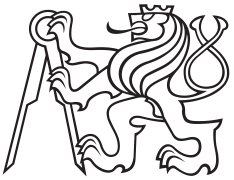
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra radioelektroniky

Objektivní analýza standardu H.265

Michal Janeček

Leden 2015

Vedoucí práce: Ing. Stanislav Vítek PhD.

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra radioelektroniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Michal Janeček**

Studijní program: Komunikace, multimédia a elektronika
Obor: Multimediální technika

Název tématu: **Objektivní analýza standardu H.265**

Pokyny pro vypracování:

Proveďte analýzu standardu H.265 (High Efficiency Video Coding, HEVC). Objektivně porovnejte dostupné implementace tohoto standardu z hlediska účinnosti a dosažených výsledků. Navrhněte deblokační filtr, který bude respektovat požadavky aplikace standardu, např. na bezpečnostní videosekvence.

Seznam odborné literatury:

[1] Yo-Sung Ho, Advanced Video Coding for Next-Generation Multimedia Services, InTeOp, 2013, ISBN: 9535109297
[2] Woods J. W, Multidimensional Signal, Image, and Video Processing and Coding, Academic Press, 2006, ISBN: 0120885166
[3] Bossen F. et al, HEVC Complexity and Implementation Analysis, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 22, No. 12, 2012

Vedoucí: Ing. Stanislav Vítek, Ph.D.

Platnost zadání: do konce letního semestru 2014/2015

V Praze dne 10. 2. 2014

Prohlášení

Prohlašuji, že jsem bakalářskou práci Objektivní analýza standardu H.265 vypracoval samostatně a použil k tomu literaturu, kterou uvádím v seznamu přiloženém k práci. Nemám námitky proti půjčování, zveřejnění a dalšímu využití práce, pokud s tím bude souhlasit katedra radioelektroniky.

Praha, 5. ledna 2015

Abstrakt / Abstract

Tato práce se zabývá objektivní analýzou kvality komprese videa pomocí kodeku H.265 (HEVC) se zaměřením na deblokační filtr obsažený v tomto standardu. Objektivní analýza je provedena na několika implementacích, které jsou v tuto chvíli dostupné a konkrétně se porovnávají metriky PSNR, SSIM a VIFP. Dále je obsahem práce návrh a testování deblokačního filtru v prostředí MATLAB.

Klíčová slova: H.265, HEVC, libde265, matLab, deblokační filtr, objektivní analýza kvality

In this document is described objective quality analysis of video compression standard H.265 (HEVC) with focus on deblocking filter of this standard and comparison of different implementations of H.265 using PSNR, SSIM and VIFP objective analysis tools. This thesis also contain my own implementation of deblocking filter written in MATLAB programming environment.

Keywords: H.265, HEVC, matLab, libde265, deblocking filter, in-loop filter, objective quality assesment

Obsah /

Kapitola 1
Úvod

Komprese videa nabyla v posledních dvou desetiletích na důležitosti a stala se nepostradatelnou součástí všedního života. S roustoucími nároky na kvalitu obrazu rostou také datové nároky, jak na úložiště multimediálního obsahu tak na přenosové sítě pro tento typ dat. Jedním z řešení se snaží být nový a nastupující kompresní standard H.265 (HEVC), pokračovatel velmi rozšířeného a oblíbeného standardu H.264. Nový standard slibuje až dvojnásobné zlepšení kvality komprimovaného videa, při zachování stejných datových nároků. V předkládané práci se budu věnovat objektivnímu posouzení kvality videa s využitím vybraných metrik na zvolených implementacích tohoto standardu.

Cílem první části práce je ukázat na podobnosti a rozdíly vybraných implementací. Toho se budu snažit dosáhnout vytvořením podpůrných funkcí v prostředí MATLAB a tyto funkce mi umožní změřit a zpracovat výsledné hodnoty měření metrik objektivní obrazové kvality videa.

V druhé části práce se budu věnovat deblokačnímu filtru ve standardu H.265. Klíčovým bude jeho vliv na kvalitu obrazu se zaměřením na detekci hran v obraze. Poté se pokusím navrhnout vlastní deblokační filtr v prostředí MATLAB.

Následující kapitoly vymezí základní pojmy nutné pro analýzu tohoto nového standardu komprese. Přičemž začnu volbou vhodných implementací, porovnám časovou náročnost komprese a přiblížím problematiku metrik objektivního měření obrazové kvality videa. Výsledky porovnám v poslední kapitole první části.

Jádrem druhé části bude, kromě zjištění vlivu deblokačního filtru na kvalitu obrazu, získání dat potřebných pro můj vlastní návrh filtru a realizace všech nezbytných funkcí pro úspěšné zpracování videa.

Kapitola 2
Použité implementace standardu H.265 (HEVC)

Pro účely této práce jsem zvolil tři implementace standardu H.265 (HEVC), volně dostupných v době začátku práce na tomto projektu. High Efficiency Video Coding byl jako nový standard videoformátu schválen před dvěma lety. Oproti svému předchůdci H.264 výrazně snižuje datový tok, přičemž kvalita obrazu by měla zůstat zachována.

Vybrané tři implementace jsem využil ke kódování testovacích videosekvencí, pro dekódování a rekonstrukci videa z komprimovaných souborů je ve všech případech použit dekodér referenční implementace HM12.1 [9].

2.1 Referenční implementace skupiny JCT-VC HM12.1

Referenční implementace skupiny JCT-VC je společným projektem ITU-T Video Coding Experts Group a ISO/IEC Moving Picture Experts Group a předpokládá se, že se v budoucnu projeví schopnosti a možnosti tohoto nového standardu pro kompresi videa.

2.2 x265

Další implementací je x265[10] , která je open source verzí standardu H.265, která má být nástupcem úspěšné open source implementace x264 standardu H.264. Projekt x265 je stejně jako jeho předchůdce vyvíjen v rámci většího projektu VideoLAN a je také uvolněn pod licencí General Public Licence.

2.3 DivX HEVC kodek

Poslední použitou implementací je DivX HEVC kodek[12] , který je proprietární implementací standardu H.265 společnosti DivX, je k dispozici zdarma jako plugin k DivX converter softwaru.

2.4 Konfigurace a nastavení použitých implementací

Porovnání nastavení a konfigurace jednotlivých implementací je zajímavé zejména pro HM12.1 a x265, protože v DivX HEVC kodéru jsou možnosti nastavení omezeny na rozlišení, snímkovací frekvenci a cílový bitový tok.

Pro HM12.1 je použita upravená základní konfigurace profilu main, viz příloha config.txt. Zde je nastavena pevná struktura GOP, vypnutý PCM mód, stejně jako kvantizační matice a vypnuté nastavením SliceMode. Pro určení významu jednotlivých nastavení a parametrů jsem vycházel z HM software manuálu obsaženém v kodeku HM12.1 [15].

U kodeku x265 jsem vycházel z přednastavené konfigurace s označením medium, která nejvíce odpovídá konfiguraci pro HM12.1 zmíněné výše. Bylo ovšem nutné několik parametrů změnit [14].

- –preset medium ... nastavení analogické k main tier nastavení v kodeku HM12.1
- –input-res ... rozlišení videa
- –fps ... snímkovací frekvence
- –f ... počet snímků
- –tu-inter-depth 1-4 ... určuje minimální velikost predikčních bloků, v HM12.1 QuadtreeTUMaxDepthInter :3
- –tu-intra-depth 1-4 ... určuje minimální velikost transformačních bloků, v HM12.1 QuadtreeTUMaxDepthIntra :3
- –b-adapt 0 ... vypne adaptabilní GOP strukturu
- –bframes 3 ... nastaví pevnou GOP strukturu
- –merange 64 ... určí rozsah hledání pohybových vektorů na plný rozsah, v HM12.1 SearchRange : 64
- –rd 1 ... nastavuje zjednodušenou RDO analýzu
- –me 3 ... přepne na hvězdicovou metodu hledání pohybových vektorů
- –tskip-fast ... zapíná zrychlené přeskakování transformačních jednotek stejně jako je tomu v HM12.1
- –bitrate ... cílový bitový tok videa
- -o ... název výstupního videosouboru

Kapitola 3
Porovnání časové náročnosti komprese pro všechny implementace

3.1 Metoda měření časové náročnosti komprese

Jako první jsem srovnával časovou náročnost kódování videa pro jednotlivé implementace kodeku v závislosti na rozlišení a cílovém bitovém toku komprimovaného videa. Jedná se pouze o orientační relativní srovnání časové náročnosti. Měření probíhalo na počítači pomocí příkazu time, pro orientaci udávám i konfiguraci počítače:

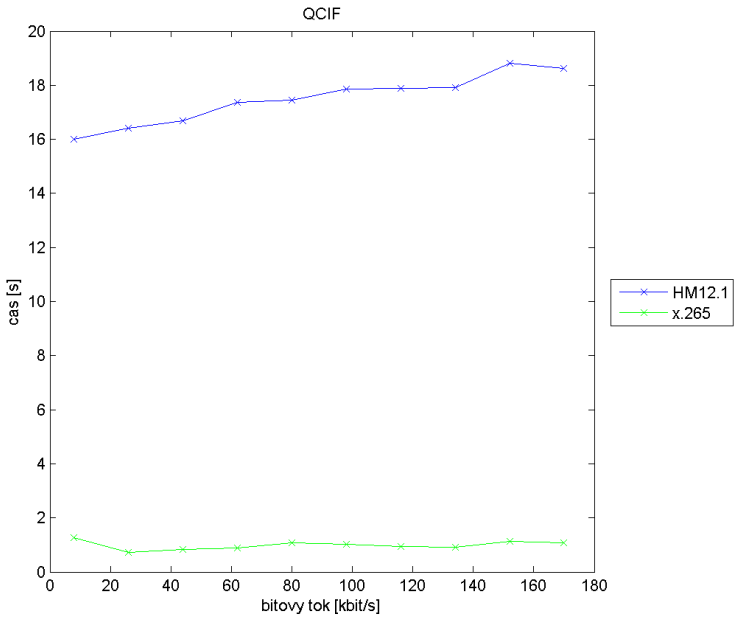
- intel core 2 duo
- 4 GB DDR2 RAM
- Windows 7 64bit Home editon SP2

U implementace HM12.1 jsem bohužel nebyl schopen zprovoznit běh na více jádrech, což alespoň částečně kompenzuji dělením výsledných časů pro kompresi dvěma. Tento přístup samozřejmě přináší velkou nepřesnost, ale vzhledem k rozdílu výsledků HM12.1 a zbylých dvou implementací to cíl měření nenarušuje. Výsledky jsou zpracované v další sekci.

