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

„SCI-FI“ HUDEBNÍ KNIHOVNA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JAN HOLAS
AUTHOR

BRNO 2012

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

„SCI-FI“ HUDEBNÍ KNIHOVNA

“SCI-FI” MUSIC LIBRARY

BAKALÁŘSKÁ PRÁCE
BACHELOR’S THESIS

AUTOR PRÁCE — JAN HOLAS
AUTHOR

VEDOUCÍ PRÁCE — Ing. LUKÁŠ POLOK
SUPERVISOR

BRNO 2012

Abstrakt

Tato bakalářská práce se zabývá počítačovým viděním jako možností pro interakci člověka s počítačem. Je zde popsána implementace hudební knihovny a přehrávače, který je ovládaný ukazováním CD krabiček hudebních alb a papírových ovládacích kartiček přehrávače na kameru, která je připojená k počítači. Práce popisuje algoritmy sloužící k segmentaci objektu ze scény na základě jeho obdélníkového tvaru a vyhodnocení shody s obrazovou databází (databazí alb) s použitím detektoru význačných bodů SURF. V závěru práce jsou shrnuty dosažené výsledky a nastíněny nápady a možnosti dalšího vývoje.

Abstract

This thesis deals with usage of computer vision as a way of interaction between human and computer. It introduces implementation of music library and audio player which is controlled by showing CD jewel cases of music albums and audio player paper control cards on a camera connected to a computer. This thesis describes algorithms for segmentation of an object from scene based on object's rectangular shape and matching that image with image database (music album database) using SURF feature detector. In conclusion, it summarizes achieved results and mentions some ideas and possibilities of further development.

Klíčová slova

Hudební knihovna, počítačové vidění, segmentace obrazu, detekce obdélníku, porovnání obrazu, SURF, OpenCV.

Keywords

Music library, computer vision, image segmentation, rectangle detection, image matching, SURF, OpenCV.

Citace

Jan Holas: „Sci-Fi“ hudební knihovna, bakalářská práce, Brno, FIT VUT v Brně, 2012

„Sci-Fi“ hudební knihovna

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Lukáše Poloka. Uvedl jsem všechny literární prameny, ze kterých jsem čerpal.

. . . . . .
Jan Holas
16.5.2012

Poděkování

Chtěl bych poděkovat panu Ing. Lukáši Polokovi za odborné rady a konzultace během řešení této bakalářské práce. Dále děkuji Lence Štajnerové za pomoc při výrobě kartonových ovládacích kartiček.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Tradiční přístup interakce člověka a počítače je klávesnice a myš. Od jejich vzniku do dnešní podoby tato zařízení prošla dalším technologickým, ergonomickým a designovým vývojem, princip přesto zůstal stejný. V nedávné době se začaly objevovat nové přístupy pro ovládání počítače, např. hlasem. Tento přístup už má za sebou delší vývoj a dnes je poměrně rozšířený i pro počítače s obyčejným hardwarovým a softwarovým vybavením. Další alternativou, která se dnes objevuje stále častěji je interakce pomocí snímání scény kamerou. Člověk pak počítač ovládá například pohybem těla, ukazováním objektů na kameru, gesty ruky nebo pohybem očí. Tyto inovativní metody mohou být využity například pro snazší umožnění přístupu k počítači handicapovaným lidem, ale i zdravému člověku může ušetřit práci a učinit ovládaní snadnější.

Tato bakalářská práce popisuje algoritmy a implementaci hudební knihovny a přehrávače, který je ovládaný prostřednictvím počítačového vidění, a to ukázáním obrázku hudebního alba (*cover art*) na krabičce CD na kameru připojenou k počítači. Pomocí detekce krabičky CD ve snímané scéně a porovnání s databází známých obrazů je album rozpoznáno a začne se přehrávat. Pokud systém album nerozpozná (dosud není v knihovně obsaženo), je nabídnuta možnost uložit dané album do knihovny. Ovládaní přehrávače (akce play, pause, stop, atd.) je řešeno formou kartiček se symboly a nápisy akce rovněž ukazovanými na kameru.

Práce se skládá z 6 kapitol. Kapitola 2 je obecná a krátce se věnuje počítačovému vidění a knihovně OpenCV, která je v této práci použita. Další kapitola 3 se zabývá segmentací popředí a pozadí a popisuje způsoby nalezení krabičky alba ve scéně. Kapitola 4 je o porovnávání nalezeného obrázku alba s uloženou databází a popisuje metody vyhodnocení shody. Tyto dvě kapitoly se zaobírají teoretickým rozborem a řešením problému. Konkrétněji implementaci hudební knihovny přibližuje kapitola číslo 5. Popisuje princip fungování aplikace, její strukturu a hlavní programové třídy. Poslední kapitolou číslo 6 je závěr. Shrnuje dosažené výsledky, popisuje možnosti dalšího vývoje a zmiňuje zjištěné problémy a omezení aplikace.

Kapitola 2

Počítačové vidění

Počítačové vidění je obor informatiky, který se zabývá obecně zpracováním a analýzou obrazových dat počítačem, získání informace z těchto dat a částečně tak napodobit lidské vidění [2]. Typické úlohy jsou segmentace a detekce objektů ve scéně a jejich porozumění, sledování a analýza pohybu nebo rekonstrukce scény. Z hlediska výpočetní techniky jde o relativně nový obor a v současnosti je stále velmi otevřený.

2.1 Knihovna OpenCV

OpenCV (Open Source Computer Vision) je multiplatformní knihovna zaměřená na práci s obrazem s velkým důrazem na zpracování videa v reálném čase. Původně byla vyvíjena společností Intel, první alfa verze byla vydána v roce 1999. Knihovna je napsána v jazyce `C`, ale v současnosti poskytuje i modernější objektové rozhraní pro jazyk `C++`, pro který je knihovna nyní především vyvíjena. Navíc existují i možnosti vazby pro `C#`, `Python`, `Java` a další jazyky. Je vydána pod licencí BSD (kromě některých algoritmů, které jsou patentované a obsahují vlastní licenci[1]).

Uživatelům, kteří knihovnu použijí, poskytuje prostředky pro pohodlnou práci s maticemi a obrazovými daty, dále obsahuje optimalizované algoritmy z různých oblastí počítačového vidění a strojového učení, které s počítačovým viděním souvisí. Podle [3] už knihovna obsahuje více než 2500 algoritmů. Knihovna je nyní velmi rozšířená se stále se zvětšující uživatelskou základnou. Podrobnější informace jsou k dispozici na webových stránkách projektu [3] nebo v publikacích Learning OpenCV [4] a OpenCV 2 Computer Vision Application Programming Cookbook [5], ze kterých jsem rovněž čerpal.

[1] Například algoritmus SURF, který je využitý v této práci, lze použít pouze k nekomerčním účelům

Kapitola 3

Segmentace pozadí a popředí

Prvním krokem, který pomůže detekovat a rozpoznat ukázané album na kameru je segmentace popředí a pozadí. Jedná se o metody k rozdělení obrazu na části se společnými vlastnostmi a jejich zachování nebo odstranění podle toho, jestli s nimi dále chceme nebo nechceme pracovat. V tomto případě potřebujeme pro další zpracování ze scény získat obdélník představující krabičku alba.

3.1 Segmentace konstantního pozadí (klíčování)

Jedná se o nejjednodušší přístup k segmentaci, který předpokládá, že pozadí bude tvořené konstantní odrazivostí nebo pohltivostí svého povrchu [2]. Spočívá ve vyhledání pixelu stejné nebo podobné barvy, která je předem známá. Tato metoda se používá například v kinematografii, kde herci hrají před plátnem z konstantní barvy, obvykle zelené nebo modré, které je poté nahrazené jiným prostředím.

Vzhledem k situaci a potřebám aplikace je taková možnost nereálná a je zde uvedena jako teoretické východisko. V následujících podkapitolách jsou popsány dva přístupy ke klíčování pozadí.

3.1.1 Prahování

Metoda prahování spočívá v porovnání dané barevné složky jednotlivých pixelů vstupního obrazu s určenou prahovou hodnotou. Tato prahová hodnota může být určena napevno, dopočítána podle hodnoty ostatních barevných složek nebo zjištěna z histogramu. Na obrázcích 3.1 je ukázán vstupní obrázek (a) a výstup po aplikaci prahování (b), kdy jsou odstraněny zelené pixely.

3.1.2 Pravděpodobnostní určení pozadí

Tento způsob klíčování se podobá prahovaní, ale je o něco robustnější vůči vlivům, kdy hodnoty jasové funkce pixelů pozadí nejsou plně konstantní. Takový jev může být následek toho, že některá místa pozadí jsou hůře nasvícena. Pro každý pixel lze spočítat pravděpodobnost, zda se jedná o pozadí či popředí, a vhodně zvoleným pravděpodobnostním prahem lze pak obraz segmentovat. Postup se skládá ze dvou kroků.

Prvním je spočítání kovarianční matice z části obrazu, o které víme, že je pozadí. Kovarianční matice [6] je taková matice, kde na hlavní diagonále jsou hodnoty rozptylu (variance)

(a) vstupní obrázek

(b) prahování

(c) pravděpodobnostní určení pozadí

Obrázek 3.1: Klíčování pozadí

veličiny náhodného vektoru $\vec{\mathbf{X}}$ a prvek matice $c_{ij}, i \neq j$ je kovariance mezi prvky x_i a x_j náhodného vektoru $\vec{\mathbf{X}}$. Kovariance mezi těmito veličinami je dána vztahem

$$cov(x_i, x_j) = E[(x_i - E(x_i)).(x_j - E(x_j))], \tag{3.1}$$

kde $E(x_i)$ je střední hodnota veličiny x_i. Vektor $\vec{\mathbf{X}}$ je v tomto případě složen ze tří veličin, barevných složek pixelu R, G a B. Každá tato veličina obsahuje hodnoty dané barevné složky jednotlivých pixelů „trénovacího“ pozadí, ze kterých je vypočítána střední hodnota. Tato kovarianční matice pak vyjadřuje rozložení pravděpodobnosti pro pozadí.

