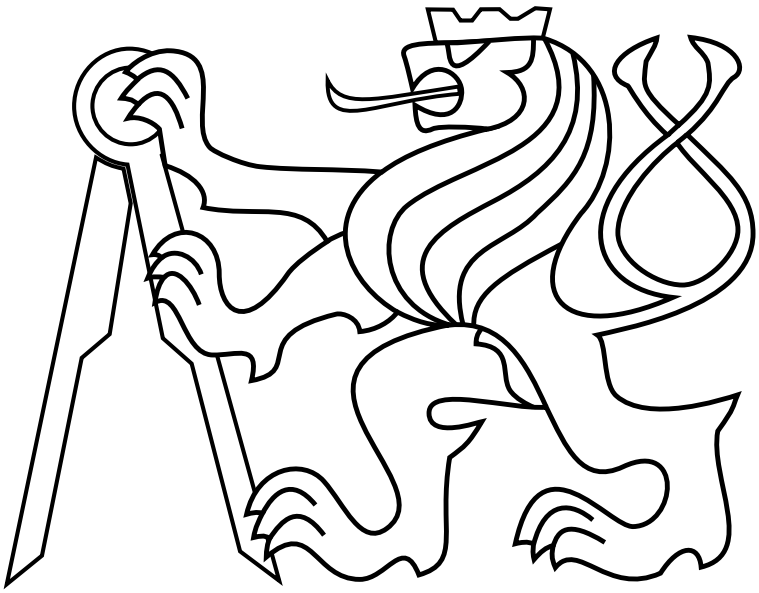
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

DIPLOMOVÁ PRÁCE



Bc. Tomáš Suchomel

Automatická produkce videozáznamu přednášky

Automated Production of Lecture Recordings

Katedra radioelektroniky

Vedoucí práce: **Ing. Stanislav Vítek, Ph.D.**

Praha 2019

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Suchomel** Jméno: **Tomáš** Osobní číslo: **406078**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra radioelektroniky**

Studijní program: **Elektronika a komunikace**

Studijní obor: **Audiovizuální technika a zpracování signálů**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Automatická produkce videozáznamu přednášky

Název diplomové práce anglicky:

Automated Production of Lecture Recordings

Pokyny pro vypracování:

Navrhněte a implementujte systém pro automatickou produkci videozáznamu přednášky. V rámci práce řešte následující úkoly:
1. navrhněte popis obrazového řešení, vhodný pro metody strojového učení
2. analýzou existujících videozáznamů stanovte pravidla vhodná pro automatizaci střihu
3. navrhněte a implementujte systém ovládání kamery, který bude aktivně aplikovat pravidla získaná v bodech (1) a (2)

Seznam doporučené literatury:

[1] FANG, Chiung-Yao, et al. Building a smart lecture-recording system using MK-CPN network for heterogeneous data sources. Neural Computing and Applications, 2018, 1-19.
[2] CHOU, Han-Ping, et al. Automated lecture recording system. In: System Science and Engineering (ICSSE), 2010 International Conference on. IEEE, 2010. p. 167-172.
[3] LIAO, Hsien-Chou, et al. An Automatic Lecture Recording System Using Pan-Tilt-Zoom Camera to Track Lecturer and Handwritten Data. International Journal of Applied Science and Engineering, 2015, 13.1: 1-18.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Stanislav Vítek, Ph.D., 13137

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **12.02.2018** Termín odevzdání diplomové práce: **08.01.2018**

Platnost zadání diplomové práce: **30.09.2019**

Ing. Stanislav Vítek, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. | Datum převzetí zadání

Podpis studenta

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem č. 1/2009 o etické přípravě vysokoškolských závěrečných prací.

V Praze dne 8. ledna 2019 .....

podpis

Rád bych tímto poděkoval panu Ing. Stanislavovi Vítkovi, Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích během vypracování této práce. Dále také své rodině a svým kamarádům za podporu po celou dobu studia.

Abstract

In today's world of the Internet, there is a lot of lectures, presentations, conferences and various discussions that are recorded and later provided to the wide public with a possibility to watch the video repeatedly. The demand for a professional record is increasing, therefore it is necessary to introduce a certain automation of the process to lower the costs of the final editing. This is the topic of the thesis below.

Index terms: lecture-recording, automatization, face tracking, Dlib, OpenCV, MFCC, mel-cepstral analysis, cinematographic rules

Abstrakt

V dnešní době internetu je mnoho přednášek, prezentací, konferencí či nejrůznějších diskuzí zaznamenáváno a dále poskytováno k možnosti opakovaného shlédnutí širokou veřejností. Poptávka po pořízení profesionálního záznamu stále roste, proto je nutné zavedení určité automatizace procesu pro snížení nákladů výsledného střihu. Právě tomuto problému je věnována následující práce.

Klíčová slova: nahrávání přednášek, automatizace, sledování obličeje, Dlib, OpenCV, MFCC, mel-kepstrální analýza, kinematografická pravidla

Obsah

Seznam obrázků

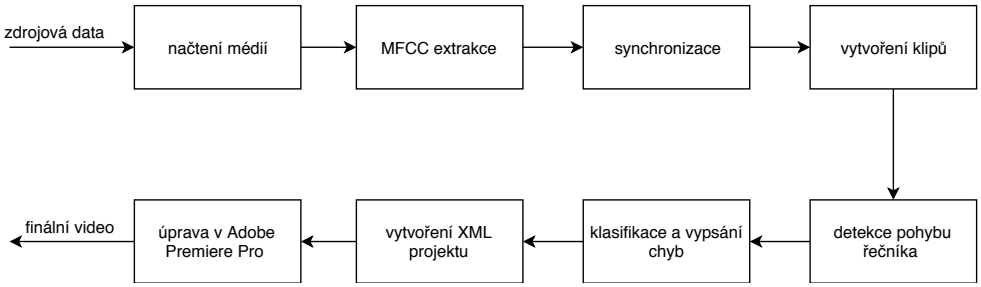
1 Úvod

V dnešní digitální době je mnoho přednášek, prezentací, konferencí či nejrůznějších diskuzí zaznamenáváno formou videa, které je dále poskytováno prostřednictvím internetu k možnosti opakovaného shlédnutí širokou veřejností. Tématu takového nahrávání se věnuje i tato práce. Proces, kterým musí veškerá data od činnosti nahrávacího týmu po výsledný střih projít, je velmi komplexní. Zahrnuje nejen samotné natáčení týmem kameramanů, ale zaznamenání zvuku řečníka a jeho prezentace samotné. Dále je nutné všechna tato data určitým způsobem zkompletovat, tedy složit do jedné časové osy a připravit střihačovi k finálnímu střihu. Ten musí celý záznam projít, a prostříhat mezi jednotlivými kamerami tak, aby odstranil veškeré chyby, které mohou přirozeně během záznamu vzniknout a zároveň, pomocí určitých pravidel, výslednému sestřihu dodal dynamičnost a celkovou atraktivitu, která budoucímu divákovi zaručí ničím nerušený vjem.

Téma plně autonomního systému pro nahrávání přednášek je poměrně dlouho řešený problém, kterému se stále věnuje více projektů. Obecným cílem těchto projektů je plná automatizace celého procesu, tedy s použitím plně robotických kamer řízených příkazy z centrálního výpočetního systému. Během řešení této práce však bylo zjištěno, že v dnešní době, se současnými technologickými možnostmi, autonomní nahrávací systém stále nemůže dosahovat kvality živého kameramana a jeho profesionální kamery či práce střihače a reálně tak nelze zajistit plnohodnotný divácký vjem. Cílem této práce tedy není vytvoření dalšího plně autonomního systému, ale automatizace dílčích částí procesu zpracování, které výrazně přispívají k jeho zjednodušení se zachováním stejné kvality. Následující kapitoly se věnují dvěma hlavním tématům. Prvním je popis a zpracování vstupních dat, tedy natočeného audiovizuálního materiálu získaného od kameramanů, která jsou následně automaticky synchronizována a převedena do projektu připraveného pro práci střihače. Druhým tématem je snaha o aplikaci kinematografických pravidel na natáčení přednášek a jejich částečná aplikace na autonomní systém detekce chyb v obraze.

Celý tento proces popisuje schéma na obrázku číslo 1. Popisu zdrojových dat a jejich zpracování, tedy načtení všech vstupních médií, se věnuje kapitola 2 a 3. Proces extrakce Mel-kepstrálních koeficientů, MFCC, jejich následné synchronizace a správného utřídění do jednotlivých klipů je dále popsán v kapitole 4. Kapitola 5 se pak věnuje rozboru ki-

nematografických pravidel a možnosti jejich aplikace na detekci kompozičních chyb, která je dále popsána v kapitole 6. Podoby zpracování výstupních dat, projektu určeného pro finalizaci záznamu ve střihovém programu a výsledků testování jsou popsány v poslední kapitole, tedy kapitole 7.



Obrázek 1: Schéma procesu.

1.1 Shrnutí současného stavu

Řešení plně autonomních systémů je postaveno na takzvaném „virtuálním režisérovi“ a „virtuálním kameramanovi“. Virtuální režisér řídí virtuální kameramany, tedy robotické kamery typu Pan-Tilt-Zoom, zkratkou PTZ, s možností horizontálně-vertikálního pohybu a proměnným zoomem. Jsou dva způsoby, kterými takový režisér může informace o dění v přednáškové místnosti získat. Jedním je čisté zpracování obrazu, druhým použití externích senzorů. Takovým senzorem může být například takzvaná hloubková kamera, která vytváří hloubkovou mapu sledovaného prostředí. Z té pak lze vyčíst model sledované osoby a na základě jejího pohybu zadávat příkazy jednotlivým virtuálním kameramanům. Takovým senzorem je například Kinect[1] firmy Microsoft, který pro svou práci využívá projekt „Building a smart lecture-recording system using MK-CPN network for heterogenous data sources“[9], 2018. Hloubkový senzor zde slouží především k získání modelu člověka, který je kombinován s klasickým sledováním obličeje, detekovaného pomocí HAAR kaskádních stylů algoritmem Viola-Jones[21] a trackovaného metodou mean-shifting. Kombinací těchto dat je získán jakýsi model řečníka s detekcí jeho gest, která jsou jedním z hlavních rozhodovacích parametrů pro výsledný střih mezi kamerami.

Dalším příkladem je starší projekt „Automated lecture recording system“[7] z roku

[1] https://developer.microsoft.com/cs-cz/windows/kinect

2010. Ten používá pouze jednu kameru, jejíž obraz dále zpracovává a určuje z něj nejen pozici obličeje řečníka, ale i pozici plátna s prezentací. Díky těmto informacím pak dokáže měnit záběr kamery v závislosti na chování a pohyby řečníka. Například pokud se řečník pohybuje směrem k plátnu, kamera oddálí záběr, aby pokryla jak řečníka, tak plátno, na kterém bude řečník pravděpodobně něco ukazovat.

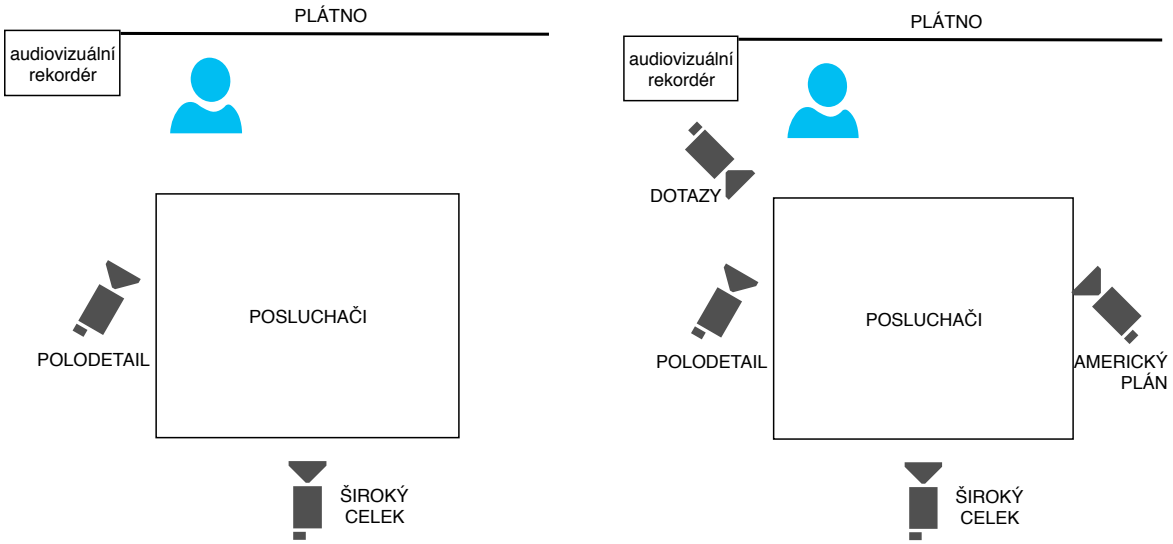
Za zmínku stojí především projekt „Realtime multi-person 2d pose estimation using part affinity fields“[6], který byl na počátku řešení práce také testován a měl se stát primárním obrazovým detektorem. Na základně naučených konvolučních neuronových sítí rozeznává dané části lidského těla, především kloubní, a výrazné části hlavy, jako jsou oči, ústa a uši. Dále porovnáváním toku lokálních histogramů v obraze dokáže získané body správně sestavit do jednotné kostry člověka i při zhoršených podmínkách, například při překrývání jedné osoby osobou druhou. Jeho řešení však nebylo možné pro tento účel použít z důvodu velké výpočetní náročnosti díky které by byl celkový výpočet pohybu řečníka, byť s integrací výkonné grafické karty, velice pomalý a tak pro řešení této úlohy nepoužitelný.

