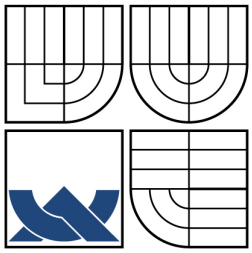
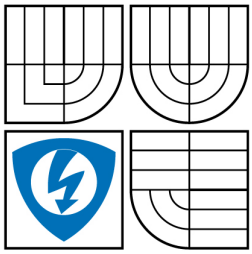


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

ODSTRANĚNÍ PROKLADU V NEKOMPRIMOVANÝCH DIGITÁLNÍCH VIDEOSEKVENCÍCH

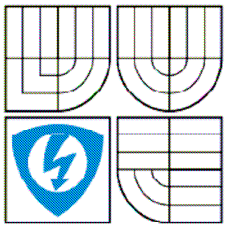
DEINTERLACING IN UNCOMPRESSED DIGITAL VIDEO SEQUENCES

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE — Bc. PAVEL MEŇHART
AUTHOR

VEDOUCÍ PRÁCE — Ing. MARTIN SLANINA, Ph.D.
SUPERVISOR

BRNO, 2009



VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav radioelektroniky

Diplomová práce

magisterský navazující studijní obor
Elektronika a sdělovací technika

Student:	Bc. Pavel Meňhart	***ID:***	46050
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Odstranění prokladu v nekomprimovaných digitálních videosekvencích

POKYNY PRO VYPRACOVÁNÍ:

Vypracujte rešerši o současně používaných formátech nekomprimovaných digitálních videosekvencí. Zaměřte se rovněž na možnosti vzájemného převodu těchto formátů. Dále nastudujte metody, které se používají k odstranění prokladu (deinterlacing) v digitálních videosekvencích.

Vytvořte program (doporučené prostředí MATLAB), který bude implementovat některé metody pro odstranění prokladu v digitálních videosekvencích.

Stanovte, která z použitých metod má subjektivně nejlepší výsledky. Tuto metodu pak aplikujte na reálné nekomprimované videosekvence různých formátů v jazyce C.

DOPORUČENÁ LITERATURA:

[1] Video Codecs and Pixel Formats [online]. Dostupné na: http://www.fourcc.org

[2] What is deinterlacing? The best method to deinterlace movies [online]. Dostupné na: http://www.100fps.com

[3] Recommendation ITU-R BT.601-6. Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios. International Telecommunication Union, 2007.

Termín zadání:	9.2.2009	***Termín odevzdání:***	29.5.2009

Vedoucí práce: Ing. Martin Slanina, Ph.D.

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

Abstrakt

Prokládání je způsob snímání a zobrazení snímku, kde se střídají sudé a liché řádky. Tento systém vznikl při tvorbě TV norem PAL a NTSC ve třicátých letech minulého století. Tímto principem se zamezilo blikání obrazu na starých televizních přijímačích. Nevýhodou takto zpracovaného obrazu je, že ho umí korektně zobrazovat pouze klasické televizní přijímače. Plasmové a LCD televize, projektory a počítačové monitory zobrazují vždy celý obraz (snímek) najednou, proto složí celý snímek z obou půlsnímků, které ale nejsou pořízeny v jednom čase, liší se v nich poloha pohybujících se objektů a jejich obrysy při zobrazení jsou dvojité, roztřepené a neostré. Tato diplomová práce se zabývá principem vzniku prokládaného videa, problémy při zobrazování takto prokládaného videa, metodami odstranění prokládání a na závěr jsou uvedeny programy, které jsem vytvořil v prostředí MATLAB a jazyce C a implementují některé z metod pro odstranění prokládání v digitálních sekvencích.

Abstract

Interlacing is the process of frame scanning and frame displaying being based on the even and uneven lines‘ interchanging. This framework originated along with the PAL and NTSC television standards‘ formation in the course of 1930s. The process mentioned above stopped frames displayed on older TV sets from flashing. Nevertheless, there is a disadvantage of the frame processed after this manner, consisting in its attribute to be displayed in a clean-cut way just on classical TV sets. However, plasma and LCD television appliances, projectors and computer monitors always display the entire frame all at once, therefore composing it of the both fields, yet not being acquired concurrently, differing in moving objects‘ positions, moreover their contours are doubled, frayed and unfocussed. This master´s thesis deals with fundamentals of the interlaced video signal‘s formation, problems coming about at featuring the interlaced video like this, followed by operating methods of deinterlacing itself. In conclusion, there are my own softwares presented, having been developed in MATLAB computer environment and software command language C, and implementing some of the operating methods designed for deinterlacing in uncompressed digital video sequences.

Klíčová slova

prokládání, odstranění prokládání, půlsnímek, vzorkování, nekomprimovaná digitální videosekvence, YUV, RGB, Bob, Weave, Blending Field, interpolace

Keywords

interlacing, deinterlacing, field, sampling, uncompressed digital video sequence, YUV, RGB, Bob, Weave, Blending Field, interpolation

Bibliografická citace

MEŇHART, P. *Odstranění prokladu v nekomprimovaných digitálních videosekvencích*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 40 s. Vedoucí semestrální práce Ing. Martin Slanina, Ph.D.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma ODSTRANĚNÍ PROKLADU V NEKOMPRIMOVANÝCH DIGITÁLNÍCH VIDEOSEKVENCÍCH jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 29. května 2009 .....

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Martinu Slaninovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne 29. května 2009 .....

podpis autora

OBSAH:

1. Úvod

S prudkým rozvojem techniky nepřichází pouze vylepšení stávajících technologií a jejich využití v odborném a profesionálním životě, ale současně s tím dochází k využití poznatků i v odvětvích techniky komerční, zábavné a domácí. Pryč jsou doby, kdy jediná možnost, jak zaznamenat obraz, bylo jeho natočení filmovou kamerou, kdy potom složité metody chemického zpracování filmového pásu a jeho omezená odolnost a životnost přinášely problémy s jeho uchováním a archivací. S objevem televizního přenosu se rozšířily nové metody záznamu tohoto obrazu. Těchto metod sice existuje velká řada, ale přesto většího rozšíření dosáhly dvě formy záznamu analogových či digitálních obrazových signálů a to záznam magnetický a optický záznam do pevného media.

Při dnešním poklesu cen a rozšíření technologií dochází pouze k jednomu problému – kompatibilitě jednotlivých způsobů snímání, záznamu a zobrazení obrazu. Můžeme se tedy potkat s problémem jak zaznamenat prokládaný televizní obraz, jak tento obraz zobrazit na moderních zobrazovacích jednotkách (LCD či PLD), jak obraz zaznamenat z hlediska jeho velkých požadavků na kapacitu záznamového media, jak převést obraz z filmového pásu do digitální podoby a jak dosáhnout vždy nejlepší kvality obrazu.

V této práci bych se chtěl věnovat převážně možnostem nekomprimovaného záznamu obrazu, který umožňuje při zpětném vyvolání zobrazit video v původní kvalitě, formátům tohoto nekomprimovaného videa a možnostem jejich vzájemného převodu. V další části diplomové práce budou objasněny problémy vzniklé prokládáním televizního obrazu v souvislosti s moderními principy zobrazování, a nalezeny metody, jak zkreslení vzniklé prokládáním odstranit.

Nejdůležitější částí této práce jsou kapitoly, věnující se praktickému implementování metod na odstranění prokládání v nekomprimovaných videosekvencích, vytvořené buď jako m.files v prostředí MATLAB, nebo jako zdrojové kódy v jazyce C. Dosažené výsledky vybraných metod jsou dále porovnány z hlediska účinnosti odstranění prokladu.

1.1. Historie záznamu a snímání obrazu

S troškou nadsázky lze říci, že „na začátku bylo světlo“. Světlo, jehož odraz od všech pozorovaných předmětů dopadá na sítnici oka a všechny informace, které umíme z tohoto odrazu získat o jasu, sytosti, hloubce, barevnosti, ostrosti atd. pozorovaného obrazu, záleží na schopnostech lidského oka. A právě některých nedokonalostí lidského oka se začalo využívat v okamžiku, kdy nastala potřeba obraz zaznamenat a znovu opakovaně reprodukovat.

Pravý začátek tohoto snímání a zaznamenávání obrazu podle [7] je spojen s vynálezem fotocitlivého materiálu, jehož vlastnosti se s osvětlením nevratně mění. Tento materiál se nanášel na celuloidové filmy a procházel složitým chemickým procesem vyvolávání a ustalování záznamu. Tento princip se do dnešní doby drží hlavně u filmů pro kina. Důvodem je hlavně vysoké rozlišení těchto filmů a jejich možná reprodukce na plátna kin velkých rozměrů - až několik metrů. Toto rozlišení je určeno hlavně citlivostí filmu (u fotoaparátů označované jako ISO100, ISO200 a ISO400), kdy se zvětšuje zrno filmu, aby pojalo více světla nebo umožnilo snímání za menšího osvětlení. Když se ale zvětší zrno, klesá rozlišení a navíc nastává problém s určením tohoto rozlišení, protože zrna nemají pravidelný čtvercový tvar a jsou rozmístěna vlastně náhodou. Kdybychom tento záznam potřebovali digitalizovat, musíme film naskenovat v určitém rozlišení. A teď se dostáváme k prvnímu jmenovanému využití nectností lidského oka – není potřeba skenovat v příliš vysokém rozlišení, protože potom by dva sousední body měly stejnou hodnotu. Navíc je otázkou, jak velké rozlišení je lidské oko vlastně schopno vnímat. Všeobecně uznávané meze jsou přibližně 5000x5000 bodů, některá literatura hovoří o 3000x3000 a často (hlavně

při renderování digitálních animací a efektů ve filmech) se mluví o 4096x4096 (označovaných jako 4K).

Dalším důležitým parametrem filmu je počet snímků za vteřinu. Těch se používá 24 za vteřinu, kde zase využíváme další nectnost oka a to jeho setrvačnost. Požadavkem je totiž plynulost pohybu a při 24 snímcích za sekundu se obraz ještě zdá trhaný. Proto při promítání filmu použijeme trik s trojnásobným zobrazením jednoho snímku přerušovaně, takže oko vnímá jakoby by měl obraz trojnásobnou frekvenci. Technicky toho dosahujeme použitím tzv. „Maltézského kříže“, který třikrát zacloní konkrétní snímek, a teprve při čtvrtém zaclonění se posune filmový pás na další políčko. Každopádně se vždy ale jedná o ***systém neprokládaný***.

1.2. Historie televizního přenosu

Rozvoj se však nedal zastavit a tak byla podle [5] v roce 1923 patentována ruským vědcem Zworykinem elektronka pro snímání obrazu - iconoskop, s první plně fotoelektrickou mozaikou. Tím se položil základ, na němž se již začalo stavět televizní vysílání. Proto v roce 1928 mohl v zámoří proběhnout zkušební provoz prvního televizního vysílání s rozlišením 30 řádků a o rok později už bylo rozlišení 60 řádků. V roce 1930 došlo i na objev principu prokládání obrazu, kterým se odstranilo blikání obrazu. Technologie se vyvíjela až do roku 1941, kdy se ustálil standard na 525 řádků.

