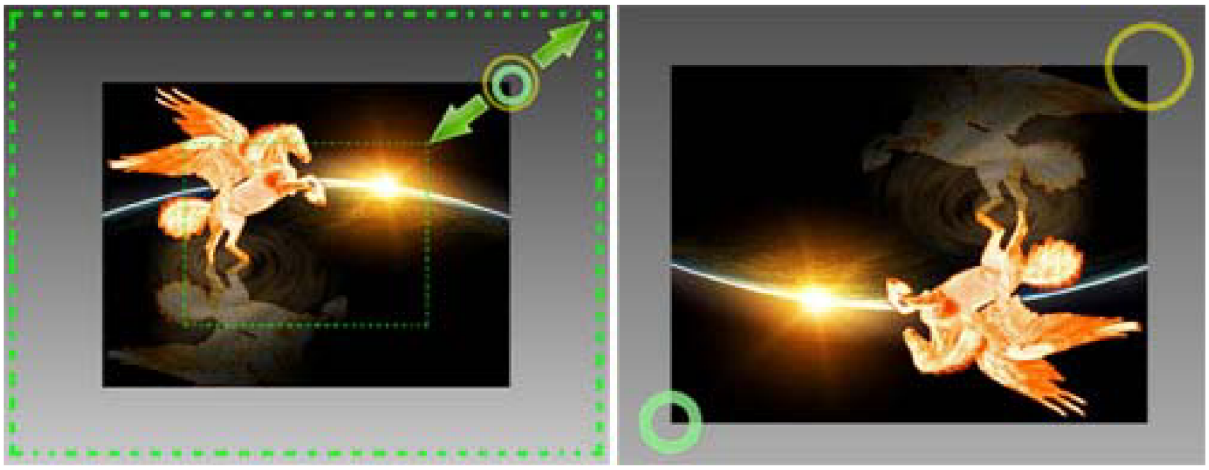


Obrázek 36: Ukázka výběru fotografie z 3D listy na pracovní plochu. Zelné kolečko označuje pozice pravé ruky v pěst a oranžové označuje levou ruku v pěst. V pravém horním rohu je uživatel při této operaci.

Když už se nachází nějaká fotografie na pracovní ploše, lze měnit její pozici. A to tak že na ní uživatel najede pravou rukou, kterou sevře v pěst a vyčká tak sekundu, než červený kruh zezelená. Poté se střed fotografie přesune na pozici ruky. Kam se pohne ruka, tam se pohne i fotografie, dokud uživatel neotevře pěst (tedy bude detekováno více jak 2 prsty).

Změnu velikosti vybraného obrázku uživatel provede tak, že přesune pravou dlaň do pravého horního rohu vybrané fotografie, kde se poté objeví žlutý kruh označující oblast, kde lze stisknout ruku v pěst pro ovládání změny velikostí. Když uživatel v této oblasti stiskne pravou ruku v pěst, aktivuje se změna velikosti vybraného obrázku a od té doby podle směru pohybu ruky uživatel, buď zmenšuje, nebo zvětšuje vybranou fotografii, dokud uživatel pěst nerozevře. Změna velikosti se provádí od středu fotografie. Když uživatel přejede přes střed fotografie rukou, zrcadlově se otočí buď v horizontálním nebo vertikálním směru, nebo se zcela zrcadlově otočí.

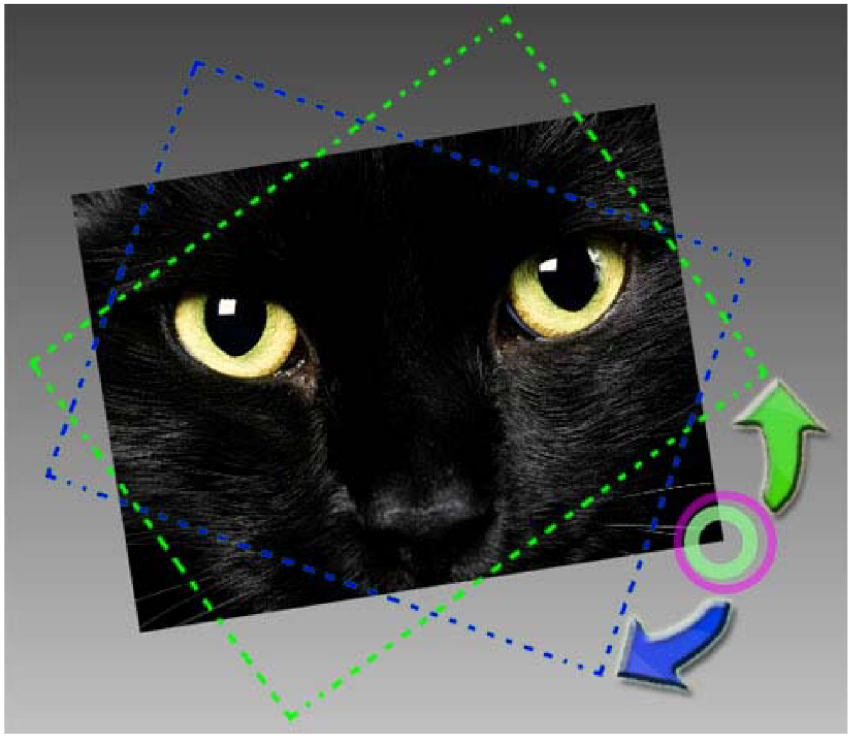
Na Obrázek 37 je znázorněna ukázka změny velikosti (vlevo) a zrcadlení (vpravo) vybrané fotografie ovládané pomocí ruky. U změny velikosti je žlutým kolečkem naznačeno okolí, kde je možné kliknout pro ovládání, dále je zde zelené kolečko označující pozici pravé ruky zavřené v pěst a šipkami je naznačen směr pohybu ruky a zelenými čerchovanými obdélníky, je naznačeno zvetšení nebo zmenšení. Šipky ani obdélníky se v aplikaci nezobrazují, jsou zde pouze pro ukázku, v aplikaci se velikost změn okamžitě s pohybem ruky.



Obrázek 37: Ukázka změny velikosti (vlevo) a zrcadlení (vpravo) ovládané jednou rukou. Zelenými šipkami s obdélníky je naznačeno zvětšení a zmenšeni fotografie.

Otočení zvolené fotografie uživatel provede podobě jako změnu velikosti popsanou výše s tím rozdílem, že musí najet na pravý dolní roh fotografie, kde se mu zobrazí fialový kruh. Po sevření pěsti pravé ruky ve fialovém kruhu začne uživatel ovládat natočení fotografie, kolem jejího středu, v rozsahu 360°. Když je uživatel spokojený s natočením zvolené fotografie, pěst rozevře a může provádět další operace.

Ukázku natočení fotografie pomocí ruky, můžete vidět na Obrázek 38, kde vidíte výše zmiňované fialové kolečko a zelené také již zmiňované kolečko, ale také zelenou a modrou šipku s čárkovaným obdélníkem, které naznačují, jak se fotografie natočí při pohybu ruky ve směru šipky.



Obrázek 38: Ukázka natočení fotografie pomoci ruky. Zelenou a modrou šipkou s obdélníky je naznačeno, jak se fotografie natočí při pohybu ruky.

Odebrání fotografie z pracovní plochy se provede podobným způsobem, jako výběr fotografie z 3D lišty na pracovní plochu. S tím rozdílem, že uživatel najede pravou rukou na zvolený obrázek, který chce odstranit, a sevře obě ruce v pěst.

Stejné ovládaní jako má mód 0 popsaný výše má také mód 1, s tím rozdílem, že nelze ovládat 3D lišta ani tlačítka, protože v tomto módě nejsou zobrazeny. Ale vzhledem k tomu, že v módě 2 je jiné zobrazení fotografií, je i jiné ovládání.

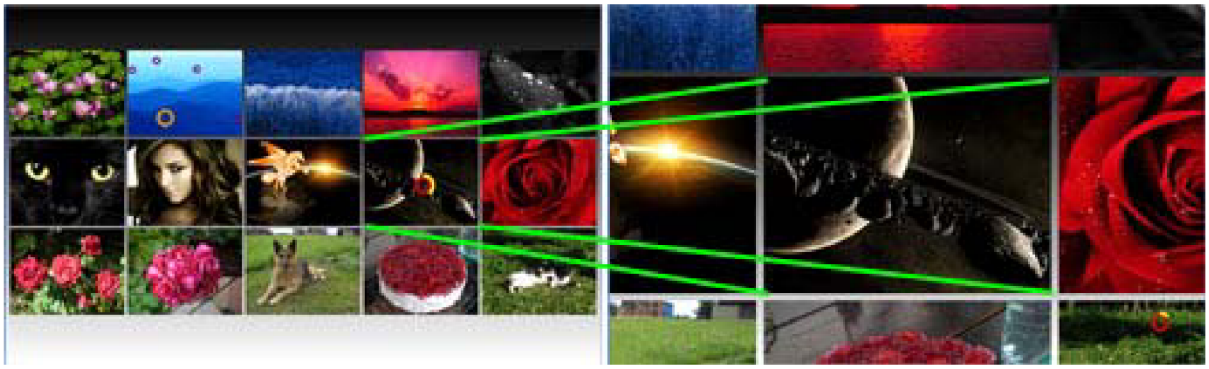
6.2.2 Ovládání aplikace pomocí rukou v módě zobrazení 2

V módě 2 lze rukou ovládat tyto funkce:

- Přiblížení náhledu na vybranou fotografii
- Výběr možných 8 směrů posunu náhledu od středové fotografie
- Vrácení náhledu na všechny fotografie