2 Popis vstupních dat

Jak již bylo zmíněno v úvodu, práce je určená nahrávání pro rozličných přednášek, konferencí a podobných prezentací. To má oproti zpracování záznamů pořizovaných v akademickém prostředí, kde se využívá především místních integrovaných audiosystémů, za následek velkou různorodost výsledných dat pro samotnou postprodukci. Ať se jedná o různý počet kamer použitých pro nahrávání, jejich typ, způsob nabrání zvukového záznamu či samotné rozložení kamer. Z tohoto důvodu je primárním cílem práce vytvořit univerzální systém, který si poradí s jakýmkoliv, předem nespecifikovaným, typem vstupních dat.

2.1 Nahrávací proces

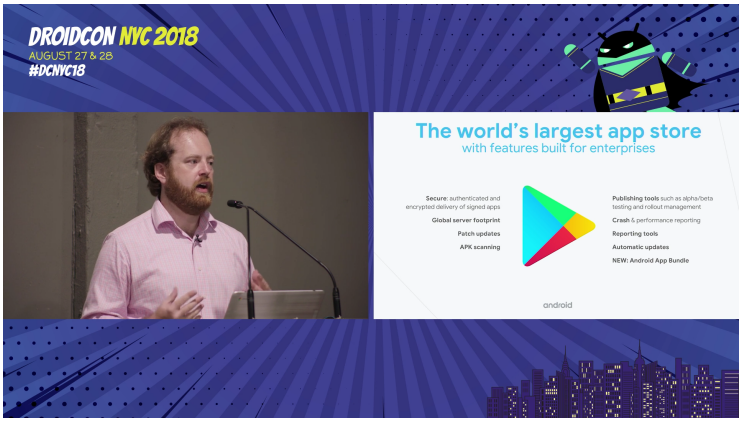
Na základě praxe získané prací pro nejmenovanou firmu věnující se právě nahrávání a následnému zpracovaní konferenčních přednášek je možné definovat základní typy rozvržení nahrávacího systému, které demonstruje obrázek číslo 2. Prvním typem je konfigurace typická pro většinu případů, která pojímá jednu hlavní kameru ovládanou kameramanem, jednu statickou záložní kameru a zvukové nahrávadlo snímající zvuk ze zvukového mixážního pultu či přímo z bezdrátového mikrofonu řečníka. V takovém případě nahrává hlavní kamera takzvaný polodetail řečníka, tedy jednoduše řečeno záběr od pasu nahoru, anglicky „closeup“ a záložní kamera velmi široký detail zvaný „wideshot“. Detailnějšímu popisu samotných typů používaných kompozic se více věnuje kapitola 5.1.1. Hlavní zvukový záznam je nahráván příslušným softwarem do počítače pomocí zvukové karty. Předpokladem k této práci je, že veškeré kamery použité pro nahrávání snímají zvuk ať už přímý z mixážního pultu anebo zvuk ze svého zabudovaného mikrofonu. Pro konference vyžadující dynamičtější záznam je možné navýšení počtu kamer například o kameru snímající záběr typu Americký plán a kameru nahrávající publikum a jejich dotazy. Nahrávání audia a prezentace je pak prováděno stejným způsobem jako u základního typu. Při postprodukci je nutné záznam zkontrolovat a podle předem daných pravidel vztahující se k typu přednášky záznam z jednotlivých kamer sestříhat a mimo jiné tak eliminovat veškeré defekty způsobené při samotném nahrávání. Jedná se především o chyby lidského faktoru, jako například tvoření špatné kompozice, třes kamery, rozostřený záběr a další



Obrázek 2: Demonstrace základního a pokročilého rozložení kamer.

chyby. Mimo jejich eliminace střihač vnáší do sestřihu svůj umělecký pohled, takzvaný rukopis střihače, kterému se však tato práce nevěnuje.

Výstupním formátem pro tuto práci je takzvané „Picture in Picture“, tedy česky obraz v obraze. Jedná se o rozložení, kdy jsou v předem dané grafické šabloně zmenšené oba obrazy - jak video řečníka, tak záznam prezentace samotné. Ukázkou může být například záznam konference Droidcon 2018[2].

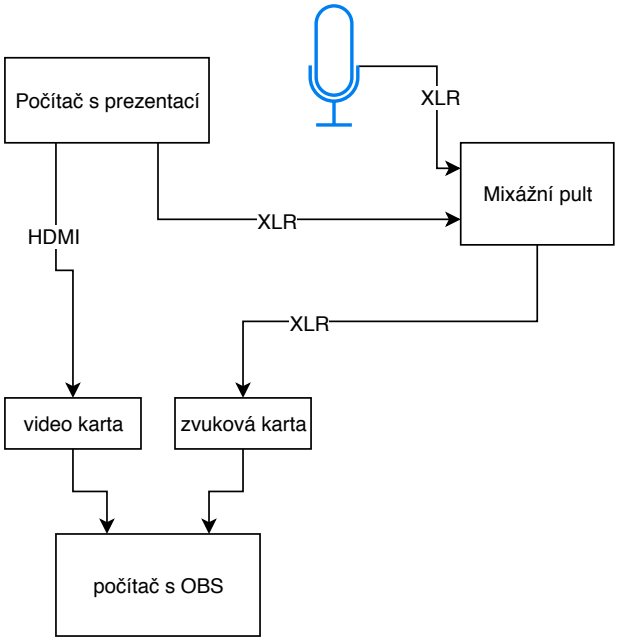


Obrázek 3: Ukázka výstupního formátu Picture in Picture.

[2]https://www.youtube.com/watch?v=kG34wec5uSQ

2.2 Záznam prezentace

Záznam obrazové prezentace je v této práci prováděn zdvojením signálu vedoucího z prezentujícího počítače na projekci a jeho následné zaznamenávání. Možností jak tento krok provést je mnoho. Lze využít buď přímo hardwarových nahrávadel nebo, jako v případě této práce, obraz nahrávat spolu se zvukem softwarově pomocí počítače s připojenými potřebnými periferiemi. Toto zapojení popisuje obrázek číslo 4. Pro tento účel byl zvolen program Open Broadcaster Software Studio[3], pomocí kterého lze nahrávat jak obrazový vstup, tak vstup zvukový. Obraz je do počítače připojen pomocí grabovací karty s HDMI vstupem. Zvuk pak připojen obdobně pomocí externí zvukové karty s XLR vstupem.



Obrázek 4: Schéma zapojení hardwaru pro záznam prezentace.

2.3 Práce s daty

Veškerá data jsou v rámci nahrávacího procesu bezpečně zálohována a uložena na centrální diskové pole. Pro správné fungování, přehlednost a jednoduchost byla navržena struktura pro samotné třídění dat se kterou dále systém pracuje. Veškerá data jsou proto dělena po tzv. sálodnech, tedy záznamech pro jeden sál za jeden den. V této mateřské složce

sálodne jsou přímo vnořeny podsložky daných zdrojů podle dělení popsaného v nahrávacím procesu 2.1, například AUDIO pro veškeré audio nahrávky, CAM1 pro zálohování dat ze všech paměťových nosičů kamery 1, CAM2 v případě užití další pohyblivé kamery, WIDESHOT pro zalohování dat statické záložní kamery se širokým záběrem. Další v praxi užívané podsložky systém pro svůj chod nevyžaduje, proto není nutné se jim v této práci věnovat.

Výchozí podmínkou při ukládání dat do těchto podsložek je jejich chronologické třídění. To je nutné nejen pro správné zarovnání dat ve výstupním projektu, ale mimo jiné také pro detekci kontinuálního video záznamu uloženém na více než jednom paměťovém médiu, jehož detekce je nutná například u kodeku AVCHD s MPEG transportním streamem.

3 Příprava dat

Pro jakékoliv operace s tak velkým objemem dat, jako jsou videozáznamy přednášek, je vhodné zavedení centrálního datového pole. Toto pole napojené na jednoduchý operační server musí splňovat určité předpoklady. Především je to velmi dobrá konektivita a vysoká rychlost čtení a zápisu na disku. Datové pole pak slouží jako centrální shromaždiště dat jak pro kameramany, tak pro postprodukční systémy.

3.1 Zpracování vstupní složky

Vzhledem k tomu, že každý výrobce nahrávacích zařízení používá svou vlastní strukturu pro ukládání dat na paměťovém médiu, které se dál dělí podle použitého kódovacího formátu, je nutné na data pohlížet jako na neuspořádanou směs různých audiovizuálních vstupů. Tento fakt umocňuje chyba lidského faktoru při zálohování dat, jako je například nepřesně zkopírovaná adresářová struktura, mylné vytváření vlastních složek a podobně. Program tedy načítá a zpracovává každý soubor odděleně a souvislosti mezi nimi hledá až zpětně na základě určitých pravidel popsaných v sekci 3.4. Celý proces zpracování vstupních dat řídí objekt MediaLoader, který nejdříve pro každé zdrojové zařízení, tedy všechny kamery, audio a záznam přednášek, projde adresářovou strukturu a spustí danou

funkci na každý soubor s podporovanou příponou. Pro takovou práci s cestami souborů je nejvhodnější knihovna os, která je součástí většiny distribucí programu Python. Pro prohledávání adresářů včetně všech vnořených složek a souborů slouží funkce *walk*, jak je uvedeno v následující ukázce kódu 1. Podporovanými příponami a tedy formáty videí jsou nejčastěji používané datové typy: MTS/M2TS, MP4, MXF, MOV, WAV a MP3.

```
import os
tree_generator = os.walk(recordings_directory)
for (root, directories, filenames) in tree_generator:
   for filename in filenames:
      _ , extension = os.path.splitext(filename)
      if extension in known_formats:
         file_path = os.path.join(root, filename)
         MediaLoader.load(file_path)
```

Výpis 1: Ukázka cyklu pro vyhledání zdrojů.

3.2 Metadata

Každý soubor, ať se jedná o video či zvuk, obsahuje určitá metadata, popisující vlastnosti souboru, jeho strukturu, pomocí jakého kodeku byl soubor kódovaný, s jakým nastavením a podobně. Metadata lze přečíst mimo jiné například pomocí open-source programu ffprobe, součástí projektu skupiny FFmpeg[18]. V případě zpracování video zdrojů lze podle metadat stanovit pomocí jakého kodeku bylo video nahráváno a tedy jaká metadata mají být očekávána a jak je správně přečíst. Tato informace je důležitá především pro detekci možného AVCHD streamu, popsaného v sekci 3.4.
Pro celkový chod programu jsou důležité především parametry:

- formát kódování videa
- čas pořízení souboru
- celková doba trvání
- snímková frekvence obrazu a tedy zda se jedná o typ NTSC či PAL

- formát kódování audio stopy
- vzorkovací frekvence audia
- doba trvání audia
- synchronizační posun audio a video stopy

Mimo již popsaný formát audia a videa je potřebné znát jeho celkovou délku pro zjištění návazností dat a jejich následnou synchronizaci, vzorkovací a snímkovou frekvenci, a u AVCHD formátu s MPEG transportním streamem (MTS) také informaci o posunu audia vůči videu, které vychází z podstaty streamu. Na proces je nasazen jednoduchý validační filtr, který zahazuje veškeré soubory kratší než 10 sekund, tedy různé testovací záznamy apod. Veškerá metadata zdrojových dat, která prošla validačním filtrem jsou uložena do centrální databáze projektu. Čas pořízení je přepočítán na UTC formát.

3.3 Linkování audia

Ve strukturách jednotlivých kamerových zdrojů, tedy složkách CAM1, CAM2 a podobně lze předpokládat, že nedojde k paralelním zdrojům. Tedy, že každý soubor ve struktuře je v čase unikátní. Toto pravidlo však neplatí pro zvuk. Zde je možné nahrávat paralelně několik kanálů najednou, více zařízení a podobně. Z tohoto důvodu veškeré zvukové soubory prochází takzvaným linkováním. Tedy procesem, kdy program zjišťuje, zda se v okolí souboru nenachází soubor se stejným časem pořízení, shodnou délkou a shodným počtem vzorků. Pokud jsou takové soubory nalezeny, program je označuje jako slinkované, tedy se dále zpracovávají dohromady. To je důležité hlavně pro případ zmíněného vícekanálového nahrávání. Dejme tomu, že v jednom kanále je nahráván řečník A a v druhém mikrofon řečníka B. Tedy korelace mezi oběma signály nikdy neproběhne, protože mezi sebou neobsahují žádný shodný signál. Tedy by je program vyhodnotil jako časově odlišné.

3.4 Detekce video streamů

Dalším procesem v rámci přípravy dat je detekce streamů. Tyto streamy se týkají především záznamů formátu MTS kodeku AVCHD. Obrazový záznam je v takovém případě segmentován na dílčí soubory dané velikosti. Podmínka pro detekci takového streamu zní volně takto: segmenty jednoho MTS streamu jsou všechny soubory, které leží v dané složce, mají 5 ciferný název (n), každý po sobě jdoucí soubor má název roven (n+1). Zároveň je nutnou podmínkou, aby rozdíl zpoždění zvuku druhého souboru mínus doba trvání souboru prvního vůči zpoždění zvuku prvního byl menší než doba možná na manuální restartování záznamu, odhadem menší než 500ms.

Segmenty streamu se nemusí nacházet pouze v jedné adresářové úrovni, ale mohou být rozděleny do více struktur. Takovým případem jsou především data z kamer, které mají dvě paměťová média, přičemž pokud se jedno naplní, zápis se automaticky přesune na médium druhé, mezi tím lze vyměnit médium první a po naplnění druhého opět záznam přesunout. Celý tento kontinuální záznam musí být správně detekován především z důvodu správného napojení dat bez ztráty vzorků zvuku.

4 Synchronizace obsahu

Jednou z hlavních částí této práce je správné utřídění a zarovnání nahraných dat do časové osy tak jak byl záznam pořizován. V ideálním případě, kde by byly veškeré kamery naprosto shodné se shodně nastaveným časovým kódem, anglicky „timecode“, nebyl by problém data jednoduše setřídit právě podle něj. Avšak tento stav v praxi téměř nikdy nenastává. Jak již bylo zmíněno v úvodu práce, systém má být absolutně univerzální pro veškerá vstupní data velmi rozličných typů, proto je nutné problém řešit poněkud složitější cestou a to vzájemnou synchronizací všech dat podle jejich zvukové stopy.