V Evropě byla situace obdobná. V roce 1929 angličan John L. Baird zavedl pokusné vysílání s 30-ti řádky na střední vlně 261,5 metru. V roce 1934 dosáhl počet řádků již 405, a tímto počtem se mohli pochlubit i Paříž, kde podporu pro vysílání zajišťoval Marconi a komerční vysílání zde začalo 1937. V tehdejším Československu se již v roce 1935 podařilo sestrojit fungující televizní zařízení (Dr. Jaroslav Šafránek). Po malé odmlce během druhé světové války, se výzkum rozjel naplno a s využitím poznatků z rozvoje vysílací techniky během války byl v roce 1952 ustaven standard pro vysílání v Evropě na 625 řádek a 50 půlsnímků za sekundu.

1.3. Problémy způsobené prokládáním

A právě předchozí poznámka o půlsnímcích dostává v současné době úplně jiný rozměr a je to základní problém této diplomové práce. Princip televize totiž vychází z vysílání a snímání obrazu po půlsnímcích (field), z nichž dva tvoří vždy jeden snímek (frame). Jeden půlsnímek je tvořen lichými řádky, druhý půlsnímek obsahuje sudé řádky a oba půlsnímky jsou získány v jiném časovém intervalu. Protože jsou takto půlsnímky proti sobě posunuty v čase, zobrazí je správně pouze televizní přijímač. Je to dáno způsobem snímání po půlsnímcích, jemuž odpovídá i způsob zobrazování klasickou vakuovou televizní obrazovkou, kde opět elektronový paprsek vychylovaný magnetickým polem vykreslí lichý půlsnímek, potom se vrátí do výchozího místa o řádek níž a vykreslí sudý půlsnímek mezi řádky lichého snímku. V době kreslení sudého půlsnímku již luminofory lichého půlsnímku pomalu zhasínají, což ale naše oko nezaznamená právě z důvodu jeho setrvačnosti. A také se vzápětí začne vykreslovat opět lichý půlsnímek, takže se obnoví případně obmění svit bodů lichého řádku. Pokud bude frekvence vykreslování dostatečně vysoká, nebude lidské oko vnímat obraz jako nepříjemné blikání. Samozřejmě se před vznikem metody prokládání přemýšlelo, jak obraz zpracovat jako celek, ale to nebylo možné z technických důvodů – potřeba velké šířky pásma apod.

Pokud tedy zobrazuje prokládaný obraz klasická vakuová televizní obrazovka, je vše v pořádku. Pokud ale zobrazíme oba půlsnímky najednou, což provádí hlavně monitor počítače (ale stejná situace nastane i v případě potřeby obraz digitálně zaznamenat),

dostaneme obraz neostrý, roztřepený u pohybujících se objektů. Navíc u snímání digitální kamerou a následné digitalizace videa není pro prokládané řádkování důvod a snímky se ukládají celé.

Proto jediný problém nastane v okamžiku, kdy chceme převést televizní obraz rozložený na půlsnímky na obraz, kde se ukládají celé snímky. Nezbývá, než odstranit proklad obrazu.

2. Základní pojmy

2.1. Digitalizace videa

Obrazové signály jsou prakticky vždy vytvářeny obrazovými snímači v analogové formě. Přechod od tohoto analogového signálu k signálu digitálnímu je možný několika způsoby, ale vždy způsoby celosvětově standardizovanými doporučeními Mezinárodní telekomunikační unie ITU (International Telecommunication Union) označovanými kódem R 601 (R 656, R 657 …). Tato doporučení nabyla nového významu se současným zaváděním digitálního televizního vysílání a přenosu DVB, a také s rozvojem metod a algoritmů pro kompresi digitálních obrazových dat.

Základní parametry podle doporučení ITU R 601, uvedenými v [3], jsou:

- vzorkovací kmitočet pro luminanční signál $U_Y(Y)$: $f_Y = 13{,}5MHz$

- vzorkovací kmitočet pro chrominanční složky C_R, C_B: $f_{CR} = f_{CB} = 13{,}5MHz$

- pro TV normy 625/50 na řádku 864 vzorků, na aktivním řádku 720 vzorků luminančního signálu
- pro TV normy 525/60 na řádku 858 vzorků, na aktivním řádku 720 vzorků luminančního signálu
- vzorky luminančních a chrominančních signálů se vzorkují na 256 úrovní, tj. 8bitů,

Pokud vezmeme v úvahu právě výše uvedené parametry, dostaneme pro záznam snímku nekomprimovaného videa velký počet vzorků:

$$p_{vz} = 625 \cdot 864 \cdot 3 = 1620000 vzorků \qquad \textbf{(1)}$$

Při výpočtu potřebné kapacity paměti pro jeden snímek zjistíme:

$$C_{4:4:4} = 1620000 \cdot 8 = 12960000b = 1620000B = 1582MB = \underline{\underline{1{,}55GB}} \qquad \textbf{(2)}$$

Je tedy zřejmé, že uložit tímto způsobem celý film o délce trvání 1,5 hod, klade vysoké nároky na přenosové rychlosti všech zařízení a na obrovskou kapacitu uložiště dat. Proto se velmi často používá různých způsobů komprese, kterou můžeme jednoduše vysvětlit jako potlačení „nadbytečné“ informace a lze je rozdělit na dvě skupiny – kompresi bezeztrátovou a ztrátovou. Bezeztrátová komprese je všeobecně použitelná metoda, která pracuje se statistickým rozložením dat (seskupování dat na základě jejich entropie) a potlačení informační nadbytečnosti je zcela reverzibilní proces. Při zpětném procesu, tj. při aplikaci dekódovacího schématu dostaneme původní obraz beze změny. Komprese ztrátová (která má smysl jen v případě obrazových, příp. zvukových signálů) pracuje s nevratným procesem, který je založen na prostorové podobnosti obrazových dat. Abychom takovou podobnost odhalili, je nutné využít transformace, která převede prostorový obrazový signál do kmitočtové domény (DCT), nebo do podoby tzv. dekompozičního obrazce (DWT). Takto

získaná data zakódujeme pomocí určitého schématu do podoby určené k přenosu - zde dochází ke ztrátě informace, tzn., že po zpětném procesu nedostaneme původní obraz, ale pouze obraz před kódováním.

2.2. Vzorkování a rozklad obrazu na složky

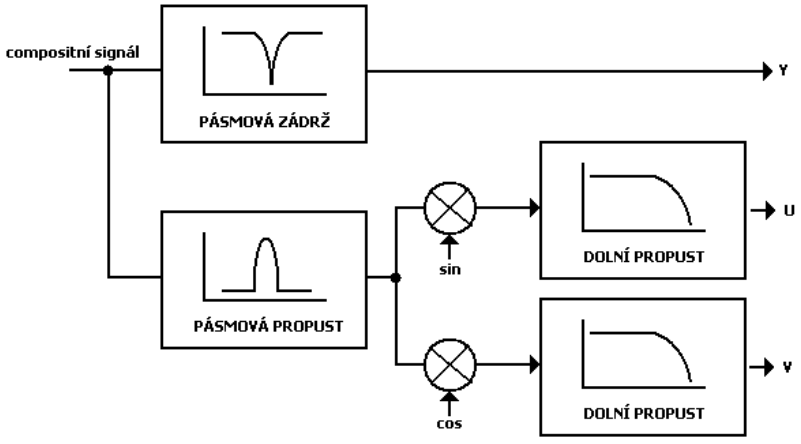
Tyto formáty vznikly podle [6] jako hlavní možnost uložení obrazu ekvivalentního analogovému formátu. Vychází tedy z potřeb uložit obraz konkrétní normy. Pokud vysvětlíme princip uložení např. na normě PAL, není u ostatních norem jiná změna, než pouhá odlišnost v počtu řádků a snímků za sekundu. PAL (a SECAM) má 576 viditelných řádek a 25 snímků/s a NTSC 480 řádků a 29.97 snímků/s. Odlišnost, která je už citelnější, je způsob kódování barev v analogové verzi. Černobílý základ obrazu je vždy stejný, protože je tvořen jasovou složkou. Barevná složka je pak namodulována s různou šířkou pásma, takže rozeznáváme formáty vzorkování s označením 4:4:4, 4:2:2 a 4:2:0. Pro představu o bitové rychlosti a vzorkovacích frekvencích jednotlivých formátů, jsou seřazeny jednotlivé položky do přehledné Tab.1 podle [8]:

Tab.1: Přehled formátů lineárního vzorkování PCM

Formát	**4:4:4**		**4:2:2**		**4:2:0**	
Parametr	Bitová rychlost [Mb/s]	Vzorkovací kmitočet [MHz]	Bitová rychlost [Mb/s]	Vzorkovací kmitočet [MHz]	Bitová rychlost [Mb/s]	Vzorkovací kmitočet [MHz]
Signál Y	108	13,5	108	13,50	108	13,500
Signál C_B	108	13,5	54	6,75	27	3,375
Signál C_R	108	13,5	54	6,75	27	3,375
Celkem	324		216		162	

Pro zpracování videa se nejčastěji bere v úvahu formát 4:2:2, tzn., že se barevný signál vzorkuje s dvakrát nižší šířkou pásma a má tedy dvakrát nižší rozlišení než jasová složka. Projevem tohoto způsobu vzorkování jsou „barevné duhy“ v obrazu pro velké kontrasty, při střídajících se svislých pruzích v obraze je pak vidět barevné moiré.

Základní principy zpracování videa v PC vychází z tohoto principu podle [6]:



Obr.1: Princip zpracování kompositního signálu před A/D převodem

Jak je vidět z Obr.1, před převodem signálu do číslicové formy, se nejprve oddělí jasová složka (Y) od barevné a barevná složka se rozdělí na dvě části U a V. Potom následuje převod pomocí rychlých A/D převodníků, přičemž barevné složky se samplují poloviční frekvencí než jasová. Často se ještě využívá metody oversampling - tedy vzorkování vyšší frekvencí než potřebnou a následná digitální úprava do požadované velikosti. Ve výsledku pak máme pro jeden frame 576 řádků po 720 jasových a 360 barevných bodech, což odpovídá rozlišení normy PAL. Je možné použít i horizontální rozlišení 768 bodů, aby byla zachována velikost poměru stran jednoho bodu 1:1.

Tímto způsobem se tedy získají tři signály – složka Y (jasová) a složky U, V (barevné). Tomuto formátu se tedy říká YUV. Existuje spoustu možností, jak uložit tyto složky do paměti - po bodech (YUVYUVYUV...), po řádcích, po plochách atd. Takto digitalizovaný signál je ale pro další zpracování méně vhodný - nelze jej jednoduše editovat (sčítat, průměrovat apod.). Převádí se proto do formátu RGB, kdy se přepočítá jasová a barevné složky na známou trojici signálů - červená R, zelená G a modrá B. Celý proces převodu z formátu YUV do RGB a zpět se provádí digitálně, čímž se dosáhne potřebné rychlosti a kvality.

3. Formáty nekomprimovaných digitálních videosekvencí

3.1. Formáty YUV

Formáty YUV lze podle [1] rozdělit do dvou skupin - packed a planar, podle uložení jasové a barevné složky v paměti. Formáty packed ukládají všechny složky YUV do tzv. makropixelů (skupina několika pixelů, např. 4), které následují po sobě. Planar formáty ukládají všechny složky zvlášť, čímž vzniknou tři virtuální plochy, které jsou ve výsledku složeny dohromady.

