VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

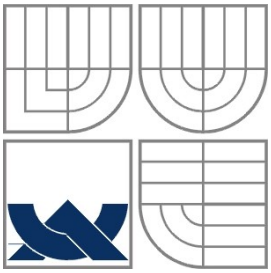
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EDITOR VIDEA

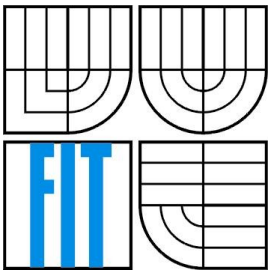
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE VÍT KOPŘIVA
AUTHOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EDITOR VIDEA
VIDEO EDITOR

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
VÍT KOPŘIVA

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. ROMAN JURÁNEK

BRNO 2009

Abstrakt

Obsahem této práce je návrh a implementace jednoduchého systému pro zpracování a střih videa. Práce rozebírá základní funkce programu: načtení video sekvence, střih, filtrace, vložení na časovou osu a vytvoření výstupní video sekvence. Jedná se o konzolovou aplikaci, která je určena zejména pro hromadné zpracování souborů do jedné výsledné video sekvence. Teoretická část práce se zabývá technologickým vývojem a současnými softwarovými prostředky zpracování video sekvencí.

Abstract

This work deals with an implementation of a simple software system for video editing. The practical part of this work examines the essential functions of the software: video sequence import, sequence cutting, filtering, timeline placement and video sequence export. This software is the command line interface application, primarily designed for batch editing of video files into the one single video sequence. Theoretical part of this thesis describes technological development of a video editing as well as the current software means.

Klíčová slova

video, střih videa, video sekvence, film, nelineární střih, lineární střih, zpracování videa, filtrace videa

Keywords

video, video editing, video sequence, film, nonlinear editing, linear editing, video processing, video filtering

Citace

Vít Kopřiva: Editor videa, bakalářská práce, Brno, FIT VUT v Brně, 2009

Editor videa

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Romana Juránka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Vít Kopřiva
15.5.2009

Poděkování

Tímto bych chtěl poděkovat Ing. Romanu Juránkovi za odbornou pomoc při tvorbě praktické i teoretické části mé práce.

Obsah

1 Úvod

Digitální video se stalo každodenní součástí našeho života. Setkáváme se s ním v podobě televizního vysílání, filmové projekce v kinech, na obrazovkách počítačů nebo na displejích mobilních telefonů. Nemusíme být ani profesionálními herci, abychom každý den stáli před objektivem kamery. Aniž si to uvědomujeme, stáváme se objektem natáčení dopravních, bezpečnostních či webových kamer na každém kroku. Vzniká tak velké množství záznamů z různých typů zařízení, které je třeba analyzovat, zpracovat a archivovat. Ke zpracování tohoto materiálu slouží video editory. Zpracování videa bylo donedávna výsadou pouze profesionálních studií. S prudkým poklesem cen video techniky a rozšířením počítačů do domácností se tato situace změnila. Dnes střihové programy umožňují proniknout široké veřejnosti do tvorby videa, zpracování obrazu i zvuku. To vše je možné zejména díky rychle se rozvíjejícím technologiím digitálního světa. Fenoménem poslední doby se stalo sdílení video klipů prostřednictvím internetu, což dále zvýšilo poptávku po těchto programech.

Cílem mé práce je vytvořit jádro takového video editoru. Program by měl umět video sekvence načíst, sestříhat, filtrovat, vložit na časovou osu, vkládat mezi ně jednoduché přechodové efekty a nakonec vytvořit výsledný video záznam v požadovaném formátu. Při práci budu vycházet ze základních znalostí profesionálního i neprofesionálního softwaru pro zpracování video záznamů. Ve stručném teoretickém úvodu práce hodlám projekt zařadit do kontextu technologického vývoje zpracování videa od počátků kinematografie až po současné digitální střižny a specializovaný software.

Úvod teoretické části práce je věnován vývoji střihových technik. Nejdříve je popsána technika střihu filmového pásu. Technika lineárního střihu popisuje metody zpracování obrazového záznamu uloženého na magnetické pásce. Pokročilejší nelineární střih představuje současný způsob zpracování videa. Jemu věnovaná kapitola shrnuje, jaké změny digitální technologie přináší do oblasti zpracování obrazu a zvuku. Následující kapitola popisuje programy pro editaci videa a porovnává jejich základní vlastnosti. Vybranými programy jsou Adobe Premiere, Virtual Dub a Kdenlive. Na programu Adobe Premiere je demonstrován obecný způsob práce se střihovými programy.

Kapitola Videoeditor se věnuje jádru bakalářské práce – tím je návrh architektury a implementace programu pro nelineární střih videa. Tento program jednoduše nazývám Videoeditor. Kapitola Editace videa vysvětluje metody střihu u Videoeditoru po jednotlivých krocích. Další kapitola popisuje filtry a efekty, které lze na vytvořené video sekvence aplikovat. Princip práce programu je demonstrován na přehrávání video sekvencí v kapitole 7. Předposlední kapitola shrnuje veškerá omezení aplikace a navrhuje, jak by bylo možné tato omezení odstranit. V závěru jsou zhodnoceny dosažené výsledky a navrženy možné způsoby pokračování v této práci.

Součástí praktické části práce je také programová dokumentace, návod k použití programu včetně demonstračních příkladů, testovací skript a ukázky výstupních video sekvencí. Tyto části a zdrojový kód programu je obsahem přiloženého DVD.

2 Techniky střihu

Obor kinematografie má za sebou už více než 110 let vývoje. Bratři Auguste a Louis Lumiérovi roku 1895 přihlásili k patentování přístroj k zachycení a projekci chronostratografických snímků, který nazvali kinematograf. K záznamu použili celuloidového pásu o šířce 35mm s dvojicí kulatých otvorů po stranách. Koncem roku 1895 uspořádali bratři Lumiérové první veřejní představení, které dalo základ k rozvoji kinematografie.

Princip kinematografu je jednoduchý a používá se prakticky dodnes. Vychází z nedokonalosti lidského oka. Lidské oko má určitou setrvačnost. Stačí jednotlivé snímky promítat dostatečně rychle a oko nevnímá snímky jako statické obrazy, ale jako souvislý děj. Dostatečná rychlost pro lidské oko představuje nejméně 10 snímků za sekundu. Pro dokonalé splynutí jednotlivých snímků a zamezení kolísání jasu je však potřeba promítat s frekvencí 24 snímků za sekundu. Tato frekvence se stala filmovým standardem [1].

V průběhu v vývoje kinematografie se film vyvíjel jak po stránce technické tak také jako forma uměleckého vyjádření. Je přitom podstatné, že tyto dvě složky jsou neoddělitelné a vždy působily ve vzájemné interakci. Jedním ze základních vyjadřovacích prostředků kinematografie je střih, přičemž střihová technika se vyvíjela zejména v souvislosti s vývojem technických možností záznamových medií. Základní techniky střihu a jeho technické prostředky jsou popsány v této kapitole.

2.1 Střih filmového pásu

Filmová kamera je založena na stejném principu jako fotoaparát. Hlavní části kamery tvoří objektiv, závěrka a světlocitlivý film. Objektiv přijímá a kontroluje světlo dopadající na film, který světlo zaznamenává. Závěrka určuje, jak dlouho bude světlo na film dopadat. Srdce filmové kamery tvoří posuvný mechanismus, který posunuje film na místo pro expozici. Tento pohyb je plně synchronizován se závěrkou. Mechanismus posune okénko filmu na místo pro expozici, otevře závěrku, nechá okénko osvítit, závěrku uzavře a posune na místo expozice další okénko. Takto je zaznamenán jeden snímek filmu. Tento děj se opakuje 24 krát za vteřinu. Stejný mechanismus se používá pro posun jednotlivých snímků před objektiv projektoru [1].

Natočený materiál putuje do fotolaboratoře, kde je film vyvolán. Teprve potom je možné natočené záběry shlédnout a dále s nimi pracovat. Vyvolání je chemický proces, pomocí kterého je získán negativ filmu. Střih je pro filmový pás destruktivní, proto je potřeba nejdříve vytvořit pracovní kopie. Ty umožňují s úpravami filmu experimentovat a postupně film zpracovávat do finální podoby. Jeden z prvních přístrojů, který se používal pro zpracování filmu, byl stroj *Moviola* (obrázek 1) pocházející z roku 1924. Tento přístroj umožňoval film stříhat, lepit a sledovat. Začátkem šedesátých let se začaly používat ploché stříhací stoly, které umožňovaly srovnání mezi více obrazovými a zvukovými pásy. Jeden z nejvýznamnějších výrobců těchto stolů je firma Steenbeck, která je vyrábí dodnes. Jakmile byl film sestříhán z pracovních kopií, byla vytvořena střihová soupiska. Podle ní se provádí střihy v originálním materiálu. Originální záběry jsou tím však nevratně zničeny. Sestříhaný originál se nazývá *Master* [1] a z něj se vytváří distribuční kopie. Fyzickým kopírování filmu se však kvalita obrazu ztrácí.

Mezi nevýhody této technologie: nemožnost kontrolovat záznam okamžitě po natočení, zničení originálního natočeného materiálu, ztráta kvality kopírováním a omezené možnosti při tvorbě efektů.

Natáčení na filmový pás se používá dodnes a princip záznamu na něj se za 100 let prakticky nezměnil. Změnila se však metoda jeho zpracování. Film už se ve většině případů fyzicky nestříhá, ale je zpracováván digitálně. Natočený film se naskenuje velmi citlivým digitálním skenerem a dále se pracuje se snímky v digitální podobě. Výhodou natáčení na filmový pás je vysoká kvalita zachycení obrazu (velké rozsahy kontrastu). Filmový pás lze naskenovat s rozlišením až 8K (8000 pixelů na řádek.), kdežto digitální kamery běžně dosahují rozlišení 4K [21]. Skenery jsou však velmi drahé a natáčení na filmový pás je nákladnější než pořízení digitálního záznamu.

Obrázek 1: První model filmového editoru Moviola z roku 1924, Steenbeck ST 1601 (60. léta), Steenbeck Flatbed (současnost), převzato z [22] a [23]

2.2 Lineární střih videa

Technika lineárního střihu videa přichází se zavedením video kamer, které zaznamenávají obraz na magnetickou pásku. Ve video kameře zůstává zachována optická soustava objektivu i závěrka, mění se však způsob záznamu obrazu. Světlo procházející objektivem dopadá na světlocitlivý elektronický snímač (např. katodová trubice), který generuje analogový signál reprezentující snímaný obraz. Tento signál lze přenášet na obrazovky více monitorů a zaznamenávat ho na magnetickou pásku.

Magnetická páska je uskladněna ve formě kotoučů nebo normalizovaných kazet, jako je například VHS-C nebo Betacam SP [9]. Jednou z výhod magnetického pásu je, že ho lze pro záznam použít opakovaně. Nejvýznamnější výhodou je však to, že natočený záznam na pásce lze ihned přehrát bez nutnosti vyvolání, což výrazně urychluje i proces zpracovaní záznamu.

Střih záznamu na magnetickém pásu s šikmým záznamem neprobíhá fyzickým stříháním pásu jako u klasického filmu, nýbrž elektronickým kopírováním záběrů z jednoho pásu na druhý. Tato technika je dnes označována jako lineární střih videa [24]. Nejjednodušší lineární střižna (obrázek 2) je tvořena dvěma video rekordéry. Jeden video rekordér, označovaný jako *Source* (1), slouží jako přehrávač zdrojových pásek a druhý rekordér *Master* (2) zaznamenává vybrané záběry na cílovou pásku. Činnost obou rekordérů řídí a synchronizuje externí řídicí jednotka *Editing controller* (3). Kontrolní jednotka umožňuje vytvářet střihy, přechodové efekty a případně z dalších zdrojů přidávat titulky a zvuk [9].

Lineárnost střihu vyplývá jednak z přístupu k záznamům (přetáčením pásky) a ze způsobu řazení záznamů. Nově vkládané záběry jsou vkládány na cílovou pásku sekvenčně neboli lineárně za sebou. Pokud je potřeba vložit záběr mezi záběry na cílové pásce, jsou původní záznamy překryty vkládaným. Proto je nutné nejdříve původní záběry nakopírovat na pomocnou pásku, záběr vložit a nakopírovat původní materiál zpět. Podstatnou nevýhodou této techniky je, že kopírováním se kvalita záznamu snižuje. Další důležitou vlastností lineárního střihu je, že probíhá v reálném čase. Tedy zápis snímku s obrazovými efekty musí trvat stejně dlouho jako zápis snímku bez efektů.

V současnosti lineární střižny nacházející uplatnění zejména ve studiích televizního zpravodajství, kde je potřeba rychle přepínat mezi mnoha živými vstupy a přidávat dodatečné textové a zvukové informace. Tyto střižny jsou nazývány online lineární střižny [24]. Je-li dnes třeba analogový záznam na video kazetách nově upravit, zpracovává se digitálně.

Obrázek 2: Lineární střižna
JVC z 80. let, lineární střižna televizního studia (1999), převzato z [25] a [24]

2.3 Nelineární střih videa

Metoda nelineárního střihu vznikla s příchodem technologie digitálního videa. Digitální video může být získáno přímo digitální kamerou nebo digitalizací analogového obrazu. Digitální kamera pracuje prakticky shodně jako analogová video kamera. Světlo procházející objektivem dopadá na světlocitlivý elektronický snímač (např. CCD prvek), který generuje digitální signál reprezentující snímaný obraz. Takto zachycený signál je zaznamenáván na digitální datové medium v podobě digitálních kazet (např. MiniDV), pevného disku, flash paměti či DVD disku. Získaný záznam je přenesen do počítače, např. pomocí rozhraní IEEE 1394 (*Firewire)* nebo USB 2.0, kde je dále zpracován. Další z možností získání digitálního videa je digitalizace záznamu na video kazetách pomocí střihových karet s analogovým vstupem nebo naskenováním filmového pásu.

Digitální technologie přináší revoluční změnu do oblasti zpracování obrazu i zvuku. Obraz a zvuk v digitální podobě lze v paměti počítače uchovávat, vytvářet libovolné množství kopií a, co je nejdůležitější, upravovat pomocí počítačových programů. Odstraňuje hlavní nevýhody zpracování analogového záznamu. Programy na zpracovaní videa dokáží přistoupit k libovolnému snímku videa v paměti za stejnou dobu (není potřeba sekvenčně převíjet zdrojovou a cílovou pásku jako u lineárního střihu) a přesouvat a kopírovat snímky bez ztráty kvality. Snímky lze řadit a zpracovávat v libovolném pořadí, proto se tato technika nazývá nelineární. Zápis snímků a tedy i jejich zpracování nemusí probíhat v reálném čase. Tak je možné na ně aplikovat výpočetně složité algoritmy, které vyžadují delší dobu pro zpracování, než je doba mezi dvěma sousedními snímky [9]. Díky tomu, že editory pracují s vnitřní paměťovou reprezentací vstupního videa, je možné, že střih a další změny se projeví ve výstupní video sekvenci a vstupní video zůstane v původní podobě. Nelineární střih je tedy nedestruktivní.