Jako druhý krok následuje vlastní segmentace. S předem spočítanou kovarianční maticí je pak možné pro každý obrazový pixel určit pravděpodobnost, zda se jedná o pozadí na základě předpokladu, že barevné složky pixelu odpovídají vícerozměrnému normálnímu (Gaussovu) rozložení. Jeho sdružená hustota pravděpodobnosti pro 3-rozměrný vektor $\vec{\mathbf{x}} = (r, g, b)$ má tvar [6]

$$f_x(r, g, b) = \frac{1}{(2\pi)^{3/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)\right), \tag{3.2}$$

kde Σ je kovarianční matice, $|\Sigma|$ je její determinant a μ je vektor středních hodnot. Výsledek klíčování touto metodou je ukázán jako obrázek 3.1 (c).

3.2 Detekce obdélníkových vzorů v obraze

Jak již bylo uvedeno dříve, metoda klíčování není praktická kvůli nutnosti zajištění klíčovacího pozadí a jeho silné závislosti na uniformitě osvětlení, aby každý pixel tohoto pozadí měl téměř stejnou hodnotu jasové funkce. Z tohoto důvodu je zvolen jiný přístup a to segmentace na základě tvaru objektu, což je v tomto případě krabička CD ve tvaru obdélníku.

Metoda detekce obdélníkových vzorů v obraze je prezentována v [7] a její varianta je použita v této práci. Využívá Houghovy transformace k nalezení rovných čar a spojení takových čar ve scéně, které splňují podmínky pro vytvoření obdélníku a vedou k získání jeho čtyř vrcholů. Takto vzniklý obdélník je ve scéně obvykle natočený a zároveň mírně deformovaný (strany nesvírají přesné pravé úhly) kvůli perspektivě. Jako poslední krok je tedy třeba provést geometrickou transformaci a oblast ohraničenou těmito vrcholy přemapovat do pravoúhlé mřížky a získat tak nový obraz připravený pro další zpracování. Postup segmentace je popsán v následujících podkapitolách.

3.2.1 Detekce hran

Pro detekci hran ve scéně je použitý Cannyho hranový detektor [8]. Vyvinul ho John F. Canny v roce 1986. Tento algoritmus byl vytvořen pro splnění požadavků optimálního detektoru [2]:

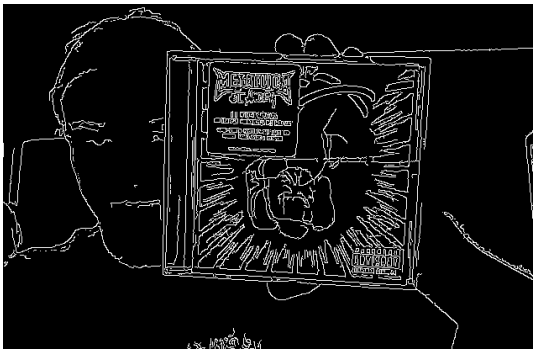
- Detekční kritérium– algoritmus by měl najít všechny významné hrany.
- Lokalizační kritérium – hrany by měly být co nejpřesněji určeny vzhledem ke vstupnímu obrázku.
- Kritérium jednoznačnosti – nesmí docházet ke zdvojení, hrana musí být detekována právě jednou.

Algoritmus se dá shrnout do 4 hlavních kroků:

1. Zbavení obrazu šumu Gaussovým filtrem.
2. Standardní detekce hran (například Sobelovým operátorem) – určení velikosti a směru gradientů.
3. Nalezení lokálních maxim z předešlých hodnot a odebrání bodů, které nejsou maximem. Hrana tak bude detekována v místě největšího gradientu.
4. Prahování s hysterezí – gradient je testován dvěma prahy, vysokým a nízkým. Pokud gradient leží nad vysokým prahem, je prohlášen za hranu. Pokud leží mezi těmito prahy, je prohlášen za hranu pouze pokud sousedí s bodem, který už za hranu prohlášen byl. Pokud leží pod nízkým prahem, je odstraněn. Tento postup slouží k odstranění nevýrazných hran. Doporučené poměry hodnot mezi vysokým a nízkým prahem jsou 1:2 nebo 1:3 [4].

Výstupem detektoru je binární obraz (složený pouze ze dvou barev). Na obrázcích 3.2 je znázorněn vstup a výstup detektoru pro různé hodnoty prahů.

(a) vstup



(b) výstup, hodnoty prahů 30 a 90

(c) výstup, hodnoty prahů 100 a 200

Obrázek 3.2: Cannyho detektor hran

3.2.2 Houghova transformace

Houghova transformace [8] [9] je algoritmus, který slouží k nalezení struktur v obraze, které lze parametricky definovat matematickou rovnicí. Využívá se především k nalezení přímek, kružnic nebo elips, tedy útvarů, které jsou snadno definovatelné. Tento algoritmus byl poprvé patentován Paulem Houghem v toce 1962, do podoby, jak je známy dnes, ho formulovali Richard Duda a Peter Hart v roce 1972.

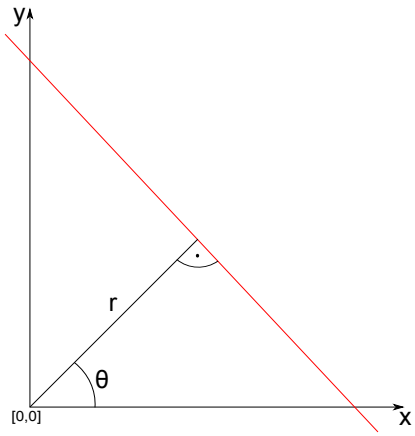
V této práci je využita Houghova transformace pro detekci čar (přímek). Možností jak matematicky vyjádřit přímku je více. Nejběžnějším způsobem vyjádření je směrnicová rovnice přímky 3.3. Nevýhodou tohoto zápisu je, že pro přímky rovnoběžné s osou y se stává parametr m nekonečným a pro použití je tak nevhodný. Z tohoto důvodu se používá polární tvar rovnice přímky 3.4, resp. čitelnější upravený tvar 3.5 stejné rovnice, kde parametr r určuje vzdálenost přímky od počátku souřadnic a θ je úhel mezi normálovým vektorem přímky a osou x. Obrázek 3.3 situaci znázorňuje.

$$y = mx + b \tag{3.3}$$

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right)x + \left(\frac{r}{\sin\theta}\right) \tag{3.4}$$

$$r = x\cos\theta + y\sin\theta \tag{3.5}$$

Každé nalezené přímce lze pak přiřadit jednoznačnou uspořádanou dvojici parametrů (r, θ) pokud platí $\theta \in \langle 0, \pi)$ a $r \in \mathbb{R}$ nebo $\theta \in \langle 0, 2\pi)$ a $r \geq 0$.



Obrázek 3.3: Přímka v polárních souřadnicích

Vlastní detekce probíhá tak, že pro každý nenulový vstupní bod obrazu (z důvodu optimalizace je nejčastěji vstupem binární obraz po detekci hran) dosadíme jeho souřadnice $[x, y]$ do rovnice 3.5 a dopočítáme r pro každý úhel $\theta \in \langle 0, \pi)$ nebo $\theta \in \langle 0, 2\pi)$. Jelikož se jedná o nekonečně mnoho hodnot, je předem stanoven krok, se kterým se úhel zvětšuje a počet dosazovaných úhlů je tak konečný. Dvojice parametrů (r, θ) pak udává souřadnice bodu v poli, tzv. akumulátoru, který je ze začátku naplněn nulovými hodnotami. Vzhledem k tomu, že známe velikost vstupního obrazu, můžeme určit i potřebné rozměry akumulátoru – vzdálenost přímky r od počátku souřadnic bude maximálně velikost úhlopříčky vstupního obrazu. Pokud je zvolen interval úhlu θ jako $\langle 0, \pi)$, r může nabývat i záporných hodnot a tato vzdálenost tak bude muset být dvojnásobná. V dopočítaném bodě (r, θ) přičteme předem stanovenou hodnotu (nejčastěji 1) a zvýšíme tak jeho intenzitu (tzv. hlasování).

Obrázek 3.4: Houghova transformace – naplněný akumulátor

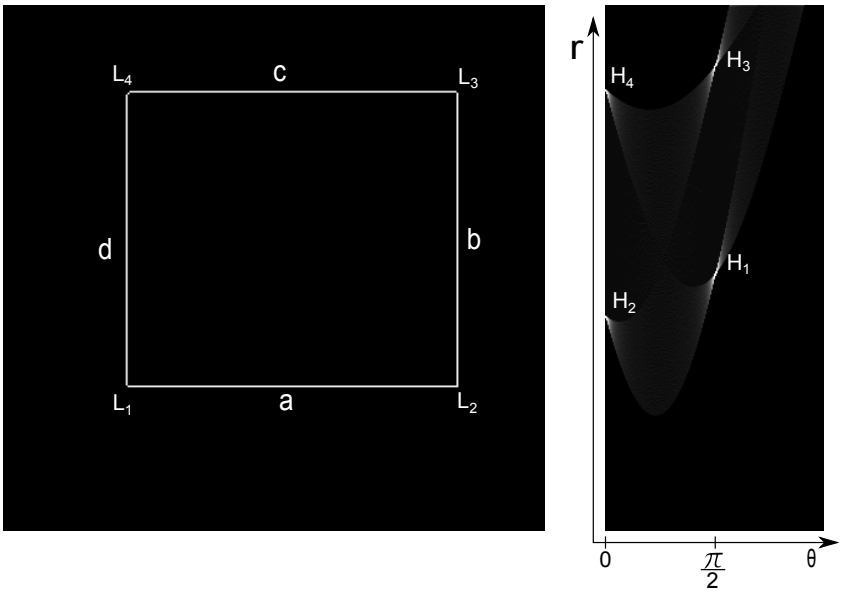
Akumulátor bude mít po naplnění (projití všech vstupních bodů zkoumaného obrazu) body s různou intenzitou. Toto můžeme vidět na obrázku 3.4. Vstupní obraz, ze kterého je akumulátor naplněn, je uveden na obrázku 3.2 (c). Pokud nenulové body ve vstupním obrazu leží na jedné přímce, budou mít stejné parametry (r, θ) a body v akumulátoru na těchto souřadnicích budou mít největší intenzitu. Pokud lokální maxima těchto intenzit leží nad určenou prahovou hodnotou (čára ve vstupním obrazu je dostatečně dlouhá), jsou souřadnice těchto bodů uloženy jako parametry (r, θ), které jednoznačně definují přímku podle rovnice 3.5.

