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

# PŘECHODY SCÉN VE VIDEU

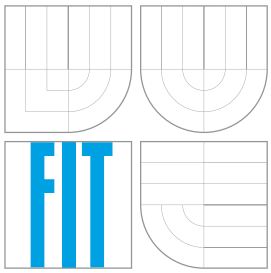
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

LUKÁŠ KLICNAR

BRNO 2010

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

# PŘECHODY SCÉN VE VIDEU

SHOT BOUNDARY DETECTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — LUKÁŠ KLICNAR
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÍTĚZSLAV BERAN
SUPERVISOR

BRNO 2010

# Abstrakt

Detekce přechodů ve videu je proces automatického nalezení hranic mezi jednotlivými scénami. Tato práce se zabývá převážně detekcí střihů, postupné přechody jsou ale rovněž uvažovány. Vysvětleny jsou základní pojmy z této oblasti a stručně představeny doposud používané metody. Stěžejní částí je návrh a implementace detektoru přechodů. Ten je založen na kombinaci dvou přístupů. Prvním je porovnávání barevných histogramů sousedních snímků. Druhý, méně tradiční, je založen na sledování výrazných bodů ve videu. Analýza průběhu těchto příznaků probíhá pomocí odhadu jeho derivace. Systém byl otestován na vlastní sadě ručně anotovaných dat. Ukázalo se, že oba příznaky jsou pro detekci přechodů vhodné. Detektor byl schopný nalézt většinu střihů při zachování dobré přesnosti. Prokázala se schopnost detekovat i některé postupné přechody.



# Abstract

Shot boundary detection is a process of automatically finding the boundaries between shots in a video. This work primarily deals with detection of hard-cuts, but gradual transitions are also considered. At first, fundamental terms of this field are explained, commonly used methods are shortly described. The main part of this work is design and implementation of system for shot boundary detection based on combination of two methods. The first one is comparison of color histograms for adjacent frames. Second approach is based on visual feature tracking. The analysis of behaviour of those features is done by estimating their first derivatives. Proposed system was tested on small, manually annotated set of test data, which showed that both features are suitable for this task. Detector proved its ability to find hard-cuts with good precision. It was also able to detect some gradual transitions.



# Klíčová slova

Přechody scén ve videu, analýza videa, segmentace videa, detekce střihů, barevný histogram, sledování bodů, Harrisův detektor rohů, KLT tracker

# Keywords

Shot boundary detection, video analysis, video segmentation, hard-cut detection, color histogram, feature tracking, Harris corner detector, KLT tracker.

# Citace

Lukáš Klicnar: Přechody scén ve videu, bakalářská práce, Brno, FIT VUT v Brně, 2010

# Přechody scén ve videu

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval poctivě a samostatně pod vedením pana Ing. Vítězslava Berana. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Lukáš Klicnar
10. května 2010


## Poděkování

Touto cestou děkuji vedoucímu bakalářské práce panu Ing. Vítězslavu Beranovi za ochotu, velmi užitečnou metodickou pomoc, cenné rady a podnětné připomínky, které mi pomohly při řešení této práce.

# Obsah

# Kapitola 1

# Úvod

Detekce přechodů scén ve videu je proces, který automaticky nalezne rozhraní mezi záběry ve videu. Ty jsou považovány za elementární jednotky pro členění videosekvencí, správné rozdělení na ně je tedy klíčové při jakékoliv práci s videem. Tato činnost obvykle tvoří první krok při zpracování upravovaného videa, jde tedy o druh předzpracování, které poskytuje základ pro další algoritmy pracující na vyšší úrovni abstrakce. Na správné detekci závisí mnoho systémů, je důležitá například pro automatickou indexaci dat pro multimediální databáze, detekci klíčových snímků při kompresi videa a další různorodé činnosti. Běžně se také používá v programech pro střih a úpravu videa.

Přechody ve videu nejsou pouze střihy, ale existuje velké množství různých typů postupných přechodů. Odlišné druhy přechodů se navíc používají v závislosti na kontextu ve videu. Příkladem může být postupné zatmění nebo prolnutí, které obvykle vyjadřuje skok v čase nebo místě. Přechody tak do jisté míry nesou určitý význam, čehož by se mohlo využít při strojovém určování sémantiky nebo rovněž klasifikaci různých žánrů.

Vývoj metod pro detekci přechodů ve videu probíhá již poměrně dlouho. Mnoho vědců a výzkumných skupin představilo velké množství různorodých způsobů. Větší část prací v této oblasti se zaměřuje zejména na detekci střihů, která je méně náročná, než nalezení postupných přechodů. Ovšem i pro ně byla publikována řada přístupů. Současným trendem se zdá být využívání strojového učení, které se na problém detekce přechodů dívá jako na úlohu klasifikace. Prezentované přístupy jsou obvykle testovány na vlastních datech, tudíž se výsledky různých metod nedají dobře porovnávat. Tento problém vyřešila konference TRECVID, která sjednocuje testování projektů různých týmů na společných testovacích datech.

Mým prvotním úkolem je seznámit se s doposud používanými přístupy, provést jejich velmi stručnou sumarizaci a rovněž sjednotit definice základních pojmů, které nejsou úplně jednoznačné. Hlavním cílem této práce však je navrhnout a implementovat vlastní detektor přechodů. Měl by zahrnovat detekci obecně na základě více příznaků získaných z obrazu, jednak pro každý z nich samostatně a rovněž umožnit jejich současnou kombinaci. Ta by teoreticky měla poskytovat lepší výsledné vlastnosti. Jeden příznak je klasicky využívaný barevný histogram, jako druhý bude představen poměrně netradiční způsob založený na sledování výrazných bodů v obraze. Navržený systém bude otestován na několika sadách vlastních testovacích dat.

Kapitola 2 se zabývá popisem základních pojmů a shrnuje nejdůležitější používané metody detekce přechodů ve videu. Diskutována je rovněž jednoznačnost představených definic. V další kapitole (3) popisuji, jakým způsobem se získávají jednotlivé příznaky z videa, které budou použity pro navržený detektor. Představen je barevný histogram a metody

získání a sledování výrazných bodů v obraze. Kapitola 4 obsahuje návrh samotného systému pro detekci přechodů záběrů ve videu. Ten je stručně představen pomocí blokového schématu a následuje popis způsobů realizace jednotlivých jeho částí. Obecný popis implementace tohoto detektoru se nachází v kapitole 5. Poslední, závěrečnou kapitolou (6) je testování a vyhodnocení různých aspektů představené metody. Nejprve jsou popsány zjišťované parametry, testovací data stručně charakterizována a stěžejní částí jsou výsledky provedených testů.

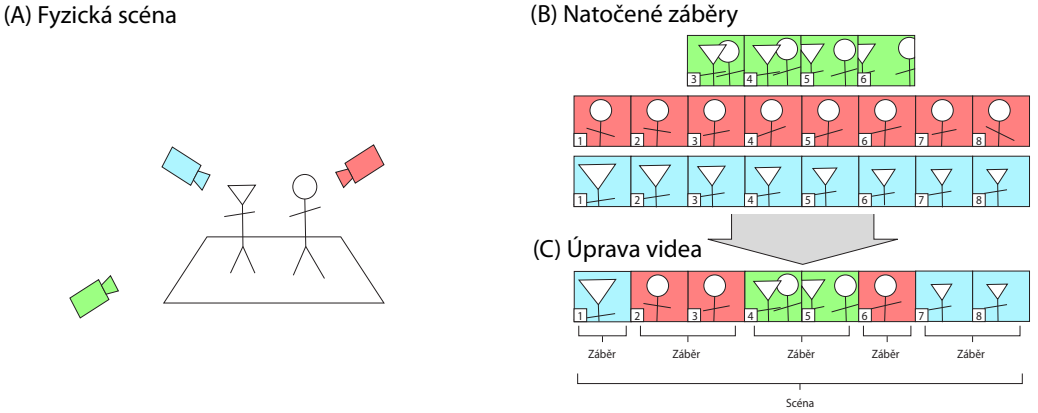
# Kapitola 2

# Přechody scén ve videu

Chceme-li se zabývat přechody scén ve videu, musíme nejprve definovat, co vlastně pojmy jako video, scéna, přechod, a podobně, znamenají. Jak uvidíme, některé z definic nejsou úplně jednoduché, ačkoliv se to na první pohled může zdát. V této kapitole jsou také diskutovány některé speciální situace a rovněž jsou stručně představeny nejčastější, doposud používané přístupy detekce přechodů.

## 2.1 Základní pojmy

Každé video sestává z posloupnosti statických obrázků (snímků), které jsou postupně promítány v pravidelné rychlosti za sebou, obvykle frekvencí okolo 25 snímků za sekundu. Důležité je zejména zabývat se náležitostmi pořízením takovéto sekvence. Na obrázku 2.1 vidíme schematické znázornění tohoto procesu. Definice obsažené v této kapitole jsou volně převzaty z [1], přičemž tento text je dále rozšiřuje.




Obrázek 2.1: Znázornění procesu tvorby videa. Fyzická scéna je nasnímána kamerami, tím vznikne sada záběrů. Ty jsou následně úpravou spojeny do jedné scény. Vytvořeno převzetím a úpravou obrázku z [3].

Mějme **fyzickou scénu**, to je místo, kde probíhá natáčení. Toto místo může být snímáno několika kamerami, každá z nich pořídí množinu záběrů. Pro upřesnění dodejme, že pro tuto práci je důležitý pouze obraz, nikoliv zvuk, proto nebude v popisu dále zmiňován.

**Záběr** je nepřerušená posloupnost snímků natočených jednou kamerou. Nutnou vlastností každého záběru je časová spojitost, tedy že nahrávání kamery bylo po celou dobu aktivní, a rovněž spojitost prostorová. To znamená, že se děj odehrával na jednom místě, jeho rozsah je však relativní. Tato vlastnost je pouze doplňující, protože vyplývá z první – je-li splněna časová spojitost, pak je zaznamenán veškerý pohyb kamery, tedy nemůže dojít ke změně fyzické scény.

Natočené záběry jsou pak upravovány, to obvykle probíhá ve střižně. Při této činnosti dochází jednak ke změně obrazových dat (základní velmi vhodnou úpravou je například ekvalizace histogramů jednotlivých snímků), ale zejména ke spojování záběrů, čímž jsou ve výsledném videu vytvořeny přechody. Ty vznikají buď implicitně, kdy jsou záběry prostě umístěny za sebe, nebo mohou být explicitně vykresleny, i přes více snímků. **Přechod** tedy definujeme jako rozhraní mezi dvěma záběry.

Sekvenci záběrů nazveme **scéna**, pokud vztahy mezi nimi splňují časovou (děj v záběrech přibližně časově navazuje) a prostorovou spojitost (stejné jako u definice záběru). Detekce scén pak záleží na zkušenostech, intuici a vnímání pozorovatele, protože visuální rozdíly mezi scénami nejsou nijak konkrétně definované. Více scén může být spojeno do jedné **videosekvence**, které pak známe jako filmy, klipy a podobně. Pokud se dále v této práci budu zmiňovat o přechodech mezi scénami, je to ekvivalentní s přechody mezi záběry a naopak.

Rozdělení videa na skupiny menších, sémanticky souvisejících celků, se nazývá **segmentace**. Ta se obvykle provádí nalezením vzájemných přechodů mezi těmito celky. Skupiny záběrů patřící do jedné scény však lze strojově těžko oddělit, proto se detekce nezaměřuje na přechody mezi scénami, ale probíhá segmentace na jednotlivé záběry. Nad těmi pak může být spuštěn jiný algoritmus, který je zpětně seskupí do scén.

## 2.2 Rozdělení přechodů

Během již poměrně dlouhé doby produkce videí byla vyvinuta a je používána řada různých typů přechodů. Jejich základní rozdělení se provádí podle doby trvání přechodu (založeno na [5]):

- **Střihy** (hard-cuts) – doba trvání je nulová, jde o skokový přechod mezi dvěma snímky.
- **Postupné přechody** (gradual transitions) – ovlivňují určité množství snímků, které tvoří dobu jejich trvání.

Nejčastějším přechodem je právě **střih** (obr. 2.2). Filmaři jej obvykle používají pro oddělení záběrů jedné scény, ve filmové a televizní produkci tvoří obvykle 80 – 90 % přechodů ([18]). Důvodem jeho častého používání je jednak to, že je přirozený, nejjednodušší na realizaci, a dále rovněž fakt, že velmi dobře prospívá „čistotě“ videa a jeho časté opakování při sledování neruší. Technicky se jedná vlastně o konkatenaci dvou záběrů, kdy po posledním snímku záběru $S_A$ ihned následuje první snímek záběru $S_B$ (převzato z [10], ke střihu dochází v čase $t_{cut}$, funkce $u(t)$ je jednotkový skok):

$$S_{cut}(t) = (1 - u(t - t_{cut}))\cdot S_A(t) + u(t - t_{cut})\cdot S_B(t) \tag{2.1}$$

Postupné přechody se ve filmové praxi obvykle používají pro oddělení scén, tedy když je děj rozdělen v časové nebo prostorové oblasti. Zatímco střih je pouze jediný typ přechodu, mezi postupné řadíme velmi mnoho různých přechodů. K nejdůležitějším patří (podle [10]):

Obrázek 2.2: Ukázka videa obsahujícího střih. Červenou čarou je vyznačena jeho pozice.

- Prolnutí (dissolve)
- Zatmění (fade)
- Efektové přechody

**Zatmění** (obr. 2.3) vznikne postupnou změnou intenzity jasu (případně i barevnosti) mezi snímky scény tak, že dochází ke ztmavení až do úplné černé (fade out). Tento děj může probíhat i v opačném pořadí (fade in). Matematický popis je následující ([10]):

$$S_{fade}(t) = f(t) \cdot S_A(t), \quad t \in <0;\ \mathrm{T}> \tag{2.2}$$

V předchozím vzorci je $T$ doba trvání přechodu. $f(t)$ charakteristická funkce zatmění, která je pro „fade in“ neklesající a její hodnoty se mění od 0 pro $t = 0$ do 1 pro $t = T$, naopak pro „fade out“ je nerostoucí a průběh hodnot je od 1 do 0. Tato funkce nemusí být nutně lineární. Po zatmění obvykle následuje opačný (fade out/in) přechod pro druhou scénu.

Obrázek 2.3: Ukázka přechodu typu „fade out“, který je ihned následován typem „fade in“. Oba takto za sebou se souhrně nazývají „crossfade“.

