Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky



Objektivní porovnání kvality video kódovacího formátu AV1

BAKALÁŘSKÁ PRÁCE

Studijní program: Aplikovaná informatika

Studijní obor: Aplikovaná informatika

Autor: Petr Hovorka

Vedoucí bakalářské práce: Bc. et Bc. Jaroslav Svoboda, MSc et MSc

Praha, květen 2020

Prohlášení

Prohlašuji, že jsem bakalářskou práci Objektivní porovnání kvality video kódovacího formátu AV1 vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne 10. května 2020 .....

Petr Hovorka

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé bakalářské práce Bc. et Bc. Jaroslavu Svobodovi, MSc et MSc za vstřícný přístup, odborné rady a cenné připomínky, které mi pomohly tuto práci dokončit.

Abstrakt

Bakalářská práce mapuje royalty-free kodeky a jejich historii. Detailně se zabývá video formátem AV1, za kterým stojí Alliance for Open Media. V teoretické části práce popisuje historický vývoj royalty-free video formátů a stručně seznamuje se samotným formátem AV1 včetně jeho implementací. V praktické části pomocí metod pro objektivní porovnání kvality videa (MS-SSIM, PSNR-HVS-M, VMAF) hodnotí efektivitu komprese různých kodérů AV1 a porovnává je s kodéry formátů H.264, H.265 a VP9. Dalším významným aspektem, který práce zahrnuje, je zhodnocení časové a paměťové náročnosti různých kodérů formátu AV1.

Bylo zjištěno, že formát AV1 dosahuje ve většině případů lepších výsledků objektivního hodnocení kvality než starší formáty. Kvalitou nejblíže referenční implementaci libaom formátu AV1 je kodér x265 formátu H.265. Dalším poznatkem je, že mezi jednotlivými kodéry AV1 jsou značné rozdíly jak ve výsledné kvalitě videa, tak v rychlosti kódování. Ta je závislá nejen na použitém kodéru, ale také na jeho nastavení a přenosové rychlosti výsledného videa. Porovnání využití paměti zobrazuje optimalizaci kodéru kodeku libaom, jehož novější verze k chodu potřebuje přibližně polovinu paměti než jeho o rok starší verze. Nejvíce paměti potřebuje kodér SVT-AV1.

Klíčová slova

AV1, AOMedia, video, kódování, VMAF.

JEL klasifikace

C89, L82, L86, L96.

Abstract

This bachelor's thesis maps royalty-free video codecs and their history. It deals in detail with AV1 codec by Alliance for Open Media. The theoretical part of the thesis describes the historical development of royalty-free video formats and briefly describes the AV1 itself, including its implementations. Practical part evaluates compression efficiency of AV1 encoders using various objective video quality assessment methods (MS-SSIM, PSNR-HVS-M, VMAF) and compares them with encoders of H.264, H.265 and VP9 video formats. Moreover, it evaluates computational complexity and memory usage of various AV1 encoders.

It was found that the AV1 format achieves better results of objective quality assessment than older formats in most cases. The closest objective quality to the reference implementation of the AV1 format (libaom) has the x265 encoder of the H.265 format. Another finding is that there are significant differences between AV1 encoders in both the resulting video quality and the encoding speed. Speed is dependent not only on the used encoder but also on encoder settings and the bit rate of the resulting video. The memory usage comparison shows significant optimization of the libaom encoder, whose newer version requires approximately half the memory to run than its one-year older version. The SVT-AV1 encoder needs the most memory.

Keywords

AV1, AOMedia, video, coding, VMAF.

JEL Classification

C89, L82, L86, L96.

Obsah

Úvod 11
1 Historie royalty-free video formátů 12
1.1 Theora 12
1.2 Dirac 12
1.3 VP8 13
1.4 VP9 13
1.5 Daala 14
1.6 Thor 14
1.7 VP10 14
1.8 Porovnání royalty-free formátů 15
2 Alliance for Open Media 16
2.1 Historie 16
2.1.1 NETVC a vznik AOMedia 16
2.2 Členové 17
3 AV1 18
3.1 Profily a úrovně 18
3.1.1 Profily 18
3.1.2 Úrovně 19
3.2 Technologie 19
3.2.1 Rozdělení na bloky pro predikci 19
3.2.2 Predikce 20
3.2.3 Transformace 22
3.2.4 Kvantizace 23
3.2.5 Filtry 23
3.2.6 Kódování entropie 23
3.3 Kontejnery 24
3.4 Kodeky 24
3.4.1 Libaom 24
3.4.2 Rav1e a dav1d 24
3.4.3 SVT-AV1 25
3.4.4 Libgav1 25
3.5 Softwarová podpora 25
3.5.1 Operační systémy 25

3.5.2 Aplikace 26
3.6 Hardwarová podpora 26
3.7 Praktické využití formátu 27
4 Metody objektivního hodnocení kvality videa 29
4.1 MS-SSIM 29
4.2 PSNR-HVS-M 30
4.3 VMAF 32
5 Objektivní porovnání kvality a rychlosti 35
5.1 Příkazy použité pro kódování 37
5.2 VQMT a VMAF příkazy použité pro testování 39
5.3 Objektivní porovnání kvality s ostatními formáty 39
5.3.1 VMAF 40
5.3.2 MS-SSIM 41
5.3.3 PSNR-HVS-M 43
5.4 Objektivní porovnání kvality AV1 kodérů 44
5.4.1 VMAF 45
5.4.2 MS-SSIM 47
5.4.3 PSNR-HVS-M 49
5.5 Objektivní porovnání rychlosti 51
5.6 Porovnání využití paměti jednotlivými AV1 kodéry 53
Závěr 55
Použitá literatura 56
Přílohy I
Příloha A: Objektivní porovnání kvality I
Příloha B: Objektivní porovnání rychlosti X
Příloha C: Porovnání využití paměti kodéry formátu AV1 XIII

Seznam obrázků

Obrázek 1 Podporované vzory rozdělení superbloků ve formátu AV1 (35) 20
Obrázek 2 Paeth prediktor (vlevo), Smooth prediktor (vpravo). Velikost bloků 4×4 (39) . 22
Obrázek 3 Nastavení AV1 na platformě YouTube 28
Obrázek 4 Schéma metody MS-SSIM (66) 30
Obrázek 5 Schéma metody PSNR-HVS-M (69) 32
Obrázek 6 Schéma VIS (70) 33
Obrázek 7 Schéma metody VMAF (72) 34
Obrázek 8 Sekvence Park Joy, Crowd Run a Sintel 36
Obrázek 9 Výsledek testu VMAF sekvence Park Joy – porovnání s ostatními formáty 40
Obrázek 10 Výsledek testu VMAF sekvence Crowd Run – porovnání s ostatními formáty 40
Obrázek 11 Výsledek testu VMAF sekvence Sintel – porovnání s ostatními formáty 41
Obrázek 12 Výsledek testu MS-SSIM sekvence Park Joy – porovnání s ostatními formáty 41
Obrázek 13 Výsledek testu MS-SSIM sekvence Crowd Run – porovnání s ostatními formáty 42
Obrázek 14 Výsledek testu MS-SSIM sekvence Sintel – porovnání s ostatními formáty ... 42
Obrázek 15 Výsledek testu PSNR-HVS-M sekvence Park Joy – porovnání s ostatními formáty 43
Obrázek 16 Výsledek testu PSNR-HVS-M sekvence Crowd Run – porovnání s ostatními formáty 43
Obrázek 17 Výsledek testu PSNR-HVS-M sekvence Sintel – porovnání s ostatními formáty 44
Obrázek 18 Výsledek testu VMAF sekvence Park Joy – porovnání kodérů AV1..... 45
Obrázek 19 Výsledek testu VMAF sekvence Crowd Run – porovnání kodérů AV1 45
Obrázek 20 Výsledek testu VMAF sekvence Sintel – porovnání kodérů AV1..... 46
Obrázek 21 Výsledek testu MS-SSIM sekvence Park Joy – porovnání kodérů AV1 47
Obrázek 22 Výsledek testu MS-SSIM sekvence Crowd Run – porovnání kodérů AV1..... 47
Obrázek 23 Výsledek testu MS-SSIM sekvence Sintel – porovnání kodérů AV1..... 48
Obrázek 24 Výsledek testu PSNR-HVS-M sekvence Park Joy – porovnání kodérů AV1 ... 49
Obrázek 25 Výsledek testu PSNR-HVS-M sekvence Crowd Run – porovnání kodérů AV1 49
Obrázek 26 Výsledek testu PSNR-HVS-M sekvence Sintel – porovnání kodérů AV1 50
Obrázek 27 Čas potřebný pro kódování sekvence Park Joy 51
Obrázek 28 Čas potřebný pro kódování sekvence Crowd Run 52
Obrázek 29 Čas potřebný pro kódování sekvence Sintel..... 52
Obrázek 30 Maximální využití paměti – porovnání kodérů AV1 54

Seznam tabulek

Tabulka 1 Royalty-free formáty 15
Tabulka 2 Profily formátu AV1 (34) 18
Tabulka 3 Úrovně formátu AV1 (34) 19
Tabulka 4 Porovnání intra-frame predikce formátů AV1 a VP9 (35) 21
Tabulka 5 Použitý HW a SW 35
Tabulka 6 Parametry testovacích sekvencí 36
Tabulka 7 Naměřené hodnoty pomocí metody VMAF sekvence Park Joy I
Tabulka 8 Naměřené hodnoty pomocí metody VMAF sekvence Crowd Run II
Tabulka 9 Naměřené hodnoty pomocí metody VMAF sekvence Sintel III
Tabulka 10 Naměřené hodnoty pomocí metody MS-SSIM sekvence Park Joy IV
Tabulka 11 Naměřené hodnoty pomocí metody MS-SSIM sekvence Crowd Run V
Tabulka 12 Naměřené hodnoty pomocí metody MS-SSIM sekvence Sintel VI
Tabulka 13 Naměřené hodnoty pomocí metody PSNR-HVS-M sekvence Park Joy (dB) .. VII
Tabulka 14 Naměřené hodnoty pomocí metody PSNR-HVS-M sekvence Crowd Run (dB) VIII
Tabulka 15 Naměřené hodnoty pomocí metody PSNR-HVS-M sekvence Sintel (dB) IX
Tabulka 16 Čas v sekundách potřebný pro kódování sekvence Park Joy X
Tabulka 17 Čas v sekundách potřebný pro kódování sekvence Crowd Run XI
Tabulka 18 Čas v sekundách potřebný pro kódování sekvence Sintel XII
Tabulka 19 Maximální využití paměti – sekvence Park Joy (MB) XIII

