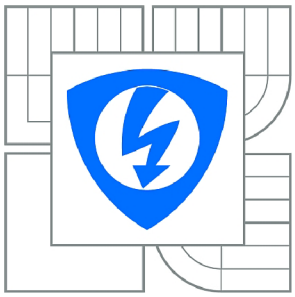
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

VIDEO MIXER/TRANSLÁTOR PRO WEBOVÉ VIDEOKONFERENCE

VIDEO MIXER/TRANSLATOR FOR WEB-BASED VIDEOCONFERENCE SYSTEM

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

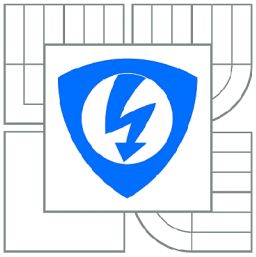
AUTOR PRÁCE — JAN HUTAR

AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR ČÍKA, Ph.D.

SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Jan Hutar	***ID:***	155115
Ročník:	3	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Video mixer/translátor pro webové videokonference

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte video kodeky H.264, VP8 a souvislosti mezi nimi. Najděte nástroje, které se dají použít na realtimové překódování z jednoho kodeku na druhý. Vyberte ten nejvhodnější a vytvořte pomocí něj serverovou aplikaci typu video mixer/translator. Aplikace bude schopna přijímat multimediální data a v reálném čase je mixovat a transkódovat za předem daných podmínek.

DOPORUČENÁ LITERATURA:

[1] MANSON, Rob. Getting Started with WebRTC. London: PACKT PUBLISHING, 2013. ISBN 978-1782166306.

[2] GOOGLE. WebRTC [online]. 2011-2012 [cit. 2013-10-16]. Dostupné z: http://www.webrtc.org/

Termín zadání:	9.2.2015	***Termín odevzdání:***	2.6.2015

Vedoucí práce: Ing. Petr Číka, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Bakalářská práce je zaměřená na základní popis, srovnání a zpracování video standardů H.264 a VP8 v reálném čase. Jedná se především o podporu videokonferenčních systémů, které pracují s těmito vzájemně nekompatibilními standardy. Práce prozkoumává a testuje dva nejvýznamnější softwarové prostředky vyhovující základním požadavkům na práci s videem v reálném čase, dostupné s LGLP licencí a to FFmpeg a GStreamer.
Práce navrhuje postup provádění video mixu vstupních datových proudů a způsob provedení překódování do požadovaného výstupního proudu. Pro realizaci a zprávu mixeru/translátoru je použit nástroj FFmpeg a vytvořeno jednoduché rozhraní dostupné z webového prohlížeče za pomocí podpory HTML, PHP a Javascript.

KLÍČOVÁ SLOVA

standard, H.264, VP8, kodek, FFmpeg, GStreamer, mixér, translátor, reálný čas

ABSTRACT

Bachelor´s thesis specialises in a basic description, comparing and processing of video standards H.264 and VP8 in real time. Particularly it concerns a support of videoconference systems working with those mutually incompatible standards. The work enquires and tests two most considerable software devices convenient with basic requirements for working with video in real time and available with LGLP licence respectively FFmpeg and GStreamer.
The thesis suggests practice video mix implementation of data input and the way of code translation into required outgoing stream. FFmpeg tool has been used for realization and message of mixer/translator and simple interface has been established available from web browser with assistance of HTML, PHP and Javascript.

KEYWORDS

standard, H.264, VP8, codec, FFmpeg, GStreamer, mixer, translator, real time

HUTAR, Jan *Video mixer/translátor pro webové videokonference*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2014. 52 s. Vedoucí práce byl Ing. Petr Číka, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Video mixer/translátor pro webové videokonference“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Petru Číkovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

(podpis autora)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské pr byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

(podpis autora)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

V dnešní době došel technický pokrok v telekomunikacích do stavu, kdy již není otázkou dostupnost telekomunikačních služeb, ale jejich kvalita. Dá se říci, že na jakémkoliv místě na této planetě je možné se připojit k telekomunikační službě, jak za využití pevné sítě, tak mobilní sítě pozemní nebo družicové. Poskytovatelé telekomunikačních služeb se tedy v dnešní době snaží zvýšit kvalitu služeb tak, aby si udrželi svého zákazníka.

Vysoká kvalita, velká dostupnost a nízká cena služeb dala prostor rozvoji videokonferencím. Obrovský telekomunikační prostor s téměř nekonečným množstvím účastníků vytváří na trhu místo pro komerční i nekomerční produkty. Je velmi mnoho možností, jak videokonference realizovat. Od použití jednoduché aplikace na mobilní telefon nebo počítač až po velmi sofistikované zařízení navržené jen k tomuto účelu. Již v roce 1996 přišla společnost ITU-T (*International Telecommunication Standardization sector*) se standardem H.323 jako doporučením pro přenos videokonference. Doporučení obsahuje standardy H.26x pro přenos komprimovaného videa. K tvorbě videokonferencí lze také použít protokol SIP (*Session Initiation Protocol*). Vzájemná komunikace mezi zařízeními pracujícími s protokolem H.323 a SIP je realizována bránou H.323/SIP, která zabezpečuje propojení signalizace. Spojení samotného přenosu již v obou případech probíhá prostřednictvím RTP (*Real Time Protocol*). Výrobci zařízení nevyužívající doporučení ITU-T H.323 si vytvořili vlastní projekty. Například společnost Google, Inc. využívá vlastní video standard VP8. Standard VP8 patří do kontejneru projektu Webm. Společnost Google se snaží o co nejsnadnější cestu, jak umožnit jednotlivým uživatelům používat videokonferenční hovory a to za pomocí webového prohlížeče. Tento projekt se jmenuje WebRTC (*Web Real-Time Communication*). Dále je také podporován a rozšiřován ve spolupráci s prohlížeči Opera a Mozzilla Firefox. Chceme-li provést video konferenci mezi zařízením standardu H.323 a projektu WebRTC, je nezbytné použít server, který bude zajišťovat nejen signalizaci, ale také transkódování z jednoho standardu do druhého a obráceně v průběhu probíhající videokonference.

Problematikou převodu mezi video standardy H.264 a VP8 se zabývá tato práce. Cílem je zprovoznit vhodnou aplikaci na serveru, která bude provádět mix příchozích video streamů do jednoho výsledného. Tento výsledný video mix bude transkódovat do vhodného video standardu a to v reálném čase.

V první části práce je stručně popsán princip komprimování videa, standardy H.264 a VP8 včetně jejich porovnání. Další kapitola se zabývá principem konverze mezi standardy a vhodnými softwarovými nástroji. V poslední kapitole je popsáno zprovoznění a ovládání serveru z příkazového řádku, nebo za pomoci vytvořeného jednoduchého rozhraní z webového prohlížeče.

1 POPIS VIDEO STANDARDŮ VP8 A H.264

1.1 Video komprese

Potřeba video komprese vznikla zároveň se vznikem digitálního záznamu videa. Nekomprimovaný digitální video soubor zabírá přibližně 32 MB za jednu sekundu na pevném disku. Takovýto objem dat a případný datový tok pro přenos je v podstatě nemožné přenášet dostupnými prostředky. Nekomprimované video se tedy používá pouze k zpracování před jeho konečnou distribucí. Použití video komprimace je tedy nezbytné k snížení datových objemů na výrazně nižší hodnotu tak, aby bylo možné je jednoduše poskytnout konečnému uživateli. S komprimovaným digitálním videem se dnes setkáváme na výstupu všech multimediálních záznamových a zobrazovacích prostředků, také u digitální televize a internetu.

Komprimované digitální video se vytváří pomocí kodéru, který využívá algoritmy navržené a popsané v některém z mnoha standardů. Video standardů je v současné době nepřeberné množství, lze je rozdělit do dvou skupin s ohledem na jejich licenci. Jednu skupinu tvoří licencované standardy, kde jsou použité algoritmy patentovány a za jejich užití se platí poplatek[1]. Druhou skupinu tvoří svobodné standardy, kde tvůrce standardu poskytl zdrojový kód a označil jej jednou z mnoha svobodných licencí, např. BSD (*Berkeley Software Distribution*).

Video komprese vytváří z nekomprimovaného video signálu komprimovaný, takový signál má několikanásobně menší nároky na datové úložiště nebo datový tok. Komprese se provádí redukcí redundantních a irelevantních složek. Velikost komprese se vyjadřuje kompresním poměrem 1.1 CR (*Compres Ratio*)[1], který je definován poměrem vstupního a výstupního signálu viz vzorec

$$CR = \frac{R_{vst}}{R_{výst}}[\ ]. \tag{1.1}$$

Odstranění redundance je bezztrátový proces oproti odstranění irelevantních složek. Odstranění irelevantních složek je nevratně ztrátový proces. Velikost takto vzniklých ztrát již velmi ovlivňuje zpět dekódovaný video signál.

- **Redundance** v obraze je taková informace, která se několikrát po sobě stále opakuje, např. stále stejná barva na určité ploše. U video signálu ji lze použitím vhodného algoritmu eliminovat a snížit tak množství přenášených informací. Takovouto eliminaci lze opět rekonstruovat. Redundance nenastává u náhodného šumu. Dále se dělí na **prostorovou** a **časovou**[2].

[1]Koncový spotřebitel má poplatek již zahrnut v ceně koupeného zařízení, které podporuje daný standard, např. DVD přehrávače, apod..

- **Irelevance** u video signálu je taková obrazová informace, která není pro lidské oko téměř viditelná nebo není vidět vůbec. Odstranění irelevantních složek je nevratné a provádí se například průměrováním. Po provedení průměrování lze informaci ještě redukovat[2].

1.1.1 Barevný model RGB a YUV

Barevný model používaný pro zobrazení obrazu na monitoru televizi nebo projektoru je RGB[2]. RGB signál je reprezentován složením barev červená, zelená a modrá. Všechny ostatní barvy vznikají sloučením těchto barev v určitém poměru. Dá se tedy vycházet z předpokladu, že barvy RGB jsou použity rovnoměrně a jsou nenahraditelné. Pokud bychom chtěli provést kompresi, byla by minimální a neefektivní. Pro zpracování obrazového signálu se využívá transformovaný model YUV, z kterého se vypočítává nejpoužívanější model YCbCr.

Model YUV znamená oddělení jasové složky Y od barevné informace U a V[3], z modelu RGB se vypočítá maticí s převodním poměrem.

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

Tento model vychází z vlastností lidského oka, které je méně citlivé na jasové složky obrazu.

Model YCbCr obsahuje jak jasovou složku Y, tak barvonosné složky Cr 1.2, Cg 1.3 a Cb 1.4, které se vypočítají vztahem [2]

$$C_r = R - Y, \tag{1.2}$$

$$C_g = G - Y, \tag{1.3}$$

$$C_b = B - Y. \tag{1.4}$$

Složku Cr není nutné přenášet, protože se dá vypočítat dosazením ostatních hodnot.

Při kompresi videa se luminanční a chrominanční složka komprimuje a zpracovává stejným způsobem, ale zvlášť a nejčastěji se využívají čtyři základní modely formátování a vzorkování modelu YCbCr. Model udává poměr, v jakém je vzorkován signál, tedy $Y : Cb : Cr$ z jednotlivých bloků. Například standard VP8 a H.264 využívá poměr 4:2:0 označený jako YV12, kde je rozlišení horizontální Cr a Cb oproti Y poloviční. Tato úprava video signálu je již nevratná a jedná se o první snížení původní kvality.

[2]RGB jsou počáteční písmena barev v anglickém jazyce, Red - červená, Green - zelená, Blue - modrá.

[3]Y - luminance, U a V - chrominance.