3.2.3 Detekce obdélníků

Z předchozího kroku máme uloženy parametry (r, θ) nalezených dostatečně výrazných přímek. Je potřeba nalézt takovou čtveřici bodů $H_1 = (r_1, \theta_1)$, $H_2 = (r_2, \theta_2)$, $H_3 = (r_3, \theta_3)$, $H_4 = (r_4, \theta_4)$, které splňují podmínky, aby přímky a, b, c a d, které tyto body reprezentují, ze svých průsečíků L_1, L_2, L_3, L_4 formovaly obdélník. Tyto podmínky jsou následující:

1. Body (přímky) se objevují v párech $P = (H_1, H_3)$, přičemž pro tuto dvojici přímek musí platit $\theta_1 - \theta_3 = 0$ (přímky mají stejný úhel, jsou tedy rovnoběžné).
2. Dvojice párů $P_1 = (H_1, H_3)$, $P_2 = (H_2, H_4)$ je na sebe kolmá, tedy $\theta_1 - \theta_2 = 90°$ (resp. $\theta_3 - \theta_4 = 90°$).

Situace je znázorněná na obrázku 3.5. Můžeme zde vidět vstupní obrázek s obdélníkovým vzorem a naplněný akumulátor s vyznačenými body.



Obrázek 3.5: Obdélník a část akumulátoru

Postup detekce se skládá z následujících kroků:

1. Nejdříve jsou porovnány body $H_1 = (r_1, \theta_1)$, $H_2 = (r_2, \theta_2)$,..., $H_n = (r_n, \theta_n)$ tak, že je porovnán každý s každým. Pokud dvojice bodů splňuje 1. podmínku $|\theta_i - \theta_j| < T_\theta$, kde T_θ je prahová hodnota, jsou tyto body uloženy jako nový pár P. Je třeba zvolit vhodný práh T_θ podle vyžadované přesnosti (měl by se blížit 0).

2. Všechny uložené páry jsou rovněž porovnány každý s každým. Pokud páry $P_k = ((r_{k_1}, \theta_{k_1}), (r_{k_2}, \theta_{k_2}))$ a $P_l = ((r_{l_1}, \theta_{l_1}), (r_{l_2}, \theta_{l_2}))$ splňují 2. podmínku $||\theta_{k_1} - \theta_{l_1}| - 90°| < T_\theta$, je uložen obdélník $R = (P_k, P_l)$.

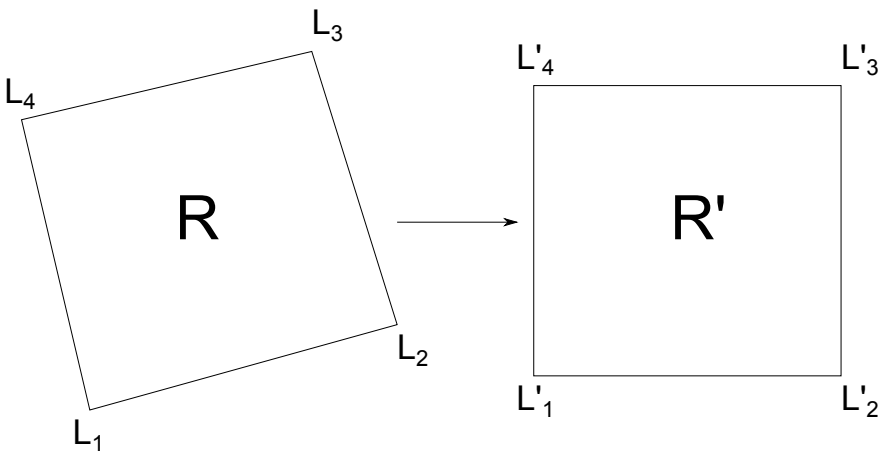
3. Výpočet průsečíků přímek. Mezi každými na sebe kolmými přímkami z uloženého obdélníku $R = (P_k, P_l)$ je vypočítán průsečík $L = (x, y)$, který reprezentuje jeho vrchol. Z každé ze dvou takových přímek určených body (r_i, θ_i) jsou vypočítány dva body (x_1, y_1), (x_2, y_2) pro první přímku a (x_3, y_3), (x_4, y_4) pro druhou přímku. Tyto body už jsou v kartézských souřadnicích a odpovídají souřadnicím pixelu ve vstupním obraze. Rovnice 3.6 [10] udávají vztahy pro převod bodu z polárních souřadnic do kartézských[1]. Pro výpočet průsečíku L dvou přímek, kde každá je zadaná dvěma body, je použit vztah 3.7 [12]. Při použití tohoto vztahu musí platit, že $(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4) \neq 0$, v opačném případě by přímky byly paralelní. Po dopočítání ostatních průsečíků získáme vrcholy obdélníku L_1, L_2, L_3 a L_4.

$$x = r.\cos\theta \quad y = r.\sin\theta \tag{3.6}$$

$$\begin{aligned} L_x &= \frac{(x_1y_2 - y_1x_2)(x_3 - x_4) - (x_1 - x_2)(x_3y_4 - y_3x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)} \\ L_y &= \frac{(x_1y_2 - y_1x_2)(y_3 - y_4) - (y_1 - y_2)(x_3y_4 - y_3x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)} \end{aligned} \tag{3.7}$$

3.2.4 Geometrická transformace

Po získání vrcholů z předchozího kroku je potřeba nalezený obdélník transformovat do nového obrazu. Obdélník může mít ve scéně různou rotaci a navíc může být mírně deformovaný kvůli perspektivě, protože podmínka na pravoúhlost stran není absolutní, ale je určena prahovou hodnotou. Problém je demonstrován na obrázku 3.6.



Obrázek 3.6: Geometrická transformace

[1] Takto získáme pouze 1 bod, pro výpočet 2 bodů musíme ke každé souřadnici tohoto bodu přičíst nebo odečíst určitou vzdálenost ve směru přímky, tedy například $x_1 = r.\cos\theta + 1000(-\sin\theta)$, $y_1 = r.\sin\theta + 1000(\cos\theta)$ a $x_2 = r.\cos\theta - 1000(-\sin\theta)$, $y_2 = r.\sin\theta - 1000(\cos\theta)$ [11].

Do cílového obrazu je potřeba namapovat odpovídající pixely ze zdrojového obrazu. Tento vztah lze vyjádřit matematicky jako $R'(x, y) = R(f_x(x, y), f_y(x, y))$, kde f je funkce, která určuje korespondující pixely. Její funkční hodnoty (souřadnice zdrojového pixelu) obvykle nespadají do celočíselných hodnot (tedy do souřadnic jednoho konkrétního pixelu), je tedy třeba danou hodnotu jasové funkce pixelu aproximovat z pixelů, ke kterým se dané desetinné souřadnice blíží.

Jelikož z předchozího kroku známe 4 ohraničující body, je možné využít homografii (též nazývaná projektivní transformace nebo kolineace) [8]. Tato transformace je dána vztahem:

$$\alpha u' = \mathbf{H} u, \tag{3.8}$$
$$u' = \begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix}, \quad u = \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix},$$
$$\alpha \in \mathbb{R} - \{0\},$$

kde α je skalární násobek, u a u' jsou homogenní souřadnice odpovídajících si bodů a $\mathbf{H}$ je matice homografie

$$\mathbf{H} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}, \tag{3.9}$$

která popisuje vztah mezi body jednotlivých obrazů. Tato matice je naší neznámou. Pro její jednoznačný výpočet jsou potřeba minimálně 4 body, které si odpovídají a z toho žádné 3 neleží na jedné přímce. Z rovnice 3.8 můžeme vyjádřit mapovací vztah:

$$R'(x, y) = R\left(\frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}\right). \tag{3.10}$$

Prvek h_{33}, který by udával měřítko, můžeme nastavit na normalizovanou hodnotu $h_{33} = 1$ [4]. Tím nám zbývá 8 neznámých. Dosazením a úpravou rovnice 3.10 získáme vztahy:

$$\begin{aligned} h_{11}x + h_{12}y + h_{13} - x'h_{31}x - x'h_{32}y - x' = 0, \\ h_{11}x + h_{12}y + h_{13} - y'h_{31}x - y'h_{32}y - y' = 0. \end{aligned} \tag{3.11}$$

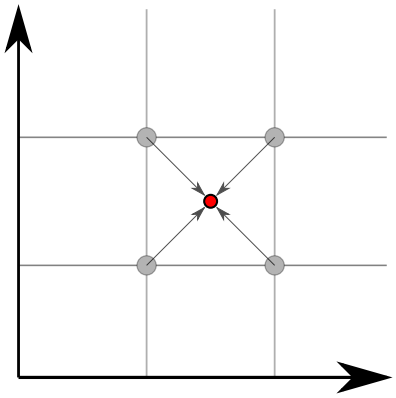
Pokud známe 4 body, lze pomocí těchto bodů a rovnic 3.11 a sestavit soustavu 8 (obecně $2n$, kde n je počet bodů) lineárních algebraických rovnic o 8 neznámých [13]:

$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 & -y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 & -x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & -y'_2y_2 & -y'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 & -x'_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y'_3x_3 & -y'_3y_3 & -y'_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 & -x'_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & -y'_4y_4 & -y'_4 \end{pmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{pmatrix} = 0. \tag{3.12}$$

Po vyřešení této soustavy už lze s vypočítanou maticí homografie provést mapování jednotlivých pixelů podle rovnice 3.10. Hodnoty pixelů, které neleží na celočíselných souřadnicích, je potřeba interpolovat z jejich okolí.

Nejjednodušší metodou je interpolace nejbližším sousedem. Zde dojde pouze k zopakování hodnoty pixelu, ke kterému se daná funkční hodnota nejvíce blíží. Tato metoda není příliš kvalitní a produkuje silný aliasing.

Lepší metoda, která je v implementaci použita, je bilineární interpolace. K určení hodnoty jasové funkce pixelu dojde ze 4 okolních pixelů, které dané neceločíselné souřadnice nejblíže ohraničují. Tato situace je znázorněna na obrázku 3.7. Výsledkem této metody je hladší a jemnější obraz (velký anti-aliasing).



