BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2017
http://excel.fit.vutbr.cz

GUI ke stavbě grafu filtrů pro FFmpeg

Roman Sichkaruk*

Abstrakt
Tento článek se zabývá tvorbou grafického uživatelského rozhraní pro tvorbu grafů filtrů FFmpeg. Tento nástroj pro framework FFmpeg chybí. K řešení je použit multiplatformní framework Qt. Výsledkem je ucelený nástroj umožňující uživateli tvorbu, editaci, ukládání a mazání grafů filtrů s možností zobrazení náhledu jejich vlivů. Zaměřuje se přímo na filtry videa. Aplikace je vhodná ke zjednodušení a urychlení seznámení a práce s grafy filtrů ffmpeg.

Klíčová slova: FFmpeg — Graf filtrů — avfilter

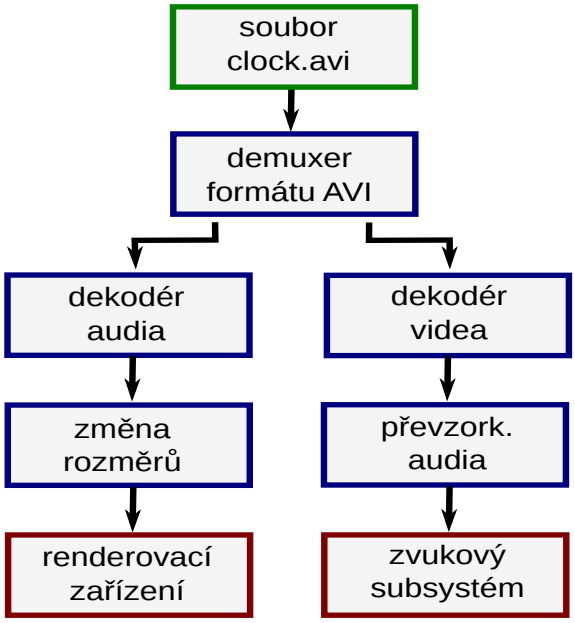
Přiložené materiály: FFmpeg dokumentace

*xsichk00@fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Úvod

V dnešní době se s pojmem filtr setkáváme často. Ať už to jsou obrazové filtry pro úpravu fotografií, nebo filtry měnící zvukové signály (aplikace pro změnu hlasu), nebo filtry videa, poskytující napří-
klad zlepšení kvality, odstranění šumu, zrychlení, zpomalení atd. Filtry lze seskupovat – vytvářet sekvence filtrů. Takovým sekvencím se říká grafy filtrů. Příklad jednoduchého grafu filtrů je na obrázku 1. Tento graf začíná vstupním zdrojem videa. Komponenta demuxer rozdělí vstupní datový tok na dva výstupní – audio a video tok. Příslušný dekodér se postará o čtení toku. Ve videu je změněn rozměr a zvukový signál je převzorkován. S grafy filtrů pracují multimediální frameworky – soubory knihoven a nástrojů pro práci s multimediálními daty. Multimediální frameworky grafy filtrů buď umějí použít nebo jsou na nich přímo založené.

Mezi významné multimediální frameworky patří např. Video for Windows, QuickTime, DirectShow, Media Foundation, GStreamer a FFmpeg.



Obrázek 1. Příklad grafu filtrů.

Většina těchto frameworků nabízí intuitivní grafické uživatelské rozhraní. Ovšem u frameworku FFmpeg takový nástroj chybí. V následující části článku vymezuji základní pojmy, potřebné k pochopení problematiky. Taktéž zde popisuji významné multimediální frameworky a příslušné nástroje s grafickým uživatelským rozhraním. Ve třetí sekci vysvětluji vlastní řešení a porovnávám jej s existujícími. Na závěr uvádím krátké shrnutí a možné směry dalšího vývoje.