**Prolnutí** (obr. 2.4) je vytvořeno jako překrytí dvou záběrů, kde se postupně mění míra jejich průhlednosti – první záběr se ztrácí a druhý se naopak postupně objevuje([10]):

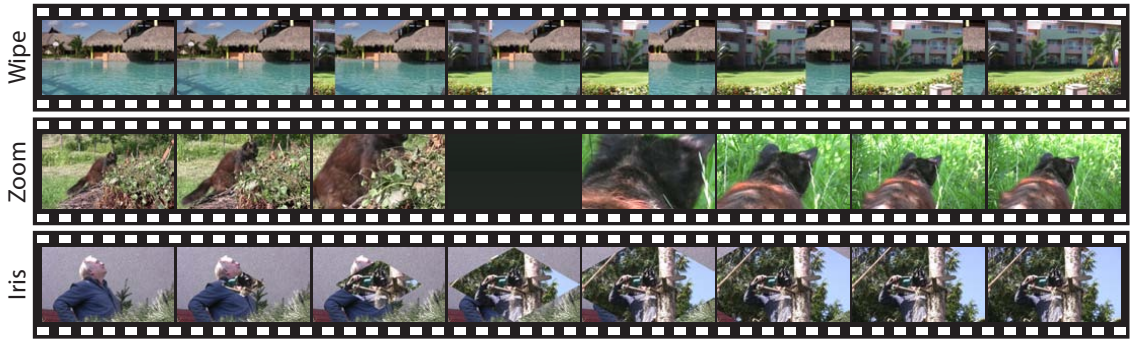
$$S_{dissolve}(t) = f(t) \cdot S_A(t) + (1 - f(t)) \cdot S_B(t), \quad t \in <0;\ \mathrm{T}> \tag{2.3}$$

Funkce $f(t)$ udává míru vlivu každého záběru na výsledek. Má stejné vlastnosti jako $f(t)$ popsaná pro „fade out“, opět může být i nelineární.

Obrázek 2.4: Ukázka přechodu typu prolnutí.

Mezi **efektové přechody** lze zařadit všechny ostatní přechody, které nespadají do dosud popsaných kategorií. Při nich se video může různě posouvat, otáčet, deformovat, často

jsou do něj vloženy i počítačem vykreslené objekty a podobně. Ukázka několika takových přechodů je na obr. 2.5.




Obrázek 2.5: Příklady dalších přechodů. Jejich názvy se mohou lišit v závislosti na programu, ve kterém byly vloženy.

Další zajímavé rozdělení přechodů představil R. Lienhart v [10]. Ten provedl rozdělení do čtyř tříd podle oblasti, ve které dochází ke změně záběrů při přechodu:

1. **Identita** – žádný ze záběrů není změněn a nejsou přidány žádné snímky. Do této třídy spadají pouze střihy.
2. **Prostorová** – jsou provedeny prostorové transformace obrazu. To je v případě přechodů jako je efekt otočení stránky (page turn), posun (slide) a podobně.
3. **Chromatická** – mění se jas či barevnost záběrů. Příkladem může být prolnutí nebo zatmění.
4. **Prostorově-chromatická** – dochází ke změnám v obou těchto oblastech zároveň. Sem patří tzv. *morphing*, což je postupná transformace jednoho objektu na druhý.

## 2.3 Jednoznačnost přechodů

Neupravené video obsahuje pouze střihy, které přesně oddělují jednotlivé záběry. Editací videa do něj však můžeme vložit místa, u kterých nelze jednoznačně rozhodnout, zda je za přechod považovat.

Typickým příkladem je tzv. obraz v obraze. Jedná se o situaci, kdy v jedné scéně je vloženo okno, ve kterém běží zcela jiná videosekvence. To je běžně používaná technika například ve zpravodajství, kdy se za moderátorem vyskytuje několik obrazovek, na kterých běží další, zcela jiné záběry (ukázka na obr. 2.6). Celkový obraz, který vidíme, budeme považovat za jednu videosekvenci.

Zmiňovaná nejednoznačnost nastává, pokud dojde ve videu na některé z obrazovek k přechodu. Ten zcela jistě nastal, ale jedná se o přechod v celé videosekvenci, když uvážíme, že záběr s moderátorem je stále nepřerušený? To právě nelze jednoznačně rozhodnout. Relativní velikost obrazovek vůči moderátorovi je poměrně malá, tudíž bychom intuitivně přechod ignorovali, ale takto bychom zcela jistě nemohli uvažovat, pokud by obě videa zabírala plochu stejnou.


Obrázek 2.6: Snímek z vysílání zpravodajství, které lze považovat za obraz v obraze. Dílčí videosekvence se vyskytují na obrazovkách v horní části.

V dostupných článcích (například [6] nebo [18]) se obvykle popsaná situace za přechod nepovažuje, ačkoliv ji mnoho detektorů pak chybně jako přechod vyhodnotí. Částečně uspokojujícím řešením je obraz v obraze detekovat a v takovém případě celkově detekci zablokovat. Lepším řešením může být vyhodnocovat každou dílčí videosekvenci samostatně, což ale klade podstatně vyšší nároky na návrh a realizaci detektoru.

Nejednoznačnost dále mohou způsobit i různé efekty do videa vložené. U některých typů přechodů může být obtížné rovněž určit jeho začátek a konec.

## 2.4 Používané přístupy

Detekcí přechodů ve videu se již zabírali někteří lidé a výzkumné skupiny, bylo použito mnoho různých přístupů. V zásadě lze princip detekce shrnout do dvou hlavních kroků, které jsou pro většinu vyvinutých metod typické:

1. Snímky videosekvence jsou popsány charakteristickými příznaky
2. Detekce přechodů probíhá vyhodnocením průběhu příznaků získaných v kroku č. 1

Příznaků popisujících obraz existuje mnoho, ne všechny se však hodí pro účel detekce přechodů. Nejčastějším přístupem je analýza podobnosti, respektive spíše nepodobnosti, snímků v čase. Ty bývají srovnávány zejména pomocí barevných histogramů (například [11]). To funguje poměrně dobře, výhodou je také relativně (oproti některým jiným metodám) nenáročný výpočet. Jiný postup srovnává spektra obrazů transformovaných diskrétní kosinovou transformací (DCT, [9]).

Další používanou metodou je hledání hran v obraze ([21]). Ty bývají popsány buď počtem bodů, případně je uvažován i jejich tvar. Pokud se hrany mezi snímky výrazně změní, může se jednat o přechod. Jednoduchým příznakem je prostý rozdíl hodnot pixelů po sobě jdoucích snímků ([16]). Ten je ale poměrně netolerantní k pohybu ve videu, proto bývá doplněn o některou z metod pro jeho kompenzaci. Dále může být použit průběh průměrné hodnoty jasu obrazu, spíše však v konjunkci s některým z předchozích přístupů. Ostatně kombinace více těchto metod je poměrně častá.

Analýza průběhu těchto příznaků bývá prováděna pomocí prahování. Nad klidovou hodnotu(tj. v době mezi přechody) je nastaven práh, při přechodu se očekává jeho překročení. Hodnota tohoto prahu velmi často nebývá pevná, ale je určována adaptivně (například [15]). Některé přístupy jsou určeny pro detekci všech typů přechodů, jiné používají odlišné algoritmy pro různé přechody.

Poměrně moderním přístupem je využití strojového učení ([12]). V takovém případě je natrénován klasifikátor (často se používá například SVM – Support Vector Machine), který používá sadu jednoduchých příznaků pro každý snímek z krátké sekvence. Ty mohou být jednoduše vypočteny, proto je detekce rychlá. Další výhodou je, že klasifikátor může rozlišovat i typy nalezených přechodů.

# Kapitola 3

# Získání příznaků z obrazu

Příznaky slouží obecně k popisu různých objektů, v našem případě se jedná o obrazy – jednotlivé snímky videosekvence. Zároveň mají za úkol popis zjednodušit, respektive abstrahovat od nepodstatných částí a charakterizovat pouze danou vlastnost, která je pro nás důležitá. Můžeme je rozdělit například na:

- **Globální** – do výpočtu je zahrnut celý obraz
- **Lokální** – výpočet probíhá pouze nad částí obrazu, danou například maskou

Jsou možná i různá jiná dělení, pro naše účely postačí výše uvedené. Další důležitou vlastností je invariance příznaků vůči operacím, jako jsou posun, rotace, změna velikosti a rovněž například změna osvětlení scény, pozice kamery a podobně.

Různých obrazových příznaků existuje velké množství, v této kapitole budou představeny pouze dva – barevný histogram a příznak založený na sledování výrazných bodů v obraze. Ty budou následně použity pro realizaci samotného detektoru.

## 3.1 Barevný histogram

Prvním použitým příznakem je barevný histogram, jeho vysvětlení začneme od klasického histogramu a postupně jej zobecníme. Informace obsažené v této kapitole jsou volně převzaty z [17] a [22].

Histogram obecně je prostředkem pro statistický popis rozdělení sady dat do stanoveného počtu tříd. Vysvětlení provedeme nejprve na histogramu jednorozměrném: Základem je rozdělení intervalu, ve kterém se vyskytují zkoumaná data, do $N$ tříd, ty musí být navíc navzájem disjunktní. Pro každou třídu je pak z této sady zjištěn počet reprezentantů, které do ní patří. Výsledek se obvykle znázorňuje jako sloupcový graf.

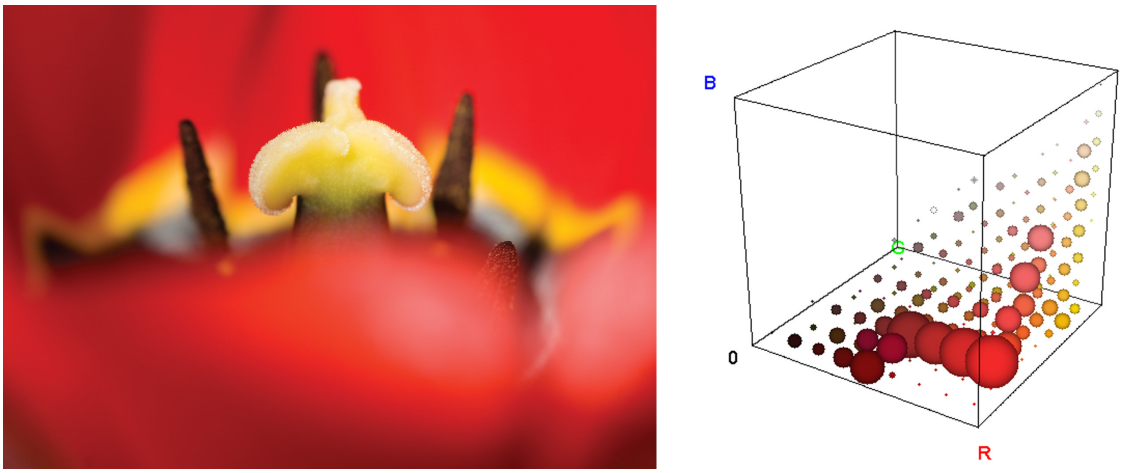
Pokud máme obraz daný jedním 8bitovým kanálem ve stupních šedi, hodnoty jednotlivých bodů jsou v rozsahu 0 – 255. Při rozdělení na $N = 4$ díly dostáváme 4 intervaly (0 – 63, 64 – 127, 128 – 191, 192 – 255). Hodnotám jednotlivých dílů pak jednoduše odpovídá počet obrazových bodů (pixelů), které mají jasovou úroveň patřící do příslušného rozsahu.

Barevný obraz se skládá (obvykle) ze třech kanálů, nejčastěji RGB, tedy červená, zelená a modrá. Kombinací těchto složek pak tvoříme různé barvy. Protože je kanálů více, můžeme sestrojit histogram pro každý samostatně – to běžně používají například fotografové ke kontrole, zda v některém kanálu nedošlo k přeexpozici. Pro účely detekce přechodů je však vhodnějším řešením počítat se všemi kanály zároveň – použitím tzv. *barevného histogramu*. Ten vznikne rozšířením jednorozměrného histogramu na více dimenzí.

| | | Červená | | | |
|---|---|---|---|---|---|
| | | **0 − 63** | **64 − 127** | **128 − 191** | **192 − 255** |
| **Modrá** | **0 − 63** | $h_{0,0}$ | $h_{0,1}$ | $h_{0,2}$ | $h_{0,3}$ |
| | **64 − 127** | $h_{1,0}$ | $h_{1,1}$ | $h_{1,2}$ | $h_{1,3}$ |
| | **128 − 191** | $h_{2,0}$ | $h_{2,0}$ | $h_{2,0}$ | $h_{2,0}$ |
| | **192 − 255** | $h_{3,0}$ | $h_{3,1}$ | $h_{3,2}$ | $h_{3,3}$ |

Tabulka 3.1: Ukázka reprezentace dvourozměrného histogramu tabulkou. Použito rozdělení na $N = 4$ díly.

Pokud použijeme pouze dva kanály, dostaneme dvourozměrný histogram. Ten lze reprezentovat tabulkou, ukázku vidíme na obr. 3.1. Pro určení příslušnosti pixelu do dané kategorie pak ověřujeme podmínky dvě, hodnota bodu v každém z jeho dvou kanálů musí spadat do příslušného intervalu (daného pozicí buňky v tabulce). Analogicky je možné provést rozšíření na tři dimenze, místo dvourozměrné tabulky tak dostaneme třírozměrnou krychli. Tu však nelze na dvourozměrný papír jednoznačně zobrazit, možná ukázka je na obr. 3.1.



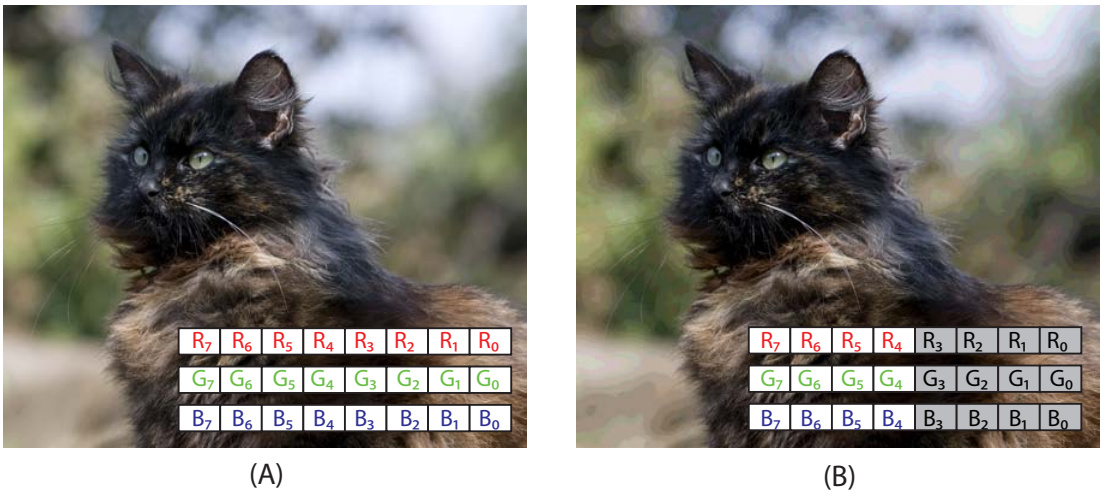

Obrázek 3.1: Znázornění barevného histogramu vypočteného z obrázku vlevo. Průměr koule je úměrný počtu bodů spadajících do dané oblasti. Vytvořeno pomocí 3D Color Inspectoru programu ImageJ.