Obrázek 3.7: Bilineární interpolace

Po tomto kroku je segmentace pomocí detekce obdélníkových vzorů kompletní a obrázek alba je tak připraven na další zpracování. Na obrázku 3.8 můžeme vidět celý výsledek tohoto procesu.

(a) scéna (b) segmentované hudební album

Obrázek 3.8: Segmentace alba ze scény

Kapitola 4

Vyhledávání v databázi obrazů

V této kapitole je popsán návrh řešení problému porovnávání obrazů (anglicky *image matching*). Pro dotazující se obraz (*query image*) je třeba vyhodnotit shodu se známou databází. Z hlediska této práce je potřeba segmentovaný obrázek hudebního alba z předchozího kroku nalézt v databázi referenčních obrázků, pokud tam existuje. K tomuto účelu je použit detektor význačných bodů, který dokáže identifikovat body v obraze, které jsou něčím zajímavé, a následně je a jejich okolí popsat pomocí charakteristických vektorů, které se nazývají deskriptory. Tyto deskriptory pak lze z různých obrazů mezi sebou přímo porovnávat a určit tak míru podobnosti.

4.1 Detektory význačných bodů

Význačné body v obraze (*local features*) jsou obecně body nebo oblasti, které se nějakým způsobem liší od svého bezprostředního okolí. Můžou to být například hrany, rohy nebo i skupiny více bodů, které mají podobné vlastnosti (bloby). Dobré význačné body a jejich detektory by měly oplývat těmito vlastnostmi [14]:

- Opakovatelnost – pokud máme dva obrazy stejné scény pořízené za různých pozorovacích podmínek (např. jiná rotace, úhel pohledu, měřítko), vysoké procento význačných bodů nalezené na jednom z obrazů by mělo být nalezeno i na druhém. Body by tedy měly být dostatečně robustní nebo invariantní vůči výše zmíněným vlivům.
- Rozlišitelnost/informativnost – struktury okolí nalezených bodů by měly být bohaté na informace a být dobře rozlišitelné pro další použití, jako je například srovnávání.
- Lokálnost – nemělo by dojít k vzájemnému pohlcení bodů.
- Množství – počet nalezených bodů by měl být dostatečně velký i pro malé objekty. Optimální číslo sice záleží na konkrétních potřebách aplikace, ale množství by mělo být ovlivněno určením jednoduchého a intuitivního prahu.
- Přesnost – bod by měl být v obraze přesně lokalizován.
- Efektivita – detekce význačných bodů by měla být použitelná v časově kritických aplikacích.

Stupeň důležitosti těchto vlastností závisí na účelu použití a nelze je všechny splnit současně v plné míře, protože se některé vzájemně vylučují. Například zvýšený stupeň robustnosti může vést k menší rozlišitelnosti, protože důležitá informace může být považována

za šum a tak ignorována. Je proto důležité mít představu o použití a z toho se odvíjejících požadavků. Existují různé detektory a deskriptory význačných bodů a z výše zmíněných důvodů nelze univerzálně vybrat nejlepší, protože každý má jiné vlastnosti. Pro účely této práce jsem se rozhodl použít detektor a deskriptor SURF, který vyhledává skvrnové struktury (bloby). Tento algoritmus je stručně popsán v následující podkapitole.

4.2 SURF

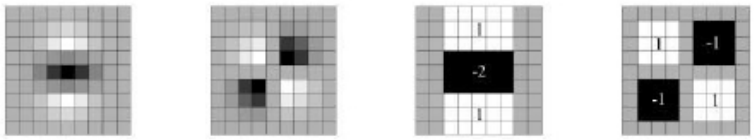
SURF (**S**peeded-**U**p **R**obust **F**eatures) je detektor, který publikovali H. Bay, T. Tuytelaars a L. Van Gool v roce 2006 [1]. Částečně je inspirován dříve představeným algoritmem SIFT [15] a zlepšuje některé jeho vlastnosti a to, jak již název algoritmu napovídá, především rychlost. Základní verze detektoru je invariantní vůči změně měřítka a rotace. Význačné body se u tohoto detektoru označují jako klíčové body a stejně budou označovány i dále v tomto textu.

4.2.1 Princip detekce klíčových bodů

Detekce klíčových bodů je založena na vyhledání extrémů determinantu Hessovy matice, přičemž tato matice je aproximovaná pomocí integrálního obrazu. Autoři tuto metodu nazvali „rychlý Hessův detektor“ (*Fast-Hessian detector*). Hessova matice je efektivní při detekování blobů. Pro pixel $\mathbf{x} = (x, y)$ obrazu I v měřítku σ má tato matice tvar

$$\mathbf{H} = \begin{pmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{pmatrix}, \tag{4.1}$$

kde L_{xx} je konvoluce obrazu I v bodě $\mathbf{x} = (x, y)$ s aproximací druhé parciální derivace Gaussovy funkce $\frac{\delta^2}{\delta x^2} g(\sigma)$, obdobně pro L_{xy} a L_{yy}. Tyto konvoluce se počítají pro různá měřítka, aby detektor byl vůči jejich změnám invariantní. Z tohoto důvodu se mění velikost použitých Gaussových filtrů derivovaných od Gaussovy funkce. Ukázka diskretizovaných filtrů a jejich aproximace pomocí obdélníkové funkce, které SURF při konvoluci používá, je znázorněna jako obrázek 4.1. Tato obdélníková aproximace překvapivě dosahuje stejných nebo lepších výsledků než věrnější aproximace (levá část obrázku).

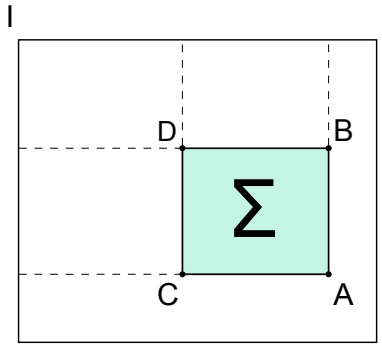


Obrázek 4.1: Zleva doprava – diskretizované a ořezané reprezentace druhé parciální derivace Gaussovy funkce v směru y (L_{yy}) a xy (L_{xy}). Následující dva obrázky představují další aproximaci těchto filtrů, kterou algoritmus SURF používá. Šedé oblasti odpovídají 0. Převzato z [1].

Integrální obraz $I_{\sum}$ je obraz, který v každém svém bodě $I_{\sum}(x)$ obsahuje součet bodů vstupního obrazu I v obdélníkové oblasti mezi tímto bodem a počátkem:

$$I_{\sum}(x) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j). \tag{4.2}$$

Tato reprezentace slouží k rychlému součtu bodů v obdélníkovém regionu kdekoliv v tomto obraze s malou konstantní složitostí, a to vztahem $\sum = A - B - C + D$, kde A, B, C a D jsou body v integrálním obrázku tak, jak je ukázáno na obrázku 4.2.

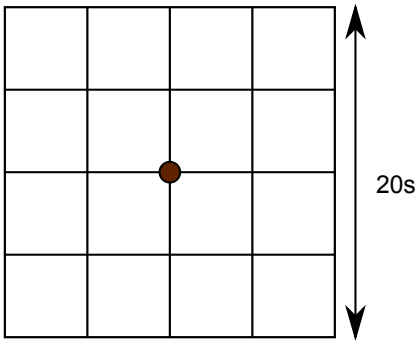


Obrázek 4.2: Integrální obrázek a součet bodů v obdélníkovém regionu

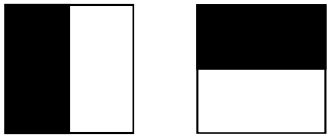
K výpočtu jsou tedy potřeba jen 2 operace odčítaní a 1 sčítání. To je výhodné k rychlému výpočtu konvolucí nad obrazem, kde konvoluční filtry mohou být libovolně velké, čehož algoritmus využívá právě při zvětšování Gaussových filtrů pro vybudování měřítkově nezávislé reprezentace obrazu, tzv. *scale-space*. Nejmenší filtr má velikost `9x9` a postupně se zvětšuje o stejný počet pixelů v rámci jednoho stupně – oktávy. U první oktávy se filtr v několika krocích zvětšuje o 6 pixelů, v každé další oktávě o dvojnásobek. Determinanty Hessovy matice (zkráceně Hessiány) vyjadřují odezvu na blob a po měřítkové normalizaci se ukládají do odezvových map (trojrozměrná struktura, kde třetím rozměrem je měřítko σ), ve kterých jsou nalezeny lokální extrémy v rámci pozice a měřítka. Pokud tyto extrémy leží nad prahovou hodnotou, jsou identifikovány jako klíčové body.

4.2.2 Konstrukce deskriptorů

SURF deskriptory popisují distribuci intenzity v okolí klíčového bodu. Konstrukce deskriptoru spočívá v sestrojení čtvercové oblasti se středem v klíčovém bodě o délce strany $20s$, kde s označuje měřítko klíčového bodu. Tato oblast je dále rozdělena do menších 16 (4x4) sub-oblastí (každá má tedy velikost $5s$), viz obrázek 4.3.



Obrázek 4.3: Oblast kolem klíčového bodu pro výpočet deskriptoru



Obrázek 4.4: Reprezentace filtru Haarova vlnka. Vlevo je uveden filtr pro výpočet odezvy ve směru x, vpravo ve směru y. Černá část má váhu -1, bíla část +1

Následně je v každé této sub-oblasti rovnoměrně vybráno 5x5 bodů, nad kterými jsou vypočteny odezvy d_x a d_y na Haarovy vlnky (viz obrázek 4.4) o velikosti $2s$ ve směru x a y. Tyto odezvy d_x a d_y jsou dále váhovány Gaussovou funkcí ($\sigma = 3.3s$) se středem v klíčovém bodě ke zvýšení robustnosti vůči geometrickým deformacím. V rámci sub-oblasti jsou poté odezvy sečteny do první části deskriptoru $v = (\sum dx, \sum dy, \sum |dx|, \sum |dy|)$. Každá z 16 sub-oblastí je tedy popsána 4-rozměrným vektorem. Zřetězením těchto vektorů vzniká 64-rozměrný deskriptor popisující klíčový bod a jeho okolí. Velikost deskriptorů přímo ovlivňuje rychlost extrakce i následného porovnávání, což je další důvod vyšší rychlosti detektoru SURF oproti SIFT, jehož deskriptory mají velikost 128.