Seznam zkratek

AC	Alternating Current koeficient
ADST	Asymetrická Diskrétní Sinová Transformace
AOM, AOMedia	Alliance for Open Media
AV1	AOMedia Video 1
AVIF	AV1 Image File Format
DC	Direct Current koeficient
DCT	Diskrétní Kosinová Transformace
DLM	Detail Loss Metric
DWT	Diskrétní Vlnková Transformace
HDR	High Dynamic Range
HLG	Hybrid Log Gamma
HVS	Human Visual System model
IETF	Internet Engineering Task Force
JVET	Joint Video Experts Team
PSNR-HVS-M	Peak signal-to-noise ratio – human visual system modification
MS-SSIM	Multiscale Structural similarity
MSE	Mean Squared Error – střední kvadratická chyba
NETVC	Internet Video Codec
VIF	Visual Information Fidelity
VMAF	Video Multimethod Assessment Fusion
VQMT	Video Quality Measurement Tool
VVC	Versatile Video Coding
W3C	World Wide Web Consortium

Úvod

Když se podíváme na složení celosvětového internetového provozu a jeho vývoj v čase, zjistíme, že podíl dat, které tvoří streamované video neustále roste. Dle odhadů společnosti Cisco by v roce 2022 mělo video představovat 82 % veškerého datového přenosu na internetu. (1)

Hledání co možná nejefektivnější komprese video obsahu je podporováno nejen rostoucí popularitou videí na internetu, ale i zvyšujícími se nároky diváků na vizuální zážitek ze sledování. Výsledkem snahy o uspokojení těchto nároků jsou videa s vyšším rozlišením, vyšším dynamickým rozsahem, 360stupňová videa pro virtuální realitu, 3D videa a podobně. Tato videa typicky vyžadují vyšší přenosové rychlosti surových (nekomprimovaných) dat, což dále podněcuje vývoj nových kompresních algoritmů a formátů kódování. Bez odpovídajícího zvýšení efektivity komprese by pro společnosti poskytující video obsah často nebylo ekonomicky únosné tato videa streamovat či vysílat. (2)

V současné době se vývoji nových video kódovacích formátů věnují dvě na sobě nezávislá uskupení. Tím prvním je Alliance for Open Media (AOMedia) vyvíjející formát AV1, jehož první verze vyšla v březnu 2018 a druhým je Joint Video Experts Team (JVET) vyvíjející formát VVC. Druhý zmiňovaný bude nástupcem formátu HEVC a jeho finální standardizace by měla být dokončena v druhé polovině roku 2020. (3) Tento formát bude na rozdíl od AV1 zatížen licenčními poplatky. (4) V tom spočívá hlavní rozdíl mezi oběma formáty. AV1 je tzv. royalty-free formát s otevřeným zdrojovým kódem, který může kdokoli použít bez jakéhokoli poplatku. (5)

Tato práce se primárně zaměřuje na již zmíněný video kódovací formát AV1 a některé jeho royalty-free předchůdce. Cílem práce je objektivně zhodnotit vizuální kvalitu formátu AV1 a porovnat ji s dalšími video formáty – VP9, H.265 a H.264. K samotnému porovnání je využito několika metod objektivního hodnocení kvality videosekvencí. Konkrétně se jedná o metody PSNR-HVS-M, MS-SSIM a VMAF. V závěru je vyhodnocena časová a paměťová náročnost kódování jednotlivých formátů videa pomocí vybraných kodérů.

K vypracování práce bylo využito výhradně elektronických zdrojů. Z důvodu vysoké aktuálnosti zpracovávaného tématu nebyly v době psaní dostupné žádné vhodné knižní publikace.

1 Historie royalty-free video formátů

Jako royalty-free lze označit takový formát kódování videa, jehož užití není podmíněno platbou licenčních poplatků. Je nutné rozlišovat mezi pojmy „formát kódování videa“ a „video kodek“. První pojem označuje samotný obsah souboru s videem – jakým způsobem jsou uložena data – tzv. specifikace, druhý pojem se užívá pro konkrétní softwarovou nebo hardwarovou implementaci, která je schopná komprese nebo dekomprese dat do nebo z již zmíněného formátu. V praxi se lze často setkat se situací, kdy je formát zatížený patenty a jeho užití je podmíněno platbou licenčních poplatků – například formát H.265, ale samotná implementace je open source (v případě již zmíněného formátu H.265 jsou to například kodéry x265 a SVT-HEVC).

Tato kapitola popisuje vybrané royalty-free video formáty seřazené chronologicky dle jejich vzniku.

1.1 Theora

Organizace Xiph.Org Foundation započala vývoj formátu poté, co jí byl v červnu 2002 darován formát VP3.2 od společnosti On2. On2 poskytlo neodvolatelnou bezplatnou licenci na všechny patenty vztahující se k VP3, která dovoluje komukoliv pro jakékoliv použití využít kodeky odvozené od VP3. 25. září 2002 byla vydána první alpha verze referenční implementace libtheora. Ta je šířena pod BSD licencí a její aktuální verze je 1.1.1 vydaná v říjnu 2009. Ke zmražení a zveřejnění specifikace formátu došlo v roce 2004.

Jelikož je Theora nadstavbou nad VP3, lze bitstream ve formátu VP3 překonvertovat drobnými syntaktickými úpravami do bitstreamu formátu Theora, aniž by bylo nutné vykonat celou kompresi znovu (ale ne naopak). (6)

Pro samotnou kompresi využívá standardní diskrétní kosinovou transformaci druhého typu s bloky o velikosti 8×8. Podporuje snímky typu I a P, B snímky nejsou podporovány. Barevný prostor je YC_bC_r s podporovaným podvzorkováním barvonosných složek 4:2:0, 4:2:2 nebo 4:4:4 při 8 bitech na barevný kanál. Bitrate je vždy variabilní. (7)

1.2 Dirac

Dirac je kompresní formát vyvinutý britskou společností BBC Research & Development. Jeho specifikace byla dokončena v lednu 2008 a další vývoj se týkal pouze opravy chyb.

Formát Dirac má dvě open source a royalty-free softwarové implementace. První z nich je dirac-research a druhá je Schrödinger. Kodek dirac-research je napsaný v jazyce C++ a jeho první verze byla vydána 17. září 2008. Druhá implementace, Schrödinger, je napsaná v ja-

zyce C a je lépe optimalizovaná, než dirac-research. Schrödinger ve verzi 1.0.9 překonal dirac-research ve většině případů jak v rychlosti kódování, tak ve vizuální kvalitě kódovaného videa.

Formát a jeho implementace byly pojmenovány podle teoretických fyziků – Paula Diraca a Erwina Schrödingera, kteří společně sdíleli v roce 1933 Nobelovu cenu za fyziku.

Dirac podporuje vysoká rozlišení, a kromě ztrátové podporuje i bezztrátovou kompresi. Využívá tzv. vlnkové transformace (stejně jako například JPEG 2000, či formát Cineform), na rozdíl od diskrétní kosinové transformace, které využívají všechny ostatní formáty zmiňované v této kapitole. Formát může být použit v kontejnerech AVI, Ogg, Matroska, ISO/IEC MPEG-4 Part 12, či být přenášen v MPEG-TS. Bitrate může být konstantní, nebo variabilní. Podporované formáty podvzorkování barvonosných složek jsou 4:4:4, 4:2:2 a 4:2:0, bitová hloubka může být 8, 10, 12 nebo 16 bitů na barevný kanál. Formát podporuje snímky typu I, P i B, volitelně lze využít pouze intra-frame kódování za pomoci snímků typu I. Tento typ kódování se jmenuje Dirac Pro a má využití především v profesionálním prostředí, BBC ho interně využívala například pro přenos HDTV materiálu při pekingských olympijských hrách v roce 2008. (8–10)

1.3 VP8

Formát vyvinutý společností On2 Technologies v roce 2008 jako nástupce formátu VP7. V současné době je vlastněný společností Google, která v roce 2010 společnost On2 akvírovala. Po této akvizici byla specifikace formátu uvolněna pod licencí Creative Commons Attribution 3.0. Následně byl uvolněn kodek libvpx pod licencí BSD.

Formát podporuje podvzorkování barvonosných složek pouze ve formátu 4:2:0 při 8 bitech na kanál. Pro kompresi využívá diskrétní kosinovou transformaci. V jednom konkrétním případě však VP8 využívá i Walsh-Hadamardovu transformaci. (11)

Formát VP8 byl v minulosti předmětem sporu o porušení patentů mezi společností Google a společností MPEG LA. Ta zastupovala 12 držitelů patentů. V březnu 2013 došlo k dohodě obou společností, jejíž detaily nejsou veřejně známé. (12)

1.4 VP9

VP9 je nástupcem formátu VP8. Na rozdíl od předchůdce je již od počátku vyvíjen Googlem. Vývoj začal v druhé polovině roku 2011, první vydání spatřilo světlo světa 17. června 2013. Referenční implementací je kodek libvpx. Kodér formátu VP9 vyvíjí i Intel. Jmenuje se SVT-VP9 a měl by být rychlejší než kodér libvpx. Zatím je v ranné verzi 0.1.0, která byla vydána v říjnu 2019. (13, 14)

Formát podporuje mnoho barevných prostorů. Díky tomu je zajištěna podpora videí s vysokým dynamickým rozsahem (HDR). Standardy HDR pro formát VP9 jsou Hybrid Log-

Gamma (HLG) a SMPTE ST 2084 (označován také jako Perceptual Quantizer), oba využívají barevný prostor Rec. 2020. Kromě ztrátové VP9 podporuje i bezztrátovou kompresi.