1.1.2 Redukce redundance časové, prostorové a entropické

- **Redundance časová** vychází z principu pohybové kompenzace. Pohybová kompenzace se provádí jednosměrně nebo obousměrně, tedy mezi snímky, které budou následovat, i mezi snímky, které již byly. Princip je takový, že uvnitř kodéru se používá několik druhů snímků. V určitém intervalu se přenese snímek kompletní jako je třeba JPEG a mezitím se pracuje pouze se snímky, které jsou v kodéru vytvořeny a obsahují pouze změny. Není tedy nutné komprimovat každý snímek kompletně, ale stačí předvídat změnu nebo opakující se pohyb v obraze. Neprohledává se celý obraz najednou, ale po částech proměnlivé velikosti, například makroblok 16 x 16 pixelů nebo blok 8 x 8, na které je rozdělen obraz. Velikosti bloků závisí na použitém kodéru [3].
  V současných kodérech se používají dva typy predikce a to Intrapredikce a Interpredikce. **Intrapredikce** analyzuje daný snímek jako samotný, bez ohledu na předchozí a následující snímky. **Interpredikce** se používá k odhadu obsahu bloků s ohledem na předchozí snímky. Hlavní součástí tohoto procesu jsou referenční snímky a vektory pohybu. Snímek je pak sestaven tak, že se vezme počáteční hodnota referenčního snímku a provede se porovnání, výsledkem je posuv a definuje jej vektor pohybu. Takto se získá konečná pozice skupiny pixelů ve snímku.
- **Redukce prostorová** využívá DCT (*Discrete Cosine Transform*), nebo také DWT (*Discrete Wavelat Transform*), kdy dochází k převodu do frekvenční oblasti. Výsledek transformace je matice prvků. V případě DCT se v levém horním rohu matice nachází informace o stejnosměrné složce obrazu. Čím více se na matici pohybujeme doprava dolů, tím vzrůstá množství informací o detailech obrazu, které se od určitého bodu stávají irelevantní a je možné je zaokrouhlit, případně odstranit [3].
- **Redukce entropie** probíhá při zdrojovém kódování a využívá ke kódování například Huffmanův kód VLC (*Variable Length Coding*) s proměnnou délkou slova. Entropické kódování v závislosti na četnosti výskytu prvku udává slovu jeho bitovou velikost. Často vyskytujícím se prvkům se přiřazuje slovo s malou bitovou velikostí, oproti prvkům s malou četností výskytu, které mají bitovou velikost mnohem větší [3].

1.1.3 Výsledná kvalita

Výsledná kvalita komprimovaného videa velmi závisí na požadavcích uživatele, kvalitě použitého standardu a nastavení kodéru při vytváření komprimovaného videa. Mezi ovlivňující parametry patří použité kvantizační tabulky, rychlost a počet průchodů kodérem. Jako další požadavek, který velmi ovlivňuje výslednou kvalitu je výsledný bitový tok. Snižováním bitového toku dochází k velkému nárůstu blokových artefaktů, kdy jsou jasně viditelné hrany jednotlivých bloků.

1.2 Video standard VP8

Standard VP8 je od května 2010 pod BSD[4] licencí. Byl vyvinut společností On2 Technologies, kterou na začátku roku 2010 převzala společnost Google Inc. Ta standard zařadila do svého projektu Webm [12] a následně uvolnila licenci.

Hlavním cílem standardu VP8 bylo vytvoření přímé konkurence standardu H.264, který byl v té době licencován jako nesvobodný. V současné době je standard VP8 velmi hojně využíván a podporován a je především znám pod názvem multimediálního kontejneru Webm. Současné HTML5 (*Hyper Text Markup Language 5*) má již také zahrnuté přehrávání tohoto formátu. Stejně tak mnoho aplikací, jako je např. YouTube, WebRTC atd. Dále také všechny majoritní webové prohlížeče tento standard podporují. Standard VP8 má již svého nástupce a to standard VP9, u kterého byl dokončen vývoj v červenci 2013. Tento formát je hardwarově výrazně náročnější. Velkou výhodou těchto standardů, a obecně kontejneru Webm, ve kterém jsou definovány i audio standardy, je jejich svobodná licence a tedy volná šiřitelnost.

1.2.1 Základní popis

Standard VP8 je primárně navržen pro přenos videa po webu. Hlavní požadavky při jeho návrhu byly malá šířka přenosového pásma, využití na co největším množství zařízení, sledovatelnost videa i při velmi nízké bitové rychlosti [5]. Blokové schéma kodéru VP8, obrázek 1.1, je shodné se standardem H.264.

Standard používá rozdělení nastavení kodéru dle kvality výsledně zpracovaného videa a rychlosti zpracování: „Encode Quality Speed“. Možná nastavení kodéru jsou uvedena níže.

- **Best** kvalita je nejlepší možná, které lze dosáhnout. Používá dva průchody kodérem. Je velmi pomalá a časově náročná[12].
- **Good** kvalita je dobrá a asi nejpoužívanější, lze zde nastavit 6 rychlostních stupňů využití procesoru[12].

[4]BSD licence je licencí pro svobodný software, umožňuje volné šíření licencovaného obsahu.

- **Real-time** o kvalitě rozhoduje rychlost kodéru. Tento parametr určuje výslednou kvalitu obrazu, stejně tak je velmi podstatný výsledný datový tok. Například používá jen jeden průchod kodérem, stejně tak menší rozlišení s sníženým počtem snímků za sekundu[12].

Standard umí zpracovat video o rozlišení až 16383 x 16383 pixelu, tedy mnohem více než 4K. Vstupní formát kodéru je RGB, který je transformován na YUV 4:2:0 [6].

Přeuspořádání snímků

Jednotlivé snímky jdou za sebou v takzvaných GOP (*Group of Pictures*). V GOP jdou snímky ve vhodném pořadí pro přehrávání, tedy dekódování. V případě kódování však kodér potřebuje snímky dostávat v jiném pořadí, protože provádí jednosměrnou nebo obousměrnou predikci, v některých případech některé typy snímků nepoužívá vůbec. Snímky jsou tedy přeskládány do potřebného pořadí. Standard VP8 používá snímky dvou typů a to I a P[5] [6].

Intrapredikce

Při odhadu obsahu obrazu standard VP8 používá 4 intrapredikční mody a to DC, Horizontální, Vertikální a TrueMotion. Módy používají velikosti makrobloků 4 x 4 nebo 16 x 16 luminančních a 8 x 8 chrominančních [6].

Intrepredikce

V tomto bloku probíhá odhad pohybového vektoru objektů v obraze. K tomu se používají tři referenční snímky nazvané klíčové snímky. Klíčové snímky obsahují Golden reference frame GF, Last frame LF a Alternate reference frame AF.
Golden reference frame je snímek uložený v bufferu. Využívá se na zobrazení a dopočítání pozadí scény při pohybu. Stejně tak při přeskoku scény jinam a zpět. Při videokonferenci je využit jako obnovovací snímek[5].
Last frame je jeden z předešlých snímků Alternate reference frame.
Alternate reference frame slouží jako referenční snímek pro vylepšení zobrazení ostatních interpredikčních snímků, nebývá zobrazován. Je tvořen několika zdrojovými snímky tak, aby bylo dosaženo co největší komprese[5].

Odhad pohybu se provádí v jednotlivých makroblocích o velikosti 16 x 16 pixelů, které mohou být rozděleny až na 16 subbloků. Odhad probíhá s Qpel[6].

[5] Označení snímků vychází s terminologie používané v MPEG.
[6] Qpel je Quater-pixel precision, v překladu z angličtiny přesnost na čtvrt pixelu.

Transformace

Pro převod prostorové oblasti do frekvenční se používá DCT nebo WHT, v závislosti na velikosti bloků a použitém módu predikce [6].

Kvantizace

Je použitá adaptabilní kvantizace, kdy je stanovena jedna kvantizační matice pro všechny makrobloky v jednom snímku [6].

Entropické kódování

Kodér používá pseudo Huffmanův stromový kód s jedním pravděpodobnostním rozdílením pro celý snímek [6].

Filtr

Provádí se rekonstrukcí snímků a vyhlazení hran jednotlivých bloků rozmazáním. Standard VP8 používá velmi adaptabilní filtr a hodnoty jeho nastavení záleží na aktuálním referenčním snímku [6].

1.3 Video standard H.264

Standard H.264 vznikl ve spolupráci ITU (*International Telecommunication Union*) a ISO (*International Organiation for Standardization*) a MPEG (*Moving Pricture Experts Group*). Jako označení se používá MPEG-4 AVC (*Advanced Video Coding*) part 10, nebo jen H.264. Je patentován a licencován u MPEG LA. Licence pro soukromé užití je volná, jinak je jeho použití zpoplatněno. V roce 2013 přišla společnost Cisco s vlastní implementací standardu H.264 jako OpenH264 [8] pod BSD licencí. Společnost Cisco sama řeší poplatky za licenci. Jedná se o velmi pokrokovou ideu, která umožnila rozšíření využití standardu OpenH.264. Například velmi rozšířený webový prohlížeč Firefox do uvolnění licence nepodporoval standard H.264 a jeho implementace se musela řešit vhodným pluginem. I přes nevýhodu licence je standard velmi hojně využíván třeba sdělovacími prostředky pro vysílání digitální televize apod.

V současné době je již ukončen vývoj nastupujícího standardu H.265. Jeho širší využití se očekává v následujících letech stejně jako u nastupujícího standardu VP9.

Velká nevýhoda standardu H.264 je tedy licence, kterou se snažily některé projekty obejít vytvořením vlastní interpretace, a to standardem x.264.

1.3.1 Základní popis

Standard H.264 byl navržen pro multimediální aplikace nové generace, jako je IPTV (*Internet Protocol Television*), digitální televize, HDTV (*High-definition Television*) atd. Při návrhu se vycházelo ze starších standardů MPEG-2, MPEG-4 Visual a H.263[7]. Při návrhu byl kladen důraz na velikost bitového proudu, tedy na malou šířku pásma. Pro standard H.264 platí téměř totožné požadavky jako na video kodek VP8. Požadavek na co nejširší hardwarové využití je také shodný. Stejně tak blokové schéma kodéru je totožné, viz obrázek 1.1.

Při nastavování kodéru lze použít předem předdefinované profily, které mají předurčené použití, tři z nich jsou uvedeny níže. Kodér je také možné nastavit podrobněji s ohledem na rychlost průchodu a obsah vstupu. Rychlost lze volit od nejrychlejší *ultrafast* po nejpomalejší *pacebo*. Pro nastavení obsahu videa lze použít například parametr *animation, film* nebo třeba *zerolatency* pro živý přenos. Člení se na několik profilů, které definují výstupní kvalitu videa.

- **Baseline Profile** BP je určený především pro videokonference a mobilní aplikace s menším výpočetním výkonem. Je zde kladen důraz na rychlost oproti výsledné kvalitě. Nepoužívá snímky B, pracuje s menším rozlišením a menším počtem snímků za sekundu.
- **Main Profile** MP je hlavní profil, dnes nahrazen profilem HP. Již není tolik vyžíván.
- **High Profile** HP je dnes používaný jako hlavní profil, využívá se pro přenos HDTV nebo ukládáni videa na Blue-Ray disky.

Přeuspořádání snímků

Jednotlivé snímky jdou za sebou v takzvaných GOP (*Group of Pictures*). V GOP jdou snímky ve vhodném pořadí pro přehrávání, tedy dekódování. V případě kódování potřebuje však kodér snímky dostávat v jiném pořadí, protože provádí jednosměrnou nebo obousměrnou predikci, nebo v některých případech některé typy snímků nepoužívá vůbec. Snímky jsou tedy přeskládány do potřebného pořadí. Standard H.264 používá snímky I, P a B[7].

Intrapredikce

K předpovědi obsahu obrazu je používáno až 9 módů s různou velikostí makrobloků. Makrobloky mohou být o velikosti 16 x 16 nebo čtyři bloky 8 x 8, případně šestnáct sub-bloků 4 x 4 body. Předpověď probíhá v rámci jednoho makrobloku. Pokud to není možné, využívá pixelů sousedních makrobloků[7].

Interpredikce

Pro odhad pohybu je použito až 16 referenčních snímků, skutečný počet použitých snímků je omezen především velikostí vyrovnávací paměti. Kromě snímků I, P a B ještě přibyly snímky SI a SP oproti předešlému MPEG-4 Visual. Při kompenzaci pohybu je používána proměnlivá velikost bloků s Qpel [7].

Transformace

Pro převod videa do frekvenční oblasti je použita celočíselná DCT nebo WHT, výsledkem je sice nižší komprese, ale rychlejší a jednodušší proces transformace.

Kvantizace

Použitá je adaptabilní kvantizace. Kvantizační matice je upravována jednotlivými bloky tak, aby docházelo k co nejmenším ztrátám při převodu vysokých frekvencí vzniklých po transformaci.

Entropické kódování

Pro entropické kódování standardu H.264 se používá pevná VLC tabulka pro snadnou implementaci bez nutnosti provádět statistickou redundanci.

Filtr

Vykonává vyhlazení hran jednotlivých bloků, jeho funkce a adaptabilita je téměř schodná s filtrem užitým u standardu VP8.