Díky faktu, že obsahem přednášek je především mluvené slovo, jsou tato zvuková data dále upravována a zjednodušována pomocí metod pro zpracování řeči, konkrétně kepstrální analýzou a váhováním Melovskou bankou filtrů. Důvody, proč není vhodná synchronizace původních dat jsou prosté. Po odstranění redundantní informace získáváme pouze ta data, která jsou pro synchronizaci potřebná. Výstupem této operace je tak pouhá jedna křivka

o výrazně nižším počtu vzorků než je u původního signálu. Tím je získána mnohonásobně vyšší výpočetní rychlost následné synchronizace se zachováním stejné přesnosti nebo dokonce vyšší.

Pro veškerá data jsou pak po párech hledané časové průniky a offsety souborů pomocí jejich vzájemné korelace. Nalezené vztahy dvojic jsou pak dále příslušně upraveny a pomocí rekurentních operací je nalezen absolutní počátek souborů v čase, tedy první nahraný soubor v pořadí. Vůči němu jsou pak vypočtené pozice všech souborů v databázi a celý záznam seřazen.

4.1 Příprava dat

Data v databázi obsahují jak zvuková tak obrazová data. Jak je popsáno v kapitole 3, oba typy dat mohou mít různé vzájemné vazby. Například zvuk je nahráván ve více paralelních kanálech, záznam obrazu je segmentován na jednotlivé části, tedy je znám jejich vzájemný offset, a tak dále. Díky těmto informacím lze výpočet zjednodušit a zpřesnit.

V první fázi je nutná extrakce zvukové stopy z veškerých dat v databázi se zachováním zmíněných vazeb. Vzhledem k tomu, že může nastat případ, kdy v rámci jednoho souboru řečník mluví do mikrofonu A, který je ukládán v kameře nebo zvukovém nahrávadle do kanálu 1 a moderátor s mikrofonem B je ukládán do kanálu 2 stejného zařízení, musí být výstupem extrakce jeden mono signál sloučením všech nalezených kanálů daného zařízení. Byť se v takovém případě jedná o paralelní záznam v rámci jednoho zařízení, signály jsou nekorelovatelné, protože nedochází k žádnému zvukovému překryvu signálu A se signálem B. Proces hledání a popisu takového slinkování je popsán v kapitole 3.3.

Pro tuto práci je do projektu implementován otevřený program FFmpeg [18], který obsahuje širokou škálu modulárních audiovizuálních filtrů včetně míchání zvukových kanálů z více zdrojů potřebného pro tento úkon. K tomu slouží níže uvedený příkaz 2, který je spouštěn pomocí Python knihovny *subprocess*.

```
ffmpeg -i "ch1.wav" -i "ch2.wav" -i "ch3.wav" -filter_complex
    "[0:a][1:a][2:a]amerge=inputs=3[aout]" -map "[aout]" -acodec pcm_s16le -ar
```

```
24000 -ac 1 "output.wav"
```

Výpis 2: Ukázka struktury položky clipitem.

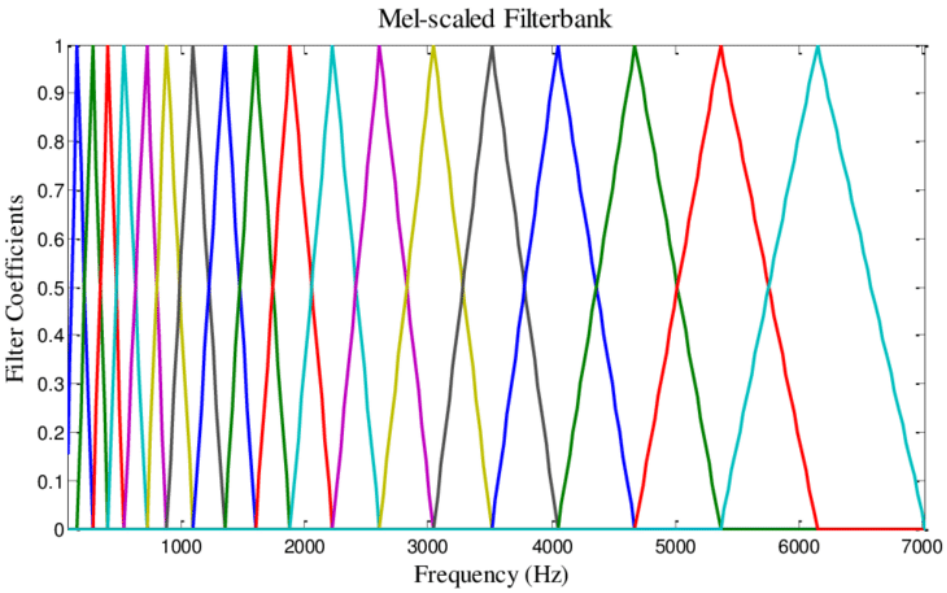
Příkaz je generován v závislosti na skladbě vstupních signálů, jak počtu slinkovaných souborů, tak počtu jejich kanálů. Uvedený příklad popisuje extrakci jednoho zvukového souboru ze tří audio stop paralelně nahrávaných do separátních souborů. Na počátku jsou definovány všechny tři vstupy parametrem „-i“. Pro sloučení kanálů je využito kombinačního filtru „-filter_complex“. Mapování vstupů je definováno argumenty na počátku, například [0:a], který připojuje audio stopy z prvního souboru, [1:a] audio stopy z druhého vstupu atd. „amerge“ dále spojuje tyto tři vstupy a směruje celé jako výstup „aout“, který je argumentem „-map“ složen. Jako další modul v procesu je přidán kodek pcm_s16le pro zakódování výstupního zvuku s bitovou hloubkou 16bit a vzorkovací frekvencí 24kHz. Argumentem „-ac“ je pak definován počet výstupních kanálů zvuku, zde 1, tedy mono. Poslední argument pak definuje cestu výstupního souboru.

Cesty k těmto souborům, při výchozím nastavení ukládané do dočasné pracovní složky, jsou zaznamenány do databáze k příslušným médiím. Tento mezistav uložení souboru na pevný disk počítače je především pro získání modulárnosti celého systému. Lze pak provádět výpočet po daných blocích, tedy nejprve extrakce všech zvuků a převedení do mono souborů, až poté výpočet kepstrálních koeficientů popsaného v další sekci. Díky tomu tak lze výpočetní výkon lépe distribuovat na procesoru či mezi více výpočetních stanic.

4.2 Mel-kepstrum signálu

V úvodu kapitoly bylo zmíněno, že zpracovávaným zvukem je řeč. Na tomto faktu je také závislé celé následné zpracování. Jednou z vlastností řeči je její nelineární rozložení ve spektrální oblasti. Z toho důvodu je také nutné využití nelineární banky filtrů, kterou je tento signál rozdělen do frekvenčních pásem pro další zpracování. Nejvhodnější způsobem takového dělení je pro tento účel využití melovské banky filtrů. Právě toto rozložení aproximuje vnímání frekvence lidským sluchem. Jejich výstup je pak základem pro výpočet Mel-kepstrálních koeficientů. Lineární rozložení melovské banky filtrů je znázorněno na obrázku 5

Banka obsahuje 128 pásem s filtry trojúhelníkového tvaru, které mají stejnou šířku



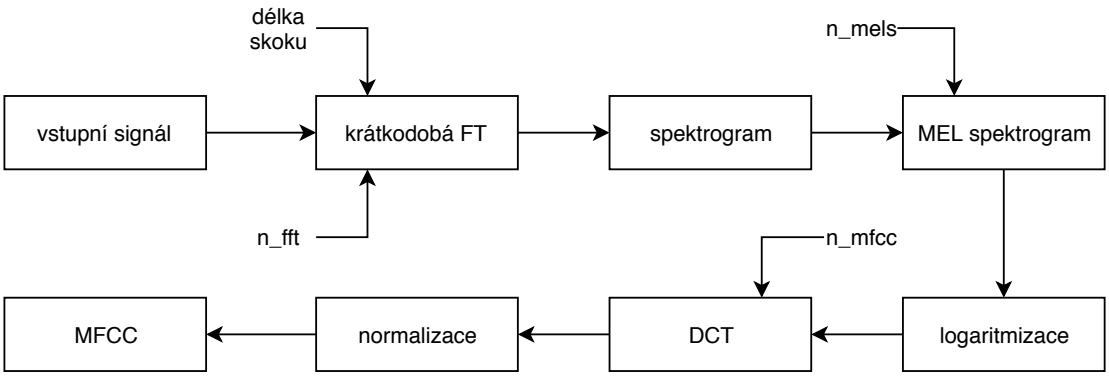
Obrázek 5: Melovská banka filtrů [16].

pásma v nelineární melovské stupnici a jednotlivá pásma se o polovinu překrývají. V lineární frekvenční stupnici tedy dostáváme filtry s rostoucí šířkou pásma pro vyšší kmitočty. Ve většině případů zpracování řeči se zanechávají pouze koeficienty, které jsou v řečovém pásmu, tedy od 300 Hz do 3500 Hz, v případě zpracování přednášek je však nutné uvažovat celé pásmo pro případ, že by záznam v danou chvíli neobsahoval řeč, ale například zvuk videa přehrávaného v prezentaci. [12][19].

Pro práci bylo využito Python knihovny Librosa[5], která právě vypočtení MFCC umožňuje. Schéma procesu je znázorněno na schématu 6. Na základě testování bylo použito následující nastavení vstupních parametrů:

- délka fft okna $n\_fft = 512$ vzorků
- *délka skoku* $= 256$ vzorků
- počet Mel pásem $n\_mel = 128$
- počet MFCC $n\_mffc = 20$

Tzv. „mel-frekvence“ je definována převodními vztahy 1 a 2.



Obrázek 6: Blokové schéma výpočtu MFCC.

$$f_{mel} = \text{Mel}(f) = 2595 log_{10}(1 + \frac{f}{700}) \tag{1}$$

$$f = \text{InvMel}(f_{mel}) = 700 \cdot (10^{\frac{f_{mel}}{2595}} - 1) \tag{2}$$

Šířka pásma je konstantní, tedy $d = \frac{24000}{129}$ a jeho středová frekvence popsaná vztahem $f_{mel,j} = j \cdot d$, kde j nabývá hodnot 1, ..., 128.

Pro další výpočty je pak výchozí hodnotou logaritmus energie v každém pásmu dán vztahem 3, kde $H_{mel,j}[k]$ reprezentuje diskrétní frekvenční charakteristiku pásma.

$$g_j = \ln \sum_{k=0}^{\frac{N}{2}} |S[k]|^2 H_{mel,j}[k] \tag{3}$$

Výstupem této mel-kepstrální analýzy je 20 reálných koeficientů na každý časový úsek zvuku získané diskrétní kosinovou transformací pomocí vztahu 4.

$$c_i = \sqrt{\frac{2}{P}} \sum_{j=1}^{P} g_j \cos\left(\frac{\pi i}{P}(j - 0.5)\right) \tag{4}$$

4.3 Průběh MFCC v čase

Možností využití získaných Mel-kepstrálních koeficientů je široká škála. Pro potřeby vzájemné korelace je však nutné množinu 20 koeficientů na vzorek vyjádřit pouze jed-

nou hodnotou. Velmi časté řešení pro rozpoznání hlasu je konstrukce takzvané euklidovské míry neboli kepstrální vzdálenosti vyjadřující rozdíl koeficientů oproti průměru reprezentujícího pozadí. Kepstrální vzdálenost je tak kritériem, které kvantifikuje rozdíly ve spektrech srovnávaných signálů. Díky tomu lze porovnávat i energeticky slabé a neznělé řečové úseky v silném frekvenčně odlišném pozadí. [19]

V případě úlohy hledání korelace dlouhých řečových promluv v tomto projektu bylo však zvoleno metody jiné. Cílem totiž není rozpoznání zda je řečník tentýž, ale klasifikace vývoje jeho řeči v čase. Pro toto řešení je nutné na koeficienty pohlížet jako na souřadnicový systém, kde každý koeficient definuje polohu v jedné jeho rovině. Každý jeden vzorek signálu popsaný 20 MFCC koeficienty tedy představuje jednu unikátní pozici ve 20 dimenzionálním prostoru. Výpočtem euklidovské vzdálenosti, dané vztahem 5, mezi pozicemi lze tak vypočíst vývoj těchto koeficientů v čase. V praxi to znamená, že takový signál pak nepopisuje řeč samotnou, ale pouze její vývoj.

$$|AB| = \sqrt{\sum_{i=1}^{20}(x_i - y_i)^2} \tag{5}$$

Tyto získané signály jsou uloženy jako pole hodnot pomocí knihovny *numpy* do pracovní složky projektu na disk počítače a cesta k nim opět do centrální databáze.

4.4 Vzájemná korelace dat

V první fázi tohoto kroku je vytvořena tabulka možných kombinací všech dvojic souborů, které jsou potenciálně vzájemně korelovatelné. U obrazových dat lze vycházet z faktu, že jedno zařízení nemůže pořizovat paralelní záznam, tedy žádný časový průnik se mezi daty nacházet nemůže. Jediný takový případ může nastat chybou lidského faktoru při zálohování duplicitních dat, ale i ten je ošetřen a na chod systému nemá negativní vliv. Oproti tomu zvuková data jsou všechna uložena jako jeden zdroj, tedy výskyt paralelního záznamu je velmi pravděpodobný. Z toho důvodu je nutné data porovnávat jak mezi sebou, tak každý soubor se zbytkem zdrojových zařízení.