Pro kompresi využívá kombinaci diskrétní kosinové transformace a asymetrické diskrétní sinové transformace (ADST) bloků o velikosti 4×4 až 32×32. (14, 15)

V současnosti je VP9 využívaným formátem na mnoha video streamovacích službách, konkrétně například na platformách YouTube a Netflix. (16, 17)

1.5 Daala

Formát a kodek vyvíjený neziskovou organizací Xiph.Org Foundation ve spolupráci s hlavním sponzorem – Mozillou. Jedná se o nástupce formátu Theora, který vyšel již v roce 2004. První experimentální verze kodeku Daala vznikla v roce 2010. Svůj název získal podle postavy admirála Natasy Daaly z filmů Star Wars. Od vstupu Mozilly do AOMedia došlo k výraznému zpomalení jeho vývoje. (18, 19)

1.6 Thor

Kodek, jehož vývoj má na starosti společnost Cisco Systems. Vznikl v reakci na cenu licence za formát H.265, která byla až 16krát dražší než v případě H.264. Z toho důvodu nebyl formát H.265 vhodný pro použití v open source software, jako jsou například webové prohlížeče. (20)

1.7 VP10

VP10 byl plánovaný video kódovací formát od společnosti Google. Mělo se jednat o přímého nástupce formátu VP9. Jeho vývoj začal v září 2014. Po vstupu Google do konsorcia AOMedia bylo rozhodnuto o začlenění VP10 do projektu AV1. Vývoj formátu VP10 tím byl ukončen a formát VP9 tak má být posledním VPx formátem. (14)

1.8 Porovnání royalty-free formátů

Tabulka 1 Royalty-free formáty

Formát	Vývojář	První vydání	Algoritmus	Referenční kodek	Licence referenčního kodeku
Theora	Xiph.Org	2004	DCT	libtheora	3-bodová BSD
Dirac	BBC R&D	2008	DWT	dirac-research / Schrödinger	MPL 1.1, GNU GPL 2, GNU LGPL 2.1 / MPL 1.1, GNU GPL 2, GNU LGPL 2, MIT License
VP8	On2/Google	2008	DCT	libvpx	3-bodová BSD
VP9	Google	2013	DCT/ADST	libvpx	3-bodová BSD
Daala	Xiph.Org	–	Lapped DCT (21)	Daala	2-bodová BSD
Thor	Cisco Systems	–	DCT	Thor	2-bodová BSD
AV1	AOMedia	2018	DCT/ADST/FilpADST/IDTX	libaom	2-bodová BSD

2 Alliance for Open Media

Alliance for Open Media (AOMedia) je neziskové konsorcium, které se zaměřuje na vývoj otevřených royalty-free technologií v oblasti multimédií – zejména otevřených alternativ ke standardům od pracovní skupiny Moving Picture Experts Group (MPEG) (22).

„Členové AOMedia se zavázali vyvinout nejlepší video technologii a učinit jí bezplatnou, aby si ji mohli všichni užít.“ (23)

2.1 Historie

Již mnoho let před založením konsorcia existovalo několik otevřených video kódovacích formátů, které si kladly za cíl konkurovat uzavřeným a patenty chráněným formátům od skupiny MPEG. Za těmito otevřenými formáty stály například společnosti a organizace Google, Cisco či Xiph.Org Foundation. (24)

2.1.1 NETVC a vznik AOMedia

Ještě před vznikem AOMedia byly snahy o vytvoření jednotného standardu pro royalty-free video kódovací formát. Název pracovní skupiny, která měla jeho vznik na starosti, je Internet Video Codec (NETVC) a stojí za ní Internet Engineering Task Force (IETF). Cílem této organizace je vyvíjet a podporovat internetové standardy. Úzce spolupracuje s konsorciem W3C. (25)

V březnu 2015 předložila organizace Xiph.Org formát Daala jako kandidáta na NETVC. (26) V červenci 2015 byl jako další kandidát předložen formát Thor společnosti Cisco Systems. (26)

Bylo s podivem, že se jako další kandidát nepřihlásila společnost Google, která v té době investovala nemalé prostředky do vytvoření otevřeného video formátu VP9. (27)

1. září 2015 sedm velkých internetových společností oznámilo vznik Alliance for Open Media. Zakládajícími členy byly společnosti Amazon, Cisco, Google, Intel, Microsoft, Mozilla a Netflix. Za cíl si daly vytvořit video kódovací formát nové generace – AV1. Jako základ pro něj měly sloužit formáty Daala, Thor a VP10 (28) Výraznou motivací pro založení Aliance byl vznik druhého patentového fondu pro HEVC (tzv. HEVC Advance) na jaře 2015, který zvyšoval nejistotu ohledně licenčních poplatků pro další generaci video formátu od skupiny MPEG. (27)

Na schůzce skupiny NETVC v březnu 2018 se jako třetí kandidát přihlásila švédská společnost Divideon s kodekem xvc. (29) Jeden z vývojářů Thoru Steinar Midtskogen potvrdil, že část formátu xvc, která je považována jako royalty-free, má lepší efektivitu komprese než Thor při zachování stejné výpočetní složitosti. (18)

Kodek xvc nezískal potřebnou důvěru pracovní skupiny v tom, že všechny jeho součásti jsou opravdu royalty-free a v budoucnu nedojde k nárokování licenčních poplatků za duševní vlastnictví od společností třetích stran. Z důvodu značného zpomalení vývoje zbývajících dvou kandidujících kodeků – Daala a Thor, ve prospěch vývoje AV1, bylo na schůzce v červenci 2018 rozhodnuto o pozastavení fyzických setkání pracovní skupiny. (18)

2.2 Členové

Vedoucími členy konsorcia s možností rozhodovat v představenstvu jsou v současnosti společnosti Amazon, Apple, ARM, Cisco, Facebook, Google, IBM, Intel, Microsoft, Mozilla, Netflix, Nvidia, Samsung a Tencent. Dále má Alliance i další členy bez nároku rozhodovat v představenstvu. Těmito členy jsou například Adobe, AMD, Alibaba, Amlogic, BBC Research & Development, Broadcom, Hulu, Realtek, VideoLAN, Vimeo a další. (30)

3 AV1

AV1, celým názvem AOMedia Video 1, je otevřený a bezpoplatkový formát kódování videa vyvíjený konsorciem Alliance for Open Media. K definitivnímu dokončení specifikace a zmražení bitstreamu došlo 28. března 2018.

Společně s formátem pro kódování videa vznikl i formát komprese obrázků AVIF (AV1 Image File Format). Specifikace byla dokončena v únoru 2019. Dle Netflixu již potřebujeme alternativu k formátu JPEG, která bude široce podporována, bude mít lepší efektivitu komprese a bude mít více podporovaných režimů a funkcí. Netflix věří, že AVIF má potenciál se touto alternativou stát. Dle jejich měření je efektivita komprese o mnoho vyšší, formát podporuje například HDR, až 12-bit barevnou hloubku, ztrátovou i bezztrátovou kompresi, průhlednost, široké barevné gamuty a podobně. Největším problémem je tedy již zmíněná podpora, která zatím není široká. (31, 32)

3.1 Profily a úrovně

Profily a úrovně jsou definovaná volitelná omezení pro kódované video. V praxi se lze setkat s hardwarovými i softwarovými dekodéry, které podporují jen určité profily a úrovně. Například v mobilních telefonech není běžně nutné dekódovat 12-bit video v 8k rozlišení při 120 snímcích za sekundu. Proto mají hardwarové dekodéry v těchto zařízeních přesně definovaná omezení a specifikují, které profily a úrovně podporují. (33)

3.1.1 Profily

AV1 definuje 3 základní profily pro dekodéry – **Main**, **High** a **Professional**. Liší se maximální bitovou hloubkou a podporovanými typy podvzorkování barvonosných složek, viz tabulka.

Tabulka 2 Profily formátu AV1 (34)

		Main (0)	**High (1)**	**Professional (2)**
Bitová hloubka		8-bit, 10-bit	8-bit, 10-bit	8-bit, 10-bit, 12-bit
Podvzorkování barvonosných složek (chroma subsampling)	4:0:0	Ano	Ano	Ano
	4:2:0	Ano	Ano	Ano
	4:2:2	Ne	Ne	Ano
	4:4:4	Ne	Ano	Ano

3.1.2 Úrovně

Úrovně jsou definovány v rozmezí 2.0 až 6.3 a jejich implementace záleží na schopnostech jednotlivých dekodérů.

Tabulka 3 Úrovně formátu AV1 (34)

Level	Maximální bitrate (Mb/s)		Příklad (rozlišení a snímková frekvence)
	Main profil	High profil	
2.0	1,5		426x240@30fps
2.1	3,0		640x360@30fps
3.0	6,0		854x480@30fps
3.1	10,0		1280x720@30fps
4.0	12,0	30,0	1920x1080@30fps
4.1	20,0	50,0	1920x1080@60fps
5.0	30,0	100,0	3840x2160@30fps
5.1	40,0	160,0	3840x2160@60fps
5.2	60,0	240,0	3840x2160@120fps
5.3	60,0	240,0	3840x2160@120fps
6.0	60,0	240,0	7680x4320@30fps
6.1	100,0	480,0	7680x4320@60fps
6.2	160,0	800,0	7680x4320@120fps
6.3	160,0	800,0	7680x4320@120fps