3.2 Naměřené tabulky a grafy

Bitový tok [kbit/s]	8	26	44	62	80	98	116	134	152	170
HM12.1 [s]	15,9	16,4	16,6	17,3	17,4	17,8	17,8	17,9	18,8	18,9
x.265 [s]	1,2	0,7	0,8	0,9	1,0	1,0	0,9	0,9	1,1	1,0

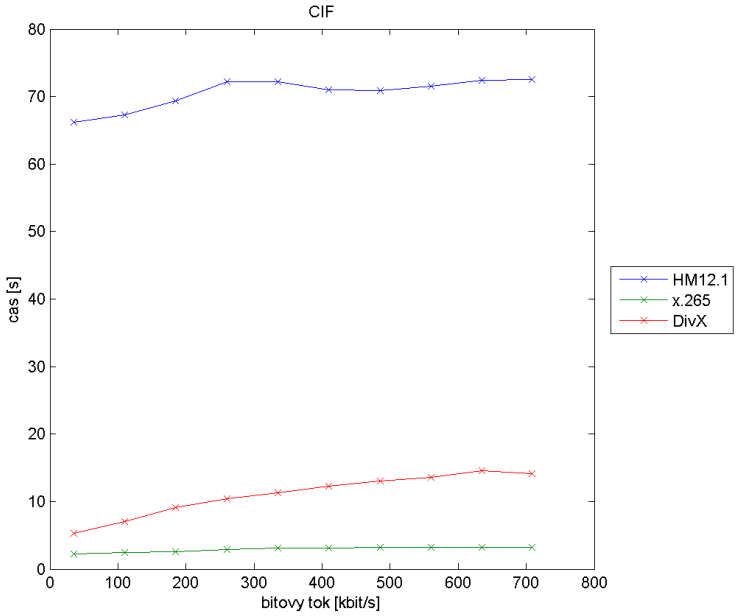
Tabulka 3.1. Tabulka měřených časů komprese pro rozlišení QCIF



Obrázek 3.1. Graf pro rozlišení QCIF

Bitový tok [kbit/s]	35	110	185	260	335	410	485	560	639	709
HM12.1 [s]	66,1	67,2	69,3	72,2	72,1	71,0	70,9	71,5	72,4	72,5
x.265 [s]	2,2	2,4	2,5	2,8	3,1	3,1	3,1	3,1	3,1	3,2
DivX [s]	5,3	7,0	9,1	10,3	11,3	12,3	13,0	13,5	14,6	14,1

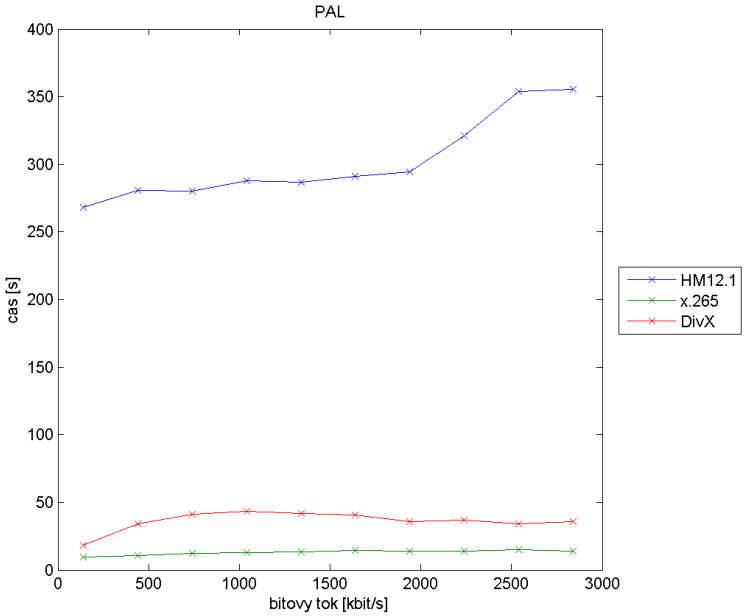
Tabulka 3.2. Tabulka měřených časů komprese pro rozlišení CIF



Obrázek 3.2. Graf pro rozlišení CIF

Bitový tok [kbit/s]	142	441	741	1041	1341	1640	1940	2240	2540	2839
HM12.1 [s]	268	280	280	288	286	291	294	321	354	355
x.265 [s]	9,8	10,8	12,2	12,8	13,5	14,2	13,7	13,8	14,9	14,0
DivX [s]	18,5	33,9	41,2	43,4	41,7	40,7	35,6	36,9	33,9	35,8

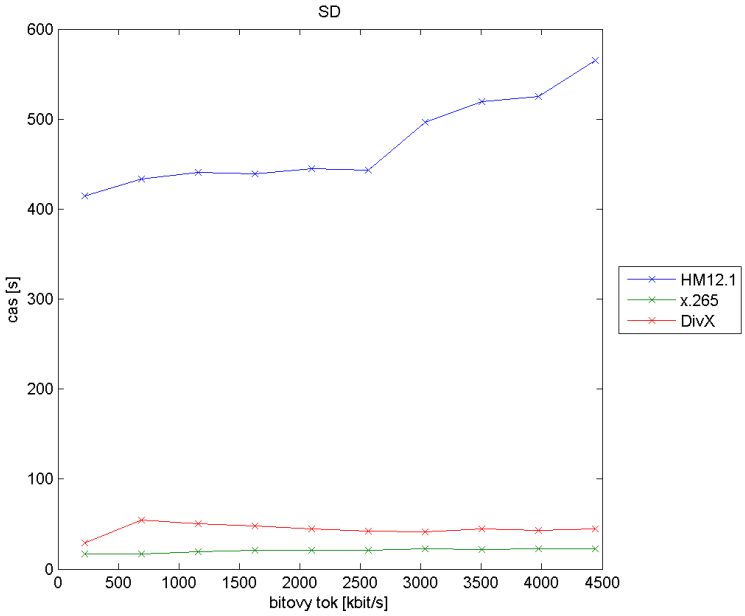
Tabulka 3.3. Tabulka měřených časů komprese pro rozlišení PAL



Obrázek 3.3. Graf pro rozlišení PAL

Bitový tok [kbit/s]	222	690	1159	1628	2096	2565	3034	3502	3971	4440
HM12.1 [s]	414	433	441	439	445	443	496	519	525	565
x.265 [s]	16,4	16,7	18,9	20,8	20,9	21,2	22,6	21,7	22,3	22,2
DivX [s]	29,1	54,5	50,4	48,0	44,4	42,3	41,4	44,4	43,0	44,9

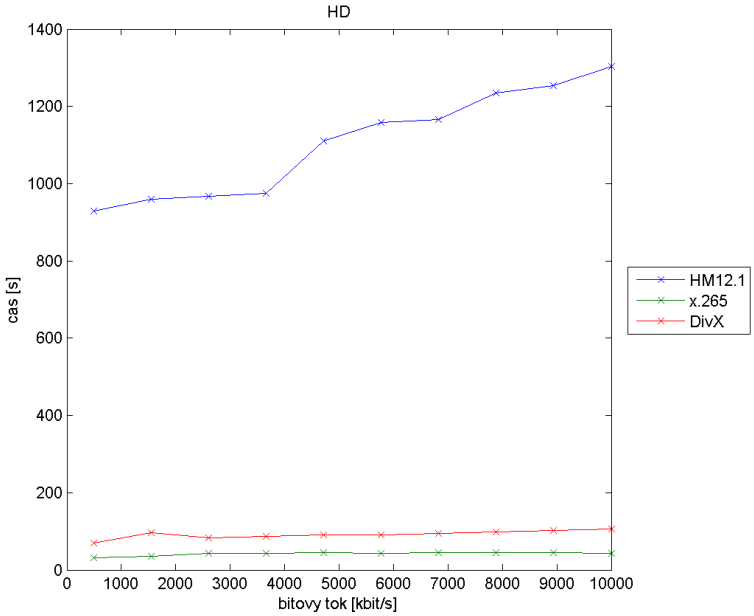
Tabulka 3.4. Tabulka měřených časů komprese pro rozlišení SD