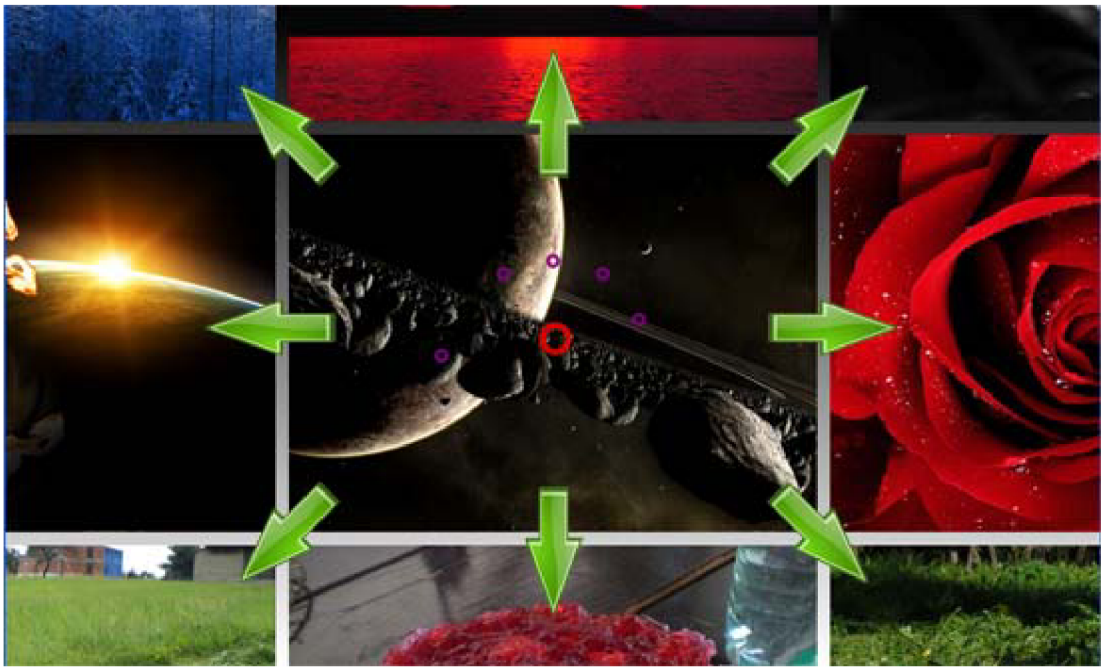
Uživatel si může přiblížit jakoukoliv ze zobrazených fotografií najetím nad ní pravou rukou a jejím sevřením v pěst. Poté se náhled animací přesune k zvolené fotografii, která je uprostřed a obklopuje ji 8 fotografii z okolí (pokud se tam nachází). Toto je stejné jako při použití gesta J v módě 2, ale ovládání rukou dává větší možnosti ve výběru a rychlosti, jakou je tento výběr proveden.

Na Obrázek 39 je naznačena ukázka přiblížení náhledu na fotografii planety s meteory nacházející se vpravo od středové fotografie s ohnivým pegasem. Na levé části obrázku vidíte pozici levé ruky a jejich pět zdvižených prstů a pravá ruka se nachází nad zvolenou fotografií a má ruku v pěst (na obrázku zachycena skoro v půlce kliknutí). Tímto způsobem se provede přiblížení náhledu, který vidíte na obrázku vpravo.



Obrázek 39: Ukázka přiblížení náhledu na zvolenou fotografii pomocí pozice ruky.

Když uživatel najel na bližší náhled vybrané fotografie, může si ji prohlédnout a poté se vydat na prohlížení 8 okolních fotografií, nebo kliknutím na středovou fotografii se oddálit na náhled všech fotografii. Účinek je stejný jako provedení gest C, D, E, G a J, ale může uživatel přejít a na rohové fotografie a navíc je ovládání rukou více intuitivní. Ale nic nebrání kombinovat ovládání gesty a rukou.



Obrázek 40: Ukázka možných směrů, kam se může uživatel náhledem podívat. Šipky naznačují směr a po kliknutí na některou z fotografií, se na ní náhled přesune. Po kliknutí na středovou fotografii se náhled oddálí pro zobrazení všech fotografií.

6.3 Ovládání klávesnicí a myší

Aplikace lze taktéž ovládat klasickým způsobem pomocí klávesnice a myši. Myší lze ovládat všechny funkce popsané výše, stejně jako ovládání rukou, protože pravá ruka ovládá kurzor myši a sevření pravé ruky v pěst se simuluje stisk levého tlačítka myši. A sevření levé a pravé ruky v pěst vyvolá simulaci stisku pravého tlačítka myši.

Klávesnicí lze ovládat všechny funkce ve všech módech stejně jako gesty, protože při rozpoznání gesta je vyslán signál pro simulování stisku kláves na klávesnici, na které aplikace reaguje výše zmíněným způsobem.

Popis kláves, které ovládají gesta:

- Gesto C, simuluje stisk šipky doleva
- Gesto D, simuluje stisk šipky doprava
- Gesto E, simuluje stisk šipky nahoru
- Gesto G, simuluje stisk šipky dolů
- Gesto H, simuluje stisk klávesy M
- Gesto J, simuluje stisk klávesy Space (mezerník)

Lze tedy říct, že až na drobné nepřesnosti lze ovládat aplikaci rukou a gesty stejně jako myší a klávesnicí.

7 Realizace rozhraní ovládané gesty

V této kapitole se dozvíte jaké nástroje a knihovny jsem použil pro vytvoření detekce rukou a gest, rozpoznávání gest a výsledné aplikace ovládané gesty a pohybem ruky. Dále jak jsem tyto části reálně řešil a vytvářel. A nakonec popis jakým způsobem jsem propojil aplikaci s ovládáním gesty.

Celou aplikaci a ovládání gesty jsem vyvíjel na platformu Microsoft Windows XP, ale měla by být spustitelná i na Windows 7. Programy jsem vytvářel a překládal na vývojářském nástroji Microsoft Visual Studio 2010.

Použité programovací jazyky:

- C/C++ pro detekci a rozpoznání gest a rukou
- C# pro práci s aplikaci ovládanou gesty
- WPF pro vytvoření vzhledu aplikace

Použité knihovny kromě standardních:

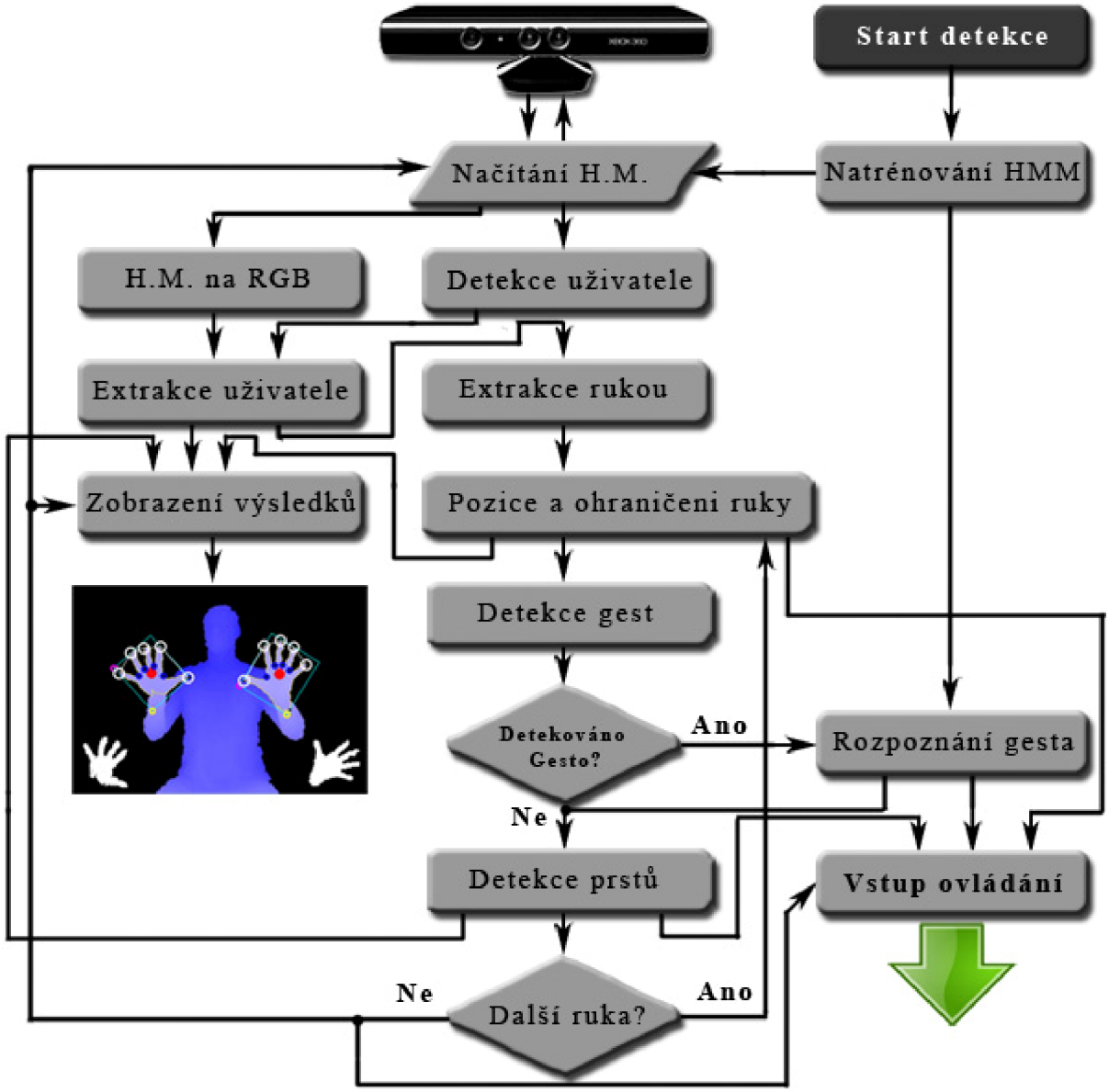
- OpenNI pro práci se senzorem Kinect a detekci uživatele
- OpenCV v2.3.1 pro práci s obrazem a s HMM
- .Net pro práci s aplikací a fotografiemi

7.1 Realizace detekce rukou a gest

V této podkapitole popíši, jak jsem implementoval detekci rukou a gest a nástin rozpoznávání gest, který bude podrobněji popsán v následující podkapitole. Text se bude týkat především rozdílu oproti návrhu.