Formáty YUV se dále označují kombinací tří čísel, např. YUV 4:2:2. Tímto značením se udává poměr mezi počtem vzorků barevné složky vůči složce jasové. V uvedeném případě je poměr 4:2, tzn., že barevná složka obsahuje polovinu bodů vůči jasové – dvěma bodům jasovým odpovídají dva body barevné. Analogicky se užívá označení např. pro formát YUV 4:1:1, což znamená, že obsahuje čtvrtinu vzorků barevné složky oproti složce jasové. V příkladu výpočtu (vztahy (1) a (2)) je uvažován formát YUV 4:4:4, který má vzorkovánu jasovou a barevnou složku rovnocenně, předpokládá se tedy, že by tento formát měl být nejkvalitnější. Často se však tento formát získává přepočítáním formátu YUV 4:2:2 a tudíž žádnou novou informaci o obraze nepřináší.

Pro uložení do paměti, je každému způsobu vzorkování – každému formátu - přiřazen identifikační znak, tzv. FourCC - Four Character Code. Pro číslicové vyjádření tohoto kódu, je vyhrazen prostor 4B (32 bitové slovo), kde je pomocí ASCII tabulky vyjádřen název zkratky formátu kódování YUV. Výběr ze seznamu používaných FourCC kódů podle informačního zdroje [2] , je uveden v Tab.2 a Tab.3:.

Tab.2: Výběr PACKED FOURCC formátů

Název kódu FourCC	Hex vyjádření kódu	Použité kódování	b/pixel	Poznámka
YUY2	32595559	4:2:2	16	Jde o nejpoužívanější formát, spolu s UYVY. Jeden makropixel se skládá dvou pixelů v jednom 32bitovém slově - 2xY a 1xUV
UYVY	59565955	4:2:2	16	Stejný jako YUY2, má jiný sled YUV složek.
UYNV	564E5955	4:2:2	16	Vychází z UYVY, písmeno N značí registraci formátu firmou NVidia
YUNV	564E5559	4:2:2	16	Opět formát registrovaný firmou NVidia, jde o přejmenovaný UYVY
Cyuv	76757963	4:2:2	16	Podobný UYVY, pouze opačné řazení řádků - spodní řádek nejdříve
Y41P	50313459	4:1:1	12	Makropixel obsahuje 8 pixelů ve třech 32-bitových slovech (někdy nazýván BTYUV)
Y41T	54313459	4:1:1	12	Odvozen od Y41P, pouze T značí průhlednost nejnižšího bitu složky Y
Y42T	54323459	4:2:2	16	Stejný jako UYVY, T opět značí transparentnost nejnižší bitu složky Y
V422	32323456	4:2:2	16	Stejný jako YUY2
CLJR	524A4C43	4:1:1	8	Makropixel obsahuje čtyři pixely ve 32-bitovém slově. Y složky mají 5 bitů a UV složky jsou 6-ti bitové.
YUVP	50565559	4:2:2	24	Jako YUY2, liší se větším počtem bitů na pixel.
UYVP	50565955	4:2:2	24	Jako UYVY, opět vyšší počet bitů na pixel.
Y211	31313259	2:1:1	8	Speciální vzorkování – Y polovičním kmitočtem, U a V čtvrtinovým
Y800	30303859	8:0:0	8	Řešení pro monochromatický obraz - obsahuje pouze Y

A nyní výběr planárních formátů:

Tab.3: Výběr PLANAR FOURCC formátů

Název kódu FourCC	Hex vyjádření kódu	Použité kódování	b/pixel	Poznámka
YV12	32315659	4:1:1	12	Celý obraz je vzorkován 8-mi bity (Y) a potom 2x2 bity U a V (v horizontálním i vertikálním směru). Využití u MPEG formátů
YVU9	39555659	8:1:1	9	Složka Y nejprve jako 8-mi bitový vzorek, potom 4x4 bity subsamplované U a V
I420	30323449	4:1:1	12	Opět Y složka nejprve 8-mi bitový vzorek, následovaný 2x2 bity subsamplované U a V
IYUV	56555949	4:1:1	12	Shodný s I420
CLPL	4C504C43	4:1:1	12	Registrovaný firmou Cirrus Logic, podobný YV12
IF09	39304649	8:1:1	9.5	8-mi bitový Y vzorek následovaný 4x4 bity subsamplovanými U a V s delta kódováním (sleduje rozdíl vůči předchozímu pixelu)

Jak je z Tab.1 a Tab.2 patrné, nejčastějšími a také nejpoužívanějšími formáty jsou packed formáty s kódováním YUV 4:2:2, zejména YUY2 a UYVY. Proto je jejich použití vhodné pro hardwarový overlay na grafických kartách, kdy je v buffer paměti grafické karty uložena informace o definici obrazu, která je dále hardwarově zpracována pro zobrazení. Každá změna nastavení (změna velikosti, jasu, obtékání, transparence apod.), se tak vykonává přímo zde a ulehčuje se tak procesoru při zobrazování. Pro overlay buffery se nepoužívají RGB formáty.

3.2. Formáty RGB

Jak bylo již dříve řečeno, formáty RGB podle [1] umožňují snadnější práci s obrazem, jeho editaci, která se v číslicové formě vždy převádí na matematické operace (sčítání, průměrování atd.). Signály RGB obsahují (a jsou tak i značeny) tři základní barevné složky - červenou, zelenou a modrou - které vždy tvoří jeden pixel. Bezdůvodné je zde subsamplování barevné složky, neboť tato zde není od barevných složek oddělena. Dělení a značení jednotlivých formátů závisí prakticky jen na počtu bitů na pixel. Některé používané FourCC kódy pro RGB formáty ukazuje Tab.4:

Tab.4: Výběr FOURCC kódů pro RGB formáty

Název kódu FourCC	**Hex vyjádření kódu**	**b/pixel**	**Poznámka**
BI_RGB	00000000	1,4,8,16,24,32	Základní formát používaný pro kódování BMP souborů a videa. Počet bitů na pixel se doplňuje za označení (např. RGB4, RGB8, atd.)
RGB	32424752	1,4,8,16,24,32	Stejné jako BI_RGB
BI_RLE8	00000001	8	Komprimace obrazu metodou Run Length Encoding s možností skoku mezi pixely. Používá se pouze u statických snímků (BMP)
RLE8	38454C52	8	Stejné jako BI_RLE8
BI_RLE4	00000002	4	Stejný princip jako u BI_RLE8, pouze menší počet b/pixel
BI_BITFIELDS	00000003	16,24,32	Vychází z RGB, má libovolný sled pixelů a libovolné kódováním uvnitř pixelu
RGBA	41424752	16,32	Opět stejné jako RGB, objevuje se navíc alpha kanál a stejné kódování jako u BI_BITFIELDS
RGBT	54424752	16,32	Stejné jako RGBA, písmeno T značí kanál určující průhlednost (transparency)

3.3. Vztahy mezi formáty

Formátů YUV je více, u RGB již není tolik možností, jakým způsobem uložit obraz. U některých RGB formátů lze použít určitou míru komprese, která při složitých výpočtech již nemá citelný vliv na ztrátu kvality, kdežto u YUV dochází při redukci k odstranění části informace.

U formátu RGB je jeden bod obrazu zastoupen třemi barevnými složkami (R - červená, G - zelená a B - modrá), u kterých se udává kvantizační stupeň, tedy počet bitů každé barevné složky. Např. u RGB24 je každé barvě vyhrazen jeden byte, což dohromady

dává 3x8=24 bitů. Podobně u RGB15 je to 3x5=15 bitů, pro RGB16 (5+6+5) atd. Pouze u RGB32 je rozdělení kvantizačních stupňů stejné jako RGB24, s tím rozdílem, že každý bod je zarovnán na 4 byty kvůli lepšímu zpracování v dnešních 32bitových počítačích. Jeden byte je tedy nevyužit vůbec, nebo použit na průhlednost - alpha. RGB se používá pro filtraci, efekty, zpracování a střih.

S ohledem na potřebnou kapacitu paměťového místa při formátování pomocí RGB se používá spořivější kódování YUV (Yellow Under Violet), které vychází přímo z analogového video signálu. Jeho přímá návaznost se projevuje v absenci kódování pomocí červené, zelené a modré, ale kódování pomocí jasové složky a dvou složek pro definici barvy, mnohdy společných i pro více bodů najednou. Výhodou je jejich značná rychlost zpracování, hodí se proto především pro následnou kompresi.

Pro využití výhodných vlastností obou formátů, je lze mezi sebou libovolně převádět, ovšem ne vždy bezeztrátově. Rozdíly v počtu bitů na pixel (určující počet dostupných barev) a typ kódování (poměr barvy k jasu) se projeví na ztrátě dat.

Převod z RGB na YUV podle [6] určují tyto vztahy:

$$Y = 0{,}257R + 0{,}504G + 0{,}098B + 16 \quad \textbf{(3)}$$

$$U(C_b) = 0{,}439R - 0{,}368G - 0{,}071B + 128 \quad \textbf{(4)}$$

$$V(C_r) = -0{,}148R - 0{,}291G + 0{,}439B + 128 \quad \textbf{(5)}$$

Při převodu z YUV na RGB lze zase vycházet z těchto vztahů:

$$R = 1{,}164(Y - 16) + 1{,}596(C_r - 128) \quad \textbf{(6)}$$

$$G = 1{,}164(Y - 16) - 0{,}813(C_r - 128) - 0{,}391(C_b - 128) \quad \textbf{(7)}$$

$$B = 1{,}164(Y - 16) + 2{,}018(C_b - 128) \quad \textbf{(8)}$$

4. Odstranění prokládání

V některých případech se odstraněním prokladu není třeba zabývat, protože se proklad neprojeví. Tato práce sice tyto případy neřeší, ale pro komplexní přehled je třeba uvést krátce, které případy to jsou. Jedná se zejména o případy, kdy se obraz převzorkovává na menší rozlišení, tzn. směrem dolů. Potom se totiž místo nutných 576 řádků pro plné rozlišení PAL zaznamenává např. 480 řádků (HUGE), někdy 288 řádků (CIF) anebo 240 řádků (STANDARD). Výrazy v závorkách jsou názvy jednotlivých formátů. Pokud se tedy použijí rozměry menší než polovina plného rozlišení, jsou již tyto principielně zbaveny prokládání. Pro větší formáty ale problém s odstraněním prokládání trvá.

4.1. Metody odstranění důsledků prokládání obrazu různým ukládáním půlsnímků

Následující metody podle [2] vycházejí z konkrétní používané normy televizního obrazu a určují různé způsoby uložení a zobrazení půlsnímků tak, aby se odstranily chyby obrazu vzniklé prokládáním.

4.1.1. Top Field First

Nejedná se tak ani o metodu odstranění prokládání, jako spíše o vhodně zvolené ukládání půlsnímků. Liché řádky se uloží do prvního půlsnímku a sudé do druhého. Pokud budou pro ilustraci celé snímky označeny písmeny velké abecedy – ABCD, a půlsnímky číslicemi 1 a 2, bude zobrazování probíhat v tomto schématu: A1 - A2 - B1 - B2 - C1 - C2 ... Dosáhlo se tak toho, že oba půlsnímky jsou z jednoho časového intervalu. Technicky je tedy třeba zachytit TV nebo převodní kartou do jednoho snímku nejprve první půlsnímek (právě liché řádky) a pak teprve druhý půlsnímek (sudé řádky). Samotný název této metody taktéž zkráceně popisuje pořadí zachycení půlsnímků - horní půlsnímek je zde zachycen jako první.