Obrázek 3.4. Graf pro rozlišení SD

Bitový tok [kbit/s]	500	1555	2611	3666	4722	5777	6833	7888	8944	10000
HM12.1 [s]	929	960	967	975	1111	1159	1166	1234	1254	1304
x.265 [s]	31,8	36,2	42,3	42,4	44,5	43,1	45,0	45,0	44,3	43,7
DivX [s]	68,8	95,6	82,2	86,0	89,8	91,3	94,2	98,3	101,9	105,3

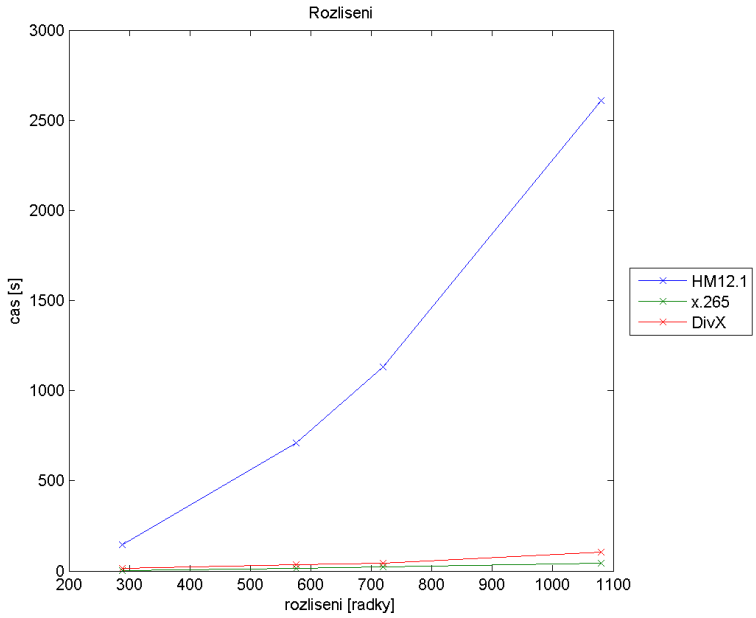
Tabulka 3.5. Tabulka měřených časů komprese pro rozlišení HD



Obrázek 3.5. Graf pro rozlišení HD

Rozlišení	QCIF	CIF	PAL	SD	HD
HM12.1 [s]	37,2	145,1	711	1130	2609
x.265 [s]	1,0	3,2	14,0	22,2	43,7
DivX [s]	-	14,1	35,8	44,9	105,3

Tabulka 3.6. Tabulka měřených časů komprese všech rozlišení



Obrázek 3.6. Graf měřených časů komprese všech rozlišení

3.3 Hodnocení výsledků

I přes velmi jednoduchou metodiku měření a malý počet naměřených hodnot se z výsledků a grafů určité závěry vyvodit dají. V první řadě je zřejmé, že referenční kodek HM12.1 rozhodně není vhodný do produkčního prostředí. Časová náročnost komprese by uživateli nevyhovovala. Zůstává tedy pouze ukázkou veškerých schopností nového standardu komprese H.265, což dokládají i nejrozsáhlejší konfigurační možnosti ze všech testovaných implementací.

Dvě další verze kodeku si vedly téměř stejně dobře a to i díky vícevláknovému zpracování, které výrazně urychlí kompresi videa na dnes již velmi běžných multi-core procesorech. Zajímavá je i závislost časové náročnosti komprese na rozlišení videa, kdy pro kodeky DivX a x265 náročnost nenarůstá zdaleka takovým tempem, jakým u HM12.1. Podobně neúměrně nenarůstá i s počtem obrazových bodů videa.

Kapitola 4
Použité metriky objektivní obrazové kvality

Dalším podstatným krokem mé práce bylo srovnání kvality dekódovaných video sekvencí pomocí objektivních metrik. Největší skupinou všech objektivních kritériíjsou plně referenční metriky. Referenční obraz a zkreslený obraz lze porovnávat z hlediska celé řady možných přístupů.

4.1 Dělení metrik

Metriky zaměřené na srovnání pixelů reference a zkresleného obrazu (Pixel-based metrics) patří mezi velmi používaná kritéria hlavně kvůli jednoduchému programování. Jejich největším nedostatkem je, že se neslučují s lidským vnímáním obrazu, pro které nejsou při porovnávání všechny pixely stejně podstatné. Mezi tyto metriky patří MSE, SRN, PSNR a UIQI. Dále můžeme využívat metriky založené na měření korelačních funkcí, které místo rozdílu počítají podobnost obrazů (CM, CCM, CDM). Třetí skupinou metrik jsou ty, jež hodnotí kvalitu hranic objektu (PDM, ESM). Lidské oko vnímá velmi citlivě chyby na hranicích objektů a pro kvalitní obraz je správné posouzení hranic zásadní. Další velká skupina metrik je založena na vizuálním systému člověka (HVS-based metrics). Všechny tyto metriky využívají lidské psycho-vizuální modely, ty jsou ale vždy zjednodušené, protože prozatím není možné vystihnout přesné chování lidského zraku. Do této kategorie patří například AE, HVS-MSE, SSIM a další. Jiným kritériem pro rozdělení metrik je měření spektrální vzdálenosti. Tyto metriky porovnávají spektra měřeného a referenčního objektu a snaží se vyvodit závěry o jeho kvalitě (SMD, SPD, SPMD a jiné). Metrika založená na diskrétní cosinové transformaci (DCT metrika) je další možností nahlížení na kvalitu obrazu. Využívá skutečnosti, že lidský zrak nehodnotí stejně všechny frekvence spektra obrazu. Poměrně novou metrikou je VIF, která je založena na informacích v obrázcích. Ta počítá informace, jež může získat mozek z obrazu v ideálním případě a porovnává je s informacemi ze zkresleného obrazu [5].

Pro účely této práce jsem vybral následující objektivní metriky měření kvality: PSNR (Peak Signal to Noise Ration), SSIM (Structural Similarity Index) a VIF (VisualInformation Fidelity).

4.2 PSNR

PSNR je metrika, která porovnává referenční a měřené video na bázi pixelů. Je definována jako poměr nejvyšší hodnoty jasu, která se ve snímku vyskytuje, a střední kvadratické chyby. Jde o jednu ze základních objektivních metrik pro porovnávání kvality obrazu, ale její výsledky nejsou příliš korelovány se subjektivními testy. Čím vyšší je hodnota PSNR, tím je podle této metriky obraz kvalitnější. Obecná rovnice pro výpočet PSNR je ukázána níže [5].

Rk – referenční obráze
Dk – zkreslený obrázek
i, j – indexy pixelů (řádky, sloupce)

$$PSNR = \frac{10.log(max(R_k^2(i,j)))}{\frac{1}{K}\frac{1}{N^2}\sum_{i,j=0}^{N} -1 \sum_{k=1}^{K}[R_k(i,j) - D_k(i,j)]^2} \quad (1)$$

4.3 SSIM

Oproti PSNR tato metrika více odpovídá skutečnému vnímání obrazu lidským zrakovým ústrojím (HVS). Zaměřuje se zejména na změny strukturální informace v obrazu. Na rozdíl od PSNR se v SSIM také neporovnávají jednotlivé pixely referenčního a měřeného snímku, ale ty se porovnávají pro okno o určené velikosti, které je posouváno přes celý snímek. Výsledné hodnoty jsou v rozsahu 0 až 1, čím blíže je výsledný index 1, tím jsou si referenční a měřený snímek podobnější a také kvalitnější [5].

4.4 VIF

Ze tří zde uvedených metrik se tato snaží co nejblíže napodobit vnímání obrazu lidským okem. Porovnává množství informací obsažených ve snímku, které je lidský vizuální systém schopen získat. Vzhledem k tomu, že používá rozklad obrazu pomocí wavelet transformace na jednotlivé složky a ty poté porovnává, je VIF matematicky nejsložitější ze zde uvedených metrik. V mém měření je ale použita zjednodušená verze této metriky VIFP, která porovnává snímky na bázi pixelů a nepoužívá tedy wavelet transformaci, její přesnost je tím ale vůči originální verzi omezena. Výsledky této metriky vycházejí v rozsahu 0 až 1, kde 1 znamená totožné snímky [5].

4.5 Použitý nástroj pro měření objektivní kvality a video sekvence

Pro měření jsem použil nástroj Video Quality Measurement Tool - VQMT 1.1 jehož autorem je Philippe Hanhart [3]. Umožňuje výpočet objektivních metrik z příkazové řádky a tím pádem i efektivní měření většího množství dat.

V tomto případě bylo použito video města pořízené z ptačí perspektivy s otáčivým pohybem doplněným pohybem aut na silnici v dolní části obrazu 4.1. Obraz má velký počet detailů, ale pohyb je spíše pomalý a nemění směr.

Obrázek 4.1. Ukázka testovací videosekvence

Jednotlivé metriky byly měřeny na padesáti snímkovém vzorku videa s rozlišením od QCIF do HD s tím, že u rozlišení QCIF, CIF a PAL bylo horizontální rozlišení upraveno pro poměr stran 16:9, jako má originální videosekvence. Konkrétní rozlišení jsou v tabulce 4.1 níže.