Detekce rukou a gest slouží pro nalezení pozic rukou a prstů a pro přípravu gesta pro rozpoznání, aby se předaly jako informace pro ovládání zmíněného prohlížeče fotografie. Problém je však v tom, že detekce rukou a gest jsem napsal spolu s rozpoznáváním gest v jazyce C/C++ a prohlížeč fotografií ovládaný gesty je v jazyce C#. Aby bylo možně tyto dvě části spojit, vytvořil jsem proto z detekce rukou a gest - .dll soubor se dvěmi externími funkcemi, který je přidán do prohlížeče fotografií. První z funkcí volá prohlížeč fotografií a tím spustí detektor rukou a gest. A druhá funkce posílá informace o pozici rukou, prstů a o rozpoznaném gestu, pokud bylo detekováno do prohlížeče fotografii, aby jej bylo možné ovládat rukou. Detekce rukou a gest je mimo jiné schopna zpracovávat vstupní signál z Kinectu nebo videozáznamu pro testování nebo ukázku činnosti.

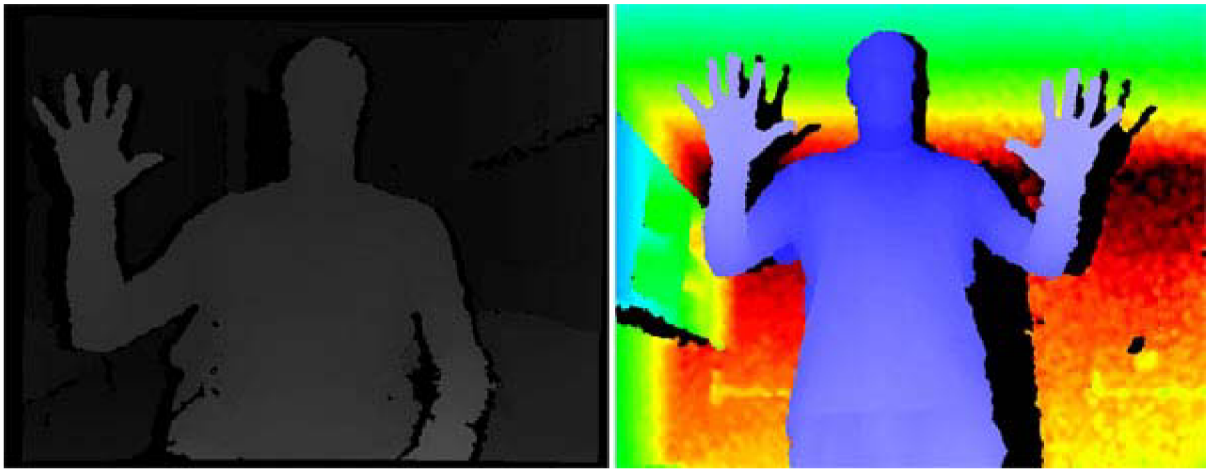
Na Obrázek 41 je znázorněno schéma detekce rukou a gest od startu detekce přes hlavní smyčku až po výstup pro ovládání gesty. Schéma popíši níže a zaměřím se na části, které jsou odlišné nebo rozšiřující oproti návrhu.



Obrázek 41: Schéma hlavní smyčky detekce rukou a gest.

Schéma hlavní smyčky detekce rukou a gest znázorněna na předchozím obrázku popisuje, jaké procesy jsou v rámci algoritmu použity. Po startu detekce vyvolaného z prohlížeče fotografie, se spustí natrénování HMM z dat gest, aby bylo možné jednotlivé gesta později rozpoznat. Dále je nutné inicializovat spojení s Kinectem pro načítání hloubkové mapy za pomoci knihovny OpenNI **[10]**. To je provedeno tak, že je načten XML soubor s parametry zařízení, poté je vytvořen generátor hloubky, který přenáší hloubkovou mapu pro pozdější zpracování. Také je vytvořen generátor uživatelů, který z hloubkové mapy detekuje uživatele ve formě masky s hodnotami uživatelského ID, kde se uživatel v obraze nachází. Toto provedení bylo převzato z příkladů a dokumentace pro OpenNI.

Nyní necháme ve smyčce načítat hloubkovou mapu a k ní masku s umístěním uživatelů. Avšak nastává problém jak tyto informace zobrazit, protože hloubková mapa z Kinectu, obsahuje jedenácti bitovou informaci vzdálenosti od zařízení přibližně v milimetrech. Zobrazení není důležité pouze, aby uživatel viděl, jak pracuje detekce, ale především pro vytváření videozáznamu pro testování a ukázku funkčnosti. Pro zobrazení se používá buď šedotónový osmi bitový a nebo čtyřiadvaceti bitový RGB obraz. Pokud se tedy použije osmi bitový obraz, ztratí se pěti bitová informace o hloubce. Pro běžné zobrazení je to postačující informace, ale pro vyhodnocování a zpracování obrazu na větší vzdálenosti nebo z videozáznamu nikoliv. Proto používám převod hloubkové mapy na RGB zobrazení.



Obrázek 42: Hloubková mapa převedena na osmi bitový šedotónový (vlevo) obraz a čtyřiadvaceti bitový RGB (vpravo) obraz.

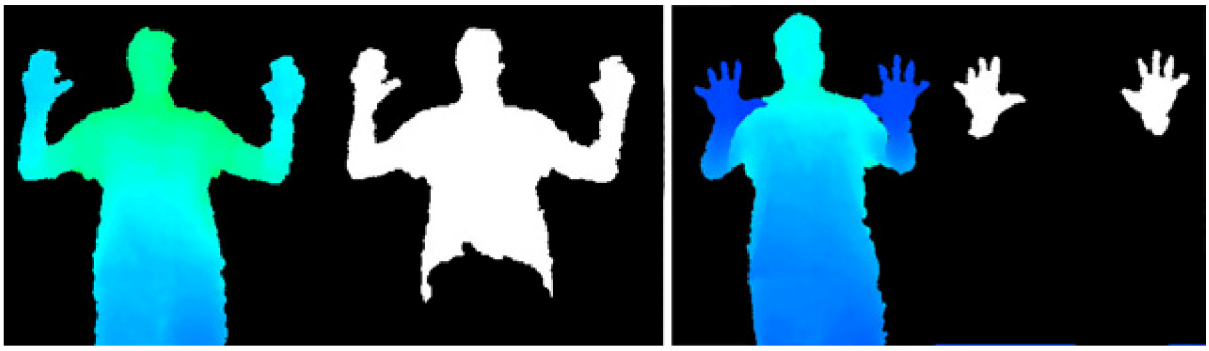
Na tento obraz dále použijeme masku uživatele s generátorem uživatelů, abychom odfiltrovali postavu od pozadí a mohly dále pracovat a vyhodnocovat užitečná data. To provádím tak, že před vyhodnocováním barvy pro daný bod je vyhodnocena informace z generátoru uživatelů, zdali je na daném bodě uživatel (poté je vyhodnocena hloubka) nebo pozadí (kde je vrácena černá barva).

Ale aby byla později provedena detekce a extrakce rukou, je nutné barevné hodnoty převést na hodnotu jasu a v celém obrazu, kde je již jen uživatel, nalézt nejvyšší hodnotu. Což je hodnota části uživatele, která je nejblíže k zařízení, s tím že předpokládáme, že nejblíže jsou ruce uživatele.

Toto je však vhodné pouze pro zobrazení nebo načítání z videozáznamu, protože hodnoty jasu se mění při přechodu do jiného pásma vzdálenosti od Kinectu a také je rozsah jasu omezen. A tak je v rozdílných místech vyhodnocována detekce rukou odlišně. Proto se do pomocné šestnácti bitové matice ukládají přesné hodnoty hloubky (pouze v místech kde je uživatel), kde se poté hledá nejbližší vzdálenost (v cca milimetrech) části těla od zařízení. Proto používám extrakci rukou rovnou z hloubky, pokud načítám data z Kinektu nebo z barvy, pokud z video záznamu

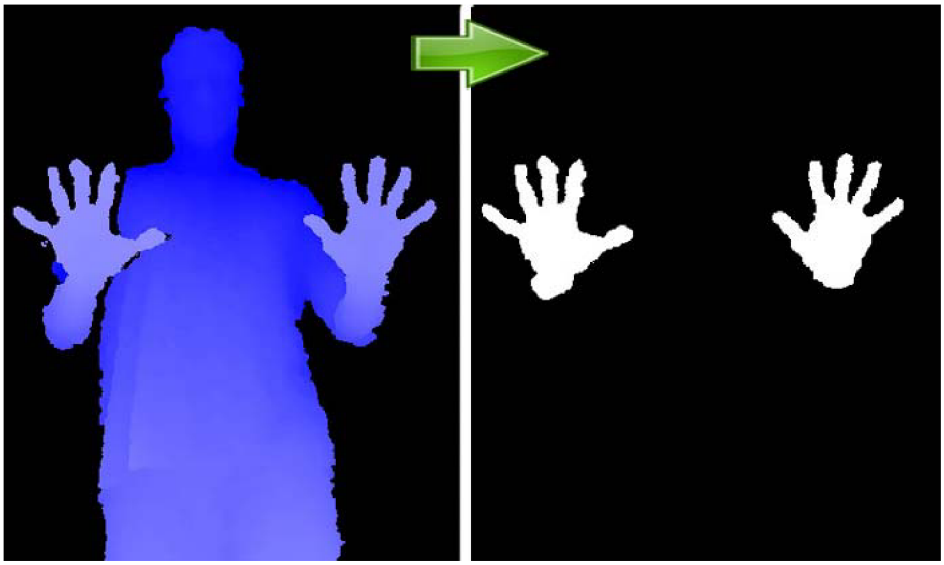
Rozdíl mezi hledáním rukou pomocí barvy (vlevo) a za pomoci hloubky (vpravo), když je uživatel na pomezí barových pásem, kde si můžete povšimnout, že při extrakci rukou z barvy

(vlevo) byla kromě rukou detekována i většina uživatele. Navíc byla přidána kontrola, aby extrakce rukou proběhala, pouze pokud jsou ruce v dostatečné vzdálenosti od těla.