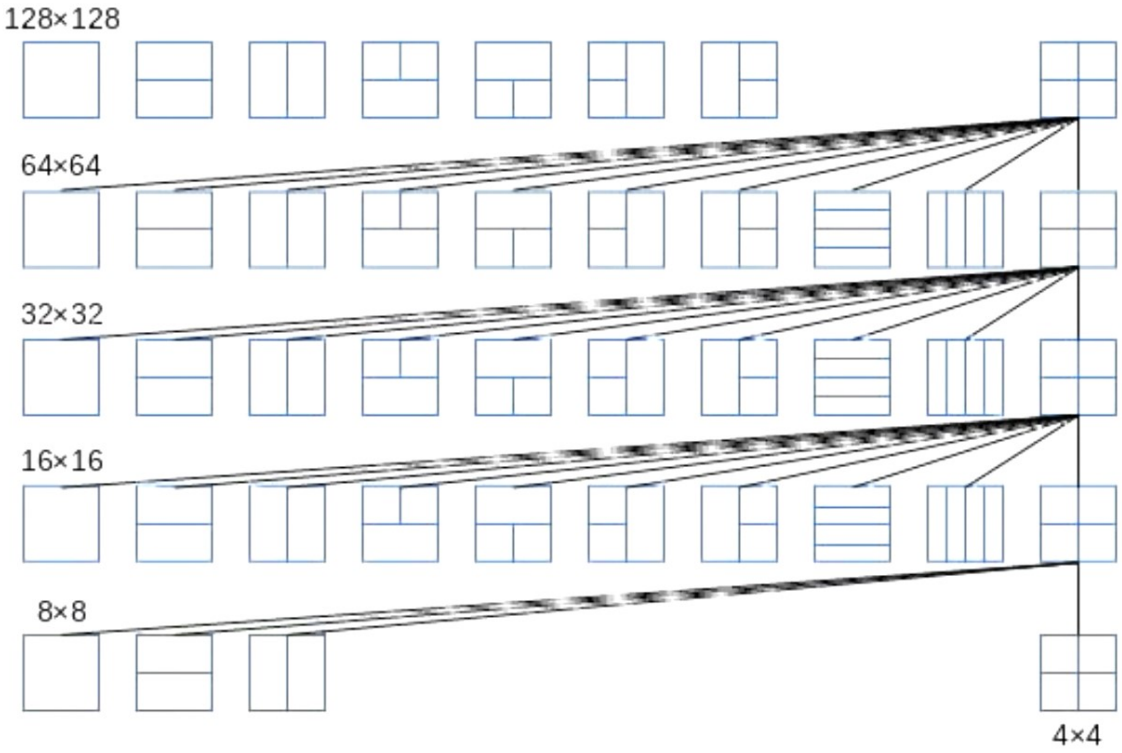
3.2 Technologie

Komprese formátu AV1 je komplexní proces, který se skládá z mnoha kroků a dílčích částí. Základ je stejně jako v případě ostatních video formátů postaven na frekvenční transformaci jednotlivých bloků obrazu. Tato kapitola je věnována zejména popisu změn a nových funkcí oproti nejbližšímu předchůdci – formátu VP9.

3.2.1 Rozdělení na bloky pro predikci

V prvním kroku se každý snímek rozdělí na bloky o stejné velikosti, nazvané superbloky. Mají čtvercový tvar a mohou být o velikosti 128×128 nebo 64×64 pixelů. Tyto superbloky mohou být dále rozděleny na menší bloky, v závislosti na různých dělících vzorech. Při rozdělení bloku na čtyři menší čtvercové bloky je lze dále rekurzivně dělit až na velikost 4×4 pixely, což je nejmenší možná velikost. Dalšími možnými vzory jsou rozdělení ve „tvaru T“ a vertikální či horizontální rozdělení na 4 či 2 pruhy. Dostupné dělící vzory jsou závislé na

velikosti bloku – bloky o velikosti 128×128 a 8×8 nepodporují vertikální a horizontální rozdělení na 4 pruhy, a bloky 8×8 mimo to nepodporují ani rozdělení ve „tvaru T“. Veškeré podporované vzory rozdělení znázorňuje Obrázek 1.



Obrázek 1 Podporované vzory rozdělení superbloků ve formátu AV1 (35)

Formát VP9 rozděluje snímek na bloky o největší velikosti 64×64 pixelů, které se dají dále dělit pomocí 3 dělících vzorů. Základním vzorem je rozdělení na 4 stejné čtverce, které se dají rekurzivně dále dělit až na velikost 4×4. Dalšími dvěma vzory je vertikální či horizontální rozdělení na dva pruhy. Názorný příklad podporovaných vzorů dělení VP9 lze spatřit v posledním řádku předchozího obrázku – lze říci, že formát VP9 podporuje pro všechny velikosti bloků pouze taková rozdělení, která podporují bloky o velikosti 8×8 formátu AV1. (35)

3.2.2 Predikce

Dalším krokem je predikce, u níž se rozlišují dva typy – intra-frame a inter-frame.

Intra-frame

Intra-frame predikce predikuje jednotlivé pixely určitého bloku na základě informací, které jsou dostupné ve stejném snímku. Nejčastěji se tyto predikce vytváří ze sousedních pixelů vlevo a nad predikovaným blokem. Podobné principy nejsou použity pouze pro kódování jednotlivých snímků videí, ale využívají je i formáty rastrových obrázků.

Tabulka 4 Porovnání intra-frame predikce formátů AV1 a VP9 (35)

AV1	VP9

• 56 směrových módů
• 6 nesměrových módů
  ○ DC, Smooth, Smooth H, Smooth V, Paeth, Palette
• 5 módů rekurzivního filtrování
  ○ DC, Vertical, Horizontal, 157°, Paeth
• Mód predikce barevné složky z jasové složky
	• 8 směrových módů
	• 2 nesměrové módy
	  ○ DC, TM

Směrová predikce odvozuje sousední pixely na základě určitého úhlu. AV1 má specifikováno celkem 56 módů (úhlů), základních 8 je v rozmezí 45–207° a pro každý je definováno 6 offsetů po 3° intervalech – tři nad hlavním úhlem a tři pod. Celkově tedy módy pokrývají rozmezí 36°– 216°, rozdíl těchto hodnot je 180°. (36)

Nesměrové módy predikce zahrnují prediktory DC, SMOOTH_V, SMOOTH_H, SMOOTH, Paeth a Palette.

- **DC** predikce pracuje na jednoduchém principu zprůměrování veškerých existujících pixelů konkrétního bloku.
- Prediktory **SMOOTH_V**, **SMOOTH_H** a **SMOOTH** predikují blok pomocí kvadratické interpolace ve svislém nebo vodorovném směru, případně jejich průměru.
- **Paeth** prediktor je jednoduchá lineární funkce, která funguje tak, že pro každý predikovaný pixel vezme hodnoty tří sousedních pixelů – vlevo, nahoře a nahoře vlevo a poté zvolí jako prediktor jeden z těchto tří pixelů – ten, který bude mít nejbližší hodnotu počátečnímu odhadu, viz pseudokód. Nahrazuje prediktor TM, který používá VP9.
- Mód **Palette** je přizpůsoben takovým užitím, jako je záznam obrazovky, prezentace, videohry a podobně, kde každý blok lze popsat malým množstvím unikátních barev. (36, 37)

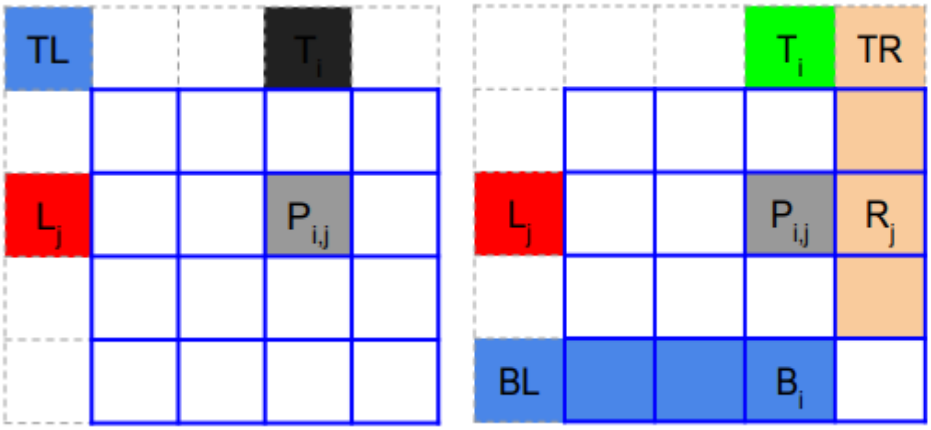
Pseudokód algoritmu prediktoru Paeth (38)

```
funkce PaethPrediktor (L, T, TL) {
     # L = vlevo, T = nahoře, TL = vlevo nahoře
     P = L + T - TL       # počáteční odhad
     PL = abs(P - L)      # vzdálenosti k L, T, TL
     PT = abs(P - T)
     PTL = abs(P - TL)
     # vrať nejbližší z L,T,TL,
     pokud PL <= PT AND PL <= PTL pak vrať L
     jinak pokud PT <= PTL pak vrať T
```

```
    jinak vrať TL
}
```

Smooth prediktor

- Předpokládejme, že R_j = TR, B_i = BL
- $P_{i,j}$ je vážený průměr L_j, R_j, T_i, B_i
- Váhy jsou ekvivalentní kvadratické interpolaci



Obrázek 2 Paeth prediktor (vlevo), Smooth prediktor (vpravo). Velikost bloků 4×4 (39)

Inter-frame

Tento typ predikce na rozdíl od přechozího typu predikuje podobu jednotlivých bloků na základě údajů z předchozích snímků.

Formát AV1 má 4 prediktory vektorů pohybu, to je o dva víc než v případě VP9. Dále AV1 rozšiřuje množství dostupných referenčních snímků pro každý snímek na 7 z původních 3 v případně VP9. (35)

3.2.3 Transformace

Formát AV1 podporuje 16 druhů transformace. Jde o 16 horizontálních/vertikálních kombinací DCT, ADST, FlipADST a IDTX. FlipADST aplikuje ADST v opačném pořadí, IDTX znamená přeskočení transformačního kódování v určitém směru, což je vhodné zejména pro kódování ostrých hran obrazu. VP9 podporuje 4 horizontální/vertikální kombinace DCT a ADST.