Formát	QCIF	CIF	PAL	SD	HD
Rozlišení [px]	144x256	288x512	576x1024	720x1280	1080x1920

Tabulka 4.1. Tabulka rozlišení testovaných videí

Videosekvence byly komprimovány jednotlivými implementacemi kodeku s cílovými bitovými toky od 100kbit/s až po 10000kbit/s pro rozlišení HD s logaritmickými rozestupy. Pro nižší rozlišení jsou cílové bitové toky sníženy poměrem stejným, jako je poměr daného rozlišení a plného HD rozlišení, viz tabulka 4.2.

Rozlišení	Bitový tok [kbit/s]											
HD	100	151	231	351	533	811	1232	1873	2848	4328	6579	10000
SD	44	67	102	155	236	360	547	831	1246	1921	2921	4440
PAL	28	43	65	99	151	230	350	532	808	1229	1868	2839
CIF	7	10	16	24	37	57	87	133	202	307	467	709
QCIF	1	2	3	5	9	13	20	31	48	73	111	170

Tabulka 4.2. Tabulka cílových bitových toků pro jednotlivá rozlišení.

Kapitola 5
Srovnání objektivní kvality obrazu jednotlivých implementací

5.1 Konfigurace komprese pro měření objektivní obrazové kavlity

Pro další postup bylo třeba stanovit podmínky testování. Videosekvence byly komprimovány jednotlivými implementacemi kodeku s cílovými bitovými toky od 100kbit/s až po 10000kbit/s pro rozlišení HD s logaritmickými rozestupy. Pro nižší rozlišení jsou cílové bitové toky sníženy poměrem stejným jako je poměr daného rozlišení a plného HD rozlišení. Těchto cílových hodnot samozřejmě nelze v reálném prostředí dosáhnout a zejména v nižších hodnotách se skutečný bitový tok může výrazně lišit od cílového. V každé implementaci je zároveň jiná přesnost této funkce a to velmi ovlivňuje výsledky měření. Kvůli těmto odchylkám je nutné brát dosažené výsledky jako orientační.

5.2 Zpracování výsledků

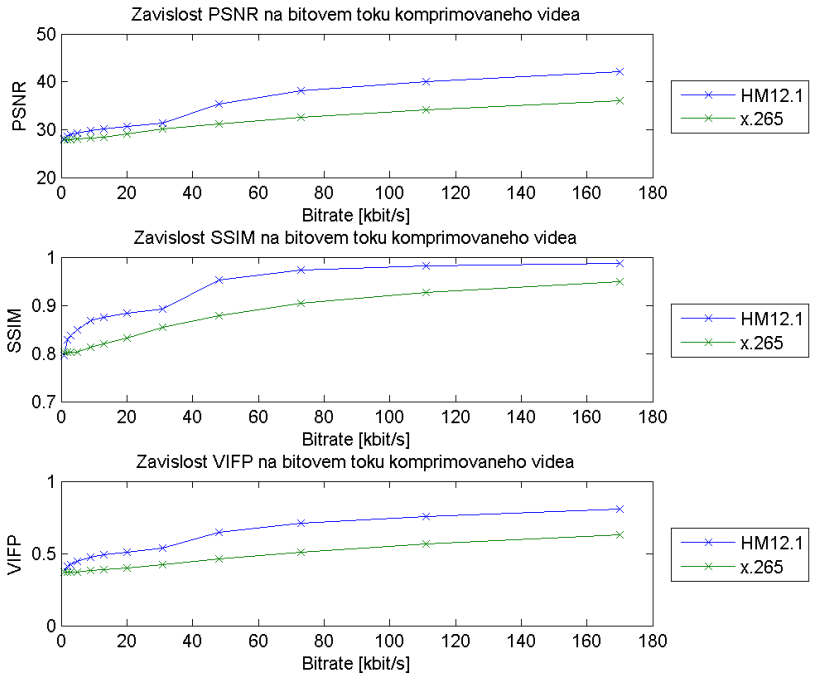
Pro zpracování originální videosekvence byl nejdříve použit nástroj encode_config.m, který zajistil přeškálování a zkrácení videa na výše zmíněná rozlišení a definovanou délku. V našem případě je délka 50 snímků, čímž je dosažen kompromis mezi přesností objektivních metrik (jsou průměrovány přes všechny snímky) a časovou náročností samotné komprese, která je v případě implementace HM12.1 a velkých rozlišení značná. Nástroj encode_config.m dále vygeneruje soubory ve formátu .bat, které automatizují kompresi s různými cílovými bitovými toky. Dále pro implementaci HM12.1 vytvoří potřebné konfigurační soubory ve formátu .conf a také soubory pro automatizaci měření jednotlivých metrik opět ve formátu .bat. Program také zajistí souborovou strukturu pro přehlednost jednotlivých měření a rozlišení, jeho poslední funkcí je vytvoření protokolu ve formátu .txt se všemi podstatnými informacemi o vygenerované měřící sadě.

Po kompresi všech videosekvencí pomocí souborů encode.bat (pro každé rozlišení je zvlášť) a výpočtu třech objektivních metrik pomocí souborů measure.bat

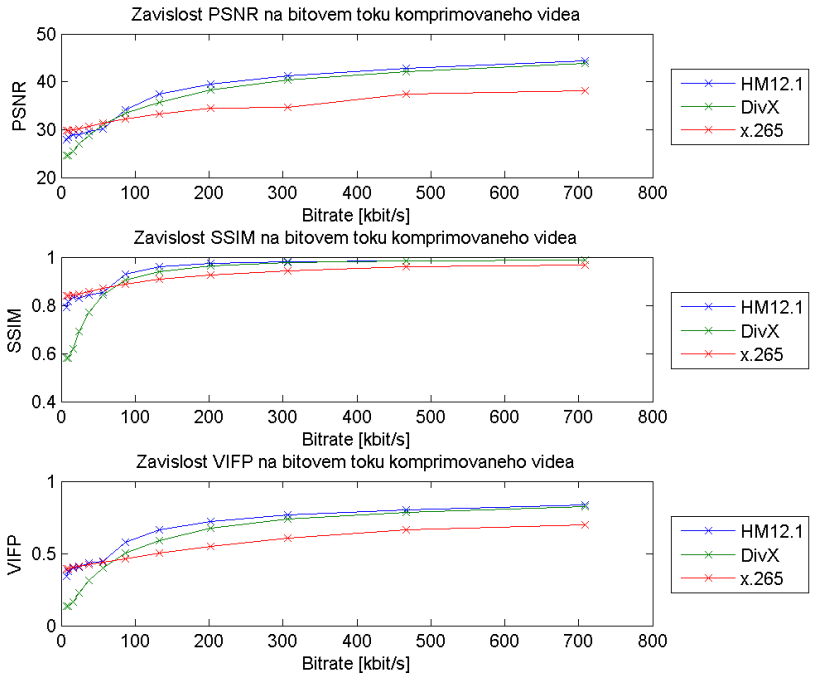
(také je pro každé rozlišení zvlášť) je potřeba zpracovat všechny výsledky do čitelné a přehledné formy. Výstupem VQMT1.1 je pokaždé tabulka ve formátu .csv, ve které je vypočtena pouze jedna metrika pro jedno konkrétní komprimované video, což znamená, že při našem konkrétním měření bylo vygenerováno 108 tabulek, které bylo potřeba zpracovat.

Pro zpracování výsledků měření jsem navrhl nástroj processResults.m, který v první řadě setřídí a sloučí data ze všech tabulek do jediné struktury. Ta je následně i uložena ve formátu .mat pro jakékoliv další využití. Dále tento nástroj vygeneruje grafy závislosti jednotlivých metrik na bitovém toku pro všechny tři použité implementace kodeku. Výjimkou je v tomto případě rozlišení QCIF, kde jsou porovnány výsledky pouze pro implementace x.265 a HM12.1, protože kodek DivX nepodporuje takto nízká rozlišení.