Obrázek 43: Rozdíly extrakce rukou z barvy (vlevo) a z hloubky (vpravo), při větší změně vzdáleností od Kinectu.

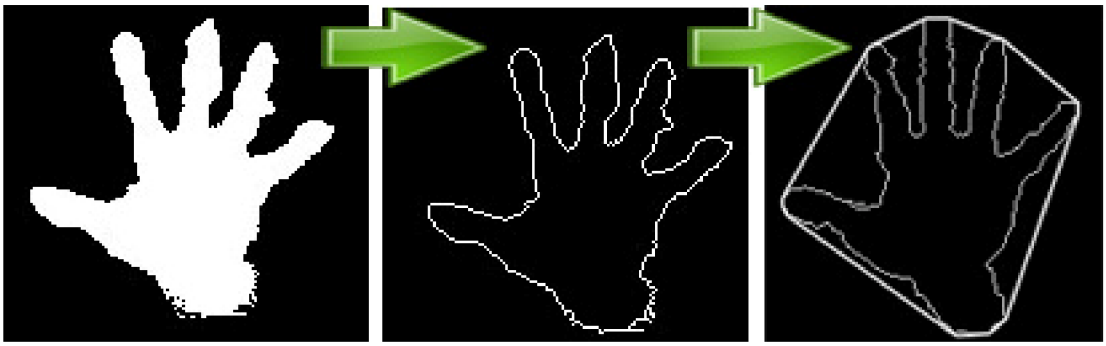
V případě filtrace z hodnot hloubky je od nalezené nejbližší vzdálenosti od Kinectu přičteno 120, tedy přibližně dvanáct centimetrů. A nakonec jsou odfiltrované všechny vzdálenější hodnoty. Takže vznikne nový obraz tvořící masku, kde jsou bíle označeny místa, která jsou nejblíže k zařízení s určitým prahem. Jak je vidět na Obrázek **44**, kde je vlevo zobrazeno převedení hloubkové mapy na barvu s extrakcí uživatele a vpravo odfiltrované ruce.



Obrázek 44: Extrakce rukou jako základ pro zpracování informací rukou.

Tímto detekce a parametrizace ruky nekončí, ale začíná. Protože i tak je zde ještě příliš přebytečných informací o tvaru a pozici ruky. Proto na tento obraz použijeme funkci OpenCV, která nalezne kontury, tedy obrysy objektu z obrazu **[16]**. A pro jednotlivé kontury, jestliže splňují určité vlastnosti velikosti (nejsou moc velké, ani malé pro velikost ruky), je provedeno další zpracování pro nalezení pozic rukou a prstů a detekce gest.

Na Obrázek 45, je zobrazena ukázka převedení extrahované ruky na konturu a následně přikresleni ke kontuře konvexní obálku, která propojuje nejkrajnější body kontury.

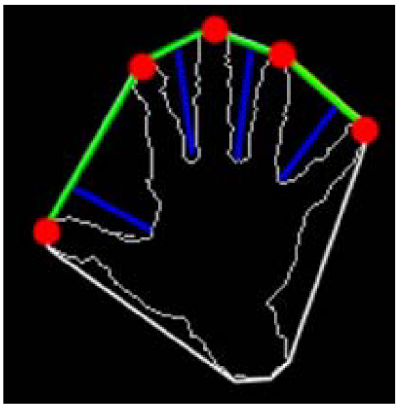


Obrázek 45: Ukázka převedení extrakce ruky (vlevo) na konturu (uprostřed) a poté přidání konvexní obálky pro detekci prstů (vpravo).

Poté, co byli nalezeny kontury, je nutné nalézt ohraničení a umístění každé ruky. To je získáno tak, že se na jednotlivé kontury aplikuje funkce OpenCV pro nalezení nejmenšího ohraničení za pomoci natočeného obdélníku. Tento obdélník se poté přikreslí do finálního obrazu spolu s červeným kruhem umístěným ve středu obdélníku, který označuje pozici ruky v obraze, která se používám pro ovládání kurzoru myši.

Z informací o ohraničení ruky se provádí detekce gesta tak, že je detekováno gesto, pokud v posledních 25 snímcích nezměnil ohraničující obdélník razantně střed a velikost. Jinak nastaví nový střed a velikost a pozoruje dále. Když bylo detekováno gesto, provede se jeho správné natočení, doplněni do čtverce a normalizuje se jeho velikost na 60x60, aby se poslalo na rozpoznávání gest, které vyhodnotí, o jaké gesto se s největší pravděpodobnosti jedná.

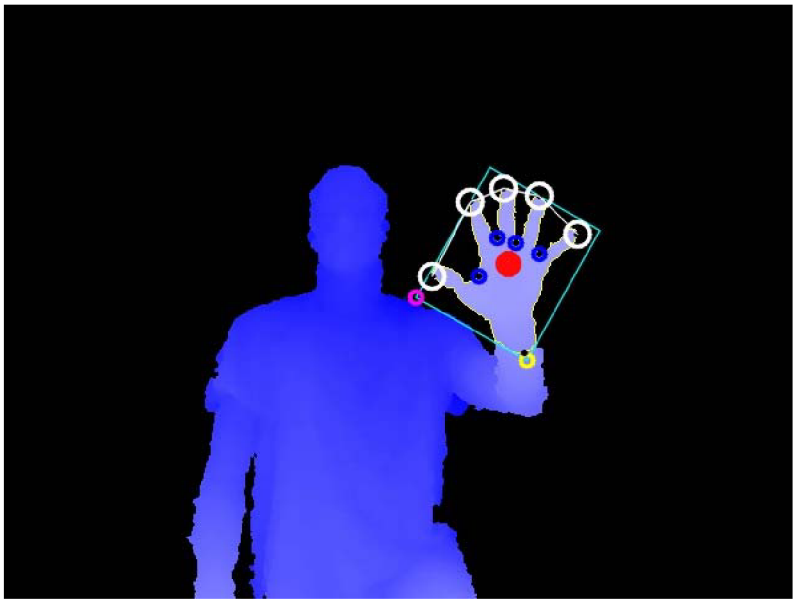
Pro nalezení prstů se konvexní obálka nevyhledává pro celou konturu, ale pouze pro body vzniklé aproximací křivky kontury zmíněné výše. To spolu s použitím speciální funkce v OpenCV pro nalezení nejvzdálenějšího bodu kontury od části konvexní obálky pro danou část kontury. Tímto způsobem získáme body, kde se nacházejí prsty, pokud splňují podmínky, že jsou v dostatečné vzdálenosti od sebe a hodnota vzdálenosti od nejvzdálenějšího bodu v kontuře od konvexní obálky je větší než přibližná polovina vzdálenosti mezi dvěma prsty. Na Obrázek 46 vidíte ukázku detekce prstů, kde červené body označují pozice prstů, zelené čary na konvexní obálce mezi prsty a modré čáry symbolizuje nalezenou nevětší vzdálenost mezi konturou a konvexní obálkou pro daný úsek.



Obrázek 46: Ukázka detekce prstů.

Pozice prstů je přenesena do výsledného obrazu, pro zobrazení a pokud byly vyhodnoceny všechny kontury, tak se zobrazí výsledky detekce rukou a prstů na obrazovku, jak můžete vidět na Obrázek 47. Kromě zobrazení se vyšle prohlížeči fotografií externí funkce se třemi poli s hodnotami pozice X a Y rukou a prstů a jedním polem, které rozlišuje, zdali se jedná o ruku nebo prst. A jednou hodnotou signalizující, jaké gesto bylo rozpoznáno.

Nakonec je zavoláno načtení další hloubkové mapy z Kinectu nebo videozáznamu a celá detekce rukou a gest se opakuje.



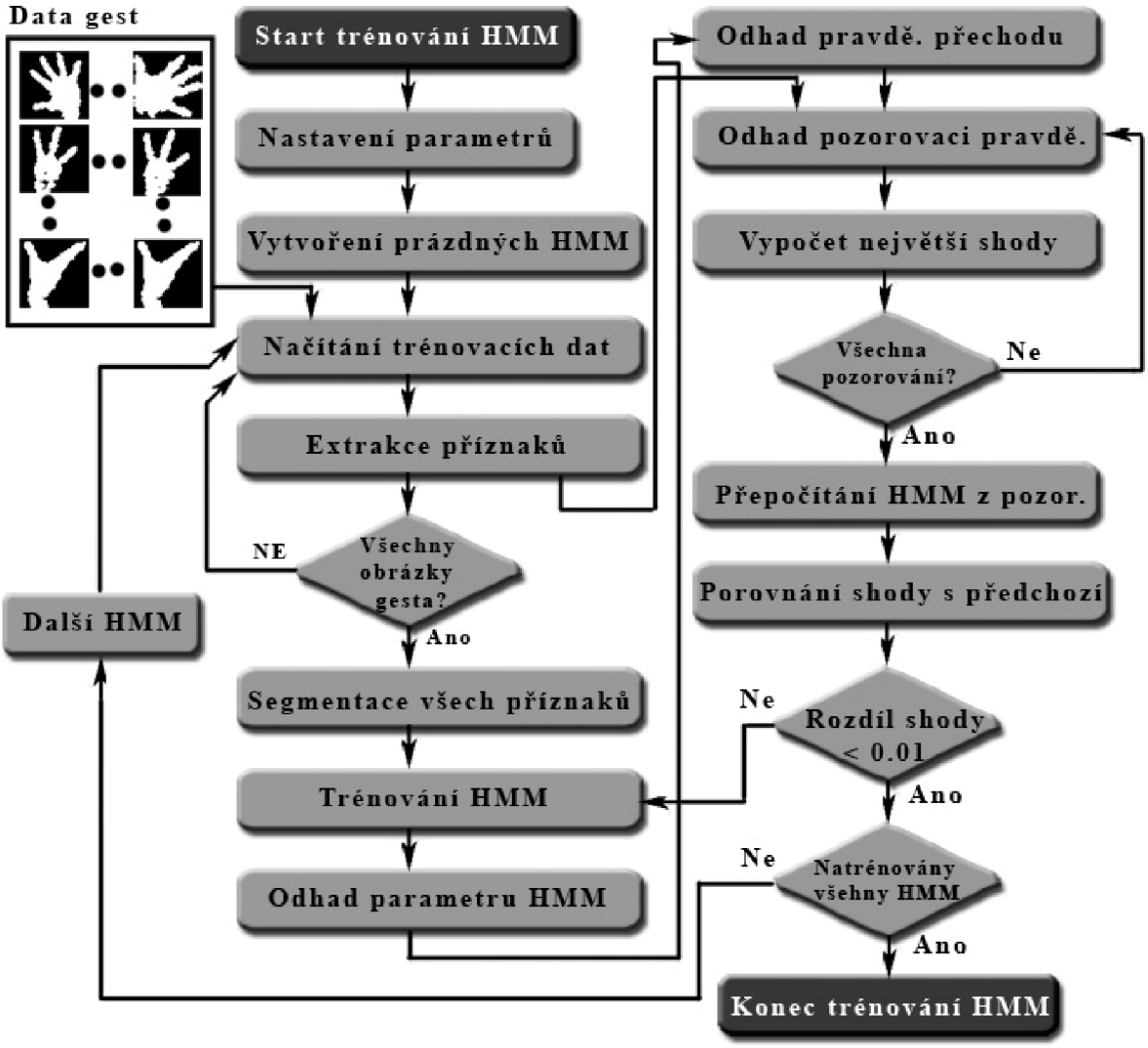
Obrázek 47: Ukázka zobrazení výsledků detekce rukou a prstů.