Pokud by se způsob ukládání půlsnímků otočil, dostali bychom jeden snímek složený ze dvou půlsnímků z jiného časového intervalu (A2 - B1) a došlo by právě k roztřepení obrazu na monitoru.

4.1.2. Pulldown

Jedná se o metodu pracující na podobném principu jako Top Field First. Jedná se tedy opět o speciální ukládání půlsnímků, tentokrát ale modifikované pro normu NTSC, kdy je situace o malinko složitější. Výsledný obraz musí mít zobrazování 30 snímků/s. Použije se tedy metoda *3:2 pulldown*, která i názvem ukazuje princip, při kterém se jeden snímek vloží do tří půlsnímků a druhý snímek do dvou půlsnímků. Zobrazení potom vypadá (s využitím stejného značení snímků a půlsnímků, jako v předchozí metodě) takto:

A1 - A2 - A1 - B2 - B1 - C2 - C1 - C2 - D1 - D2 - E1 - E2 - E1 - F2...

Ze zápisu je vidět, že každý druhý stejný půlsnímek je zobrazen dvakrát. Výsledný počet půlsnímků je potom $\frac{24 \cdot (3+2)}{2} = 60$ **(9)**

4.1.3. Bottom field first

Jedná se o uspořádání půlsnímků v jednom snímku při digitalizaci videa, zejména při DV PAL, které se vyvinulo z principů digitalizace obrazu normy NTSC. Princip *Bottom field first,* lze i podle názvu vysvětlit jako uložení a zobrazení spodního půlsnímku jako první.

Pokud se stane, že zdroj obrazu bude v *Bottom field first*, a výsledek je směřován do *Top field first*, nastane zvláštní paradox, kdy by mohlo dojít k trhanému přehrávání obrazu, nebo dokonce časovému paradoxu, kdy by některý půlsnímek sahal do „budoucnosti“ (pocházel ze snímku z následného časového intervalu). Tato chyba nenastane, pokud bude správně určena a neustále přenášena informace o tom, který půlsnímek je první, nebo pokud se obraz bude zobrazovat prokládaně na TV.

4.2. Jednoduché metody deinterlacing

Vzhledem k tomu, že jsou oba půlsnímky z jiného intervalu, dojde při jejich pouhém sloučení do jednoho snímku a zobrazení najednou k rozostření okrajů objektů a zejména pro objekty rychle se pohybující dojde k velkému narušení celistvosti okrajů. Proto existuje celá řada metod, které odstraňují důsledky prokládání. Tyto metody jsou různě účinné a mají také různou výpočetní náročnost.

4.2.1. Sloučení půlsnímků (WEAVE)

Tato metoda se používá v případech, kdy není explicitně zadána jiná metoda deinterlacingu, nebo jako příprava obrazu pro další metody deinterlacingu. Princip této metody spočívá v sestavení celého snímku pouhým sloučením (sečtením) lichého a sudého půlsnímku, přičemž řádky obou půlsnímků se přebírají beze změny. Výsledný obraz odpovídá jednoduchosti metody – pro statické scény je obraz velmi ostrý, čím je však scéna dynamičtější, je obraz více postižen deinterlacingem. Framerate je dále poloviční, tedy 25 snímků/s. Příklad takto upraveného obrazu podle [4] ukazuje Obr.2.:

Obr.2: Výsledek sloučení půlsnímků

4.2.2. Vypuštění jednoho půlsnímku (BOB)

Pokud bude jeden půlsnímek vynechán a ke složení kompletního obrazu se použijí obrazové řádky přítomného půlsnímku dvakrát po sobě, vznikne obraz bez rozkladu, ovšem s menší kvalitou a rozlišením ve svislém směru. Informace o nepoužitém půlsnímku se však vytrácí. Zachová se tak počet snímků za vteřinu, tedy 50, a obraz se chvěje nahoru dolů, protože půlsnímky jsou posunuty právě o jeden řádek. Výsledek této metody podle [4] ukazuje Obr.3.:

Obr.3: Výsledek vypuštění lichého půlsnímku

Stejnou metodu lze modifikovat i tak, že se vypustí naopak sudé řádky. Výsledný obraz podle [4] ukazuje Obr.4.:

Obr.4: Výsledek vypuštění sudého půlsnímku

Porovnáním Obr.3 a Obr.4 lze ukázat základní nevýhodu této metody, která spočívá v tom, že vypuštěním jednoho půlsnímku přijdeme o část obrazové informace, zobrazené v horizontálním směru.

4.2.3. Blending (BLEND FIELD)

Tato metoda je založena na míchání (sečtení) půlsnímků připravených jinou metodou deinterlacing. Pokud se tedy připraví způsobem uvedeným v kap.4.2.2. oba půlsnímky a sečtou se tak, aby výsledný obraz získal původní rozlišení, bude výsledkem úpravy obraz, který je opět kvalitnější a účinněji odstraňuje proklad, ovšem přináší oproti originálu rozmazání, jakoby způsobené dvojí expozicí. Příklad takového obrazu podle [4] ukazuje Obr.5:

Obr.5: Výsledek sečtení dvou půlsnímků z Obr.3 a Obr.4

4.3. Pokročilé metody deinterlacing

Pokročilé metody jsou již početně více náročné, neboť často analyzují celé skupiny snímků, ze kterých si odvodí pohybové vektory jednotlivých částí obrazu a pomocí těchto rekonstruují jednotlivé části obrazu podle potřeby tak, že u rychle se měnících částí scény se postarají o co nejlepší obraz s maximem detailů bez vzájemného posunutí a rozostření. Porovnání výsledků jednotlivých skupin deinterlacingu podle [4] ukazuje Obr.6:

Obr.6: Porovnání výsledků metod deinterlacing

4.3.1. Interpolace snímků

Pro dosažení dobrých výsledků lze použít metodu, kdy se jeden půlsnímek přebírá beze změny a druhý se interpoluje ze sousedních půlsnímků stejného druhu – pokud necháme lichý půlsnímek beze změny, pak se sudý interpoluje z nejbližších sudých půlsnímků. Lepšího výsledku lze dosáhnout interpolací obou půlsnímků a současně také zdokonalením principu interpolace, kde je možno využít interpolaci lineární nebo vyšších řádů.

4.3.2. Adaptivní algoritmy

Tyto metody poskytují nejlepší výsledky a svojí kvality dosahují složitými výpočty a operacemi. Umožňují měnit adaptivně odstranění prokladu podle vlastností právě zpracovávaného úseku scény (její dynamiky) nebo také části jediného snímku. Dosahuje se tak ostrý obraz pro statické scény obrazu a hladké obrysy pohybujících se objektů při dynamických scénách obrazu.

4.4. Shrnutí metod deinterlacing

Z velkého množství samotných metod, nebo jejich kombinací, lze vybírat dle vlastního uvážení, možností digitalizačního programu nebo ovladače příslušné karty či požadavků na výsledný obraz. Deinterlacing lze provést jednorázově – jedním průchodem při digitalizaci, nebo lépe nadvakrát, kdy se nejprve použije metoda sloučení půlsnímků (Kap.4.2.1.) a na výsledek se nasadí vhodný algoritmus. Výhody a nevýhody jednotlivých metod deinterlacingu podle [2] ukazuje Tab.5:

Tab.5: Přehled a porovnání vlastností metod deinterlacing

Metoda	Výhody	Nevýhody
Míchání půlsnímků (Blending Fields) 720x576 => 720x576 25fps	Plynulý obraz. Vhodný pro všechny programy. Obraz se nemusí nejprve převádět na půlsnímky.	Obraz je rozmazaný v dynamických scénách a mnohdy i ve scénách statických. Nedostatečný kompresní poměr.
Vynechání půlsnímku (Discarding Fields Single Field Mode) 720x576 => 720x288 25fps	Vhodný pro všechny programy. Jasný obraz. 100 % deinterlaced – prokládané řádky jsou odstraněny. Obraz se nemusí nejprve převádět na půlsnímky. Proces je velmi rychlý. Ostřejší obraz než při míchání půlsnímků.	Ztráta poloviny informací o obrazu. V klidných scénách malá ztráta ostrosti, protože výška snímku vznikne natažením půlsnímku. Hrubější struktura obrazu. Video není plynulé. Větší viditelnost artefaktů komprimace.
Adaptivní deinterlacing (Adaptive deinterlacing) 720x576 => 720x576 25fps	Obraz se nemusí nejprve převádět na půlsnímky. Při dobrém nastavení rozmaže okraje v rychlých pohybech zatímco chrání ostrost klidných scén.	Vždy neodstraní všechny prokládané linky. Někdy odstraní potřebná video data. Někdy má komplikované parametry. Obraz může být nepřirozeně zastřený (unsharp).
Bob (Bob) 720x576 => 720x288 50 fps	Plynulý obraz. 100 % deinterlaced film, protože některé prokládané linky jsou vypuštěny.	Větší viditelnost artefaktů komprimace. V klidných scénách nečinný. Ztráta ostrosti, protože každý snímek je vytvořen zvětšením půlsnímku. Potřeba rychlejšího procesoru, zvláště z důvodu framerate 50fps. Velká výsledná velikost souboru.

Bob + weave (Combination of Bob+Weave) (=Progressive Scan) 720x576 => 720x576 50 fps	Výborný plynulý a ostrý obraz. 99 % deinterlaced film, což znamená malou možnost výskytu neostrosti okraje. V klidných scénách se neuplatňuje (prokládání nevadí) a v dynamických scénách se projeví plynulostí.	Není rozšířen u všech programů kvůli náročnosti. Pokud má být framerate 50fps, potřebujeme rychlejší procesor nebo rychlejší kodek. Protože film musí být rozdělen do půlsnímků, je rychlost kódování omezena pouze programem, a ten může být docela pomalý. Výsledná velikost souboru je větší než s jinými metodami.
Adaptivní deinterlacing (Motion compensation) 720x576 => 720x576 50 fps	Všechny pozitivní rysy. Velmi kvalitní výsledek.	Zatím se nestal standardem.

5. Implementace metod pro odstranění prokládání

5.1. Obecná pravidla

Při prohlížení videosekvencí na monitoru počítače dojde tedy k tomu, že počítač prostřednictvím jeho grafické karty spojí půlsnímky dohromady. Prokládání lze prakticky odstranit už při natáčení, kdy bude kamera nastavena na režim Progressive, nebo-li 25p, kdy však dojde k menší plynulosti rychlých pohybů a kamery, vybavené touto funkcí, patří většinou do střední a vyšší třídy. Samozřejmě, pokud je již video natočeno jako prokládané, nelze jej nikdy 100% zbavit prokládání, lze se pouze více či méně přiblížit kvalitou k originálu. Tato účinnost závisí na výkonu použité metody odstranění prokládání.

Programů a filtrů na odstranění prokládání existuje celá řada, ale pouze některé z nich dosahují dobrých výsledků. Pro všechny metody ale platí podmínka mít na vstupu půlsnímky oddělené. Ve firemních filtrech, např. pro program Avisynth se toto oddělení půlsnímků provádí skriptem:

```
AVISource("video.avi")
SeparateFields()
```

Pro potřeby této práce jsou použity již připravené půlsnímky z ftp serveru ftp.ebu.ch, který slouží právě pro potřeby testování obrazu ve vysokém rozlišení s možností stažení jednotlivých půlsnímků.