Barevný histogram tedy poskytuje informace o rozložení barev v obraze, abstrahující od prostorových vlastností (tvaru) objektů v něm přítomných. To by mohlo být překážkou například pro jejich klasifikaci, ovšem pro detekci přechodů se jedná naopak o výhodu. Počet dílů, na které je barevný prostor rozdělen, musí být dostatečně malý, aby se obraz vhodně zobecnil, ale zároveň ne příliš, aby se neztratila schopnost rozlišit odlišné snímky.

Zobecnění obrazu barevným histogramem vlastně odpovídá kvantizaci na nižší počet hladin (viz [11]). Barevný obraz používá obvykle hloubku 24 bitů, tedy každý kanál je reprezentován 8 bity. Tato kvantizace může být provedena právě zahozením méně významných bitů (obr. 3.2).

Barevný histogram může být vypočten pro jakýkoliv barevný prostor. Nejčastěji je vstupní obraz v prostoru RGB, převod do jiného je samozřejmě možný, jedná se však o operaci navíc. Otázkou dalšího zkoumání je, jaký barevný prostor je pro detekci přechodů nejvhodnější, případně má-li vůbec použitý prostor (výraznější) vliv. Tato otázka je řešena v kapitole 6.5.

Z hlediska klasifikace příznaků budeme barevný histogram považovat za globální, ačkoliv může být spočítán i pouze pro část obrazu jako lokální. Je invariantní vůči translaci a rotaci (pro malé změny).




Obrázek 3.2: Kvantizace obrazu pomocí zahození méně významných bitů. Obrázek (B) vznikl ponecháním pouze čtyř nejvýznamnějších bitů z každého bodu obrázku (A).

## 3.2 Sledování bodů

Každá scéna (záběr) je charakteristická množstvím bodů. Ty se v ní po dobu jejího trvání vyskytují, případně jich malá část může zmizet, například pohybem kamery do strany, nebo když se pohybující se objekt schová za překážku a podobně. Pokud však dojde k přechodu na jinou scénu, tyto body zmizí všechny. To platí v případě střihu, u postupného přechodu se ztrácejí postupně, což může činit jejich projev méně výrazným. Budeme-li tyto body sledovat, pak můžeme detekovat okamžik jejich zmizení – tedy vlastně místo ve videu, kde k přechodu došlo. Pro samotné sledování potřebujeme zajistit tyto činnosti:

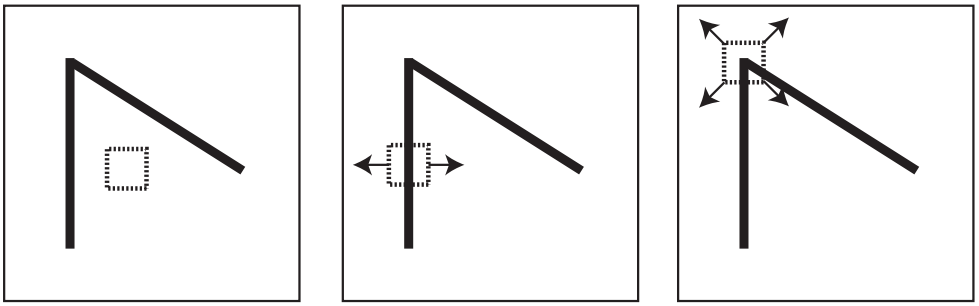
1. Získat množinu bodů, které chceme sledovat
2. Zajistit jejich opětovné nalezení na dalších snímcích

## 3.3 Harrisův detektor

Jako sledované body nemůžeme využít všechny přítomné v obraze. Jednak jejich velké množství by způsobilo vysokou výpočetní náročnost, ale zejména by zahrnutí všech bodů nebylo účelné. Vhodné body pro sledování totiž musí zajistit podmínku provedení činnosti č. 2, tedy musí být snadno k nalezení na dalším snímku, který může být mírně posunutý,

pootočený a podobně. To rozhodně nesplňuje každý, příkladem může být libovolný bod uvnitř jednobarevné plochy, který těžko rozlišíme od ostatních. Vhodné body jsou takové, které se výrazně liší od svého okolí. To splňují tzv. rohy, které nám může poskytnout právě Harrisův detektor.

Myšlenka je taková, že pokud po obraze posouváme malé čtvercové okno, pak změna intenzity bodů pod ním by měla být při posunu ve všech směrech velká. Podle směrů, ve kterých dochází k této (výrazné) změně, pak můžeme rozlišit několik typů oblastí v obraze (dle [7]). Ilustrace je na obr. 3.3.




Obrázek 3.3: Různé druhy oblastí v obraze. Šipkami je vyznačeno, ve kterých směrech dochází při posunu okna k výrazné změně intenzity. U ploch (A) není změna v žádném směru, u hran (B) v jednom a rohy (C) tuto změnu vyvolávají ve všech směrech. Ilustrace je vytvořena podle [7].

Následující vzorce jsou převzaty z [7]. Intenzitu změny při posunu o [u, v] (ve vodorovném a svislém směru) můžeme vyjádřit:

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)] \tag{3.1}$$

$w(x,y)$ je funkce okna, která má hodnotu 1 uvnitř a 0 mimo něj. Pro malé hodnoty posunu [u, v] můžeme intenzitu změny nahradit bilineární aproximací:

$$E(u,v) \simeq \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \tag{3.2}$$

$M$ je matice 2x2 složená z derivací obrazu podle souřadnic:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x Iy & I_y^2 \end{bmatrix} \tag{3.3}$$

Důležitá jsou vlastní čísla $\lambda_1, \lambda_2$ matice M, určují totiž intenzitu změny v jednotlivých směrech. Dále je z nich vytvořena funkce $R$:

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \tag{3.4}$$

$k$ je konstanta, jejíž hodnota se volí empiricky, obvykle v rozsahu 0.04 – 0.06. Ovlivňuje „kvalitu“ nalezených bodů, myšleno jako jejich výraznost. Při hledání rohů se vypočte $R$

pro celý obraz, jednotlivým nalezeným bodům pak odpovídají lokální maxima větší než zadaný práh, která se hledají v okolí určené velikosti. Ukázka nalezených bodů je na obr. 3.4.

Obrázek 3.4: Body nalezené pomocí Harrisova detektoru.

## 3.4 KLT tracker

Tracker slouží k nalezení pozice každého sledovaného bodu v aktuálním snímku, tím získáme jeho trajektorii, po které se v průběhu videosekvence pohyboval. Tyto změny popisujeme pomocí optického toku, který pro každý bod udává dvourozměrný vektor rychlosti obsahující přírůstky souřadnic.

Cílem je pro každý sledovaný bod najít vektor posunutí $\begin{bmatrix} u & v \end{bmatrix}$ tak, aby se bod a jeho (malé) okolí posunulo nad oblast v novém snímku, kde se nachází sledovaný bod – tedy kde je lokální rozdíl obou obrazů nejmenší. Jedním z přístupů, které k tomu můžeme použít, je algoritmus KLT (Kanade-Lucas-Tomasi), jehož základ byl představený v r. 1981. Vychází ze třech základních předpokladů (podle [2]):

1. Jas sledovaných bodů se s pohybem nemění.
2. Pohyb bodů mezi jednotlivými snímky je relativně malý.
3. Sousední body patřící stejné ploše se pohybují po podobných trajektoriích.

Základní charakteristikou algoritmu je, že vektor posunutí je nejprve odhadnut a dále je postupně iterativně zpřesňován. To umožňuje dosáhnout logaritmické časové složitosti. Podle předpokladu č. 1 platí (viz [2]):

$$I(x + u{\cdot}dt, y + v{\cdot}dt, t + dt) = I(x, y, t) \tag{3.5}$$

Optický tok je pak získán minimalizací funkce udávající metriku pro rozdíl obou obrazů:

$$\sum_{x,y} [I(x + u{\cdot}dt, y + v{\cdot}dt, t + dt) - I(x, y, t)]^2 \rightarrow min. \tag{3.6}$$

Přístup použitý v algoritmu Lucas-Kanade využívá gradientní metodu pro minimalizaci. Podrobnější vysvětlení tohoto algoritmu naleznete například v [2] nebo [19]. Důležité je, že vzhledem k tomu, že pohyb mezi snímky je malý (ideálně nulový, předpoklad č. 2), můžeme hledání bodu omezit na malé okolí jeho aktuální pozice dané oknem $M$.

Vylepšení algoritmu můžeme provést použitím pyramid ([2]). Ta vznikne jako množina stejných obrazů, ovšem každý další je v nižším rozlišení (zpravidla čtvrtinovém). Pomyslný vrchol pyramidy pak tvoří obraz s nejnižším rozlišením – zde začíná výpočet. Je vypočten optický tok a ten je promítnut na další úroveň pyramidy, tento krok se opakuje, dokud není výpočet proveden v obraze s nejvyšším rozlišením. Takový přístup umožňuje sledovat i body, které se mezi snímky od sebe poměrně hodně vzdálí, tedy které se pohybují velkou rychlostí.

## 3.5 Predikce pohybu pomocí Kalmanova filtru

KLT tracker se nepříliš dobře vyrovnává se situací, kdy zmizí veškeré sledované body, a poměrně velké množství z nich se snaží spojit s jinými body v novém záběru. Takové body si ovšem navzájem neodpovídají, toto chování je chybné. Často se však stane, že body jsou od sebe poměrně vzdálené, což odpovídá velkému skoku v jejich rychlosti. Ten teoreticky možný je, ovšem většina existujících objektů se takto nechová.

Pokud bychom měli model pohybu každého bodu, mohli bychom na základě kontroly podle tohoto modelu odhalit takovéto náhlé změny a bod vyloučit ze zpracování. Rovněž by byla vyřešena otázka počátečního odhadu pro hledání bodů v aktuálním snímku – byla by dána predikcí pomocí tohoto modelu. K tomu nám může posloužit Kalmanův filtr. Informace v této kapitole jsou volně převzaty z [13], [2] a [20].

Kalmanův filtr byl představen v r. 1960. Tento filtr slouží k vytváření modelů stavů dynamických diskrétních systémů. Pracuje rekurzivně, kdy nový stav vzniká korekcí filtru z předchozího kroku na základě nově naměřených hodnot. Není uchovávána tedy celá historie stavů, ale pouze aktuální. Abychom mohli Kalmanův filtr použít, musí být splněny následující podmínky:

1. Modelovaný systém je lineární
2. Náhodný proces musí být popsatelný střední hodnotou a rozptylem

Podmínka č. 1 je důležitá právě proto, aby se budoucí stav dal vyjádřit jako funkce stavu aktuálního. Činnost filtru je poměrně složitá, proto zde nastíním pouze základní myšlenky a zabývat se budu návrhem filtru pro predikci pohybu objektu. Detailní popis principů Kalmanova filtru lze najít například v [20].

Filtr sestává z několika základních komponent. *Stavový vektor* reprezentuje aktuální odhad stavu systému, další součástí je *kovarianční matice*, ukazující přesnost měření náhodného procesu. Filtr musí být nejprve inicializován počátečním odhadem stavu, další činnost se potom sestává ze dvou neustále se opakujících kroků, ke kterým dochází pro každé měření (například každý časový okamžik, kdy je změřena pozice pohybujícího se objektu). Tyto kroky jsou následující:

1. **Predikce** – předpověď stavového vektoru a kovarianční matice.

2. **Korekce** – máme k dispozici novou naměřenou hodnotu, na jejímž základě je vypočten tzv. Kalmanův zisk, který udává míru zlepšení odhadu. Ten je společně s kovarianční maticí rovněž v tomto kroku aktualizován.

Nyní se dostáváme k samotné realizaci predikce pozice pohybujícího se objektu pomocí Kalmanova filtru. Pohyb bodu můžeme popsat třemi veličinami – polohou $x$, rychlostí $v$ a zrychlením $a$. To popisují následující rovnice (dále převzato z [13]):

$$x(t+T) = x(t) + v(t)\cdot T + \frac{1}{2}a(t)\cdot T^2 \tag{3.7}$$

$$v(t+T) = v(t) + a(t)\cdot T \tag{3.8}$$

Stavový vektor bude obsahovat informace o všech třech veličinách:

$$\tilde{x} = \begin{bmatrix} x \\ v \\ a \end{bmatrix} \tag{3.9}$$

Stavová rovnice je následující:

$$x(t+T) = \phi(T)x(t) + w(T), \tag{3.10}$$

kde $x(t+T)$ je nový a $x(t)$ aktuální stav systému, $w(T)$ představuje náhodný proces s charakterem bílého šumu (chyba měření). Přechodová matice $\phi(T)$ realizující přechod od aktuálního stavu k novému má tvar:

$$\phi(T) = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \tag{3.11}$$

$T$ představuje časový interval mezi jednotlivými měřeními. Pro měření provedené v každém snímku videosekvence můžeme zvolit $T = 1$.

# Kapitola 4

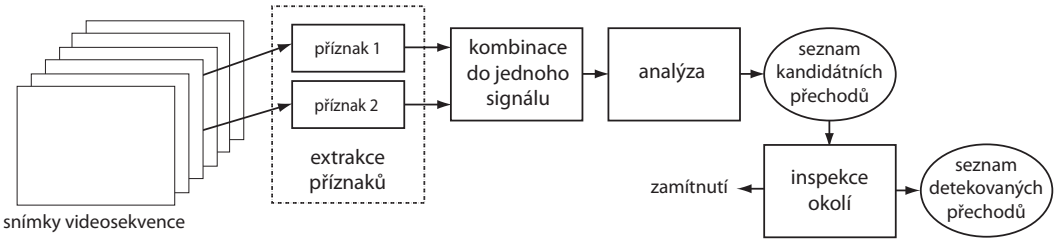
# Detektor přechodů

V této kapitole je představen samotný systém pro detekci přechodů. Nejprve je vysvětlen obecně pomocí bloků, dále je zavedena metrika pro měření podobnosti mezi snímky a rovněž způsob analýzy průběhu této metriky. Je uveden způsob detekce pomocí více různých příznaků a dodatečné kontroly nalezeného přechodu. Diskutovány jsou i možnosti, které ve výsledném systému nejsou použity, ale obecně mohou být uvažovány.

Detekce je založena na odhadu míry podobnosti mezi snímky, která je v místech přechodů nejmenší, respektive dochází k její charakteristické změně. Blokové schéma navrženého systému je zobrazeno na obr. 4.1. Ze zdrojové videosekvence jsou z každého snímku extrahovány příznaky, tak jak bylo představeno v kapitolách 3.1 a 3.2. Ty jsou použity právě ke stanovení metriky podobnosti – dva různé příznaky poskytují dvě hodnoty, které jsou následně zkombinovány do jednoho jediného průběhu v závislosti na čísle snímku.