7.2 Realizace rozpoznávání gest

Tato podkapitola popisuje realizaci rozpoznávače gest od natrénování HMM a až po samotné rozpoznávání. Každé gesto se trénuje ze 40 obrázků tohoto gesta a ze 4x10 z různých natočení nebo i s části předloktí ukázka těchto trénovaných gest můžete vidět v kapitole 8. Natrénování HMM i rozpoznávání gest jsem psal v jazyce C/C++ s pomocí OpenCV, kde používám přímo

implementovaných funkcí pro práci s 2D HMM. U trénování i rozpoznávání gest jsem se inspiroval ze své bakalářské práce **[15]**, kde je ale používám odlišným způsobem.

Aby bylo možné rozpoznávat, o jaké gesto se s největší pravděpodobností jedná, musí být gesta natrénována do struktury 2D HMM. Jak to funguje, můžete vidět na Obrázek 48, kde je znázorněné schéma natrénování HMM. Natrénování HMM trvá dlouhou dobu, přibližně 25 sekund po kterou se musí počkat, než je možné rozeznávat gesta.



Obrázek 48: Schéma natrénování HMM pro pozdější rozpoznávání.

Schéma natrénování HMM znázorněna na předchozím obrázku popisuje, jaké procesy jsou v rámci algoritmu použity. Po startu trénování jsou nastaveny potřebné parametry a je vytvořeno 32 prázdných 2D HMM (4 HMM pro každé gesto). Poté se začne hlavní smyčka, která začíná převedením obrázku s gesty (60x60) na pozorovací strukturu. A to zavoláním funkce, která informace ze snímku převede na hodnoty spektra v blocích 12x12 pixelů pomocí DCT - popsané v podkapitole 2.2 - se vy-počítá 9 koeficientů, které se uloží do pozorovací struktury jak pozorovací vektor. Poté se celý blok poposune o 4 pixely doprava a znovu se celý blok přepočítá. To se provádí pro celý

řádek a poté se posune blok o 4 pixely dolů, dokud se neprojde celý snímek, ze kterého získám 13x13x9 (horizontální, vertikální posuny a každý blok má devět vypočítaných koeficientů) – tedy 1521 hodnot reprezentujících daný snímek gesta. Pro pozorovací strukturu se provede jednotná segmentace, která provede rozčlenění snímku pro práci s HMM.

Jakmile jsou všechny snímky pro dané 2D HMM převedeny na pozorováci struktury začne se s trénováním, kde za pomoci specializovaných funkci vytvořím odhad parametrů a přechodových pravděpodobností, které se v každé iteraci zpřesňuje. Poté se cyklem zavolá pro všechny pozorované obrázky funkce, která spočítá Gaussinovu pravděpodobnost pro každé pozorování, které se vyskytuje v každém z vnitřních HMM stavů. Dále se provádí Viterbiho algoritmus **2.1.3** pro vestavěné HMM, který vyhodnocuje pravděpodobnost nejlepší shody mezi pozorovaným obrázkem a daným HMM a provádí segmentaci obrazu pozorování HMM stavy. Tato segmentace se provádí na základě nalezené shody. Viterbiho algoritmus vrací hodnotu pravděpodobnosti, na kolik procent se daný snímek shoduje se sledovaným HMM. Tyto pravděpodobnosti se sčítají pro všechny snímky, které byly převedeny do trénovaného HMM, vyděleny počtem snímků, vynásobeny délkou každého pozorovacího vektoru - v našem případě devíti - a porovnány s předchozí pravděpodobností.

Pokud je tento rozdíl menší než jedna setina je HMM natrénováno a začíná trénování dalšího HMM, dokud nejsou natrénovány všechny HMM. Tím se ukončí cele trénování.

Obrázek 49: Schéma rozpoznání gesta.

Schéma rozpoznání gesta znázorněna na předchozím obrázku popisuje, jaké procesy jsou v rámci algoritmu použity. Vstupem do rozpoznávače gest je upravené gesto z detekce gest. Toto gesto se převede na pozorovací strukturu, jako tomu bylo u trénování. Poté je porovnána s každým HMM, kde je nalezena pravděpodobnost shody s tímto gestem. A nakonec je nalezeno, jaké HMM se s největší pravděpodobností shoduje s tímto gestem. Výstupem je poté číslo HMM, z kterého se odvodí o jaké gesto šlo.

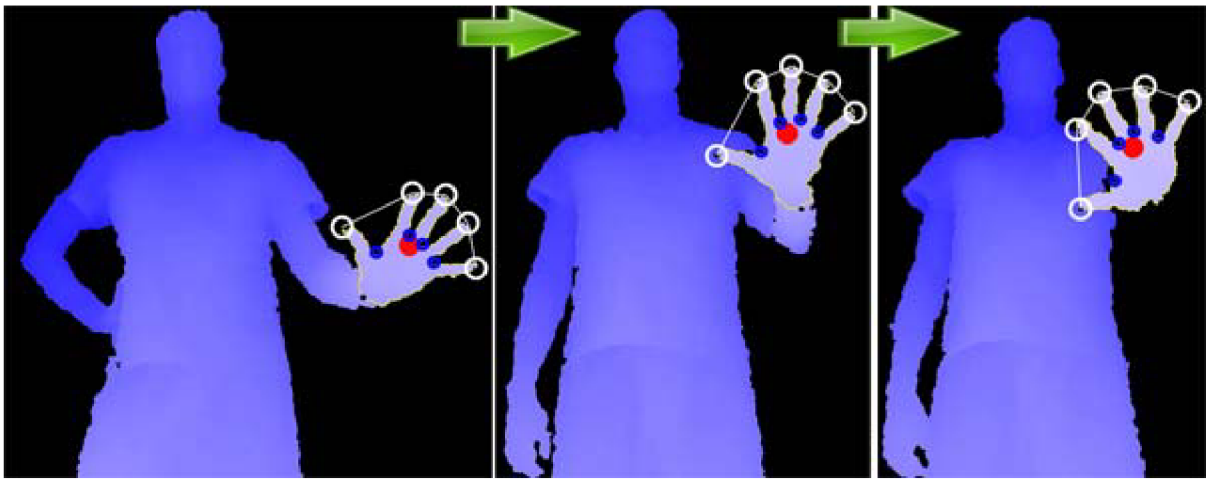
7.3 Realizace ovládání gesty

Ovládání gesty funguje jako most mezi detektorem rukou a gest a ovládáním mého prohlížeče fotografií. Ovládání gesty vezme pozice rukou a prstů, které zobrazí do aplikace, aby uživatel věděl, kde má ruce a mohl tak ovládat prohlížeč intuitivně. Z pozic rukou se zjistí, která ruka je nejvíce v pravé části, aby mohla ovládat kurzor myší, a z počtu prstů je odvozeno, jestli je ruka v pěst, pro simulaci stisku tlačítka myší. Pokud je zjištěno, že bylo rozpoznáno gesto, provede se simulace stisku klávesy příslušnicí tomuto gestu. Prohlížeč fotografii reaguje na myšku a stisk klávesnice. A tak provede příslušnou operaci Např.: Posun fotografii na 3D liště, manipulace s fotografiemi atd.

Obrázek 50: Schéma ovládání gesty.