4.3 Vyhodnocení shody

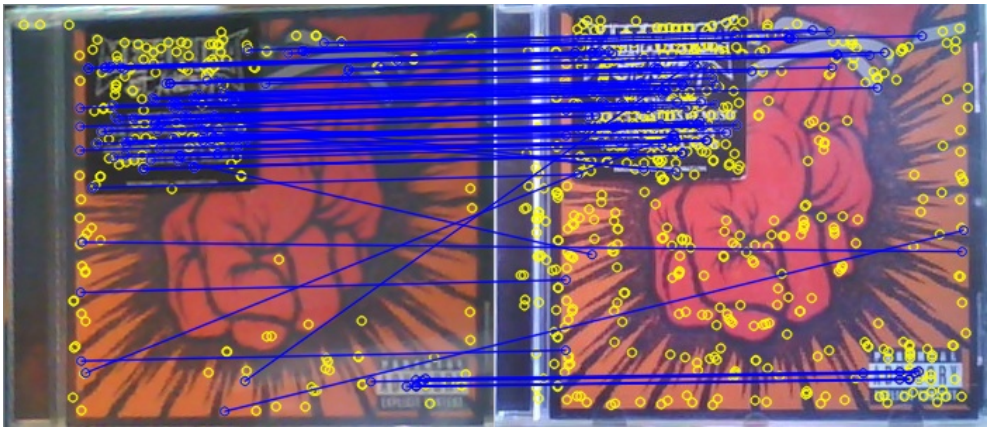
V následujících podkapitolách jsou popsány dvě metriky pro vyhodnocení shody mezi porovnávanými obrazy. Obě tyto metody jsou v aplikaci implementovány a jsou volitelné pomocí argumentu příkazového řádku.

4.3.1 Porovnávání deskriptorů podle vzdálenosti

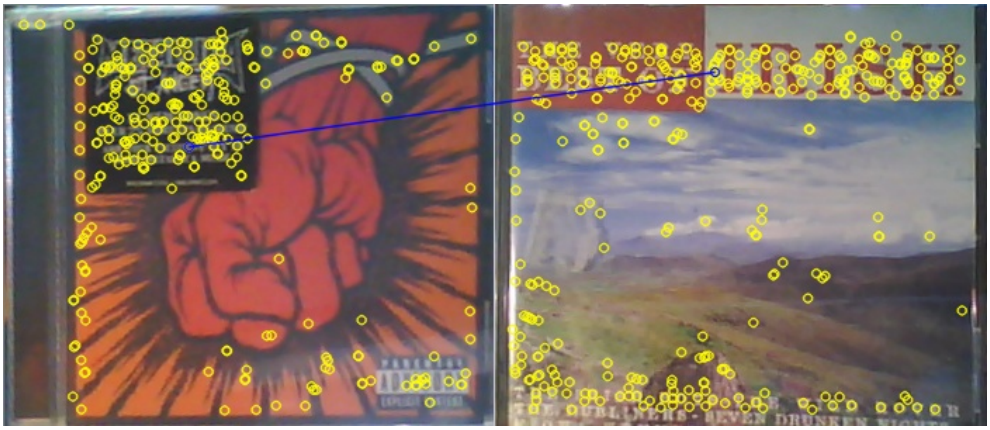
Tato metoda je založena na vyhledání nejbližšího souseda vstupního deskriptoru ve známé databázi. Jako nejbližší soused deskriptoru d_{query} je z databáze vybrán takový deskriptor $d_{database}$, ke kterému má nejmenší vzdálenost. Předpoklad je takový, že pokud vzdálenost těchto deskriptorů je dostatečně malá a leží tedy pod určitou prahovou hodnotou, můžeme prohlásit, že se jedná o deskriptory popisující v obou obrazech stejné klíčové body a jejich okolí. Pokud je mezi vstupním deskriptorem a jeho nejbližším sousedem z databáze vzdálenost větší než daný práh, je vstupní deskriptor neuvažován a zahozen. U každého deskriptoru z databáze víme, ke kterému referenčnímu obrázku alba patří. Proto pokud je některý jeho deskriptor vybrán jako nejbližší soused, můžeme u tohoto obrazu „zahlasovat“ pro shodu. Referenční obraz, který má nejvíce hlasů a tento počet hlasů je dostatečně vysoký a překračuje nastavenou prahovou hodnotu, je z databáze vybrán. Mezi vstupním a tímto vybraným obrazem je takto prohlášena shoda.

Vzhledem k tomu, že SURF deskriptory jsou 64-dimenzionální vektory, je jako metrika vzdálenosti mezi deskriptory použita Euklidovská vzdálenost (nazývaná také Euklidovská metrika) [2], která je pro dva vektory $\vec{p} = (p_1, p_2, ..., p_n)$ a $\vec{q} = (q_1, q_2, ..., q_n)$ v n-rozměrném prostoru dána vztahem:

$$\mathbf{d}(\vec{p}, \vec{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + ... + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$



(a) vlevo – vstupní obrázek, vpravo – referenční obrázek



(b) vlevo – vstupní obrázek, vpravo – referenční obrázek

Obrázek 4.5: Porovnávání deskriptorů podle vzdálenosti

Ukázka vizualizace výsledku tohoto procesu pro dvě alba v databázi je uvedena jako obrázek 4.5. Na těchto obrázcích je vlevo zobrazen vstupní obrázek alba a vpravo referenční obrázek z databáze. Žlutými kolečky jsou vyznačeny nalezené klíčové body. Modrými čarami jsou spojeny takové klíčové body z obou alb v databázi, mezi kterými měly jejich deskriptory nejnižší vzdálenost podle výše popsaného postupu. U prvního obrázku (a) jasně můžeme vidět, že pro stejná alba (avšak jiné obrázky segmentované v různém prostředí) došlo k mnoha shodám[1], zatímco z deskriptorů z druhého alba (b) byl vybrán pouze 1 deskriptor. To je vzhledem k tak nízkému počtu bráno jako falešná detekce.

4.3.2 Kvalita homografie

Homografie už byla zmíněna v podkapitole 3.2.4. V tomto případě je její použití trochu odlišné. Zatímco v předchozím případě byla homografie použitá pro transformaci a vypočítána ze 4 bodů, o kterých jsme s jistotou věděli, že si odpovídají, zde je použita jako metrika shody dvou obrazů. Matice homografie je zde počítána z mnohem více bodů, o kterých ale s jistotou nevíme, zda si v obou obrazech odpovídají. Robustní iterativní metodou RANSAC jsou vyhledány takové body, které tvoří největší množinu korespondencí bodů,

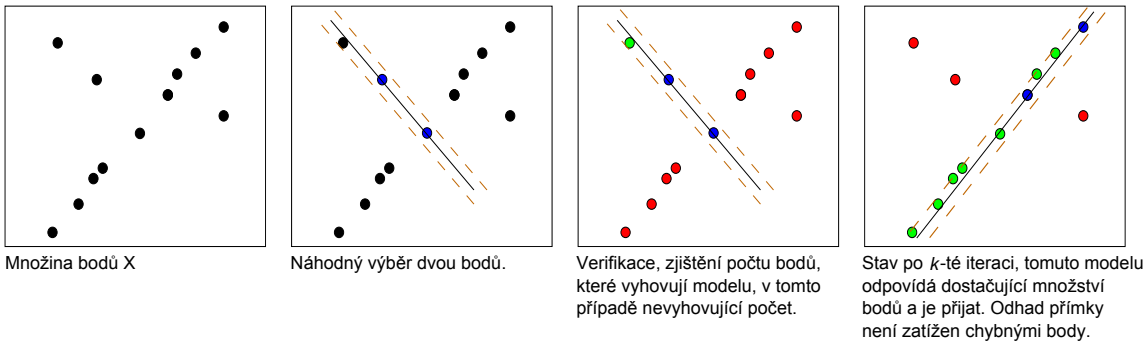
[1] Ačkoliv některé nejsou přesné a nespojují stejné body, tak spojují stejný vzor, který se v obrázku opakuje – deskriptory takového místa jsou pak podobné

které odpovídají jedné homografii. Vyřazeny jsou takové body, které takto určené homografii neodpovídají.

První krok je podobný jako u předchozí metody. Pro každý vstupní deskriptor je nalezen nejbližší soused z databáze deskriptorů. V tomto případě ale neexistuje vzdálenostní práh mezi deskriptory a je tak propojeno mnohem více bodů. Pro obrazy, u kterých je počet shodných deskriptorů větší než určitá prahová hodnota, je počítána matice homografie. Tyto obrazy označíme jako kandidátní obrazy. Pro každý kandidátní obraz je následně počítána homografie metodou RANSAC.

RANSAC (**RA**ndom **SA**mple **C**onsensus) [8] je algoritmus určený pro odhad správných parametrů matematického modelu ze sady naměřených dat. Algoritmus přímo předpokládá, že tyto data kromě správných dat (*inliers*) obsahují i chyby, které do modelu nezapadají (*outliers*). Je založen na náhodném výběru n vzorků z množiny $\mathbf{X}$, n by měl být minimální potřebný počet vzorků pro výpočet modelu. Ze zvoleného vzorku, který je považován za *inliers*, jsou následně vypočteny parametry modelu (hypotéza). Poté dojde ke stanovení počtu vzorků, které tomuto modelu vyhovují a podporují tak původní hypotézu (*consensus set* nebo *support set*). To, jestli vyhovují, je dané určením míry tolerance k chybě e spočítané pomocí chybové funkce (verifikace). Pokud tomuto modelu odpovídá dostatečné množství vzorků (*consensus set* je dostatečně velký), je model přepočítán ze všech přijatých vzorků a proces ukončen. V opačném případě se proces vrací na začátek s další iterací. Po dosažení maximálního počtu iterací algoritmus končí s neúspěchem, protože během těchto iterací nedošlo k nalezení dostatečně vyhovujícího modelu.

Na obrázku 4.6 je pro ukázku znázorněn postup algoritmu pro nalezení bodů z množiny $\mathbf{X}$, které nejlépe formují přímku.