2. Teoretický základ

Základem a nezbytnou součástí jakéhokoliv multimediálního frameworku je filtr. Filtr se v pojetí multimédií jeví, jako modul zaobalující nějakou funkcionalitu. Většinou lze filtry rozdělit do třech základních kategorií:

- **Zdrojové** – jsou vstupním bodem toku dat, např. multimediální kontejner do kterého je načten soubor uložený lokálně na disku, nebo stažený z internetu. Zdrojovými filtry můžou být zachytávací nebo nahrávací zařízení.
- **Transformační** – provádí změny v těchto datech. Mezi transformační filtry se řadí kodéry, dekodéry, muxery, demuxexy, filtry pro práci s obrazy, zvukovými signály, pro změnu formátu dat a mnoho dalších.
- **Výstupní** – poskytují rozhraní pro práci s výstupními toky dat (uložení do souboru, tisk, zobrazování na renderovacím zařízení).

Při průchodu grafem filtrů se může stát, že se vlivem transformací změní formát dat. V případě, že se formáty nebo typy dat na spoji dvou filtrů liší, filtry se jednoduše nespojí. Některé frameworky (např. FFmpeg) umožňují filtrům se domluvit na typu dat, umí tedy provádět omezené automatické typové konverze. Starší frameworky (Video for Windows) tuto vlastnost nemají. Pokud si ale typy odpovídají, vzniká spojení a data tečou dále. Místa spojení dvou filtrů označujeme jako *pady* nebo také *piny* v závislosti na frameworku. Pady mohou být vstupní, nebo výstupní. Počet padů je definován typem filtru. Některé mají pouze jeden výstupní (typicky zdrojové filtry), jiné pouze jeden vstupní (výstupní filtry). Transformační filtry mají jeden a více vstupních a výstupních padů.

Některé frameworky mají fixní graf filtrů (Video for Windows), čímž se přesně definuje i způsob práce frameworku. Ovšem valná většina umožňuje vytvářet grafy filtrů přesně podle potřeby uživatele. Ke zjednodušení seznámení a práce s frameworky existují nástroje zaobalující funkcionalitu frameworku do grafického uživatelského rozhraní. Příklady těchto nástrojů jsou uvedeny u každého frameworku v následujících částech sekce.

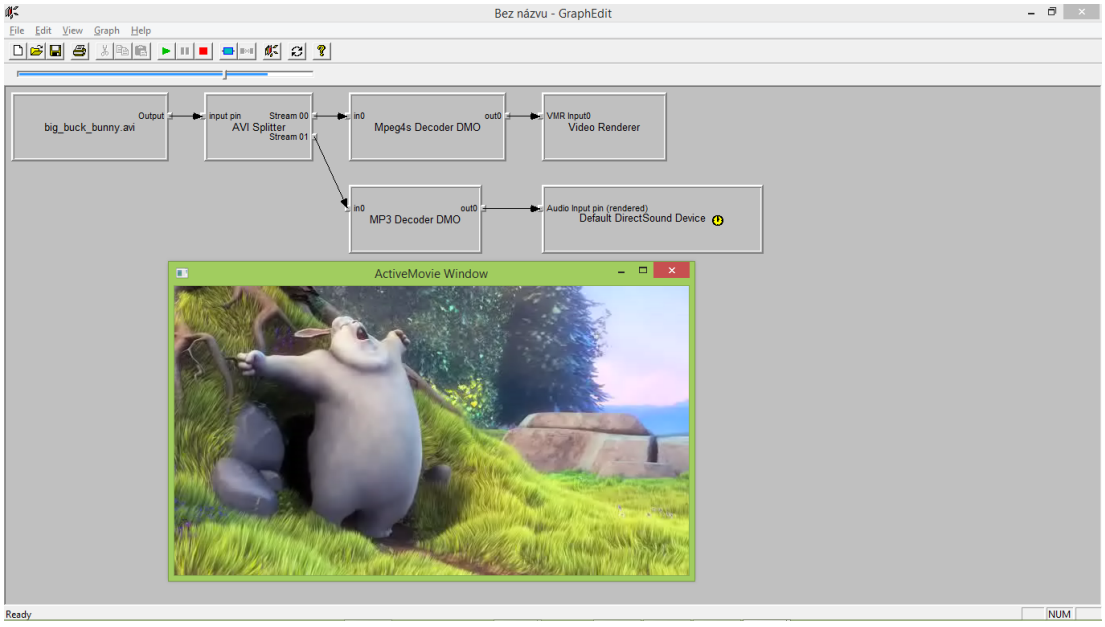
2.1 DirectShow

DirectShow[1] je multimediální framework vyvinutý firmou Microsoft, jako nástupce Video for Windows. Implementace je v jazyce C++ a je založena na technologii COM (*Component Object Model*). Framework umožňuje vývojářům přistoupit k jeho základním prvkům, čímž zjednodušuje rozšíření funkčnosti, tvorbu přidavných modulů. Další innovací oproti Video for Windows je automatická konverze barevných modelů.