Typický problém, který se objevuje při zpracování digitálního videa je velikost datového toku a objem dat. Zejména v minulosti kapacita pevných disků a propustnost přenosových kanálů počítačů nedostačovala pro zpracování videa v plné kvalitě. Proto byla zavedena metoda o*ffline zpracování*.

Metoda offline zpracování znamená provádět střih videa v nízkém rozlišení na méně výkonných počítačích. Tato metoda se dnes využívá ve filmové postprodukci. Nejdříve je celý filmový pás naskenován pomocí skeneru zvaného telecine (obrázek 3) v nízkém rozlišení. Tento materiál je zpracován na osobních počítačích pomocí video editoru, je proveden střih a korekce barev. Výsledkem práce je střihová soupiska EDL (Edit decision list), která obsahuje všechny provedené střihy a další úpravy. Soupiska je předána filmovému skeneru, který automaticky zachytí v plném rozlišení pouze ty záběry, které jsou v soupisce, a provede požadované barevné korekce. Materiál v plném rozlišení je dále zpracováván na výkonných nelineárních střihových systémech jako je například Avid Symphony (obrázek 3) [26].

Obrázek 3:Nelineární střihový systém Avid Symphony,
systém telecine s digitálním filmovým skenerem, převzato z [28] a [27]

Rozvojem osobních počítačů a zlevňováním výpočetní techniky se otevřely dveře do tvorby videa široké veřejnosti. Pro základní zpracování videa stačí vlastnit digitální kameru, osobní počítač a střihový program. O střihových programech pojednává následující kapitola.

3 Střihové programy

Střih představuje nejdůležitější a nejnáročnější část zpracování filmu. Nejdříve je potřeba všechny natočené záběry shlédnout a oříznout jejich délku na požadovanou. Materiál z kamery bývá často pořízen za různých světelných podmínek a proto je potřeba upravit kvalitu obrazu jednotlivých záběrů pomocí filtrů, které umožňují změnit úroveň jasu, kontrastu a celkového barevného vyvážení. Následuje hlavní tvůrčí činnost, kdy jsou záběry řazeny do posloupnosti zvané scénář. Záběry navazující na sebe jsou propojeny pomocí přechodových efektů. Střih se nevztahuje pouze na obraz, ale i na doprovodný zvuk. Film pracuje s více zvukovými stopami, typicky obsahuje zvlášť stopu pro dialogy a pro hudbu. Stopy lze upravovat a navzájem míchat pomocí zvukových filtrů a efektů. Za všechny tyto úkoly má zodpovědnost osoba nazývaná střihový editor, jehož mocným nástrojem je střihový program, který bývá přezdíván „digitální nůžky“.

Programů pro úpravu a zpracování videa dnes existuje na trhu celá řada. Hlavním kriteriem pro jejich rozdělení je prostředí, ve kterém budou použity. Záleží tedy na tom, zda jsou určeny pro profesionální účely nebo pro zpracování domácího videa. Jednotlivé programy se liší zejména počtem nástrojů, efektů a v neposlední řadě i cenou.
Z řad profesionálních nástrojů lze jmenovat Avid Media Composer, Apple Final Cut Proa Adobe Premiere Pro. Tyto programy podporují editaci videa ve velmi vysoké kvalitě, umožňují jemné úpravy obrazu a zvuku a obsahují širokou škálu filtrů a efektů, které lze detailně nastavovat. Cena těchto profesionálních nástrojů se pohybuje od 800 po 2500 dolarů [29]. Alternativou ke komerčním střihovým aplikacím jsou open source projekty jako je Kdenlive, Cinelerra a Kino, které se svou užitnou hodnotou komerčním programům v mnohém vyrovnávají.

U aplikací pro zpracování domácího videa je kladen důraz zejména na jednoduchost a přehlednost jejich ovládání. Tyto aplikace mohou být i součástí operačního systému. Typickými zástupci jednoduchých editorů videa jsou Apple iMove a Microsoft Windows Movie Maker. Slouží zejména pro potřeby uživatelů, kteří chtějí video ze své kamery jednoduše získat a provést v něm základní úpravy jako je například změna jasu, doplnění titulků a zvukového doprovodu.

Grafické rozhraní všech video editorů se skládá z několika společných prvků. Je to okno s časovou osou, okno pro přehrávání zdrojových záběrů, okno pro přehrávání časové osy a seznam zdrojových záběrů. Toto uspořádání se snaží napodobit vzhled starých střihových stolů (obrázek 2). Grafické prvky zabírají velké množství prostoru na pracovní ploše a proto je vhodné používat i více monitorů. Význam jednotlivých prvků je popsán v následující podkapitole na programu Adobe Premiere.

V průběhu práce jsem měl možnost si vyzkoušet práci s video editory: Adobe Premiere Pro 2.0, Kdenlive 0.5 a Virtual Dub v1.8.8. V praktické části práce se pak snažím implementovat systém, který má podobné možnosti jako jednoduchý video editor.

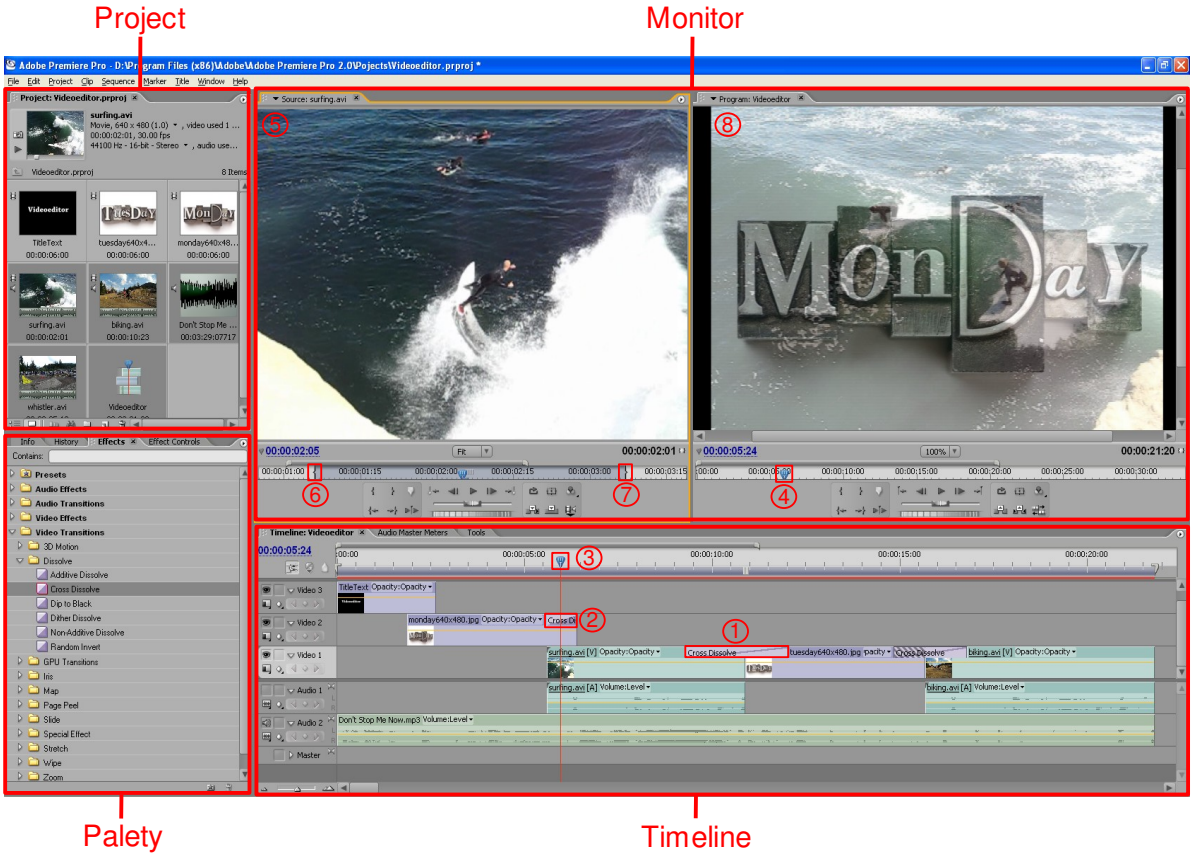
3.1 Adobe Premiere

Adobe Premiere je profesionální nástroj pro nelineární střih videa. Jedná se o jeden z nejznámějších programů tohoto druhu a na poli zpracování videa je podobně úspěšný jako Adobe Photoshop pro zpracování fotografie. Jeho první verze z roku 1991 byla vyvinuta pro platformu počítačů Macintosh, o dva roky později byl program přenesen i na platformu Windows.

Program je distribuován ve dvou verzích, které se liší počtem funkcí a cenou. Adobe Premiere Pro je plnohodnotná verze, která obsahuje rozsáhlou nabídku nástrojů, filtrů, efektů a podporuje zpracování a export videa mnoha formátů. Například zpravodajská stanice BBC používá Adobe Premiere již dlouhá léta. Od profesionální verze je odvozena verze Adobe Premiere Elemets, která je zaměřena na zpracování domácího videa z minikamer, digitálních fotoaparátů a mobilních telefonů. Prostředí programu je celkově zjednodušeno a funkčně omezeno. Tato verze je dodávána jako součást vybavení k některým střihovým kartám a video kamerám.

Fenomén sdílení videa na internetu dal podnět ke vzniku produktu Adobe Premiere Express. Ten umožňuje jednoduchou editaci videa online pomocí webového prohlížeče.

Jak již bylo zmíněno, většina střihových aplikací si je navzájem podobná, princip práce s nimi bude vysvětlen na aplikaci Adobe Premiere Pro 2.0. Většina střihové práce se odehrává ve třech hlavních oknech: okno časové osy – *Timeline*, okno pro přehrávání – *Monitor* a okno s načtenými záběry – *Project*. Doplňující nástroje a nabídky se nachází v okně *Palety*. Následující obrázek znázorňuje pracovní plochu editoru.



Obrázek 4: Pracovní plocha Adobe Premiere Pro 2.0

Okno Timeline

Okno obsahuje časovou osu, na kterou jsou umisťovány načtené záběry a zvukové nahrávky. Cílem úprav na časové ose je uspořádat záběry tak, aby tvořily výsledný scénář.

Časová osa je vodorovně rozdělena na několik video stop (*Video1, Video2 a Video3* v obrázku 4) a zvukových stop (*Audio1, Audio2*). V programu může vystupovat až 99 stop. Stopa s nižším pořadovým číslem (*Video1),* má nižší prioritu, zůstává na pozadí. Pokud jsou dva záběry souběžně umístěny do různých stop, bude záběr z nižší stopy (*Video1*) překryt záběrem z vyšší stopy (*Video2*). Pokud se překrývají zvukové stopy, tak dojde k jejich promíchání.

Umístěním dvou záběrů za sebou do stejné video stopy je provedeno jejich spojení. Záběry mohou být spojeny ostrým střihem nebo pomocí přechodových efektů. Ty jsou je vytvořeny vložením přechodu z palety *Video Transitions* mezi spojované záběry (viz bod 1 v obrázku 4). Přechodové efekty lze uplatnit i na překrývající se záběry umístěné v různých stopách (2). Tento přístup dovoluje kombinovat obraz i z více než dvou stop.

V horní části okna se nachází měřítko osy. Osu lze nastavit ve snímcích nebo v čase. Snímková osa se využívá zejména pro přesné nastavení přechodových efektů. Důležitou součástí časové osy je *jezdec* (3). Jím se lze po ose pohybovat. Jezdec je zároveň propojen s druhým jezdcem (4) v okně *Monitor,* které zobrazuje, co se na daném místě nachází. Tak lze kontrolovat průběh tvorby. V pozici jezdce je možné učinit střih, který daný záběr rozdělí na dva nezávislé záběry. S každým záběrem pak lze samostatně pracovat – přesouvat, vymazat, filtrovat atd. Výhodou a charakteristickým rysem nelineárního střihu je možnost vložit záběr i dodatečně mezi záběry ostatní, přičemž záběry za místem vložení jsou odsunuty. Tento princip se nazývá obrazový a zvukový *insert* [5].

Video lze doplnit titulky. S titulky se zachází stejně jako se záběry. Používá se pro ně samostatná stopa (např. *Video3*), která tvoří popředí. Titulek tvoří text na průhledné ploše, která se může po obrazovce pohybovat, mizet či měnit barvy.

Na záběry na časové ose lze aplikovat obrazové a zvukové efekty jejich přetažením z palety *Effects.* Tyto efekty mohou například: obraz zaostřit, rotovat, deformovat nebo měnit jeho barevné složky. Efekty je možné i kombinovat. Nelze je však aplikovat pouze na část záběru. Pokud se má efekt projevit pouze na části záběru, je nutné ho nejdříve rozdělit pomocí střihu jezdcem. Efekty je možné nastavit v paletě *Effect Controls*. Pro tvorbu nadstandardních efektů je také možné využít program Adobe After Effects.

Okno Monitor

Je složeno ze dvou náhledových oken *Source* a *Program.* Okno *Source* (5) slouží pro úpravy a přehrávání načtených záběrů. Součástí okna je přehrávací osa, na které se nastaví body střihu *in* (6) a *out* (7) a dále nástroje pro úpravu kvality obrazu a zvuku. V okně *Program* (8) je možné sledovat tvorbu na časové ose.