Transformační bloky AV1 mohou mít velikost nejvýše 64×64, VP9 maximálně 32×32. Transformační bloky mohou nebo nemusí být stejně velké, jako predikční. V případě VP9 musí být všechny transformační bloky čtvercové a pro každý predikční blok lze vždy zvolit pouze jeden rozměr transformačního bloku, rozměry nelze v rámci jednoho predikčního bloku kombinovat. AV1 přináší podporu obdélníkových transformačních bloků (poměr 1:2/2:1 a 1:4/4:1) pro obdélníkové predikční bloky, viz kapitola 3.2.1. Navíc lze kombinovat několik rozměrů transformačních bloků uvnitř jednoho predikčního bloku. (35, 36)

3.2.4 Kvantizace

AV1 má 6 kvantizačních parametrů – (Y‘, C_b, C_r) × (DC, AC). VP9 má takové parametry 4 – (jasová složka, barevná složka) × (DC, AC) – nerozlišuje odděleně C_b a C_r komponenty. DC (Direct Current) je takovým koeficientem, který je v grafu reprezentován přímkou, nemá žádnou frekvenci a popisuje obecnou intenzitu konkrétní složky konkrétního transformačního bloku. AC (Alternating Current) koeficienty jsou všechny ostatní, reprezentované jinou křivkou než přímkou. DC koeficient bloku bývá mnohonásobně vyšší než jednotlivé AC koeficienty. (40)

Oba formáty umožňují výběr 8 různých sad ze zmíněných kvantizačních parametrů pro každý snímek. Tyto sady jsou predikovány nejen z předchozího snímku (inter-frame), ale i ze snímku samotného (intra-frame), což je největší rozdíl oproti VP9, které pro tyto sady používá pouze inter-frame predikci. (35)

3.2.5 Filtry

Základním typem filtrů jsou tzv. **loop filtry**, které minimalizují různé kompresní artefakty. Formát AV1 obsahuje 3 filtry tohoto typu. Prvním loop filtrem je tzv. „**deblocking**“ filtr, který se stará o minimalizaci blokových artefaktů. Je velice podobný tomu, který je využit v případě formátu VP9. Druhým filtrem je kombinace „directional deringing“ filtru vyvinutého původně pro formát Daala a „constrained low-pass“ filtru z formátu Thor, které dohromady tvoří tzv. „**Constrained Directional Enhancement Filter**“ (CDEF). Ten odstraňuje artefakty a šum okolo ostrých hran, je směrový – dokáže tyto hrany následovat. Poslední loop filtr je nazvaný „**Loop Restoration**“ a slouží k odstranění šumu vzniklého při DCT pomocí rozmazání. Na rozdíl od CDEF není směrový, filtruje ve všech směrech. (41)

Zcela jiný důvod použití než předchozí zmíněné loop filtry má filtr „**film grain synthesis**“. Ten vychází z toho, že filmové zrno je velice složité zakódovat tak, aby byl výsledek kvalitní a nedocházelo k vytvoření závažných kompresních artefaktů. Tento filtr zmíněný problém obejde tak, že nejdříve zanalyzuje charakter zrna, zaznamená jeho parametry, a následně zrnitou složku ze vstupního videa zcela odstraní. Parametry zrna jsou při přehrávání předány dekodéru, který šum synteticky opět vytvoří. (36)

3.2.6 Kódování entropie

Pro samotné kódování entropie je zvolen nebinární aritmetický kodér, který byl původně součástí formátu Daala. VP9 využívá binární aritmetické kódování. Využití nebinárního aritmetického kódování přináší pro formát AV1 výhodu v potenciálně lepší paralelizaci jinak sériových procesů při binárním kódování. (35)

3.3 Kontejnery

Multimediální kontejner je soubor obsahující jeden nebo více proudů multimediálních dat (může obsahovat například jednu či více video stop, zvukové stopy, titulky atp.). Kromě samotných multimediálních dat jsou v souboru obsažena i dodatečná metadata popisující jednotlivé komponenty konkrétního kontejneru. (42)

Kontejnery, do kterých lze umístit videa ve formátu AV1 (32, 42):

- MPEG-4 Part 14 (.mp4)
- WebM (.webm)
- Matroska (.mkv)
- MPEG transport stream (.ts)
- IVF (.ivf)

3.4 Kodeky

Slovo kodek je složeninou prvních slabik slov kodér a dekodér a jedná se o zařízení, či software, který dokáže transformovat datový proud nebo signál. Pro účely této práce se o kodéru bavíme jako o softwaru či zařízení, které umožňuje zakódovat datový proud videa (stream) do určitého formátu vhodného například pro účely přenosu či uchování. Dekodér takto zakódovaná data dokáže zpět dekódovat do takové podoby, která je vhodná například k přehrávání. Rozdíl v kvalitě původního videa a v kvalitě zakódovaného videa ovlivňuje komprese. Různé techniky komprese videa, které specifikuje formát AV1, byly popsány v kapitole 3.2. Videoformát AV1 má v současné době 3 nejrozšířenější implementace – referenční libaom, dále dvojici kodéru rav1e a dekodéru dav1d, a v neposlední řadě kodek SVT-AV1. Porovnání objektivní kvality kodérů zmíněných kodeků je v kapitole 5.4 této práce.

3.4.1 Libaom

Libaom je referenční implementace pro formát AV1. Kodér je nazván **aomenc** a dekodér **aomdec**. Jako původně výzkumný kodek ukazuje efektivní využití veškerých nových vlastností a funkcí představených s formátem AV1, avšak za cenu nižší rychlosti kódování a dekódování než v případě ostatních kodeků. Tato rychlost se postupně s dalšími optimalizacemi zvyšuje. (32) Toto tvrzení je podloženo i testem v kapitole 5.5, kde je porovnání rychlosti kodéru aomenc ve verzi z března 2019 s verzí z března 2020.

3.4.2 Rav1e a dav1d

Rav1e je open source kodér formátu AV1, který má určeny tyto cíle:

- Být rychlejší než libaom
- Mít lepší kvalitu než libaom
- Ideálně splnit oba předchozí cíle současně (35)

V současnosti stále nemá implementovány veškeré nástroje AV1 a není optimalizovaný pro zachování vysoké kvality videa, kterou poskytuje referenční implementace. Rav1e je tedy několikanásobně rychlejší než aomenc, což je zatím vykoupeno nižší kvalitou výsledného kódovaného videa. Tento aspekt je podrobně popsán v kapitole 5.4.

Jméno kodéru je zkratkou „Rust AV1 encoder“. (43) Rust je systémový programovací jazyk vyvinutý společností Mozilla Research, který je syntakticky podobný jazyku C++. Na rozdíl od něj je ale navržen tak, aby poskytoval lepší paměťovou bezpečnost při zachování vysokého výkonu. (44)

Dav1d je dekodér vyvíjený stejnou komunitou, jehož název je podobně jako v případě rav1e, odvozen ze spojení „dav1d is an AV1 decoder“.

3.4.3 SVT-AV1

Open-source implementace vyvíjená zejména společnostmi Intel a Netflix. Kodér je v souladu se specifikací AV1 pro všechny 3 profily (Main, High, Professional) a podporuje všechny nástroje AV1, které napomáhají dosáhnout vysoké efektivity komprese. V porovnání s referenční implementací libaom dosahuje podobné efektivity komprese a současně mnohem nižší kódovací latence na vícejádrových zařízeních díky lepší paralelizaci procesů. Právě škálovatelnost díky paralelizaci byla hlavním cílem Intelu při vývoji kodeku. To umožňuje adaptaci na mnoho procesorových jader včetně nejnovějších serverů, které mohou mít i desítky jader.

SVT-AV1 je napsáno v jazyce C a může být zkompilováno pro všechny majoritní platformy (Linux, Windows a macOS). Kodek má také četné optimalizace pro platformu x86, které zvyšují rychlost kódování. (45)

3.4.4 Libgav1

Dekodér vyvíjený Googlem zaměřený primárně na Android zařízení s architekturou procesorů ARM, kde je referenční aomdec příliš pomalý. První verze byla představena v říjnu 2019. (46) O tomto dekodéru zatím není známo mnoho detailů, dle testování serveru phoronix.com byla verze 2019-10-05 přibližně 7× pomalejší, než dav1d v0.4.0. (47) Je šířen pod svobodnou licencí Apache License 2.0. (48)

3.5 Softwarová podpora

3.5.1 Operační systémy

Windows – v operačním systému Windows 10 je nativní dekódování videí ve formátu AV1 dostupné po stažení volně dostupného rozšíření „AV1 Video Extension (beta)“ z Microsoft Store. Tato podpora byla přidána ve verzi 1809 (vydání z října 2018). (49)

MacOS – v aktuální verzi macOS Catalina není formát nativně podporován vůbec. Jedinou možností je využít software třetích stran, viz kapitola 3.5.2.

Linux – podpora v Linuxu je závislá na nainstalovaných kodecích. Populární kolekce knihoven a nástrojů pro práci s mediálními formáty, mezi které se řadí například FFmpeg a Libav, kodeky pro formát AV1 obsahují. Tyto kolekce jsou zahrnuty v základní instalaci mnoha linuxových distribucí.

Chrome OS – dekodér pro AV1 byl přidán ve verzi Chrome OS 70, která byla vydána v říjnu 2018. (50) V současné verzi pro dekódování využívá rav1e, volitelně lze zkompilovat verzi s dekodérem libgav1 pro zařízení postavené na architektuře ARM. (51)

Android – podpora formátu AV1 byla přidána v Androidu 10 (původně označovaném jako Android Q). První beta verze byla vydána v březnu 2019, finální pak v září téhož roku. Ve stejné verzi byla představena i podpora pro HDR10+ a audio kodek Opus. (52)

iOS – současná verze iOS 13 nemá nativní podporu formátu AV1. Lze využít aplikační software třetích stran, viz 3.5.2.

Ačkoliv se Apple přidal k AOMedia již v lednu 2018, nativní softwarovou podporu zatím nemá jeho mobilní ani desktopová platforma.

3.5.2 Aplikace

Dekódování formátu AV1 je podporováno ve většině současných desktopových webových prohlížečů, jako je **Chrome**, **Firefox**, **Opera** a **Microsoft Edge** (s příslušným doplňkem popsaným v 3.5.1). V prohlížeči Safari videa ve formátu AV1 z důvodu neexistující podpory dekódování AV1 v MacOS nelze přehrávat. (53)

Multimediální přehrávač **VLC** na desktopech experimentálně podporoval dekódování formátu AV1 od verze 3.0, která přišla v únoru 2018. Mobilní verze pro Android a iOS začaly dekódování formátu podporovat přibližně o rok později a místo dekodéru aomdec využívají dekodér dav1d. (54)

Kolekci svobodného softwaru pro práci se zvukovými a obrazovými materiály **FFmpeg** lze zkompilovat se všemi dostupnými kodeky popsanými v kapitole 3.4. Jako první v dubnu 2018 ve verzi FFmpeg 4.0 „Wu“ přišla podpora kodeku libaom. Jednou ze součástí této kolekce je knihovna libavcodec obsahující kodeky. Tuto knihovnu využívá mnoho volně dostupných i placených softwarových produktů (zcela či jen částečně tuto knihovnu využívají například již zmíněné prohlížeče Chrome a Firefox nebo přehrávač VLC). (55, 56)

3.6 Hardwarová podpora

V době psaní této práce (březen 2020) není hardwarová podpora dekódování videí ve formátu AV1 široká. Ačkoliv první čipy schopné hardwarově dekódovat AV1 byly představeny několika čínskými výrobci již v druhé polovině roku 2019, první zařízení, která tyto čipy využívají, přichází na trh až nyní. Prvním čipem pro mobilní telefony s podporou hardwarového dekódování AV1 je MediaTek Dimensity 1000 a jeho slabší verze 1000L. Na evropském trhu se zatím žádné zařízení s tímto čipem neprodává. (57) Americká společnost Qualcomm,

která je největším hráčem v segmentu SOC pro mobilní telefony (58), zatím žádný čip schopný hardwarově dekódovat formát AV1 nepředstavila.