DirectShow umožňuje vytvářet téměř libovolné grafy. Tyto grafy se skládají z filtrů třech základních typů (zdrojové, transformační a výstupní) a jsou vzájemně propojeny pomocí pinů. Je důležité, aby se propojené filtry dohodly na typu dat. V opačném případě dojde k chybě programu. Pro jednodušší tvorbu grafů filtrů framework obsahuje editační nástroj *GraphEdit* (viz. obrázek 2), který poskytuje grafické rozhraní pro uživatele. Znázorňuje filtry, jako krabičky s textovým popiskem, vymezujícím jméno filtru. Filtry jsou ke zvolení po rozkliknutí tlačítka *Insert filters* a dvojklikem si je můžete přidat na pracovní prostor. GraphEdit umožňuje přehrávání náhledu grafů filtrů, které lze kdykoliv pozastavit.

2.2 Media Foundation

Media Foundation[1] je multimediálním frameworkem, který nahradil zastaralý DirectShow. V základu je velice podobný DirectShow. Jádrem je model COM a grafy filtrů. Graf filtrů je pojmenován topologie (*topology*) a filtr – uzel (*node*). Definovány jsou 4 typy uzlů



Obrázek 2. Grafické rozhraní nástroje GraphEdit.

– zdrojové, transformační, výstupní a nový typ filtrů *tee*. Tee uzly rozdělují tok dat více směry. Uzly jsou vzájemně propojené. Pojmem *upstream* se označuje uzel poskytující data a *downstream* je uzel přijímající data. Stejně, jako DirectShow, existuje editační nástroj – *TopoEdit* (viz. obrázek 3). K vkládání uzlů slouží kontextové menu a dvojklik myší. Nástroj je velice jednoduchý a slouží k vytváření topologií a kontrole, zda jsou funkční. K této kontrole slouží možnost náhledu.

2.3 GStreamer

GStreamer[2] je open-source multiplatformní multimediální framework. Framework je založený na grafech filtrů označovaných pipeliny (*pipelines*). Jednotlivým filtrům se říká elementy (*elements*). Tyto elementy jsou propojeny pomocí bodů, kterým se říká pady.

Implementaci a načítání elementů obstarává příslušný zásuvný modul. Zásuvnými moduly pro tento framework jsou dynamické knihovny. Jejich načítání probíhá za běhu. Framework je schopen propojovat elementy automaticky na základě dat protékajících těmito elementy. Pro zjednodušení vytváření pipeline je k dispozici nástroj *gst-editor* (viz. obrázek 4). Tento nástroj umožňuje zvolit filtr, dvojklikem jej přidat na pracovní plochu, editovat jeho vlastnosti a zobrazit informace o padech. Nechybí zde ani přehrávač aktuální pipeliny.

2.4 FFmpeg

FFmpeg je multimediální framework umožňující dekódování, kódování, transkódování, multiplexování, demultiplexování, streamování, filtrování a přehrávaní multimédií.

FFmpeg nabízí velké množství filtrů jak pro video, tak i pro audio stopy. Typy filtrů se příliš neliší od obecného konceptu. Grafy filtrů FFmpeg mohou být dvou typů:

- **jednoduché** – mají jeden vstup a jeden výstup,
- **komplexní** – mohou mít více vstupů a výstupů.

Pro vizualizaci grafů filtrů existuje konzolový nástroj *graph2dot*. Tento nástroj převádí řetězec s grafem filtrů na jeho obrázkovou reprezentaci. Plnohodnotný nástroj s grafickým uživatelským rozhraním pro sestavení grafů filtrů ovšem chybí a jeho implementace je řešena v následující sekci.