Pro všechny možné kombinace je dále vypočtena vzájemná korelace pro nalezení dvojic, které se v čase překrývají. Korelace je prováděna pomocí Python knihovny *Scipy*

a je pro signály x a y definována vztahem číslo 6, pro $k = 0, 1, ..., ||x|| + ||y|| - 2$, kde $||x||$ je délkou signálu x, $N = max(||x||, ||y||)$.[14]

$$z_{[k]} = (x * y)(k - N + 1) = \sum_{l=0}^{||x||-1} x_l y^*_{l-k+N-1} \tag{6}$$

Při korelaci vzájemně korelovatelných signálů vzniká ve výstupním signále ostrý vrchol v místě, kde je korelace nejvyšší. Pro detekci tohoto vrcholu je vytvořeno vyhledávací okno o šířce 40 vzorků. Okno je posouváno po krocích dlouhých 20 vzorků. Proces vyhledávání lokálních maxim lze funkcí 3.

```
for x in range(0, len(corr) - win_size, step):
    local_peak_idx = np.argmax(corr[x:x + win_size]) + x
    base_value = (corr[x] + corr[x + win_size]) / 2
    diff_array[local_peak_idx] = abs(corr[local_peak_idx] - base_value)
```

Výpis 3: Ukázka cyklického procesu vyhledávacího okna.

Ta říká, že pro celý signál *corr* jsou po krocích velikosti *step* je hledáno lokální maximum v celé šíři váhovaného okna. Dále je zjištěna hodnota základny *base* jako průměr velikosti počátečního a koncového vzorku okna a rozdíl velikosti lokálního maxima vůči hodnotě základny je uložen do paměťového pole *diff_array* pod indexem jeho pozice.

Zjištěná maxima uložena v poli *diff_array* jsou dále porovnávána mezi sebou. Pozice nejvyššího je brána jako hledaná potenciální hodnota korelace. Zda je nalezená korelace validní je ověřováno dvojím způsobem. Za prvé je sledováno zda je nejvyšší maximum větší než daná limitní hodnota. V rámci projektu bylo testováním zjištěno, že takovouto mezní ideální hodnotou je číslo 300000. Pokud je nalezené maximum nižší, jedná se o korelaci s malým odstupem signálu od šumu a může tak docházet k neplatným nálezům, anglicky „False Positives“. Druhým sledovaným faktorem je porovnání hlavního maxima s dalším nalezeným, které se vyskytuje ve vzdálenosti minimálně 3 vzorků od pozice hlavního maxima. Poměr velikosti hlavního maxima a tohoto nalezeného je pak brán jako úroveň věrohodnosti výpočtu. Nejnižší povolené skóre, poměr hlavního ku vedlejšímu, je 1,5. Pokud není jedna z podmínek splněna, dvojice signálu je označena jako nekorelovatelná.

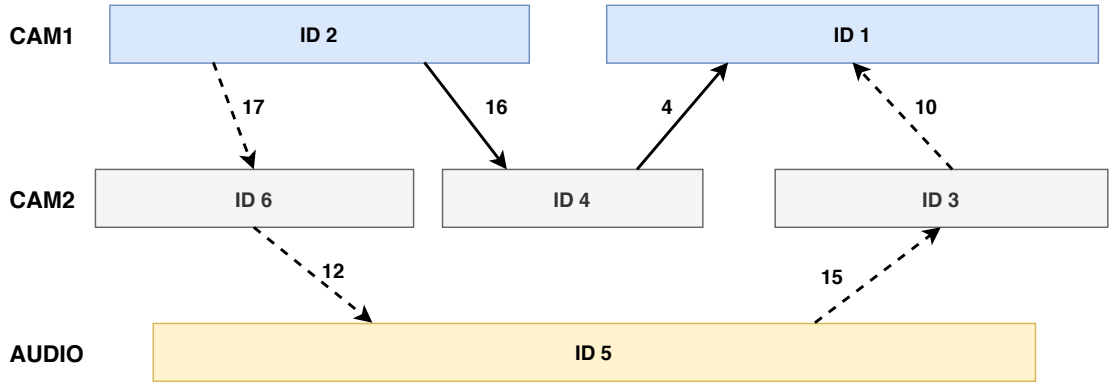
4.5 Eliminace redundantních vztahů

Veškeré tyto nalezené korelace jsou uloženy jako vztah dvou identifikátorů médií, jejich vzájemného offsetu a skóre věrohodnosti. Uložené vztahy je však možné dále zjednodušit. Jak již bylo v práci zmíněno, velká většina obrazových záznamů je natáčena ve formátu AVCHD, tedy se jedná o více souborový segmentovaný záznam. V případě detekce takového obrazového streamu pomocí metody popsané v 3.4, je tak znám vztah, tedy vzájemný offset, mezi segmenty samotnými. Pro všechny takové případy jsou tedy veškeré nalezené vztahy z předchozího bodu převedeny na první segment streamu. Tyto vztahy jsou mezi sebou porovnány a v případě nálezu více vztahů s navzájem identickým médiem jiného zdroje, například druhé kamery, je ponechán pouze vztah s nejvyšším věrohodnostním skórem. Tím je dosaženo velmi vysoké přesnosti celého systému. Pro zápis výstupních vztahů je pravidlem, že identifikátor prvního média musí mít nižší koeficient než identifikátor druhého. Podle toho je také polarizován offset - s kladným nebo záporným znaménkem. Díky tomuto řazení je při následujících krocích eliminována možnost zacyklení programu při výpočtu rekurzivních operací.

Získaná seřazená data jsou dále dělena do skupin vztahů, nových polí, které mezi sebou nemají žádný časový průnik. To je realizováno pomocí jednoduchého postupu. Cyklická funkce vezme každý vztah a zkontroluje, zda již existuje takové pole, které by obsahovalo jeden z identifikátorů definující daný vztah. Pokud takové pole existuje, je o tento vztah rozšířeno. Pokud ne, je tímto vztahem vytvořeno pole nové. Tím se veškerá data dělí na jednotlivé nahrávací bloky, ve velké většině samotné přednášky nemající žádné médium, které by je vzájemně spojilo. V praxi je to stav, kdy jsou o přestávce mezi přednáškami zastavené záznamy veškerých nahrávacích zařízení.

Dalším důležitým krokem v každé skupině je odstranění všech možných případů vícecestných vztahů mezi dvěma soubory. Tento problém je popsán na obrázku číslo 7. Jsou zde uvedeny tři zdroje - dvě kamery a jeden zvukový záznam. Sledovou kolizí je vztah mezi médiem ID 2 a médiem ID 1. Tyto dvě média mohou být spojeny dvěma cestami jak je znázorněno pomocí šipek. První trasa, čárkovaná, je delší a spojuje média přes 3 další. Druhá cesta je kratší, v její cestě je médium jen jedno společné. U každého vztahu je uvedeno skóre jeho věrohodnosti. Cílem vyhledávacího modulu není najít nejkratší spoj mezi dvěma soubory, ale právě ten nejspolehlivější. Z obrázku je tedy patrné, že hleda-

nou cestou je první zmíněná. Důvodem je velmi nejistý nález vztahu mezi ID 4 a ID 1 s věrohodnostním skórem nabývajícího hodnoty 4. Tento fakt může být dán například tím, že se na počátku média ID 1 mění řečníci, tedy jeho počátek neobsahuje řeč a korelovatelnost s ID 4 je v tuto chvíli velmi slabá, avšak stále postačující pro případ, že by vztah nebylo možné jiným způsobem ověřit.

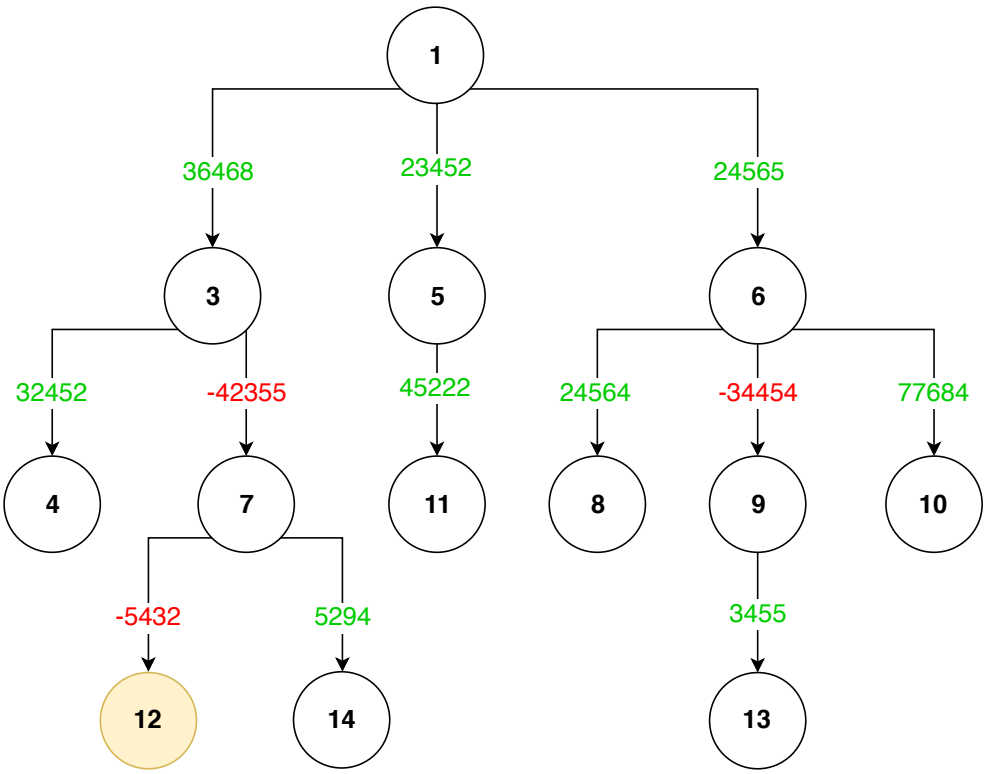


Obrázek 7: Demonstrace vícecestného vztahu mezi soubory.

4.6 Výpočet pozic klipů

Pro usnadnění dalších výpočtů jsou veškerá data převedena do datové struktury typu strom. Každé médium je reprezentováno jedním uzlem, které je tvořeno objektem obsahujícím identifikátor daného média, offsetem a věrohodnostním skórem k jeho rodiči a seznamem dalších uzlů, které jsou na něj dále napojeny. Ukázka příkladu stromu jedné skupiny je na obrázku číslo 8. Jak bylo určeno v předchozích bodech, data jsou ukládána tak, aby byla zachována posloupnost, že identifikátor prvního média v pořadí je vždy nižší než identifikátor následujícího. U tohoto příkladu je nejnižším nalezeným identifikátorem a tedy hlavním identifikátorem skupiny index 1, který je tedy kořenovým uzlem celého stromu. Offset vůči jeho dětem, tedy médiím v další úrovni, je vyjádřen ve počtech snímků videa nebo na tuto hodnotu přepočtená délka audia.

Cílem celého procesu synchronizace je získání pozice počátku každého média, podle kterých je pak možné veškerá data seřadit do jedné časové osy. Nejprve je však nutné najít počáteční médium celé skupiny. Opět s odkazem na ukázku v obrázku číslo 8, kořenové médium stromu id 1 je dočasně bráno jako počátek. Vůči němu jsou pak pomocí rekurzivní



Obrázek 8: Demonstrace rozložení a offsetů médií uložené v datové struktuře typu strom.

operace počítány, respektive sčítány, offsety jednotlivých médií. Například offset média s id 10 vůči id 1 bude 24565 + 77684, tedy 102249 snímků, tj. při snímkovací frekvenci 25 snímků za sekundu je médium přibližně o 1 hodinu a 10 minut zpožděno oproti médiu id 1. Opačným příkladem je ale offset pro id 13, které po stejném přepočtu vychází se záporným znaménkem, konkretně -6434 snímků, z čehož vyplývá, že médium s id 13 bude kořenové médium o necelých 5 minut předbíhat.

Veškerá tato data jsou ukládána do objektu typu slovník, kde se klíčem každého záznamu stává identifikátor média a jeho hodnotou offset vůči nule, tedy vůči kořenovému médiu. Po naplnění celého pole jsou hodnoty prohledány a vybráno médium s nejnižším offsetem, v případě příkladu se jedná o offset identifikátoru 12 s hodnotou -11319. Nalezené médium je tak absolutním počátkem celé skupiny. Offset tohoto média vůči kořenovému je pak přičten ke každému offsetu ve skupině a médium 12 se tak stává počátkem celé skupiny v čase.

4.7 Uložení dat

V posledním kroku je nutné data uložit do centrální databáze. Formát každého záznamu odpovídá požadavkům Final Cut Pro XML projektu popsanému v podkapitole 7.1. Příklad záznamů v databázi a jeho struktury je uveden na obrázku číslo 9.

	c_id	m_id	s_id	track	name	duration	start	end	in	out	timebase	ntsc	fps
1	0	0	1	audio	audio/0/ZO...	161017	0	161017	0	161017			
2	1	1	1	cam1	cam1/1/000...	36299	18982	55281	0	36299	25	FALSE	25.0
3	2	2	1	cam1	cam1/2/000...	36252	55280	91532	0	36252	25	FALSE	25.0
4	3	3	1	cam1	cam1/3/000...	36372	91532	127904	0	36372	25	FALSE	25.0
5	4	4	1	cam1	cam1/4/000...	30913	127904	158817	0	30913	25	FALSE	25.0
6	5	9	1	wideshot	wideshot/9/...	36791	21877	58668	0	36791	25	FALSE	25.0
7	6	10	1	wideshot	wideshot/10...	36876	58668	95544	0	36876	25	FALSE	25.0
8	7	11	1	wideshot	wideshot/11...	36660	95544	132204	0	36660	25	FALSE	25.0
9	8	12	1	wideshot	wideshot/12...	26677	132204	158881	0	26677	25	FALSE	25.0
10	9	5	5	cam1	cam1/5/000...	36887	0	36887	0	36887	25	FALSE	25.0
11	10	6	5	cam1	cam1/6/000...	35976	36887	72863	0	35976	25	FALSE	25.0
12	11	7	5	cam1	cam1/7/000...	34321	72863	107184	0	34321	25	FALSE	25.0
13	12	13	5	wideshot	wideshot/13...	36935	58	36992	0	36934	25	FALSE	25.0
14	13	14	5	wideshot	wideshot/14...	37140	36992	74132	0	37140	25	FALSE	25.0
15	14	15	5	wideshot	wideshot/15...	33109	74132	107241	0	33109	25	FALSE	25.0
16	15	8	8	wideshot	wideshot/8/...	2508	0	2508	0	2508	25	FALSE	25.0

Obrázek 9: Příklad tabulky pro uložení záznamů o klipech.