Výrobci grafických karet (ať už integrovaných do jednoho čipu s procesorem, tak dedikovaných), jako jsou společnosti Intel, AMD a Nvidia, zatím hardwarovou podporu AV1 s žádným novým produktem oficiálně neoznámili. Různé spekulace a úniky hovoří o tom, že by hardwarová podpora AV1 mohla přijít s novou generací procesorů Intel Rocket Lake, jejíž uvedení na trh se očekává nejdříve ke konci roku 2020, spíše však v první polovině roku 2021. (59)

V lednu 2020 společnost LG oznámila, že všechny její 8K televize představené v roce 2020 s procesorem α9 třetí generace, podporují dekódování AV1. (60)

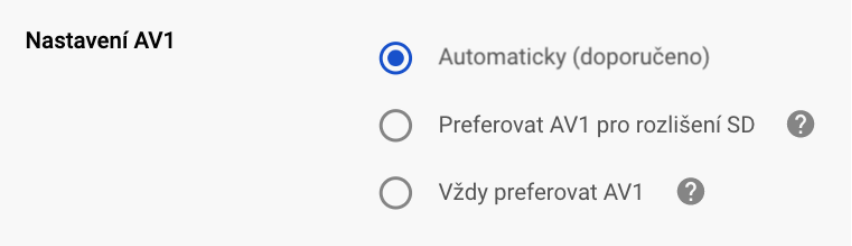
3.7 Praktické využití formátu

V současnosti (březen 2020) je z „velkých hráčů“ ve využití formátu nejdále Netflix a YouTube.

V únoru 2020 začal **Netflix** streamovat videa ve formátu AV1 do zařízení s operačním systémem Android. Stačí si v mobilní aplikaci povolit funkci úspory dat. Netflix zavedením AV1 slibuje 20% zlepšení efektivity komprese oproti VP9. Nižší spotřeba dat je zatím vykoupena vyšší spotřebou samotného zařízení, na kterém je obsah přehráván, protože zatím chybí hardwarová podpora dekódování. Pro dekódování aplikace využívá dekodér dav1d, u kterého společnost nejdříve zoptimalizovala dekódování 10-bit obsahu na procesorech s architekturou ARM. Nyní Netflix sponzoruje další open-source snahy o optimalizaci dekódování 10-bit videí. Cílem společnosti je v budoucnu využívat AV1 všude. (61)

YouTube již v září 2018 zpřístupnilo playlist videí překódovaných do AV1. Podpora v té době byla pouze v prohlížeči Chrome při povolení experimentálních funkcí a v beta verzi Firefoxu. Videa v tomto playlistu byla překódována do AV1 pouze do rozlišení 480p, varianty s vyšším rozlišením používaly VP9. (62) Nyní YouTube používá formát AV1 u vybraných videí v základu (není nutné měnit nastavení YouTube účtu, dokonce není nutné být přihlášen). AV1 je u takových videí dostupné pouze v rozlišení 1080p a nižším, vyšší rozlišení využívá VP9. Z mého vlastního testování je patrné, že AV1 je dostupné u videí, která jsou alespoň měsíc stará, ale ne starší než dva roky, a počet zhlédnutí se u nich počítá v milionech. Dalším poznatkem je, že se tato videa automaticky spouští v AV1 v prohlížečích Google Chrome a Mozilla Firefox na všech třech hlavních desktopových platformách (Windows, macOS, Linux (Ubuntu)). Kromě prohlížeče a operačního systému záleží také na hardware – na starém notebooku z roku 2006 lze AV1 vynutit v nastavení, ale v základu (volba Automaticky) se videa přehrávají ve formátu VP9. Pokusy o přehrání na slabém Chromebooku s procesorem Intel Celeron N3060 skončily také neúspěšně – v tomto případě v nastavení Youtube kompletně chyběla položka Nastavení AV1, ačkoliv jsem si ověřil, že toto zařízení s Chrome OS videa ve formátu AV1 přehraje, viz 3.5.1. Tato položka v nastavení logicky chyběla i v nepodporovaných prohlížečích, jako je Safari v macOS nebo Edge ve Windows. Nic na tomto faktu v případě prohlížeče Edge nezměnilo ani doinstalování rozší-

ření „AV1 Video Extension (beta)“ z Microsoft Store, který byl popsán v kapitole 3.5.1. Stejným neúspěchem skončil i test prohlížeče Microsoft Edge Beta, který je v základu postavený, stejně jako Chrome, na projektu Chromium.



Obrázek 3 Nastavení AV1 na platformě YouTube

Na konci dubna 2020 vyšla aktualizace aplikace YouTube pro Android TV, která přidala podporu AV1 na vybraných zařízeních schopných hardwarového dekódování tohoto formátu. (63)

Mezi další online služby, které nyní formát AV1 interně testují, ale s reálným nasazením pro širokou veřejnost v nejbližší době nepočítají, lze zařadit **Vimeo**, **Facebook** a **Twitch**. (32)

4 Metody objektivního hodnocení kvality videa

Obrazová kvalita může být hodnocena objektivními či subjektivními metodami. Subjektivní metody jsou založeny na hodnocení kvality pozorovateli (lidmi), kteří nejsou odborníky v dané problematice. Běžně se subjektivní metody dělí na tzv. single-stimulus a double-stimulus. V případě single-stimulus hodnocení pozorovatel hodnotí pouze kvalitu testovacího videa, originální video nezná. U druhého zmíněného typu subjektivních metod má divák k dispozici kromě testovacího i originální (referenční) video. Výstupy subjektivních metod jsou hodnotnější, ale provedení je velice nákladné.

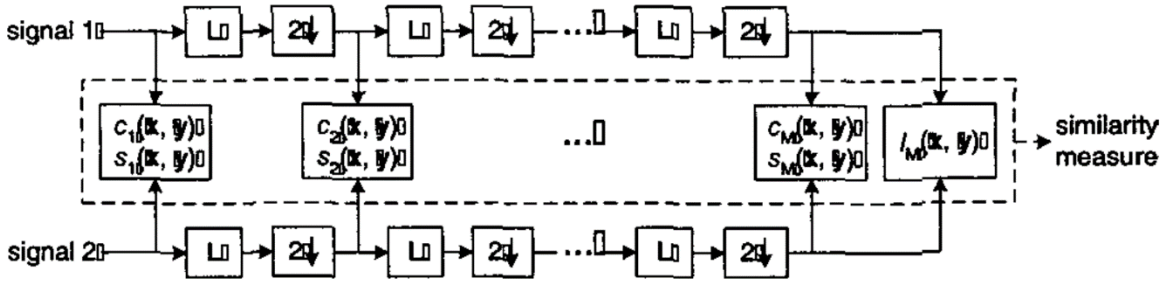
Na druhé straně jsou metody objektivní, které využívají řadu různých algoritmů k analýze poškození a chyb obrazu. Obecně platí, že hlavním cílem všech objektivních metod je se svými výsledky co nejvíce přibližovat subjektivním metodám. Metody objektivního hodnocení kvality se dělí do tří skupin – full-reference (FR), reduced-reference (RR) a no-reference (NR). První typ hodnotí kvalitu testovacího obrazu pomocí jeho porovnání s referenčním obrazem, u kterého je předpoklad, že kvalita je maximální možná. RR metriky hodnotí kvalitu testovacího a referenčního videa na základě porovnání rysů, které jsou z obou obrazů vytaženy. Posledním typem jsou no-reference metody, které se pokouší o hodnocení kvality testovacího obrazu bez porovnání s originálem. (64, 65)

Tato kapitola se věnuje vybraným full-reference objektivním metodám.

4.1 MS-SSIM

Metoda Multi-Scale Structural SIMilarity (MS-SSIM) je založena na předpokladu, že lidský vizuální systém je velice citlivý na strukturální informace ve sledované scéně. Z toho lze usoudit, že metoda měření strukturální podobnosti mezi původní (nekomprimovanou) scénou a scénou po komprimaci může poskytnout dobrý odhad vnímané kvality výsledného zkomprimovaného obrazu.

Metoda vychází z metody SSIM, která počítá strukturální podobnost z jedné (původní) velikosti obrazu. MS-SSIM měří strukturální podobnost dvou obrazů nejen v původní velikosti, ale iterativně aplikuje low-pass filtr a snižuje velikost obrazu na polovinu. Tento přístup vychází z toho, že lidské vnímání detailů nezávisí pouze na množství detailů obrazu samotného, ale také na vzdálenosti obrazu od pozorovatele a percepční schopnosti jeho vizuálního systému. (66)



Obrázek 4 Schéma metody MS-SSIM (66)

- *signal 1 = x, signal 2 = y* – dvě vzájemně zarovnané diskrétní křivky
- *L* – aplikace low-pass filtru
- *2↓* – snížení velikosti obrazu na polovinu
- $c_j(x,y)$ – výsledek porovnání kontrastu obrazu dvou signálů
- $s_j(x,y)$ – výsledek porovnání struktury obrazu dvou signálů
- $l_j(x,y)$ – výsledek porovnání jasu obrazu dvou signálů
- *M* – index nejvyšší iterace

Z tohoto schématu lze vyčíst, že porovnání kontrastu a struktury obrazu dvou signálů probíhá při každé iteraci a porovnání jasu pouze při dosažení M-té iterace.

Toto schéma lze popsat a metodu MS-SSIM vyhodnotit pomocí matematického zápisu:

$$MS\text{–}SSIM(x,y) = [l_M(x,y)]^{\alpha_M} \cdot \prod_{j=1}^{M} [c_j(x,y)]^{\beta_j} [s_j(x,y)]^{\gamma_j}$$

kde exponenty α_M, β_j a γ_j jsou použity k úpravě relativní důležitosti různých komponent (kontrast, struktura a jas). (66)

4.2 PSNR-HVS-M

Vychází z metody PSNR-HVS, jejíž název je zkratkou slov Peak Signal to Noise Ratio based on Human Visual System neboli špičkový odstup signálu k šumu založený na lidském zrakovém vnímání. Signálem jsou originální data a šumem jsou chyby způsobené při kompresi. PSNR-HVS, na rozdíl od PSNR, bere v potaz různé vlastnosti lidského zrakového systému, jako je například vnímání kontrastu. PSNR-HVS-M dále zohledňuje kontrastní maskování.