Obrázek 4.6: Příklad RANSAC - odhad přímky z množiny bodů

V našem případě je hledaným modelem homografie, resp. její matice. Z předchozího kroku máme určené dvojice korespondujících bodů pro kandidátní obrazy, které pravděpodobně obsahují chyby (body, které si neodpovídají). Metodou RANSAC je vypočítaná homografie pro každý kandidátní obraz ze 4 bodů tak, aby *consensus set* byl co největší, tedy nalezenou homografii podporovalo co nejvíce ostatních bodů. To je určené prahovou hodnotou chyby zpětné projekce.

Jako shoda obrazu s databází je poté vyhodnocen takový kandidátní obraz, který má nejvíce dvojic korespondujících bodů (*inliers*) po odstranění chyb metodou RANSAC. Pro jeden kandidátní obraz je vstup a výsledek znázorněn na obrázcích 4.7. Zde na obrázku (a) můžeme vidět všechny nalezené klíčové body obou obrazů vyznačené bílými kolečky.

V druhém obrázku (b) jsou zeleně vyznačeny pouze takové body, které si odpovídají v obou obrazech podle nejlepší odhadnuté homografie. Těchto bodů je dostatečný počet, proto obrazy můžeme označit za shodné.

(a) vstupní a referenční obrázek a jejich klíčové body

(b) klíčové body, které si odpovídají

Obrázek 4.7: Homografní korespondence bodů mezi dvěma obrazy metodou RANSAC

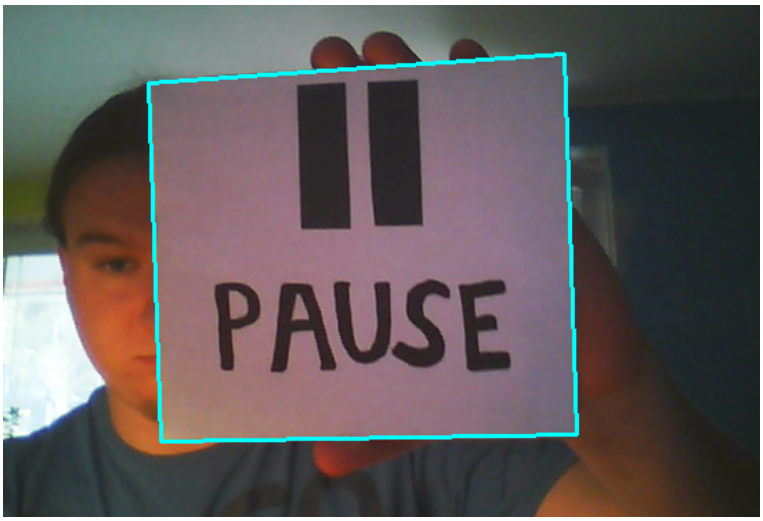
Kapitola 5

Implementace

Aplikace „Sci-Fi“ hudební knihovna je implementována v jazyce `C++` za využití knihovny OpenCV verze 2.3.1. Pro vlastní přehrávání hudebních souborů `.mp3`, `.ogg`, `.wav` a `.wma` je použita audio knihovna irrKlang 1.3.0b [16].

5.1 Popis aplikace

Jak již bylo řečeno v úvodu, cílem této práce je implementovat hudební knihovnu, která bude ovládaná pomocí ukazování krabiček alb a ovládacích kartiček na kameru. Příklad, jak taková kartička může vypadat, je uveden jako obrázek 5.1. Aplikace se skládá z konzolového okna, do kterého jsou vypisovány některé informace o činnosti programu, a hlavního grafického okna, které přehrává scénu snímanou kamerou. Než je snímek vykreslen, je ve spodní části vybaven o úzký řádek informující o stavu přehrávače.



Obrázek 5.1: Příklad ovládací kartičky ve scéně

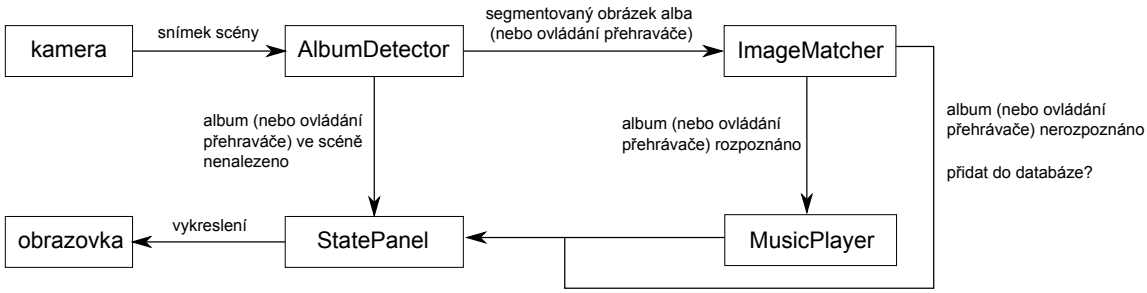
Po spuštění programu je načtena databáze o uložených albech do paměti, je inicializována výchozí systémová kamera a od tohoto okamžiku je aplikace v činnosti. Program se ukončuje klávesou `Escape`. Lze ho také kdykoliv pozastavit (snímání i přehrávání hudby) klávesou `Enter` a opět rozběhnout libovolnou klávesou.

Pro přidání nového hudebního alba je třeba držet krabičku CD před kamerou dostatečně dlouho a stabilně, aby během tohoto času nedošlo k neúspěšné segmentaci (pak by byl proces restartován). Po třech neúspěšných porovnání je další snímání pozastaveno a je nabídnuta možnost přidat nové album. Toto lze potvrdit klávesou `y` nebo odmítnout klávesou `n` (musí být aktivní hlavní okno s přehráváním snímané scény). Při potvrzení je uživatel vyzván, aby napsal do konzole název hudební skupiny, alba a pomocí dialogu vybral cestu, kde jsou uloženy hudební soubory. Před tímto procesem je vidět jaký obrázek byl segmentován a ten také bude uložen jako referenční, je proto vhodné, aby byl dostatečně kvalitní a případně postup opakovat, protože během segmentace nemusel být vlivem okolního prostředí zvolený nejlepší kandidát.

Pokud dojde k úspěšnému rozpoznání hudebního alba nebo ovládací kartičky, další porovnávání se neprovádí, dokud nedojde k neúspěšné segmentaci (nenalezení objektu ve scéně) a resetování tak vnitřního atributu. Je to z důvodu zabránění několikanásobné detekce během krátkého okamžiku ukázání krabičky alba nebo ovládací kartičky na kameru. Pro opakování akce je proto nutné objekt „schovat“ a znovu ukázat.

5.2 Struktura aplikace

Tato část a následující podkapitoly se věnují popisu programových tříd. Diagram činnosti celé aplikace a jejích hlavních procesů je uveden jako obrázek 5.2.



Obrázek 5.2: Diagram procesů programu

5.2.1 Třída AlbumDetector

Třída `AlbumDetector` provádí segmentaci obrazu z kamery v hlavní programové smyčce. Obsahuje jedinou veřejnou metodu `detect`, která jako první parametr bere vstupní snímek z kamery a jako druhý parametr předem vytvořené úložiště pro obraz o velikosti `300x350` pixelů pro případně nalezený obrázek hudebního alba nebo ovládací kartičky (dále už bude uváděno pouze album, protože pro aplikaci jsou album i ovládací kartička reprezentovány stejně). Uvnitř této metody jsou volány funkce z knihovny OpenCV `cv::Canny` pro detekci hran a `cv::HoughLines` pro Houghovu transformaci. Z důvodu optimalizace je uloženo nejvýraznějších 30 čar[1] do vektoru (`std::vector`) struktur `Line`, která obsahuje členy `int r` a `float theta` (úhel je v obloukové míře). Následuje propojení čar do párů podle principu popsaného v podkapitole 3.2.3 a uložení těchto párů do struktur `Pair` se členy `Line line1`

[1] nebo celkový počet, pokud jich je méně

a `Line line2`. Jako prahová hodnota, pod kterou musí být rozdíl úhlů v absolutní hodnotě, je experimentálně zvolena hodnota 0.085 radiánů. Dalším krokem je spojení vhodných párů do struktur `TRectangle` skládající se ze členů `Pair pair1` a `Pair pair2`. Pro každý obdélník jsou následně vypočítány jeho vrcholy, výška a šířka.

Vzhledem k tomu, že nalezených obdélníků je ve scéně obvykle více, je třeba podle jejich vlastností nevhodné vyfiltrovat a zvolit nejvhodnějšího kandidáta. Předem vyřazeny jsou obdélníky, které mají šířku nebo výšku menší než 200 pixelů. Druhým kritériem je poměr stran. Rozměry celé krabičky CD jsou 14cm na šířku a 12cm na výšku, poměr těchto stran je po zaokrouhlení 1.667. Z možných kandidátů je zvolen právě jeden obdélník, který se nejvíce blíží tomuto poměru. Pokud se této hodnotě žádný kandidát neblíží (toto je určeno spodní a vrchní prahovou hodnotou), metoda končí neúspěchem. Pokud je vybrán vhodný obdélník, pomocí funkce `cv::getPerspectiveTransform` je ze souřadnic zdrojových bodů (vrcholy vybraného obdélníku) a cílových bodů (souřadnice rohů parametrem předaného úložiště) vypočítána transformační matice, která je použita při vlastní transformaci. Tato transformace je zprostředkovaná funkcí `cv::warpPerspective`. Po tomto kroku je segmentace kompletní a metoda končí úspěchem.

5.2.2 Třída ImageMatcher

Tato třída obsahuje nástroje na správu (načtení, uložení nebo editaci) databázových souborů (viz podkapitola 5.4) a metody pro vyhledávání obrazů v databázi. Metoda `match` této třídy je volána, pokud předchozí detekce byla úspěšná. Přijímá jeden parametr – obraz alba, segmentovaný třídou `AlbumDetector`. Z tohoto obrazu jsou algoritmem SURF extrahovány klíčové body a vypočítány jejich deskriptory. Pro vlastní porovnání těchto klíčových bodů a deskriptorů třída obsahuje dvě metody – `homographyMatch` (podle kvality homografie, výchozí metoda) a `distanceMatch` (podle vzdálenosti deskriptorů, možnost zvolit argumentem příkazového řádku).

`distanceMatch` končí úspěchem, pokud pro jeden obraz existuje alespoň 35 shodných deskriptorů, tedy takových deskriptorů, jejichž vzdálenost leží pod určitou prahovou hodnotou. Tento práh je nastavený na experimentálně zvolenou hodnotu 0.17. Při úspěchu je jako výsledek vrácen obraz (resp. index alba, ke kterému obraz patří) s nejvíce shodnými deskriptory.