Každý záznam má své identifikační číslo (*c_id*), číslo média na které je klip vázán (*m_id*), číslo skupiny, do které klip patří, název tracku, vrstvy, ve které bude klip zobrazen a jméno klipu. Další parametry vyjádřené v počtu snímků definují jeho dobu trvání (*duration*), pozici ze které bude klip začínat (*start*), pozici ve které skončí (*end*) a parametry definující od kterého snímku do kterého (*in* a *out*) bude video přehráváno. Poslední parametry popisují typ jak bude video vykreslováno. Tedy jeho časovou základnu (*timebase*), definice zda se jedná o kódování typu „NTSC“ či „PAL“ a snímkovací frekvence samotného videa (*fps*).

5 Kritéria kvality

Obecným cílem při záznamu přednášek je hrubé zaznamenání řečníka prezentujícího svou přednášku pro vzdělávací, marketingové, či jiné účely. Zde je jediným pravidlem, aby byl řečník dobře slyšet a aby byla v obraze dobře vidět jeho prezentace. Se stále se zlepšujícími technologickými možnostmi se však objevuje také obecná snaha zvyšovat kvalitu videozáznamu samotného. Takové zlepšení neznamená jen použití kamer s kvalitnějším obrazovým senzorem pro vyšší rozlišení a věrnější barevné podání, ale také vnesení určitých estetických pravidel, která se velmi podobají pravidlům používaných v kinematografii. S pomocí těchto pravidel pak lze zaručit co možná nejpřirozenější vjem diváka s omezením veškerých rušivých elementů, které by mohly jeho pozornost odvádět. Díky tomu pak divák může i takový záznam naplno prožít stejně jako se to děje například při sledování kvalitně natočeného filmu.

Kompletní popis obrazové scény, plné pokrytí veškerých případů pravidel a případných chyb, které mohou nastat a popis jejich řešení je velmi obsáhlé téma. Ač se to nezdá, i takovouto práci totiž lze označit za jistý druh umění. Tato pravidla totiž nevycházejí jen z‘nějakých předepsaných norem. Zkušený kameraman nebo střihač je dokáží aplikovat na základě svých pocitů, zkušeností a do své práce jistým způsobem propsat.

Hlavním zdrojem pro tuto kapitolu je vybrána kniha Josefa Valušiaka Základy střihové skladby [20], kde jsou základní kinematografická pravidla popsána. Snahou kapitoly je pak za pomoci dalších pramenů aplikace těchto pravidel na natáčení a střih přednášek.

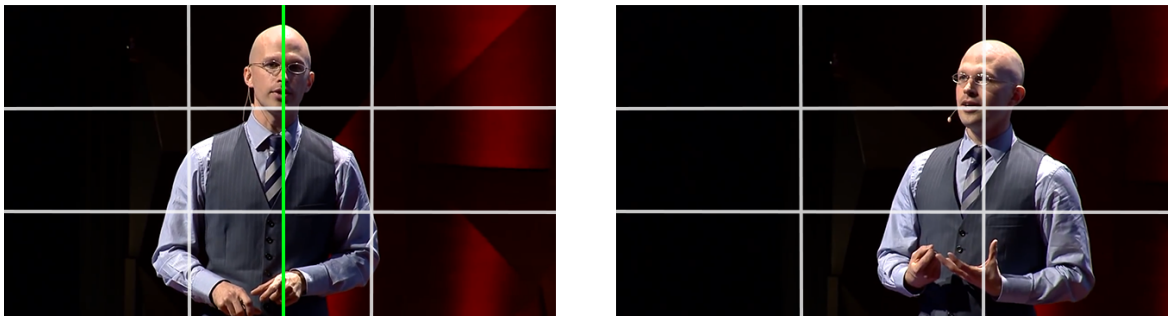
5.1 Pravidla pro natáčení

K výsledné podobě stříhaného záznamu přispívá velkou mírou samotný kameraman, který data pro střihače připravuje. Jeho úkolem je kontinuální snímání řečníka s dodržením předem domluvené obrazové kompozice a typu záběru za pomoci především svislého a horizontálního pohybu kamerou, hovorově zvaného jako „švenkování“ a úpravou transfokátoru kamery, tedy zoomu. Veškeré tyto pohyby musí být jemné a maximálně přirozené. Hlavní kameraman nebo jeho asistent se dále stará o rozestavení kamer, rozdělení typů jejich záběrů, nastavení světel na pódiu a kontrolu shodného vyvážení bílé na všech kamerách.

5.1.1 Kompozice

Typy záběru pro nahrávání přednášek jsou v podstatě analogické k rozdělení podle Valušiaka [20]:

- **Velký celek**, anglicky wideshot - přímý pohled na celé pódium včetně projekce, vždy jako statický záběr určený především pro zálohu, případně jako informativní záběr popisující divákovi dané prostředí
- **Celek** - použit jako záloha u menších akcí, v případě diskusí zabírá všechny diskutující, záběr stále jako statický
- **Polocelek** - postava je ukázána celá, stává se primárním objektem v záběru
- **Americký plán** - záběr přibližně po kolena, běžné pro prostřih s polodetailem, je používána v případě většího pohybu řečníka a širokých gestikulacích
- **Polodetail** - od poprsí postavy, nejčastěji používaný záběr
- **Detail** a **Velký detail** - pro záznam přednášek se příliš nepoužívají, maximálně při požadavku zachycení emocí publika



(a) (b)

Obrázek 10: Porovnání symetrické (a) a nesymetrické (b) kompozice, databáze TEDx.

Kompozici obsahu lze dělit na symetrickou a nesymetrickou, jak je znázorněno na obrázku číslo 10 použitého z databáze konferencí TEDx[3] formátu TED[4]. První zmíněná

[3]https://www.youtube.com/watch?v=5MgBikgcWnY
[4]https://www.ted.com/

kompozice je známa spíše pod pojmem středová, z čehož vychází, že vizuální těžiště řečníka, tedy hlavní zájmová oblast, leží v tomto případě na horizontální ose přímo ve středu obrazu. Pokud se řečník dívá přímo do kamery, lze tuto metodu použít, avšak pro většinu případů je pro diváka tato symetrie velmi nepříjemná.

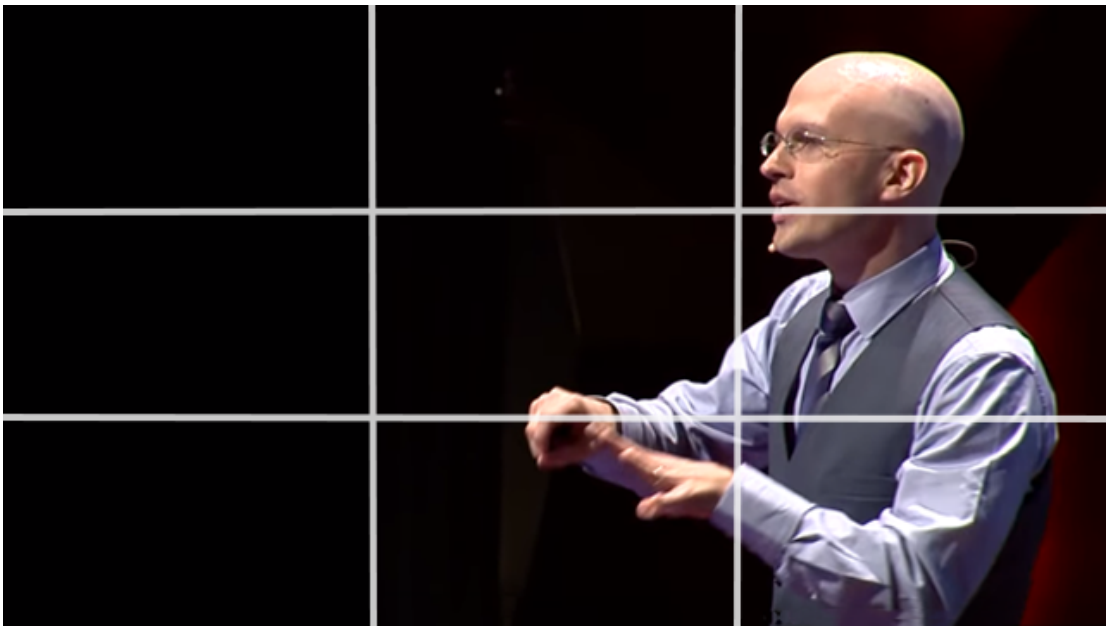
Nejčastěji používanou kompozicí je tedy kompozice nesymetrická, kterou velmi stručně popisuje také například článek D. Hulense a T. Rumese o autonomním snímání řečníka pomocí PTZ kamery [13]. Jako hlavní dvě pravidla uvádí takzvané pravidlo třetin a dodržení dostatečného prostoru nad hlavou. Vizuální těžiště řečníka je tak na horizontální ose mírně vychýleno ze středu obrazu na opačnou stranu než je řečník natočen. Tedy pro případ, kdy se dívá směrem doleva, těžiště jeho těla musí být v pravé polovině obrazu a naopak. Míra vychýlení je přirozeně závislá na úhlu mezi spojnicí řečníka a kamery vůči směru, kterým je natočena jeho souřadnicová soustava. V případě, že kamera snímá pouze detailní záběr na řečníkův obličej, určení výsledného úhlu je dáno natočením hlavy, jejíž popsání v prostoru je dnes relativně snadno řešitelný problém. Základní detekci lze provést například detekcí zájmových bodů obličeje, anglicky „landmarks“, pomocí otevřené knihovny Dlib[15]. Pro spolehlivější a robustnější řešení je však nutné využití neuronových sítí. Takovému případu se věnuje například otevřený projekt OpenFace [5], který rozpoznává kompletní popis obličeje pomocí konvolučních neuronových sítí.

Vzhledem k tomu, že na přednášky jsou snímány především záběrem typu polodetail, je řešení podstatně složitější. Vizuálním těžištěm se totiž oproti hlavě stává horní polovina těla, v tomto případě hruď. Mimo její sledování je však také nutné zaměřit se na gestikulace, tedy ruce, které se tak stávají nedílnou součástí jeho projevu. Detekci takového systému se věnuje několik projektů, některé za pomoci přídavného senzoru pro zjištění hloubkové mapy, jiné pouhým zpracováním obrazu. Tomuto problému se dále věnuje kapitola 6.

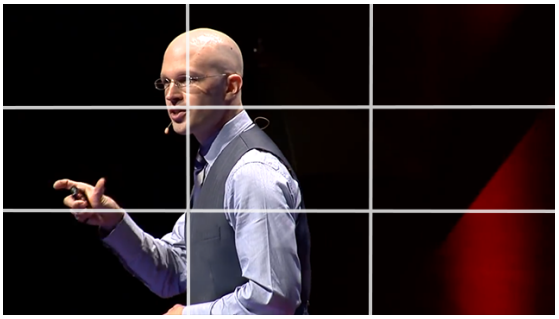
Poměr vychýlení zjištěného těžiště od horizontálního středu obrazu je přibližně lineárně závislý vůči úhlu natočení sledované postavy. Postava však nesmí žádnou částí těla překročit okraj obrazu. Viz porovnání číslo 12. Hraniční případ takové kompozice je znázorněn na obrázcích číslo 11. Pro vertikální zarovnání platí pravidlo místa nad hlavou, anglicky „head room“, taktéž popsaného v článku [13]. I zde je využíváno pravidla třetin. Hlava řečníka by se měla nacházet na přelomu horní třetiny obrazu. Pravidlo místa nad hlavou pak tuto

[5] https://cmusatyalab.github.io/openface/

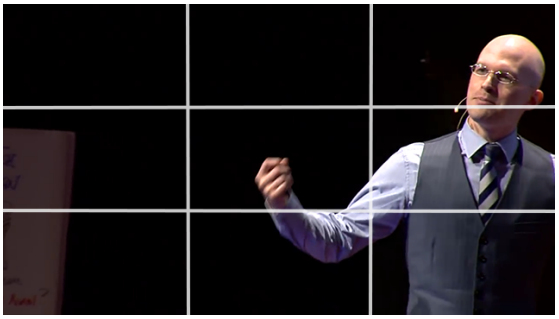
podmínku rozšiřuje o fakt, že místo nad hlavou řečníka musí být co možná nejmenší, ale v případě záběru polodetailu hlava nesmí přesahovat horní okraj obrazu. Případy porušení tohoto pravidla jsou znázorněny na obrázku číslo 13.



Obrázek 11: Hraniční případ správné nesymetrické kompozice.



(a) postava vychýlená na špatnou stranu.

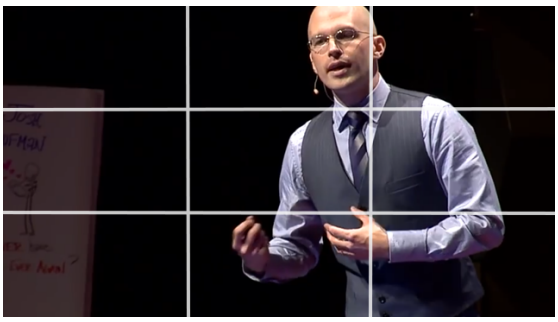


(b) příliš vychýlené, gestikulace mimo záběr.

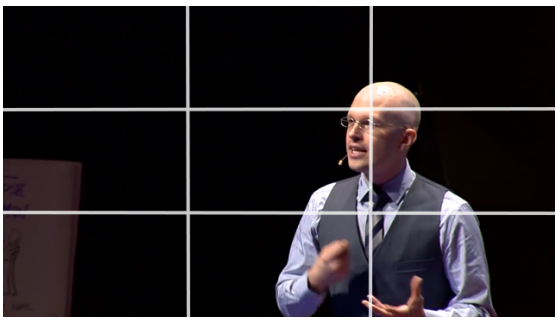
Obrázek 12: Demonstrace chyb v horizontální rovině. Použito z databáze TEDx.