3. Vlastní řešení

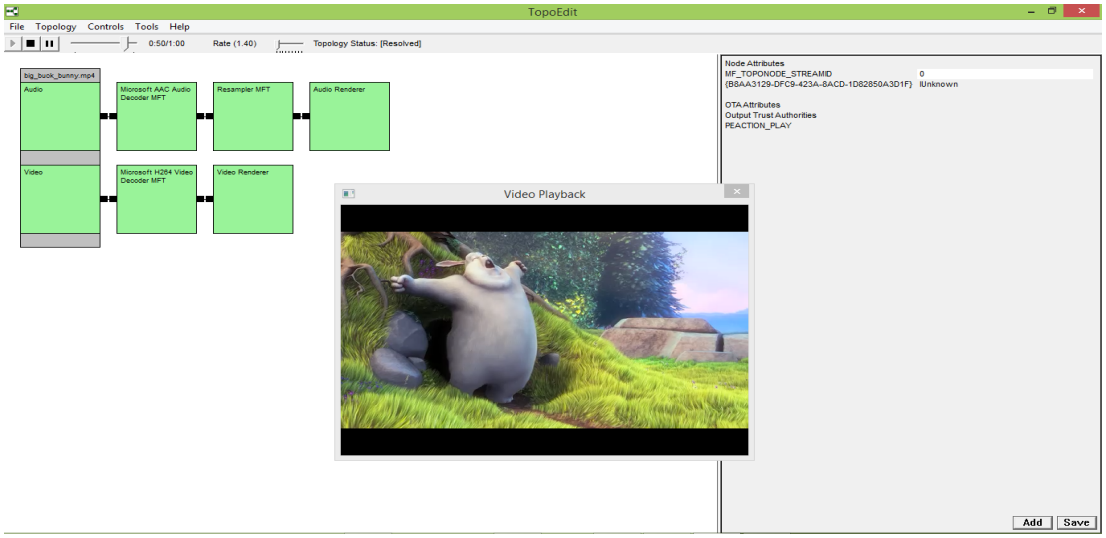
Důležitou vlastností FFmpeg je jeho multiplatformnost. Knihovny FFmpeg jsou implementovány v jazyce C. Proto k implementací nástroje byl zvolen multiplatformní framework Qt umožňující snadnou integrací knihoven napsaných v jazyce C. Ostatní implementační detaily jsou popsány v následujících sekcích.

3.1 Namapování FFmpeg struktur

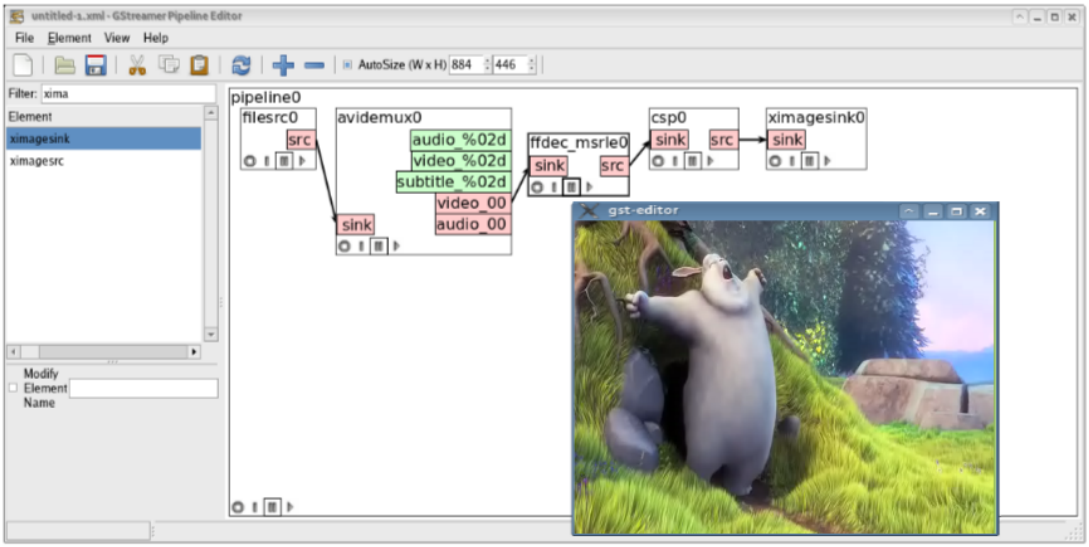
Mapování vnitřních struktur FFmpeg na renderovací Qt objekty je znázorněno v tabulce 1.