Tyto půlsnímky, číslované v pořadí sejmutí, jsou uloženy ve formátu .sgi a je proto nutné je převést do formátu .bmp, který lze potom v prostředí Matlab načíst do matice příkazem:

imread()

Pokud by se takto načetl obrázek ve stupních šedi, vznikne matice dvourozměrná, v případě obrázku barevného bude matice trojrozměrná. A nyní již k praktickému odstranění prokládání dle některých metod uvedených výše.

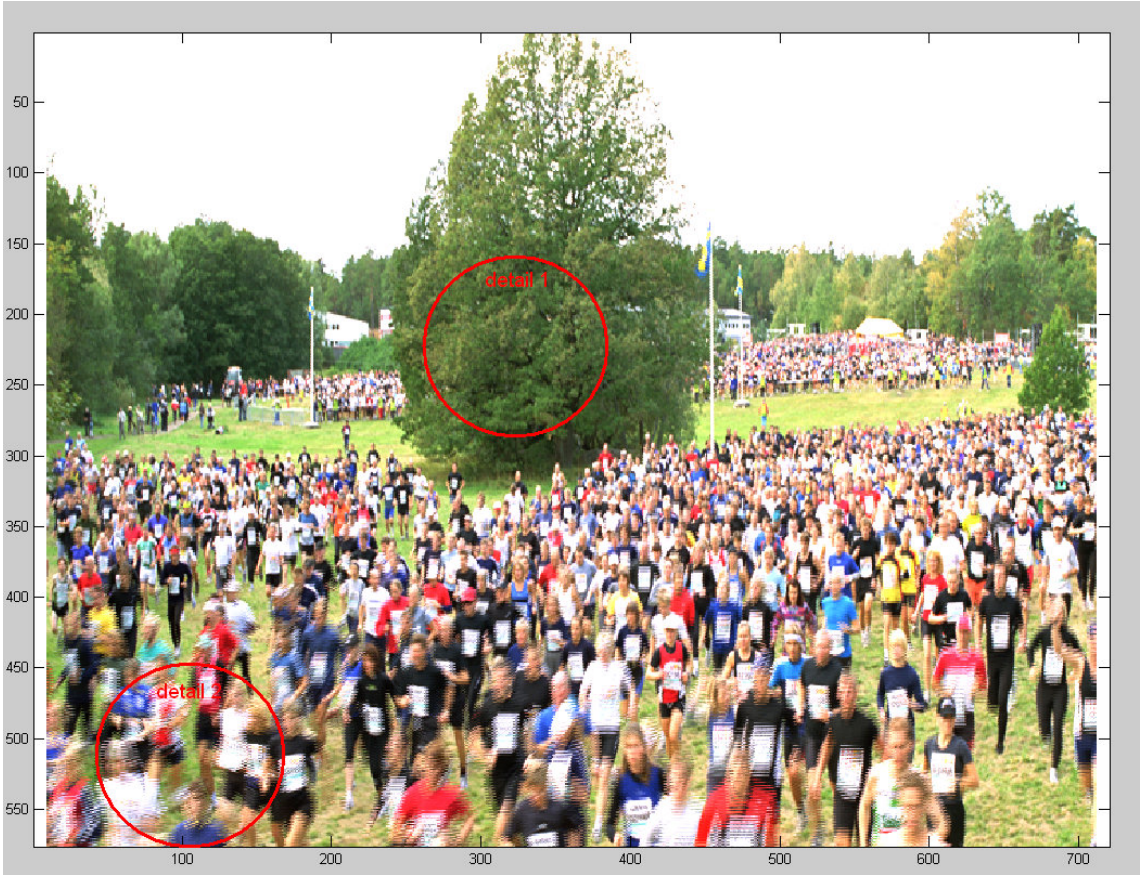
5.2. Implementace Weave

```
1   %%Diplomova prace (Pavel Menhart)
2   %%
3   %%Odstraneni deinterlacing - Weave.
4   %%
5   %%nacteni licheho pulsnimku do matice
6 - obrL=imread('03556.bmp');
7   %%nacteni sudeho pulsnimku do matice
8 - obrS=imread('03557.bmp');
9   %%slouceni pulsnimku
10 - obr=obrL+obrS;
11  %%vykresleni vysledneho obrazku
12 - imshow(obr);
```

Obr.7: Výpis programu metody Weave

Protože se jedná o metodu základní a mnohdy i přípravnou pro další metody odstranění prokládání, nedosahuje takové kvality výsledného obrazu. Jak je vidět z výsledku sloučení půlsnímků, statické části obrazu jsou ostré (detail 1), kdežto dynamické části obrazu trpí rozkladem i nadále (detail 2). Příklad obrazu upraveného metodou *weave* ukazuje Obr.8.

Protože je každý nový snímek vytvořen sečtením dvou půlsnímků, zůstává framerate nadále poloviční, tedy 25 snímků/s. V konkrétním zobrazeném případě je rozlišení každého původního půlsnímku 720x576 bodů, tudíž i výsledný obrázek má rozlišení 720x576 bodů, neboť řádky se přejímají beze změny a pouze se sčítají hodnoty jednotlivých obrazových bodů, jak je vidět ve výpisu m-file na Obr.7 v řádku 10.



Obr.8: Výsledek metody weave

5.3. Implementace Bob

Tato metoda odstranění prokladu vypouští jeden půlsnímek a ke složení kompletního obrazu používá obrazové řádky přítomného půlsnímku dvakrát po sobě.

Výsledný obraz je samozřejmě 100% zbaven prokladu, ovšem vypuštěním jednoho půlsnímku dochází ke ztrátě informace o půlsnímku druhém a výsledný obraz má menší kvalitu a rozlišení ve svislém směru. Jak je vidět na Obr.10, kontury osob v popředí, které se pohybují, jsou velmi výrazně hranaté.

```
1    %%Diplomova prace (Pavel Menhart)
2    %%
3    %%Odstraneni deinterlacing - Bob.
4 -  clear all; close all;
5    %%nacteni prvniho licheho pulsnimku do matice
6 -  obrA=imread('03556.bmp');
7    %%nacteni stejneho licheho pulsnimku do matice
8 -  obrB=imread('03556.bmp');
9    %%rozlozeni matice obrazku do radkoveho vektoru
10 - obrL=reshape(obrA,1,414720,3);
11 - obrS=reshape(obrB,1,414720,3);
12   %%vypusteni lichych radku
13 - vL=obrL(1:2:end);
14 - vS=obrS(1:2:end);
15   %%slozeni noveho vektoru
16 - obr=[vL;vS];
17   %%slozeni vysledneho obrazku
18 - OBR=reshape(obr,[],720,3);
19   %%vykresleni snimku
20 - imshow(OBR);
```

Obr.9: Výpis programu metody Bob

Obr.10: Výsledek metody bob

5.4. Implementace Blend Field

U této metody se nejprve připraví dva půlsnímky, vzniklé dvojnásobným, po sobě jdoucím použitím lichého (sudého) půlsnímku tak, jak je popsáno metodou BOB. Teprve potom se použije metoda Blend Field, při které se oba připravené půlsnímky sečtou.

```
%%Diplomova prace (Pavel Menhart)
%%
%%Odstraneni prokladani - metoda Blend Field
close all; clear all;
%%nacteni prvniho licheho pulsnimku do matice
obrA=imread('03556.bmp');
%%nacteni stejneho licheho pulsnimku do matice
obrB=imread('03556.bmp');
%%nacteni prvniho sudeho pulsnimku do matice
obrC=imread('03557.bmp');
%%nacteni stejneho sudeho pulsnimku do matice
obrD=imread('03557.bmp');
%%rozlozeni matic obrazku do radkoveho vektoru
obrL=reshape(obrA,1,414720,3);
obrS=reshape(obrB,1,414720,3);
obrLL=reshape(obrC,1,414720,3);
obrSS=reshape(obrD,1,414720,3);
%%vypusteni lichych radku
vL=obrL(2:2:end);
vS=obrS(2:2:end);
vLL=obrLL(1:2:end);
vSS=obrSS(1:2:end);
%%slozeni noveho vektoru
obr1=[vL;vS];
obr2=[vLL;vSS];
%%slozeni vysledneho obrazku
img1=reshape(obr1,[],720,3);
img2=reshape(obr2,[],720,3);
%%slouceni pulsnimku
OBR=img1+img2;
%%vykresleni snimku
imshow(OBR);
```

Obr.11: Výpis programu metody Blend Field

Jak je vidět z výsledku této metody (viz Obr.12), výsledný snímek je opět kvalitnější, oproti metodě *bob* obsahuje informace i z druhého půlsnímku, ovšem celkově obraz oproti originálu vykazuje rozmazání, jakoby způsobené dvojí expozicí.

Zajímavým atributem takto upraveného snímku je i zvýšení jasu, způsobené sčítáním hodnot jasu obou předpřipravených půlsnímků. Pro odpovídající zobrazení by tedy bylo nutno tento jas pomocí filtru opět snížit, aby nedocházelo ke zkreslení obrazu v jasové oblasti.

Obr.12: Výsledek metody Blend Field

5.5. Implementace Bob + Weave

Z jednodušších metod pro odstranění prokládání lze vytvořit i velmi účinnou kombinaci, označovanou jako Bob+Weave. Spočívá v přípravě dvou půlsnímků metodou Bob, tedy vypuštěním jednoho typu půlsnímků a následné implementaci metody Weave, při které dochází ke sloučení takto připravených půlsnímků. Rychlost kódování je potom omezena pouze programem, který tuto aplikaci provádí, a právě tento faktor může být omezující. Výsledná velikost souboru je větší než u jiných metod.

Výsledný obraz, jak ukazuje Obr.14, je velmi účinně zbaven prokládání, ovšem okraje obrazu a obrysy jednotlivých objektů jsou ještě neostré, roztřepené.

```
%%Diplomova prace (Pavel Menhart)
%%
%%Odstraneni prokladani - metoda Bob+Weave
%%
clear all;
close all;
%%---------------------------------------%%
%%PRIPRAVA LICHEHO PULSNIMKU metodou BOB%%
%%---------------------------------------%%
%%nacteni prvniho licheho pulsnimku do matice
obrA=imread('03556.bmp');
%%nacteni stejneho licheho pulsnimku do matice
obrB=imread('03556.bmp');
%%rozlozeni matic obrazku do radkoveho vektoru
obrL=reshape(obrA,1,414720,3);
obrS=reshape(obrB,1,414720,3);
%%vypusteni lichych radku
vL=obrL(1:2:end);
vS=obrS(1:2:end);
%%slozeni noveho vektoru
obr1=[vL;vS];
%%slozeni vysledneho snimku
OBR1=reshape(obr1,[],720,3);
%%---------------------------------------%%
%%PRIPRAVA SUDEHO PULSNIMKU metodou BOB%%
%%---------------------------------------%%
%%nacteni prvniho sudeho pulsnimku do matice
obrC=imread('03557.bmp');
%%nacteni stejneho sudeho pulsnimku do matice
obrD=imread('03557.bmp');
%%rozlozeni matic obrazku do radkoveho vektoru
obrLL=reshape(obrC,1,414720,3);
obrSS=reshape(obrD,1,414720,3);
%%vypusteni lichych radku
vLL=obrLL(1:2:end);
vSS=obrSS(1:2:end);
%%slozeni noveho vektoru
obr2=[vLL;vSS];
%%slozeni vysledneho snimku
OBR2=reshape(obr2,[],720,3);
%%---------------------------------%%
%%SLOUCENI PULSNIMKU metodou WEAVE%%
%%---------------------------------%%
OBR=OBR1+OBR2;
%%vykresleni vysledneho snimku
imshow(OBR);
```

Obr.13: Výpis programu metody Bob + Weave

Obr.14: Výsledek metody Bob + Weave

5.6. Implementace Interpolace snímků

Jak je vidět z obrázků Obr.8, Obr.10 a Obr.12, předchozí metody nevykazují u snímků dokonalou kvalitu obrazu, pouze dokázaly odstranit chybu způsobenou prokládáním. Lepších výsledků dosahují metody, kdy se jeden půlsnímek přebírá beze změny a druhý se interpoluje ze sousedních snímků stejného druhu, nebo ještě lépe – každý půlsnímek se získává interpolací ze snímků stejného druhu.