Nejběžnější jednotkou veličiny PSNR je decibel. Čím vyšší jsou její hodnoty, tím je obraz kvalitnější (resp. podobnější svému originálu).

Veličina PSNR je definována takto:

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

- MAX_I – nejvyšší možná hodnota obrazového pixelu – v případě barevné hloubky 8-bit je tato hodnota 255
- MSE – střední kvadratická chyba (rozptyl chyb):

$$MSE_{PSNR} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(\delta(i,j))^2$$

$$\delta(i,j) = [a(i,j) - \hat{a}(i,j)]$$

- M – výška obrazu
- N – šířka obrazu
- a – původní snímek
- $\hat{a}$ – komprimovaný snímek
- (i,j) – pozice pixelu

Z výše uvedených vzorců je zřejmé, že pokud jsou porovnávané obrazy stejné, PSNR není definováno, jelikož by MSE bylo rovno nule, kterou nelze dělit.

Výpočet metody PSNR-HVS i PSNR-HVS-M se řídí stejným vzorcem, jako PSNR, liší se způsob výpočtu střední kvadratické chyby. PSNR-HVS dělí obraz na bloky o rozměru 8×8 pixelů. Rozdíl mezi originálním a zkomprimovaným blokem ($\delta(i,j)$) je pro každý blok vážen koeficienty funkce CSF (Contrast Sensitivity Function). $\delta(i,j)$ je zde vypočítáno využitím koeficientů diskrétní kosinové transformace (DCT):

$$\delta_{PSNR-HVS}(i,j) = \delta(i,j) \cdot CSF_{Cof}(i,j)$$

Metoda PSNR-HVS-M dále pro každý 8×8 blok zohledňuje kontrastní maskování (CM):

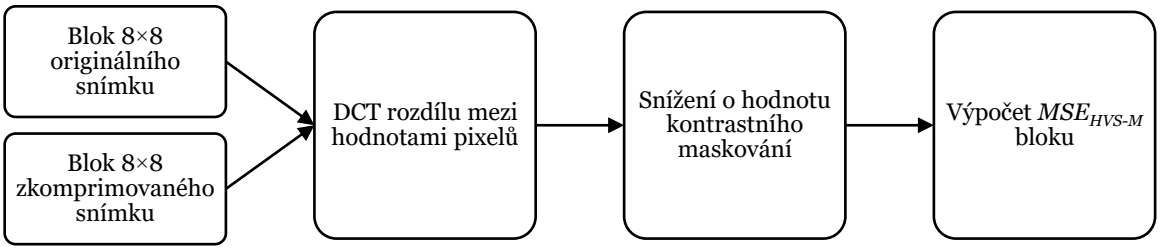
$$\delta_{PSNR-HVS-M}(i,j) = (\delta(i,j) \cdot CM(i,j)) \cdot CSF_{Cof}(i,j)$$

Výpočet střední kvadratické chyby metody PSNR-HVS-M:

$$MSE_{HVS-M} = \frac{1}{MN}\sum_{I=1}^{M/8}\sum_{J=1}^{N/8}\left(\sum_{i=1}^{8}\sum_{j=1}^{8}(\delta_{PSNR-HVS-M}(i,j))^2\right)$$

- (I,J) – pozice bloku 8×8 ve snímku
- (i,j) – pozice pixelu v bloku 8×8

(67, 68)



Obrázek 5 Schéma metody PSNR-HVS-M (69)

4.3 VMAF

VMAF neboli Video Multimethod Assessment Fusion je nejpokročilejší z uvedených metod. Je vyvíjena společností Netflix ve spolupráci s Univerzitou Jižní Kalifornie a Laboratory for Image and Video Engineering (LIVE) na Texaské univerzitě v Austinu.

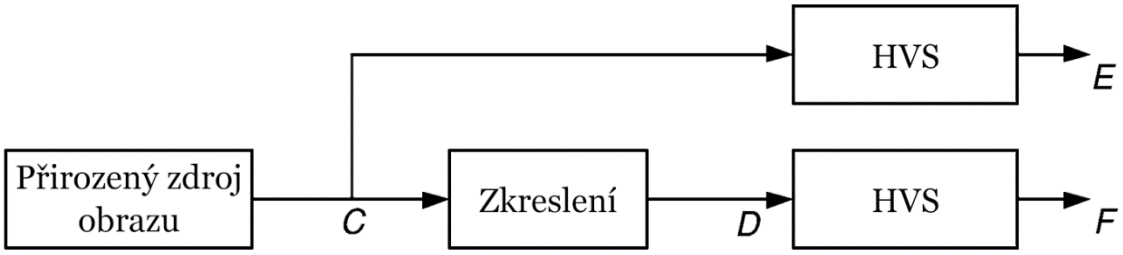
Subjektivní predikce se v případě VMAF získává kombinací několika základních metrik kvality. Zdůvodněním tohoto přístupu je skutečnost, že každá metrika má své silné i slabé stránky – záleží na charakteristice obsahu, typu artefaktů a míře samotného zkreslení způsobeného kompresí. Výsledky těchto metrik jsou následně sloučeny do finální metriky. Algoritmus, který má toto sloučení na starosti, je založen na strojovém učení a přiřazuje váhy jednotlivým základním metrikám takovým způsobem, aby finální metrika zachovala veškeré silné stránky základních metrik a finální skóre tak bylo co nejpřesnější. Název tohoto algoritmu je Support Vector Machine (SVM). Model strojového učení je trénován a testován na základě subjektivního testování a díky němu by výstupy metody VMAF měly co nejvíce korelovat s výsledky subjektivních metod.

Základními metrikami pro určení finální metriky VMAF jsou Visual Information Fidelity (VIF), Detail Loss Metric (DLM) a Temporal feature extraction (TI).

VIF – Visual Information Fidelity

Metrika kvality obrazu založená na předpokladu, že kvalita je komplementární ke ztrátě věrnosti informací. Pokud v obrazu není žádné zkreslení (způsobené například kompresí), tento signál projde lidským zrakovým systémem (HVS) a následně vstoupí do mozku. Pro zkreslený obraz tato metrika předpokládá, že signál prošel přes další „zkreslení“ než vstoupil do HVS, viz Obrázek 6.

Hlavním cílem je změřit pouze tu ztrátu informací, kterou dokáže zachytit lidský mozek (po průchodu HVS člověka).



Obrázek 6 Schéma VIS (70)

Metrika je odvozena z kvantifikace dvou veličin vzájemné informace:

- Vzájemné informace mezi vstupem a výstupem HVS, pokud není přítomné žádné zkreslení (C – E)
    - Informace, které může mozek získat z referenčního obrazu
- Vzájemné informace mezi vstupem „zkreslení“ a výstupem HVS (C – F)
    - Informace, které může mozek získat ze zkresleného obrazu

Tyto dvě veličiny jsou následně zkombinovány za účelem vytvoření míry věrnosti vizuální informace (VIF):

$$VIF = \frac{Vzájemné\ informace(C;F)}{Vzájemné\ informace(C;E)}$$

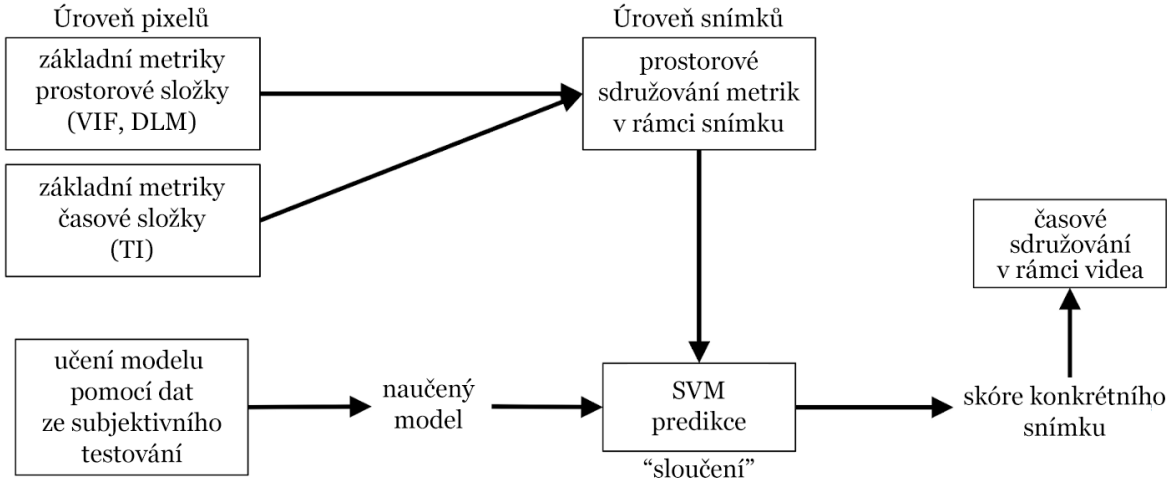
Skóre VIF je běžně reprezentováno jako ztráta věrnosti kombinující čtyři měřítka. V metodě VMAF je metrika VIF upravena tak, že každé měřítko je zastoupeno jako samostatná základní metrika. (65, 70, 71)

DLM – Detail Loss Metric

Měří ztrátu detailů, které ovlivňují viditelnost obsahu a další poškození, která odvádí pozornost diváka.

TI – Temporal feature extraction

Předchozí dvě základní metriky se zaměřují na obrazovou kvalitu konkrétního snímku. TI hodnotí kvalitu na základě časové složky videa. Jedná se o jednoduchou míru rozdílu mezi páry sousedních snímků. Výpočet je prováděn jako průměr absolutních rozdílů hodnoty jasové komponenty pixelů. (65)



Obrázek 7 Schéma metody VMAF (72)

5 Objektivní porovnání kvality a rychlosti

Pro porovnání s formátem AV1 byly vybrány formáty VP9, H.265 a H.264. Byly vybrány z toho důvodu, že jsou v současnosti nejrozšířenější. (73) Formát AV1 je přímým nástupcem formátu VP9. Ten ve většině případů využívá například největší video streamovací služba YouTube. Největším konkurentem formátu VP9 je dnes formát H.265 (HEVC), který se využívá například pro digitální televizní vysílání DVB-T2 na našem území. Formát H.264 (MPEG-4 AVC) je z výběru nejstarší, ale přesto si drží vysoký podíl na trhu. Tento formát prošel dlouhým vývojem, jeho první verze vyšla už v roce 2003, a v roce 2017 vyšla zatím poslední revize 25. (74) AV1 je z výběru nejnovější a je velice zajímavé sledovat, kam se vyvíjí komprimační technologie.