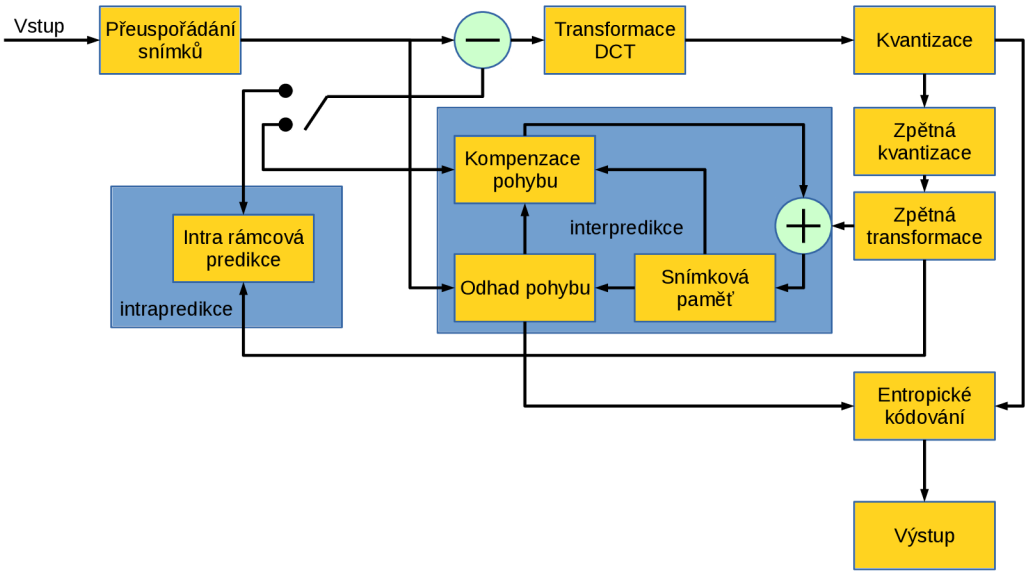
1.4 Porovnání video kodeků VP8 a H.264

Základní rozdíly mezi video kodeky VP8 a H.264 jsou vyjmenovány v tabulce 1.2[4]. Není jednoduché posoudit, který standard je lepší nebo kvalitnější. Porovnání se provádí subjektivní[7] nebo objektivní[8] metodou.

Video kodek VP8 nepoužívá snímky B a tedy jeho výsledný bitový tok je o něco větší, výhodou nepoužití těchto snímků je ale nižší hardwarová náročnost při zpracování. Nevýhoda kodeku VP8 je malé rozpětí možného nastavení chodu, protože kvalitu a rychlost sloučil do jednoho parametru o třech možnostech.

[7] Výzkumem a zkoušením na divácích.

[8] Některým z mnoha softwarových nástrojů .



Obr. 1.1: Kodér VP8 a H.264

Kodek H.264 má velkou výhodu v možnostech nastavení rychlosti a kvality a tím je i dána jeho variabilita, kterou lze přesněji nastavit pro konkrétně požadovanou aplikaci.

Pro potřeby videokonferencí, kdy jde především o rychlost a ani jeden kodek nepoužívá snímky B, jsou tyto rozdíly naprosto zanedbatelné, viz stejné schéma kodéru, obrázek 1.1. Větší rozdíly se projevují až při vyšších nárocích na kvalitu a rychlost zpracování [4]. V hodnocení používanosti a rozšířenosti těchto dvou kodeků je jednoznačným vítězem H.264.

1.5 Multimediální kontejner

Kontejner je obálka multimediálních datových souborů, která sdružuje data různých formátů a standardů do jednoho souboru. Obsahuje zejména jednu nebo více video a audio stop, dále pak soubory s titulky a menu. Kontejner sám o sobě nenese žádnou informaci o vnitřní kompresi uložených dat, ta je určena použitým kodekem. Při přehrávání je zajištěna synchronizace a uživatel sám volí prostřednictvím přehrávače, které datové stopy chce přehrát. V rámci jednoho kontejneru bývají například uloženy audio soubory v různých jazycích. Pro přehrávání nebo vytváření obsahu je zapotřebí muxer nebo demuxer. Demuxer provádí otevření obálky a rozděluje jednotlivé datové proudy do vhodných kodérů výstupního zařízení. Muxer je používán při tvorbě multimediálního kontejneru, kdy jsou jednotlivé datové proudy sloučeny dohromady. Formáty kontejnerů se liší podle jejich schopností a mezi nejznámější patří: AVI, MPEG-PS, MPEG-TS, ASF, MP4, FLV, WEBM, MKV, atd.

Postup kódování	**H.264**	**VP8**
Profily	BP (*Baseline Profile*)	Real-time
	MP (*Main Profile*)	Normal
	HP (*High Profile*)	Best
Itrapedikce	Používá 9 módů pro jednotlivé různé velikosti bloků, jejich množství a velikost záleží na použitém profilu.	Používá 4 módy s velikostí makrobloků 4 x 4 a 16 x 16 luminiscenčních bodů a 8 x 8 chrominančních bodů.
Interpredikce	Podporuje až 16 referenčních snímků.	Používá tři refenční snímky.
	Druhy rozdělení na 16 x 16, 16 x 8, 8 x 16 bloky 8 x 8 mohou být rozděleny na 8 x 8, 8 x 4, 4 x 8 nebo 4 x 4.	Druhy rozdělení na 16 x 16, 16 x 8, 8 x 16, 8 x 8 a 4 x 4.
	Chrominační pohybové vektory jsou počítány přímo z Luminančních vektorů pohybu.	Chrominanční pohybové vektory jsou průměrovány z pohybových vektorů makrobloků.
	Interpolační filtr Qpel používá šestikrokovou interpolaci na lumeny a bilinearní na chrominační vektory.	Interpolační filtr Qpel používá šestikrokovou interpolaci na lumeny a čtyř nebo šesti na chrominační.
	Používá snímky I, P, B, SI a SP.	Používá snímky I, P, GF, LF a AF, snímky B nejsou použity.
Transformace	Celočíselná DCT => menší komprese a větší rychlost.	DCT nebo WHT => pomalejší a výpočetně náročnější.
Kvantizace	Adaptabilní pro jednotlivé makrobloky.	Adaptabilní pro jednotlivé snímky.
Entropické kódování	Je použit adaptabilní aritmetický kodér.	Je použit aritmetický kodér bez adaptability.

Obr. 1.2: H.264 vs VP8

1.5.1 Kontejner mp4

Kontejner mp4 je definován standardem ISO/IEC 14496-14:2003 a je znám pod názvem MPEG-4 part 14. Nahrazuje zastaralý kontejner AVI. Kontejner může mimo video a audio stop obsahovat ještě menu a velké množství titulků. Kontejner je určen pro velké množství multimediálních formátů, podrobnosti jsou uvedeny zde [13]. Například audio-standardy podporuje MP3 a AAC, jako video-standardy podporuje H.26x atd. Přehrávání nepodporují všechny webové prohlížeče.

1.5.2 Kontejner mkv

Multimediální kontejner je velice rozšířený a vytvořil jej nezávislý projekt Matrjoška. Kontejner mkv vnitřně umožňuje obsáhnout téměř jakýkoliv multimediální datový formát, podporuje titulky, menu, indexaci, kapitoly atd. To je jeho velkou výhodou, protože může obsahovat standardy společnosti MPEG LA a také společnost Inc. Google. Dle popisu na stránkách výrobce zatím nepodporuje standard VP8, ale pouze jeho starší verze [14].

1.5.3 Kontejner webm

Kontejner webm je založen na kontejneru Matrjoška (mkv). Kontejner webm je určen jen pro tento projekt a může tedy obsahovat jen multimediální formáty tohoto projektu, jako je například audio-standard Vorbis a video-standard VP8 [12].

2 NÁSTROJE PRO PŘEKÓDOVÁNÍ VP8 A H.264

2.1 Možnosti konverze mezi standardy

Konverze mezi jednotlivými formáty, které vzniknou kódováním určitého standardu, se provádí až na vyjímky stejným způsobem. Video soubor se dekóduje zpět do formátu YUV, který je také označen jako raw[1] a následně je zakódován jiným algoritmem do požadovaného formátu. Takto provedená konverze je vždy ztrátová. Ztráty vznikají především tak, že každý kompresní algoritmus používá jiné způsoby redukce irelevantních složek. Pokud je to možné, je nejlepší ukládání do jiného formátu jen z originálu. Vyhneme se tak několikanásobným průchodem různými kodéry a dekodéry.

2.2 Softwarové nástroje pro konverzi video kodeků v reálném čase

Hledání vhodného nástroje bylo zaměřeno pouze na nástroje, které jsou opatřeny některou ze svobodných licencí a jejich použití nepodléhá žádnému zpoplatnění. Další kritérium byla nezávislost na platformě, neboli operačním systému, na kterém pracují. Těmito kritérii pak prošly dále zmíněné.

Aplikací vhodných ke konverzi mezi video kodeky H.264 a VP8 je velké množství. Po bližším prozkoumání se ukázalo, že jen některé aplikace dokáží pracovat v realtime režimu konvertování. Po další podrobnější selekci vyplynulo zjištění, že většina aplikací vychází v podstatě ze tří frameworků, a to FFmpeg, Libav a Gstreamer. Protože jsou to vlastně základní kameny většiny používaných přehrávačů a transkoderů, mezi které patří například VLC* (*VideoLAN Client*), Arista transkodér, MPlayer atd., rozhodl jsem se použít právě tyto frameworky a vybrat z nich nejvhodnější. Velkou výhodou těchto nástrojů je, že nejsou zatíženy žádným GUI (*Graphical User Interface*), stejně tak jejich funkce lze přesně definovat a omezit pouze na konkrétní funkci a tím vším snížit vlastní režii nástroje.

[1] Raw z anglického slova surový, v technice nekomprimovaná data.

2.2.1 FFmpeg a Libav

Nástroj projektu Libav[9] byl vyřazen z výběru z důvodu popsaném v dalším odstavci, následující popis bude zaměřen pouze na nástroj FFmpeg [10].

Projekt Libav vznikl rozdělením projektu FFmpeg, kdy část původních vývojářů přestala souhlasit se směrem, kterým se FFmpeg začal ubírat. Projekt Libav se postupně ubírá svou cestou, ale základní myšlenka a forma zůstává stejná. Funkce jsou přejmenovány, ale knihovny zůstaly stejné. Vzhledem k této neodladěnosti od FFmpeg jsem software Libav neinstaloval, protože jsem se obával potenciálních kolizí mezi oběma softwary. Dle dostupných zdrojů [9] je zřejmé, že se jedná o naprosto totožný nástroj, jako je FFmpeg a obecný popis bude odpovídat i Libav.

Projekt FFmpeg založil Fabrice Bellard v roce 2000. Následně jej v 2004 převzal Michal Niedermayer. Název pochází z MPEG a FF z "fast forward". Cílem projektu je vytvořit svobodný software, který je schopen přehrávat, konvertovat a streamovat většinu dostupných video a audio formátů. V první řadě se vývojáři zaměřili na implementaci stávajících formátů, někdy i velmi složitou cestou, jako je reverzní inženýrství. Některé tyto způsoby porušují patenty v některých státech, zejména v USA (*United States of America*) a jejich použití je v těchto státech nelegální[10].

Na základě zkušeností vyvinuli vývojáři vlastní standardy včetně kontejneru, nebyly nikdy masivně nasazeny.

Struktura FFmpeg se dělí na nástroje a knihovny.

Nástroje

- **ffmpeg** utilita do příkazového řádku pro konvertování, grabování a real-time konvertování audio a video formátů.
- **ffserver** je HTTP a RTSP mutimediální streamovací server pro živý přenos s podporou posunu času.
- **ffplay** jednoduchý audio a video přehrávač založený na SDL (*Simple Direct-Media Layer*) a FFmpeg knihovnách.
- **ffprobe** utilita do příkazového řádku pro zobrazení informací o konkrétním mediu.

Knihovny

- **libavcodec** obsahuje všechny audio a video enkodéry a dekodéry.
- **libavformat** obsahuje demuxery a muxery pro audio a video kontejnerové formáty.
- **libavutil** obsahuje generátor náhodných čísel, datovou strukturu, matematické rutiny, jádro multimediálních utilites a mnoho dalšího.

- **libavdevice** knihovna umožňuje komunikovat se zařízeními přes multimediální rozhraní.
- **libavfilter** obsahuje filtry.
- **libswscale** obsahuje rutiny pro změnu rozlišení a barevného modelu videa.
- **libswresample** implementuje optimalizované převzorkování audia a změnu jeho formátu.