5.1.2 Nastavení kamer

Úlohou hlavního kameramana, či jeho asistenta je dále sjednocení veškerého barevného nastavení kamer. Kamery musí mít stejně vyváženou bílou, musí mít správně



(a) hlava přesahuje horní okraj obrazu.



(b) příliš mnoho místa nad hlavou

Obrázek 13: Demonstrace chybného místa nad hlavou. Použito z databáze TEDx.

a shodně nastavený poměr clony a úrovně citlivosti senzoru pro shodný jas. Kamery dále musí být správně rozmístěny tak, aby bylo možné splnit pravidlo dvojí změny zmíněné v podkapitole 5.2.

5.2 Pravidla pro střih

Základem střihu je odstranění veškerých rušivých elementů, které ve videích mohou nastat a zajištění, že je řečník v každém záběru dobře vidět. Hlavní chybou je tedy stav, kdy se řečník ocitá mimo záběr či na hranici okraje obrazu. V případě jedné hlavní kamery a jedné záložní statické pak stačí pouze na druhou zmíněnou přestřihnout. Pokud se však jedná o složitější více-kamerový systém, je nutné záběry jednotlivých kamer klasifikovat a vyjádřit určitým skórem kvality. Vzhledem k tomu, že se práce zabývá především dvou-kamerovým systémem, tento problém zde není více řešen.

Pro přirozený prostřih dvou záběrů je nutné dodržet jejich dostatečnou odlišnost[20], které by se dalo nazvat také jako „pravidlo dvojí změny“. Významem pravidla je, že dva po sobě jdoucí záběry snímající jeden a ten samý objekt musí dodržovat alespoň dvě odlišná kompoziční nastavení. Ve většině případů je to pak odlišný úhel pohledu a odlišný typ záběru. Například, pokud kamera snímající polodetail stojí na pravé straně sálu, druhá kamera se záběrem typu americký plán bude postavena na opačné straně, tedy pravé.

5.2.1 Pravidlo osy

Dalším pravidlem, které je nutno u střihu řešit je takzvané „pravidlo osy“, které zjednodušeně říká, že při prostřihu z jedné kamery na druhou musí obě kamery fyzicky stát na druhé straně pomyslené dělící osy, než sledovaný objekt či jejich skupina. To určuje orientaci diváka vůči objektu i prostoru, v němž se objekt nachází. Bližší popis tohoto pravidla včetně případů možností jeho porušení je popsán v práci[22].

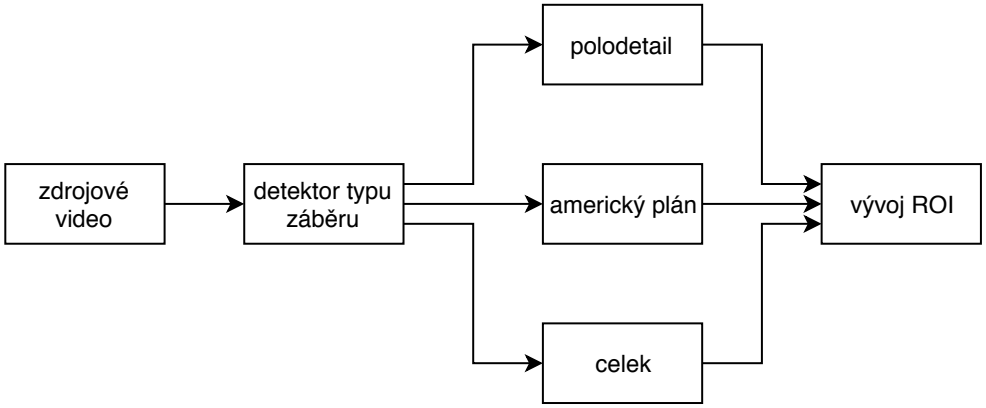
6 Detekce chyb

Téma hledání chyb v obraze je velmi komplexní. Popisu chyb, které lze v obraze hledat, se blíže věnuje kapitola 5, která jako hlavní chyby uvádí tři typy. Chybu kompozičního rozložení v obraze, chybu špatného ostření a chybu třesem kamery. Tato práce se věnuje především první zmíněné. Veškeré tyto chyby vyžadují na vstupu obraz s určením umístění hlavy řečníka. Kompoziční detektor sleduje její pozici, detektor ostření sleduje, zda je obličej správně zaostřen a detektor třesu obrazu může pomocí optického toku porovnávat pohyb hlavy vůči pohybu pozadí. Tím lze zajistit stabilitu detekce i v případě, že řečník stojí před dynamicky se měnící projekcí.

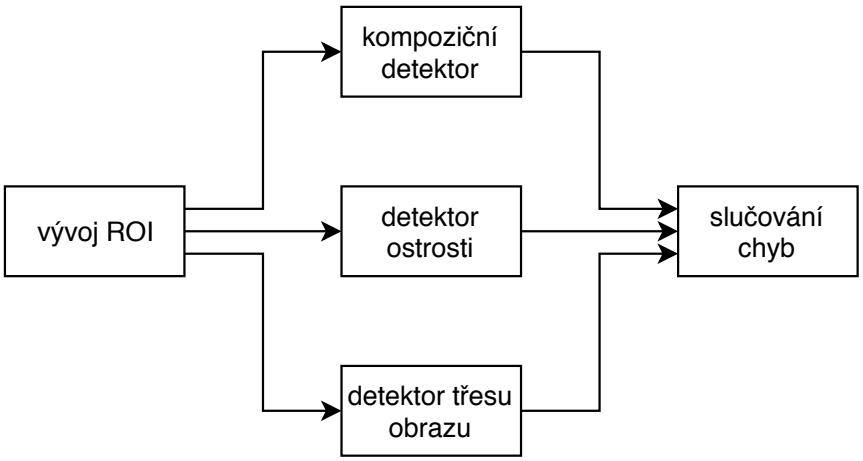
V rámci této kapitoly jsou popsány dvě hlavní části. První je pomocí metod pro detekci a trackování vytvoření přehledu o vývoji sledovaného objektu, dále používáno jako zkratka ROI, tedy z angličtiny „Region Of Interest“, tedy oblast zájmu, v čase. Blokové schéma návrhu rozložení modulů v této části je popsáno na obrázku číslo 14. Vstupem tohoto procesu je zdrojové video, tedy dané obrazové médium, které je dále zpracováno pomocí kompozičního detektoru. Úkolem tohoto detektoru je rozlišení typu záběrů. Tento krok je důležitý zejména v případě, kdy systém na vstupu neví o jakou kameru se jedná a tedy jaký typ záběru by měl očekávat. Na něm totiž záleží další zpracování. Detekce v záběru typu polodetail se budou zpracovávat odlišně oproti americkému plánu či statickému celku. V rámci práce je řešen první typ záběru, tedy detekce řečníka v obraze hlavní kamery, která natáčí vždy polodetail. Na výstupu procesu jsou data uložena jako tok ROI v čase pro další zpracování popsaného v podkapitole 6.2.

Druhou popisovanou částí je způsob samotné detekce chyb. Návrh celého tohoto pro-

cesu je popsán schématem na obrázku 15. Jeho vstupem je vývoj ROI popsaný v předešlé části úvodu. Pomocí tohoto toku jsou dále postupně zjištěny všechny zmíněné typy chyb, které jsou ve finále sloučené do jedné časové osy. Práce se věnuje prvnímu zmíněnému detektoru, tedy kompozičnímu pomocí sledování vývoje pohybu hlavy řečníka.



Obrázek 14: Schéma návrhu modulů pro získání vývoje ROI.



Obrázek 15: Schéma návrhu modulů pro detekci chyb z vývoje ROI.

6.1 Pohyb řečníka v čase

Cílem tohoto procesu je popsání vývoje pohybu řečníka a každé další osoby v čase pomocí detekce jejich obličejů a jejich následného trackování. Základní myšlenkou použitou pro tento proces je úvaha, že není vyloženě nutné znát identitu řečníka, jelikož každá osoba,

která se v obraze vyskytne musí splňovat hlavní kompoziční pravidla uvedená v podkapitole 5.1.1. Tedy hlavním úkolem modulu pro zjištění toku ROI u polodetailního záběru je určení pozic nalezených obličejů v obraze.

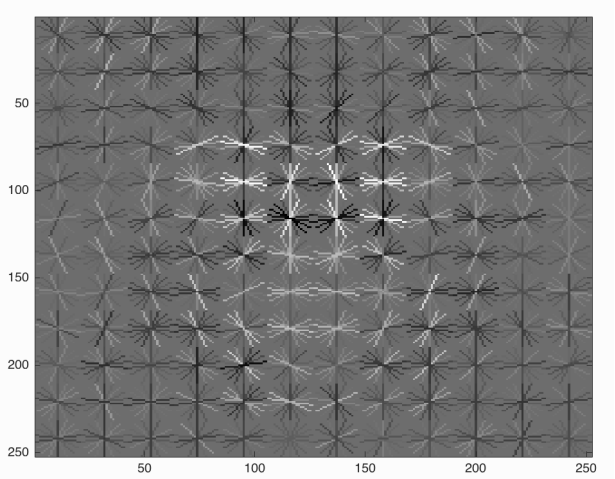
6.1.1 Detekce řečníka

Řídící proces pak probíhá způsobem znázorněným schématem na obrázku číslo 18. Všechna média z databáze jsou načtena po celcích, tedy kompletních video streamech, jejich získání popsáno v podkapitole 3.4. Pro ukládání dat procesu je pro každý video stream zřízen objekt zvaný „Multitracker“. Multitracker obsahuje za prvé sběrné pole o délce rovné počtu snímků video streamu, tedy se v podstatě jedná o jakousi časovou osu. Podle té je dále nazýváno z angličtiny jako „timeline“ a jeho buňky odpovídajícím pořadí snímku „timecode“. Mimo toto pole jsou součástí multitrackeru také, jak z jeho názvu vyplývá, objekty samostatných trackerů. Každý tracker, také strukturovaným objektem, obsahuje mimo své unikátní identifikační číslo vlastní časovou osu, do které je postupně vývoj sledovaného objektu tímto trackerem ukládán, zároveň tracker obsahuje veškeré funkce pro reinicializaci či aktualizaci stavu ROI při zavedení nového snímku.

Pro zjednodušení celkového procesu je načítán pouze každý druhý snímek. Tyto snímky jsou následně přeškálovány na rozměr 640x360 pixelů (velikost určena na základě testování pro účely získání toku ROI jako plnohodnotně dostačující) a nahrávány do zásobníku o velikosti 1000 snímků, tedy do paměti RAM. Velikost takového zásobníku je přibližně 220 MB. Dále následuje detekce v klíčových snímcích. Pravidla jsou nastavena tak, že je detekce zkoušena každé 2 sekundy, tedy u videa s 25 fps je to každý 50. snímek, a v případě, že není detekován žádný obličej, vyhledávací krok se sníží a detekce probíhá v každém 5. snímku, dokud není řečník nalezen. Tato data jsou uložena do timeline Multitrackeru pod příslušný timecode.

Pro takovou úlohu bylo využito algoritmu pro detekci obličeje na základě kaskády Histogramů Orientovaných Gradientů, zkráceně HoG. Algoritmus pro extrakci HoG počítá výskyt orientací hran v blízkém lokálním okolí obrazu. Tento obraz je tak rozdělen do určitého počtu malých regionů, zvaných buňky, pro které je následně vypočítán histogram. [8] Díky takovému vyjádření obrazu je detekce velmi odolná vůči jasovým změnám v obraze.

Vyjádření obrazu obsahujícího obličej pomocí HoG algoritmu pak demonstruje obrázek číslo 16. Takto převedený obraz je dále porovnáván s modelem obličeje a při dostatečné shodě je detekce označena jako validní. Jejím výstupem pak jsou informace o souřadnicích a velikosti nalezeného obličeje a věrohodnostní skóre, se kterým byl obličej nalezen. V projektu bylo využito HoG detektoru knihovny Dlib[15].



Obrázek 16: Demonstrace obrazu obličeje vyjádřeného pomocí algoritmu HOG.[1]

6.1.2 Proces trackování

Jádrem procesu trackování je algoritmus zvaný Kernelized Correlation Filters, zkratkou KCF. Základní ideou trackování s použitím korelačních filtrů je odhad takového obrazového filtru, který filtrací se vstupním obrazem produkuje vhodnou odezvu, typicky Gaussovského tvaru centrovaného do středu hledaného regionu. Pro následující snímek je pomocí stejného filtru zjištěna odezva, jejíž maximum, určuje novou pozici cíle. [10] Pro implementaci filtru v projektu bylo využito knihovny OpenCV [4].