Tabulka 1. Namapování FFmpeg struktur

FFmpeg struktura	Vlastní třída	Význam
AVFilterPad	Pad	Bod propojení
AVFilter	Filter	Filtr
AVFilterLink	Wire	Spojení filtrů
AVOption	FilterParameter	Parameter filtru



Obrázek 3. Grafické rozhraní nástroje TopoEdit.



Obrázek 4. Grafické rozhrani nástroje gst-editor.

Ze všech struktur FFmpeg je nutno vybrat pouze potřebné vlastnosti. Třída *Pad* v sobě uchovává unikátní jméno padu. Třída *Filter* obsahuje odkazy na vstupní a výstupní pady, jméno filtru a seznam parametrů. Tyto parametry jsou objekty třídy *FilterParameter* a uchovávají informaci o typu vstupních dat (číslo s plovoucí řádovou čárkou, textový řetězec, atd.), jméně parametrů, rozsashu hodnot a aktuálně nastavené hodnotě. Nakonec třída *Wire* v sobě zaobaluje odkazy na vstupní a výstupní pady, které jsou propojené. Každá z těchto tříd dědí renderovací Qt widget (Wire, Filter, Pad), nebo je přímo modifikována pomocí modelového pohledu (FilterParameter).

3.2 Implementace pracovní plochy

Pro potřeby pracovní plochy byla vytvořená třída *Scene*, která dědí všechnou potřebnou funkcionalitu od třídy *QGraphicsScene*. Třída Scene implementuje zoomování, posun jak horizontálně, tak i vertikálně. Další důležitou částí implementace je možnost použití *Drag&Drop* mechanismu, který chyběl u předchozích existujících řešení pro jiné frameworky. Tento mechanismus umožňuje jednoduše přetáhnout filtr na zvolené místo v rámci pracovní plochy. Plocha si udržuje seznam umístěných filtrů a dovoluje je libovolně přemisťovat a odstraňovat.

3.3 Implementace seznamu filtrů

Při vytváření seznamu platných filtrů se musi načíst pouze aktuálně dostupné filtry v závislosti na nainstalované verzi FFmpeg. Tento seznam je upraven tak, aby obsahoval pouze filtry zpracovávající video stopy. O každém filtru je zjištěn počet vstupních a výstupních padů a seznam parametrů. Ke každému parametru a filtru se rovněž uloží kratký popisek, který je uživateli přístupný při kliknutí tlačítka nápovědy. Samotný filtr v seznamu tvoří třída potomek tlačítka *QPushButton*. Navíc je zde implementována možnost

začátku přetažení na pracovní plochu. Instance třídy Filtr jsou uloženy do objektu třídy *QTableWidget* a ten je vložen do objektu dědící třídu *QDockWidget*. Kromě seznamu filtrů obsahuje dokovací widget textové políčko, které po zadání textu skryje všechny filtry neobsahující zadaný řetězec.

3.4 Implementace seznamu parametrů

Dalším potomkem třídy *QDockWidget* je seznam, znovu tvořený objektem *QTableWidget*, obsahujícím parametry filtrů. Každý filtr v sobě nese hashovací tabulku se jmény a hodnoty parametrů. Při výběru filtrů na pracovní ploše, se automaticky inicializuje tabulka s parametry podle hashovací tabulky tohoto filtru.

3.5 Propojení jednotlivých filtrů

O propojení jednotlivých filtrů se stará třída *Connectivity*. Tato třída v sobě udržuje seznam objektů třídy *Wire*. Umožňuje je vytvářet a kontroluje logickou správnost propojení. Také provádí aktualizaci míst spojů v případě posunu filtrů na pracovní ploše. Třída *Wire* v sobě zaobaluje odkazy na propojené pady a úsečku spojující pady vizuálně na ploše.