Ovládání

Veškeré ovládání programu FFmpeg se odehrává v příkazovém řádku. Nastavování požadovaných vlastností programu se provádí pomocí konkrétních parametrů a vhodné syntaxe. Kompletní návod je k nalezení na stránkách projektu FFmpeg [10]. Zde je ukázka pár nejzákladnějších příkazů:

Spustí program FFmpeg a provede konverzi souboru input.mp4 do souboru output.avi, parametr -i označuje vstupní soubor;

```
ffmpeg -i input.mp4 output.avi
```

Spustí FFserver dle konfiguračního souboru a spustí stream na zvolenou adresu http://...;

```
ffserver -f doc/ffserver.conf &
ffmpeg -i INPUTFILE http://localhost:8090/feed1.ffm
```

Spustí v okně přehrávání video soubor;

```
ffplay video.mkv
```

Otevře a prozkoumá soubor, vypíše podrobnosti o způsobu komprimace.

```
ffprobe video.mkv
```

```
Input #0, matroska,webm, from '/home/kachnizon/Stazene/s.
   mkv':
  Metadata:
    encoder         : libebml v1.3.0 + libmatroska v1.4.1
    creation_time   : 2014-10-27 14:32:37
  Duration: 02:00:46.19, start: 0.000000, bitrate: 1826
     kb/s
    Stream #0:0: Video: mpeg4 (Advanced Simple Profile) (
       XVID / 0x44495658),
     yuv420p, 848x384 [SAR 1:1 DAR 53:24], 23.98 fps,
        23.98 tbr, 1k tbn, 23.98 tbc (default)
    Metadata:
      title           : Mat
    Stream #0:1(cze): Audio: ac3, 44100 Hz, stereo, fltp,
        128 kb/s (default)
    Metadata:
      title           : CZ
    Stream #0:2(cze): Subtitle: subrip (default)
    Metadata:
      title           : CZ
```

Shrnutí

Seznámení s nástrojem FFmpeg probíhalo velice dobře, jedná se o velmi kvalitní nástroj s dobrou podporou. Nástroj FFmpeg je velmi hojně využíván mnoha projekty a je poměrně obecně znám. Není problém o něm najít mnoho podrobných návodů včetně zkušeností uživatelů.

S instalací jsem neměl problém. Probíhala následujícím způsobem, protože mnou používaný systém Ubuntu 14.04 nemá FFmpeg zahrnut v repositáři. Ze stránek projektu FFmpeg bylo potřeba aktuální vydání FFmpeg−2.4.3.tar.bz2 nahrát ručně. Po rozbalení archivu je nutné provést komplici a tvorbu spouštěcího souboru ffmpeg_2.4.3−1−i386.deb, který se již spustí a zavede z repositáře. Takto nainstalovaný program již obsahuje všechny potřebné nástroje i knihovny pro svou funkci a je možné jej okamžitě začít používat.

Několik základních funkcí jsem si bez potíží snadno otestoval. Pokud uživateli postačují základní nastavení, nepotřebuje velkou znalost parametrů.

2.2.2 GStreamer

Projekt GStreamer [11] je multimediální framework založený na grafu filtrů. Je naprogramován v jazyce C jako objektově orientovaný systém. Primárně je distribuován v desktopovém prostředí GNOME. Jinak je multiplatformní a je opatřen jednou ze svobodných licencí pro volné šíření a užívání.

Projekt založil v roce 1999 Erik Walthinsen. Mnoho nápadů pro návrh a řešení jádra nacházel v Oregon Graduate Institute. První vydání bylo v roce 2001 a jeho popularita rychle rostla zejména u uživatelů a programátorů na Linuxové platformě. Velké množství společností a aplikací používá jádro GStreameru a patří mezi ně například Nokia, Motorola, Intel a mnoho dalších[11].

Struktura GStreameru je založena na filtrech, nástrojích a pluginech, které sdílejí různé knihovny.

Vybrané nástroje

- **gst-launch** přehrávač;
- **xmllaunch** spouští XML vytvoření například v gst-editoru;
- **gst-inspect** zobrazí podrobnější informaci o souboru a obsahu kontejneru;
- **gst-editor** aplikace umožnující jednoduché skládání filtru a tvorbu pipeline s exportem do XML.

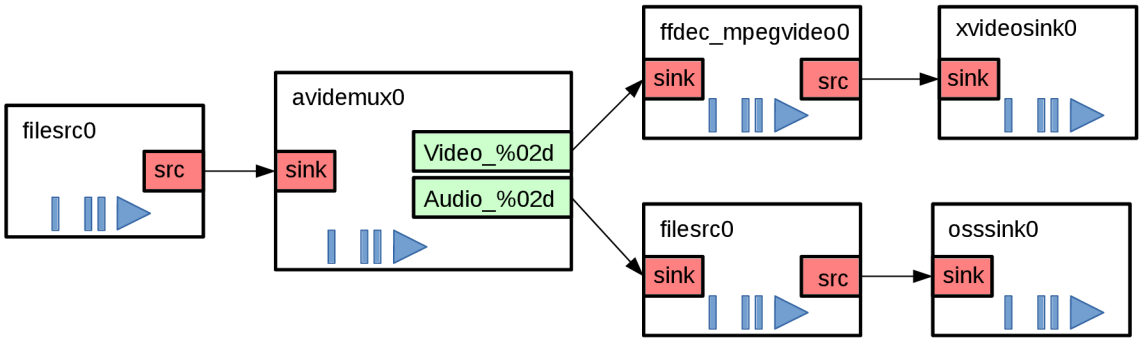
Vybrané pluginy

- **gst-plugins-base** základní zásuvný modul obsahující široké spektrum knihoven i funkcí samotného programu;
- **gst-plugins-good** obsahuje kontejnery s kodery a dekodery nejkvalitněji zpracované s licencí LGLP, zároveň mají nejlepší podporu od GStreameru;
- **gst-plugins-bad** obsahuje kontejnery a kodeky, které se kvalitou velmi blíží obsahu gst-plugins-good, ale nemají dostačující dokumentaci nebo širokou podporu;
- **gst-plugins-ugly** obsahuje takové kodeky a kontejnery, které by mohly mít problém s legálností;
- **gst-libav** obsahuje více jak 100 kodeků, kontejnerů a standardů, které jsou transparetně poskytovány FFmpeg/Libav;
- **gst-rtsp-server** je multimediální server s podporou HTTP, RSTP, streamování živého vysílání atd.

Ovládání se provádí v příkazovém řádku pomocí pipeline[2], obrázek 2.1. Každý komponent programu je samostatný objekt se vstupem a výstupem, který se pomocí

[2]Pipeline je anglické slovo pro potrubí.

pipeline propojí. Pro snadnější ovládání lze použít gst-editor, v kterém se jednotlivé objekty propojí a výsledek je možné exportovat do XML souboru. Podrobný návod lze nalézt zde `http://docs.gstreamer.com/display/GstSDK/Home`.



Obr. 2.1: Příklad pipeline

Pár příkladů pipeline:

Spustí se se přehrávač v samostatném okně s testovacím obrazcem, obrázek 2.2;

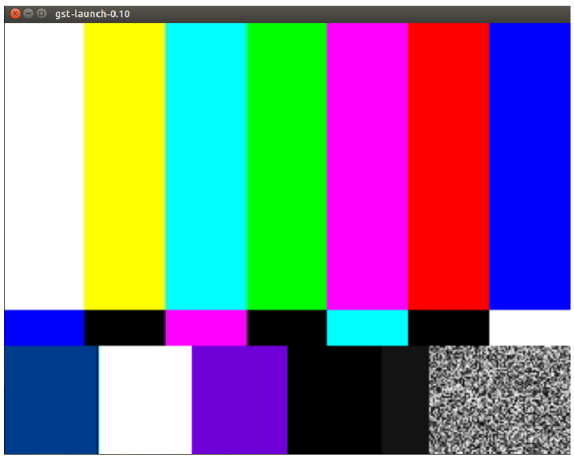
```
gst-launch-0.10 videotestsrc ! ffmpegcolorspace !
   autovideosink
```

Spustí gst-lunch jako zdroj **src** media použije uvedenou cestu **location**, otevře kontejner matroska **demux** a obsažené video vloží do kontejneru matroska **mux**, nakonec celý video soubor uloží na požadované místo **location**;

```
gst-launch-0.10.exe souphttpsrc location=http://docs.
   gstreamer.com/media/sintel_trailer-480p.webm !
   matroskademux name=d d.video_00 ! matroskamux !
   filesink location=sintel_video.mkv
```

Otevře webm kontejner a dekóduje audio a video a uloží je do kontejneru ogg na požadované umístění;

```
gst-launch-0.10 uridecodebin uri=http://docs.gstreamer.
   com/media/sintel_trailer-480p.webm name=d ! queue !
   theoraenc ! oggmux name=m ! filesink location=sintel.
   ogg d. ! queue ! audioconvert ! audioresample !
   flacenc ! m.
```



Obr. 2.2: Testovací obrazovka

Vypíše podrobnosti o souboru a v následující podobě vypíše.

```
gst-discoverer-0.10 video.mp4
```

```
kachnizon@kachnizon:~$ gst-discoverer-0.10 /home/
   kachnizon/Stazene/d.flv
Analyzing file:///home/kachnizon/Sta%C5%BEen%C3%A9/d.flv
Done discovering file:///home/kachnizon/Sta%C5%BEen%C3%A9
   /d.flv

Topology:
  container: Flash
    video: H.264
    audio: MPEG-4 AAC

Properties:
  Duration: 0:52:00.033333333
  Seekable: yes
  Tags:
      trvani: 3120033333333
      kodek zvuku: MPEG-4 AAC
      kodek videa: H.264
```

Shrnutí

Nástroj GStreamer je velmi precizně zpracovaný, stejně tak má velmi velkou a kvalitní vývojářskou podporu. Je velmi oblíbený a slouží jako základní kámen mnoha projektů. I přes jeho velké využívání není tolik znám mezi menšími uživateli jako FFmpeg. Stejně tak kromě oficiálního návodu není jednoduché najít rady a zkušenosti jednotlivých uživatelů. Pro běžného uživatele může být odrazující jeho o něco složitější ovládání a větší rozsah příkazů. Od vývojářů je zpracován velmi kvalitně a podrobně, ale bohužel je také značně obsáhlý návod, jehož prostudování zabere mnohem víc času než u FFmpeg.

Instalace a zprovoznění mi zabralo o trochu více času, ale jeho postup je vlastně obdobný jako u FFmpeg, viz předchozí popis. Po instalaci samotného gst-streameru je nutné ještě doplnit některé zásuvné moduly. Nepodařilo se mi dokončit kompilaci nástroje gst-editor, při kompilaci se neustále vyskytovaly chyby. Bez nástroje editor všechny důležité části fungují tak, že se bez něj dá pracovat. Bohužel je škoda jeho absence z důvodu exportu do XML pro usnadnění zadávání příkazů.

Po instalaci jsem provedl několik úspěšných testů, které proběhly bez problémů. Pro začátek a testování je náročnější syntaxe pro příkazy do řádku a tvorbu pipeline. Ovládání je pro začátek složitější než u FFmpeg.

2.3 Porovnání a výběr

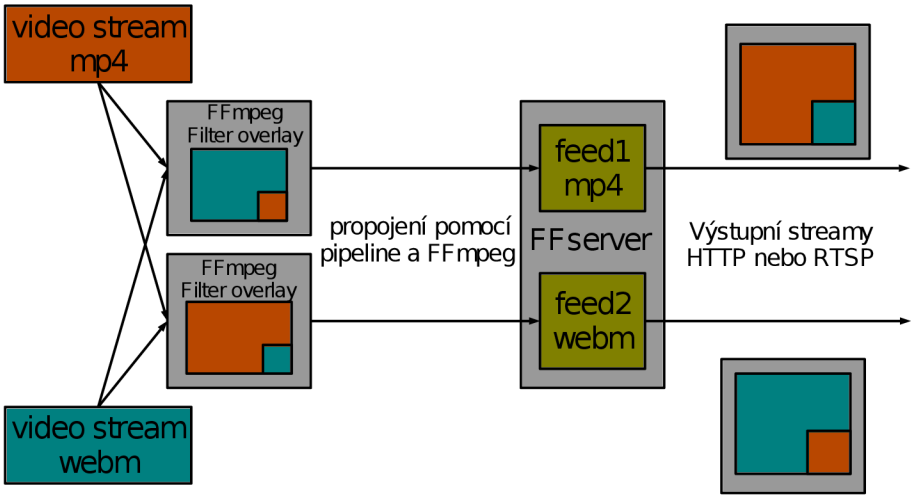
Srovnání nástrojů FFmpeg a GStreamer není jednoduché. Oba tyto frameworky jsou napsány velmi kvalitně a mají velmi velkou vývojářskou podporu. Pro splnění zadání bakalářské práce dokáží oba posloužit velice dobře. Mezi oběma nástroji je však pár rozdílů. Například FFmpeg má nástroj, který si spouští dle zadání jednotlivé knihovny, GStreamer je postaven na zásuvných modulech obsahující potřebné nástroje.