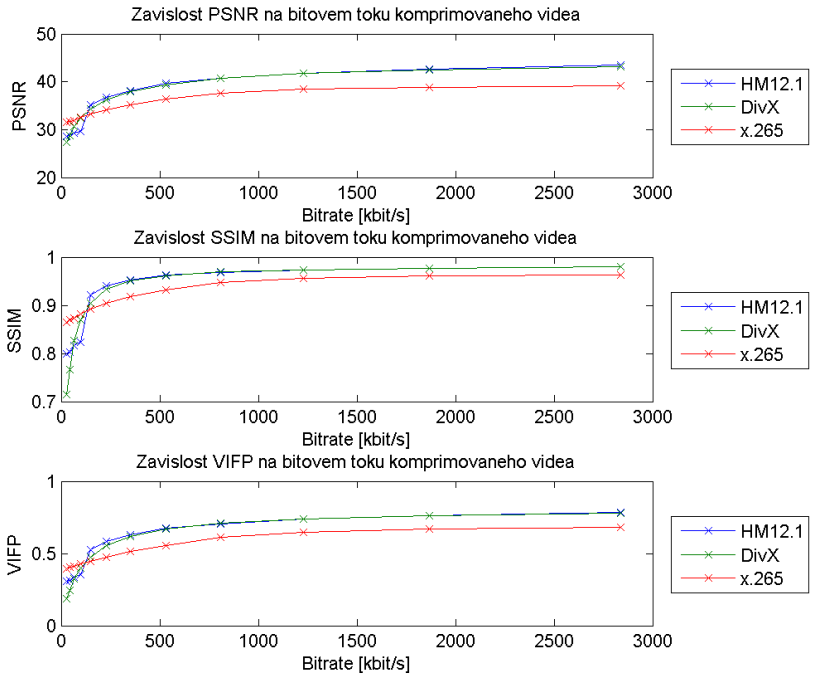
5.3 Grafy výsledků



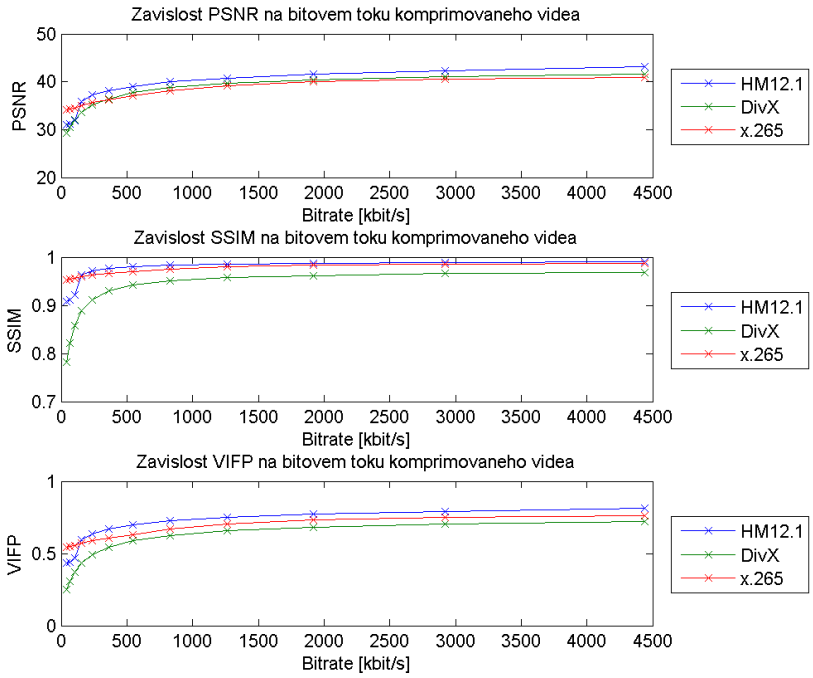
Obrázek 5.1. Graf pro rozlišení QCIF



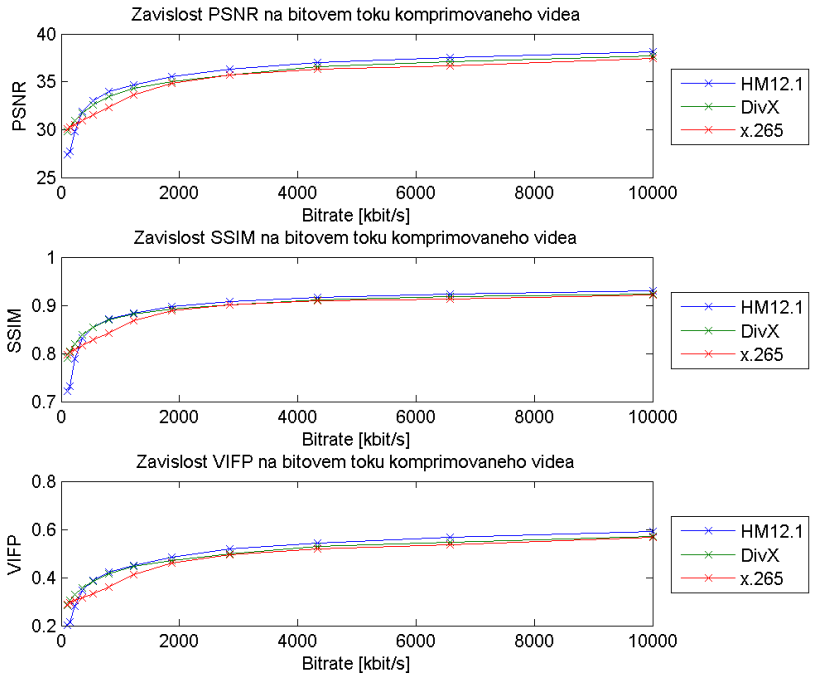
Obrázek 5.2. Graf pro rozlišení CIF



Obrázek 5.3. Graf pro rozlišení PAL



Obrázek 5.4. Graf pro rozlišení SD



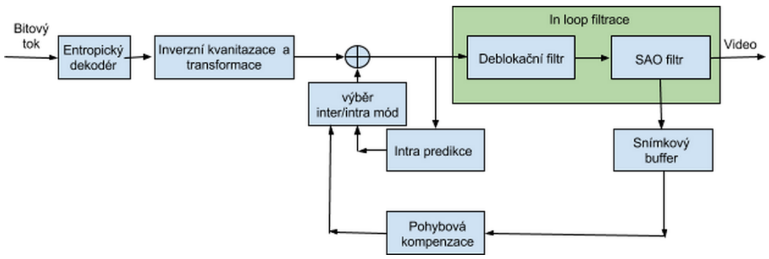
Obrázek 5.5. Graf pro rozlišení HD

5.4 Hodnocení výsledků

V první řadě je z naměřených hodnot zřejmé, že se v objektivních výsledcích všechny tři implementace s vyšším rozlišením a s roustoucími cílovými bitovými toky stále více přibližují. Je také patrné, že implementace x.265 má na nižších rozlišeních celkově nižší hodnoty než zbylé dva kodeky. Měření pro velmi nízké hodnoty cílového bitového toku jsou bohužel výrazně ovlivněny možnostmi jednotlivých implementací skutečně dosáhnout požadovaných hodnot. Na příklad pro rozlišení HD a pro nejnižší cílový bitový tok 100kbit/s dosáhl kodek DivX hodnoty 2,5x vyšší, HM12.1 1,4x vyšší a x.265 hodnoty 3,6x vyšší. Pro rozlišení CIF a cílový bitový tok 7kbit/s dosáhl kodek DivX hodnoty 2x vyšší, HM12.1 4,3x vyšší a x.265 dokonce 5,7x vyšší, proto je nutné brát tyto nezanedbatelné nepřesnosti v úvahu. I když pro mé měření má tento jev negativní vliv, učitou vypovídací hodnotu si zachovává, a i přesto výsledky měření poskytují informace o srovnání všech tří kodeků.

Kapitola 6
Deblokační filtr v H.265

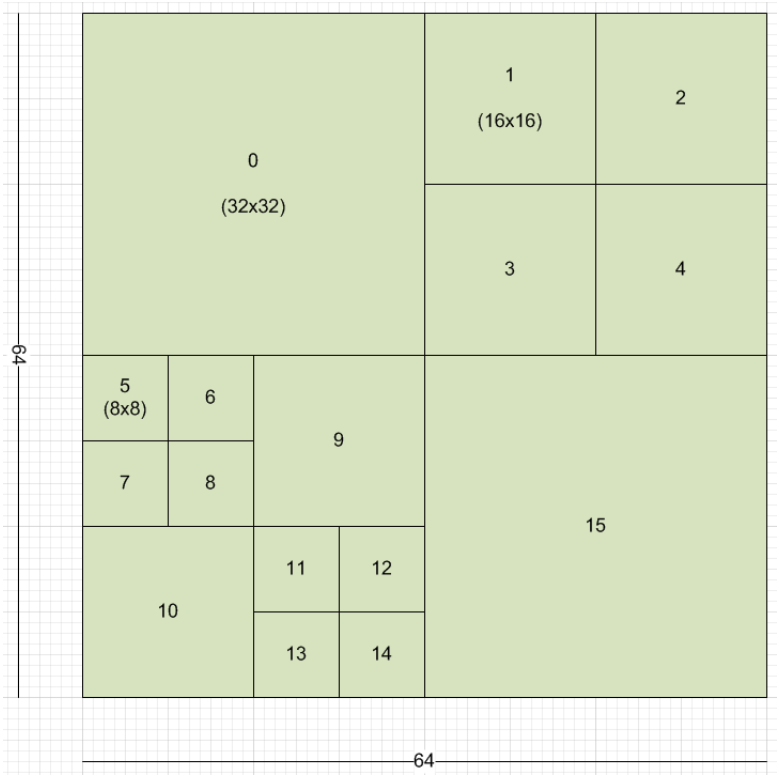
Funkcí deblokačního filtru ve standardu H.265 je, stejně jako u předchozích verzí kodeku, odstranění reliktů po blokovém zpracování videa. Relikty výrazně ovlivňují subjektivní kvalitu výsledného videa bez ztrát obrazové informace, tedy rozmazání přirozených hran v obraze. K dosažení tohoto cíle je v deblokačním filtru navržena rozhodovací logika umožňující omezení dopadů filtrace na ztrátu obrazové informace. Deblokační filtr je aplikován na rekonstruované snímky ve fázi nazývané In Loop filtering společně s SAO filtrem, což je nová součást kodeku, která se poprvé objevuje právě ve standardu H.265 [1].