Okno Project

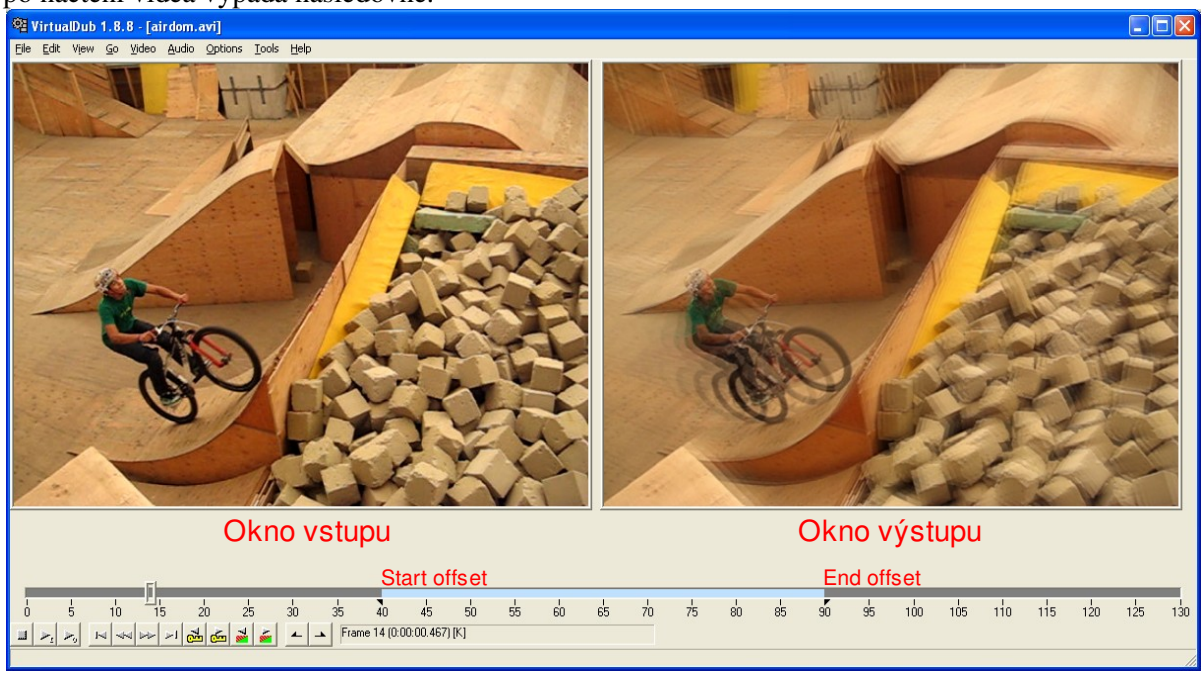
Okno zobrazuje a organizuje zdrojový materiál: načtené záběry, obrázky, vytvořené titulky a zvukové nahrávky. Záběry jsou do projektu načteny buď přímo z kamery, nebo ze souborů uložených na pevném disku případně na jiném datovém mediu. Každý záběr se skládá z video a audio stopy. Přetažením záběru na časovou osu se stane záběr součástí vytvářeného scénáře.

Informace o programu Adobe Premiere byly převzaty z materiálů [7], [6] a [20].

3.2 VirtualDub

Jedná se o jednoduchý program pro zachycení a zpracování videa na platformě Windows. Program je distribuován pod licencí GNU GPL. VirtualDub nepatří mezi tradiční aplikace pro střih videa. Aplikace je primárně určena pro komprimaci a filtraci videa a zvuku uloženého ve formátu AVI. Umí video ořezat, upravit jeho obrazové a zvukové vlastnosti, změnit kodek a výsledek zapsat do kontejneru AVI. Je užitečným nástrojem zejména při přípravě materiálu před dalším zpracováním ve střihových programech. Za vznikem a vývojem celé aplikace stojí jediná osoba, kterou je Avery Lee. VirtualDub původně vznikl jako jednoduchý nástroj pro potřeby autora a časem se stal pro svoji výkonnost, jednoduchost a dostupnost velmi oblíbeným a vyhledávaným.

Program je optimalizován pro rychlé lineární operace s videem. Podporuje načtení videa ve formátu AVI, MPEG-1 a řady BMP obrázků. Specialitou programu VirtualDub jsou rozšířené možnosti při načítání videa. Umožňuje například soubory načíst a zpracovat dávkově, pracovat s poškozenými AVI soubory a odstranit z videa půlsnímky (deinterlace). Pracovní plocha programu po načtení videa vypadá následovně.



Obrázek 5: Pracovní plocha Virtual Dub v1.8.8 (aplikace filtru motion blur)

Levé okno zobrazuje načtené video v původní podobě a pravé okno výslednou sekvenci po úpravách. Pod nimi je snímková osa s jezdcem platná pro obě okna. Střih video sekvencí ve VirtultualDubu pracuje na velmi jednoduchém principu, obsahuje pouze funce pro přidání a vymazání záběru. Na snímkovou osu lze přidávat videa pomocí funkce *Append AVI Segment.* Tato funkce zařadí vkládané video za poslední vložené. Postup připomíná práci při lineárním střihu video pásky. Střih je uskutečněn odstraněním nevhodné části videa. Pomocí bodů *Start offset* a *End offset* je možné označit nevhodnou část a vymazat ji.

Jednou z nejdůležitějších částí VirtualDubu jsou filtry. Ty jsou aplikovány vždy na celou načtenou video sekvenci a výpočet většiny z nich probíhá v reálném čase. Program obsahuje širokou paletu integrovaných filtrů a tu je možné rozšířit i o filtry vlastní.

Neméně důležitou schopností programu je komprimace dat. Umožňuje určit komprimaci zvlášť pro video a audio, přičemž využívá kodeků nainstalovaných v systému. Pro výstupní video sekvenci lze dále nastavit parametry jako je počet klíčových snímků, rozlišení, snímkovou frekvenci a další. Informace o programu VirtualDub byly převzaty z materiálů [19] a [6].

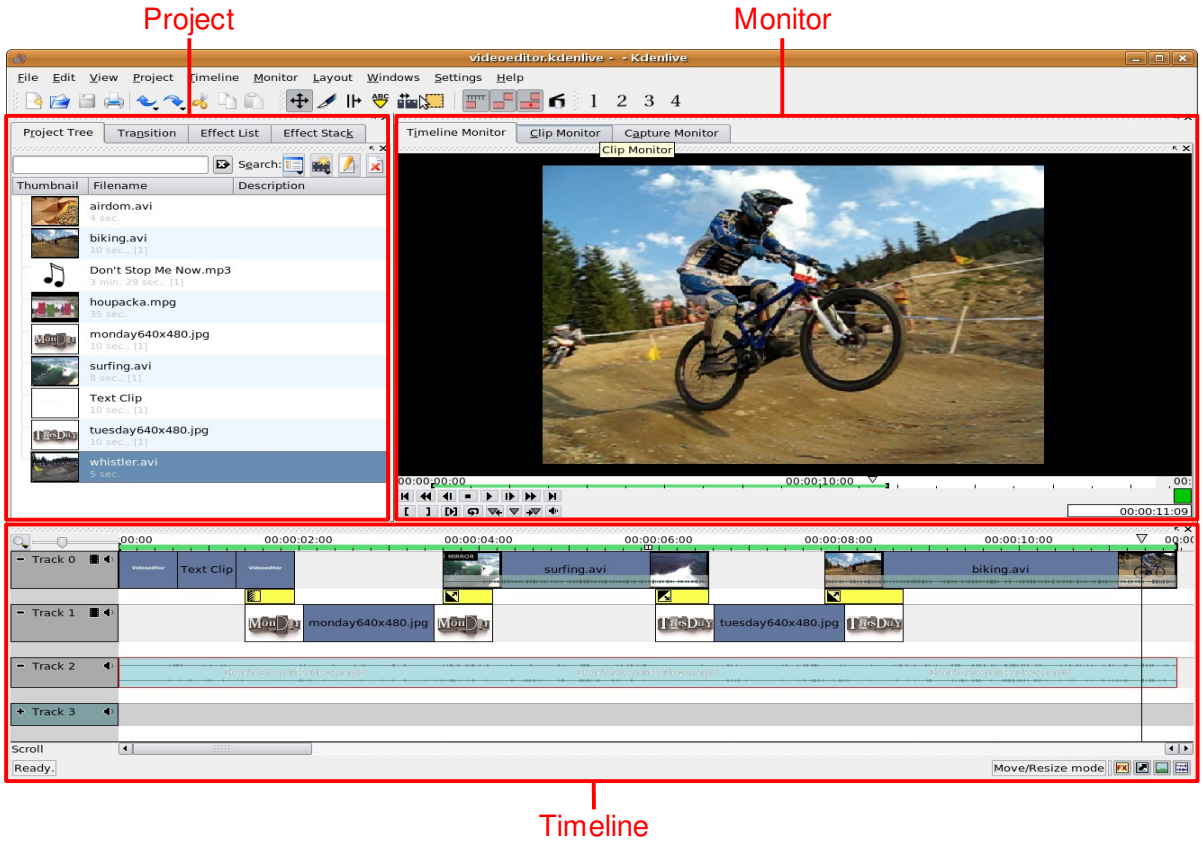
3.3 Kdenlive

Kdenlive je nelineární video editor pro platformu GNU/Linux a FreeBSD. Jedná se o open source projekt (lincence GNU GPL), který založil Jason Wood v roce 2002. Dnes je projekt zpravován malým týmem vývojářů a komunitou dobrovolníků. Kdenlive je považován za jeden z nejpokročilejších programů pro střih videa pod operačním systémem GNU/Linux.

Program je postaven na frameworku MLT (Media Lovin' Toolkit) a FFmpeg [17]. Multimediální framework MLT zajišťuje práci se záběry, stopami, efekty a přechody. FFmpeg umožňuje nahrávání a konverzi formátů videa a zvuku.

Kdenlive podporuje širokou škálu formátů, jako je například AVI, MPEG, DV. Umožňuje také pracovat s videem ve formátu HDV. Umí video zachytávat z DV zařízení a některých webkamer. Umí také zachytit dění na pracovní ploše a export videa na DVD disky. Obsahuje základní paletu efektů a přechodů, nástroje pro vytváření titulků a obrázkových klipů.

Grafické rozhraní aplikace vychází z tradičního modelu střihových programů. Obsahuje opět okno *Project*, *Monitor* a *Timeline*.



Obrázek 6: Pracovní plocha Kdenlive 0.5

Okno *Project* nabízí záložku s projektovým stromem obsahující načtený materiál, záložku video přechody, efekty a nastavení efektů. Okno *Monitor* obsahuje záložky pro přehrávání časové osy, načtených záběrů a zachytávaného videa z kamery. *Timeline* – časová osa může obsahovat neomezené množství video a audio stop. Záběry na ose znázorňují počáteční a koncový snímek. V časové ose lze zapínat a vypínat efekty, přechody a zobrazování zvukové stopy. Video efekty zahrnují například maskování části obrazu, klíčování na barvu a deformace obrazu a zvuku.

V průběhu práce na projektu jsem měl příležitost si vyzkoušet Kdenlive 0.5. Aktuální verze Kdenlive 0.7.3, vydaná v dubnu 2009, je rozšířena o další prvky jako například panel historie, nové efekty a také o českou lokalizaci. Informace o programu byly převzaty z materiálů [16], [17] a [18].

4 Videoeditor

Tato kapitola se věnuje jádru bakalářské práce, které představuje návrh architektury a implementace programu pro nelineární střih videa. Úvodní část je věnována charakteristice programu, použitým implementačním prostředkům a návrhu architektury. Ten obsahuje popis střihového modelu a stručné seznámení s komponentami, které tvoří základní kameny video projektu.

Videoeditor je konzolová aplikace, která poskytuje základní paletu nástrojů pro editaci videa. Aplikace umožňuje:

- nelineární střih videa,
- práci se statickým obrazem,
- filtraci obrazu,
- přehrávání video sekvencí,
- export výsledné sekvence do souboru.

Program je určen zejména pro dávkové zpracování velkého množství video souborů do jedné výstupní sekvence. Veškerá logika editace je skrytá v XML konfiguračním souboru, který reprezentuje založený *projekt*. V projektu je určeno, jaká videa budou načtena, jak budou upravena a které filtry na ně budou aplikovány. Výslednou sekvenci, představující *scénář* projektu, lze potom přehrát nebo zapsat do souboru. Parametry zapisovaného videa je možné zvolit při spuštění programu.

4.1 Implementace

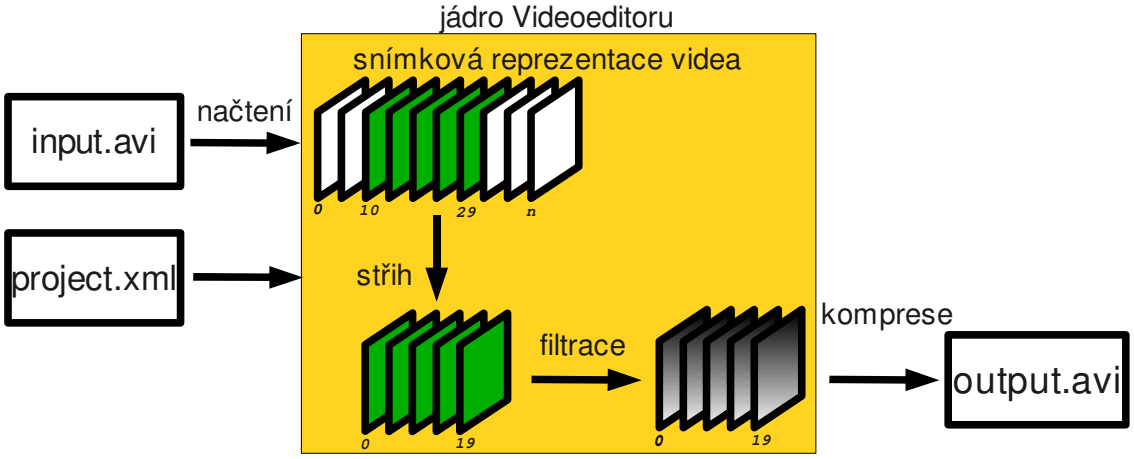
Aplikace Videoeditor je implementována v jazyce C++, bylo využito principů objektově orientovaného návrhu. Pro veškerou práci s obrazem byla použita multiplatformní knihovna OpenCV [3].

OpenCV (Open Source Computer Vision) je knihovna, která byla vyvinuta pro zpracování problémů v oblasti počítačového vidění. Tato knihovna byla zvolena, protože je volně dostupná a poskytuje širokou škálu funkcí pro zpracování obrazu a videa. Úspěšný překlad a chod programu je podmíněn instalací knihovny OpenCV verze 1.0 a vyšší.

Pro práci s XML dokumenty byla využita knihovna TinyXML [4]. Tato knihovna obsahuje jednoduchý XML parser.

Aplikace byla vyvíjena a laděna pod operačním systémem GNU/Linux. Kód programu je přenositelný i na operační systém MS Windows.