Pro bakalářskou práci jsem se rozhodl použít nástroj FFmpeg z důvodu jednodušší syntaxe pro nastavení konkrétní funkce. Nástroj GStreamer je složitější nástroj a rozsah nastudování zabere mnohem více času, než mi dovoluje realizací bakalářská práce.

Nástroj FFmpeg vyhovuje všem kriteriím real-time translátoru a mixéru mezi standardy H.264 a VP8. Nástroj GStreamer také vyhovuje těmto požadavkům, z důvodu neúspěšnosti zprovoznění všech nástrojů, složitosti ovládání a menšího rozšíření nebyl pro práci použit.

3 REALIZACE SERVEROVÉ APLIKACE

Pro realizaci mixéru/translátoru mezi standardy H.264 a VP8 byl vybrán a použit nástroj FFmpeg. Funkce aplikace na serveru je následující. Příchozí video streamy jsou nejprve mixovány dohromady. Vznikne tak jeden video stream, který obsahuje všechny vstupní streamy dohromady vhodně rozvržené. Tento výstupní stream je pomocí pipeline a FFmpegu odeslán na vstup serveru FFserver. FFserver zajistí převod do požadovaných výstupních formátů. Na výstupu FFserveru je RTP stream dostupný v podobě RTSP nebo HTTP. Parametry výstupních streamů je možné nastavit při spouštění FFserveru. Tímto způsobem je realizován jeden uživatelský průchod. Pro další variace výstupů je nutné provést odlišné video-mixy, pokud by byly požadovány. Větší počet mixovaných výstupů dovede pak obsloužit jeden FFserver s větším množstvím definovaných průchodů. Na obrázku je znázorněno blokové schéma pro dva uživatele s rozdílnými požadavky na vstupy a výstupy, obrázek 3.1.



Obr. 3.1: Blokové schéma mixéru a translátoru pro dva uživatele

V průběhu videokonference je přenášen obraz i zvuk zároveň v jednom multimediálním kontejneru. Celý řetězec mixéru/translátoru je řešen tak, aby bylo možné na vstup připojit multimediální kontejner. Obsah kontejneru je rozdělen na jednotlivé streamy, které jsou zpracovány zvlášť. Na výstupu FFserveru jsou převedeny do požadovaných formátů a opět zapouzdřeny do vhodných kontejnerů.

Pro práci s standardem H.264 je zvolen multimediální kontejner mp4, pro standard VP8 multimediální kontejner webm.

Dále jsou navrženy dva způsoby možného nastavování a ovládání. První možný způsob je formou příkazů a parametrů do terminálu, případně vzdáleně přes zabezpečený přístup pomocí SSH. Druhý je pomocí webového rozhraní, které bylo

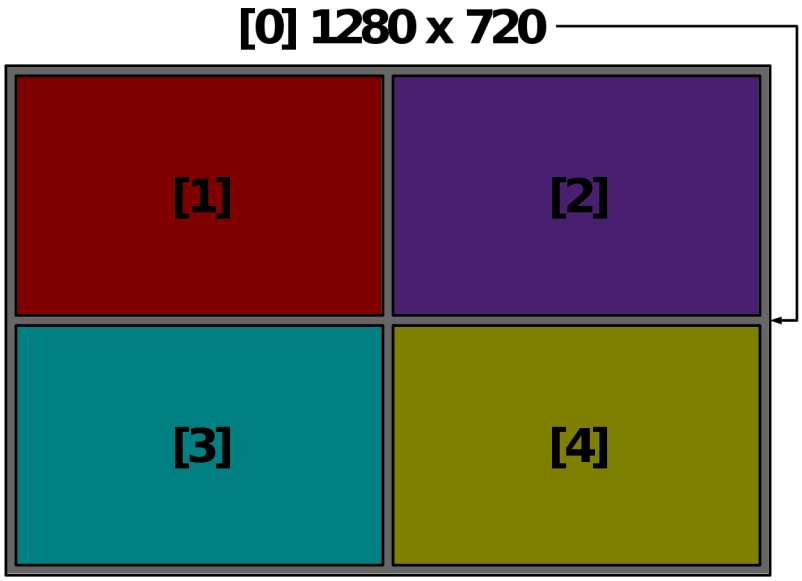
vytvořeno v rámci bakalářské práce za pomocí HTML, PHP a Javascriptu. Ovládání za pomocí webového rozhraní je předem upraveno tak, aby uživatel bez velkých znalostí ovládání mohl na serveru připojit RTP vstupy do soketů, nastavit FFserver, nadefinovat podobu mixu a na další stránce si spustit náhled výstupů.

Oba způsoby je možné provádět vzdáleně s tím, že práce s terminálem vyžaduje větší znalosti, ale také umožňuje mnohem větší rozsah nastavení.

3.0.1 Video mix

Mixování jednotlivých video proudů do jednoho výsledného proudu je běžnou součástí každého videokonferenčního řetězce. V případě, že spolu komunikují dva uživatelé, by bylo možné výsledný mix provádět až na koncovém zařízení u každého uživatele. Protože ale videokonferenční zařízení podporují komunikaci většího množství uživatelů, nebylo by možné každému jednotlivému příjemci posílat několik jednotlivých datových toků. Mnohem efektivnější je jednotlivé datové toky propojit na serveru a každému uživateli poslat jeden výsledný proud.

Video mix prostřednictvím aplikace FFmpeg je možné provádět prostřednictvím filtru *-filter_ complex*, jehož součástí je parametr *-overlay* [15]. Rodina filtru *-filter_ complex* pod sebou skrývá velkou rodinu možných video filtrů pro zpracování a úpravu videa [**?**].

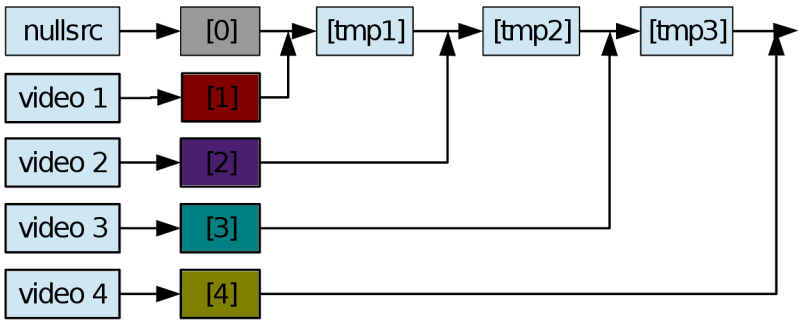


Obr. 3.2: Příklad výsledného video mixu

Funkce je poměrně jednoduchá. Je třeba nadefinovat vstupy parametrem *-i*, následně parametrem *nullsrc=size=1280x720 [0]* velikost rámu, kam budou jednotlivá videa vkládána. Další parametr *[0:v] scale=640x360 [1]* udává velikost jednotlivého

vloženého obrazu, číslo v závorce jeho pořadí. Následující parametry definují umístění jednotlivých obrazů na ose x a y do prázdného předem definovaného rámu *[0][1] overlay=shortest=1 [tmp1]; [tmp1][2] overlay=shortest=1:x=640 [tmp2]; [tmp2][3] overlay=shortest=1:y=360 [tmp3]; [tmp3][4] overlay=shortest=1:x=640:y=360*, je možné i jednotlivě vložená videa vzájemně překrývat. Po provedení videomixu je nutné ještě definovat formát výstupních dat, tedy video, audio standard a výsledný kontejner *-c:v libx264 -c:a aac output.mkv*.

Schéma jednotlivého provádění škálování a vkládání videa do prázdného rámu je znázorněn na obrázku 3.3.

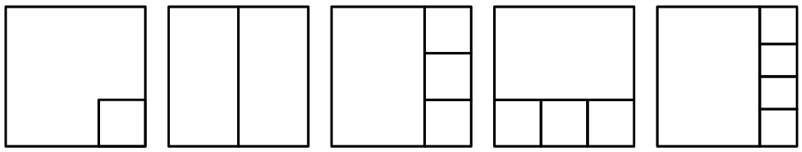


Obr. 3.3: schéma funkce filtru overlay

Příklad zápisů parametrů video mixu a výsledný vzhled, obrázek 3.2.

```
ffmpeg
-i 1.avi -i 2.avi -i 3.avi -i 4.avi
-filter_complex "
nullsrc=size=1280x720 [0];
[0:v]  scale=640x360 [1];
[1:v]  scale=640x360 [2];
[2:v]  scale=640x360 [3];
[3:v]  scale=640x360 [4];
[0][1] overlay=shortest=1 [tmp1];
[tmp1][2] overlay=shortest=1:x=640 [tmp2];
[tmp2][3] overlay=shortest=1:y=360 [tmp3];
[tmp3][4] overlay=shortest=1:x=640:y=360 "
-c:v libx264 -c:a aac output.mkv
```

Příklad možných video mixů pro videokonference, obrázek 3.4, které jsou použity ve webové aplikaci.



Obr. 3.4: Příklady video mixu

3.0.2 Nastavení parametru FFmpeg

Základní příkaz je uveden níže. Není-li již doplněn o další přepínače provede převod s totožnými parametry (jako je kvalita, bitová rychlost, rozměr, počet snímků za sekundu atd.) stejně, jako má vstupní video souboru input.h264. Výstupní standard nebo kontejner je určen koncovkou výstupního souboru, v tomto případě kontejnerem webm nebo parametrem přepínače.

```
ffmpeg -i input.h264 prepinac output.webm
```

Další příkaz RTP stream na vstupní feed FFserveru.

```
ffmpeg -i rtsp://[IP adresa streamu:port/stream.mp4 http
   ://[IP adresa serveru:port]/feed.ffm
```

Za vstupní soubor na pozici přepínač lze doplňovat parametry zpřesňující výstupní soubor. Všechny jsou uvedené v dokumentaci projektu FFmpeg [10], zde je vybráno pár nejpoužívanějších v tabulce 3.5.

3.0.3 Nastavení parametrů FFserver

Aplikace FFserver slouží jako server k distribuci multimediálních datových proudů. Na vstupy, které se nazývají feed, lze připojit jak datový soubor, tak datový proud RTP. Jeden FFserver může obsluhovat mnoho různých datových proudů najednou. Nastavení funkce se provádí zápisem parametrů do souboru *ffserver.conf*, který je primárně čten z umístění */etc/ffserver.conf*. Součástí tohoto nastavení jsou i přesné parametry výstupních datových toků. Jako součást FFserveru je možné spustit webový status server dostupný na *http://[IP adresa serveru:port]/stat.html*, kde lze sledovat počet připojených uživatelů, objemy datových toků atd. Výstup je pak dostupný jak pomocí protokolu HTTP, tak RTSP. Nastavení serveru lze rozdělit do čtyř částí. V první části se nastavují obecné parametry jako port, počet připojených uživatelů nebo maximální šířka přenosového pásma. V druhé části se definují vstupy serveru nazývané feed. Ve třetí se nastavují parametry výstupních streamů a jejich propojení s vstupy. V poslední čtvrté části přesměrování z stránky status serveru. Nevýhoda aplikace FFserver je, že parametry nelze měnit za chodu, ale aby se změny projevily, je server nutné vždy po úpravě restartovat [21].

Přepínač	Funkce	Příklad
-i	určuje vstupní soubor	-i
-b	datový tok bit/s	1024k je 1024 000bit/s
-s	velikost výsledného videa v pixelech	1024x720
-targer pal-dvd	nastaví výsledné parametry tak, aby výsledné video bylo vhodné k tvorbě DVD disku, lze použít i mnohem více příkazů uvedených v příkladu	vcd, svcd, dvd, dv, dv50, pal-, ntsc-, film-
-vcodec	za přepínač lze doplnit požadovaný standard pro video	-vcodec libxvid
-acodec	za přepínač lze doplnit požadovaný standard pro zvuk	-acodec libmp3lame
-pass	počet průchodů kodérem	1 nebo 2

Obr. 3.5: Parametry

FFserver se spouští příkazem, nebo lze nastavit automatické spouštění pomocí daemona. Parametr -f označuje cestu, kde je konfigurační soubor uložen.

```
ffserver -f /etc/ffserver.conf
```

Nastavování parametrů FFserveru je velké množství a velmi záleží k jakému účelu bude server provozován. Zde uvedený popis se týká pouze nastavení potřebných pro realizaci bakalářské práce. Použitím značky # je řádek zakomentován a při načítání parametrů je vynechán.