Dalším krokem je analýza tohoto signálu. Jejím výsledkem je seznam nalezených přechodů, přičemž každý přechod je určen číslem prvního snímku nové scény. To je jednoznačné u střihů, ale u postupných přechodů se aktuální a nová scéna určitou dobu prolínají. V takovém případě by měla analýza určit nejlépe začátek a konec přechodu, zjednodušeně postačí pouze detekovat právě jeden „střih“, který se bude nacházet uvnitř průběhu postupného přechodu. Navržený systém používá druhou, jednodušší možnost.

Seznam získaný v předchozím kroku ale ještě nemusí být konečný. Jako poslední stupeň detektoru slouží inspektor okolí. Ten na základě znalosti širšího kontextu přechodu může provést jeho zamítnutí, čímž se zlepší výsledná přesnost. Tato součást je pouze volitelná a nemusí být při detekci použita, je zamýšlena spíše pro některá specifická vstupní data.




Obrázek 4.1: Blokové schéma navrženého systému pro detekci přechodů

## 4.1 Metrika podobnosti snímků

Hodnoty příznaků z videa popisují jeden snímek, přechody jsou ale určeny minimálně dvěma snímky (postupné přechody více). Vzhledem k tomu, že detekce probíhá v podstatě jako hledání míst ve videosekvenci, kdy se hodnota příznaku prudce změní, veličina vhodná pro popis tohoto chování bude vyjadřovat míru shody mezi snímky a detekce bude probíhat v lokálních extrémech, kde je tato míra nejmenší. Získané hodnoty je potřeba převést do jednotného tvaru, ve kterém se dají vzájemně srovnávat a případně kombinovat. K popisu podobnosti mezi dvěma snímky tedy zavádím metriku $d$, která je definována následovně:

- Popisuje relaci mezi dvěma snímky. Zpravidla se jedná o sousední snímky (aktuálně zpracovávaný a předchozí).
- Nabývá hodnot z intervalu <0; 1>.
- Hodnota 0 vyjadřuje maximální podobnost (zcela totožné snímky) a naopak 1 znamená největší míru neshody. Orientace je takto zvolena z praktických důvodů, protože výpočet obvykle probíhá jako suma dílčích rozdílů, tedy výsledek je nulový právě v případě absolutní shody.

Tato metrika je nejprve zavedena pro každý druh příznaku samostatně, následně je možná jejich kombinace do jedné. Pro barevný histogram je podle [11] uvažována suma absolutních hodnot rozdílů jednotlivých dílů dvou histogramů:

$$d_{hist}[n] = \sum_i |(H_n(i) - H_{n-1}(i)| \tag{4.1}$$

Hodnoty barevných histogramů musí být normalizované, aby bylo zaručeno, že výsledek bude spadat do požadovaného intervalu. Normalizace znamená dělení hodnoty každého dílu jejich celkovým součtem:

$$H_{norm}(i) = \frac{H(i)}{\sum_{j=1}^{N} H(j)} \tag{4.2}$$

Pro sledování bodů by stačilo zohlednit počet $N^{out}$ ztracených bodů z předchozího snímku, kvůli lepší detekci postupných přechodů je ale vhodné uvažovat i počet bodů $N^{in}$ nově vzniklých. Aby byly hodnoty srovnatelné mezi různými snímky, jsou děleny počtem všech bodů $\sigma$ v odpovídajícím snímku. To zajistí rovněž posunutí do intervalu <0; 1>. Jako výsledek se bere maximum:

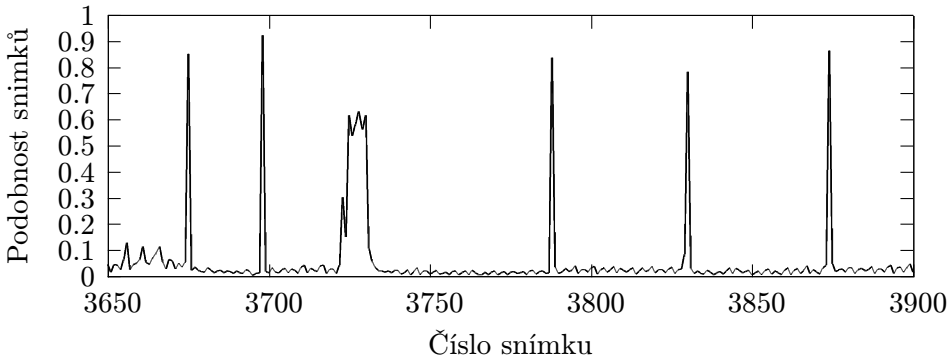
$$d_{tracker}[n] = \max\left(N^{out}_{n-1}/\sigma_{n-1},\ N^{in}_{n}/\sigma_{n}\right) \tag{4.3}$$

## 4.2 Analýza získaného průběhu

Průběh metriky podobnosti pro jednotlivé snímky tvoří diskrétní signál. Jednotlivé přechody se v něm objevují jako charakteristické tvary:

- **Střihy** se projeví jako velmi výrazné skokové změny – obvykle špičky, případně přechod na jinou úroveň signálu po delší dobu.

- **Postupné přechody** jsou charakteristické delší dobou trvání a trojúhelníkovým tvarem signálu. Tento tvar platí zejména pro typy *dissolve* a podobné, v případě efektových přechodů může být různě zkreslený.




Obrázek 4.2: Ukázka signálu pro analýzu. Průběh obsahuje 5 střihů a 1 postupný přechod.

Ukázka naměřeného signálu je na obr. 4.2. Nejčastěji používaným způsobem detekce je srovnávání hodnoty signálu s nastaveným prahem, a pokud dojde k jeho překročení, místo je označeno jako přechod. V zásadě jsou možné dva způsoby prahování:

1. **Globální práh** – jeho hodnota je nastavena pevně po celou dobu analýzy. Tento způsob je jednoduchý, ale hlavní nevýhoda spočívá v tom, že bez znalosti vstupních dat lze jen velmi těžko předpokládat minimální úroveň, pod kterou se bude vždy signál vyskytovat mezi přechody.

2. **Adaptivní práh** – hodnota se průběžně přizpůsobuje tak, aby aktuální klidová hodnota signálu byla pod prahem a ten byl překročen pouze v případě výrazných změn. Tímto způsobem je možné nalézt i méně výrazné přechody, aniž by muselo dojít k nastavení nízké hodnoty prahu (jako v případě globálního prahování) a tím negativního ovlivnění přesnosti detekce.

V navrženém systému jsem se však rozhodl použít jiný způsob analýzy. Vzhledem k tomu, že v místě přechodu dochází k velké změně signálu, první derivace je zde rovněž vysoká. Naopak v případě klidového stavu je derivace velmi blízká nule. Detekce tedy probíhá tak, že ze signálu je odhadnuta první derivace konvolucí s jádrem [-1, 1], což vlastně odpovídá rozdílu dvou po sobě jdoucích hodnot:

$$d' \approx [-1, 1] * d, \quad d[n] = d[n] - d[n-1] \tag{4.4}$$

Protože signál obsahuje jemný šum, je nutné jej před touto operací vyhladit, jinak by se tento šum derivací velmi zvýraznil. Vyhlazení je možno provést konvolucí s obdélníkovým oknem, případně jádrem tvaru Gaussova průběhu. Navíc vzhledem k asociativitě derivace a konvoluce je možné vyhlazení a aproximaci derivace sloučit do jednoho operátoru:

$$\frac{d}{dx}(h * f) = \frac{dh}{dx} * f \tag{4.5}$$

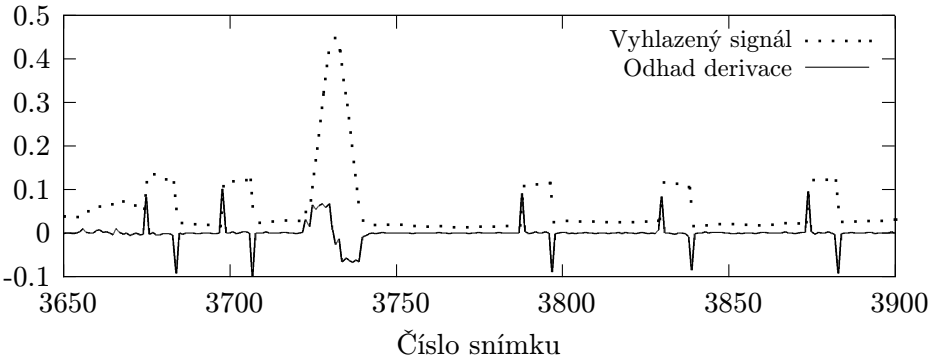
Například pro vyhlazení oknem délky 4 by výsledný operátor vypadal takto:

$$[-1,1] * \left( \left[ \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4} \right] * f \right) = \left[ -\frac{1}{4}, 0, 0, 0, \frac{1}{4} \right] * f \tag{4.6}$$

Pro Gaussův průběh by se postupovalo obdobně. Míra vyhlazení (v příkladu určena délkou okna) musí být dostatečně velká, aby šum příliš neovlivňoval derivaci, ale zároveň nesmí dojít i k vyhlazení samotných stop přechodů. Po této operaci je ještě provedeno prahování, kdy hodnoty derivace menší než stanovený práh T jsou nahrazeny za nulové:

$$d_{thr}[n] = \begin{cases} d[n] & \text{pokud } \ d[n] > T \\ 0 & \text{jinak} \end{cases} \tag{4.7}$$

Zatímco hodnota signálu mezi přechody závisí na konkrétním obsahu scény, derivace odpovídá typu přechodu, proto práh může být nastaven globálně. Odhad derivace pro ukázkový signál vidíme na obr. 4.3. Pro ilustraci není na této ukázce provedeno prahování.




Obrázek 4.3: Odhad derivace signálu z obr. 4.2 po vyhlazení a bez použitého prahování.

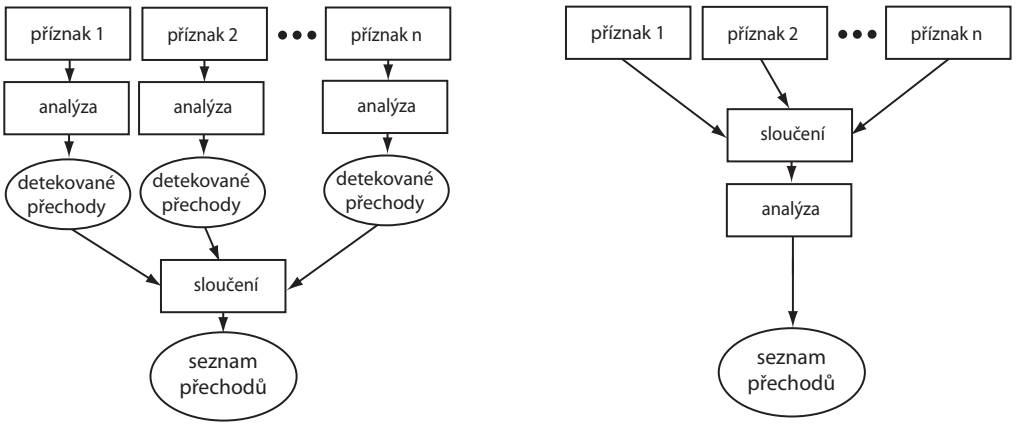
Přechody se v průběhu derivace projeví jako dvojice impulsů, u střihů jsou oddělené, v případě postupných přechodů plynule navazují. Detekce je pak založena na hledání těchto impulsů. Tento přístup nerozlišuje jednotlivé typy přechodů, u postupných lze pak předpokládat horší schopnost nalezení, protože způsobují mnohem menší změnu signálu než střihy.

## 4.3 Kombinace více příznaků

Detekce může probíhat za použití pouze jednoho příznaku, ale pokud jich máme k dispozici více, bylo by vhodné využít jejich různých vlastností a kombinací vytvořit detektor lepší. Zlepšení se očekává v oblasti přesnosti nebo spolehlivosti detekce. Podobný přístup se používá například u klasifikátoru AdaBoost, který vytváří silnější klasifikátor pomocí kombinace více klasifikátorů slabých. Jsou možná dvě hlavní řešení:

1. **Spojení je provedeno před detekcí** – vytvoření nové hodnoty kombinací obou samostatných příznaků
2. **Spojení následuje po detekci** – ta nejprve proběhne pro každý příznak samostatně a následuje sloučení nalezených přechodů.

Schématické znázornění obou možností zobrazuje obr. 4.4. Výsledek u druhého přístupu může odpovídat průniku nebo sjednocení množin přechodů detekovaných z jednotlivých příznaků, v závislosti na tom se pak může snížit počet falešných detekcí (průnik), případně zlepšit úspěšnost detekce (sjednocení), ale jeden parametr vždy na úkor druhého. Naopak výhodou prvního přístupu je, že může dojít ke zlepšení přesnosti i spolehlivosti detekce zároveň. Pokud jeden (nebo více) příznak způsobuje vysoké množství falešných detekcí, pak kombinace před detekcí může jeho vliv snížit. Pokud ale sjednocujeme seznamy již nalezených přechodů, nemůžeme rozhodnout, které přechody vyloučit, tudíž v tomto případě horší přesnost jednoho příznaku ovlivní výsledek mnohem výrazněji.




Obrázek 4.4: Dva různé zůsoby kombinace příznaků

V navrženém systému jsem zvolil variantu č. 1, kdy z dílčích metrik podobnosti pro jednotlivé příznaky vytvořím novou hodnotu, která ovšem opět splňuje podmínky metriky. Výsledek obecně odpovídá lineární kombinaci jednotlivých hodnot:

$$d = \alpha_1 \cdot d_1 + \alpha_2 \cdot d_2 + ... + \alpha_n \cdot d_n,\ \alpha \in <0;\ 1>,\ \sum_{i=1}^{N} \alpha_i = 1 \tag{4.8}$$

Protože používám pouze dva příznaky, lze vzorec přepsat do tvaru:

$$d = \alpha \cdot d_{tracker} + (1 - \alpha) \cdot d_{hist},\ \alpha \in <0;\ 1> \tag{4.9}$$

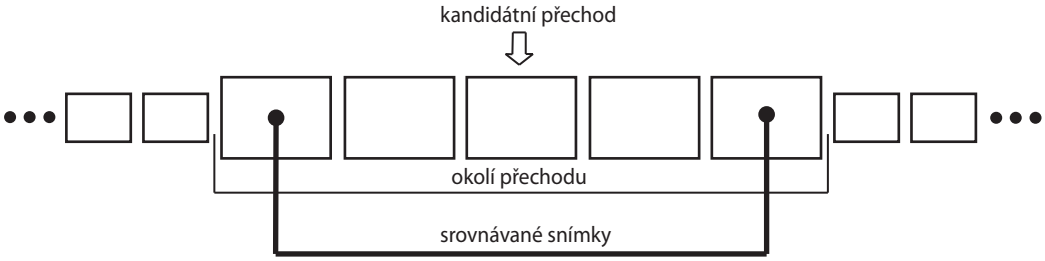
Z této nově vzniklé metriky pak probíhá detekce standardním způsobem, tak jak bylo představeno v kap. 4.2. Váha $\alpha$ ovlivňuje vlastnosti detektoru, ale obecně nelze tento vliv charakterizovat, protože záleží na použitých příznacích a jejich chování v různých situacích. Hodnota váhy může být v jednodušším případě dána staticky po celou dobu, pokud bychom ale měli k dispozici například odhad aktuální spolehlivosti detekce z každého příznaku, váha by se mohla adaptivně přizpůsobovat. Pokud by tak jeden z příznaků kupříkladu způsoboval falešné detekce při rychlém pohybu, jeho vliv by se dočasně snížil.