Metoda `homographyMatch` funguje podobně. Po porovnání deskriptorů (tentokrát neexistuje minimální vzdálenostní práh) je počítána matice homografie pro každý obraz, který má více než 35 shod. Výpočet homografie je zajištěn metodou `cv::findHomography`, které se jako parametry předají souřadnice odpovídajících si klíčových bodů. Pokud u těchto kandidátních obrazů je alespoň 35 bodů odpovídající jedné homografii (*inliers*), je obraz vyhodnocen jako shodný a metoda končí úspěchem a je vrácen index obrazu s největším počtem *inliers*. Tato metoda je zvolena jako výchozí z důvodu lepší přesnosti rozpoznávání, ačkoliv je o něco pomalejší.

Vlastní porovnávání deskriptorů je řešeno prostřednictvím knihovny FLANN (Fast Library for Aproximate Nearest Neighbour), která je součástí OpenCV. Tato aproximační metoda vyhledání nejbližšího souseda je zvolena z důvodu rychlosti porovnávání, protože při velkém množství deskriptorů v databázi by přesné porovnávání hrubou silou (tedy porovnávání každého vstupního deskriptoru s každým v databázi) bylo časově náročné a neefektivní.

Jak již bylo uvedeno u konkrétních porovnávacích metod, pokud dojde ke shodě s některým albem z databáze, je vrácen jeho index. Pokud album rozpoznáno není, je vrácen pří-

znak neúspěchu a je inkrementován vnitřní atribut reprezentující počet pokusů o porovnání. Tento atribut je potřebný pro rozhodnutí programu, zda album definitivně nerozeznává, a pro následné nabídnutí uložení alba do databáze. Pokud není album ze scény segmentováno nebo dojde k úspěšnému rozpoznání, je tento atribut resetován.

Mezi další důležité atributy této třídy také patří databáze alb a jejich klíčových bodů a deskriptorů, které jsou po startu programu načteny do paměti. Album je reprezentováno strukturou `Album`, která obsahuje textové informace o názvu alba, hudební skupiny, cestu k referenčnímu obrázku a cestu k adresáři s hudebními soubory. Tyto struktury jsou uloženy ve standardním `C++` vektoru.

V této třídě je rovněž obsažena metoda `addNewAlbum` na přidání nového alba. Po uživatelském vstupu dojde k zapsání nového alba do databázových souborů a uložení referenčního obrázku.

5.2.3 Třída MusicPlayer

Jedná se o zaobalení audio knihovny irrKlang a zajišťuje ovládání přehrávače. Pokud v hlavní programové smyčce dojde k rozpoznání alba, je volána metoda `doPlayerAction`, která bere jako parametr strukturu `Album` vybranou z databáze na základě indexu z předchozího procesu vyhledání. V této metodě se rozhodne, zda se jedná o ovládací kartičku nebo hudební album. Pokud se jedna o ovládací kartičku, porovná se její název a provede se odpovídající akce (volání metod `play`, `stop`, `next` apod.). V případě hudebního alba je sestaven playlist hudebních souborů načtených z adresáře, ke kterému je uložena celá cesta, a je započato přehrávání.

5.2.4 Třída StatePanel

Třída `StatePanel` je v celém procesu posledním článkem a metodou `drawState` zajišťuje vložení stavového řádku do spodní části snímku před samotným vykreslením na obrazovku. Jako atributy obsahuje textové informace o současně přehrávaném albu, název přehrávaného souboru, hlasitost, nastavení smyčky přehrávání a poslední akci programu. Je napojena na třídu `MusicPlayer`, která zajišťuje nastavování těchto atributů při každé změně.

5.3 Databázové soubory

Aplikace pracuje s několika databázovými soubory, které jsou umístěny v adresáři `database`. Tyto soubory jsou v `XML` formátu a jsou spravovány jádrovým modulem `persistance` knihovny OpenCV. Jejich obsah je během spuštění programu načten do paměti.

Nejdůležitějším souborem, který nelze programem vygenerovat a musí být při spuštění aplikace dostupný, je databáze ovládacích kartiček `controls.xml`. Obsahem je název akce, kterou reprezentují a relativní cesta k referenčnímu obrázku. Struktura je následující:

```
<?xml version="1.0"?>
<opencv_storage>
   <control-0>
     <name>"play"</name>
     <imagePath>"control/play.jpg"</imagePath>
   </control-0>
   ...
</opencv_storage>
```

Dalším souborem je vlastní databáze alb `albumDatabase.xml`. Tento soubor obsahuje potřebné informace o uložených albech. Pokud tento soubor neexistuje (první spuštění programu nebo smazání databáze), je automaticky vygenerován s prázdným obsahem. Pokud soubor nějaká data obsahuje, příklad jejich struktury je takovýto:

```
<?xml version="1.0"?>
<opencv_storage>
   <album-0>
     <bandName>"Metallica"</bandName>
     <albumName>"St. Anger"</albumName>
     <imagePath>"album\metallica-st_anger-ref.jpg"</imagePath>
     <musicPath>"D:\mp3\Metallica\2003 - St. Anger"</musicPath>
   </album-0>
   <album-1>
     <bandName>"Wyrd"</bandName>
     <albumName>"Vargtimmen pt. 2"</albumName>
     <imagePath>"album\wyrd-vargtimmen_pt_2-ref.jpg"</imagePath>
     <musicPath>"D:\mp3\Wyrd\2004 - Vargtimmen Pt. II"</musicPath>
   </album-1>
   ...
</opencv_storage>
```

Z důvodu optimalizace a lepší použitelnosti programu jsou pro každý referenční obrázek alba a ovládací kartičky uložena do souboru i data reprezentující jejich klíčové body a deskriptory, protože jinak by se musely s každým spuštěním programu znovu detekovat a spočítat, což by bylo časově náročné. Jedná se o soubory `keypoints_SURF.xml` a `descriptors_SURF.xml`. Po spuštění programu je kontrolována základní synchronizace těchto dvou souborů s databází alb a ovládacích kartiček. Kontroluje se jejich počet záznamů, který musí být stejný jako počet záznamů v databázi alb + počet záznamů v databázi ovládacích kartiček. Pokud počty nesedí, jsou oba soubory `keypoints_SURF.xml` a `descriptors_SURF.xml` automaticky znovu vygenerovány.

V adresáři `database` jsou kromě těchto `XML` souborů umístěny ještě referenční obrázky alb a ovládacích kartiček ve formátu `jpg`, aby se na ně dalo nahlédnout a případně je změnit, a především, aby se z nich daly v případě potřeby znovu vygenerovat výše zmíněné databáze klíčových bodů a jejich deskriptorů.

5.4 Konfigurační soubory

V adresáři `settings` jsou uloženy konfigurační soubory s parametry, které umožňuje OpenCV nastavit pro detektor SURF a knihovnu FLANN. Tyto parametry jsou experimentálně nastaveny na vhodné hodnoty pro dobrou použitelnost z hlediska poměru přesnosti a rychlosti. Jako v předchozích případech, při změně těchto parametrů je nezbytně nutné opět vygenerovat databáze klíčových bodů a deskriptorů.

Kapitola 6

Závěr

V této kapitole jsou shrnuty dosažené výsledky této bakalářské práce. Jsou zde dále uvedeny omezení a známe problémy, které vyplývají z použitých technologií. Jako poslední podkapitola jsou zde nastíněny některé nápady pro možný budoucí vývoje této hudební knihovny.

6.1 Dosažené výsledky

Během řešení této práce se podařilo splnit zadání a implementovat hudební knihovnu ovládanou prostřednictvím počítačového vidění. Bylo nutné nastudovat a použít vhodné algoritmy s důrazem na použitelnost výsledné aplikace.

Jako nejvhodnější postup pro segmentaci krabičky CD ze scény se z jevil přístup založený na segmentaci podle tvaru objektu. Kvalitu segmentace touto metodou i tak mohou nepříznivě ovlivnit vzory v obrázku hudebního alba, které mohou rovněž tvořit obdélníky, případně jiné hrany ve scéně jako například obraz na stěně nebo rohy pokoje. Tyto hrany pak mohou ležet rovnoběžně nebo kolmo k držené krabičce a dostat se tak do možnosti zformovat obdélník. Na obrázku 6.1 můžeme vidět příklad takto ovlivněné segmentace. Ačkoliv je v tomto případě krabička CD v obrazu obsažena a pravděpodobně by stále došlo ke správnému rozpoznání, protože SURF je invariantní vůči změně měřítka, nebyl by ale takto segmentovaný obraz vhodný k použití jako referenční obrázek alba.

Obrázek 6.1: Špatná segmentace vlivem prostředí

Vyhledávání v databázi obrazů je poměrně otevřené téma a neexistují univerzální přístupy. Aby aplikace mohla běžet v různých prostředích, byla na porovnávání obrazů zvolena metoda detekce význačných bodů v obraze a porovnávání jejich deskriptorů, které jsou invariantní vůči změnám osvětlení a mírné deformaci obrazu z důvodu transformace. Jako detektor význačných bodů byl použit detektor SURF, který v současné době poskytuje rozumný poměr mezi robustností a rychlosti detekce. Díky knihovně OpenCV a jejímu rozhraní pro detektory je aplikace navržena tak, aby mohl být detektor jednoduše změněn za jiný, který knihovna OpenCV například budoucnu implementuje a poskytne přesnější nebo rychlejší výsledky.

Celý systém byl úspěšně testován zatím s největším počtem 39 položek v databázi (dohromady hudební alba a ovládací kartičky) s celkovým počtem 10 661 deskriptorů. Časová náročnost detekce klíčových bodů a počítání jejich deskriptorů trvá na průměrném notebooku v závislosti na jejich počtu asi 50-100ms. V této aplikaci není nutné, aby klíčový bod byl invariantní vůči rotaci, proto je detektor nastaven tak, aby nepočítal orientaci klíčového bodu (tzv. *upright* verze SURF, zkráceně U-SURF), což velkou mírou přispívá k rychlosti algoritmu i rozlišovací schopnosti deskriptorů. Vlastní porovnávání deskriptorů trvá díky knihovně FLANN 5-15ms, což vyhovuje požadavkům aplikace.