Základní nastavení

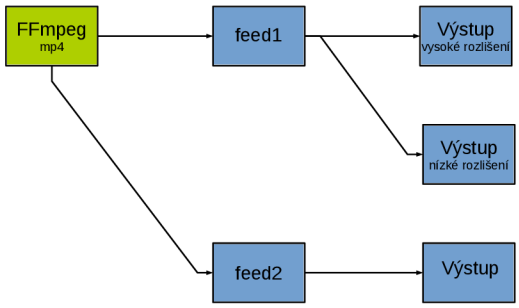
Nastavení portů FFserveru, počet připojených klientů, maximální velikost datového toku, CustomLog bude vytvářet standardní záznam do souboru o průběhu činnosti serveru.

```
Port 8090
RTSPPort 8080
RTSPAddress 0.0.0.0
BindAddress 0.0.0.0
MaxHTTPConnections 10
MaxClients 10
MaxBandwidth 10000
CustomLog -
NoDaemon
```

Feed

Nastavení vstupů feed *<Feed>Parametry</Feed>* používá zápis do párového tagu. Feed1.ffm je název vstupu, File je adresář dočasného úložiště s definovanou velikostí. ACL allow omezuje adresy vstupu na za parametrem uvedené. Jeden vstup může být použit pro několik zpracování. Je typické jeden vstup rozdělit na několik s rozdílným rozlišením a bitovým tokem tak, aby se uživatel připojil na ten, který odpovídá jeho přístupu k síti a zařízení, obrázek 3.6.

```
<Feed feed1.ffm>
        File /tmp/feed1.ffm
        FileMaxSize 200k
        #ACL allow 127.0.0.1
</Feed>
```



Obr. 3.6: Příklad propojení vstupů a výstupů FFserver

Nastavení parametrů streamu pro kontejner mp4

Nastavení výstupního datového proudu *<Stream>Parametry</Stream>* používá zápis do párového tagu. Provádí nastavení formátu výstupního datového proudu rtp. Výstup bude dostupný na *rtsp://IP adresa serveru:RTSPport/mix.mp4*. Feed definuje, který vstup má být použit. Dále je uvedena knihovna video standardu, počet snímků, rozlišení, velikost bitového toku videa, nastavení video kodéru na ultrafast, nastavení standardu zvuku, atd. Lze vyřadit zvuk nastavením parametru NoAudio. Další parametry, které nejsou uvedeny, ale jsou potřeba, nastaví FFserver při startu na primárně přednastavené, například velikost bufferu apod.

```
<Stream mix.mp4>
        Format rtp
        Feed feed1.ffm

        VideoCodec libx264
        VideoFrameRate 25
        VideoBitRate 512
        VideoSize 320x240
        AVOptionVideo crf 23
        AVOptionVideo preset ultrafast
        AVOptionVideo flags +global_header

        AudioCodec aac
        Strict -2
        AudioBitRate 128
        AudioChannels 2
        AudioSampleRate 44100
        AVOptionAudio flags +global_header
</Stream>
```

Nastavení parametrů kontejneru webm

Nastavení výstupního datového proudu *<Stream>Parametry</Stream>* používá zápis do párového tagu. Provádí nastavení formátu výstupního datového proudu na multimediální kontejner webm. Výstup bude dostupný na *http://IP adresa serveru:HTTPport/live.webm*. Feed definuje, který vstup má být použit. Dále je uvedena knihovna video standardu, počet snímků, rozlišení, velikost bitového toku videa, nastavení video kodéru na realtime, využití procesoru, použití jen I snímku.

```
<Stream live.webm>
        Format webm
        Feed feed1.ffm

        VideoCodec libvpx
        VideoFrameRate 25
        VideoBitRate 512
        VideoSize 720x300
        AVOptionVideo qmin 8
    AVOptionVideo qmax 30
        AVOptionVideo quality realtime
        AVOptionVideo flags +global_header
        AVOptionVideo cpu-used 9
    VideoBitRate 1200
    VideoIntraOnly

        AudioCodec vorbis
        Strict -2
        AudioBitRate 128
        AudioChannels 2
        AudioSampleRate 44100
        AVOptionAudio flags +global_header

</Stream>
```

Status server

Nastavení status serveru dostupného na *http://[IP adresa FFserveru:port]/stat.html*. Parametr ACL určuje povolený rozsah IP adres, které se mohou na server připojit.

```
<Stream stat.html>
        Format status
        ACL allow localhost
        ACL allow 192.168.0.0 192.168.255.255
</Stream>
```

3.0.4 HTML5, a Javascript

HTML5

Jedná se o značkovací jazyk pro tvorbu webových stránek. HTML5 je poslední vydanou verzí. Nejvýznamnější rozdíl oproti starším verzím je přímá podpora multimédií v prohlížeči. Zpracování HTML kódu probíhá přímo v prohlížeči a funguje bez připojení k internetu, nevyžaduje žádnou kompilaci. Jeho výhodou je, že výsledek při tvorbě webové stránky je prakticky hned viditelný. Pro psaní jsou používány párové a nepárové tagy. Pomocí HTML se velmi dobře vytváří formuláře. Toho jsem využil při tvorbě aplikace, kdy pomocí formuláře

```
<form action="" method="post">
  <input type="datetime" name="" value="">
</form>
```

jsou zadány parametry nastavení FFmpegu a FFserveru. Parametry jsou dále předávány na server PHP. Také jsem využil podpory přehrávání multimediálního obsahu pro zobrazení výstupu z FFserveru.

```
<video width="320" height="240" controls>
  <source src="http://192.168.2.3:8090/video.webm" type="
     video/webm">
</video>
```

Javascript

Javascript je skriptovací, multiplatformní, objektově orientovaný jazyk, který je používán pro dynamické webové stránky. Javascript používaný pro webové stránky je takzvaný asynchronní. Znamená to, že se nahraje společně s webovou stránkou při načtení a vykonává pak funkce jen na straně uživatele. Nevýhoda je, že uživatel nemusí mít prohlížeč s podporou javascriptu nebo jej může mít zakázaný a tím interakce stránky nebude fungovat. Kód javascriptu se zahrnuje zároveň při psaní webové stránky mezi tagy <script></script>, nebo je jej možné uložit do samotného souboru s označením js a ten zahrnout do hlavičky webové stránky. Funkce javascriptu jsou vykonávány hned, jak na ně prohlížeč při čtení kódu narazí. V aplikaci jsem použil jednoduché skripty pro generování nových řádků do formuláře.

Plugin VLC media player

Webové prohlížeče v základu nepodporují přehrávání videa zapouzdřeného v RTSP protokolu, je nutné použít vhodný zásuvný modul, který to umožní. Protože VLC

má velmi rozšířený a jednoduchý přehrávač, dostupný pro většinu používaných prohlížečů je použit také pro realizaci aplikace, více o zásuvném modulu zde [22].

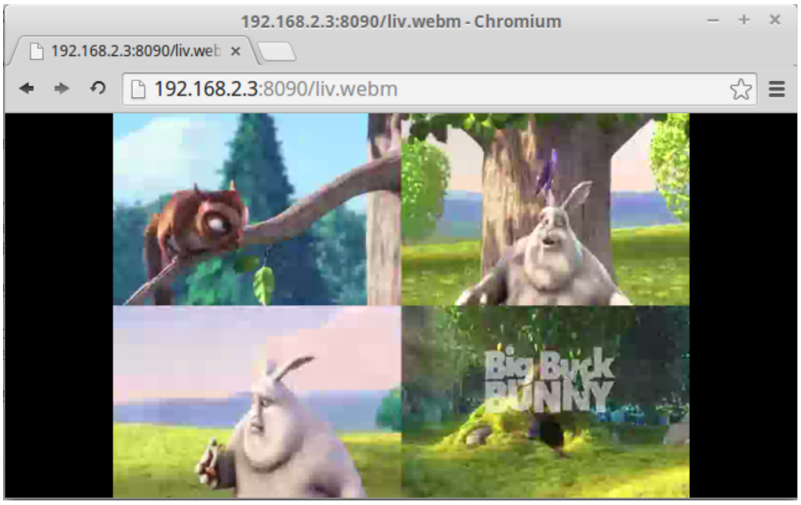
3.0.5 Realizace

Na provoz serverové aplikace je nutné použít tyto aplikace: FFmpeg, FFserver, Apache server, PHP server, SSH - openssh-server, editor Nano, VLCwebPlugin. Jednoduchý popis, včetně odkazů na podrobnější popisy, je uveden níže.

Jako server byl použit přenosný počítač ASUS X51L s procesorem Intel Pentium Dual CPU T2370 1,73GHz 2x s operační pamětí 2GB. Jako operační systém byl použit Linux Ubuntu 14.04 32-bit.

Ovládání a správa serveru je možná pomocí terminálu, při vzdálené správě je použito spojení SSH. Pro editaci parametrů FFserveru do souboru *ffserver.conf* prostřednictvím terminálu lze použít editor Nano.

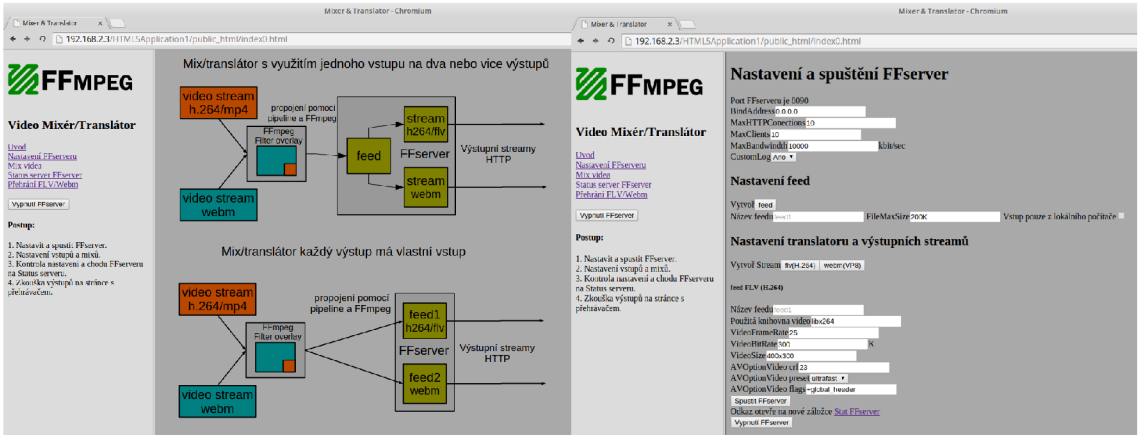
Pro snadnější ovládání serveru byla vytvořena jednoduchá aplikace dostupná z webového prohlížeče, obrázek 3.8, která umožňuje také shlédnutí výstupních streamů FFserveru, obrázek 3.7.



Obr. 3.7: Příklad zobrazení výstupu FFserver

FFmpeg a FFserver

Popisem kompilace, instalace a zprovoznění se již zabývá kapitola 2.2.1 FFmpeg a Libav [10].



Obr. 3.8: Ukázka z aplikace

Apache server

Apache HTTP server[1] je v současné době nejrozšířenější webový server s svobodnou licencí. Verze použitého softwaru: Apache/2.4.7. Po provedení instalace webového serveru Apache je přes libovolný webový prohlížeč přístupná vlastní webová aplikace na adrese `localhost/`. Na této adrese je možný přístup do adresáře /var/www, odkud jsou serverem spouštěné webové stránky. Instalace na server je možná v podobě balíčku LAMP[2], nebo jenom samotný a to příkazem *sudo apt-get install apache2* [20].

PHP server

PHP (*Hypertext Preprocessor*)[3] je skriptovací jazyk určený především pro programování dynamických internetových stránek. Při použití PHP jsou skripty prováděny na straně serveru a uživateli je přenášen až výsledek. Verze použitého softwaru: PHP Version 5.5.9-1ubuntu4.5. Instalace je možné provést tak, jak bylo uvedeno výše a to balíčkem LAMP, nebo samostatně *sudo apt-get install php5 libapache2-mod-php5* [20] .