## 4.4 Inspekce okolí

Část nalezených přechodů je detekována chybně, tedy tyto přechody ve zkoumané videosekvenci neexistují. Například při použití barevného histogramu toto chování způsobují náhlé

jasové změny, způsobené odpálením blesku fotoaparátu. U sledování bodů k němu může dojít v případě prudké změny rychlosti či směru pohybu. Vhodné by bylo tyto detekce dodatečně vyřadit, aby nezhoršovaly přesnost systému.

Na základě širšího okolí přechodu je možné usoudit, zda blízké snímky přechodu patří stejnému záběru (pak dojde k zamítnutí), nebo dvěma (či obecně více) různým. To lze provést například porovnáním barevných histogramů snímků vzdálených N od pozice kandidátního přechodu (ukázka na obr. 4.5). Práh pro zamítnutí by měl být nastaven tak, aby k němu došlo pouze v případech, kdy jsou si snímky velmi podobné. Jinak by mohlo dojít ke značnému zhoršení úspěšnosti detekce, což je ale v menší míře stejně očekáváno.




Obrázek 4.5: Ilustrace porovnávaných snímků při inspekci okolí, N = 2.

Jako dobře použitelná velikost okolí se jeví hodnota alespoň N = 2, podobnost méně vzdálených snímků u postupných přechodů je obvykle totiž příliš vysoká. Velikost není shora omezena, ale při příliš velké hodnotě by mohl rozsah překročit délku obou záběrů a především jejich obsah by se mohl značně změnit, aniž by k přechodu došlo.

# Kapitola 5

# Implementace

Implementaci navrženého systému jsem provedl v jazyce C++ za využití knihovny OpenCV pro zpracování obrazu a toolkitu WxWidgets k vytvoření grafického uživatelského rozhraní. Práce spočívala ve dvou hlavních krocích: Jádro projektu představuje realizace *knihovny obsahující detektor* a potřebné komponenty. Druhým krokem bylo vytvoření *demonstrační aplikace* používající detektor z knihovny. Tato kapitola obsahuje popis struktury a použití detektoru, vysvětlení nejdůležitějších tříd a některých implementačních detailů.

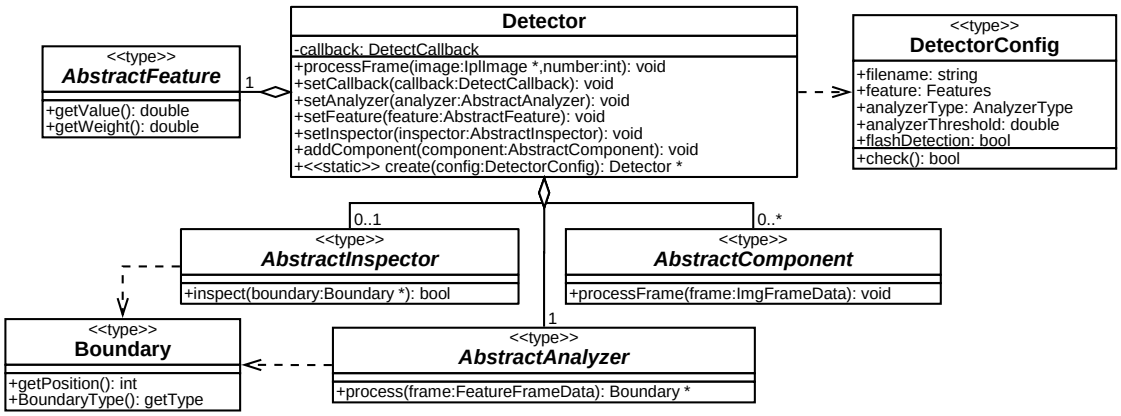
Návrh i implementace byla provedena za využití objektového přístupu, struktura bude dále demonstrována zejména pomocí diagramů tříd. Na každém diagramu jsou zobrazeny pouze nejdůležitější třídy související s ilustrovanou částí programu. Rovněž metody a atributy nemusí být zobrazeny všechny, ale opět pouze takové, které jsou nutné pro pochopení nebo demonstraci daného řešení.

## 5.1 Struktura detektoru

Jádro detektoru bylo navrženo modulárně s ohledem na budoucí rozšiřitelnost. Při návrhu jsem vycházel z myšlenky životního cyklu detektoru, kdy jeho činnost lze popsat rámcově v několika obecných krocích:

1. Inicializace
2. Předání dalšího snímku z videa – jednotlivé součásti detektoru (tracker, modul pro výpočet barevného histogramu, atd.) jej zpracují
3. Analyzátor rozhodne, zda došlo k přechodu
4. Pokud ano, inspektor jej následně potvrdí nebo zamítne
5. Činnost se opakuje od bodu 2, jsou-li další snímky
6. Ukončení činnosti, uvolnění zdrojů

Samotný detektor implementuje třída `Detector`, její struktura přímo reflektuje popsaný životní cyklus. Základními částmi jsou komponenty detektoru odvozené od abstraktní třídy `AbstractComponent`, dále příznak, ze kterého probíhá detekce zapouzdřený třídou `AbstractFeature`, analyzátor (`AbstractAnalyzer`) a inspektor (`AbstractInspector`).




Obrázek 5.1: Diagram tříd detektoru. Jednotlivé komponenty třídy `Detector` jsou zobrazeny pouze jako abstraktní, konkrétní budou představeny dále.

Jednotlivé asbtraktní třídy plní roli rozhraní, které je navrženo obecně tak, aby daná komponenta mohla implementovat jakýkoliv způsob či algoritmus. Diagram základních tříd detektoru je zobrazen na obr. 5.1.

Některé algoritmy potřebují zpracovávat snímky videa a přistupovat k nim jako k obrazům. Příslušné třídy se pak musí zaregistrovat jako *komponenty* detektoru. Ten totiž každý nový snímek nejprve předá všem komponentám (odpovídá kroku 2 v životním cyklu), které jej zpracují a výsledky si uloží uvnitř. Dále je zavolán analyzátor, který zjistí hodnotu nastaveného příznaku, a pokud je nalezen přechod, vrátí nalezený přechod. Kroky popsané doposud jsou povinné, jako volitelná možnost je přiřadit inspektor, kterému je v takovém případě nalezený přechod nakonec předán. Ten pak rozhodne o jeho zamítnutí. Pro každý nalezený přechod je zavolána nastavená *callback* funkce, ve které je obslužný kód.

Vytvoření instancí všech potřebných objektů a správné nastavení jednotlivých vazeb je relativně komplikované. Pro zjednodušení jsem vytvořil třídu `DetectorConfig`, ve které lze nastavit jednotlivé parametry detektoru. Podle návrhového vzoru *Factory* pak může být na základě této konfigurace vytvořena instance detektoru, který lze okamžitě použít.

## 5.2 Tracker

Tato třída je implementována jako komponenta detektoru, slouží k vyhledání výrazných bodů a jejich sledování v průběhu videa. Je základem pro získání hodnoty příznaku založeném právě na sledování bodů. Prvotním úkolem je nalezení bodů vhodných ke sledování – k tomu jsem využil implementaci GoodFeaturesToTrack z knihovny OpenCV. Ta může využívat Harrisův detektor, který byl popsán v kap. 3.3, případně jiný algoritmus. Množství nalezených bodů lze ovlivnit jednak nastavitelným limitem, hodnotou „kvality“ bodu a dále například jejich minimální vzájemnou vzdáleností. Diagram tříd je zobrazen na obr. 5.2.

Každý sledovaný bod je reprezentován třídou `TrackedPoint`. Ta zapouzdřuje funkce predikce pozice a pro budoucí rozšíření rovněž uchovává trajektorii bodu v posledních N snímcích. K uložení slouží kruhový buffer. Jako Kalmanův filtr pro predikci pozice jsem využil opět implementaci OpenCV. Filtr je inicializován aktuální pozicí bodu a odhady rychlosti a zrychlení podle vztahů:

$$v[n] \approx x[n] - x[n-1] \tag{5.1}$$

$$a[n] \approx v[n] - v[n-1] \tag{5.2}$$

Korekce filtru je prováděna už pouze pozicí bodu. Protože odhad zrychlení je možné provést až po třech snímcích, je do této doby jako predikce vracena současná poloha. Odhad pozice je využíván jako výchozí poloha pro hledání bodu na dalším snímku. Dále je nastaveno omezení vzájemné vzdálenosti predikce a nalezené polohy, ta se nesmí lišit více než je zadaný limit – v opačném případě je bod vyřazen. Sledování bodů zajišťuje KLT tracker, jeho implementace v OpenCV Kalmanův filtr neobsahuje (viz [8]), proto je použit uvedeným způsobem samostatně.

Při zavolání funkce pro zpracování dalšího snímku jsou nejprve vyhledány body z předchozího snímku (kromě prvního volání, kdy dochází k inicializaci), nenalezené a vyřazené z důvodů popsaných výše jsou vyřazeny ze zpracování. Dále dochází k hledání bodů nových a jejich přídání ke stávajícím. Ukládány jsou celkové počty bodů v aktuálním a minulém snímku, dále množství bodů ztracených a nových. Sledované body měly tendenci, zvláště po přidání nových, se postupně přibližovat a tvořit husté shluky, které zbytečně snižovaly výkon. Proto ještě dochází k „pročištění“ bodů tak, aby dodržovaly určenou minimální vzdálenost.

## 5.3 Příznaky

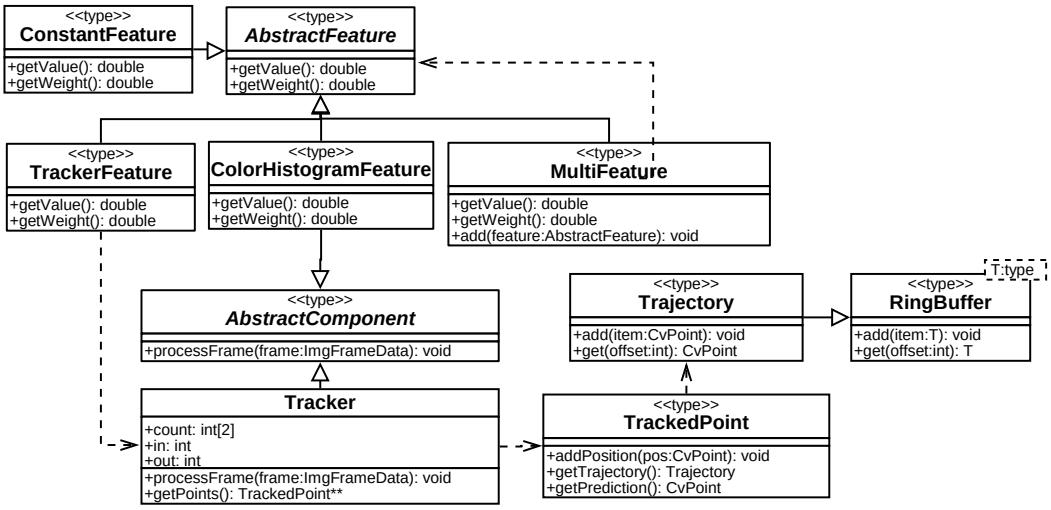
Příznaky obecně vrací hodnotu splňující kritéria metriky podobnosti zavedené v kapitole 4.1. Dále musí umožnit vrácení váhy – tu může příznak volit automaticky, případně může být nastavitelná uživatelem, výchozí váha je 1.0. Implementoval jsem tyto potřebné příznaky (diagram tříd na obr. 5.2):

- `ColorHistogramFeature` – podobnost snímků daná srovnáním barevných histogramů.
- `TrackerFeature` – porovnání na základě sledování bodů.
- `ConstantFeature` – vrací konstantní nastavenou hodnotu.
- `MultiFeature` – kombinace více přízaků do jednoho.

`ColorHistogramFeature` musí být zároveň komponenta detektoru, protože potřebuje počítat barevné histogramy snímků, tedy musí k nim přistupovat jako k obrazovým datům. K tomu využívá implementaci histogramů v OpenCV, ale pro porovnávání je využita vlastní funkce realizující výpočet podle kap. 4.1. Počet dílů histogramu jsem stanovil podle [11] na celkových 512, tedy 8 dílů pro každý barevný kanál.

`TrackerFeature` pro svou činnost vyžaduje referenci na `Tracker`, který musí být registrován jako komponenta detektoru. Z toho jsou získávány hodnoty pro výpočet podobnosti podle vztahů uvedených v kapitole 4.1

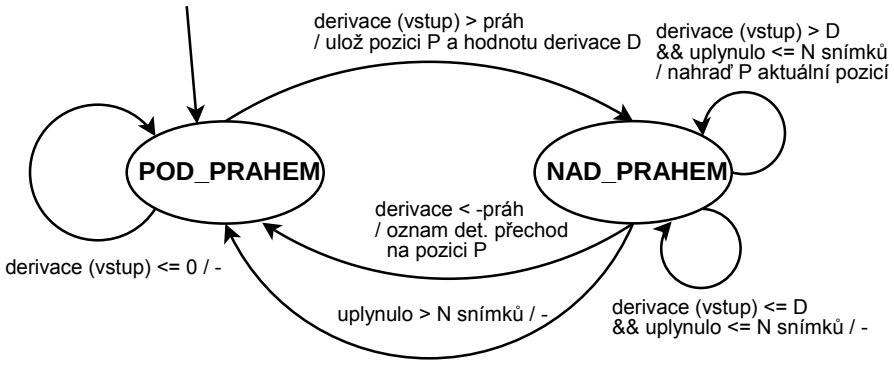
`MultiFeature` je sama příznakem a zároveň umožňuje na sebe vázat příznaky jiné. Výsledná hodnota odpovídá váženému průměru všech vázaných příznaků. Váhy jsou získány z nich samotných.




Obrázek 5.2: Diagram tříd pro příznaky, tracker a třídy, které používá.

## 5.4 Analyzátor

Implementoval jsem jediný analyzátor realizující analýzu popsanou v kapitole 4.2, ten obsahuje třída `DerivativeAnalyzer`. Vyhlazení signálu probíhá konvolucí s obdélníkovým oknem nastavitelné délky, dále je odhadována derivace odečtením po sobě jdoucích hodnot. Způsob detekce lze popsat jednoduchým konečným automatem zobrazeným na obr. 5.3:




Obrázek 5.3: Konečný automat znázorňující činnost analyzátoru pro detekci přechodů.