3.6 Exportování/importování grafů filtrů

Existji dvě možnosti ukládání a načítání grafu filtrů. První je možnost uložení textového řetězce ve formátu nástroje *ffmpeg*. Graf filtrů je tvořen jednotlivými řetězci filtrů oddělenymi středníkem. Filtry v řetězci filtrů jsou odděleny čárkou. Znak "=" za názvem filtru definuje začátek seznamu parametrů filtrů. Pokud tento znak chybí, je filtr inicializován s přednastavenými parametry. Parametry jsou ve tvaru *klič=hodnota*, kde *klič* je jméno parametru a *hodnota* je jeho hodnota. Filtry v řetězci filtrů jsou propojeny automaticky podle pořadi. Řetězce filtrů se propojují pomocí spojení ve tvaru *[pad]*, kde *pad* je unikátní jméno spojení. Pokud řetězec filtrů obsahuje toto propojení před ukončovácím středníkem, napojí se na řetězec, který obsahuje toto propojení na začátku. Druhou možností je uložení do formátu *XML*. V tomto případě jsou filtry serializovány s parametry typu – jméno, pozice na pracovní ploše, jména padů, seznam parametrů filtru ve tvaru klíč-hodnota. Dále jsou ukládána jednotlivá propojení s informací o jménech propojených padů. Při načítání se filtry umístí na pozice při uložení.

3.7 Ukládání videa, náhled grafu filtrů

Ukládání a přehrávání videa je implementováno za použití FFmpeg knihoven. Nejdříve je načten kontext vstupního formátu na základě jména souboru. Dále je inicializován graf filtrů. Následuje čtení proudu dat. Data se dekódují a získané snímky jsou předané grafu filtrů, který snímek zpracuje a vrátí zpracovaný snímek. V případě uložení se snímek zakóduje příslušným kódérem a zapíše do výstupního souboru. V případě zobrazení náhledu se převádí formát snímku z YUV420 na RGB888 a jednotlivé bity se uloží do obrázku typu *QImage*. Po uplynutí doby mezi snímky, počítané jako 1/snímková frekvence za sekundu, se zobrazí tento obrázek na přehrávací ploše. Rozdíl v rychlosti mezi přehrávacím nástrojem ffmpeg a náhledem implementovaným v této práci je zcela zanedbatelný.

3.8 Porovnání s existujícími řešeními

Porovnání s existujícími řešeními lze nejlépe vyjádřit tabulkou 2.

4. Závěr

Cílem této práce byla tvorba grafického uživatelského rozhraní pro vytváření a editaci grafů filtrů FFmpeg.

Výsledný vzhled je na obr. 5. V levé části je seznam filtrů, vpravo seznam parametrů aktuálně vybraného filtru crop. Na pracovní ploše je graf filtrů, jehož výsledek je zobrazen v náhledu pod ním. Filtr codecview je přetahován ze seznamu filtrů na pracovní plochu.Vlastnosti aplikace byly ze značné míry převzaté z existujících řešení pro jiné frameworky. Některé vlastnosti byly implementovány nově (např. Drag&Drop), nebo rozšířeně (např. možnosti práce s plátnem). Celkově je nástroj vhodný ke zjednodušení práce s FFmpeg, ke zrychlení vyladění grafů filtrů a umožňuje vyhnout se zbytečným chybám při stavbě grafu na příkazové řádce. Při vytvoření větších grafů filtrů umožňuje práci kdykoliv uložit a načíst. Zabudována nápověda zrychlí seznámení s filtry FFmpeg pro nováčky.

Mezi možné směry dalšího vývoje bych zařadil možnost zpracování více vstupů a výstupů, práci s audio tokem dat.

5. Poděkování

Chtěl bych poděkovat vedoucímu mé práce Ing. Davidovi Bařinovi za odborné vedení a cenné rady.