4.2 Návrh architektury



Obrázek 7:Schéma zpracování jedné video sekvence

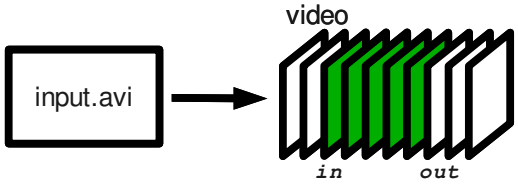
Návrh architektury programu vychází z koncepce nelineárního střihu, který definuje jednotlivé kroky zpracování video sekvencí. Schéma na obrázku 7 vymezuje tento postup.

Vstupní video soubor je nejdříve načten jako sekvence jednotlivých snímků. Snímková reprezentace videa je velmi podobná klasickému filmovému pásu. K jednotlivým snímkům je přistupováno přes jejich indexy a mezi nimi jsou provedeny střihy. Střih je proveden podle střihové soupisky, která je obsahem projektu. Celá logika digitálního střihu, na rozdíl od střihu filmu, nespočívá v přemisťování a kopírování snímků z jednoho místa na druhé, ale pouze v přepočítání snímkových indexů. Tento přístup vede ke značným výpočetním i paměťovým úsporám. Navíc změny provedené editorem se nepromítnou do vstupních video souborů.

Důležitou součástí úprav videa je filtrace, kdy jsou na jednotlivé snímky uplatněny příslušné filtry a efekty. Nakonec jsou upravené snímky komprimovány a zapsány do souboru. Žluté pole vyznačené v obrázku 7 označuje jádro střihového programu, které bylo implementováno.

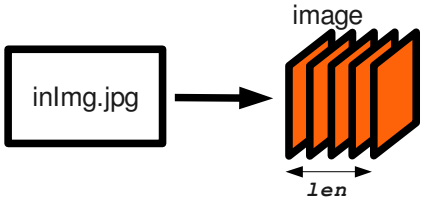
Na základě analýzy modelu střihu a požadovaných vstupů byly navrženy hlavní komponenty video projektu. Projekt obsahuje tři objekty nesoucí obraz:

1) **Video** obsahuje záběr určitě délky ze zdrojového videa.



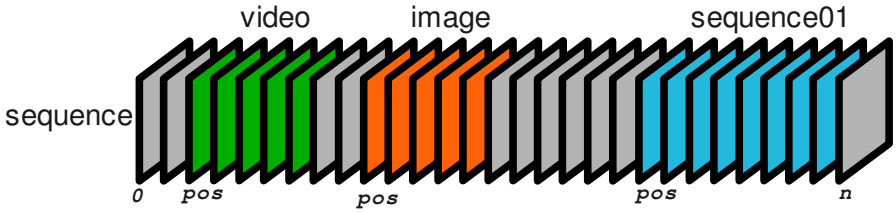
Obrázek 8: Objekt video

2) **Obrázek** představuje sekvenci snímků určité délky vytvořenou ze zdrojového obrázku.



Obrázek 9: Objekt obrázek

3) **Sekvence** tvoří snímkovou osu, na jejíž pozice jsou umísťovány jednak zdrojové objekty *video* a *obrázek* a nebo další *sekvence*. Sekvence se stává univerzálním objektem, který může vystupovat jak v roli obálky pro zdrojové záběry, tak v roli samotného scénáře projektu.



Obrázek 10: Objekt sekvence

5 Editace videa

Princip práce s Videoeditorem je podobný jako u střihových programů, které byly popsány v kapitole 3. Nejdříve je založen nový projekt. Poté jsou načteny sekvence se záběry, které budou v projektu použity. Následuje hrubý střih, kdy jsou načtené sekvence řazeny do posloupnosti na snímkové ose scénáře. V průběhu tvorby projektu je možné si vytvářené sekvence kdykoliv prohlédnout v okně pro přehrávání. Jakmile je dokončen hrubý střih projektu, lze jednotlivé sekvence doplnit efekty. Efekty zahrnují filtrací snímků, doplnění titulků a změnu rychlosti přehrávání. Nakonec se podle obsahu scénáře zapíše výsledné video do souboru, přičemž lze zvolit jeho rozlišení a snímkovou frekvenci.

5.1 Projekt

Projekt je zapsán v XML konfiguračním souboru. Struktura XML dokumentu byla navržena tak, aby tvořila rozhraní mezi jádrem střihového programu a budoucím grafickým rozhraním. Je tedy určena především pro strojové čtení.

Nový projekt je založen spuštěním programu s parametrem:
`-n|--newProj <jméno nového projektu>`

```
./videoeditor -n projects/newProject.xml
```

Pokud existuje v adresáři projekt se stejným jménem, bude původní projekt nahrazen novým. Obsah nového dokumentu tvoří prázdný projekt:

```
<?xml version="1.0" encoding="US-ASCII" ?>
<videoconfig>
    <!-- Project content -->.
</videoconfig>
```

Pro přehrání libovolné video sekvence v projektu, je program spuštěn s parametry:
`-p|--project <jméno projektu> -s|--seqID <identifikátor sekvence>`.

```
./videoeditor -p projects/firstProject.xml -s Week
```

Zápis video sekvence do souboru v původním rozlišení:
`-o|--output <jméno souboru>`

```
./videoeditor -p projects/firstProject.xml -s Week -o output/Week.avi
```

Zápis video sekvence s novou hodnotou rozlišení a snímkové frekcence:
`-r|--resolution <výška x šířka> -f|--fps <snímková frekvence>`

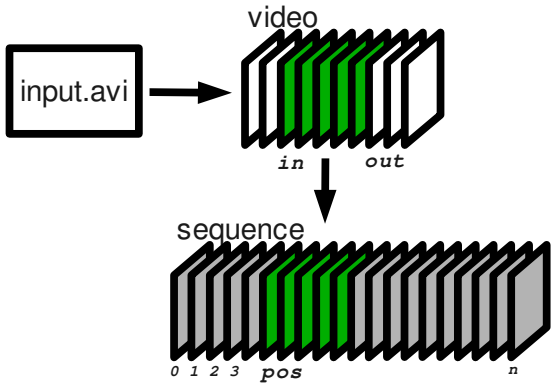
```
./videoeditor -p projects/firstProject.xml -s Week -o output/Week720x576.avi
 -r 720x576 -f 25
```

Všechny ukázky v následujícím textu lze najít v projektu `firstProject.xml` a je možné je ihned vyzkoušet. Výstupní video sekvence z ukázek se nachází v adresáři `examples/firstProject/`.

5.2 Načtení záběrů

V první fázi střihu jsou vybrány záběry, které budou v projektu použity. Záběry mohou být načteny z video souboru nebo vytvořeny ze statického obrázku.

5.2.1 Načtení videa



Obrázek 11: Načtení videa

Princip načtení videa vystihuje obrázek 11. Element *video* v projektu představuje výsek snímků načtený ze zdrojového video souboru. Zdrojový soubor je určen parametrem *src*. Délka záběru se nastavuje pomocí vstupního bodu (*in*) a výstupního bodu (*out*) videa. Tyto dva parametry jsou základem pro veškeré střihy. Aby bylo možné s videm dále pracovat, je třeba ho umístit na snímkovou osu sekvence. Pozici počátku videa na snímkové ose sekvence udává parametr *pos*.

Jsou tedy určeny tři body: vstupní a výstupní bod videa a počáteční bod v sekvenci. Čtvrtý bod střihu – koncový bod v sekvenci – je programem dopočítán. Tento princip se nazývá tříbodový střih [5]. Reprezentace videa v projektu je následující:

```
<video src="video/surfing.avi" in="0" out="240" pos="0"/>
```

Parametry:

- **src=** Cesta ke zdrojovému video souboru.
  Videoeditor zpracovává pouze určitý formát video souborů. Podrobnou specifikaci těchto formátů lze nalézt v kapitole 8. Základní podmínkou je, že každý snímek videa musí být klíčový. Jedině tak je možné přistupovat k libovolnému snímku ve videu.
- **in=** Index prvního snímku videa.
  Nabývá hodnot celých kladných čísel od 0 do počtu snímku video souboru. Počet snímků video souboru lze zjistit příkazem: `./videoeditor --getFrames video/surfing.avi`
- **out=** Index za posledním snímkem videa.
  Nabývá hodnot celých kladných čísel větších jak 0 a menších jak počet snímků video souboru. Pokud není omezení rozsahu hodnot dodrženo, je zobrazen černý snímek.
- **pos=** Pozice na snímkové ose sekvence, do které je video umístěno.
  Nabývá hodnot celých kladných čísel od 0 do teoreticky neomezené délky snímkové osy sekvence. Při umísťování záběrů na snímkovou osu je třeba dávat pozor, aby se záběry v sekvenci vzájemně nepřekrývaly. To je zajištěno tak, že pozice vkládaného záběru je rovna součtu pozice a délky předchozího záběru.

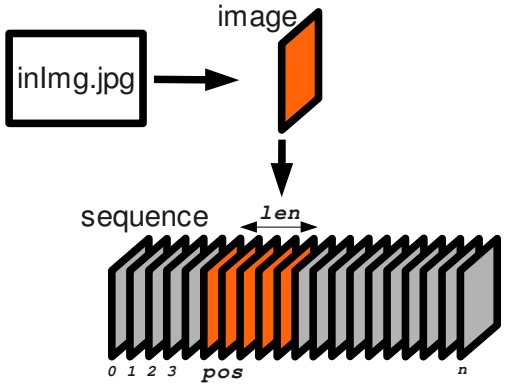
Pro střih překrývajících se sekvencí existují obecně dva přístupy [5]: *vložení (insert)* a *překrytí (overlay)*. Střih *vložením* pracuje na stejném principu jako lepení filmového pásu. Snímky původní sekvence jsou rozstřiženy na pozici vložení, nová sekvence je vložena a snímky, které se původně nacházely za místem vložení, jsou odsunuty na konec nově vložené sekvence.

Střih *překrytím* jednoduše nahradí původní snímky, snímky vkládanými. Videoeditor střih překrývajících se sekvenci neuvažuje. Zavedení těchto principů by mělo smysl zejména v kombinaci se zavedením video stop, které umožňují mezi překrývajícími se sekvencemi vytvářet přechody.

Nasleduje ukázka načtení jednoho záběru. Záběr je načten z video souboru `surfing.avi`, má délku 240 snímků a je umístěn na začátek snímkové osy sekvence *Surf*.

```
<?xml version="1.0" encoding="US-ASCII" ?>
<videoconfig>
   <sequence id="Surf">
      <video src="video/surfing.avi" in="0" out="240" pos="0"/>
   </sequence>
</videoconfig>
```

5.2.2 Načtení obrázku



Obrázek 12: Načtení obrázku

Princip načtení ukazuje obrázek 12. Element *image* v projektu představuje sekvenci snímků vytvořenou ze zdrojového obrázku. Obrázek je načten jako jediný snímek a v sekvenci je zobrazen *len*-krát za sebou. Zdrojový obrázek je určen parametrem *src***.** Stejně jako u načítání *videa*, je pomocí parametru *pos* obrázková sekvence umístěna na snímkovou osu dané sekvence. Reprezentace obrázku v projektu je následující:

```
<image src="image/television720x576.jpg" len="50" pos="0"/>
```

Parametry:

- **src=** Cesta ke zdrojovému obrázku..
  Veškerý vstupní obraz musí mít stejné rozlišení a barevnou hloubku 24b RGB. Jinak by nebylo možné snímky jednotně zpracovávat.
- **len=** Počet snímků obrázkové sekvence.
- **pos=** Pozice na snímkové ose sekvence, ve které je obrázková sekvence umístěna. Pro tento prametr platí stejná pravidla jako v elementu *video*.

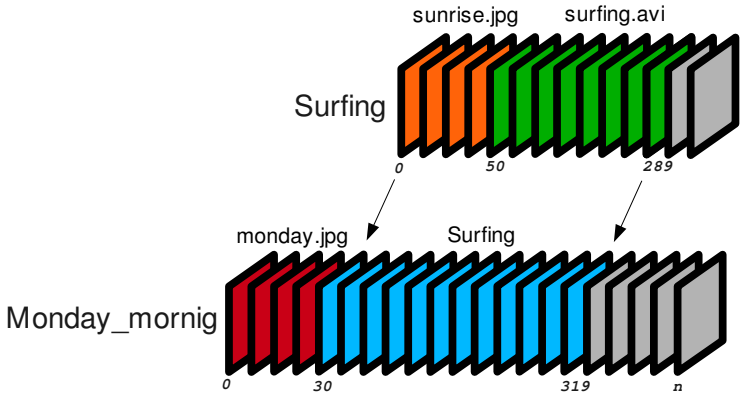
Příklad načtení obrázku:

```
<?xml version="1.0" encoding="US-ASCII" ?>
<videoconfig>
  <sequence id="Television">
    <image src="image/television720x576.jpg" len="50" pos="0"/>
  </sequence>
</videoconfig>
```

5.3 Vytvoření scénáře

Další fází střihu je tzv. „spojovaní". V této fázi jsou načtené záběry řazeny na snímkovou osu sekvence zvané scénář. Při vytváření scénáře je důležité mít možnost členit jednotlivé záběry do logických celků. Sekvence právě tuto možnost dává. Úpravy na snímkové ose scénáře se pak stávají snazší a přehlednější. Jak sekvenci vytvořit, jak je možné sekvence mezi sebou odkazovat a jaké důležité role sekvence při střihu zastává popisují následující podkapitoly.

5.3.1 Vytvoření sekvence



Obrázek 13: Vytvoření sekvence

Element *sequence* v projektu tvoří snímkovou osu na jejíž pozice jsou umisťovány zdrojové záběry *video* a *image* a další *sequence*. Při vkládání záběrů do sekvence je délka její snímkové osy teoreticky neomezená. Ovšem v situaci, kdy je sekvence vkládána do jiné sekvence, je délka vkládané sekvence vypočítána jako součet pozice a délky jejího posledního elementu (viz obrázek 13). Sekvence nacházející se v projektu jsou rozlišeny svými jedinečnými jmény.