Pokud hodnota derivace překročí práh a je kladná, jedná se o místo střihu nebo začátek postupného přechodu, číslo snímku, při kterém k překročení došlo je uloženo. Při nízkém prahu však toto může způsobit i šum a následující přechod by pak byl chybně detekován. Předpokládáme, že skutečný přechod je výraznější než šum, proto pokud v tomto stavu automatu je na vstupu hodnota derivace vyšší než uložená hodnota, je předchozí pozice nahrazena aktuální.

Protože se každý přechod projevuje jako kladný impuls následovaný záporným, čeká se na tento záporný impuls, než je ohlášena detekce. Toto čekání způsobí zpoždění ohlášení

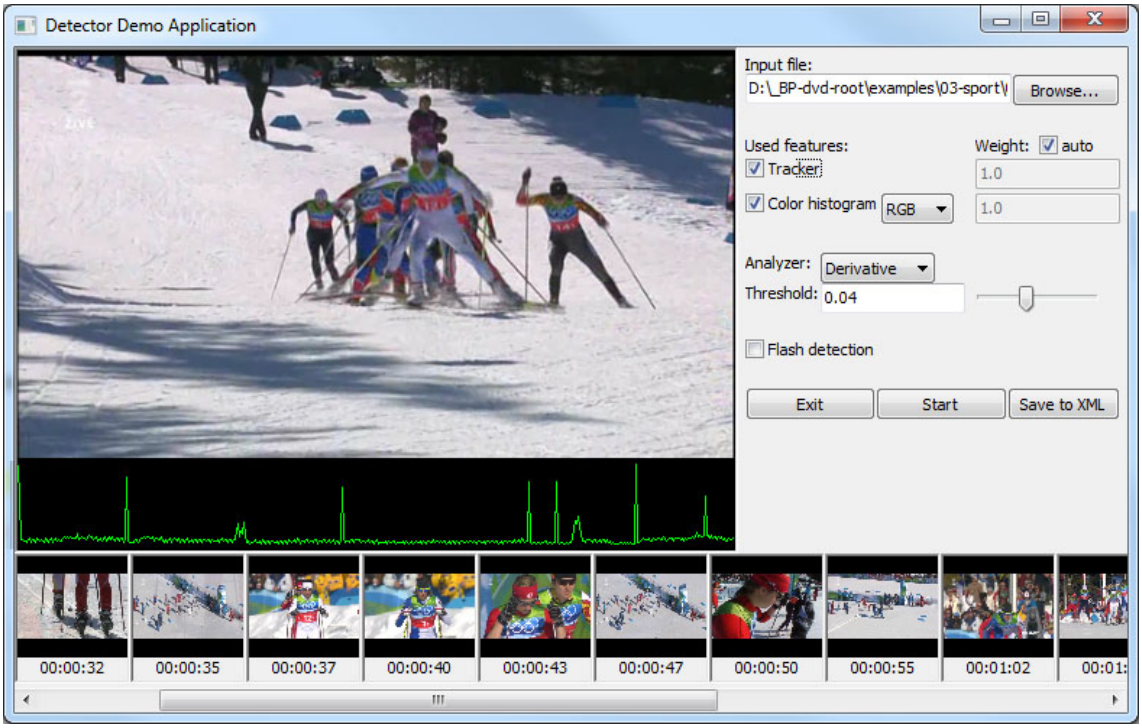
přechodu, délka čekání je ale omezena na určitý počet snímků, po kterém se automat vrací do počátečního stavu. Dané omezení reflektuje minimální vzdálenost jednotlivých přechodů.

## 5.5 Inspektor okolí

Inspektor okolí je rovněž komponenta detektoru – využívá kruhový buffer k ukládání N posledních snímků, aby z nich v případě potřeby mohly být vypočteny histogramy pro dodatečné srovnání. Protože je třeba prohlížet i okolí přechodu v budoucnosti (snímky, které ještě nebyly načteny), detektor by měl zajistit volání inspektoru odloženě až v době, kdy jsou k dispozici příslušné snímky. To však není nutné vzhledem ke zpoždění danému návrhem analyzátoru, pro zkoumaný rozsah parametrů je dostatečné. Pokud by ale knihovna byla rozšířena, může vzniknout nutnost tento odklad doimplementovat.

## 5.6 Demonstrační aplikace

Pro demonstraci jsem vytvořil aplikaci s grafickým uživatelským rozhraním (obr. 5.4), využil jsem toolkit WxWidgets. Je zamýšlena pouze pro předvedení činnosti detektoru, k provedení testů jsem použil konzolovou aplikaci `Detector`. Poskytuje možnost nastavení základních parametrů, v průběhu činnosti zobrazuje aktuálně zpracovávaný snímek a pro ukázku také průběh měřeného signálu metriky podobnosti. Protože je činnost detektoru výpočetně velmi náročná, kvůli zajištění odezvy uživatelského rozhraní běží v samostatném vlákně.




Obrázek 5.4: Ukázka uživatelského rozhraní demonstrační aplikace.

# Kapitola 6

# Testování a vyhodnocení

Velmi důležitou součástí realizace systému je následné zjištění a ověření jeho vlastností. Tato kapitola se věnuje právě této oblasti. Nejprve jsou popsány zjišťované a měřené parametry, testovací data jsou stručně charakterizována a stěžejní částí této kapitoly jsou výsledky provedených testů.

## 6.1 Sledované parametry

Testovací data obsahují množinu přechodů $B_o$, výsledkem detekce je druhá množina $B_d$, zahrnující nalezené přechody. Úkolem vyhodnocení je tyto množiny vhodným způsobem porovnat a z tohoto srovnání posoudit zkoumanou vlastnost. Při detekci mohou pro každý nalezený přechod nastat tyto situace ([14]):

1. V místě detekovaného přechodu se opravdu přechod nachází. To je samozřejmě optimální, jedná se o korektní detekci. Tento případ označujeme jako TP (true positive).

2. Místo bylo označeno jako přechod, ale ten se ve zdrojovém videu na dané pozici nenalézá. Byl detekován falešný přechod, to označujeme jako FP (false positive). Jedná se o nežádoucí chování, které se snažíme co nejvíce potlačit.

3. Ve videosekvenci je přechod, který se ale v množině $B_d$ nenachází. Přechod tedy nebyl detekován, označujeme zkratkou FN (false negative).

Detekce vlastně odpovídá klasifikaci do dvou tříd „V daném snímku je přechod“ a „V daném snímku není přechod“. Chyba I. druhu je pak falešná detekce, chyba II. druhu znamená nenalezený přechod. Úspěšnost takové klasifikace vyjadřujeme pomocí několika nejdůležitějších veličin ([4]). Jejich názvy pro jednoznačnost ponechám v angličtině:

**Recall** značí, kolik přechodů, ze všech obsažených ve videosekvenci, bylo nalezeno. Určí se podle vztahu:

$$recall = \frac{TP}{TP + FN} = \frac{\text{Počet správně nalezených přechodů}}{\text{Celkový počet přechodů ve videosekvenci}} \tag{6.1}$$

**Precision** vyjadřuje množství falešných detekcí, určí se jako:

$$precision = \frac{TP}{TP + FP} = \frac{\text{Počet správně nalezených přechodů}}{\text{Celkový počet nalezených videosekvenci}} \tag{6.2}$$

Na tyto veličiny se také můžeme dívat z hlediska pravděpodobnosti. Recall odpovídá pravděpodobnosti, že náhodně vybraný přechod z množiny $B_o$ je detekován. Dále pokud náhodně vybereme přechod z množiny $B_d$, pak pravděpodobnosti, že se tento přechod nachází i v množině $B_o$, odpovídá precision.

Je zřejmé, že hodnoty těchto veličin spadají do intervalu <0; 1>. Recall 1.0 znamená, že všechny existující přechody byly nalezeny, ale nic neříká o počtu falešných detekcí. Naopak precision 1.0 vyjadřuje, že všechny nalezené přechody opravdu ve videu jsou, ale bez ohledu na hodnotu recall. Proto je nutné posuzovat vždy obě veličiny společně. K tomu se používá další veličina, která odpovídá výslednému skóre detektoru. Jedná se o **F-measure** (název opět ponechávám bez překladu):

$$F\text{-}measure = F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{6.3}$$

Pokud chceme zvýšit (nebo snížit) vliv některé z dílčích veličin na celkové skóre, pak nepoužijeme $F_1$, ale obecně $F_\beta$. Při testování budu vždy uvádět pouze hodnoty $F_1$, kvůli úplnosti však doplním vztah pro výpočet:

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{6.4}$$

## 6.2 Křivka precision-recall

Různým nastavením parametrů detektoru můžeme pro stejná testovaná data veličiny představené v kapitole 6.1 ovlivňovat. Je tak možné například nastavením hodnoty prahu zvýšit hodnotu recall na úkor precision a naopak (snížením prahu obecně dojde k nalezení i méně výrazných přechodů, ale tím zároveň nebude odstraněn šum, který způsobí falešné detekce).

Pokud budeme měnit jeden parametr detektoru a do grafu vynášet závislost precision na recall, vznikne charakteristická křivka. Ta vyjadřuje škálovatelnost detektoru, tedy přibližně v jakém poměru se dají vynášené veličiny mezi sebou měnit. V ideálním případě by měla tvar konstantní funkce s hodnotou 1, ale prakticky začne vždy od určité hodnoty recall klesat.

Srovnání dvou těchto křivek není jednoduchý úkol. Musíme porovnávat místa s konstantní hodnotou jedné z veličin, ve kterých lze například říct, že při stejném relativním množství falešných detekcí druhý detektor korektně nalezl větší množství přechodů a podobně.

## 6.3 Testovací data

Testy probíhaly na vlastní sadě testovacích dat. Tato data bylo nutné nejprve získat a poté provést jejich anotaci. To znamená stručně je charakterizovat a zejména najít přechody v nich obsažené, označit jejich pozice (začátek a konec u postupných přechodů), případně rovněž zaznamenat druh přechodu. Anotaci je obecně v některých případech možné provádět automaticky, kdy se využije nástroj, který tuto činnost provede. Takový nástroj je ale právě detektor přechodů, tudíž by jeho použití nebylo žádoucí s ohledem na výpovědní hodnotu výsledků. Proto jsem anotaci provedl manuálně, tedy každou videosekvenci jsem snímek po snímku prošel, konkrétně prostřednictvím programu VirtualDub. Obsažené přechody jsou uloženy v souboru ve formátu XML.

K testování jsem zvolil úseky televizního vysílání, video bylo zachyceno prostřednictvím digitálního DVB-T tuneru, tedy přímo vysílaný proud dat ve formátu MPEG-2. Dále bylo odstraněno prokládání, případně sníženo rozlišení kvůli rychlejšímu zpracování. Rovněž byla odstraněna zvuková stopa, protože ta je pro testy nepotřebná. Posledním krokem byla komprese do formátu MPEG-4, za účelem snížení velikosti výsledných souborů.

Obsah testovacích dat byl volen s ohledem na to, aby odpovídala různým typickým situacím, které se nejčastěji ve vysílání vyskytují. K dispozici tedy byly tyto sady:

- **Film** (seriál) – složení záběrů odpovídá filmové tvorbě s poměrně klidnými záběry, neobsahuje příliš rychlého pohybu. Náročnost detekce zde očekávám nejnižší.
- **Zpravodajství** – obsahuje velmi různorodou směs záběrů, mnohé z nich mají horší technickou kvalitu.
- **Sport** – část záznamu zimních olympijských her 2010. Charakteristickým rysem je přítomnost velmi rychlého pohybu a rovněž rychlých přechodů od detailních záběrů k celku (zoomování).

Další informace o jednotlivých sadách dat shrnuje tabulka 6.1. Největší výpovědní hodnotu by mělo srovnání navrženého systému s detektorem představeným v některé z jiných prací (například v [18]). K tomu by bylo nutné testy provádět na stejných datech, ta bohužel ale nejsou volně k dispozici, tudíž se mi je nepodařilo získat.

| | | Počet přechodů | | |
|---|---|---|---|---|
| Název | Délka | Celkem | Střihy | Postupné přechody |
| Film | 00:20:44 | 412 | 390 | 22 |
| Zpravodajství | 00:20:41 | 351 | 299 | 52 |
| Sport | 00:34:36 | 279 | 198 | 81 |

Tabulka 6.1: Přehled o délce testovacích dat a počtech obsažených přechodů.

## 6.4 Způsob vyhodnocení

Vlastnosti detektoru byly měřeny veličinami představenými v kapitole 6.1. Pro jednoznačnost je dále důležité stanovit způsob porovnání nalezených přechodů s referenční anotací. Přechod je považován za nalezený, pokud pro daný typ platí následující podmínky:

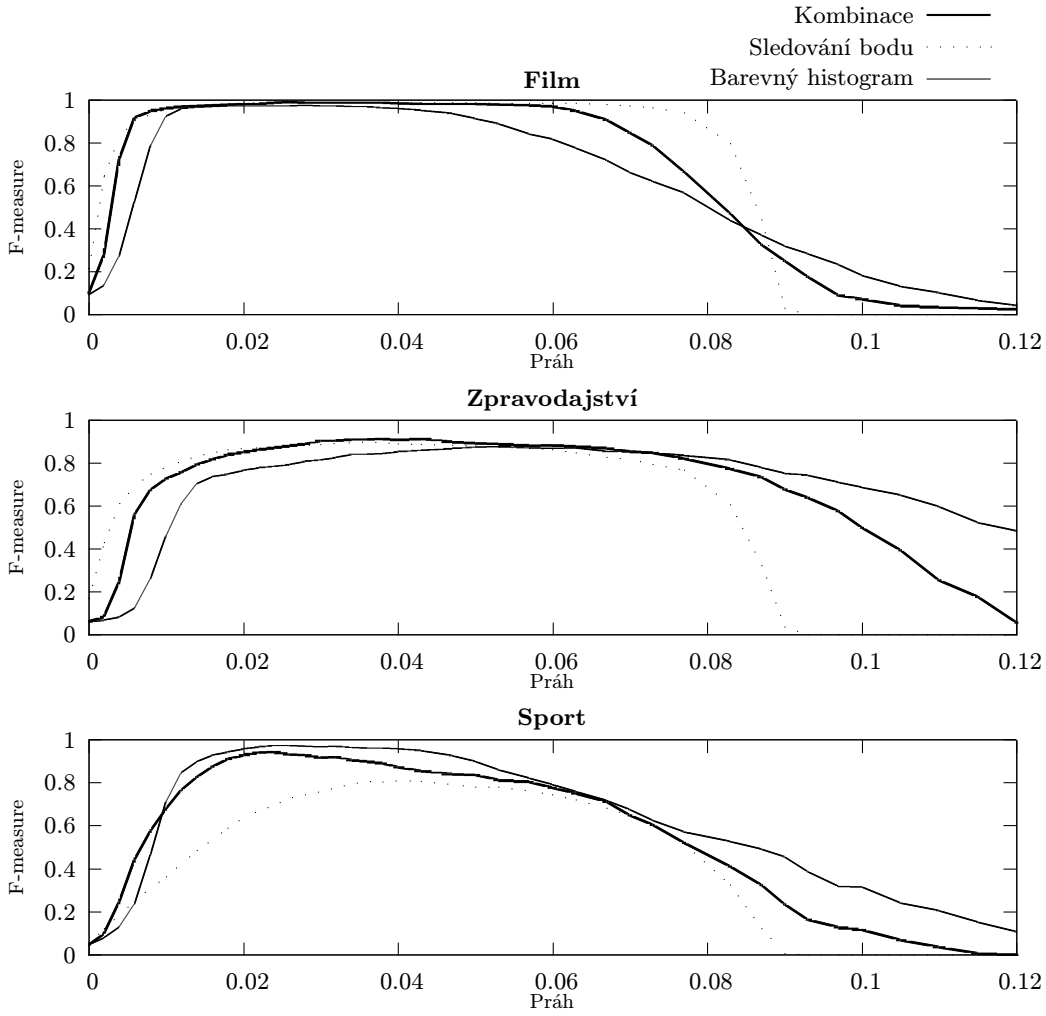
- **Střihy** – detekovaný přechod se musí přímo shodovat právě s jedním z referenčních. Jejich pozice se mohou lišit maximálně o $N = 1$ snímků, z důvodu toho, že pozice může určovat buď poslední snímek starého záběru, nebo první snímek nového.
- **Postupné přechody** – pozice detekovaného přechodu musí být v intervalu určeným začátkem a koncem právě jednoho z referenčních postupných přechodů, a to pouze takového, který ještě nebyl „nalezen“ žádným z předchozích ověřovaných přechodů.