Zkráceně lze metody interpolace obrazu popsat tak, že se jedná o matematické postupy, kdy se při následném zpracování dat dopočítávají hodnoty bodů, z nichž se potom skládá obrázek, srovnáním s fyzicky existujícími body. Tyto nově vzniklé body se nacházejí na souřadnicích, které původně v obrazu nebyly. Velmi rozsáhlé použití interpolačních metod je při procesech zmenšování či zvětšování obrazu, kdy je třeba dopočítat hodnoty jasu bodů na nových souřadnicích.

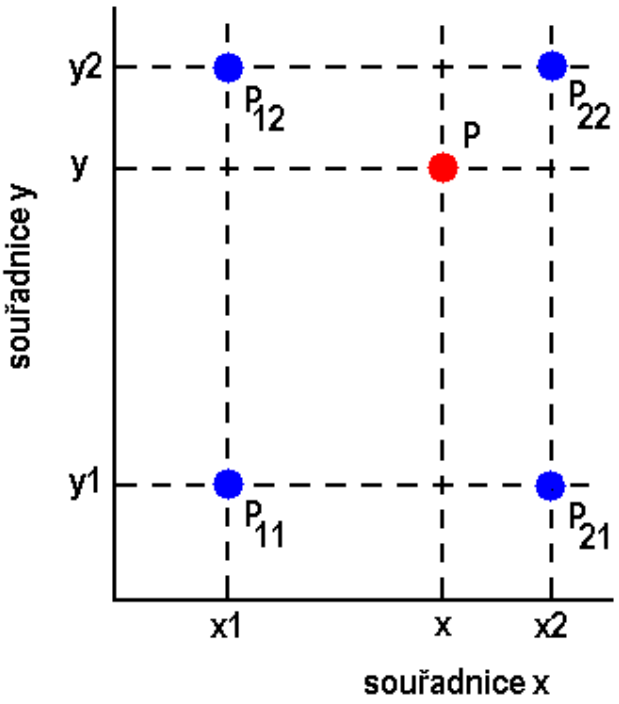
Matematických principů interpolace je celá řada. Krátký přehled těchto metod poukáže na rozdíly mezi jednotlivými metodami, složitostí jejich výpočtů a teoreticky dosažitelných výsledků při interpolaci obrazu.

a) metoda nejbližšího souseda

- hodnota jasu bodu v novém obraze je nahrazena hodnotou jasu nejbližšího „sousedního“ známého bodu. Tato interpolační metoda je velmi hrubá a geometricky nepřesná, ale lze ji použít pro všechny typy obrazů. Přitom její největší výhodou je, že zachovává původní hodnoty jasu bodů, tzn., že tato metoda negeneruje žádná nová data.

b) bilineární interpolace

- hodnota bodu v novém obraze je vypočtena jako vážený průměr čtyř nejbližších sousedních bodů v původním obraze. Výsledný obraz již neobsahuje nespojitosti a kontury výstupního obrazu jsou hladké. Tato metody již původní hodnoty obrazových bodů, protože je přepočítává z existujících hodnot. Pro definici této metody slouží Obr.15:



Obr.15: Princip bilineární interpolační metody

Hodnota jasu nového bodu P (značený červeně) se vypočítá z hodnot jasů a polohy sousedních čtyř bodů P_{11} až P_{22} (značeny modře). Tyto čtyři původní body originálu jsou definovány podle [9] takto:

$$\begin{aligned} P_{11} &\ldots f(x, y) \\ P_{12} &\ldots f(x, y+1) \\ P_{21} &\ldots f(x+1, y) \\ P_{22} &\ldots f(x+1, y+1) \end{aligned} \qquad \textbf{(10 - 13)}$$

kde: (x,y) ... jsou souřadnice bodu
f ... má význam hodnoty jasu

Poloha nově vypočteného bodu bude tedy

$$(x + \Delta x, y + \Delta y) \qquad \textbf{(14)}$$

kde: $\Delta x, \Delta y$ ... jsou vzdálenosti vypočteného bodu od výchozího bodu P_{11}

Definice bilineární interpolace je dána:

$$f(x + \Delta x, y + \Delta y) = b_1 + b_2 \cdot \Delta x + b_3 \cdot \Delta y + b_4 \cdot \Delta x \cdot \Delta y \qquad \textbf{(15)}$$

kde: b_1, b_2, b_3, b_4 ... koeficienty posunu lze zjistit dosazením hodnot 0 a 1 za vzdálenosti $\Delta x, \Delta y$ takto:

$$\begin{aligned} b_1 &= f(x, y) \\ b_2 &= f(x+1, y) - f(x, y) \\ b_3 &= f(x, y+1) - f(x, y) \\ b_4 &= f(x+1, y+1) + f(x, y) - f(x+1, y) - f(x, y+1) \end{aligned} \qquad \textbf{(16 – 19)}$$

Dosazením výrazů pro jednotlivé koeficienty (16 – 19) do definice pro bilineární interpolaci (15) lze potom definovat výsledný vztah pro výpočet hodnoty jasu z hodnot čtyř sousedních bodů následovně:

$$\begin{aligned} f(x + \Delta x, y + \Delta y) &= [f(x+1, y) - f(x, y)] \cdot \Delta x + [f(x, y+1) - f(x, y)] \cdot \Delta y + \\ &+ [f(x+1, y+1) + f(x, y) - f(i+1, y) - f(x, y-1)] \cdot \Delta x \cdot \Delta y + f(x, y) \end{aligned} \qquad \textbf{(20)}$$

c) bikubická interpolace

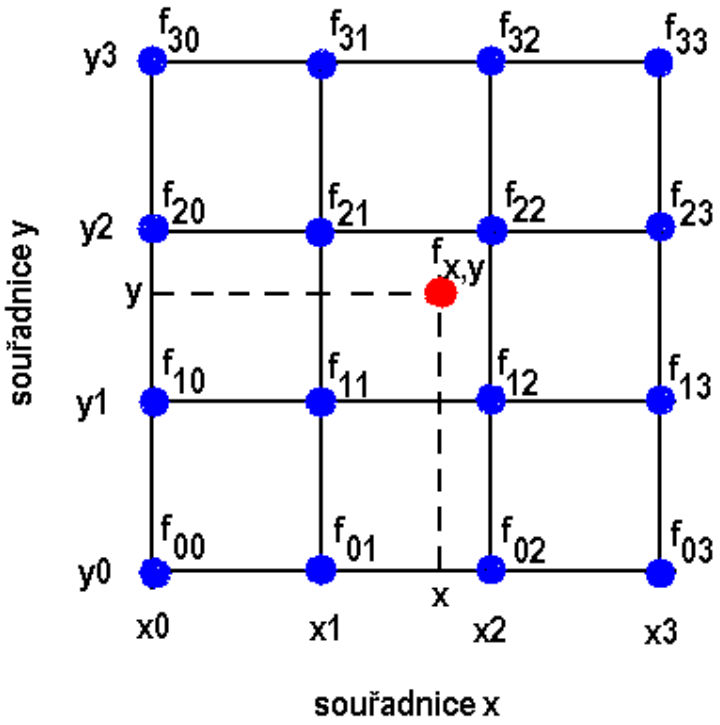
- pro aproximaci v tomto případě se používá okolí složené ze 16-ti bodů. Model obrazové funkce je zpřesněn tím, že je lokálně interpolován bikubickým polynomem. Hodnota nového bodu se tedy vypočítá jako vážený průměr šestnácti nejbližších bodů v původním obraze.

Z hlediska geometrické přesnosti a ostrosti výsledného obrazu podává tato metoda lepší výsledky než metody předešlé, dochází však ke znatelné změně hodnot původních bodů a výrazně narůstá výpočetní náročnost. Názorně ukazuje situaci při bikubické interpolaci Obr.16.

Metoda získává hodnotu jasu nového bodu $f_{x,y}$ (označený červeně) z jasu okolních 16-ti bodů $f_{00} - f_{33}$ (označených modře). Výpočet pomocí bikubických polynomů je definován následovně:

$$f(x, y) = \sum_{i,j=0}^{3} c_i(x) c_j(y) f_{ij} \qquad \textbf{(21)}$$

kde: $c_i(t)$ ... jsou kubické polynomy



Obr.16: Princip bikubické interpolační metody

d) kubická interpolace

- na rozdíl od předchozích způsobů je zde využita trojúhelníková síť a pro aproximaci se používá 10-ti bodů. Model obrazové funkce je lokálně interpolován kubickým polynomem. Oproti bikubické interpolaci má kubická interpolace tu výhodu, že aproximuje polynomem nižšího řádu. Aproximační funkce má tedy větší stabilitu.

Definiční vztah pro výpočet jasu nového bodu z jasu bodů známých je dán:

$$f(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} c_i(x)c_j(y)\,f_{ij} \qquad \textbf{(22)}$$

kde: $c_i(t)$ ... váhové funkce lze vyjádřit jako:

$$\begin{aligned} c_0(t) &= (1-t)^3 \\ c_1(t) &= 3t^3 - 6t^2 + 4 \\ c_2(t) &= -3t^3 + 3t^2 + 3t + 1 \\ c_3(t) &= t^3 \end{aligned} \qquad \textbf{(23 – 26)}$$

Ze samotných definicí jednotlivých principů interpolace vyplývá, že nejlepších výsledků by měla dosahovat bikubická interpolace, která je výpočetně nejsložitější. Při volbě kompromisu mezi výpočetní složitostí, rychlostí a dosažitelným výsledkem, byla implementována metoda bilineární interpolace.

```
%%Diplomova prace (Pavel Menhart)
%%
%%Odstraneni prokladani - metoda Interpolace
%%
clear all;
close all;
%%nacteni tri po sobe jdoucich pulsnimku stejneho druhu
obrA=imread('03556.bmp');
obrB=imread('03558.bmp');
obrC=imread('03560.bmp');
%%rozlozeni matic obrazku do radkoveho vektoru
obr1=reshape(obrA,1,414720,3);
obr2=reshape(obrB,1,414720,3);
obr3=reshape(obrC,1,414720,3);
%%slozeni noveho vektoru
obrN=[obr1;obr2;obr3];
%%opetovne slozeni matice slozenych obrazku
obrN=reshape(obrN,[],720,3);
%%zmenseni obrazku a bilinearni interpolace
obr=imresize(obrN,[576 720],'bilinear');
%%nacteni pulsnimku opacneho druhu
obrD=imread('03557.bmp');
%%slouceni pulsnimku
IMG=obr+obrD;
%%vykresleni vysledneho snimku
imshow(IMG);
```

Obr.17: Výpis programu bilineární interpolační metody

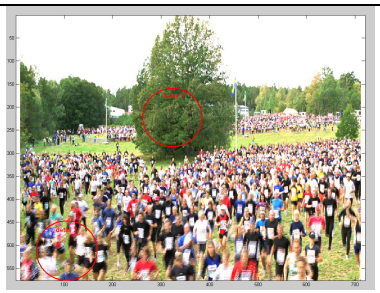
Z hlediska geometrické přesnosti dává tato metoda lepší výsledky než metody předešlé. Výsledný obraz má ostrý vzhled, ale dochází ke změně hodnot původních pixelů. Dosažený výsledek je zobrazen na Obr.18.