Obrázek 6.1. Schéma umístění In Loop filtrace [4]

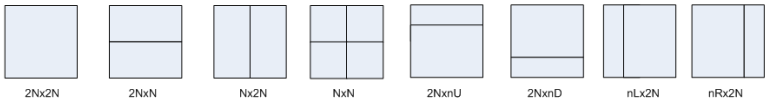
6.1 Rozhodovací proces filtru

Deblokační filtr se aplikuje pouze na hranicích mřížky 8x8 bodů pro jasovou (luma) i barevné (chroma) složky, v čemž se liší od standardu H.264, kde byla tato mřížka jemnější, tj 4x4 bodů. Tím je dosaženo nižší výpočetní náročnosti dekodéru. Další podmínkou je, že filtrovány jsou pouze hranice transformačních bloků (TU) a predikčních bloků (PU). Ve standardu H.265 je obraz dělen na základní jednotky CTB (Coding Tree Block) o velikostech 64x64, 32x32 nebo 16x16 obrazových bodů. Každý CTB segment je dále dělen na kódovací jednotky CU (Coding Unit), jejichž velikost může být kromě 64x64, 32x32 a 16x16 bodů ještě 8x8 obrazových bodů, jak je vidět na obrázku 6.2.



Obrázek 6.2. Příklad dělení CTB[13]

Z každého CU je pro účely zakódování residuálního obrazového signálu odvozena jedna či více transformačních jednotek TU, které mohou nabývat velikostí od 4x4 obrazových bodů. Pro pohybovou predikci je z jednotek CU odvozen další typ bloků a to predikční bloky PU dělením podle jednoho z následujících způsobů: 2Nx2N, 2NxN, Nx2N, NxN, 2NxnU, 2NxnD, nLx2N, nRx2N. Kde N je polovina délky strany CU a n její čtvrtina. Toto dělení je více zřejmé z následujícího obrázku 6.3.



Obrázek 6.3. Módy dělení CU na PU[13]

Dále je rozhodnuto o síle aplikovaného filtru Bs (Boundary filter strength), kde Bs nabývá hodnot 6.1 [2].

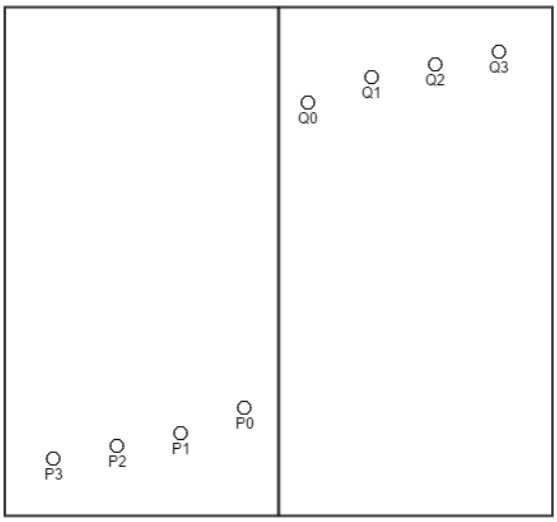
Bs	0	1	2
Filtrace	žádná	slabá	silná

Tabulka 6.1. Síla aplikovaného filtru podle parametru Bs

1) Alespoň jeden ye sousedních bloků je intra predikován ... Bs = 2

2) Hranice je mezi dvěma TU a alespoň jeden má nenulový residuální koeficient ... Bs = 1
3) Absolutní rozdíl mezi korespondujícími složkami pohybových vektorů dvou sousedních bloků je větší či roven 1 (integer pixels) ... Bs = 1
4) Predikce pohybu pro dva sousední bloky odpovídá dvou různým referenčním blokům, nebo je počet pohybových vektorů pro oba bloky různý (bipredikce/unipredikce) ... Bs = 1
5) Jinak ... Bs = 0

A nakonec podle variace obrazového signálu je rozhodnuto o parametrech aplikovaného filtru 6.4.



Obrázek 6.4. Ukázka variace signálu na deblokované hranici [2]

Pro sílu aplikovaného filtru je nutné ještě projít následujícím rozhodovacím procesem. V obrázku níže je ukázána hranice mezi sousedními bloky P a Q, pro zjednodušení jsou výpočty podmínek prováděny pouze na vzorcích první a čtvrté řady (pro horizontální filtrování jsou to první a čtvrtý sloupec), viz 6.5.

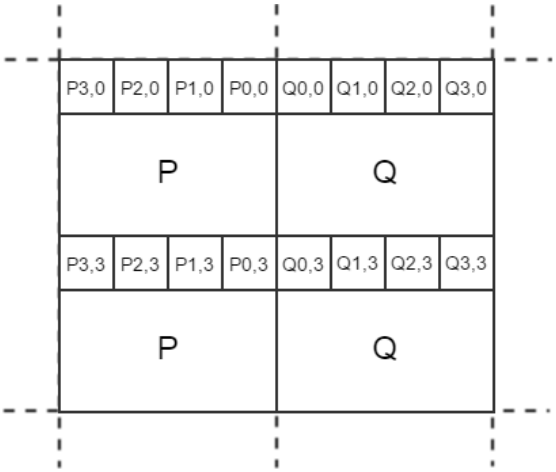
V první řadě je otestována diskontinuita jasového signálu na obou stranách hranice.

$$|P_{20}+2.P_{10}+P_{00}|+|P_{23}+2.P_{13}+P_{03}|+|Q_{20}+2.Q_{10}+Q_{00}|+|Q_{23}+2.Q_{13}+Q_{03}| > \beta \quad (1)$$

Dále je rozhodnuto o síle aplikovaného filtru.

$$proi = 1, 3$$

$$|P_{2i} - 2.P_{1i} + P_{0i}| + |Q_{2i} - 2.Q_{1i} + Q_{0i}| < \beta/8 \quad (2)$$



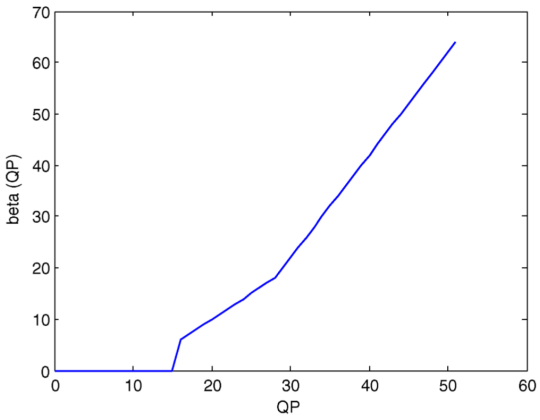
Obrázek 6.5. Schéma indexování obrazových bodů na filtrované hranici [2]

$$|P_{3i} - P_{0i}| + |Q_{0i} - Q_{3i}| < \beta/8 \tag{3}$$

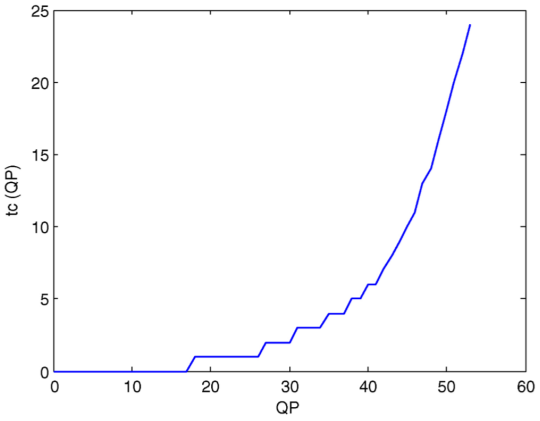
$$|P_{0i} - Q_{0i}| < 2,5.tc \tag{4}$$

Pokud jsou tyto tři podmínky splněny, pak je použito silné filtrování, v opačném případě je použito normální.

Hodnoty beta a tc jsou pevně dané a zajišťují vazbu mezi rozhodovacím procesem filtru a kvantizačním parametrem QP (průměrem QP pro blok P a Q), respektive cílovým bitovým tokem komprimovaného videa, viz následující grafy 6.6 a 6.7.



Obrázek 6.6. Závislost parametru beta na kvantizačním parametru [2]



Obrázek 6.7. Závislost parametru tc na kvantizačním parametru [2]

Pro určení významu deblokačního filtru a In Loop filtrování pro kvalitu obrazu obecně a jeho následné zpracování, jsem komprimoval určený vzorek videa ve všech možných nastaveních In Loop filtrování. Konkrétně: se zapnutým deblokačním a SAO filtrem, následně bez SAO filtru, bez deblokačního filtru, s oběma filtry deaktivovanými a nakonec s deblokačním filtrem ovlivněným nastavením odsazení parametrů beta a tc (v tomto případě jejich maximálních a minimálních hodnot). Porovnával jsem změny PSNR, SSIM a VIFP metrik, viz tabulka 6.2 níže a také změny detekce hran pomocí Cannyho hranové detekce (Canny Edge Detection), toto měření je provedeno jen pro první čtyři případy, tedy bez nastavení parametrů beta a tc. Porovnání nalezených hran jsem řešil zjištěním rozdílu v počtech pixelů určených jako hrana mezi referenčním (nekomprimovaným) a zkoumaným videem a jejich průměrováním přes všechny snímky testovaného videa, výsledky jsou v další tabulce 6.3.