SSH

SSH (*Secure Shell*) je program pro zabezpečenou komunikaci pomocí terminálu. Aby bylo možné SSH komunikaci používat, je nutné na vzdálený počítač nainstalovat SSH server, v případě systému Linux je to balíček openssh-server. Balíček

[1] `https://httpd.apache.org/`

[2] Jedná se o populární kombinaci programů vhodných pro webový server a to Apache2, PHP a MySQL databáze.

[3] `http://php.net`

Název souboru	Rozlišení š x v	Video [kbit/s]	Zvuk [kbit/s]	Celkem [kbit/s]	Velikost [MB]
BBB_ x264_ 1920x1080.mp4	1920 x 1080	2493	128	2621	195,3
BBB_ x264_ 300x200.mp4	300 x 200	301	128	429	32
BBB_ VP8_ 1920x1080.webm	1920 x 200	2492	128	2620	195,2
BBB_ VP8_ 300x200.webm	300 x 200	423	112	535	39,9

Obr. 3.9: Video sekvence

lze do systému doplnit pomocí repositáře, nebo za pomocí příkazu *sudo apt-get install openssh-server*. SSH server se automaticky spouští po naběhnutí systému. Pro přístup používá port 22, který je nutný povolit na firewallu, pokud je zakázán. Nejjednodušší přístup z vzdáleného počítače je příkazem *ssh <uživatelské_ jméno>@<jméno_ počítače nebo IP adresa>* [19].

Editor Nano

Jedná se o malý a jednoduchý nástroj pro editaci textových souborů a jeho hlavní předností je jednoduchost a dostupnost prostřednictvím terminálu [18]. Program Nano lze nainstalovat příkazem do terminálu *sudo apt-get install nano* a spustit příkazem *nano*.

Použité video sekvence

Vzorky video sekvencí, které byly pro testování použity, jsou z animovaného filmu Big Buck Bunny. Film byl celý vytvořen jako propagace open source softwaru. Na stránkách organizace xiph.org [17] je volně stažitelná pro účely testování nekomprimovaná verze ve formátu y4m FullHD o velikosti 44,5GB, nebo v podobě 14315 jednotlivých snímků ve formátu PNG (*Portable Network Graphics*) a audio stopa flac 110 MB. Délka videosekvence je 9 minut a 56 sekund. Z tohoto nekomprimovaného formátu za pomocí nástrojů FFmpeg byly vytvořeny celkem čtyři video sekvence, dvě standardem x.264 a dvě standardem VP8. Dvě sekvence s nízkým rozlišením, dvě v FullHD tedy 1080p. Všem videosekvencím bylo zachováno původních 25 snímků. Dále byly vzorky umístěny do kontejneru webm a mp4 včetně zvukové stopy v standardu Vorbis a MP3. Všechny použité video sekvence jsou obsaženy na přiloženém DVD. Všechny video sekvence přiložené na DVD slouží pouze k provádění testování v rámci této bakalářské práce.

Zdroj RTP

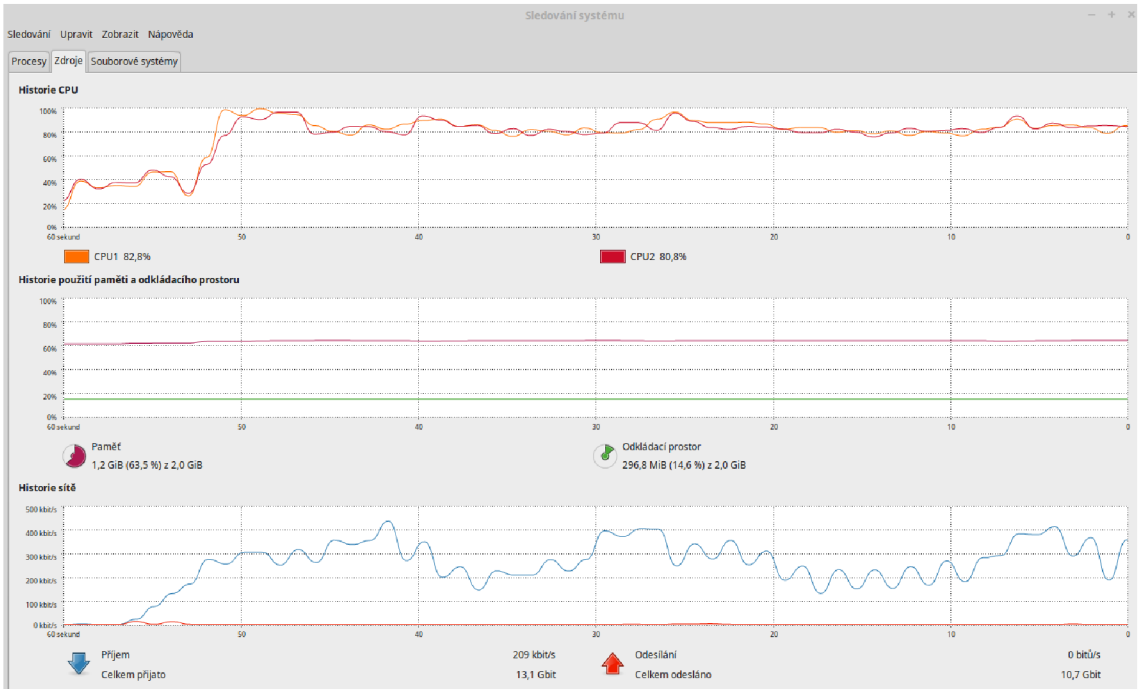
Jako zdroj RTP streamů byl používán VLC media player verze 2.2.1. VLC media player je svobodný multiplatformní multimediální přehrávač a framework s otevřeným zdrojovým kódem [16]. Mimo jiné také využívá knihovny z frameworku FFmpeg.

3.0.6 Zhodnocení hardwarové náročnosti

Po zprovoznění FFserveru bylo provedeno testování jeho schopností překódovávání v reálném čase dvou rozdílných vstupů a výstupů najednou. Při tomto testování byl rovněž vytvářen zdroj vysílání pomocí VLC media playeru na stejném počítači. Zatížení systému se pohybovalo okolo 20% v závislosti na nastavených parametrech výstupů. Nedocházelo k zahazování snímků a vše pracovalo plynule.

Při testování samotného mixu dvou proudů a ukládání do souboru již zatížení začínalo velmi stoupat k 80%. Také se ukázalo, že průchod je pomalejší než doba trvání sekvence. Pokud zpracování mixu trvá příliš dlouho, nezbývá již čas na zpracování FFserverem. To se také potvrdilo, takže na vstupu začaly být zahazovány snímky a výstup začal být trhaný. Proto bylo potřeba snížit zatížení systému a zvýšit rychlost průchodu mixérem. Snížení vytížení systému bylo dosaženo přesunutím vysílače multimediálního proudu na jiný počítač. Rychlost průchodu mixérem se zlepšila a dále se potvrdilo, že velmi záleží na nastavení rychlosti kodéru na výstupu mixéru. Dobrých výsledků u kodéru x.264 bylo dosaženo při nastavení *-preset fast* nebo *ultrafast*. Při tomto nastavení kodéru průchod mixérem trvá přibližně 50% času a je již použitelné pro práci v reálném čase, zbývá dostatek času pro zpracování FFserverem.

Následně byl zprovozněn kompletní řetězec, tedy mixér a FFserver pro dva rozdílné streamy. Při použití sekvencí s nízkým rozlišením se celkové zatížení pohybuje na hranici okolo 80-90%. Vše pracovalo plynule včetně přehrávání výsledku, obrázek 3.10. Při použití sekvence ve vysokém rozlišení je systém zatížen na 100%, obrázek 3.11, a průchod není plynulý, jsou zahazovány celé bloky a výstup je nepoužitelný. Pro tuto hardwarovou konfiguraci serveru je možné použít pouze video vzorky nízké kvality tak, aby zatížení bylo pod hranicí 100% a vše pracovalo plynule. Pro zpracování vzorků v reálném čase s rozlišením FullHD by bylo potřeba výrazně výkonnější konfigurace serveru.



Obr. 3.10: Vytížení systému na 80-90% po spuštění mixu/translátoru nízký bitový tok

Obr. 3.11: Přetížení systému po spuštění mixu/translátoru vysoký bitový tok

4 ZÁVĚR

Cílem bakalářské práce bylo prostudovat funkci video standardu H.264 a VP8 a za pomoci vhodné aplikace navrhnout video mixér a translátor pro podporu videokonferencí. Mixér a translátor za pomocí aplikace FFmpeg by mohl sloužit jako součást MCU serveru, který by zajišťoval propojení mezi vícero druhy videokonferenčních zařízení a aplikací. Zejména pak pro videokonferenční aplikaci v webovém prohlížeči WebRTC, která využívá a rozšiřuje HTML5 o vlastní API a používá nekomerčního projektu Webm od společnosti Google Inc. Tato videokonferenční aplikace využívá standardy projektu Webm, které nemají podporu všech internetových prohlížečů, jako například Internet Explorer, který podporuje pouze standard H.264 a je jeden z nejrozšířenějších. Stejně dobře by serverová aplikace mohla být využita i pro jiné projekty.

Z dostupných zdrojů, jako je například IEEE Explore Digital Library, jsou v základních bodech popsány standardy H.264 a VP8 a porovnány jejich technické rozdíly. Výsledkem porovnání je, že kvalita těchto standardů je velmi podobná, liší se v některých technický detailech s tím, že blokové schéma kodéru je stejné. Některé drobné slabší stránky standardu VP8, například nepoužití snímků B, velmi dobře kompenzuje jeho svobodná licence.

V rámci práce jsem testoval dva velmi významné open source projekty pro zpracování videa a zvuku a to GStreamer a FFmpeg. Pro realizaci zadání byl vybrán FFmpeg. V průběhu realizace jsem se potýkal s problémy při zprovoznění, kdy mnou používaný hardware se prokázal jako nedostatečný, zejména při provádění škálování jednotlivých snímků a skládání do jednoho výstupního souboru. To se ukázalo jako hardwarově velmi náročné a při špatném nastavení trvalo déle, než samotné video. Tím by byl video mix pro práci v reálném čase nepoužitelný. Tento problém se podařil vyřešit výrazným snížením datového toku video streamu a použitím testovacích video sekvencí jen s nízkým bitovým tokem.

Zadání práce bylo splněno a aplikace je plně funkční, umožňuje realizovat video mix a překlad do rozdílných video standardů v reálném čase. Všechny součásti aplikace jsou po instalaci na server schopné pracovat bez dalšího potřebného přístupu k internetu. Server lze spravovat pomocí parametrů psaných do terminálu nebo pomocí vytvořené webové aplikace, která umožňuje základní nastavení parametrů výsledného výstupu.

LITERATURA

[1] HANUS, Stanislav. *Základy televizní techniky III: přednášky.* Vyd. 1. V Brně: Vysoké učení technické, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2010, 103 s. ISBN 9788021442061.

[2] VÍT, Vladimír. *Televizní technika: Přenosové barevné soustavy.* 1. vyd. Praha: BEN - technická literatura, 1997, 719 s. ISBN 808605604x.

[3] ŘÍČNÝ, Václav a Tomáš KRATOCHVÍL. *Základy televizní techniky: přednášky.* 1. vyd. Brno: VUT, 2004, 160 s. ISBN 8021426861.

[4] Feller, C. and Wuenschmann, J. and Roll, T. and Rothermel, A. *The VP8 video codec - overview and comparison to H.264/AVC.* Consumer Electronics - Berlin (ICCE-Berlin), 2011 IEEE International Conference on, 2011, 57-61 s. Dostupné z URL: `http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6031852`.

[5] Bankoski, J. and Wilkins, P. and Yaowu Xu. *Technical overview of VP8, an open source video codec for the web.* Multimedia and Expo (ICME), 2011 IEEE International Conference on, 2011, 1-6 s. ISSN 1945-7871. Dostupné z URL: `http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6012227`.

[6] RFC 6386. *VP8 Data Format and Decoding Guide* [online]. 2013 [cit. 23-04-2015].
Dostupné z:`http://datatracker.ietf.org/doc/rfc6386/?include_text=1`

[7] Kalva, H. *The H.264 Video Coding Standard.* MultiMedia, IEEE, 2006, 86-90 s. ISSN 1070-986X. Dostupné z URL: `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1709847&isnumber=36072`.