Obr.18: Výsledek bilineární interpolační metody

5.7. Zhodnocení implementovaných metod

Při porovnání jednotlivých implementovaných metod pro odstranění prokládání mezi sebou, vznikne závěr, že se výsledky metod více či méně blíží k obrazu, který v určitém rozlišení a ostrosti připomíná původní obraz, ale tento fakt je pouze subjektivní, protože obraz zatížený prokladem, se k originálu vždy pouze blíží. Pro lepší srovnání jsou seřazeny všechny implementované metody do přehledné tabulky (Tab.6), kde lze vypozorovat, že nejlepších výsledků lze dosáhnout metodami Bob+Weave a hlavně interpolačními metodami.

Tab.6: Přehled a porovnání implementovaných metod deinterlacing

Weave	
Bob	
Blend Field	
Bob + Weave	
Bilineární interpolace	

5.8. Implementace metod deinterlacing v jazyce C

Po porovnání jednotlivých implementovaných metod pro odstranění prokládání v prostředí MATLAB, které je uvedeno v kap.5.7 a v Tab.6, vychází jako nejúčinnější interpolační metoda, která bude dále algoritmizována v jazyce C. Dále bude provedena algoritmizace i jednodušší metody, jejichž porovnání bude sloužit jako důkaz vhodněji zvolené metody k odstranění prokladu.

Při uvažování nekomprimované videosekvence, ve které je barevná složka modulována se stejnou šířkou pásma jako je šířka pásma jasové složky, mluvíme o vzorkování s označením 4:4:4 a prakticky to znamená, že jasová složka jednoho snímku zabírá v paměti stejné místo, jako zabírá chrominanční složka Cr a to stejné místo zabírá i chrominanční složka Cb. Při načítání souboru .yuv tedy načítáme stejný počet bajtů pro jasovou složku a složky Cr a Cb, přičemž počet bajtů je dán počtem řádků x počet bodů na řádku.

5.8.1. Implementace metody Bob

V prvním algoritmu, jehož nejdůležitější část zdrojového kódu je na Obr.19, se jako velmi jednoduchá metoda odstranění prokladu používá princip, kdy se lichý řádek jednoho snímku použije dvakrát po sobě. Toto opakování lichého řádku odpovídá metodě Bob, popsané v kapitole 5.3.

```
/*------------------------------------------------------------------------*/
/*                          Odstraneni prokladu                           */
/*------------------------------------------------------------------------*/

 for(i=0; i < frames*lines; i = i+2)
      for(j=0; j < ptsinline; j ++)
    {
            /* prekopirovani radku vzorek po vzorku */
        *(pOutY + ptsinline*i     + j) = *(pInY + ptsinline*i + j);
            /* vlozeni kopie aktualniho radku */
        *(pOutY + ptsinline*(i+1) + j) = *(pInY + ptsinline*i + j);
      }
/*------------------------------------------------------------------------*/
/*                          Zapis do souboru                              */
/*------------------------------------------------------------------------*/

/* ulozeni radku snimku */
  if (fwrite(pOutY, sizeof(char), frames*lines * ptsinline, fOut) !=
(frames*lines * ptsinline))
  {
      printf("Selhal zápis do souboru.\n");
      return 5;
  }
```

Obr.19: Část zdrojového kódu metody Bob

Základem této jednoduché metody je nejprve načtení jednoho řádku, v našem případě lichého řádku vzorek po vzorku do výstupního souboru pOutY pomocí:

```
*(pOutY + ptsinline*i     + j) = *(pInY + ptsinline*i + j);
```

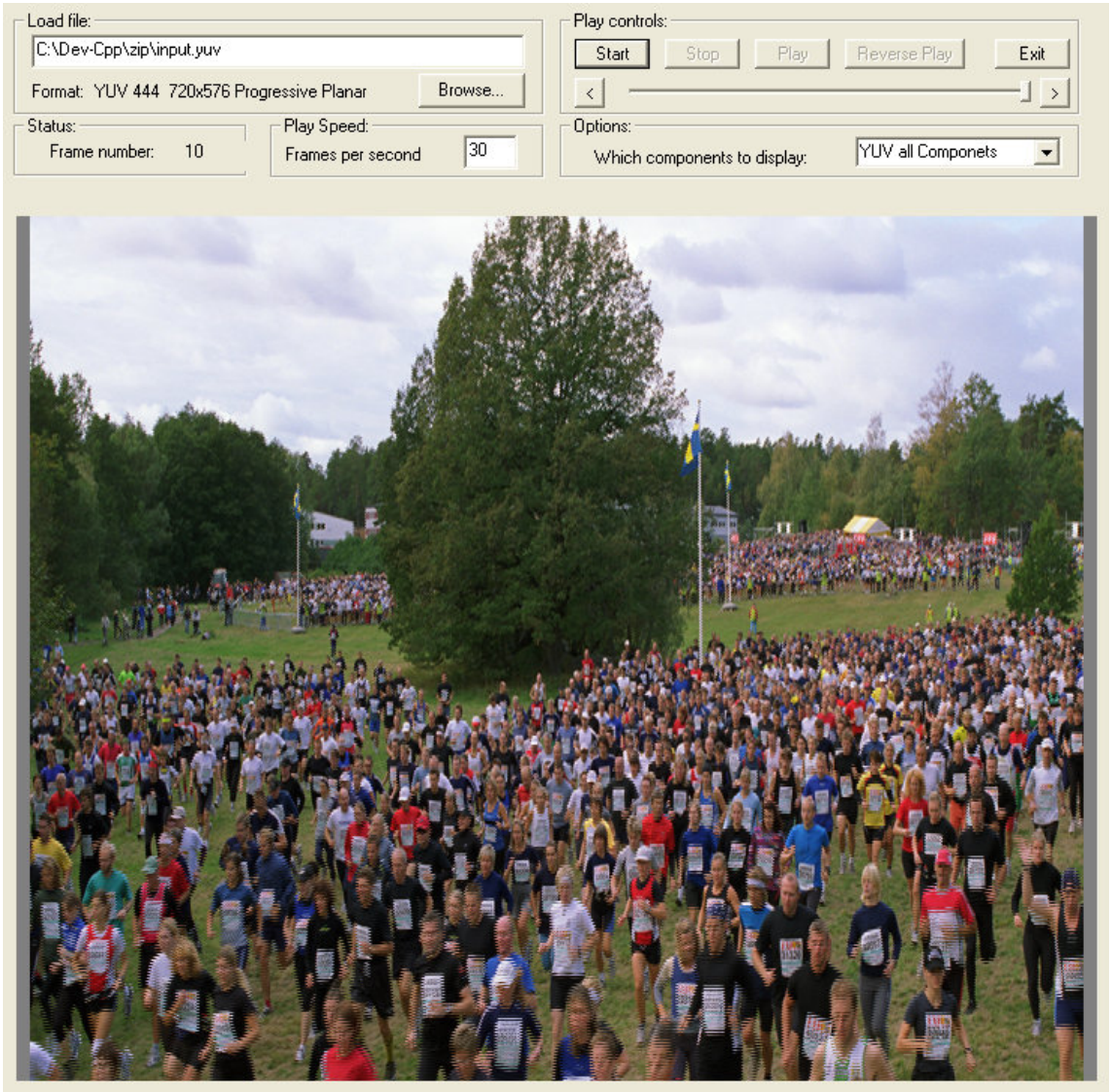
a poté se ve výstupním souboru posune pointer na nový řádek, kam se nakopíruje stejný, lichý, již zpracovaný řádek pomocí:

```
*(pOutY + ptsinline*(i+1) + j) = *(pInY + ptsinline*i + j);
```

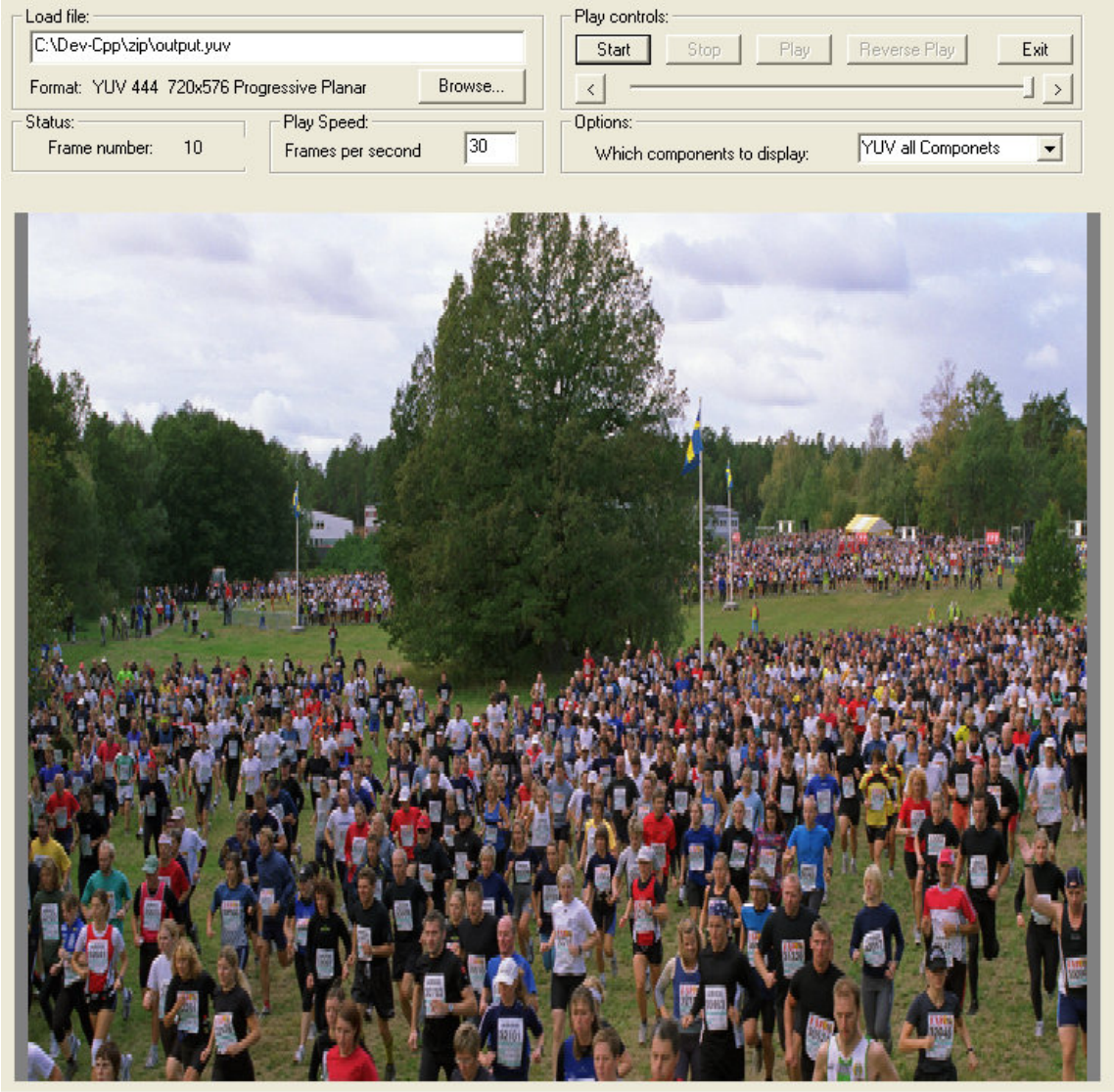
Celý proces se opakuje do doby, než proměnná *i* nedosáhne zadaného počtu řádků. Význam jednotlivých proměnných ve zdrojovém kódu je následující:

pInY ... pointer na místo uložení začátku vstupního snímku
pOutY ... pointer na místo uložení začátku výstupního snímku
lines ... počet řádků snímku
ptsinline ... počet bodů v řádku
frames ... počet snímků
pInY + ptsinline*i + j ... výpočet adresy, na které je uložen jas bodu se souřadnicemi *i* a *j*
*(pInY + ptsinline*i + j) ... vyčtení obsahu tohoto paměťového místa, tedy hodnoty jasu