Nastavení	PSNR	SSIM	VIFP
Standardní	23,755	0,6275	0,174
DB vypnutý	23,633	0,617	0.167
SAO vypnuté	23,753	0,623	0,171
DB a SAO vypnuté	23,621	0,618	0,169
beta = 6	23,719	0,624	0,171
beta = -6	23,682	0,618	0,166
tc = 6	23,735	0,624	0,171
tc = -6	23,686	0,619	0,169
beta, tc = 6	23,842	0,631	0,177
beta, tc = -6	23,724	0,622	0,170

Tabulka 6.2. Tabulka naměřených PSNR, SSIM a VIFP hodnot

Nastavení	Rozdíl [px]
Standardní	106420
DB vypnutý	106600
SAO vypnuté	106900
DB a SAO vypnuté	105180

Tabulka 6.3. Tabulka naměřených rozdílů detekovaných hran

Následují ukázky snímků s detekovanými hranami pro představu rozdílu mezi jednotlivými konfiguracemi kodeku. Je z nich zřejmé, že rozdíl mezi detekovanými hranami na videu se standardní konfigurací (deblokačním filtrem aktivním) na obrázku 6.8 a na tom nefiltrovaném 6.9 je skutečně minimální. To odpovídá i naměřeným hodnotám a lze tedy usoudit, že deblokační filtr nemá žádný nebo jen velmi malý vliv na fuknci detekce hran.

Obrázek 6.8. Detekované hrany při zapnutém deblokačním filtru.

Obrázek 6.9. Detekované hrany při vypnutém deblokačním filtru.

Kapitola 7
Návrh deblokačního filtru v prostředí MATLAB

Tato kapitola se věnuje návrhu deblokačního filtru dle standardu H.265 [1] v prostředí MATLAB. Základním problémem tohoto návrhu je export dat potřebných pro rozhodovací proces filtru z dekodéru komprimovaného bitového toku, čemuž je dále věnována podkapitola 7.1. Dalším krokem je načtení těchto deblokačních dat a samotného videa do prostředí MATLAB a nakonec filtrace obrazu.

7.1 Export dat z dekodéru

Prvním a nejsložitějším problémem návrhu deblokačního filtru je získání veškerých dat z dekodéru. Vzhledem ke komplexnosti referenčního dekodéru HM12.1 jsem použil kodek libde265 [11], který je psaný neobjektově a neimplementuje veškeré funkce referenčního kodeku HM12.1, zároveň ale neomezuje funkci deblokačního filtru.

V rámci libde265 jsem upravil soubor deblock.cc (který zajišťuje funkci deblokačního filtru pro dekodér) tím, že jsem napsal funkci exportDblkData (v souboru deblock.cc na řádcích 379 až 617). Tato funkce fungující analogicky jako funkce derive_boundaryStrength obsažené v původní verzi programu. Funkce exportDblkData ukládá informace o kvantizačních parametrech, intra predikci, pohybové predikci a parametrech pro filtraci obou doplňkových barevných složek (chroma složky U a V) do binárního souboru.

Každý snímek se zpracovává dvakrát a to nejdříve pro filtraci vertikálních hran, a pak pro filtraci horizontálních hran. Pokaždé jsou tedy vytvořena dvourozměrná pole s rozměry danými rozlišením snímku a to pro kvantizační parametry (qp_mat), parametry použití intra predikce (intra_mat), obsah nenulových koeficientů sousedních bloků (nonzero_mat), informace o referenčních blocích pohybové predikce sousedních bloků (samepic_mat) a parametry vektorů pro pohybovou predikci sousedních bloků (sameref_mat). Po těchto maticích jsou do výstupního souboru připojeny také parametry cQpPicOffsetCb a cQpPicOffsetCr, které určují odsazení kvantizačních parametrů obou chroma složek od těch uložených pro jasovou složku obrazu. Kromě kvantizačních parametrů a odsazení kvantizačních

parametrů jsou všechny další informace ukládány jen v případě, že se na deblokační mřížce 8x8 pixelů nachází také rozhraní dvou bloků (CU, TU nebo PU).

7.2 Načtení dat do prostředí MATLAB

V hlavní fukci deblockingFilter.m jsou pro tento účel použity funkce loadFile.m a loadDeblkParam.m. První z nich načte snímek z video souboru ve formátu .yuv a to jen pro barevné vzorkovací schéma 4:2:2, tedy pokud mají obě doplňkové chroma složky dvakrát menší rozlišení než jasová složka obrazu. Výstupem této funkce jsou tři matice obsahující tři složky snímku vstupního videa. Druhá funkce loadDeblkParam.m načte data pro filtraci z binárního souboru vytvořeného funkcí dekodéru popsaném v první podkapitole. Děje se tak zlášť pro každý směr filtrace, tato funkce nám vrátí matice obsahující data ze všech uložených polí popsaných v první podsekci této kapitoly, tedy qp_mat, intra_mat, nonzero_mat, samepic_mat, sameref_mat a cQpPicOffsetCb s cQpPicOffsetCr.

7.3 Odvození parametru Bs

Tuto roli zaujímá funkce deriveBs.m, která z dat načtených funkcí loadDeblkParam.m vytvoří matici obsahující rozhodnutí o síle použitého filtru pro každou hranici dvou bloků v obraze. Její funkce probíhá přesně, jak je popsáno v podkapitole 6.1.

7.4 Filtrování jasové složky

Dalším krokem je filtrování jasové (luma) složky obrazu funkcí lumaFilter.m. Jejím vstupem jsou: jasová složka obrazu, parametry Bs a kvantizační parametry. Navíc je možné ovlivnit funkci filtru parametry tcOffset a betaOffset, jež mají vliv na výslednou sílu aplikovaného filtru. Samotný filtr funguje přesně, jak je specifikováno v referenčních materiálech pro standard H.265 [1] v sekci 8.7.2.5.

7.5 Filtrování doplňkových barevných složek

Obě barevné složky jsou filtrovány funkcí chromaFilter.m. Vstupní proměnné jsou: barevná složka obrazu, parametry Bs (spočtené pro jasovou složku), kvantizační parametry jasové složky a dvě proměnné tcOffset a qpOffset. Proměnná tcOffset má stejnou funkci jako ve filtru jasové složky a ovlivňuje rozhodovací proces filtru a jeho výslednou sílu. Parametr qpOffset je stejný jako jeden

z cQpPicOffsetCb a cQpPicOffsetCr (podle toho jaká složka obrazu je zpracovávána) a obsahuje hodnotu odsazení kvantizačních parametrů barevné složky obrazu od té jasové. Jinak filtr funguje stejně, jak je specifikováno v referenčích materiálech pro standard H.265 [1] v sekci 8.7.2.5.

7.6 Výsledky filtrace

Ověření funkce navrženého filtru probíhalo na vzorovém videu o rozlišení PAL a cílovým bitovým tokem 14kbit/s. V konfiguraci jsem vypnul funkci SAO filtru, abych předešel případným nepřesnostem, které by z jeho použití mohly plynout. Je totiž aplikován až po filtru deblokačním a na zpracovaném videu by se to mohlo projevit. Ukázka originálního videa je níže na obrázku 7.1 získaného ze zdroje [8]. Dále je snímek ze stejného videa po dekódování bez použití deblokačního filtru, obrázek 7.2.

Obrázek 7.1. První snímek originální, nezpracované videosekvence

Obrázek 7.2. První snímek videa bez použití deblokačního filtru

Nakonec jsou tu první snímky videosekvence, nejdříve filtrovaného standardním dekodérem H.265 (obrázek 7.3) a druhý je deblokován mnou navrženým deblokačním filtrem v prostředí MATLAB (obrázek 7.4).

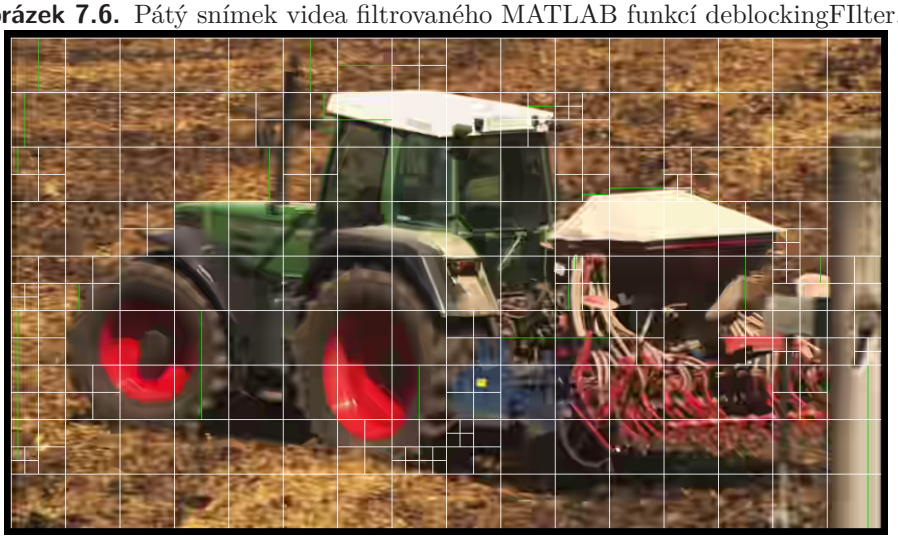
Obrázek 7.3. První snímek videa filtrovaného standardním dekodérem H.265

Obrázek 7.4. První snímek videa filtrovaného MATLAB funkcí deblockingFIlter.m

Na prvním snímku není vidět rozdíl mezi oficiálním a mnou navržený filtrem, v tomto případě filtr pracuje spravně. Bohužel při přechodu kodéru na pohybovou predikci, jako je tomu například u pátého snímku ukázaného na obrázcích níže 7.5 a 7.6, je již rozdíl značný. Kontrolou a podrobnou analýzou tohoto problému, jak ve funkcích v prostředí MATLAB tak na straně dekodéru, jsem zjistil, že problém vzniká při exportu dat pro deblokační filtr funkcí exportDblkData v souboru deblock.cc a když tato funkce detekuje hrany a ukládá o nich informace jen, jak je znázorněno na obrázku z nástroje sherlock265 [11] (součást libde265), který umožňuje zvýraznit hrany CTB, CU, PU a TU ve snímku videosekvence 7.7. Tato informace vyvolává otázku, proč se tedy v standardním dekodéru videa filtrují i hrany mimo ty, které identifikuje nástroj sherlock265. Nicméně se mi již nepodařilo vyřešit tuto chybu a v tomto ohledu by byla nutná další analýza souboru deblock.cc a celého dekódovacího procesu v rámci libde265.