S jeho pomocí je prováděn další krok procesu, tedy dopředné trackování. Každý tracker odpovídá jednomu sledovanému objektu od počátku jeho nálezu ve video streamu po jeho ztrátu, v té době je také veden jako aktivní. Odpovídá tedy jedné sledované osobě. Pro každý snímek jsou nejprve všechny aktivní trackery validovány. Validace je nastavena tak, že tracker může samovolně trackovat obličej až 10 snímků bez ověření detekcí. Oveření probíhá detekcí obličeje uvnitř samotného ROI trackeru. Přesněji, je vyříznuta

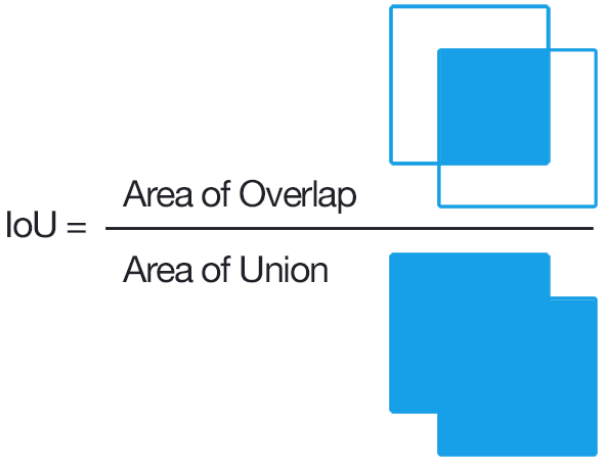
část snímku s rozměrem o něco větším než je aktuální ROI. V této části je pak provedena obvyklá detekce obličeje. Pokud je obličej nalezen, trakcer je na dalších 10 snímků označen jako validovaný. Vzhledem k tomu, že detekce probíhá pouze ve velmi malém obrazovém poli, jedná se oproti detekci v celém snímku o poměrně rychlou operaci. V případě, že není možné takto tracker validovat, je ukončen a označen jako neaktivní. Jeho získané hodnoty zůstávají uloženy, tracker však již v rámci dopředného trackování nebude aktualizován. V opačném případě je v daném timecode dále ověřováno, zda se nejedná o klíčový snímek a tedy neexistují záznamy o detekovaných obličejích. Pokud ne, všechny aktivní trackery se aktualizují novým snímkem, ROI data uloží do objektů trackerů a reference k nim do timeline Mulitrackeru a cyklus se opakuje pro další snímek, tedy další timecode. V případě, že se však o klíčový snímek jedná a timeline obsahuje informace o detekovaných obličejích, dochází k další fázi ověřování a to porovnání dat trackerů s těmito detekcemi.

6.1.3 Munkreuv algoritmus

Ověření je poměrně složitým problémem, který je řešen pomocí implementace Munkreova algoritmu, známého také jako Maďarský algoritmus. Algoritmus řeší problém, kdy je potřeba co nejlépe přidělit daný počet různě náročné práce dělníkům s různou sazbou. Blíže je popsán například v článku[11]. V tomto případě je použit pro přiřazení aktuálních ROI všech aktivních trackerů, tedy aktuálně sledovaných objektů, s ROI objektů, které byly detekovány pomocí detektoru. Zda je jedná o průnik dvou ROI je řešeno pomocí Jaccardova indexu, známého v angličtině také jako vztahu Intersection over Union. Vztah s názornou ukázkou je vyjádřen pomocí vztahu na obrázku číslo 17. Pokud je získaný poměr větší než 0.3, vztah je brán jako ověřený. Všechny ověřené trackery jsou reinicializovány podle detekovaného ROI, pro nově detekované objekty (další řečník) jsou inicalizovány nové příslušné trackery. V případě neověření vztahu je daný tracker pouze aktualizován. Ukončen může být až pouze v případě neplatné validace v dalších cyklech.

Pomocí dopředného trackování jsou tak založeny trackery všech osob, které se v sekvenci snímků zásobníku vyskytly a jejich vývoj v čase uložen do timeline. Celý zásobník je dále trackován zpětně, snímky se tedy ze zásobníku odečítají pozpátku. Proces probíhá podobně jako u dopředného trackování pouze s vynecháním Munkreova algoritmu. Trackery podléhají pouhé validaci.

Celá časová osa, timeline, obsahující veškerá data o vývoji ROI jednotlivých trackerů je pak uložena do pracovní složky a cesta k ní do centrální databáze.

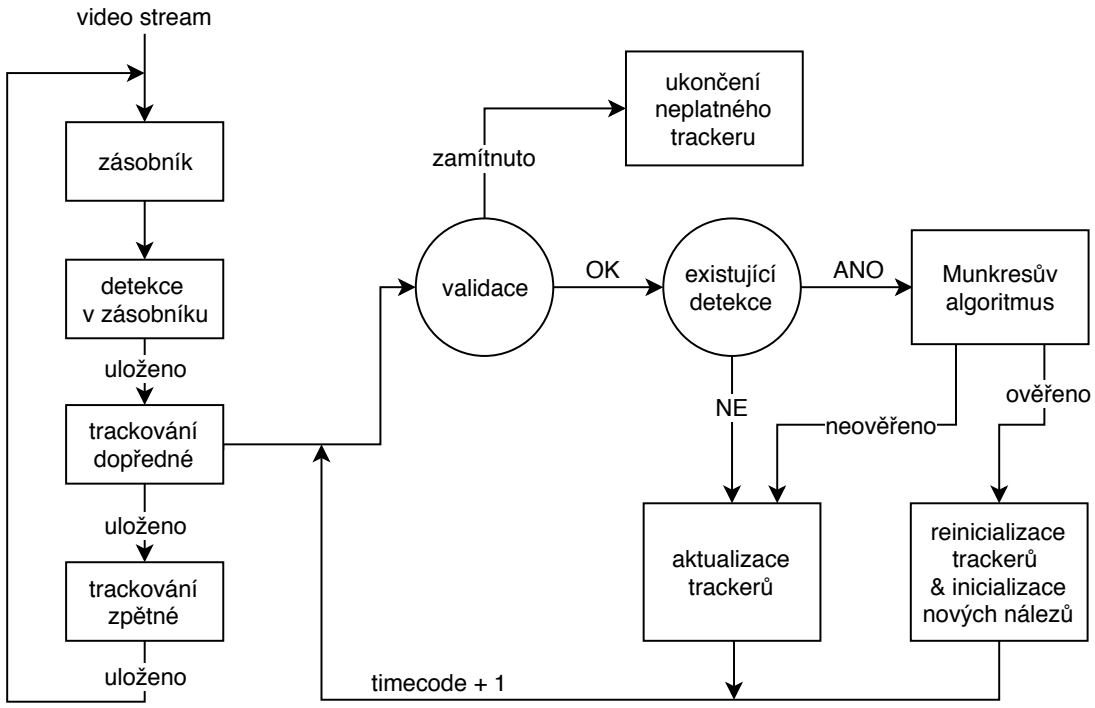


Obrázek 17: Určení Jaccardova indexu pro ověření shody detekce a trackování.[17]

6.2 Dektor kompozičních chyb

V rámci projektu je řešena otázka základního určení kompozičních chyb v obraze, které jsou definovány výskytem a polohou obličeje řečníka a dalších osob objevujících se ve video streamu. Z podkapitoly věnující se kompozičnímu rozložení v obraze 5.1.1 vycházejí pro tento úkol následující pravidla. Obličej se musí pohybovat v horních dvou třetinách obrazu, musí dodržet popsané pravidlo místa nad hlavou a nesmí překročit okaje obrazu. Pro detekci takových chyb byly definovány dva stavy. Jedním stavem je stav „LOST“, který nastává ve chvíli, kdy v obraze není nalezen žádný řečník. Druhým je stav „WARNING“, který nastává ve chvíli, kdy minimálně jedna osoba nesplňuje následující podmínky: horní okraj hlavy osoby nesmí překročit horní hranici obrazu, levý okraj hlavy se nesmí dostat do prostoru levé 1/16 obrazu, totéž platí zrcadlově pro pravý kraj. Šířka tohoto prostoru byla určena na základě testování. Pro spodní hranici platí pravidlo, kdy střed hlavy nesmí překročit úroveň 2/3 obrazu.

Hodnoty pole časové osy jsou postupně vyčítány, kontrolovány podle zmíněných pravidel a štítkovány v případě nálezu. Pro ukládání a výpis chyb je pak definováno poslední pravidlo, které říká, že chyby ležící od sebe blíže než je časová vzdálenost 5 sekund, jsou



Obrázek 18: Popis procesu získání ROI dat z jednoho video streamu.

seskupeny a uloženy jako chyba jedna. Počátek, konec a typ všech chyb je následně uložen do centrální databáze.

6.3 Možné zlepšení

V této kapitole se nabízí široké množství případného zlepšení. Mimo zmíněného detektoru ostrosti, detektoru třesu obrazu pomocí optického toku je to také zpracování zvuku, kde lze sledovat nejen například jeho výpadky, ale také pomocí jeho rozpoznávání určit zda hovoří stále jedna a ta samá osoba nebo se jedná o dotaz z publika, tedy i v případě přítomnosti řečníka v obraze by mělo být prostřihnuto na statickou kameru s širokým záběrem celého sálu. Dále je to samotná kompozice, kterou však lze celkově parametrizovat až ve chvíli, kdy je z obrazu správně detekována celá postava řečníka, konkrétně budoucí možné využití projektu [6].

7 Vytvoření finálního projektu

Poslední částí je samotné zpracování výstupních dat. Cílem práce je připravení projektu pro střihačský software, ve kterém jsou jak synchronizované veškeré vstupní zdroje, tak označena riziková místa výskytu chyb. Testování bylo prováděno na Adobe Premiere Pro, verze CC 2017, jednom z nejpoužívanějších střihačských softwarů. Do jeho vlastních projektových dat není možné nijak zasahovat. Pro samotné vytvoření projektu bylo využito formátu *Final Cut Pro XML Interchange* [2], vyvinutého firmou Apple. Tento formát je konstruován k jednoduchému přenosu projektových informací mezi jednotlivými střihačskými programy, tedy je podporován i zmíněným Adobe Premiere Pro.

Úkolem střihače je pak v projektu zkontrolovat nalezené chyby, podle svého uměleckého záměru podle těchto chyb záznam sestříhat, upravit začátek, konec a hotový sestřih vložit k záznamu samotné prezentace nahrané podle podkapitoly číslo 2.2.

7.1 Struktura Final Cut Pro XML formátu

Final Cut Pro XML Interchange formát využívá sadu konvencí založenou na konvencích formátu XML, podle kterého je dokument sestavován a dále překládán. Díky tomu je zajištěna poměrně velká stručnost a jednoduchost kódu. Ukázka jeho struktury je znázorněna pomocí tohoto zkráceného výpisu 4.

```
<xmeml version="1">
  <sequence>
    <name>Sequence 1</name>
    <duration>129934</duration>
    <rate>
      <timebase>25</timebase>
      <ntsc>FALSE</ntsc>
    </rate>
    <media>
      <video>
        <track>
```

```
                ... obrazovy zaznam prezentace
                <clipitem>
                </clipitem>
            </track>
            <track>
                ... obraz zalozni kamery
            </track>
            <track>
                ... obraz hlavni kamery
            </track>
        </video>
        <audio>
            <track>
                ... zvuk hlavni kamery
            </track>
            <track>
                ... zvuk zalozni kamery
            </track>
            <track>
                ... hlavni zaznam zvuku
            </track>
        </audio>
    </media>
  </sequence>
</xmeml>
```

Výpis 4: Ukázka struktury Final Cut Pro XML Interchange formátu.[2]

Struktura je tvořena sekvencí, která má své jméno, dobu trvání, snímkovací frekvenci a další informace o jejím nastavení. Obsahem sekvence jsou dále média dělená na video a audio sekci. Každá sekce dále obsahuje jakési vrstvy, „tracky“, ve kterých jsou jednotlivé zdrojové audiovizuální zdroje vloženy. Tyto zdroje jsou vkládány formou jednotlivých klipů. Jejich strukturu dále popisuje kód číslo 4.

```
<clipitem id="clipitem_1">
    <name></name>
    <duration>36299</duration>
    <start>18982</start>
    <end>55281</end>
    <in>0</in>
    <out>36299</out>
    <file  id="file-1">
        <pathurl>file:path/d1_track1/cam1/00025.MTS</pathurl>
        <rate>
            <timebase>25</timebase>
            <ntsc>FALSE</ntsc>
        </rate>
    </file>
    <marker>
        <in>1</in>
        <out>129</out>
        <name>lost</name>
    </marker>
    <marker>
        <in>1991</in>
        <out>2056</out>
        <name>warning</name>
    </marker>
</clipitem>
```

Výpis 5: Ukázka struktury položky clipitem.[2]

Každý klip má svůj předem přiřazený identifikační název, je zde definována jeho délka, místo, pomocí atributu *start* určen počátek v tracku, atributy *in* a *out* pak popisují od jakého snímku po který bude video vykresleno. V sekci *file* je dále popsána cesta k souboru *pathurl* a další informace jako jeho originální snímkovací frekvence a typ kódování.

V neposlední řadě každý klip obsahuje, v případě nálezu značky, anglicky „marker“, pro výpis nalezených chyb. V ukázce jsou popsány dva případy, jedním je chyba lost, která trvá od snímku 1 do snímku 129, druhá chyba „warning“ trvající od snímku 1991 do 2056.

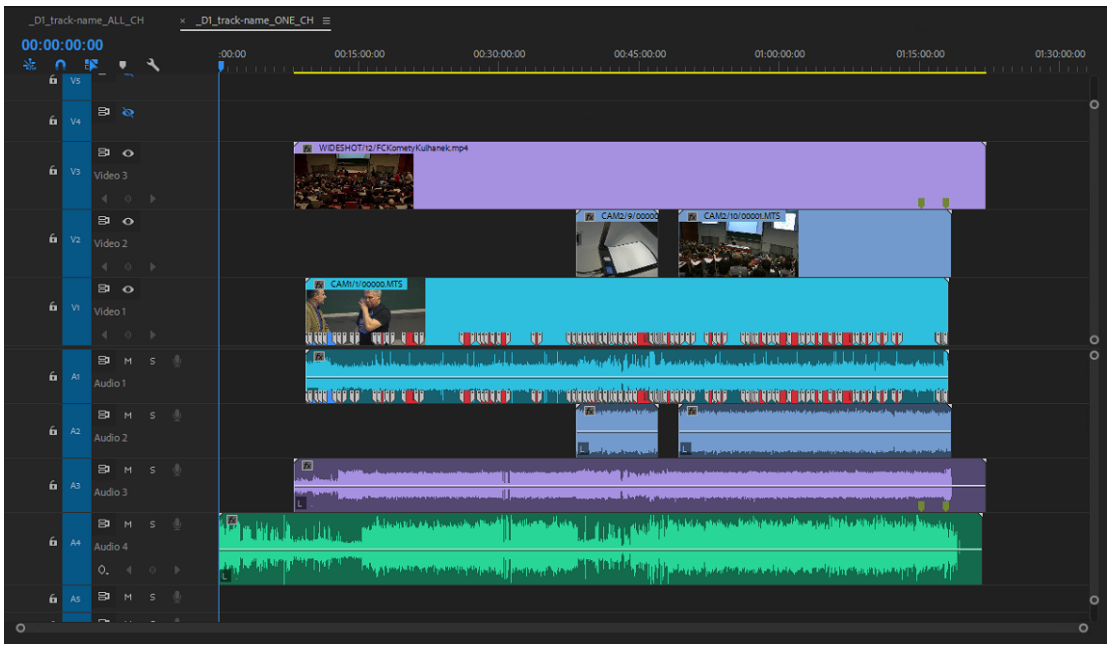
8 Výsledky testování

Obrázek číslo 19 demonstruje vygenerovaný projekt sestavený z dat ze tří kamer a jednoho zvukového nahrávadla. Doby výpočtu této přednášky jsou následující ve formátu hh:mm:ss:

- Zpracování vstupních dat a vytvoření databáze: 0:00:16
- FFmpeg konverze: 0:09:18
- MFCC extrakce: 0:01:54
- Synchronizace všech dat: 0:00:09
- Zjištění pohybu řečníka v čase (detekce a trackování): 0:22:23
- Proces hledání chyb: 0:00:03

Velikost vstupních dat je 58 GB, doba přednášky 1 hodina 20 minut, celkový čas výpočetního procesu 34 minut 3 sekundy. Proces lze dále zrychlit spuštěním až 5 procesů najednou, přičemž celková doba výpočtu se zvýší na přibližně 50 minut. To znamená že za jednu hodinu lze spočítat až 5 obdobně dlouhých přednášek.