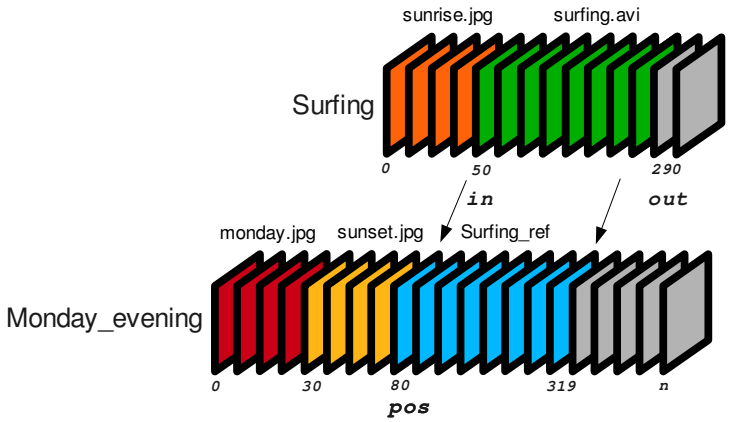
Obrázek 13 zachycuje, jak byly vytvořeny sekvence *Surfing* a *Monday_morning*. Obsah obrázku odpovídá následujícímu zápisu:

```
<sequence id="Monday_morning">
 <image src="image/monday640x480.jpg" len="30" pos="0"/>
 <sequence id="Surfing" pos="30">
  <image src="image/sunrise640x480.jpg" len="50" pos="0"/>
  <video src="video/surfing.avi" in="0" out="240" pos="50"/>
 </sequence>
</sequence>
```

Parametry:

- **id=** Řetězcový identifikátor sekvence.
  Identifikátor – jméno sekvence musí být v rámci projektu jedinečné.
- **pos=** Pozice na snímkové ose sekvence, ve které je vkládaná sekvence umístěna.
  Tento parametr je povinný pouze při vkládání sekvence do jiné sekvence, platí pro něj stejná pravidla jako u elementu *video*.

5.3.2 Odkazování sekvence



Obrázek 14: Odkazování sekvence

Zvláštním případem sekvence je sekvence s vlastností *ref*, která umožňuje odkazovat se na již dříve vytvořené sekvence. Na odkazovanou sekvenci lze pak nahlížet stejně jako na zdrojový záběr, který je možné dále střihat opět pomocí parametrů *in*, *out* a *pos*. Toho způsobu střihu lze s výhodou využít při dolaďování projektu (*trimming* [5]). Dovoluje měnit délku sekvence, aniž by bylo nutné upravovat vstupní a výstupní body zdrojových záběrů.

Následuje ukázka sekvence *Surfing_ref*, která se odkazuje na sekvenci *Surfing*, přičemž z ní odstřihne obrázkovou sekvenci *sunrise.jpg* (obrázek 14).

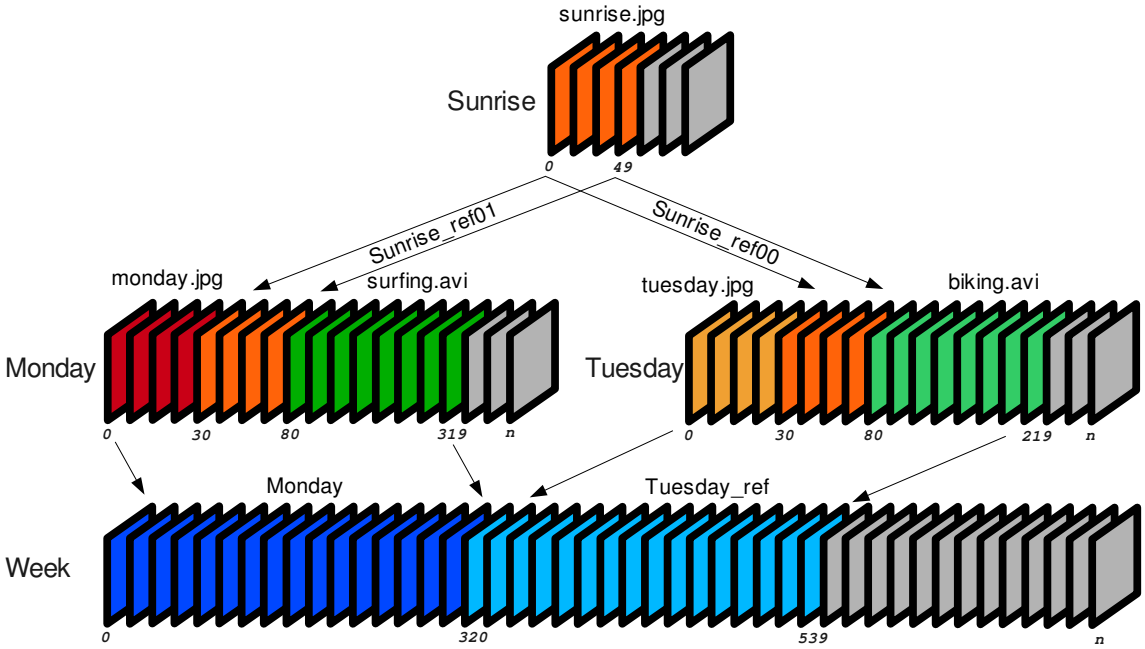
```
<sequence id="Monday_evening">
  <image src="image/monday640x480.jpg" len="30" pos="0"/>
  <image src="image/sunset640x480.jpg" len="50" pos="30"/>
  <sequence id="Surfing_ref" ref="Surfing" in="50" out="290" pos="80"/>
</sequence>
```

Parametry:

- **ref=** Identifikátor odkazované sekvence.
  Odkazovat lze pouze ty sekvence, které byly v projektu již dříve vytvořeny. Toto omezení zabraňuje zacyklení referencí. Další parametry *in* a *out* mají efekt pouze v kombinaci s parametrem *ref*. Pokud nejsou parametry *in* a *out* uvedeny, je odkazovaná sekvence vložena v plném rozsahu.
- **in=** Vstupní bod do odkazované sekvence.
- **out=** Výstupní bod z odkazované sekvence.

5.3.3 Role sekvence

Sekvence zapouzdřuje všechny obrazové elementy. Zastává role, které jsou ve střihových programech zavedeny.



Obrázek 15: Role sekvence: pojmenovaný záběr, scéna, scénář

1. **Pojmenovaný záběr**
   Představuje ten nejjednodušší případ, kdy pojmenovaná sekvence obsahuje jediný záběr. Pojmenovaný záběr lze odkazovat z jiných sekvencí (např. *Sunrise_ref00* v obrázku 15) a tak je umožněno stříhané záběry použít v projektu vícenásobně. Ukázkou pojmenovaného záběru je sekvence *Sunrise* nacházející se v následující ukázce.

2. **Scéna**

   Sdružuje více záběrů týkající se typicky jednoho časového úseku ve scénáři. Příkladem scény je sekvence *Tuesday*. Ve střihových programech se scény umisťují do video stop (*timelines* [5]). Obsah stop se může vzájemně překrývat a vytvářet přechody mezi scénami. Stopy se dále používají zejména pro oddělení zpracování zvuku, titulků a dalších efektů. Přidáním stop do Videoeditoru by bylo významným funkčním rozšířením. Podkapitola 8.2 obsahuje návrh, jak by bylo možné program o video stopy rozšířit.

3. **Scénář**

   Vytváří seřazenou posloupnost scén pro export. Scénář lze vytvářet nejen pomocí odkazů na hotové scény (*Tuesday_ref*), ale i definováním scény uvnitř scénáře (*Monday*). Výsledným scénářem projektu je sekvence *Week* obsahující scény *Monday* a *Tuesday*. Zápis projektu, který kompletně odpovídá obsahu obrázku 15, je uveden na následující straně.

Sekvence tedy umožňuje dekompozici zpracovávaného projektu na logické celky, které lze samostatně přehrávat, zapisovat, upravovat, vzájemně odkazovat a vnořovat.

```
<?xml version="1.0" encoding="US-ASCII" ?>
<videoconfig>

 <sequence id="Sunrise">
  <image src="image/sunrise640x480.jpg" len="50" pos="0"/>
 </sequence>

 <sequence id="Tuesday">
  <image src="image/tuesday640x480.jpg" len="30" pos="0"/>
  <sequence id="sunrise_ref00" ref="Sunrise" pos="30"/>
  <video src="video/biking.avi" in="10" out="150" pos="80"/>
 </sequence>

 <sequence id="Week">
  <sequence id="Monday" pos="0">
   <image src="image/monday640x480.jpg" len="30" pos="0"/>
   <sequence id="sunrise_ref01" ref="Sunrise" pos="30"/>
   <video src="video/surfing.avi" in="0" out="240" pos="80"/>
  </sequence>
  <sequence id="tuesday_ref" ref="Tuesday" pos="320"/>
 </sequence>

</videoconfig>
```

6 Obrazové efekty

Jakmile je dokončen hrubý střih projektu, lze jednotlivé sekvence a záběry doplnit efekty. Efekty zahrnují filtraci obrazu, doplnění titulků a změnu rychlosti přehrávání.

6.1 Filtrace obrazu

Videoeditor obsahuje filtry, které využívá pro různé typy úloh jako je detekce hran (Cannyho, Sobelův hranový detektor), vyhlazování obrazu (Gaussův filtr), rozmazání snímků v čase (Temporal a Average smooht), úprava hodnot jasu a kontrastu a převod obrazu na šedotónový a inverzní.

Filtry lze uplatnit obecně na jakoukoli snímkovou sekvenci. Mohou být tedy přiřazeny zvlášť objektům *video*, *obrázek* či *sekvence*. Každému elementu může být přiřazeno více filtrů, které jsou řazeny do fronty. Během zpracování sekvence jsou pak na jednotlivé snímky postupně aplikovány filtry v takovém pořadí, v jakém jsou zapsány v příslušném elementu směrem shora dolů.

Filtry jsou sekvencím a záběrům v projektu přiřazovány elementem *filter*. Druhy filtrů jsou od sebe odlišeny pomocí řetězcového identifikátoru *id* a specifické vlastnosti jednotlivých filtrů jsou nastaveny pomoci příslušných parametrů.

Obecná syntaxe zápisu filtru je následující:

```
<filter id="jmeno_filtru" [parametry]/>
```

Parametry:

- **id=** Řetězcový identifikátor filtru. Nabývá hodnot: *contrast*, *brightness*, *grayScale*, *invert*, *gauss*, *sobel*, *canny*, *tempSmooth*, *avrgSmooth* a *setText*. Následuje popis jednotlivých filtrů.

6.1.1 Převod na šedotónový obraz

Převod barevného obrazu na obraz ve stupních šedi. Převod spočívá v průměrování základních barevných složek RGB. Výpočet je uskutečněn pomocí rovnice [7] určující intenzitu obrazu:

$$Intensity = 0{,}299 \cdot R + 0{,}587 \cdot G + 0{,}114 \cdot B$$

Váhové koeficienty barev byly určeny na základě citlivosti lidského oka na intenzitu jednotlivých barevných složek. Následuje příklad uplatnění filtru na celou sekvenci *grayScale_seq*.

```
<sequence id="grayScale_seq">
  <image src="image/sydney640x480.jpg" len="30" pos="0"/>
  <video src="video/surfing.avi" in="0" out="100" pos="30"/>
  <filter id="grayScale"/>
</sequence>
```

Filtr lze stejným způsobem uplatnit na elementy *image* a *video*. Příklady použití všech filtrů lze nalézt v přiloženém projektu `filterTest.xml`. Tam se nachází i sekvence *Lena*, na které jsou demonstrovány účinky všech filtrů.

```
<sequence id="Lena">
  <image src="image/lena512x512.jpg" len="75" pos="0">
    <filter id="grayScale"/>
  </image>
</sequence>
```

Následující obrázek znázorňuje snímek před (vlevo) a po (vpravo) aplikací filtru.

Obrázek 16: Gray scale filtr: originální snímek, šedotónový snímek

6.1.2 Převod na inverzní obraz

Filtr vytvoří barevný negativ původního obrazu. Vytvoření negativu spočívá v inverzi hodnot jednotlivých barevných kanálů. Inverzní hodnota kanálu je vypočtena jako rozdíl původní hodnoty kanálu a hodnoty bílé barvy. Výpočet pro 24b RGB obraz vyjadřuje následující vztah:

$$InvertedChannel = 255 - channel$$

Následuje ukázka zápisu filtru a obrázek s výsledkem jeho aplikace.

```
<filter id="invert"/>
```

Obrázek 17: Invert filtr: originální snímek, negativ snímku

6.1.3 Úprava jasu

Filtr umožňuje úpravu hodnot jasu – intenzity jednotlivých barevných složek 24b RGB obrazu. Jedná se o nejjednodušší způsob úpravy jasu obrazu, kdy se tóny barvy rovnoměrně ztmaví nebo zesvětlí. Výpočet spočívá v přičtení konstanty *val* k původní hodnotě barevného kanálu *ch*. Výpočet vyjadřuje následující vztah:

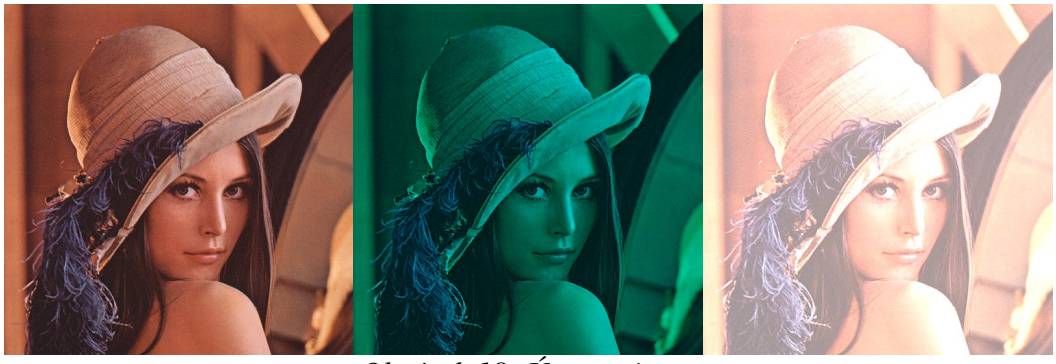
$$changedChannel = channel + val$$

Intenzity, které výpočtem přesáhnou interval 0 až 255 jsou oříznuty na hranice intervalu. To má za následek slití nejsvětlejších nebo nejtmavších tónů původního obrazu dohromady. Dochází tedy ke ztrátě detailu světla nebo stínu. Vhodnější metodou pro úpravu jasu by byla úprava pomocí křivek [7]. V následující ukázce je z obrazu odečtena červená složka. Výsledek je zobrazen na obrázku 18.

```
<filter id="brightness" ch="red" val="-255"/>
```

Parametry:

- **ch=** Řetězcový identifikátor barevného kanálu RGB. Nabývá hodnot: red, green a blue.
- **val=** Hodnota, která bude přičtena k původní hodnotě barevného kanálu.



Obrázek 18: Úprava jasu:
originál, odečtení červené složky, přičtení 125 všem složkám

6.1.4 Úprava kontrastu

Filtr slouží pro úpravu kontrastu 24b RGB obrazu. Kontrast je upraven tak, že se intenzita obrazu změní v určitém poměru vůči intenzitě původní. Nejjednodušší metodou je vynásobit původní obraz tímto poměrem. Hodnota poměru *percent* je udána v procentech. Výpočet pak vyjadřuje následující vztah:

$$changedChannel = channel \,.\, percent/100$$

Vypočtené intenzity jsou opět oříznuty intervalem 0 až 255. I v tomto případě by byla úprava kontrastu pomocí křivek vhodnější metodou [7].