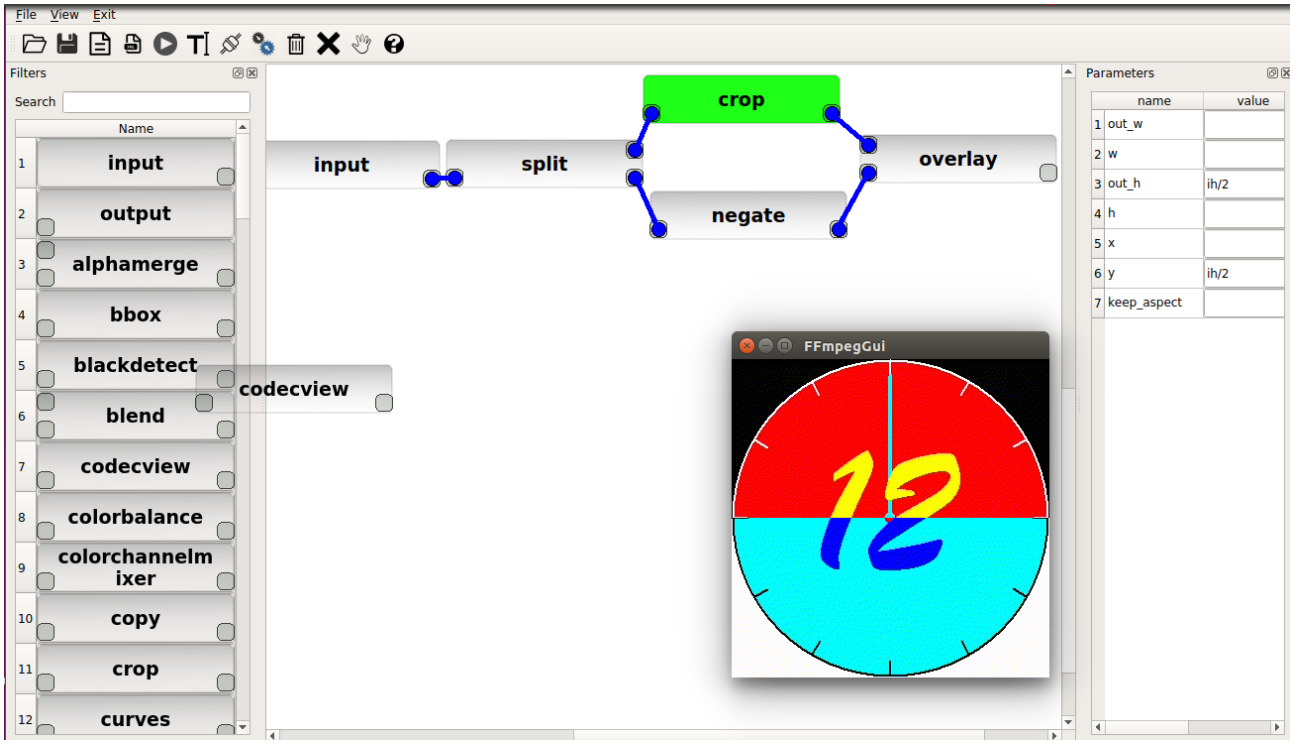
Literatura

[1] Microsoft. *Microsoft MSDN*. https://msdn.microsoft.com. [Online; navštíveno 15.03.2017].

[2] GStreamer. *GStreamer documentation*. https://gstreamer.freedesktop.org/documentation/. [Online; navštíveno 15.03.2017].

Tabulka 2. Porovnání s existujícími řešeními

	Vlastní řešení	Gst-editor	Topoedit	GraphEdit
Vložení filtru	drag&drop	dvojklik	dvojklik	dvojklik
Spojení filtru	zmáčknutí tlačítka	tažení myší	tažení myší	tažení myší
Seznam filtrů	v hlavním okně	v dialogovém okně	v dialogovém okně	v hlavním okně
Editor vlastnosti filtrů	v hlavním okně	v dialogovém okně	v hlavním okně	v dialogovém okně
Možnost nastavení umístění panelů	ano	ne	ne	ne
Podpora náhledu	ano	ano	ano	ano
Dynamická inspekce vlastnosti grafu	ano	ano	ne	ne
Uložení/načtení grafu	ano	ano	ne	ano
Nápověda pro každý filtr a parameter	ano	ne	ne	ne
Vytvoření textové reprezentace grafu	ano	ano	ne	ne



Obrázek 5. Ukázka vytvořené aplikace.