[8] OpenH264. *OpenH264* [online]. 2015 [cit. 20-04-2015].
Dostupné z: `http://www.openh264.org/`

[9] Projekt Libav. *Libav* [online]. 2015 [cit. 15-03-2015].
Dostupné z: `https://libav.org/`

[10] FFmpeg. *FFmpeg* [online]. 2015 [cit. 15-03-2015].
Dostupné z:`https://www.ffmpeg.org/`

[11] GStreamer. *GStreamer* [online]. 2015 [cit. 15-03-2015].
Dostupné z:`http://gstreamer.freedesktop.org`

[12] Webm. *Webm* [online]. 2015 [cit. 12-04-2015].
Dostupné z:`http://www.webmproject.org`

[13] Mp4. *Mp4ra* [online]. 2015 [cit. 10-05-2015].
Dostupné z:`http://www.mp4ra.org`

[14] Matroska. *Matroska* [online]. 2015 [cit. 10-05-2015]
Dostupné z:`http://www.matroska.org`

[15] Projekt FFmpeg. *FFmpeg* [online]. 2015 [cit. 25-04-2015].
Dostupné z: `http://www.ffmpeg.org/ffmpeg-all.html#overlay-1`

[16] VLC media player. *VideoLAN Organization* [online]. 2015 [cit. 25-04-2015].
Dostupné z: `http://www.videolan.org/vlc`

[17] The xiph open source community. *The xiph open source community* [online]. 2013 [cit. 25-04-2015].
Dostupné z: `https://media.xiph.org`

[18] The GNU nano. *Text Editor GNU Nano* [online]. 2009 [cit. 25-04-2015].
Dostupné z: `http://www.nano-editor.org`

[19] Opensshserver. *Instalace SSH serveru* [online]. 2012 [cit. 25-04-2015].
Dostupné z: `http://wiki.ubuntu.cz/ssh`

[20] Apache s MySQL a PHP. *Instalace LAMP* [online]. 2013 [cit. 20-04-2015].
Dostupné z: `http://wiki.ubuntu.cz/servery/apache_s_mysql_a_php`

[21] FFserver. *FFserver* [online]. 20135 [cit. 20-04-2015].
Dostupné z: `https://www.ffmpeg.org/ffserver.html`

[22] VLC Web Plugin. *VideoLAN Organization* [online]. 2015 [cit. 20-04-2015].
Dostupné z: `https://wiki.videolan.org/Documentation:WebPlugin/`

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

ITU	*International Telecommunication Union*
ITU-T	*International Telecommunication Standardization sector*
MCP	*Motion compensation prediction*
SIP	*Session Initiation Protocol*
VLC	*Variable Length Coding*
MPEG	*Moving Pricture Experts Group*
ISO	*International Organiation for Standardization*
AVC	*Advanced Video Coding*
RTP	*Real Time Protocol*
DCT	*Discrete Cosine Transform*
DWT	*Discrete Wavelat Transform*
CR	*Compres Ratio*
BSD	*Berkeley Software Distribution*
IPTV	*Internet Protocol Television*
HTML5	*Hyper Text Markup Language 5*
HTML	*Hyper Text Markup Language*
HDTV	*High-definition Television*
XML	*Extesible Markup Language*
USA	*United States of America*
GUI	*Graphical User Interface*
VLC*	*VideoLAN Client*
LGLP	*Lesser General Public License*
BP	*Baseline Profile*
MP	*Main Profile*

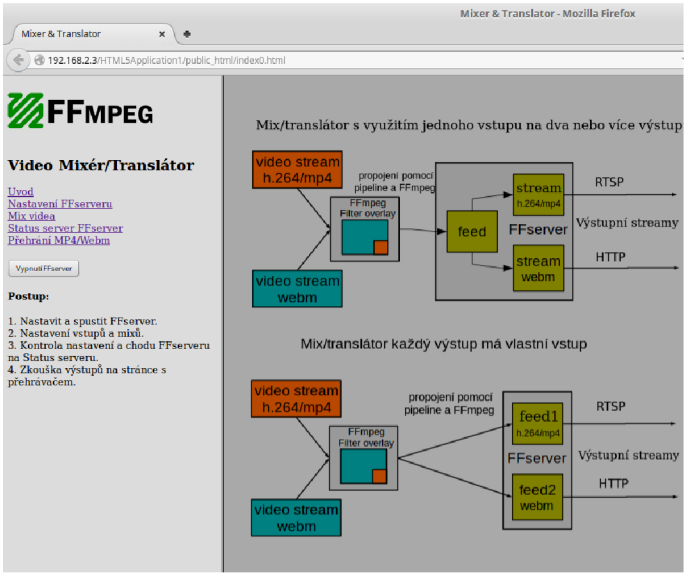
HP	*High Profile*
SDL	*Simple DirectMedia Layer*
HTTP	*Hypertext Transfer Protocol*
FTP	*File Transfer Protocol*
PHP	*Hypertext Preprocessor*
GF	*Golden reference frame*
AF	*Alternate reference frame*
LF	*Last Frame*
RTSP	*Real Time Sreaming Protocol*
WHT	*Walsh-Hadamardova Transform*
GOP	*Group of Pictures*
WebRTC	*Web Real-Time Communication*
DC	*Direct Current*
SSH	*Secure Shell*
PNG	*Portable Network Graphics*

A NÁVOD OVLÁDÁNÍ WEBOVÉ APLIKACE

A.1 Úvodní zobrazení a možnosti propojení

Na úvodní stránce jsou uvedeny dvě základní možnosti, jak provést propojení mixéru s steamovacím serverem. První možností je vytvořit jeden vstup (feed) a ten propojit s jednotlivými výstupy (streamy), kdy každý stream může vytvářet výstup v jiném kodéru a s jiným rozlišením.

Druhá možnost je vytvořit dva a více vstupů, kdy lze na každý vstup (feed) přivést jiný zdroj. Tato aplikace umožňuje vytvoření pouze jednoho mixu, druhý vstupní stream musí být nadefinován ručně pomocí parametrů a terminálu, obrázek A.1.



Obr. A.1: Úvodní stránka aplikace

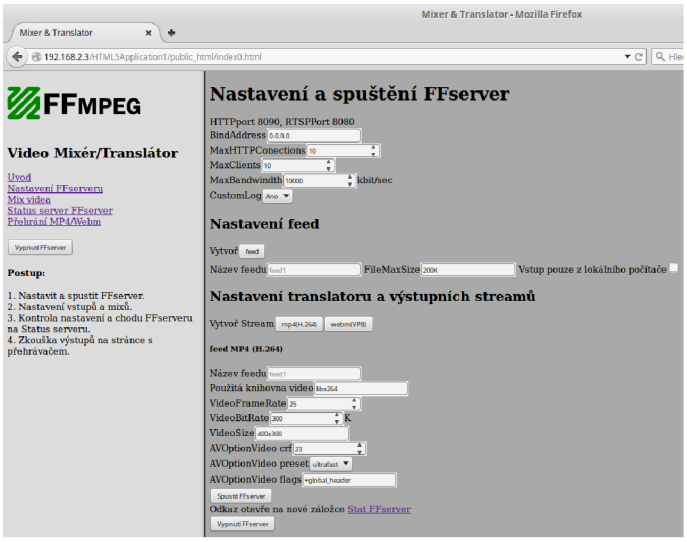
A.2 Nastavení vstupů a výstupů serveru

Do nastavení je možné se dostat přes odkaz „Nastavení FFserveru“ v menu. Úvodní volby umožňují nastavit počet připojených klientů a šířku pásma, pokud hodnoty nebudou měněny, budou použity přednastavené a zobrazené.

Nastavení feed, pomocí tlačítka feed jsou generovány vstupy, každému vstupu (feed) je nutné přiřadit název. Nastavení FileMaxsize udává velikost dočasné paměti Temp, není nutné měnit.

Nastavení translátoru generuje výstupní streamy, každému výstupnímu streamu je nutné přiřadit některý nadefinovaný feed. Ostatní parametry udávají nastavení kodéru.

Po provedení nastavení je nutné spustit server, provede se tlačítkem „Spustit ffserver“. Ověření jeho chodu a parametrů je možné odkazem nebo tlačítkem „Status FFserver“. Případné vypnutí tlačítkem „Vypnutí FFserver“, obrázek A.2.



Obr. A.2: Nastavení výsledného mixu videa

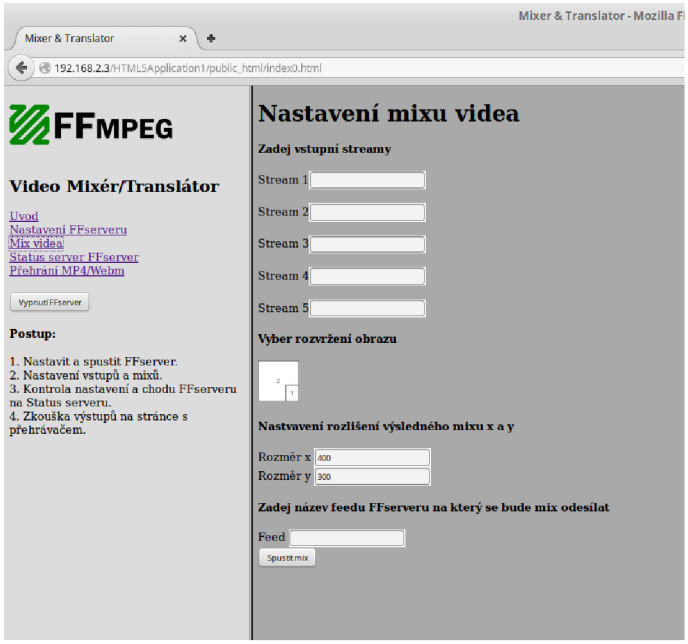
A.3 Nastavení mixu

Do nastavení je možné se dostat před odkaz „Mix videa“ v menu. Do pole Stream 1 - 5 je možné nadefinovat vstup do video mixeru, je možné zadat cestu k souboru na disku nebo adresu streamu. Rozměr definuje velikost mixovaného videa, vhodné je nastavit na stejnou hodnotu jako u předchozího nastavení (viz Nastavení výstupů serveru) nebo na hodnotu největší z nastavovaných, jinak by docházelo ke ztrátě kvality ještě před vstupem do streamovacího serveru.
Pole feed určuje, do kterého vstupu (feed) serveru bude mix propojen, obrázek A.3.

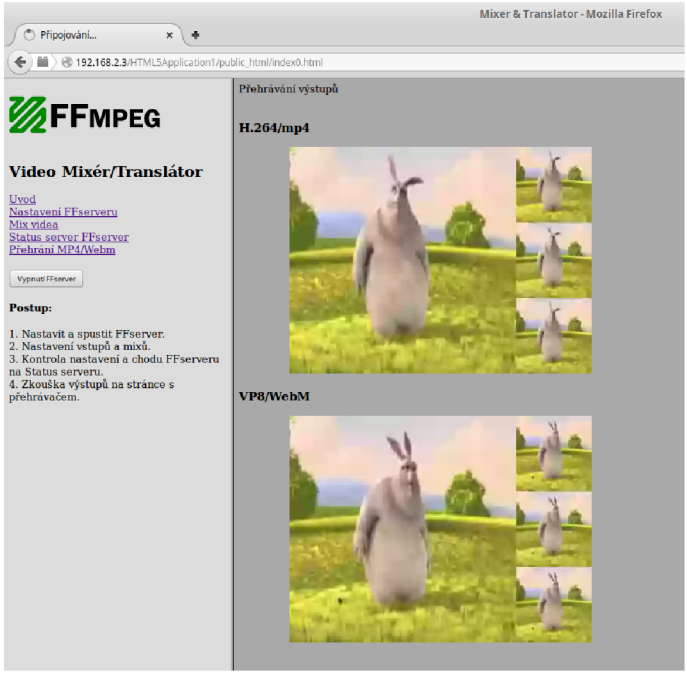
A.4 Přehrávání video

Na zobrazení je možné se dostat před odkaz „Přehrávání MP4/Webm“ v menu. Přehrává jeden výstup z výstupu MP4 a druhý z Webm, obrázek A.4.

Chod serveru se ukončí tlačítkem „Vypnutí FFserver“.



Obr. A.3: Nastavení výsledného mixu videa



Obr. A.4: Zobrazení výstupu FFserveru

B OBSAH PŘILOŽENÉHO CD

B.1 Aplikace mixer_translator

B.2 Video sekvence Big_Buck_Bunny