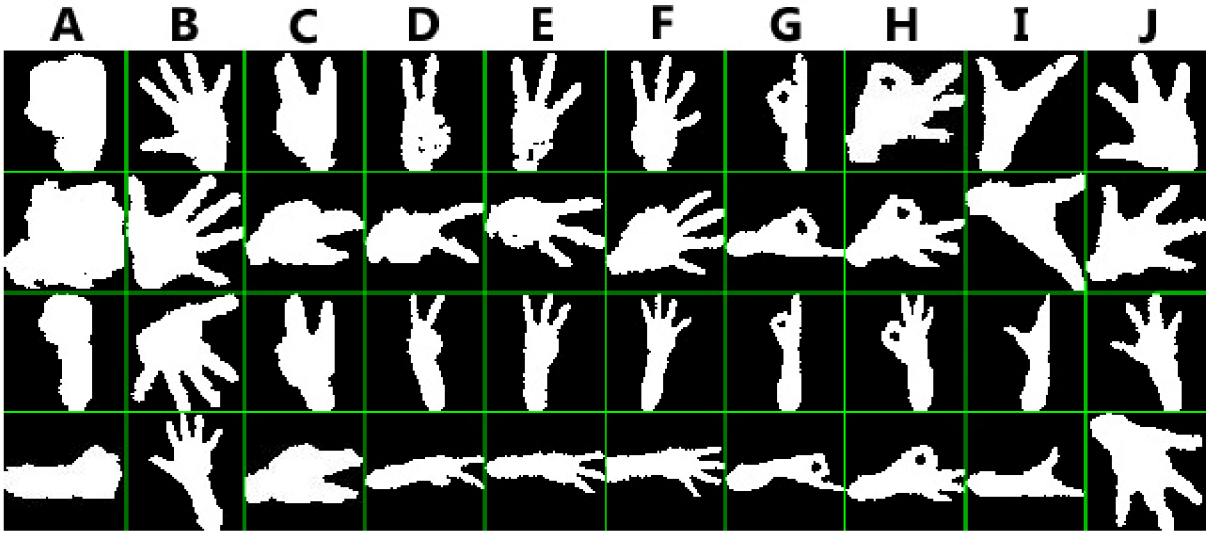
8 Testování úspěšnosti rozpoznání gest

Tato kapitola je zaměřena na testování rozpoznávání gest. A to především na vytváření testovacích dat a způsobu jejich vyhodnocování. Pro testování a vyhodnocování správnosti rozpoznání gest, jsem vytvořil deset videí, jedno pro každou třídu gest. Na každém videu provádím gesto pro něj určené v různých úhlech a vzdálenostech od senzoru Kinect, aby bylo zjištěno jak kvalitně funguje detektor gest a při jakých okolnostech jsou gesta vyhodnoceny špatně, tedy zaměněna za jiné.



Obrázek 51: Ukázka tří snímků z testovacího videozáznamu, pro gesto B po detekci rukou a prsů.

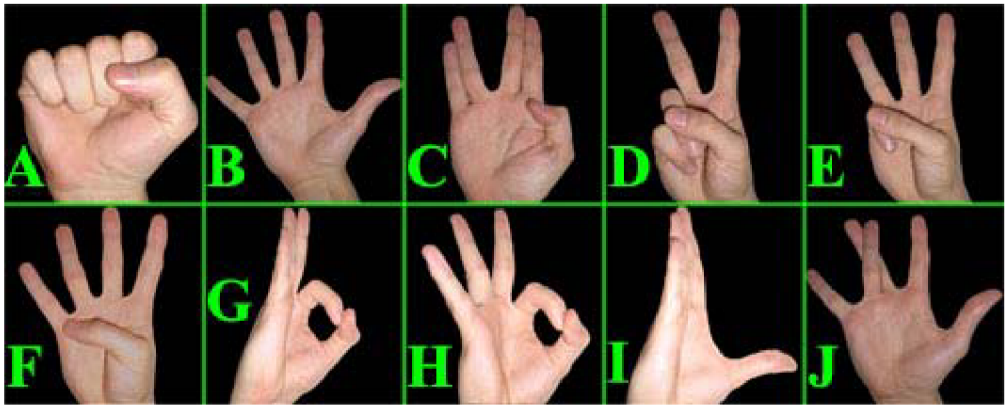
Na následujícím Obrázek 52 jsou zobrazeny ukázky z vyhodnocovaných gest. V každém sloupci jsou gesta jedné třídy, které by měla aplikace schopna rozeznat. Jak je vidět, gesta jedné třídy by měla aplikace být schopna rozeznávat s různým natočením a to dokonce i gesta, kde se nahází předloktí a je vyhodnocována pouze silueta statického gesta ruky.



Obrázek 52: Ukázka gest, které je možné rozpoznat.

Testování rozpoznávání gest probíhalo tak, že je spuštěno testované video s uživatelem provádějící dané gesto a v aplikaci, u které byla zkrácena doba detekce gest pomoci HMM na 2 snímky, bylo ohraničení ruky přibližně stejně veliké. Poté se vyhodnotilo, o jaké gesto se s největší pravděpodobnosti jednalo, a tato informace je zaznamenaná ve formě inkrementace pole tříd gest u vyhodnoceného vzorku. Jakmile aplikace projde celý videozáznam, uloží se všechny informace o počtech nalezených gest do souboru, ze kterého hodnoty přepíši do tabulky. A takto postupuji s každým videem, dokud nejsou všechny hodnoty zapsány v tabulce.

8.1 Testováni první sady gest



Obrázek 53: Ukázka první sady gest pro rozpoznávání.

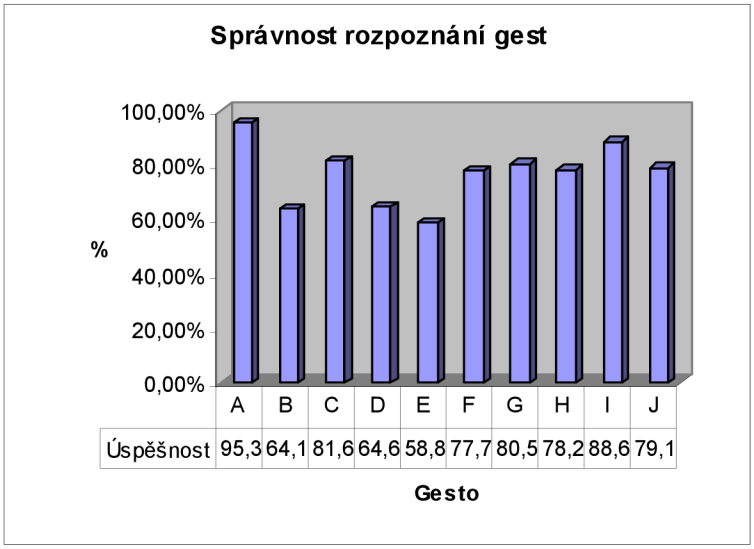
Výše zmíněným způsobem jsem zaznamenal u první sady gest 12245 rozpoznaných gest, z niž bylo 9338 gest detekováno správně. Tedy celková úspěšnost tohoto detektoru gest pro tyto gesta činí 76,66%. Všechny hodnoty můžete vidět v Tabulka **1**:

Gesta	A	B	C	D	E	F	G	H	I	J	Celkem	Úspěšnost
A	**863**	2	19	0	3	3	0	1	1	13	**905**	**95,36%**
B	1	**720**	4	1	4	63	0	0	0	330	**1123**	**64,11%**
C	17	138	**972**	7	1	23	23	7	0	2	**1190**	**81,68%**
D	0	42	188	**802**	159	15	0	0	0	34	**1240**	**64,68%**
E	6	39	77	20	**788**	317	0	18	15	59	**1339**	**58,85%**
F	1	18	1	0	2	**923**	0	0	0	242	**1187**	**77,76%**
G	2	4	0	0	24	2	**1000**	20	183	6	**1241**	**80,58%**
H	2	21	1	2	15	107	1	**909**	0	104	**1162**	**78,23%**
I	1	39	3	6	25	1	43	2	**1391**	58	**1569**	**88,66%**
J	0	67	0	0	1	200	0	1	0	**1020**	**1289**	**79,13%**
											12245	**76,9%**
Chyba	30	370	293	36	234	731	67	49	199	848		
	1,1%	13,0%	10,3%	1,3%	8,2%	25,6%	2,3%	1,7%	7,0%	29,7%		

Tabulka 1: Hodnoty naměřené z testovacích videí pro první sadu gest. Řádky tabulky jsou hodnoty vygenerované z videí pro danou třídu gest a sloupce označují, kolikrát byla jaká třída gest v tomto videu vyhodnocena. Na diagonále je žlutě označen počet správně detekovaných gest. A na posledních dvou sloupcích je celkový počet vyhodnocených gest v daném videozáznamu a procentuální úspěšnost nalezení správného gesta. V předposledním řádku jsou hodnoty celkových chyb, které byly

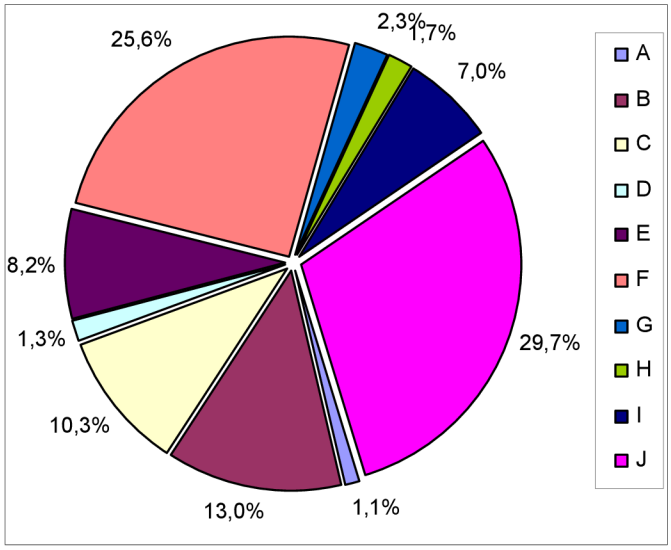
zaznamenány ze všech testovacích videí. A v posledním řádku tabulky jsou procentuálně ohodnoceny chyby oproti celkové chybě ze všech testovacích videozáznamů.

Z hodnot tabulky vyplývá, že nejúspěšnější jsou pro rozpoznáni gesta A, I a C v tomto pořadí. Nejhůře jsou při tom vyhodnocena gesta E, B a D. Hodnoty procentuální úspěšnosti rozpoznání gesta pro první sadu gest jsou znázorněny graficky v následujícím sloupcové grafu **1**.



Graf 1: Úspěšnost rozpoznání jednotlivých gest pro první sadu gest.