V následující ukázce je hodnota kontrastu zvýšena na 150% původní hodnoty. Výsledek je zobrazen na obrázku 19.

```
<filter id="contrast" percent="150"/>
```

Parametry:

- **percent=** Hodnota kontrastu vyjádřena v procentech vzhledem k původnímu kontrastu obrazu.

Obrázek 19: Úprava kontrastu: originál, kontrast 150%, kontrast 50%

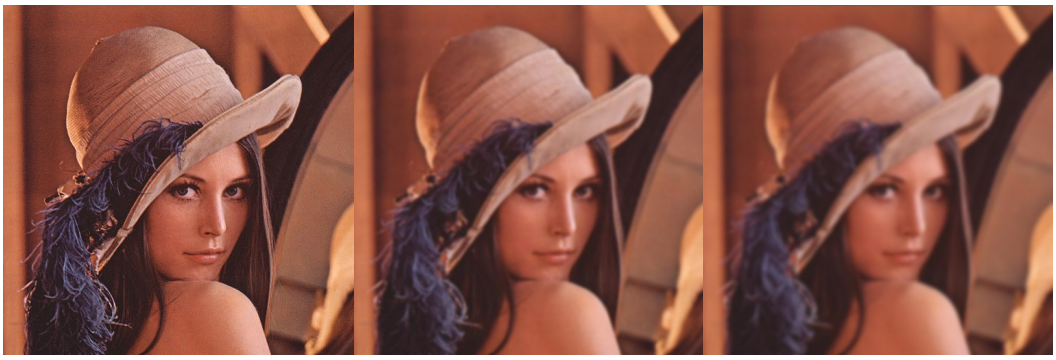
6.1.5 Gaussův filtr

Gaussův filtr se používá ke snížení obrazového šumu a rozmazání detailu v obraze [7]. Princip Gaussovského vyhlazovaní spočívá v konvoluci obrazu konvoluční maticí, jejíž koeficienty jsou vypočteny pomocí modifikované dvourozměrné Gaussovy funkce [11]. Vypočtená matice má tvar kruhového jádra, kde se váhy koeficientů snižují od středu matice. Stupeň rozmazání obrazu pak závisí na velikosti konvoluční matice (*kSize)*. V následující ukázce je obraz rozmazán Gaussovým filtrem s konvolučním jádrem o velikosti 11x11.

```
<filter id="gauss" kSize="11"/>
```

Parametry:

- **kSize=** Velikost konvoluční matice. Nabývá hodnot lichých přirozených čísel.



Obrázek 20: Gaussův filtr: originál, kSize 11, kSize 17

6.1.6 Sobelův hranový detektor

Ve Videoeditoru slouží tento filtr pro detekci vodorovných a svislých hran v obraze. Princip práce filtru je založen na konvoluci obrazu Sobelovým gradientním operátorem. Sobelův operátor provádí měření 2D gradientu obrazu v každém bodě. Na základě hodnot gradientů zvýrazňuje vysoké frekvence v obraze, které odpovídají hranám. Operátor se skládá ze dvou konvolučních jader, jedno pro odhad gradientu *Gx* ve vodorovném směru a druhé pro odhad gradientu *Gy* pro svislý směr.

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Obrázek 21: Konvoluční jádra Sobelova operátoru, převzato z [11]

Velikost absolutního gradientu obrazu je aproximována následujícím vztahem [7]:

$$|G| = |Gx| + |Gy|$$

Hodnoty gradientu jednotlivých pixelu pak tvoří obraz, který obsahuje hrany zvýrazněny bílou barvou. Filtr lze natavit také pouze pro detekci vodorovných nebo svislých hran pomocí parametru *ver* a *hor*.

V následující ukázce jsou v obraze detekovány vodorovné i svislé hrany (obrázek 22).

```
<filter id="sobel" hor="1"  ver="1"/>
```

Parametry:

- **ver**, **hor=** Příznaky pro detekci vodorovných hran a svislých hran. Nabývají hodnot 1 a 0.

Obrázek 22: Sobelův filtr:originál, detekce vodorovných i svislých hran, detekce pouze vodorovných hran

6.1.7 Cannyho hranový detektor

Cannyho hranový detektor je algoritmus, který zahrnuje několik kroků pro získání optimálních výsledků při detekci hran v obraze. Výsledkem detekce hran je binární obraz. Algoritmus zahrnuje následující kroky [12]:

1) **Odstranění šumu:** Obraz je vyhlazen Gaussovým filtrem s konvolučním jádrem o velikosti *kSize*.
2) **Určení gradientu:** Gradient obrazu je určen aplikací Sobelova operátoru.
3) **Ztenčeni hran:** Z hodnot gradientů jsou vybrány pouze lokální maxima. Tak je zajištěna detekce hrany pouze v místě největšího gradientu – přechodu.
4) **Prahování s hysterezí:** Prahováním hodnot gradientu je zajištěna eliminace nežádoucích hran. Prahy *tL* a *tH* určují minimální a maximální hodnotu, mezi kterými může hodnota gradientu kolísat. Prahování probíhá následovně: Každý pixel jehož hodnota gradientu je větší než *tH* je vybrán jako hranový. Potom, každý pixel, který je s tímto hranovým pixelem spojen a hodnota jeho gradientu je větší než *tL* je také označen jako hranový. Pokud je jeho hodnota menší jak *tL*, je hrana ukončena. Dochází tak k omezení vlivu šumu a hrany zůstávají souvislé.

Následuje ukázka detekce hran pomoci Cannyho algoritmu.

```
<filter id="canny" tL="50"  tH="200" kSize="3"/>
```

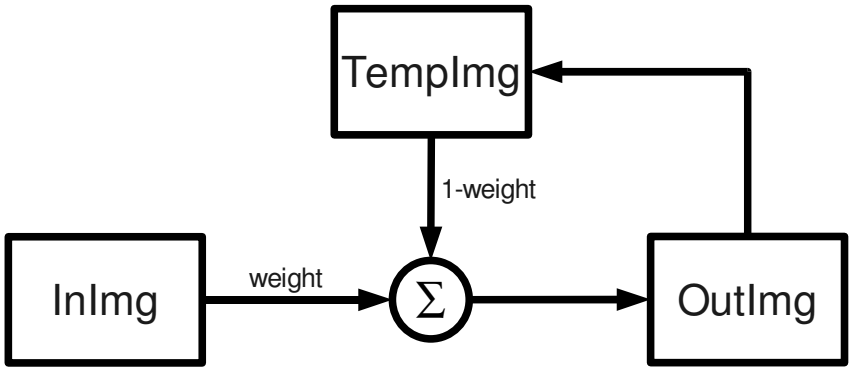
Parametry:

- **kSize=** Velikost konvolučního jádra Gaussova filtru. Nabývá hodnot 3, 5 a 7.
- **tH=** Horní práh gradientu. Optimální nastavení hodnot prahů záleží na daném kontextu. Obecně jsou dosaženy dobré výsledky při nastavení horního prahu na vysokou hodnotu a spodního prahu na nízkou hodnotu. Nastavením hodnoty horního prahu příliš nízko vede ke zvýšení detekce nežádoucích hran.
- **tL=** Spodní práh gradientu. Nastavením spodního prahu na příliš vysokou hodnotu vede k fragmentaci zašumělých hran.

Obrázek 23: Cannyho hranový detektor:
originál, detekce s tL=50 a tH=200, detekce s tL=100 a tH=250 (kSize=3)

6.1.8 Temporal smooth

Filtr Temporal smooth vyvolává efekt rozmazání pohybujících se objektů ve videu. Efektu rozmazání je dosaženo klouzavým průměrováním snímků [13]. Výstupní snímek *OutImg* (obrázek 24) je vypočítán jako vážený součet vstupního snímku *InImg* s váhou *weight* a průměrného snímku *TempImg* s váhou 1 – *weight*. Výstupní snímek vytváří nový průměrný snímek *TempImg*. Předchozí popis vystihuje následující schéma.



Obrázek 24: Schéma filtru Temporal smooth

Výpočet intenzity pixelu průměrného snímku pro jeden barevný kanál je popsán vztahem:

$$TempImg(x,y)_n = InImg(x,y)_n \cdot weight + TempImg(x,y)_{n-1} \cdot (1 - weight)$$

Míra rozmazání obrazu závisí na váze vstupního snímku *weight*. V následující ukázce má vstupní snímek váhu 0, 5. Na obrázku 25 je patrné, že míra rozmazání se zvyšuje, snižuje-li se váha vstupního snímku.

```
<filter id="tempSmooth" weight="0.5" start="0" end="130"/>
```

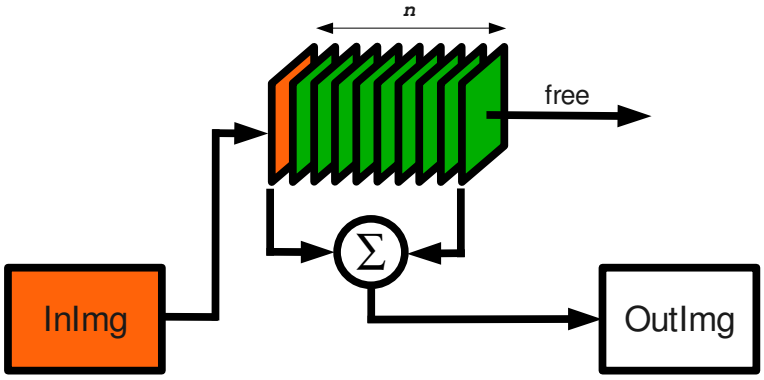
Parametry:

- **weight=** Váha vstupního snímku. Nabývá hodnot desetinných čísel z intervalu *<0;1>*.
- **start** a **end=** Vymezují interval snímků v sekvenci, na které bude filtr aplikován.

Obrázek 25: Temporal Smooth filtr:
originál, vstupní snímek s váhou 0.5, vstupní snímek s váhou 0.3

6.1.9 Average smooth

Filtr Average smooth rozmazává snímky na základě průměrování posledních *n* snímků. Vstupní snímek *InImg* (obrázek 26) je zařazen na konec fronty o *n* snímcích a první snímek je z fronty odstraněn (*free*). Výstupní snímek *OutImg* je vypočítán zprůměrováním všech snímků ve frontě. Ve výsledném snímku je pak pohybující se objekt zobrazen jako *n* siluet. To je způsobeno tím, že jsou jednotlivé snímky ve frontě průměrovány se stejnou váhou. Princip práce filtru ukazuje následující schéma.



Obrázek 26: Schéma filtru Average Smooth

Míru rozmazání obrazu určuje délka fronty *buf*. Interval snímku, na které bude filtr uplatněn určují parametry *start* a *end*. Účinek filtrace se projeví až při naplnění celé fronty, do té doby nejsou snímky filtrovány. Průměrovat snímky už při jejich vkládání do fronty by mohl být dalším způsobem řešení. V následující ukázce je obraz rozmazán průměrováním posledních pěti snímků.

```
<filter id="avrgSmooth" buf="5" start="5" end="130"/>
```

Obrázek 27: Average Smooth filtr:
originál, průměrování 5ti snímků, průměrování 10ti snímků

6.1.10 Doplnění titulků

Titulky jsou důležitou součástí video záznamu, zvláště takového, který neobsahuje zvuk. Videoeditor obsahuje velmi jednoduchý nástroj setText, který umožňuje umístit titulek na určité místo v obraze. Obsah titulku je nastaven parametrem *text.* Ten je umístěn na souřadnici (*pX, pY*) označující levý horní roh titulku. Barvu textu určují parametry *R, B, G*. Počátek souřadnicového systému tvoří levý horní roh obrazu.

V následující ukázce je titulek „Videoeditor“ umístěn do levého horního rohu obrazu.

```
<filter id="setText" text="Videoeditor" ptX="10" ptY="30" R="3" G="180" B="200"/>
```

6.2 Úprava rychlosti

Dalším důležitým nástrojem pro střih videa je úprava rychlosti přehrávání sekvencí. Změny rychlosti je dosaženo opakováním nebo naopak vynecháním některých snímků v sekvenci. V projetu je rychlost sekvencí upravena pomocí elementu *speed* a lze ji uplatnit na elementy *video* a *sequence*. Změna se udává v celých násobcích původní rychlosti sekvence.