6.2 Známé problémy a omezení

Ačkoliv byla snaha o nejuniverzálnější řešení, i tak se, vzhledem k použitým technologiím, najdou některá omezení. Jelikož je porovnávání obrazů založeno na detekci význačných bodů, je důležité, aby vůbec takové body byly nalezeny a bylo jich dostatečně mnoho s vysokou mírou opakovatelnosti nalezení. Proto systém neumí rozpoznat alba, jejichž obrázek je poměrně minimalistický (například jednobarevný podklad a malý nápis). Detekce význačných bodů je negativně ovlivněna i špatným osvětlením a tím vzniklým šumem a nekvalitním obrazem. Možnost vylepšení detekce je nastíněno v podkapitole, která se zabývá možnostmi dalšího vývoje.

Dále bych ještě zmínil problém, který je spíše fyzikálního rázu. Je dán povahou světla a plastovou krabičkou CD, na které dochází k zrcadlení zářících objektů ležících naproti držené krabičce CD. Prakticky se to týká především notebooků, které mají zabudovanou webkameru nad displejem v jeho víku. Při horším okolním osvětlení a velkém jasu displeje dojde při ukázaní krabičky CD na takovouto webkameru k odrazu toho, co máme na obrazovce. Ztratí se tak původní obraz a prakticky je na ní pak něco jiného, tak jak je znázorněno na obrázku 6.2, což může velmi výrazně zhoršit až znemožnit správné rozpoznání, především u velmi tmavých obrázků alb. Řešením je zajištění dobrého okolního osvětlení a snížení jasu displeje, případně použití jiné kamery, za kterou neleží stejně směrovaný zdroj světla. Druhým řešením by mohl být i polarizační filtr umístěný před kameru, ale to nebylo odzkoušeno.

Obrázek 6.2: Displej notebooku odražený na krabičce CD

6.3 Možnosti dalšího vývoje

V této podkapitole bych rád uvedl některé myšlenky a nápady pro možný budoucí vývoj „Sci-Fi“ hudební knihovny.

První věcí je zlepšení detekce význačných bodů použitím adaptivního detektoru. Myšlenka spočívá v tom, že pokud dojde k nalezení nedostatečného počtu klíčových bodů, detekce by se provedla opakovaně po upravení některých parametrů detektoru a zvýšením jeho citlivosti.

Další myšlenkou je vylepšení ovládání přehrávače (příkazy play, pause, stop, atd.). Místo ukazování kartiček na kameru by bylo možné ovládání upravit na rozpoznávání gest ruky, což by mohlo být praktičtější řešení, jelikož k tomu člověk nepotřebuje další předměty. Z tohoto důvodu mě napadl i další přístup, který by pro tuto součást opustil počítačové vidění, a to ovládání hlasem.

V neposlední řadě by bylo možné knihovnu vybavit o bohatší uživatelské prostředí a GUI, ve kterém by mohl být obsažen manažer pro správu databázových souborů. Přímo v aplikaci by pak šlo prohlížet uloženou databázi včetně referenčních obrázků a případně ji editovat nebo mazat její položky. Zároveň by tento manažer mohl disponovat i modulem pro přehrávání a převod hudby přímo z CD, pokud ještě není v databázi, do mp3 či jiného kompresního audio formátu.

Poslední myšlenka se týká `XML` databáze deskriptorů. Jedná se o poměrně rozsáhlá data narůstající s každou další položkou v databázi. Pro představu, velikost souboru s deskriptory pro 39 položek v databázi je `11.5MB`. Bylo by proto rozumné uvažovat nad formami komprese a dekomprese tohoto souboru, o které by se starala sama aplikace.

Literatura

[1] Bay, H., Tuytelaars, T. a Gool, L. V. SURF: Speeded Up Robust Features. In Leonardis, A., Bischof, H. a Pinz, A. (ed.). *9th European Conference on Computer Vision*. Graz: Springer, May 2006. S. 404–417. ISBN 3-540-33832-2.

[2] Šonka, M. a Hlaváč, V. *Počítačové vidění*. 1. vyd. Praha: Grada, 1992. ISBN 80-854-2467-3.

[3] Bradski, G. *Domovoská stránka knihovny OpenCV* [online]. 2012, 2012-03-17 [cit. 7. dubna 2012]. Dostupné na: <`http://opencv.willowgarage.com/wiki/`>.

[4] Bradski, G. a Kaehler, A. *Learning OpenCV: Computer Vision with OpenCV Library*. 1. vyd. Sebastopol: O'Reilly Media, 2008. ISBN 978-0-596-51613-0.

[5] Laganière, R. *OpenCV 2 Computer Vision Application Programming Cookbook*. 1. vyd. Brimmingham: Packt Publishing, 2011. Quick Answers to Common Problems. ISBN 978-1-84951-324-1.

[6] Bishop, C. M. *Pattern Recognition and Machine Learning*. 1. vyd. New York: Springer Science Business Media, 2006. ISBN 03-873-1073-8.

[7] Jung, C. R. a Schramm, R. Rectangle Detection based on a Windowed Hough Transform. In Araújo, A. d. A. (ed.). *Proceedings of the Computer Graphics and Image Processing, XVII Brazilian Symposium*. Los Alamitos, Calif.: IEEE Computer Society, Oct 2004. S. 113–120. ISBN 0-7695-2227-0.

[8] Šonka, M., Hlaváč, V. a Boyle, R. *Image processing, analysis, and machine vision*. 3. vyd. Mason, OH: Thomson, 2007. International student ed. ISBN 04-952-4438-4.

[9] Fisher, R., Perkins, S., Walker, A. et al. *Hough transform* [online]. 2003, 2004-03-01 [cit. 12. dubna 2012]. Dostupné na: <`http://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm`>.

[10] Wikipedia Contributors. *Polar coordinate system* [online]. 2012, 2012-04-08 [cit. 12. dubna 2012]. Dostupné na: <`http://en.wikipedia.org/wiki/Polar_coordinate_system`>.

[11] OpenCV documentation contributors. *OpenCV documentation - Feature Detection* [online]. 2010, 2010-11-30 [cit. 13. dubna 2012]. Dostupné na: <`http://opencv.willowgarage.com/documentation/cpp/imgproc_feature_detection.html#HoughLinesP`>.

[12] Wikipedia Contributors. *Line-line intersection* [online]. 2012, 2012-03-2 [cit. 12. dubna 2012]. Dostupné na: <http://en.wikipedia.org/wiki/Line-line_intersection#Mathematics>.

[13] Beneda, M. *Homografie a epipolární geometrie* [online]. 2010, 2010-10-31 [cit. 25. dubna 2012]. Dostupné na: <http://trilobit.fai.utb.cz/homografie-a-epipolarni-geometrie>.

[14] Tuytelaars, T. a Mikolajczyk, K. *Local Invariant Feature Detectors: A Survey.* 1. vyd. Hanover, MA: Now Publishers, 2008. ISBN 16-019-8138-4.

[15] Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision.* Nov 2004, roč. 60, č. 2. S. 91–110. ISSN 0920-5691.

[16] Gebhardt, N. *IrrKlang - an audio library for C++, C# and .NET and high level 3D and 2D sound engine* [online]. 2012, 2012-04-08 [cit. 15. dubna 2012]. Dostupné na: <http://www.ambiera.com/irrklang/>.

Příloha A

Obsah CD

Základní adresářová struktura přiloženého CD a popis adresářů je následující:

- /project/
  - /bin/ – Přeložený program pro platformu Win32 připravený k použití. Obsahuje potřebné knihovny a aplikační soubory ke spuštění.
  - /doc/ – Dokumentace kódu vygenerovaná programem Doxygen.
  - /src/ – Obsahuje Visual Studio 2008 Project, zdrojové texty programu, knihovny a aplikační soubory.
- /latex/ – Tato textová zpráva v pdf včetně zdrojových souborů v LaTeXu.
- /poster/ – Obsahuje elektronickou verzi plakátu prezentující tuto práci.

Příloha B

Manuál k instalaci a použití

B.1 Instalace

Na přiloženém CD je obsažena zkompilovaná aplikace se všemi knihovnami a potřebnými aplikačními soubory připravena k použití na platformě 32-bit Windows. Pro spuštění je nutné překopírovat adresář `project/bin/` na pevný disk a spustit binární soubor `musicLibrary.exe`. Ke správnému běhu je zapotřebí, aby byla k počítači připojena a nainstalována kamera. Pro spuštění programu je také potřeba, aby operační systém obsahoval runtime komponenty Visual C++ knihoven. Standardně by na systému Windows měly být nainstalovány, pokud nejsou, lze je stáhnout a doinstalovat ze stránek Microsoftu na adrese `http://www.microsoft.com/en-us/download/details.aspx?id=29`

Program je možné znovu sestavit ze zdrojových souborů. V adresáři `project/src/` je obsažen Visual Stuido 2008 Project se všemi zdrojovými soubory a knihovnami. Po zkopírování tohoto adresáře na pevný disk a přeložení projektu z Visual Studia (klávesa `F7`) bude výsledný binární soubor uložen v adresáři `project/src/build/`. Aby nově sestavený program `musicLibrary.exe` fungoval, je potřeba do tohoto adresáře nakopírovat potřebné knihovny a soubory (např. z adresáře `project/bin/`).

B.2 Návod k použití

Program `musicLibrary.exe` lze spustit s následujícími parametry:

- `-r` – Znovu sestaví databázi klíčových bodů a deskriptorů podle `albumDatabase.xml` (po vygenerování se program ukončí)
- `-d` – Debugovací mód, do konzole je vypisováno více ladících informací
- `-method <d, h>` – Zvolí metodu pro srovnávání obrazu, možnosti:
  - `d` – Vzdálenost deskriptorů
  - `h` – Kvalita homografie (výchozí)
- `-h` – Vytiskne nápovědu programu do konzole a ukončí program

Princip fungování a ovládání programu je popsán v podkapitole 5.1. Pro ovládání přehrávače jsou připraveny a dodány ovládací kartonové kartičky. Je možnost použít jiné přehráním příslušného referenčního obrázku novým snímkem dané kartičky, poté je potřeba znovu vygenerovat databázi klíčových bodů a deskriptorů (parametr `-r`).