Výsledek zpracování obrazu touto jednoduchou metodou, aplikovanou na reálnou videosekvenci, ukazuje kopie obrazovky získaná po spuštění videosekvence před a po volání funkce bob.exe, kterou zobrazuje Obr.20 a 21:



Obr.20: Část vstupní videosekvence před odstraněním prokladu



Obr.21: Část výstupní videosekvence po odstranění prokladu

5.8.2. Implementace Interpolační metody

U metody, která používá interpolaci ze dvou po sobě následujících snímků, je třeba načíst odpovídající řádky jednoho snímku do jedné proměnné a druhého snímku do druhé proměnné, aby mezi těmito řádky mohlo dojít k interpolaci. Nejdůležitější část zdrojového kódu je opět uvedena na Obr.22.

V proměnné pInY1 je načten vzorek po vzorku lichý řádek prvního snímku a v proměnné pInY2 je načten vždy odpovídající vzorek z druhého snímku:

```
*(pOutY + ptsinline*i      + j) = ((*(pInY1 + ptsinline*i + j) + *(pInY2 +
ptsinline*i + j))/2);
```

Po vykonání interpolace mezi odpovídajícími vzorky je z výsledků sestaven lichý řádek výsledného obrazu, který se uloží do lichého řádku souboru pOutY:

```
(fwrite(pOutY, sizeof(char), lines * ptsinline, fOut)
```

Sudý řádek se do výstupního souboru pouze kopíruje z prvního snímku:

```
*(pOutY + ptsinline*(i+1) + j) = *(pInY1 + ptsinline*(i+1) + j);
```

```
/*-------------------------------------------------------------------------*/
/*                          Odstraneni prokladu                            */
/*-------------------------------------------------------------------------*/

 for(i=0; i < lines; i = i + 2)
      for(j=0; j < ptsinline; j ++)

    {
/*Z odpovidajicich lichych radku obou snimku se interpoluje novy lichy radek*/
        *(pOutY + ptsinline*i     + j) = ((*(pInY1 + ptsinline*i + j) +
*(pInY2 + ptsinline*i + j))/2);
/* Sudy radek noveho snimku se prejima beze zmeny z prvniho snimku */
        *(pOutY + ptsinline*(i+1) + j) = *(pInY1 + ptsinline*(i+1) + j);
      }
/*-------------------------------------------------------------------------*/
/*                          Zapis do souboru                               */
/*-------------------------------------------------------------------------*/

  if (fwrite(pOutY, sizeof(char), lines * ptsinline, fOut) != (lines *
ptsinline))
  {
      printf("Selhal zapis do souboru.\n");
      return 5;
  }
```

Obr.22: Část zdrojového kódu interpolační metody

V obou případech práce se snímky se musí vždy brát v úvahu uspořádání snímku, ve kterém jsou obsaženy liché i sudé řádky, a proto je zařazen ve funkci for parametr i = i+2 tak, aby se rozlišily liché a sudé řádky a bylo umožněno přistupovat k nim jednotlivě.

6. Závěr

Dostupnost digitálních kamer, DVD rekordérů a jednoduché možnosti zpracování videa na počítačích pro každého, jsou velkou výhodou pro lovce a zpracovatele domácího videa, ale zároveň i velkou výzvou. Mnoho uživatelů si totiž klade pouze otázku *jak*, ale už se nezeptají *proč*? A právě na otázku proč, měla odpovědět tato diplomová práce.

Každý uživatel totiž velmi lehce získá záznam krásných zážitků ze svého života, a poté najednou vyvstane problém s jejich úpravou a uložením. Jak jsem uvedl v kapitole 3, formát RGB umožňuje snadnou editaci, střih a další práci s obrazem, ale má větší objem dat, kdežto formát YUV je zase spořivější co se týká nároků na paměť, a proto se hodí k ukládání. Převody mezi těmito formáty navzájem jsem popsal v kapitole 3.3.

Hlavním tématem této diplomové práce je ale odstranění prokladu (deinterlacing) v digitálních videosekvencích. Přehled metod k odstranění prokladu jsem uvedl v kapitole č.4 a pro lepší přehlednost jsem jednotlivé metody shrnul v tabulce v kapitole 4.4.

Pokud tedy není jiná možnost pořízení videosekvencí a námi nasnímané videosekvence jsou prokládané, musíme použít některou z deinterlacing metod, jejichž popis, implementaci a dosažené výsledky jsem uvedl v kapitole 5. Každou z uvedených metod se více či méně účinně odstraní prokládání obrazu, ovšem vždy za cenu větší či menší ztráty originálních dat, ať už vypuštěním půlsnímku u metody Bob, nebo výpočtem u interpolačních metod. Použitými metodami se upraví obraz tak, že jej lze zobrazovat na LCD a plasmových televizních zobrazovačích a počítačových monitorech. Dle hodnocení jednotlivých metod se jako nejúčinnější ukázaly metody Bob+Weave a interpolační metoda, za nimi lze vyzdvihnout metodu Bob, a teprve potom jednoduché metody Weave s viditelnými hranatými okraji a jako poslední metodu Blending Field s rozmazanými okraji.

Pro lepší názornost a zviditelnění rozdílů mezi účinností jednotlivých metod, jsem k odstranění prokladu použil více metod aplikovaných na statických půlsnímcích získaných z plynulé videosekvence. Každý m-file, který jsem vytvořil v prostředí Matlab, je pro lepší srozumitelnost doplněn řadou komentářů a vysvětlivek.

V další části diplomové práce jsem se zabýval implementováním subjektivně nejlepších metod pro odstranění prokladu (deinterlacing) v jazyce C. Z předešlých kapitol vyplývá, že subjektivně nejlepší výsledky dosahují interpolační metody, jejichž stručný přehled a matematické principy jsem uvedl v kapitole 5.6.

V kapitole 5.8 jsem tedy pro srovnání implementoval dvě metody pro odstranění prokladu, jejichž nejdůležitější části zdrojového kódu, opatřené popisky a vysvětlení funkce, jsem uvedl v kapitolách 5.8.1 a 5.8.2. Pro ilustraci jsem zde také ukázal obraz před a po zpracování.

Dosažením aplikace metod na reálné nekomprimované videosekvence v jazyce C jsem splnil zadání této diplomové práce, ovšem jako možné cesty další práce na této problematice se jeví jednoznačně algoritmizace složitějších a dokonalejších interpolačních metod, doplněné např. i možností natočení vlastní prokládané videosekvence a jejího zpracování těmito metodami.

7. Seznam literatury

[1] WILSON, Dave.*Video Codecs and Pixel Format Definition* [online]. Dostupné z WWW: <http://www.fourcc.org/>.

[2] *What is Deinterlacing? The best method for deinterlace movies* [online]. Dostupné z WWW: <http://www.fourcc.org/>

[3] Recommendation ITU-R BT.601-6. Studio encoding parameters of digital television for standard 4:3 and wide_screen 16:9 aspect ratios. International Telecommunication Union, 2007.

[4] KWOLEK, Jirka. Temné dědictví aneb Analogová „špecifiká“. *PC tuning* [online]. 02.04.2007 [cit.2008-04-10]. Dostupné na WWW: <http://pctuning.tyden.cz/>

[5] JAHODA, R. Komerční formáty videa a TV. *tvfreak* [online]. 14.03.2002 [cit.2008-04-10]. Dostupné na WWW: <http://www.tvfreak.cz/art_doc-B274916590DAB0AFC125727C0059E59E.html?lotus=1&Highlight=0,komer%C4%8Dn%C3%AD,form%C3%A1ty,videa>

[6] JAHODA, R. Formáty obrazu videa. *tvfreak* [online]. 03.10.2001 [cit.2008-04-10]. Dostupné na WWW: <http://www.tvfreak.cz/art_doc-AF3799F3A349EE89C125727C0059F8E5.html>

[7] JAHODA, R. Video a prokládání. tvfreak [online]. 27.08.2004 [cit. 2008-04-11]. Dostupné na WWW: <http://www.tvfreak.cz/art_doc-5E559746593300E2C125727C005965DB.html>

[8] ŘÍČNÝ, Václav.*Videotechnika (přednášky)*[online].Brno: FEKT, VUT v Brně, 2006 [cit. 2008-11-20]. Dostupný z WWW: <https://www.feec.vutbr.cz/et/skripta/urel/Videotechnika_S.pdf>

[9] DOBEŠ, Michal. *Zpracování obrazu a algoritmy v C#*. 1.vyd., Praha: BEN – technická literatura, 2008. 144 s. ISBN 978-80-7300-233-6.

8. Dodatky

8.1. Seznam použitých zkratek a symbolů

$U_Y(Y)$	luminanční signál
f_Y	vzorkovací kmitočet luminančního signálu
C_R, C_B	chrominanční signály
f_{CR}, f_{CB}	vzorkovací kmitočet chrominančních signálů
p_{vz}	počet vzorků obrazu
$C_{4:4:4}$	kapacita celosnímkové paměti
ASCII	THE AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE
DCT	DISCREET COSINUS TRANSFORM
DVB	DIGITAL TELEVISION BROADCASTING
DWT	DISCREET WAVELET TRANSFORM
FOURCC	FOUR CHARACTER CODE
ISO	INTERNATIONAL STANDARDIZING ORGANIZATION
ITU	INTERNATIONAL TELECOMMUNICATION UNION
LCD	LIQUID CRYSTALS DISPLAY
NTSC	NATIONAL TELEVISION SYSTEM COMMITTEE
PAL	PHASE ALTERNATING LINE
PCM	PULZE CODE MODULATION
PLD	PLAZMA DISPLAY
RGB	RED GREEN BLUE
SECAM	SEQUENCES DE COULEURS A MEMOIRE
YUV	YELLOW UNDER VIOLET

8.2. Seznam obrázků

8.3. Seznam tabulek