Vizuálního efektu zrychlení je dosaženo vzorkováním každého *n*-tého snímku. Délka zrychlené sekvence na snímkové ose se pak *n*-krát zmenší. Při dělení délky je počet snímků zaokrouhlen nahoru. Míra zrychlení *n* je nastavena parametrem *faster.* Následuje příklad dvojnásobného zrychlení sekvence *Surfing*. Sekvence má pak délku 120 snímků.

```
<sequence id="Surfing">
  <video src="video/surfing.avi" in="0" out="240" pos="0"/>
  <speed faster="2"/>
</sequence>
```

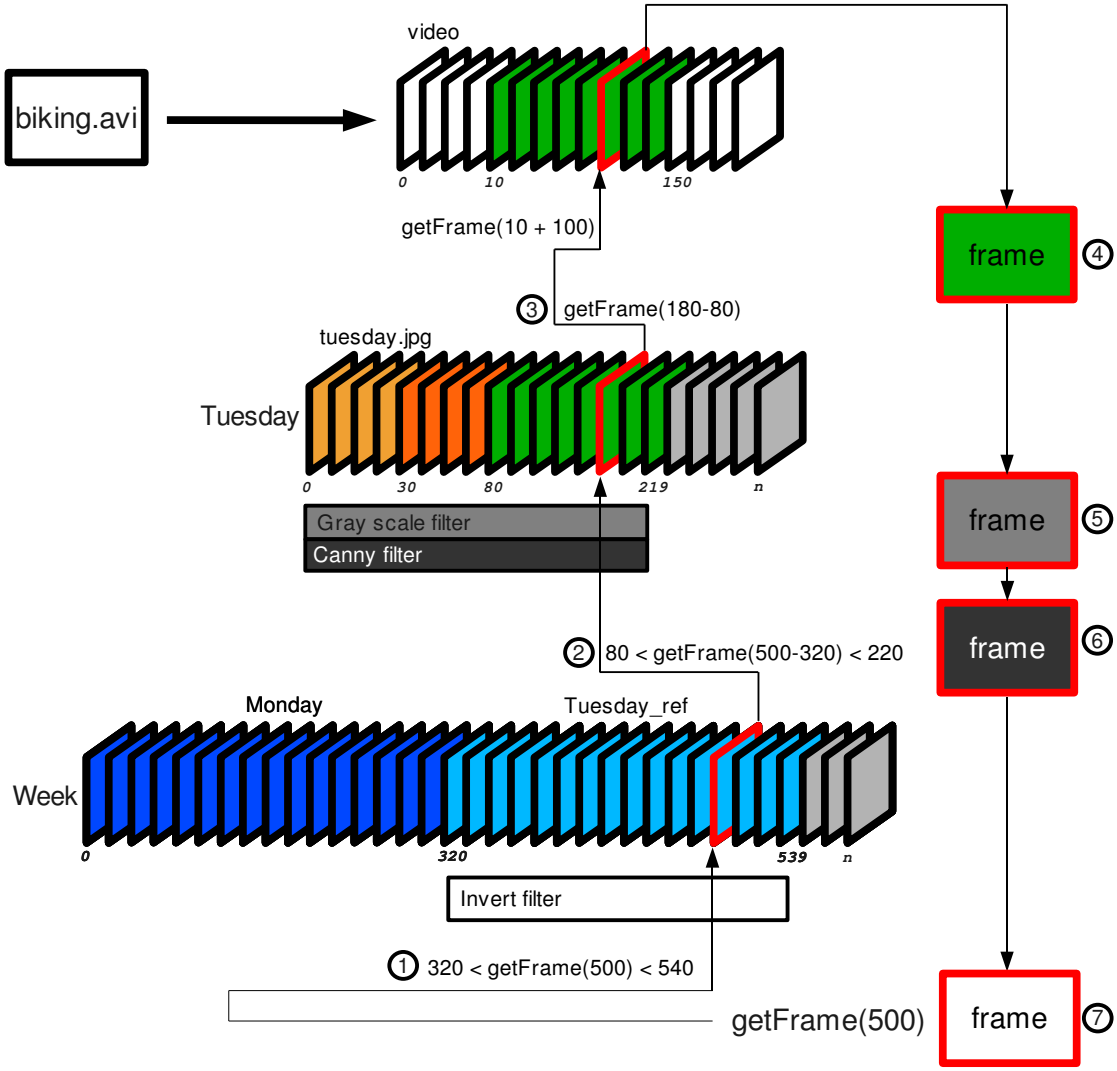
Pří zpomalení sekvence je každý snímek *n*-krát opakovaně vzorkován. Tím se délka sekvence na snímkové ose *n*-krát zvětší. Při velkém zpomalení dochází k porušení plynulosti přehrávání a pohyby se ve videu jeví jako nepřirozeně mechanické. To je způsobeno tím, že snímková frekvence nově příchozích snímků klesne pod 10 snímků za sekundu. Snímková frekvence 10 představuje hranici, při které člověk vnímá pohyb v obraze ještě jako plynulý. Používá se například v animacích [10]. Většina střihových programů umožňuje úpravu rychlosti i o necelé násobky původní rychlosti. K tomu je potřeba vypočítat hodnoty mezilehlých snímků pomocí interpolace. Metod interpolace je více – například lineární a geometrická. Tyto metody se liší zejména složitostí a efektivitou [5].

Míra zpomalení *n* je nastavena parametrem *slower*. V následující ukázce je zdrojové video nejdříve dvojnásobně zpomaleno a poté je celá sekvence *Surfing* dvojnásobně zrychlena. Výsledná rychlost sekvence je pak stejná jako původní rychlost videa (pozn. Ukázka slouží pouze pro demonstraci funkce).

```
<sequence id="Surfing">
  <video src="video/surfing.avi" in="0" out="240" pos="0">
    <speed slower="2"/>
  </video>
  <speed faster="2"/>
</sequence>
```

7 Přehrávání a zápis sekvence

Přehrávání spočívá v postupném zobrazení jednotlivých snímků sekvence. Následující obrázek demonstruje, jak je ze sekvence získán jeden snímek.



Obrázek 28: Získání snímku 500 ze sekvecne Week

Obrázek 28 obsahuje stejný scénář *Week* jako obrázek 15 v kapitole 5.3. V obrázku 28 je navíc doplněno zobrazení video souboru a na sekvence jsou nově aplikovány filtry (zápis situace na obrázku je uveden v projektu `getFrame.xml`).

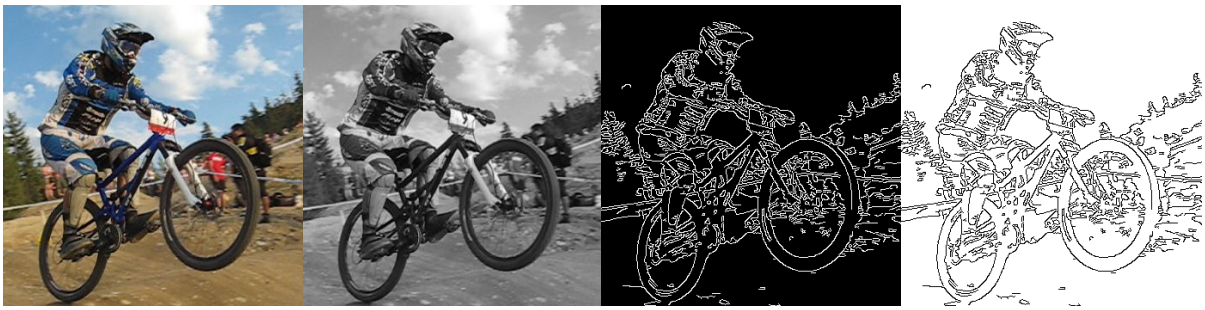
Princip přehrávání je vysvětlen na příkladu získání snímku 500 ze sekvence *Week*. Pro získání snímku sekvence slouží metoda `getFrame(index)`, jejímž parametrem je index snímku. Tato metoda je implementována pro všechny objekty nesoucí obraz. Jak již bylo zmíněno v kapitole 4.2, každá sekvence může mít na své snímkové ose umístěny objekty video, obrázek a další sekvence. Tyto objekty budou jednotně nazvány *zdrojové sekvence*. Například sekvence *Tuesday_ref* je jednou ze dvou zdrojových sekvencí sekvence *Week*. Zdrojová sekvence *Tuesday_ref* obsazuje snímkový interval 320 až 539 v sekvenci *Week*. Následuje popis postupu získání snímku 500. Jenotlivé body postupu jsou vyznačeny v obrázku 28.

(1) `getFrame(500)` prochází zdrojové sekvence v sekvenci *Week* a zjišťuje zda požadovaný snímek patří do jedné z těchto zdrojových sekvencí. Tedy zjišťuje, do kterého obsazeného intervalu snímek 500 náleží. V tomto případě byla podmínka (320 < 500 < 540) splněna pro zdrojovou sekvenci *Tuesday_ref*. To znamená, že se snímek nachází na pozici 500-320 v sekvenci *Tuesday*.
(2) `getFrame(180)` prochází zdrojové sekvence v sekvenci *Tuesday* a zjišťuje, do kterého obsazeného intervalu patří snímek 180. Ten patří do intervalu 80 až 220, který je obsazen už samotným zdrojovým videem, které bylo vystřiženo z video souboru `biking.avi`.
(3) `getFrame(100)` získá snímek 100 z videa. Snímek 100 ve vystřiženém videu odpovídá snímku 100+10 v načteném video souboru.

Zdrojový snímek je tedy nalezen pomocí rekurzivního hledání snímkového indexu ve zdrojových sekvencích. Pokud snímek nepatří do žádné ze zdrojových sekvencí, znamená to, že pozice v sekvenci není obsazena a je zobrazen černý snímek. Jakmile je snímek nalezen ve zdrojovém videu, je postupně filtrován filtry, které přísluší sekvencím, do kterých snímek patří. V tomto případě filtrace probíhá následovně:

(4) Barevný snímek získaný z video souboru.
(5) Převod barevného snímku na šedotónový.
(6) Binární obraz po aplikaci Cannyho hranového detektoru.
(7) Negativ binárního obrazu. Výsledný snímek obsahuje černé hrany na bílem pozadí.

Postup filtrace dokumentuje obrázek 29.



Obrázek 29: Filtrace snímku 500: bod 4, bod 5, bod 6, bod 7 -výsledný snímek

Princip zápisu sekvence je stejný jako u jejího přehrávání. Jediný rozdíl je v tom, že získaný snímek není zobrazen, ale je zapsán do video souboru. Nastavení, která určují vlastnosti exportované sekvence, závisejí na zvolených parametrech při spuštění programu.

Videoeditor umožňuje zapisovat videa do AVI kontejnerů s kompresí MJPEG. Při exportu lze nastavit rozlišení a snímkovou frekvenci výstupního videa. Při volbě těchto parametrů je třeba dbát na omezení maximálního datového toku, které vyplývá z kompresních omezení použitého kodeku. Kodek MJPEG byl vybrán jednak proto, že patří mezi doporučené v knize [2] a také proto, že má všechny snímky klíčové. Díky této vlastnosti je možné exportovaná videa ve střihači dále zpracovávat bez nutnosti jejich konverze do mnohonásobně objemnějšího RAW formátu.

8 Omezení a rozšíření

Videoeditor má určitá programová omezení, která vyplynula z implementace a návrhu. Tato kapitola na omezení upozorňuje a navrhuje možné způsoby jejich odstranění.

8.1 Grafické rozhraní

Videoeditor nemá grafické rozhraní pro jeho ovládání. Jedná se o konzolovou aplikaci. Cílem práce bylo vytvořit funkční stříhací jádro, nad kterým bude pracovat budoucí grafické rozhraní. Hlavním úkolem grafického rozhraní bude upravovat střihovou soupisku, která je obsahem projektu.

Jednou z možností jak toto rozhraní doplnit je implementovat jednoduché rozhraní pro zápis do XML dokumentu a postupně přidávat další základní grafické prvky jako je okno projektu, přehrávání, okno s časovou osou a další. Takovýchto grafických rozhraní existuje celá řada. Naskýtá se tedy možnost využít grafického rozhraní podobného open source projektu, jehož střihová soupiska by byla snadno převoditelná na XML soupisku Videoeditoru. Takovým programem by mohl být například video editor Cinelerra, který zapisuje střihovou soupisku EDL ve formátu XML [8].

8.2 Video stopy

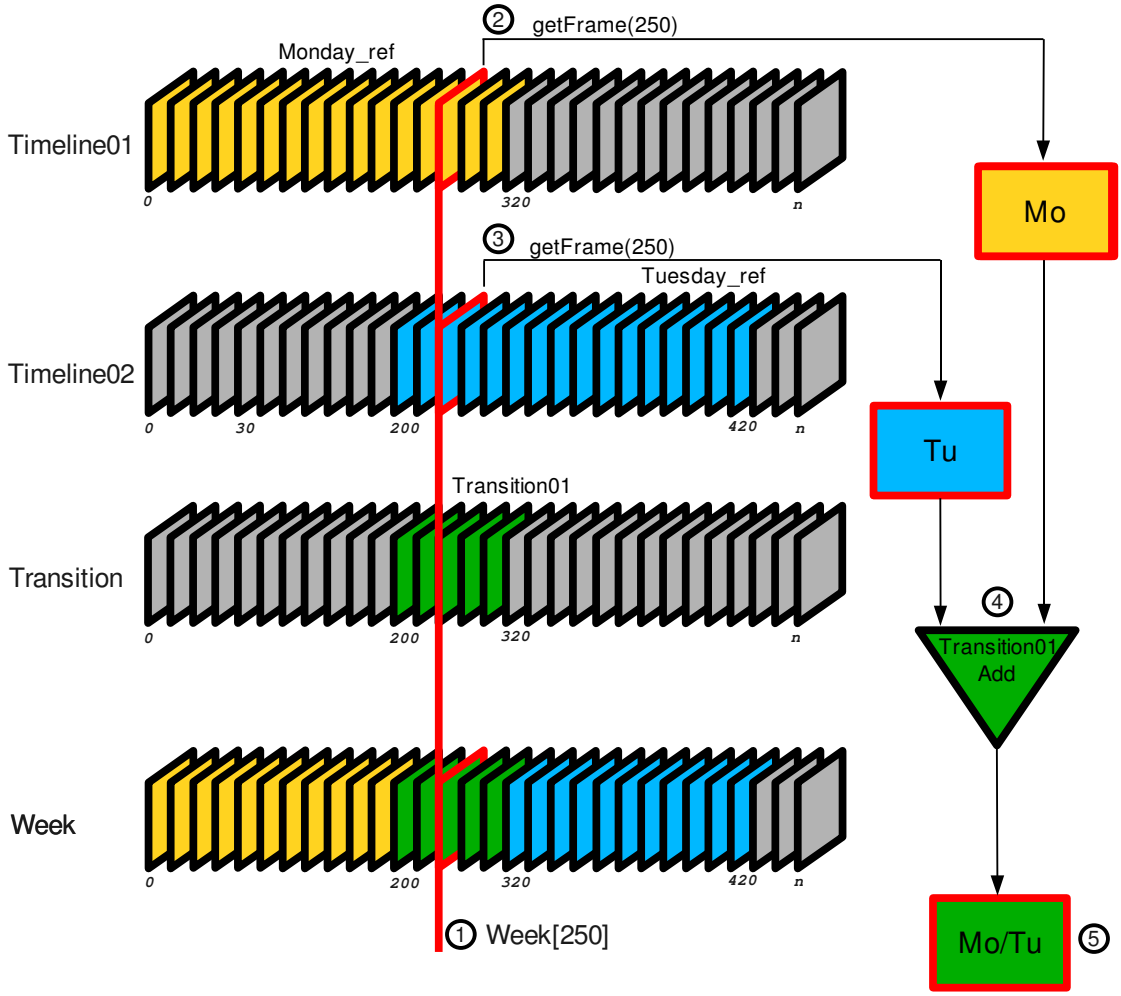
Videoeditor neumožňuje práci s více video stopami (dále jen stopa). Implementace vychází z předpokladu, že stříhaná sekvence má jedinou snímkovou osu a na tuto osu jsou řazeny nepřekrývající se zdrojové sekvence. Sekvence má tedy jedinou stopu, kterou tvoří lineární seznam zdrojových sekvencí. Jednomu snímku v sekvenci pak vždy odpovídá jediný zdrojový snímek.