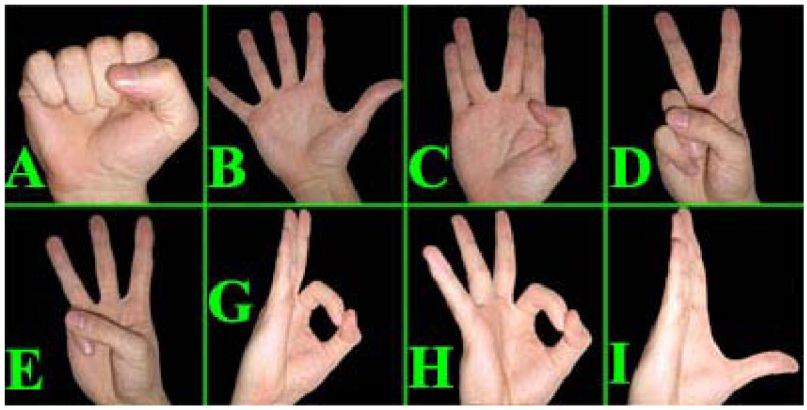
Poslední dva řádky pod hlavní tabulkou označují součet chyb pro jednotlivé třídy sečtené ze všech naměřených hodnot z každého videozáznamu. A jejich procentuální ohodnocení z celkové chyby. Pro lepší znázornění, jsou tyto hodnoty vloženy do následujícího výsečového grafu **2**:



Graf 2: Procentuální ohodnocení celkové chyby pro jednotlivá gesta.

Z těchto hodnot vyplývá, že největší chybu (počet špatně rozpoznaných gest) způsobují gesta J a F, které vytváří dohromady až 55,3 % z celkové chyby. Gesto J je nejhůře vyhodnocováno u gest B a F. Gesto F působí největší chybu u gest E a J. Tyto chyby vznikají, protože je mezi těmito gesty jen málo rozdílu, a proto jsem se rozhodl, gesta F a J z trénovacích dat odstranit pro zlepšení rozpoznávání gest. Gesto J by navíc bylo pro mnoho uživatelů obtížné na provedení.

8.2 Testovaní sady po odebrání gesta F a J



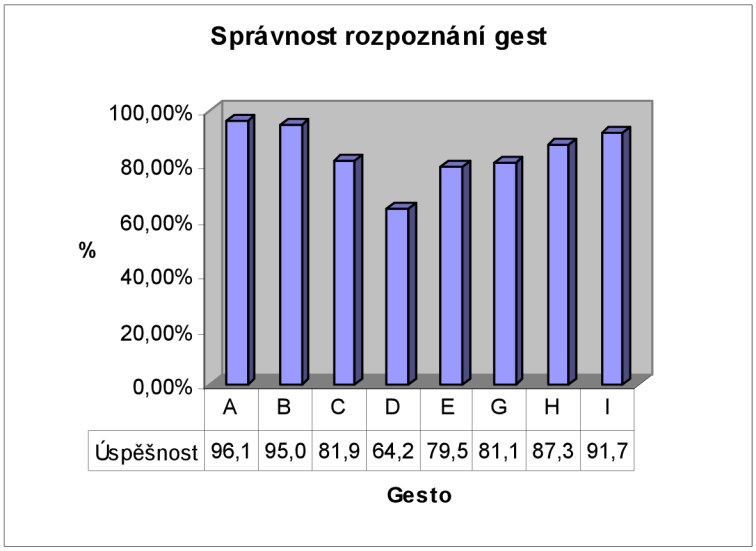
Obrázek 54: Ukázka sady gest pro rozpoznání bez gesta F a J.

Odebráním gest F a J z rozpoznávače gest a opakováním testování bylo rozpoznáno celkem 9776 gest a z toho 8441 bylo rozpoznáno správně. Celková úspěšnost rozpoznávání byla 86,3 %, která se zlepšila oproti původním výsledkům téměř o 10 %. Jak se hodnoty změnily, můžete vidět v následující Tabulka **2**.

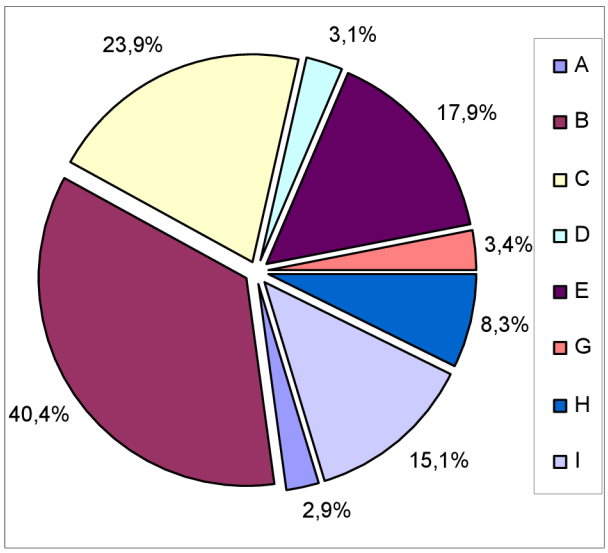
Gesta	A	B	C	D	E	G	H	I	Celkem	Úspěšnost
A	**870**	7	21	0	4	0	2	1	**905**	**96,13%**
B	1	**1067**	14	1	4	0	34	2	**1123**	**95,01%**
C	21	157	**972**	11	1	11	13	0	**1186**	**81,96%**
D	4	86	193	**804**	163	0	0	1	**1251**	**64,27%**
E	6	131	81	20	**1065**	0	26	10	**1339**	**79,54%**
G	4	3	1	0	17	**1007**	22	187	**1241**	**81,14%**
H	2	119	3	2	20	1	**1015**	0	**1162**	**87,35%**
I	1	37	6	7	30	34	14	**1440**	**1569**	**91,78%**
									9776	**84,65%**
Chyba	39	540	319	41	239	46	111	201		
	2,9%	40,4%	23,9%	3,1%	17,9%	3,4%	8,3%	15,1%		

Tabulka 2: Hodnoty naměřené z testovacích videí, pro sadu gest bez gesta F a J. Řádky tabulky jsou hodnoty vygenerované z videí pro danou třídu gest a sloupce označují, kolikrát byla jaká třída gest v tomto videu vyhodnocena. Na diagonále je žlutě označen počet správně detekovaných gest. Na posledních dvou sloupcích je celkový počet vyhodnocených gest v daném videozáznamu a procentuální úspěšnost nalezení správného gesta. V předposledním řádku jsou hodnoty celkových chyb, které byly zaznamenány ze všech testovacích videí. A v posledním řádku tabulky jsou procentuálně ohodnoceny chyby oproti celkové chybě ze všech testovacích videozáznamů.

Nejúspěšněji rozeznatelná gesta jsou nyní gesta A, B a I, jejíž úspěšnost je větší než 90 %. Nejhůře s rozpoznáním je na tom nyní gesto D s hodnotou cca 65 %, které je od druhého nejhůře rozpoznaného gesta téměř s 15 % rozdílem. Z nesprávně vyhodnocených dat také vyplývá, že toto gesto se často zaměňuje za gesto C a E a působí tak velký nárůst u celkové chyby pro tyto gesta. Proto jsem se rozhodl gesto D nahradit novým gestem, které by nebylo tak podobné k ostatním gestům. Hodnoty procentuální úspěšnosti rozpoznání gesta jsou znázorněny graficky v následujícím sloupcovém grafu **3**.

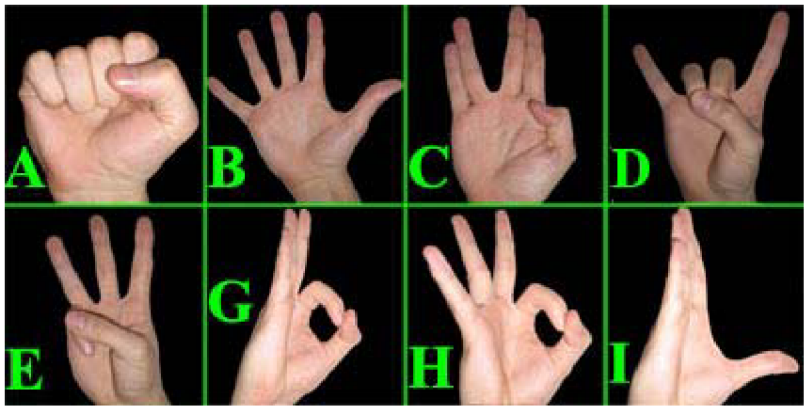


Graf 3: Úspěšnost rozpoznání jednotlivých gest pro sadu gest bez F a J.



Graf 4: Procentuální ohodnocení celkové chyby pro jednotlivá gesta.

8.3 Testování po úpravě gesta D



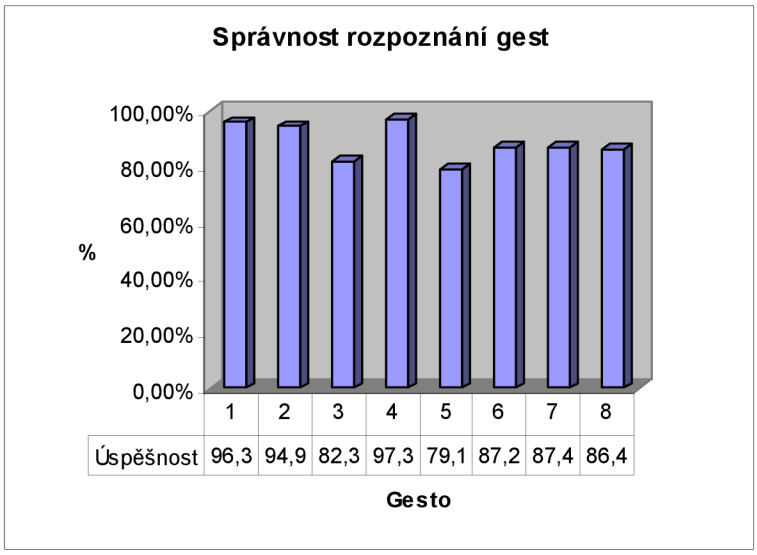
Obrázek 55: Ukázka sady gest pro rozpoznání se změněným gestem D.

Změnou gesta D v sadě gest a opakováním testování bylo rozpoznáno celkem 9969 gest a z toho 8841 bylo rozpoznáno správně. Průměrná úspěšnost rozpoznávání je tedy 88,9 %, která se zlepšila oproti původním výsledkům téměř o 12 %. Jak se hodnoty změnily, můžete vidět v následující Tabulka **3**.