## 6.5 Provedené testy

Testováním je kromě „výkonu“ samotného detektoru nutné zjistit i vhodné hodnoty některých parametrů a jejich vliv na detekci. Implicitně je při kombinaci příznaků použita váha $\alpha = 0.5$, histogramy jsou počítány v prostoru RGB, počet sledovaných bodů je omezen na 500 a inspektor okolí není použit. Protože je možné provádět detekci i za použití pouze jediného příznaku, jsou výsledky uváděny samostatně pro oba příznaky i jejich kombinaci.

### Optimální velikost prahů

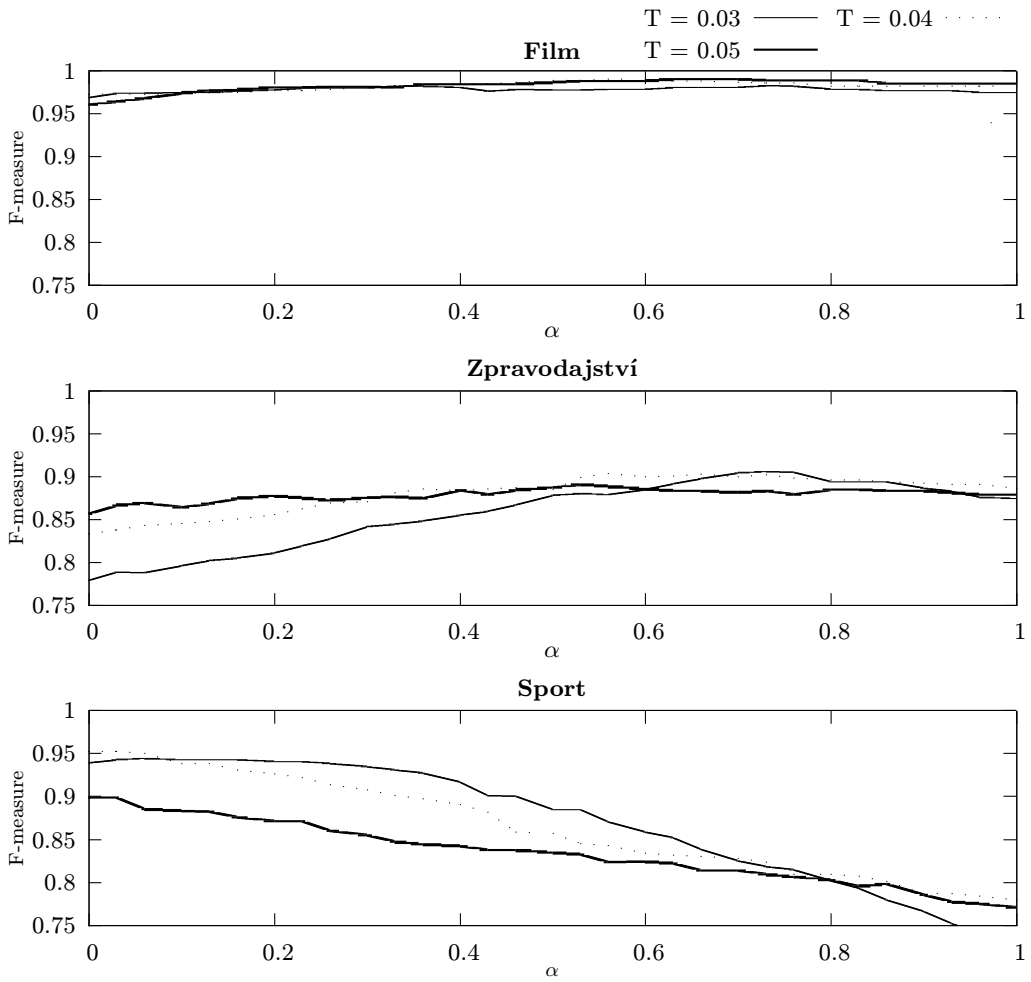
Hlavním parametrem ovlivňujícím detekci je práh, kterým se prahuje odhad derivace analyzovaného signálu. Při nízké hodnotě dojde k falešným detekcím způsobeným šumem, naopak příliš vysoká hodnota prahu může být vyšší než odezva některých přechodů, které pak nebudou nalezeny. Závislost výsledného skóre detektoru na prahu můžeme vidět na obr. 6.1. Z výsledků plyne, že pokud analyzujeme neznámá data, u kterých nemůžeme chování ohadnout, jeví se jako vhodný práh hodnoty v intervalu přibližně 0.03 – 0.05.




Obrázek 6.1: Závislost F-measure na velikosti prahu pro samostatné příznaky a jejich kombinaci.

### Váha příznaků v kombinaci

V kapitole 4.3 byl uveden použitý způsob kombinace příznaků, u kterého je ovlivnitelným parametrem váha $\alpha$. Její hodnoty větší než 0.5 dávají vyšší vliv sledování bodů a naopak hodnoty menší než 0.5 barevnému histogramu. Úkolem je nyní zjistit, jakým způsobem tato váha ovlivňuje detekci. Na obr. 6.2 vidíme závislost skóre detektoru na hodnotě $\alpha$. Z grafů je zřejmé, že tento vliv nelze obecně popsat, protože se liší v závislosti na testovacích datech. To je způsobeno tím, že například sledování bodů funguje hůře při rychlém pohybu, proto u sportu je lepší váhu přidat barevnému histogramu.

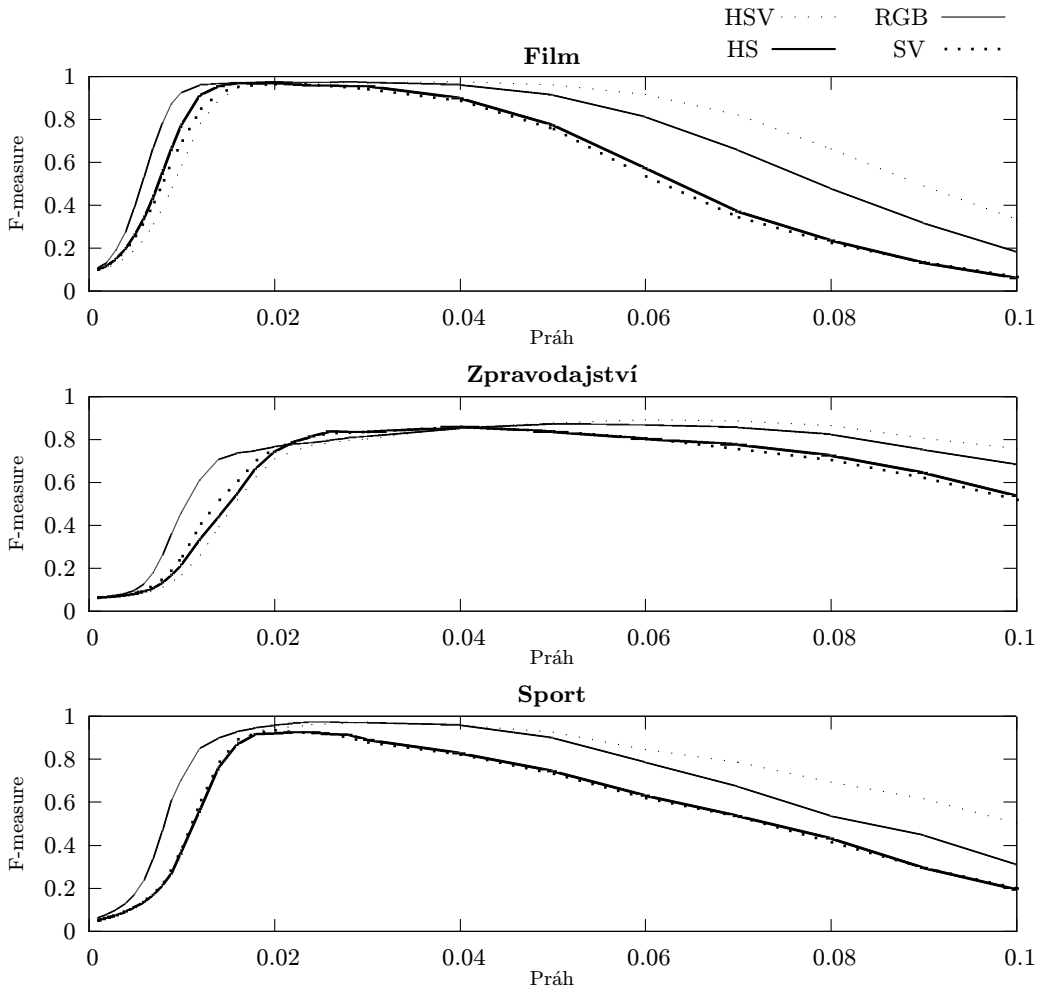



Obrázek 6.2: Závislost F-measure na váze $\alpha$ pro různé hodnoty prahu $T$.

### Vliv barevného prostoru

Cílem tohoto testu je zjistit, zda barevný prostor pro výpočet barevného histogramu ovlivňuje výsledek detekce. Podle [11] by neměl být rozdíl mezi prostory RGB (který bývá použit nejčastěji) a HSV. Zajímavou možností je nevyužití všech kanálů, ale použít pouze některé jejich kombinace – konkrétně HS a SV. Pokud se podíváme na obr. 6.3 zobrazující

závislost výsledného skóre na prahu pro různé barevné prostory, zjistíme, že pro žádná z testovacích dat se výsledky v rozsahu prahů, kdy detektor dává dobré výsledky, příliš neliší. To je v souladu s [11], jejich zjištění navíc rozšiřuji o testy s použitím pouze dvou kanálů z HSV. V takovém případě jsou obecně výsledky mírně horší, ale při vhodném nastavení prahu mohou být pořád velmi dobře použitelné.




Obrázek 6.3: Závislost F-measure na hodnotě prahu při použití pouze příznaku barevného histogramu. Průběhy jsou zobrazeny pro různé barevné prostory, případně pouze některé kombinace kanálů.

## Časová náročnost

Důležitým parametrem je rovněž časová náročnost, tedy jak dlouho analýza dat trvá. Protože to závisí na délce zpracovávaného videa, rozhodl jsem tuto náročnost vyjádřit jako násobek $\gamma$ této délky:

$$\gamma = \frac{\text{Doba trvání analýzy}}{\text{Délka videosekvence}} \tag{6.5}$$

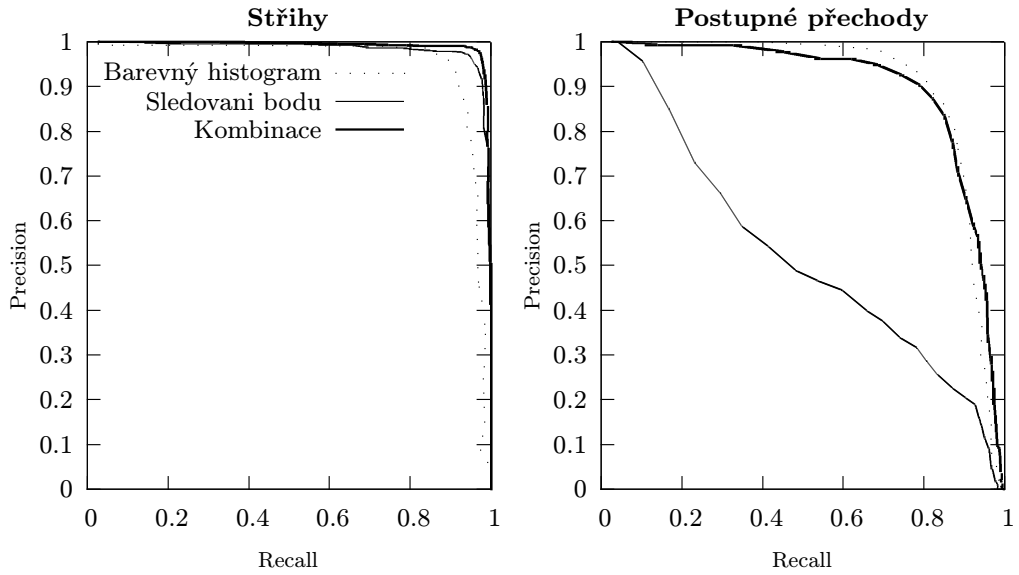
Testy probíhaly na osobním počítači s dvoujádrovým procesorem Intel Core 2 na frekvenci 1,8 Ghz, 2 GB RAM, operační systém Windows 7, 32 bit. Při použití pouze barevného histogramu (barevný prostor RGB) jsem zjistil průměrnou hodnotu $\gamma = 0.38$, tedy výpočet běží rychleji než reálný čas. Pro samostatné sledování bodů při nastaveném limitu 500 bodů jsem naměřil $\gamma = 1.42$. Tato hodnota se snižuje zejména s počtem sledovaných bodů, například pro omezení na 100 bodů klesne na $\gamma = 1.21$, ovlivnit ji lze ale i dalšími parametry trackeru (například velikost okna pro hledání). Pro kombinaci obou příznaků se stejným nastavením vychází průměrně $\gamma = 1.55$.