Obrázek 7.5. Pátý snímek videa filtrovaného standardním dekodérem H.265

Obrázek 7.6. Pátý snímek videa filtrovaného MATLAB funkcí deblockingFIlter.m



Obrázek 7.7. Pátý snímek videa s vyznačenými hranicemi CTB, CU, PU a TU pomocí nástroje sherlock265

Kapitola 8
Závěr

Úkolem bylo posoudit vlastnosti nového standardu komprese videa H.265 na třech zvolených implementacích z hlediska objektivního měření kvality obrazu. Dalším cílem byla podrobná analýza deblokačního filtru obsaženém v tomto standardu a vlastní návrh takového filtru v prostředí MATLAB. Ukázalo se, že komprese referenčním kodekem HM12.1 je časově velmi náročná a s velkou pravděpodobností proto v této podobě nebude atraktivní pro širší využití. Naopak verze x265 a DivX tento nedostatek potlačily a jsou pro praktické užití vhodnější. Zajímavým zjištěním bylo, že časová náročnost komprese pro různá rozlišení nestoupá stejným tempem jako počet obrazových bodů.

Z měření objektivní kvality videa pomocí tří vybraných metrik vyplynuly očekávané výsledky. Nejdůležitějším zjištěním bylo, že jednotlivé implementace se kvalitativně přibližují se zvyšujícími se bitovými toky komprimovaných videosekvencí. Naopak při nízkých hodnotách bitových toků jsou výsledky měření neurčité a nemají valnou vypovídací hodnotu, protože všechny tři implementace mají příliš velkou odchylku při snaze o dosažení požadovaného bitového toku komprimovaného videa.

V dalším kroku se podařilo zjistit, že deblokační filtr má jen minimální vliv na objektivní kvalitu videa, stejně jako na detekci hran ve výsledném obraze. Návrh filtru v prostředí MATLAB se zdařil a za specifické situace (v prvním snímku sekvence, který byl intra predikován) vykazoval uspokojivé výsledky, srovnatelné s původním filtrem obsaženým v dekodéru videa.

Výsledek práce přibližuje pouze úzkou část rychle se vyvíjející problematiky komprese a obecně zpracování videa, ale její závěry mohou přispět k pochopení nového standardu komprese H.265, kterou pravděpodobně čeká velké rozšíření napříč uživatelskou i komerční sférou.

Kapitola 9
Reference

[1] JCT-VC. High Efficiency Video Coding (HEVC) text specification draft 10 (for FDIS Last Call), 14–23 Leden. 2013. http://phenix.it-sudparis.eu/jct/doc_end_user/current_document.php?id=7243, staženo z URL 18. března 2013.

[2] Andrey Norkin, Gisle Bjøntegaard, Arild Fuldseth, Matthias Narroschke, Masaru Ikeda, Kenneth Andersson, Minhua Zhou, and Geert Van der Auwera. HEVC Deblocking Filter, 12 Prosinec. 2012. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6324414, staženo z URL 28. dubna 2014.

[3] Philippe Hanhart. VQMT: Video Quality Measurement Tool, ke stažení http://mmspg.epfl.ch/vqmt, staženo z URL 31. března 2014

[4] Mihir Mody. Understanding in-loop filtering in the HEVC video standard, 21. června 2013. http://www.edn.com/design/consumer/4417218/Understanding-in-loop-filtering-in-the-HEVC-video-standard, staženo z URL 26. listopadu 2014.

[5] Lukáš Krasula. Objektivní metriky kvality obrazu, Praha 2011. Bakalářská práce České vysoké učení technické, Fakuklta elektrotechnická.

[6] Programová dokumentace k prostředí MATLAB dostupná na URL http://www.mathworks.com/help/matlab/

[7] Nekomprimované videosekvence pro testování kvality komprese dostupné ne URL ftp://ftp.ldv.ei.tum.de/videolab/public/SVT_Test_Set/, staženo 30. ledna 2013.

[8] Nekomprimované videosekvence pro testování kvality komprese dostupné na URL ftp://ftp.ldv.e-technik.tu-muenchen.de/pub/test_sequences/1080p/, staženo 23. října 2014

[9] Referenční implementace standardu H.265 dostupná na URL http://hevc.kw.bbc.co.uk/svn/jctvc-a124/branches/HM-12.1-dev/, staženo 3. ledna 2014.

[10] Open source implementace H.265 od organizace VideoLAN ve verzi z 29. listopadu 2013 dostupná na URL https://github.com/videolan/x265, staženo 24. února 2014.

[11] Open source implementace H.265 dostupná na URL https://github.com/strukturag/libde265, staženo 3. září 2014.

[12] HEVC implementace od společnosti DivX dostupná na URL http://www.divx.com/en/software/hevc-plugin, staženo 10. ledna 2014.

[13] How HEVC/H.265 works, technical details diagrams dostupné na URL http://forum.doom9.org/showthread.php?t=167081, staženo dne 5. ledna 2015.

[14] x265 Evaluator's Guide dostupné z 29. listopadu 2013 dostupná na URL https://github.com/videolan/x265, staženo 24. února 2014.

[15] JCT-VC. HM Software manual, pro verzi kodeku HM12.1 dostupná na URL http://hevc.kw.bbc.co.uk/svn/jctvc-a124/branches/HM-12.1-dev/, staženo 3. ledna 2014.

[16] YUV player, software pro přehrávání souborů ve formátu .yuv dostupný na URL http://raw-yuvplayer.sourceforge.net/, staženo 13. února 2013.

Kapitola 10
Seznam Zratek

HEVC - High Efficiency Video Coding
PSNR - Peak Signal to Noise Ratio
SSIM - Structural Similarity Index
VIF - Visual Information Fidelity
VIFP - Pixel based Visual Information Fidelity
CTB - Coding Tree Block
CU - Coding Unit
TU - Transform Unit
PU - Prediction Unit
cQpPicOffsetCb - Quantization Cb Offset
cQpPicOffsetCr - Quantization Cr Offset
Bs - Boundary Strength
SAO - Sample Adaptive Offset
MSE – Mean Squared Error
SNR – Signal to Noise Ratio
UIQI – Universal Image Quality Index
CM – Corelation Measure
CCM – Cross-Corelation Measure
CDM – Czekanowski Distance Measure
PDM – Pratt Distance Measure
ESM - Edge Stability Measure
HVS – Human Visual System
AE – Absolute (HVS) Error
HVS MSE – Normalizovaný MSE podle HVS
SMD – Spectral Magnitude Distortion
SPD – Spectral Phase Distortion
SPMD – Spectral Phase-Magnitude Distortion
DCT – Diskrétní Cosinová Transformace
GOP - Group of Pictures

Kapitola 11
Seznam příloh

- složka debfiltr
  - deblockingFilter.m
  - deriveBs.m
  - chromaFilter.m
  - loadDeblkParam.m
  - loadFile.m
  - lumaFIlter.m
  - saveFile.m
  - README.txt - jednoduchý návod k použití funkce deblockingFilter.m
  - video.bin - ukázkový binární soubor s daty pro filtraci videa video.yuv
  - video.yuv - ukázkový vzorek videa v rozlišení PAL pro filtraci
- složka edgecomp
  - edgecomp.m
  - README.txt - jednoduchý návod k použití funkce edgecomp.m
- složka nastroje
  - VQMT - nástroj pro měření objektivní kvality obrazu [3]
  - yuvplayer.exe - software pro přehrávání .yuv souborů [16]
- složka vqmtfunkce
  - encode_config.m
  - encode_protocol.m
  - graphResults.m
  - hmConfig.m
  - movcut.m
  - processResults.m
  - videoScale.m
  - config.txt
  - README.txt

- deblock.cc - upravená verze obsahující funkci exportDblkData
- config.txt - používaný konfigurační soubor pro kodek HM12.1
- video-filtered.yuv - ukázka videa filtrovaného pomocí funkce deblockingFilter.m (rozlišení PAL)
- video-nofilter.yuv - ukázka videa bez použití deblokačního filtru (rozlišení PAL)
- video-reference.yuv - ukázka standardně filtrovaného videa (rozlišení PAL)
- libde265.zip - obsahuje upravený soubor deblock.cc a návod k použití README