X	A	B	C	D	E	G	H	I	Celkem	Úspěšnost
A	**872**	7	22	0	2	0	1	1	**905**	**96,35%**
B	1	**1066**	23	0	4	0	28	1	**1123**	**94,92%**
C	19	151	**977**	4	1	19	15	0	**1186**	**82,38%**
D	0	4	0	**1446**	34	1	0	0	**1485**	**97,37%**
E	6	134	111	0	**1060**	0	21	7	**1339**	**79,16%**
G	2	4	1	0	26	**1066**	6	117	**1222**	**87,23%**
H	0	123	4	0	16	1	**1005**	0	**1149**	**87,47%**
I	0	120	10	0	34	46	1	**1349**	**1560**	**86,47%**
									9969	**88,92%**
Chyba	28	543	171	4	117	67	72	126		
	2,5%	48,1%	15,2%	0,4%	10,4%	5,9%	6,4%	11,2%		

Tabulka 3: Hodnoty naměřené z testovacích videí po změně gesta D. V řádcích tabulky jsou hodnoty vygenerované z videí pro danou třídu gest a sloupce označují, kolikrát byla jaká třída gest v tomto videu vyhodnocena. Na diagonále je žlutě označen počet správně detekovaných gest. A na posledních dvou sloupcích je celkový počet vyhodnocených gest v daném videozáznamu a procentuální úspěšnost nalezení správného gesta. V předposledním řádku jsou hodnoty celkových chyb, které byly zaznamenány ve všech testovacích videích. A v posledním řádku tabulky jsou procentuálně ohodnoceny chyby oproti celkové chybě ze všech testovacích videozáznamů.

Z tabulky vyplývá, že po změně gesta D se úspěšnost navýšila u gest A, C, D, G a H ostatní gesta o něco klesla. Největší změny si můžete všimnout u změněného gesta D, u kterého je nyní úspěšnost rozpoznání až 97 %. Změnou na lepší gesto se zvedla úspěšnost až o 33 % u tohoto gesta. Hodnoty procentuální úspěšnosti rozpoznání gesta jsou znázorněny graficky v následujícím sloupcovém grafu **5**.



Graf 5: Úspěšnost rozpoznání jednotlivých gest pro sadu gest bez F a J a změněným gestem D.

Úspěšnost rozpoznávače gest by se mohla dále zlepšovat, ale nikdy nebude stoprocentní. Zlepšovat by se dalo např.:

- Volba více odlišných gest
- Zlepšení trénování
- Změnou parametrů HMM
- Nerozpoznávat jen siluetu gesta, ale i gesto s hloubkovou informací
- Zamezením více druhů natočení gesta
- Rozšíření rozpoznávání o další gesta

9 Závěr

Cílem mé diplomové práce bylo navrhnout metodu ovládání počítače gesty. Během práce bylo nutno překonat mnoho překážek a úskalí. Přesto mohu říci, že se požadovaného cíle podařilo dosáhnout. Vytvořil jsem program, který dokáže vyhledat ruku s prsty a detekovat gesta, která jsou poté rozpoznána z dat hloubkového senzoru Kinect. K němu jsem vytvořil prohlížeč fotografií, na kterém demonstruji možnosti a způsoby ovládání pomocí rukou a gest. Můj rozpoznávač gest používající skrytých Markovových modelů není stoprocentně úspěšný. Ale je použitelný k ovládání ukázkové aplikace gesty.

Vytvořená ukázková aplikace pro prohlížení a manipulaci s fotografiemi je plně ovladatelná pomocí pohybů rukou a gest uživatele. Stejně jako by to provedl uživatel myškou a klávesnicí, ale na vzdálenost 1 až cca 2,5 metru rukou od Kinectu.

Mnou navržený rozpoznávač dokáže rozpoznat osm statických gest s průměrnou úspěšností 89 %. O způsobu testování a vyhodnocování úspěšnosti rozpoznávače gest jste se mohli dočíst v kapitole 8. Rozpoznávač gest se dá rozšířit o rozpoznávání dalších gest a dále zlepšovat úspěšnost např.: Volbou odlišnějších gest, změnou parametrů HMM nebo změnou porovnávání ze siluet gest na gesta s informací o hloubce.

Dle mého názoru je detekce rukou z hloubkové mapy mnohem robustnější a má lepší vyhlídky do budoucna než jiné metody, například z jedné kamery a to především proto, že nejsou tolik závislé na osvětlení v místnosti (kromě přímého slunečního světla) a poskytují více informací o prostoru než pouhá barva a jas. Oblast ovládání počítače nebo jiného zařízení třeba TV pomocí gest a pohybu rukou má do budoucna velký potenciál a bude se nadále určitě dále rozvíjet. Už nyní se i na českém trhu nově vyskytují herní konzole a i nové 3D LED televize, například od společnosti Samsung, které jsou úspěšně ovládané hlasem, gesty a pohybem těla, ale jejich pořizovací cena je prozatím stále ještě vysoká 65 000,- Kč.

Do budoucna by se dala vylepšit detekce uživatele, která začíná být nestabilní, když se objeví více uživatelů. Kvalitněji by bylo možné udělat i detekci rukou a gest. Kromě zlepšování ovládání rukou a gesty by bylo možné rozšířit ovládání i na jiné aplikace nebo celý počítač. Stačilo by vytvořit průhledné plátno, na které bych vykresloval pozice rukou a prstů, pro intuitivní ovládání myšky. A zvolením klávesových zkratek simulovaných při rozpoznání gesta, pro ovládání aplikací. Tyto principy udávají směr pro další možný rozvoj ovládání počítače gesty. Snad bude tato práce inspirací pro čtenáře, kteří mají zájem o toto téma a chtěli by vytvořit vlastní ovládání gesty.

Literatura

[1] Blunsom, P.: *Hidden Markov Models*, 2004. [online]. [cit. 2012-05-22]. Dostupné z URL: http://ww2.cs.mu.oz.au/460/2004/materials/hmm-tutorial.pdf

[2] Warakagoda, N.: *Definition of Hidde Markov Model*, 1996. [online]. [cit. 2012-05-22] Dostupné z URL: http://jedlik.phy.bme.hu/~gerjanos/HMM/node4.html#SECTION00220000000000000000

[3] Rabiner, L. R.: *A Tutorial on Hidden Markov Models and Selected Applications in Sérech Recognition*, FELLOW, IEEE, 1989. [online]. [cit. 2012-05-22] . Dostupné z URL: http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/tutorial%20on%20hmm%20and%20applications.pdf

[4] Webová stránka: *Hidden Markov models*, 1998. [online]. [cit. 2012-05-22] Dostupné z URL: http://www.cse.unsw.edu.au/~waleed/phd/tr9806/node12.html

[5] Webová stránka: *Diskrétní kosinová transformace*, Dostupné z URL: http://cs.wikipedia.org/wiki/Diskr%C3%A9tn%C3%AD_kosinov%C3%A1_transformace

[6] Bradski, G.; Kaebler, A.: *Learning OpenCV: Computer Vision with the OpenCV Library*, O`reilly, 2008, s 322-337

[7] Intel: *Open Source Computer Vision Library Reference Manual*, s. 326-334 2001. Dostupné z URL: http://www.cs.unc.edu/Research/stc/FAQs/OpenCV/OpenCVReferenceManual.pdf

[8] Medicton Group: *I4Control*, 2008: . [online]. [cit. 2012-01-05]. Dostupné z URL: http://www.i4control.eu/

[9] Systém I4Control®: *bezkontaktní ovládání osobního počítače*. [online]. [cit. 2012-01-05]. Dostupné z URL: http://cyber.felk.cvut.cz/i4c/cz_system.html

[10] OpenNI Documentation: *Programmer Guide*. [online]. [cit. 2012-01-05]. Dostupné z URL: http://openni.org/Documentation/ProgrammerGuide.html

[11] Pastebin: *Basic code*. [online]. [cit. 2012-01-05]. Dostupné z URL: http://pastebin.com/e5kHzs84

[12] Builder: *Bitové operátory a bitové pole,* 2001. [online]. [cit. 2012-01-06]. Dostupné z URL: http://www.builder.cz/art/cpp/cpp_bitoperator.html

[13] Kathuira Pulpit, *Hand Gesture Recognition*, Japan Advanced Institute of Science and Technology, 2011. Dostupné z URL: http://www.jaist.ac.jp/~s1010205/resources/Computer-Vision/report.pdf

[14] Wikipedia. *Kinect* – Wikipedia, the free encyclopedia, 2011. [Online; accessed 2012-01-09]. Dostupné z URL: http://en.wikipedia.org/wiki/Kinect

[15] Lukáš Jaroň: *S počítačem bez myši a klávesnice*, bakalářská práce, Brno, FIT VUT v Brně, 2010.

[16] Rafael C. Gonzalez, Richard E. Woods: *Digital image processing Third Edition*, Pearson Prencie Hall 2008, s.643-649

[17] Webová stránka: *MGESTYK gesture-based control,* 2008. Dostupné z URL: http://www.mgestyk.com/videos.html

[18] Webová stránka: *Jako v Minority Report – ovládání PC gesty*, 2008. Dostupné z URL: http://magazin.stahuj.centrum.cz/jako-v-minority-report-ovladani-pc-gesty/

[19] Webová stránka: *Samsung EyeCAN: ovládání počítače očima*, 2012. Dostupné z URL: http://extrahardware.cnews.cz/samsung-eyecan-ovladani-pocitace-ocima

[20] Garratt Gallagher: *Kinect Hand Detection*, MIT CSAIL, 2010. Dostupné z URL: http://www.youtube.com/watch?v=tlLschoMhuE

Příloha A

Obsah DVD

Přiložené DVD obsahuje:

- Spustitelný program (./bin/)
- Krátké demonstrační video z běhu aplikace (./demo/)
- Text práce v elektronické podobě (./doc/)
- Plakát (./doc/)
- Instalace prvku pro kompilaci kódu (./install/)
- Zdrojový kódy programu (./src/)