V následujícím odstavci je navrženo, teoretické řešení práce s více stopami ve Videoeditoru. Zavedení více stop pro střih znamená zavést více snímkových os pro jednu sekvenci a doplnění stopy přechodů. Zavedení více stop ilustruje obrázek 30 na již známém scénáři *Week.*

Sekvence *Week* obsahuje dvě snímkové stopy (*Timeline1* a *Timeline2*) a stopu přechodů (*Transition*). Stopa přechodů definuje chování překrývajících se snímků na snímkových stopách. Důležité je uvědomit si, že indexy snímků na osách stop si vzájemně odpovídají.

Jeden snímek sekvence (*Week[250]* viz bod 1 v obrázku 30) je získán kombinací odpovídajících si snímků ležících na stopách (*Timeline01[250]* a *Timeline02[250]*). Jeden snímek stopy je získán popsanou metodou `getFrame(index)` (bod 2 a 3). Jakým způsobem jsou získané snímky zkombinovány určuje přechod (*Transition01)*. Ten si lze představit jako dvouvstupový filtr, který vstupní snímky například sečte (bod 4). Výsledkem přechodu je snímek *Mo/Tu,* který představuje v sekvenci *Week* snímek 250 (bod 5).

Pokud na stopě přechodů (*Transition*) není pro daný snímek definován žádný přechod, je zobrazen obsazený snímek ležící na stopě s vyšší prioritou. Nejvyšší prioritu má první snímková stopa sekvence (*Timeline01*). To znamená, že pokud není obsazen snímek na stopě *Timeline01* je zobrazen snímek ze stopy s nižší prioritou (*Timeline02*). Jestliže není snímek obsazen na žádné ze stop, je zobrazeno černé pole.



Obrázek 30: Práce s více stopami: Ukázka získání snímku 250 ze sekvence Week.

Následuje ukázka, jak by mohl vypadat zápis situace na obrázku 30. Ze zápisu je patrné, že element **sequence** by mohl zastupovat i roli stopy.

```
<sequence id="Week">
  <sequence id="Timeline01">
    <sequence id="Monday_ref" ref="Monday" pos="0"/>
  </sequence>
  <sequence id="Timeline02">
    <sequence id="Tuesday_ref" ref="Tuesday" pos="200"/>
  </sequence>
  <transition id="Transition01" type="Add" srcA="Timeline01"
     srcB="Timeline02" start="200" end="320"/>
</sequence>
```

Doplnit program o práci s více video stopami by nevyžadovalo mnoho změn v architektuře programu a přispělo by k významnému funkčnímu rozšíření.

8.3 Další omezení

Další omezení vyplývají z podstatné části z použitých implementačních prostředků. Aplikace má následující omezení pro:

Vstup

Videoeditor podporuje práci zejména s nekomprimovaným formátem videa. Nekomprimovaný formát je vhodný, protože umožňuje snadné načtení jednotlivých snímků (není potřeba dekódování). Snímky neobsahují chyby způsobené ztrátovou komprimací a dekomprimací. To je důležité například při detekci hran v obraze.

Natočený materiál bývá často komprimován ztrátovými kodeky jako je DV, MPEG a další. V takovém případě je nutné videa před zpracováním ve Videoeditoru převést do doporučeného formátu. Knihovna OpenCV [14] doporučuje pro uložení videa požívat kontejner AVI v kombinaci s kodeky: **RGB(A) ('DIB'), RAW I420 ('I420'), RAW I420 ('IYUV')**. Pro statický obraz jsou doporučeny formáty: **BMP, DIB, JPEG, JPG, JPE, PNG, PBM, PGM, PPM, SR, RAS, TIFF.** Převod video souboru na podporovaný formát je uveden v přiloženém skriptu `test.sh`.

Pokud je nekomprimované video příliš velké, je možné ho komprimovat. Při komprimaci však musí být kodek nastaven tak, že každý snímek videa bude klíčový. Jedině tak je možné, aby Videoeditor mohl zpracovávat jeho jednotlivé snímky. Vhodný kodekem je například MJPEG.

Aby bylo možné snímky jednotně zpracovávat je potřeba, aby veškerá vstupní obrazová data měla stejné rozlišení a barevnou hloubku 8 bitů na RGB kanál.

Dalším omezením je, že kódování XML dokumentu je nastaveno na `US-ASCII`, v projektu tedy nelze použít znaky s diakritikou.

Výstup

Použitím kodeku MJPEG je maximální datový tok výstupního videa omezen na 100 Mb/s. Maximální rozlišení je potom 800x600 při 25 snímcích za sekundu.

Dalším omezením je, že program neumožňuje práci se zvukovou stopou. Pro práci se zvukem by bylo možné použít například knihovnu FFmpeg [15].

9 Závěr

ílem této práce bylo navrhnout a implementovat jednoduchý video editor – systém pro zpracování a střih videa. Na úvod do problematiky, kterou se zabývá tato práce, bylo důležité zjistit, jak se techniky střihu vyvíjely a jakých principů se využívá dnes. Dále bylo nezbytné porovnat dostupné programy zpracovávající video a identifikovat jejich společné základy a rozdílné vlastnosti. Cílem práce bylo na základě těchto poznatků navrhnout a vytvořit program, který by měl umět video sekvence načíst, sestříhat, filtrovat, vložit na časovou osu, vkládat mezi ně jednoduché přechodové efekty a nakonec vytvořit výsledný video záznam v požadovaném formátu. Tyto požadavky se podařilo z velké části splnit.

Současná digitální technologie je logickým vyústěním předchozího technologického vývoje. Všechny dosavadní technologické změny ve vývoji kinematografie vedly ke zvýšení uživatelského komfortu a ke zjednodušení a zrychlení zpracování obrazu. Výhody digitálního videa oproti ostatním technologiím jsou mnohé, nejvýznamnější z nich však jsou: (1.) možnost vytvářet neomezené množství kopií bez ztráty kvality, (2.) zpracování pomocí počítačových programů umožňuje téměř jakoukoliv manipulaci s obrazem, (3.) snadná dostupnost a distribuce a další.

V rámci práce byl vytvořen program, který obsahuje základní paletu nástrojů pro střih a úpravy videa. Při tvorbě programu jsem usiloval o to, aby architektura programu byla co nejjednodušší a snadno rozšiřitelná. Program umožňuje video sekvence načíst ze souboru, upravit jejich délku, seřadit je do výsledné sekvence a tu nakonec přehrát či zapsat do souboru. Vytvořené sekvence lze doplnit efekty, které zahrnují filtraci snímků, doplnění titulků a změnu rychlosti přehrávání. Veškerá logika zpracování vstupních video souborů je obsažena v XML dokumentu. Ten je obdobou střihové soupisky EDL, kterou používají i další střihové aplikace. XML dokument by tak mohl být použit jako rozhraní pro spolupráci s dalšími aplikacemi.

Program neumožňuje práci s více video stopami, není tedy možné mezi záběry vkládat přechodové efekty. Neumožňuje zpracovávat zvukovou stopou a obsahuje pouze základní sadu filtrů.

Program je vhodný zejména pro hromadné zpracování video záznamů. Ten by mohl najít své uplatnění ve spolupráci s aplikacemi pro automatický střih, které se například používají pro zpracování dlouhých video záznamů z bezpečnostních kamer. Výstupem těchto aplikací je střihová soupiska, podle které by Videoeditor sestříhal výslednou sekvenci.

Možným pokračováním této práce je tvorba grafického rozhraní, které by přehledně zobrazovalo průběh editace. To by znamenalo vytvořit základní grafické prvky střihového programu jako je okno s časovou osou, přehrávací okno a projektové okno. Hlavním úkolem rozhraní by bylo zapisovat rozhodnutí uživatele do XML dokumentu. V rámci grafického rozhraní by bylo vhodné doplnit funkce pro střih metodou vložení a překrytí, jak bylo zmíněno v kapitole 5.

Další pole pro možné rozšíření práce představuje implementace filtrů. Stávající řešení jejich implementace je velmi jednoduché a poskytuje omezené možnosti nastavení. Například filtry pro úpravu jasu a kontrastu by bylo vhodné implementovat pomocí metody křivek, která dosahuje mnohem lepších výsledků (jak bylo zmíněno v kapitole 6). Vhodným řešením by bylo vytvořit samostatnou knihovnu pro tvorbu filtrů, která by obsahovala sadu komponent pro úpravu obrazu video sekvencí. Tak by bylo možné program snadno rozšiřovat o další filtry a efekty.

Významným rozšířením programu by byla práce s více video stopami. Stopy jsou důležité zejména pro tvorbu přechodových efektů. Součástí práce je návrh řešení zavedení více stop pro sekvence (kapitola 8). Toto řešení vychází z metody (*AB editing*), kterou dříve používal program Adobe Premiere. Tato metoda umožňuje vytvářet přechody mezi dvěma překrývajícími se záběry umístěnými v různých stopách.

Podrobný popis rozhraní aplikace, uvedený v programové dokumentaci, a návod pro použití programu včetně ukázkových příkladů je součástí přiloženého DVD. Obsah DVD je uveden v sekci Přílohy

Literatura

[1] Monaco, J.: *Jak číst film: Svět filmů, medií a multimédií.* Praha, Albatros, první vydání, 2004, ISBN 80-00-01410-6

[2] Bradski,G and Kaehler, A.: *Learning OpenCV: Computer Vision with the OpenCV Library*. Sebastopol, r O'Reilly Media, 2008, ISBN 978-0596-51613-0

[3] OpenCV library [online]. 21.1.2009 [cit. 20.4.2009]. Dostupný z WWW: <http://sourceforge.net/projects/opencvlibrary/>

[4] TinyXML [online]. 6.1.2007 [cit. 20.4.2009]. Dostupný z WWW: <http://sourozenectví/projects/tinyxml/>

[5] Bolante, A.: *Adobe Premiere 6.5: Názorný průvodce*. Praha, Computer Press, 2003, ISBN 80-7226-827-9

[6] Beránek, P.: *Digitální video v praxi*. Praha, Mobil Media a.s., druhé vydání, 2003, ISBN 80-86593-34-7

[7] Žára, J. et al: *Moderní počítačová grafika*. Brno, Computer Press, 2004, ISBN 80-86593-34-7

[8] Cinelerra linux video editing [online]. 21.3.2009 [cit. 30.3.2009]. Dostupný z WWW: <http://cinelerra.org/>

[9] Motl, D.: Zpracování videosekvencí. Diplomová práce, Brno, Fakulta elektrotechniky a informatiky VUT v Brně, 2001.

[10] Wikipedia: Persistence of vision [online]. 27.2.2009 [cit. 28.2.2009]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Persistence_of_vision>

[11] Fisher, R. et al: Hypermedia image processing reference HIPR2 [online]. 1.1.2004 [cit. 10.3.2009]. Dostupný z WWW: <http://homepages.inf.ed.ac.uk/rbf/HIPR2/wksheets.htm>

[12] Green, B.: Canny Edge Detection [online]. 1.1.2002 [cit. 19.3.2009]. Dostupný z WWW: <http://www.pages.drexel.edu/~weg22/can_tut.html>

[13] Wikipedia: Moving average [online]. 4.3.2009 [cit. 1.3.2009]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Moving_average>

[14] OpenCVwiki: Background information [online]. 4.11.2008 [cit. 26.12.2008]. Dostupný z WWW: <http://opencv.willowgarage.com/wiki/VideoCodecs>

[15] FFmpeg library [online]. 9.3.2009 [cit. 11.4.2009]. Dostupný z WWW: <http://www.ffmpeg.org/>

[16] Wood, J., Rocha, F.: Kdenlive [online]. 4.10. 2008 [cit. 23.4.2009]. Dostupný z WWW: <http://cs.wikibooks.org/wiki/Kdenlive>

[17] Kdenlive [online]. 15.4. 2009 [cit. 23.4.2009]. Dostupný z WWW: <http://kdenlive.org/homepage>

[18] Drábek, J.: Kdenlive – nelineární video editor [online]. 18.3. 2009 [cit. 23.4.2009]. Dostupný z WWW: <http://www.abclinuxu.cz/clanky/multimedia/kdenlive-1-uvod-instalace-prostredi>

[19] Lee, A.: VirtualDub [online]. 08.4. 2009 [cit. 24.4.2009]. Dostupný z WWW: <http://www.virtualdub.org/index.html>

[20] Wikipedia: Adobe Premiere Pro [online]. 8.3.2009 [cit. 22.3.2009]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Adobe_Premiere_Pro>

[21] Tomaides, P.: Digitální kamera na kliku [online]. 28.5.2008 [cit. 6.4.2009]. Dostupný z WWW: <http://www.ceskatelevize.cz/program/port/technosfera/302-digitalni-kamera-na-kliku/>

[22] Abbott, A.: Moviola story [online]. 6.4.2009 [cit. 1.5.2009]. Dostupný z WWW: <http://www.moviola.com/faq/70>

[23] Steenbeck editing tables [online]. 23.9.2008 [cit. 1.5.2009]. Dostupný z WWW: <http://www.steenbeck.com/3-3.html>

[24] Wikipedia: Linear video editing [online]. 6.4.2009 [cit. 20.4.2009].

Dostupný z WWW: <http://en.wikipedia.org/wiki/Linear_video_editing>
[25] JVC Pro equipment [online]. 23.4.2009 [cit. 1.5.2009].
Dostupný z WWW: <http://pro.jvc.com/pro/photos/default.htm>
[26] Tomaides, P.: Filmové triky digitálního věku [online]. 19.12.2007 [cit. 6.4.2009].
Dostupný z WWW:
<http://www.ceskatelevize.cz/program/port/tema/232-filmove-triky-digitalniho-veku/>
[27] Henninger media services: Telecine [online]. 2.5.2009 [cit. 3.5.2009].
Dostupný z WWW: <http://www.henninger.com/cine/transfer/>
[28] Video editing rentals: Avid Systems [online]. 28.4.2009 [cit. 3.5.2009].
Dostupný z WWW: <http://www.video-editing-rentals.com/avid-rentals.htm>
[29] Wikipedia: Comparison of video editing software [online]. 15.4.2009 [cit. 22.4.2009].
Dostupný z WWW: <http://en.wikipedia.org/wiki/Comparison_of_video_editing_software>

Přílohy

Přiložené DVD obsahuje:

- zdrojový kód programu (adresář: `videoeditor/src/`),
- programovou dokumentaci a návod k použití programu (`videoeditor/doc/index.html`),
- testovací skript (`videoeditor/test.sh`),
- ukázkové příklady (`videoeditor/projects/`),
- testovací data (`videoeditor/video/` a `videoeditor/image/`),
- ukázky výstupních video sekvencí (`videoeditor/examples/`),
- text práce v elektronické podobě (`text/EditorVidea.pdf`),
- prezentační plakát (`text/plakat.pdf`).