Obrázkem číslo 20 je demonstrována přesnost výpočtu. Lze vidět 4 různé zvukové stopy, přiblížené na nejnižší rozlišení projektu, tedy 1/25 sekundy, tj. 40 ms na krok (vycházející ze snímkovací frekvence videa). Na obrázku je vidět, že zvuk první, světle modré barvy, je zhruba o pětinu kroku, tedy přibližně 8ms, posunut oproti zbylým zdrojům. Dáno je to především tím, že se jedná o méně kvalitní zvuk interního mikrofonu kamery, která ho nahrála. Tato nepřesnost je však při výsledném poslechu zanedbatelná.



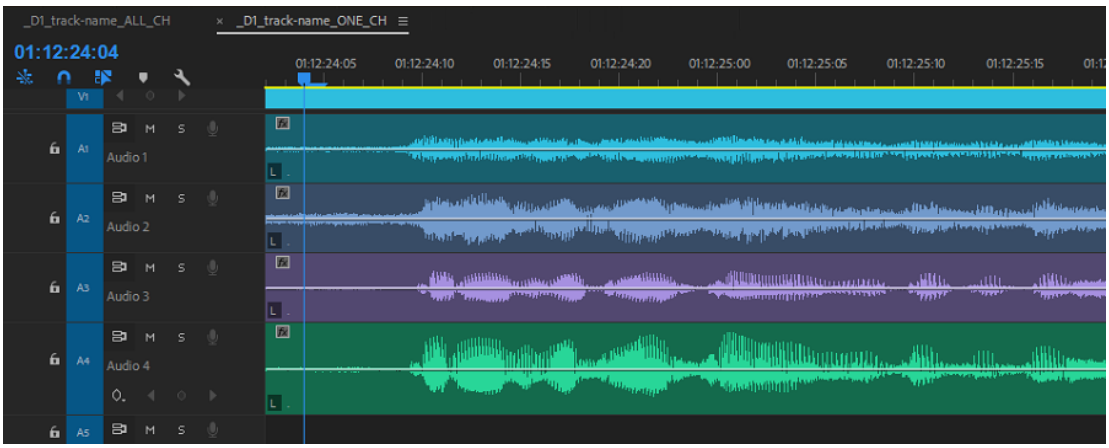
Obrázek 19: Ukázka výstupu projektu naimportovaného do Adobe Premiere Pro.

Poslední obrázek, číslo 21, demonstruje nález chyby ve videu, konkrétně typu WARNING. Jedná se o kompoziční chybu, kdy byla v obraze mimo řečníka nalezena i druhá osoba, která nesplňuje definovaná kritéria. Správně by měl v tuto chvíli kameraman takzvaně „přizoomovat“ tak, aby se fotograf ocitl mimo záběr.

9 Závěr

Během vytváření práce bylo zjištěno, že současné technické možnosti stále neumožňují plnohodnotné nahrazení živého kameramana či střihače při záznamu přednášek se zachováním stejné kvality. Proto je mnohem zajímavější a přínosnější zaměřit se na částečnou automatizaci procesu od zpracování dat natočených lidským nahrávacím týmem a jejich přípravy pro další editaci. V rámci této automatizace byl úspěšně navrhnut a v jazyce Python implementován univerzální systém umožňující synchronizaci všech zmíněných audiovizuálních zdrojů včetně systému detekování kompozičních chyb.

Všechna audiovizuální data ze složky zdrojových materiálů jsou načtena a jejich informace získané pomocí extrakce metadat uloženy do centrální projektové SQL databáze.



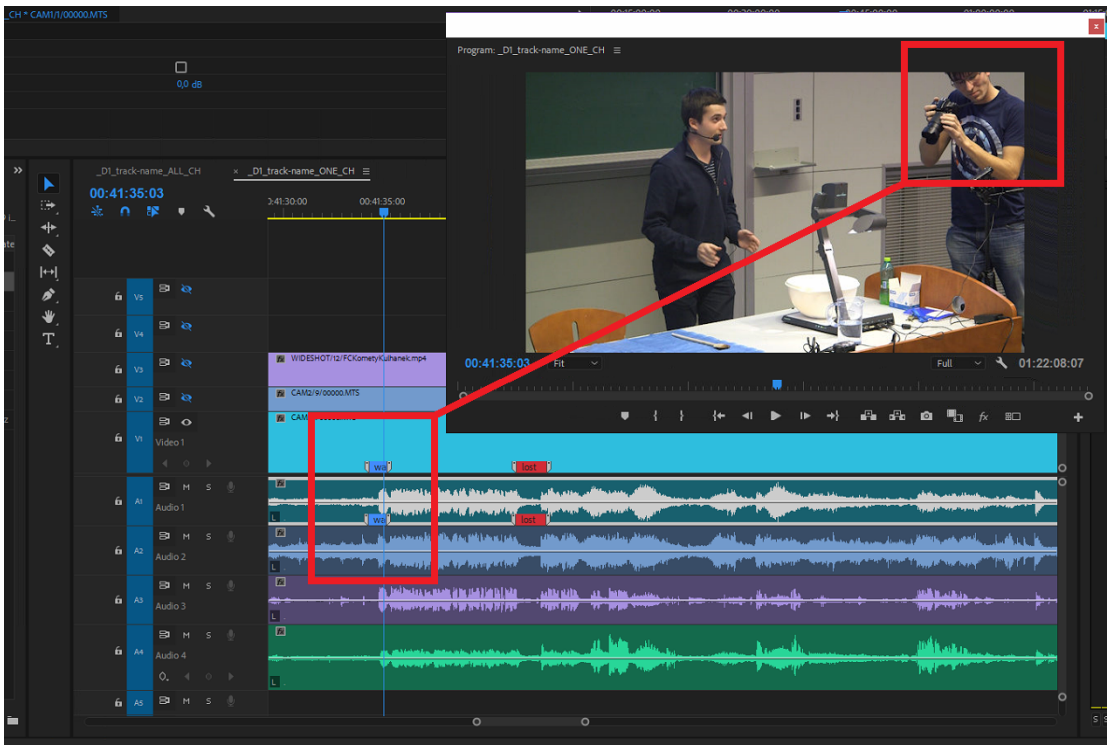
Obrázek 20: Demonstrace přesnosti výpočtu zobrazeného v Adobe Premiere Pro.

V dalším kroku jsou pak data synchronizována pomocí vzájemného porovnávání zvukových signálů reprezentovaných koeficiety mel-kepstrální analýzy a aplikací rekurzivních funkcí, kterými je vypočtena pozice každého zdroje na pomyslné časové ose.

Nedílnou částí práce je návrh aplikace kinematografických pravidel na realizaci záznamu pro zachování maximální vizuální kvality. Ty jsou částečně aplikovány v poslední části projektu, kterou je proces detekce chyb. Chyby jsou hledány pomocí detekce a trackování všech řečníků v obraze a následného porovnávání pozice jejich obličeje v obraze vůči předem stanoveným pravidlům. Systém rozeznává dva základní chybové stavy. Prvním je případ, kdy řečník v obraze zcela chybí, v druhém stavu je řečník nalezen, avšak obraz nesplňuje kritéria pro správné kompoziční rozložení.

V závěru je vytvořen přenosový projekt Final Cut Pro XML Interchange formátu, do kterého jsou převedena veškerá synchronizovaná data, utříděná do jednotlivých vrstev projektu, včetně přidání štítků označujících místa a typ nalezených chyb. Takový projekt již lze přímo otevřít ve střihovém programu Adobe Premiere Pro, který byl pro tuto práci vybrán.

V průběhu vytváření práce bylo také testováno samotné řízení kamery pomocí mikropočítače Arduino UNO s mikrokontrolérem ATmega328P. Mikropočítač řeší na základě příkazů z centrálního počítače jak pohyb servo motorů pohybujících kamerou v horizontálním a vertikálním směru, tak samotné ovládání kamery, tedy jejího zoomu, ostření a dalšího nastavení pomocí protokolu LANC vyvinutého firmou SONY. Pro testování bylo



Obrázek 21: Demonstrace nalezené chyby typu WARNING.

použito poloprofesionální kamery Canon XA25. Práce však byla v průběhu vytváření ze zmíněných důvodů, po domluvě s vedoucím práce, přeorientována spíše na řešení problému zpracování dat od reálných kameramanů.

Celý proces se přímo nabízí k další práci na jeho vylepšení. Především v oblasti detekce chyb, které jsou blíže popsány v kapitole 5. Pro detekci složitějších kompozičních nerovností je však nutné vytvoření prostorového modelu celé postavy čistě zpracováním obrazu, tedy bez použití jakéhokoliv externího senzoru. Některé metody, používající příslušné neuronové sítě, tento problém řeší, avšak dostupný výpočetní výkon je vůči jejich náročnosti stále nedostačující. Takový postup přispěje i ke snazšímu rozšíření na složitější vícekamerové systémy.

Další možností rozšíření je přidání kompletního modulu pro zpracování zvuku. Nejen pro automatické vytváření ekvalizačních filtrů pro jeho úpravu, ale také samotné rozpoznání řeči. To lze použít například pro kontrolu, že daná kamera opravdu sleduje právě hovořícího řečníka. V takovém případě se už obecně jedná o velmi komplexní téma, které

by pokrylo další práci podobného rozsahu.

Reference

[1] Hog face detection with a sliding window. URL http://sklin93.github.io/hog.html, 2016. [Online k 20.12.2018].

[2] Final Cut Pro XML Apple. Interchange format, apple computer. *Inc, September*, 23, 2006.

[3] JH BAILEY et al. Open broadcaster software. *Software, OBS Project*, 62, 2017.

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[5] McFee Brian, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, , and Oriol Nieto. librosa: Audio and music signal analysis in python. *In Proceedings of the 14th python in science conference*, pages 18–25, 2015.

[6] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1611.08050*, 2016.

[7] Han-Ping Chou, Jung-Ming Wang, Chiou-Shann Fuh, Shih-Chi Lin, and Sei-Wang Chen. Automated lecture recording system. In *System Science and Engineering (ICSSE), 2010 International Conference on*, pages 167–172. IEEE, 2010.

[8] Oscar Déniz, Gloria Bueno, Jesús Salido, and Fernando De la Torre. Face recognition using histograms of oriented gradients. *Pattern Recognition Letters*, 32(12):1598–1603, 2011.

[9] Chiung-Yao Fang, An-Chun Luo, Yu-Shan Deng, Chia-Ju Lu, and Sei-Wang Chen. Building a smart lecture-recording system using mk-cpn network for heterogeneous data sources. *Neural Computing and Applications*, pages 1–19, 2018.

[10] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[11] Magnus Hieronymus and J Nycander. Finding the minimum potential energy state by adiabatic parcel rearrangements with a nonlinear equation of state: An exact solution in polynomial time. *Journal of Physical Oceanography*, 45:150423121849009, 04 2015.

[12] Xuedong Huang, Alex Acero, Hsiao-Wuen Hon, and Raj Reddy. *Spoken language processing: A guide to theory, algorithm, and system development*, volume 1. Prentice hall PTR Upper Saddle River, 2001.

[13] D. Hulens, T. Goedemé, and T. Rumes. Autonomous lecture recording with a ptz camera while complying with cinematographic rules. In *2014 Canadian Conference on Computer and Robot Vision*, pages 371–377, May 2014.

[14] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[15] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[16] Yusnita mohd ali, Paulraj M P, Sazali Yaacob, Raghad Yusuf, and Shahriman Abu bakar. Analysis of accent-sensitive words in multi-resolution mel-frequency cepstral coefficients for classification of accents in malaysian english. *International Journal of Automotive and Mechanical Engineering*, 7:1053–1073, 06 2013.

[17] Adrian Rosebrock. Intersection over Union (IoU) for object detection. URL https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection, 2016. [Online k 20.12.2018].

[18] FFmpeg Team. Ffmpeg. *URL http://FFmpeg.org*, 2013.

[19] Jan Uhlíř, Pavel Sovka, Petr Pollák, Hanžl Václav, and Čmejla Roman. *Technologie hlasových komunikací*. Nakladatelství ČVUT Praha, 2007.

[20] J. Valušiak and Akademie múzických umění v Praze. *Základy střihové skladby: určeno pro posl. fak. filmové a televizní*. SPN, 1983.

[21] Paul Viola, Michael J Jones, and Daniel Snow. Detecting pedestrians using patterns of motion and appearance. In *null*, page 734. IEEE, 2003.

[22] Marek Zelina. Teorie střihové skladby a její postavení v audiovizuální tvorbě v roce 2010. 2010.