### Inspektor okolí

Inspektor okolí v testech zatím nebyl použit, aby se ukázaly vlastnosti pouze čistě detektoru bez jiných vlivů. Při testování se ukázal jeho přínos jako značně problematický. Jednak detektor podává dobré výsledky i bez něj, ale zejména (dle očekávání) se zhoršila schopnost detekce postupných přechodů, bohužel k tomu došlo i pro střihy, a to velmi výrazně. Při použitém způsobu porovnání se ukázalo, že rozdíly mezi správnými a falešnými přechody nejsou dostatečně velké, tudíž se mně nepodařilo najít takové parametry, které by zaručily zamítnutí z největší části pouze falešných detekcí. Proto bez uvedení podrobnějších testů tuto součást ponechávám spíše k dalším úvahám a její praktické použití spíše nedoporučuji.

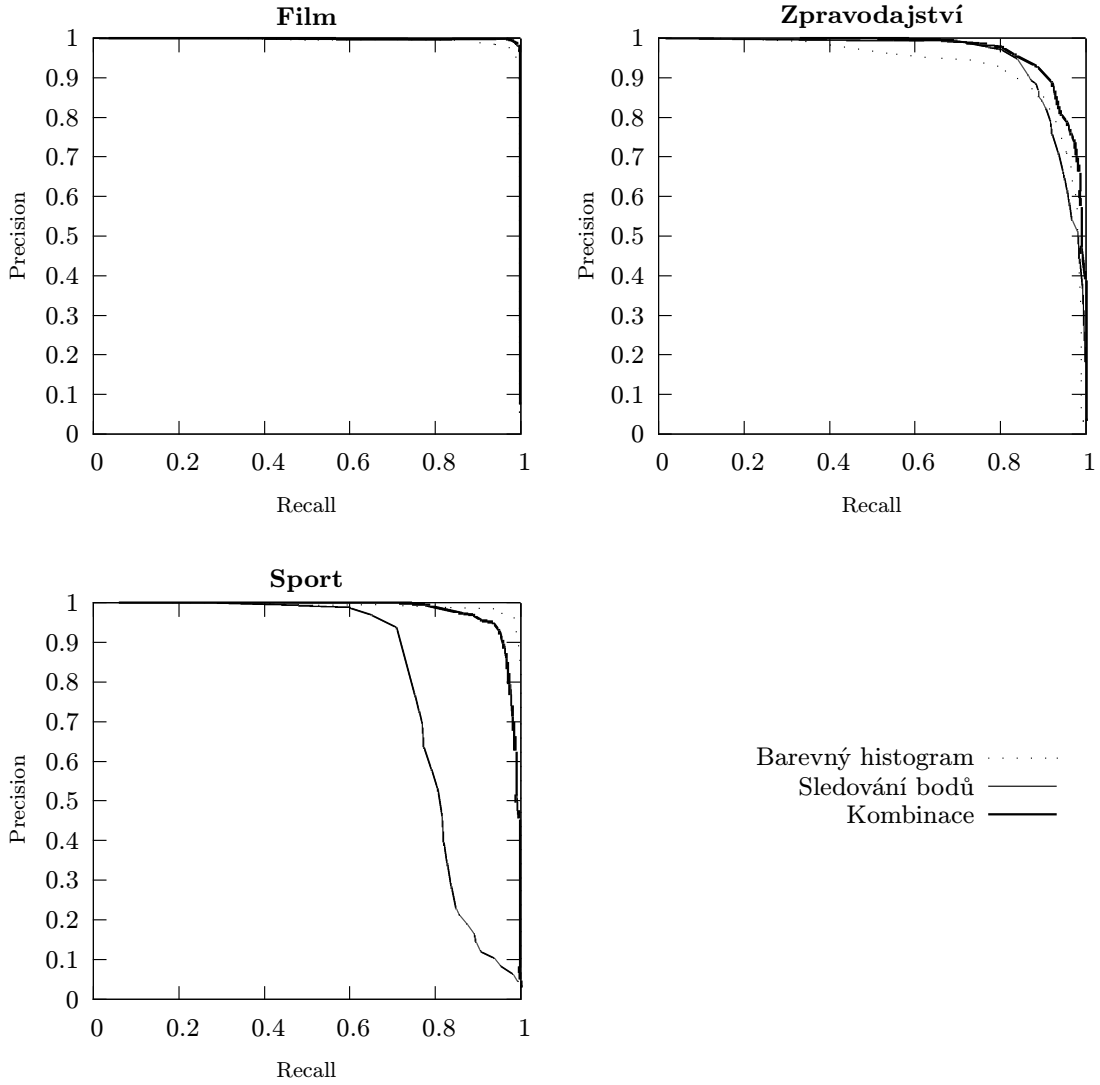
### Úspěšnost a přesnost detekce

Předchozí testy ověřovaly pouze některou z vlastností. Hlavním úkolem je však charakterizovat „výkonnost“ detektoru jako celku – k tomu využijeme právě křivku precision-recall. Při jejich sestrojování jsem měnil hodnotu prahu, spustil detekci a hodnoty jsem zanesl do grafu. Výsledné křivky vidíme na obr. 6.4 a obr. 6.5.




Obrázek 6.4: Křivky precision-recall odděleně pro střihy a postupné přechody. Pro jejich sestrojení byly použity sady „sport“ a „zpravodajství“ spojené jako jedno video. Sada „film“ nebyla použita.

Důležitým zjištěním je, že sledování bodů se nehodí pro analýzu videa, které obsahuje rychlý pohyb. Pokud potřebujeme zachovat vysokou přesnost (hodnotu precision), je nalezeno pouze asi 70 – 80 % přechodů To může být sice považováno za použitelný výsledek, ovšem v kontextu detekce pomocí druhého příznaku a kombinace je vidět výrazný propad. Náročnost zpracování sady „Film“ se ukázala jako nízká, systém je schopen pomocí jakéhokoliv příznaku nalézt více než 98% přechodů při zachování precision velmi blízké hodnotě 1.0. Co se týče zhodnocení celkového „výkonu“ kombinace signálů, dá se říct, že poskytuje dobrý výkon ve všech testovaných datech, tudíž může být označena jako univerzální.




Obrázek 6.5: Křivky precision-recall pro detekci ze samostatných příznaků a jejich kombinaci. Pro jejich sestrojení byly zahrnuty všechny typy přechodů.

Zajímavé je posouzení detekce pro střihy a postupné přechody samostatně. Opět je použita stejná křivka, ale pro výpočet hodnot recall jsou uvažovány pouze počty přechodů (cel-

kový, nalezené) odpovídajícího typu. U výpočtu precision není rozdíl. Také bylo provedeno spojení testovacích sad dohromady a ty analyzovány jako celek – k posouzení zkoumaného chování není třeba je rozlišovat. Výsledky shrnují grafy v obr. 6.4.

Pokud uvažujeme příznaky samostatně, pak lze říci, že pro detekci střihů se více hodí sledování bodů, nicméně rozdíl oproti histogramu není příliš velký. Markantnější propad však lze pozorovat při detekci postupných přechodů pomocí sledování bodů, které tento druh detekuje poměrně špatně. Kombinace příznaků opět poskytuje relativně univerzální chování.

# Kapitola 7

# Závěr

Primárním cílem této práce bylo navrhnout a implementovat systém pro automatickou detekci přechodů záběrů ve videu. Tento úkol se podařilo splnit, přestože představený detektor obsahuje řadu nedostatků. Před návrhem systému jsem nejprve prostudoval dostupnou literaturu z této oblasti, podařilo se mi sestavit definice nejdůležitějších pojmů a seznámil jsem se s nejpoužívanějšími přístupy pro detekci přechodů. Na základě získaných znalostí jsem pak provedl návrh vlastního detektoru.

Obrazové příznaky pro extrakci jsem zvolil dva. Barevný histogram je tradičně používaný, byl vybrán jednak kvůli možnosti srovnání a také vzhledem k jeho poměrně dobrým výsledkům v projektech jiných autorů. Příznak založený na sledování důvodů jsem zvolil zejména pro prozkoumání a demonstraci dalšího způsobu, protože jsem v této podobě jeho použití v jiných pracích nenašel. Rovněž se mi podařilo najít způsob kombinace těchto příznaků tak, aby detekce mohla probíhat zároveň z obou.

Testování proběhlo na vlastní sadě manuálně anotovaných dat. Náročnost těchto dat, zejména sady „film“, se ukázala jako poměrně nízká. V budoucnu by bylo vhodnější detektor vyhodnotit na jiných datech, nejlépe takových, která by mohla poskytnout možnost srovnání s jinými projekty. Výsledky provedených testů posloužily jednak k nalezení vhodných hodnot nejdůležitějších parametrů detektoru, dále se mi podařilo potvrdit tvrzení z [11]. Tedy že vlastnosti detekce pomocí barevných histogramů se příliš neliší pro barevné prostory RGB a HSV, toto zjištění dále rozšiřuji o testy použití pouze kombinace kanálů HS a SV. V takovém případě jsou výsledky mírně horší, ale v rozsahu parametrů, kde detektor má nejlepší výsledky, stále velmi dobře použitelné.

Co se týče celkového posouzení schopnosti detekce, lze říci, že na daných datech si navržený detektor vedl relativně dobře. Systém byl schopen detekovat většinu střihů, a to i při zachování dobré přesnosti. Barevný histogram poskytoval na všech sadách velmi dobré výsledky, sledování bodů vykazovalo mírné problémy při scénách s velmi rychlým pohybem. To se navíc výrazně nehodí pro detekci postupných přechodů typu prolnutí, jejich odezva v tomto příznaku je příliš malá. Kombinace příznaků poskytuje univerzální prostředek, který vykazoval výborné výsledky na všech sadách. Pokud bych měl použít pouze jediný příznak, zvolil bych barevný histogram, nejen vzhledem k nižší výpočetní náročnosti.

Mezi nedostatky metody založené na sledování bodů bych zařadil vysokou náročnost na výkon procesoru danou počtem sledovaných bodů a dále již zmíněnou špatnou schopnost detekce některých postupných přechodů. Nevýhodou barevného histogramu je vyvolávání falešných detekcí například při odpálení blesku fotoaparátu. Tomu jsem se snažil zabránit představeným inspektorem okolí, nepodařilo se mi jej však navrhnout tak, aby zároveň nedošlo k zamítnutí i velké části přechodů správných. Proto jsem se tomuto modulu v popisu

testování příliš nevěnoval a ponechávám jej spíše k dalším úvahám a zkoumání, protože stále věřím, že se jedná o cestu mírného zvýšení přesnosti detekce.

Jako budoucí práci na tomto detektoru bych viděl možnost přidat další příznaky a dále adaptivní váhování těchto příznaků v kombinaci. To by znamenalo, že hodnoty vah jednotlivých příznaků nebudou dány pevně, ale měnily by se například v závislosti na odhadu momentální spolehlivosti detekce z tohoto příznaku. Pokud by tedy aktuální scéna obsahovala velmi rychlý pohyb, váha sledování bodů by se dočasně snížila, protože tento příznak při pohybu funguje hůře. Dalším rozšířením by mohlo být nalezení začátku a konce postupných přechodů, v současném stavu je detekován pouze skokový přechod uvnitř tohoto rozsahu. Pro zvolené druhy přechodů by rovněž mohl být natrénován klasifikátor a výsledkem detekce by nebyla pouze pozice přechodu, ale také jeho typ.

# Literatura


[1] Bimbo, A. D.; Bertini, M.: *Video Automatic Annotation - Shot and Scene detection, Low level content annotation* [online]. [cit. 2010-05-03].
URL http://encyclopedia.jrank.org/articles/pages/6919/Video-Automatic-Annotation.html

[2] Bradski, G.; Kaehler, A.: *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, 2008, ISBN 978-0-596-51613-0.

[3] Cotsaces, C.; Nikolaidis, N.; Pitas, I.: *Video Shot Boundary Detection and Condensed Representation: A Review*.

[4] Davis, J.; Goadrich, M.: *The Relationship Between Precision-Recall and ROC Curves*. In *ICML 06: Proceedings of the 23rd international conference on Machine learning*, Pittsburgh, Pennsylvania, USA, June 25–29, 2006.

[5] Dunn, J.: *Faster smarter digital video*. Microsoft Press, 2002, ISBN 978-0-735-61873-2.

[6] Ewert, R.; Beringer, C.; Kopp, T.; aj.: *University of Marburg: Shot Boundary Detection and Camera Motion Estimation Results*. In *TRECVID 2005 Workshop Notebook Papers, National Institute of Standards and Technology*, MD, USA, 2005.

[7] Frolova, D.; Simakov, D.: *Harris Corner Detector*. Část prezentace z Matching with Invariant Features, The Weizmann Institute of Science, 2004.

[8] Intel Corporation: *Open Source Computer Vision Library – Reference Manual* [online]. [cit. 2010-05-03].
URL http://www.cs.unc.edu/Research/stc/FAQs/OpenCV

[9] Koumaras, H.; Gardikis, G.; Xilouris, G.; aj.: *Shot boundary detection without threshold parameters. J. Electron. Imaging, Vol. 15*, 2006.

[10] Lienhart, R.: *Reliable Transition Detection in Videos: A Survey and Practitioners Guide*, ročník 1. Citeseer, 2001.

[11] Mas, J.; Fernandez, G.: *Video Shot Boundary Detection Based on Color Histogram*. In *TRECVID 2003 Workshop Notebook Papers, National Institute of Standards and Technology*, MD, USA, 2003.

[12] Ngo, C.-W.: *A Robust Dissolve Detector by Support Vector Machine*. In *Proceedings of the eleventh ACM international conference on Multimedia*, Berkeley, CA, USA, 2003.

[13] Potúček, I.: *Sledování pohybu objektů v obraze.* Brno, diplomová práce, FIT VUT v Brně, 2002.

[14] Rijsbergen, C. J. V.: *Information retrieval.* Butterworths, 1979, ISBN 0-408-70929-4.

[15] Robles, O. D.; Toharia, P.; Rodríguez, A.; aj.: *Using Adaptive Thresholds for Automatic Video Cut Detection.* 1995.

[16] Shahraray, B.: *Scene change detection and content-based sampling of video sequences.* In *Digital Video Compression: Algorithms and Technologies, Proc. SPIE*, San Jose, CA, USA, February 07, 2006.

[17] Shapiro, L.; Stockman, G.: *Computer Vision.* Prentice Hall, 2003, ISBN 978-0-130-30796-5.

[18] Smeaton, A.; Over, P.; Doherty, A.: *Video Shot Boundary Detection: Seven Years of TRECVid Activity. Computer Vision and Image Understanding*, 2009, ISSN 1077-3142.

[19] Svoboda, T.: *Kanade-Lucas-Tomasi Tracking.* 2008, prezentace k předmětu Robotika a strojové vnímání, Czech Technical University in Prague, Center for Machine Perception, Praha, ČR.

[20] Welch, G.; Bishop, G.: *An Introduction to the Kalman Filter.* 1995.

[21] Zabih, R.; Miller, J.; Mai, K.: *A Feature Based Algorithm for Detecting and Classifying Scene Breaks.* In *Proc. ACM Multimedia 95*, San Francisco, CA, 1995.

[22] Zuech, N.; Miller, R.: *Machine Vision.* Springer, 1989, ISBN 978-0-442-23737-0.