Kodeky, které byly použity pro testování:

- AV1 – libaom
- AV1 – SVT-AV1
- AV1 – rav1e
- VP9 – libvpx
- H.265 – x265
- H.264 – x264

Pro účely objektivního porovnání kvality byly použity všechny 3 metody popsané v kapitole 4. Veškeré testy byly provedeny na počítači s parametry, které uvádí Tabulka 5.

Tabulka 5 Použitý HW a SW

CPU	Intel Core i5-4440 @ 3,1 GHz
RAM	16 GB DDR3 @ 1600 MHz
GPU	AMD R9 280
SSD	Samsung EVO 850 120GB
Systém	Ubuntu 18.04.2 LTS
FFmpeg (měření 2019)	N-93151-gff03418 (19. 2. 2019)
FFmpeg (měření 2020)	N-96947-ge35f966 (12. 3. 2020)
Libavcodec (měření 2019)	58.47.102
Libavcodec (měření 2020)	58.75.100
SVT-AV1	0.8.1
Rav1e	0.3.1
VQMT	1.1
VMAF Development Kit (VDK)	1.3.13

K testování byly vybrány 3 testovací sekvence dostupné na webu Xiph.Org Foundation (75).

Tabulka 6 Parametry testovacích sekvencí

Sekvence	Rozlišení	Barevná hloubka	Subsampling	Framerate	Počet snímků
Park Joy	3 840×2 160	8-bit	4:2:0	50	500
Crowd Run	3 840×2 160	8-bit	4:2:0	50	500
Sintel	4 096×1 744	8-bit	4:2:0	24	240



Obrázek 8 Sekvence Park Joy, Crowd Run a Sintel

Volba bitrate

Hodnoty bitrate byly voleny tak, aby měly při porovnávání co nejvyšší vypovídající hodnotu. Při prvním testu byly zkoušeny hodnoty bitrate od 1 000 kbit/s do 20 000 kbit/s, ale ukázalo se, že některé kodeky takto nízký datový tok ignorují a jejich minimum při kódování 8-bit 4k videa s 50 snímky za sekundu je někde okolo 2 000 kbit/s. Naopak nejvyšší hranice 20 000 kbit/s se zdála jako nedostatečná, kodeky při tomto datovém toku nedosahovaly nikterak oslnivých výsledků. Pro maximalizaci efektivity, o které se zmiňuji v kapitole 5.5, bylo vhodné, aby počet kódovaných datových toků byl dělitelný čtyřmi. Nakonec jsem se tedy rozhodl pro 16 datových toků v rozmezí 2 000 kbit/s – 24 000 kbit/s. Rozdělení je následovné:

- V rozmezí 2 000 kbit/s – 10 000 kbit/s je interval mezi kódovanými datovými toky 1 000 kbit/s
- V rozmezí 12 000 kbit/s – 24 000 kbit/s je interval 2 000 kbit/s

Volba rychlostních presetů

V případě kodeků **x264** a **x265** byl zvolen druhý nejpomalejší preset, tj. „veryslow“. Nejpomalejší preset „placebo“ není dle nápovědy nástroje FFmpeg doporučován, protože přináší zvýšení kvality finálního videa nejvýše o 1 % v porovnání s presetem „veryslow“ za cenu mnohem delší doby kódování. (76)

U **libvpx** byl zvolen nejpomalejší preset.

U kodeku **libaom** byl zvolen defaultní, tj. druhý nejpomalejší preset. Nejpomalejší nebyl zvolen z podobného důvodu, jako preset „placebo“ v případě x264 a x265. Navíc časová náročnost je i v případě druhého nejpomalejšího presetu stále velmi vysoká, viz kapitola 5.5.

Kodér **SVT-AV1** umožňuje nastavení presetu v rozmezí 0 až 8, kde nižší hodnoty znamenají vyšší kvalitu a nižší rychlost. Pro testování bylo přistoupeno k využití rychlostních presetů 1 a 4. Preset 1 byl nakonec z důvodu nízké rychlosti, viz kapitola 5.5, použit pouze pro sekvenci Park Joy. Preset 4 představuje kompromis mezi rychlostí kódování a kvalitou výsledného videa.

Rav1e umožňuje řídit rychlost a kvalitu kódování pomocí presetů v rozmezí 0-10, kde 0 je nejpomalejší a 10 nejrychlejší. Pro testování byly stejně jako v případě SVT-AV1 zvoleny presety 1 a 4.

Je vhodné poznamenat, že rychlostní presety jednotlivých kodérů nejsou vzájemně zaměnitelné – například preset 4 kodéru SVT-AV1 může být interně nakonfigurován jinak než preset 4 kodéru rav1e. Záleží na tvůrcích jednotlivých kodérů, jaké řízení rychlosti naimplementují.

5.1 Příkazy použité pro kódování

Kromě kodéru SVT-AV1 byly všechny kodéry testovány pomocí programu FFmpeg. Plugin do FFmpeg, který umožňuje kódování pomoci SVT-AV1 existuje, ale kódování pomocí něj v mém případě vždy dříve či později skončilo signálem chyby SIGSEGV, proto jsem se uchýlil k použití dodaného nástroje k tomuto kodéru.

H.264 – x264

```
ffmpeg -benchmark -y -i $vstup.y4m -c:v libx264 -preset veryslow -b:v $bitrate -threads 1
-an $vystup.mkv
```

H.265 – x265

```
ffmpeg -benchmark -y -i $vstup.y4m -c:v libx265 -preset veryslow -b:v $bitrate -x265-
params pools=1 -an $vystup.mkv
```

VP9 – libvpx

```
ffmpeg -benchmark -y -i $vstup.y4m -c:v libvpx-vp9 -pass 1 -b:v $bitrate -threads 1 -speed
4 -an -f matroska /dev/null &&
ffmpeg -benchmark -y -i $vstup.y4m -c:v libvpx-vp9 -pass 2 -b:v $bitrate -threads 1 -speed
0 -an $vystup.mkv
```

AV1 – libaom

```
ffmpeg -benchmark -y -i $vstup.y4m -c:v libaom-av1 -b:v $bitrate -strict experimental -
threads 1 -an $vystup.mkv
```

AV1 – rav1e

```
ffmpeg -benchmark -y -i $vstup.y4m -c:v librav1e -b:v $bitrate -speed $preset -threads 1 -an $vystup.mkv
```

AV1 – SVT-AV1

```
SvtAv1EncApp -i $vstup.yuv -w $sirka -h $vyska -n $pocetSnimku --fps $fps --rc 2 --tbr $bitrate --preset $preset --lp 1 -b $vystup.mkv
```

Proměnné

- **$vstup.y4m** – cesta ke vstupnímu souboru ve formátu YUV4MPEG2
- **$vystup.mkv** – cesta, kam se má uložit překódované video ve zvoleném kódovacím formátu. Jako kontejner je zvolena Matroska (.mkv)
- **$bitrate** – hodnota bitrate
- **$preset** – rychlostní přednastavení kodéru
- **$sirka** a **$vyska** – rozměry videa v pixelech
- **$pocetSnimku** – celkový počet snímků sekvence
- **$fps** – snímková frekvence videa

FFmpeg přepínače

- **-benchmark** – přidá do konzole po každém překódovaném videu informaci o tom, jak dlouho daná operace trvala a jaké maximální množství paměti vyžadovala
- **-y** – povolení přepsání existujících souborů
- **-i** – cesta ke vstupnímu souboru (následováno proměnnou $vstup)
- **-c:v** – zvolený kodek
- **-preset** – zvolený preset (poměr rychlost/kvalita) (x264 a x265)
- **-speed** – zvolený preset (poměr rychlost/kvalita) (libvpx-vp9, rav1e)
- **-b:v** – cílový bitrate výstupního souboru (následováno proměnnou $bitrate)
- **-an** – kódování bez zvuku
- **-threads** – omezení počtu používaných vláken procesoru (x264, libvpx-vp9, libaom-av1)
- **-x265-params pools=1** – omezení používaných vláken procesoru (x265)
- **-pass** – pořadí průchodu (1/2)
- **-f** – formát výstupního souboru
- **-strict experimental** – povolení kódování AV1, z důvodu nízké rychlosti je stále experimentální

SVT-AV1 přepínače

- **-i** – cesta ke vstupnímu souboru (následováno proměnnou $vstup)
- **-w** – šířka videa v pixelech (následováno proměnnou $sirka)
- **-h** – výška videa v pixelech (následováno proměnnou $vyska)

- **-n** – počet snímků (následováno proměnnou $pocetSnimku)
- **--fps** – snímková frekvence videa (následováno proměnnou $fps)
- **--rc** – řízení datového toku – hodnota 2 znamená variabilní bitrate
- **--tbr** – cílový bitrate výstupního souboru (následováno proměnnou $bitrate)
- **--preset** – preset kodéru, hodnoty v rozmezí 0 (nejvyšší kvalita) až 8 (nejvyšší rychlost), následováno proměnnou $preset
- **--lp** – omezení počtu používaných vláken procesoru
- **-b** – specifikace výstupního souboru

5.2 VQMT a VMAF příkazy použité pro testování

Pro testování pomocí metod popsaných v kapitole 4 byly využity dva nástroje. Pro metodu PSNR-HVS-M a MS-SSIM to byl nástroj VQMT (77), pro metodu VMAF to byla knihovna VMAF napsaná v Pythonu (78).

VQMT

```
vqmt $originalni.yuv $prekodovane.yuv $vyska $sirka $pocetSnimku 1 $vystup PSNRHVSM MSSSIM
```

VMAF

```
run_vmaf yuv420p $sirka $vyska $originalni.yuv $prekodovane.yuv --out-fmt text > $vystup
```

Výstupem jsou textové soubory. Jejich obsahem jsou výsledky jednotlivých metod – jak za jednotlivé snímky, tak zprůměrované za celou sekvenci. Pro rychlé získání jednotlivých údajů ze všech textových souborů pro danou sekvenci byl použit nástroj grep.

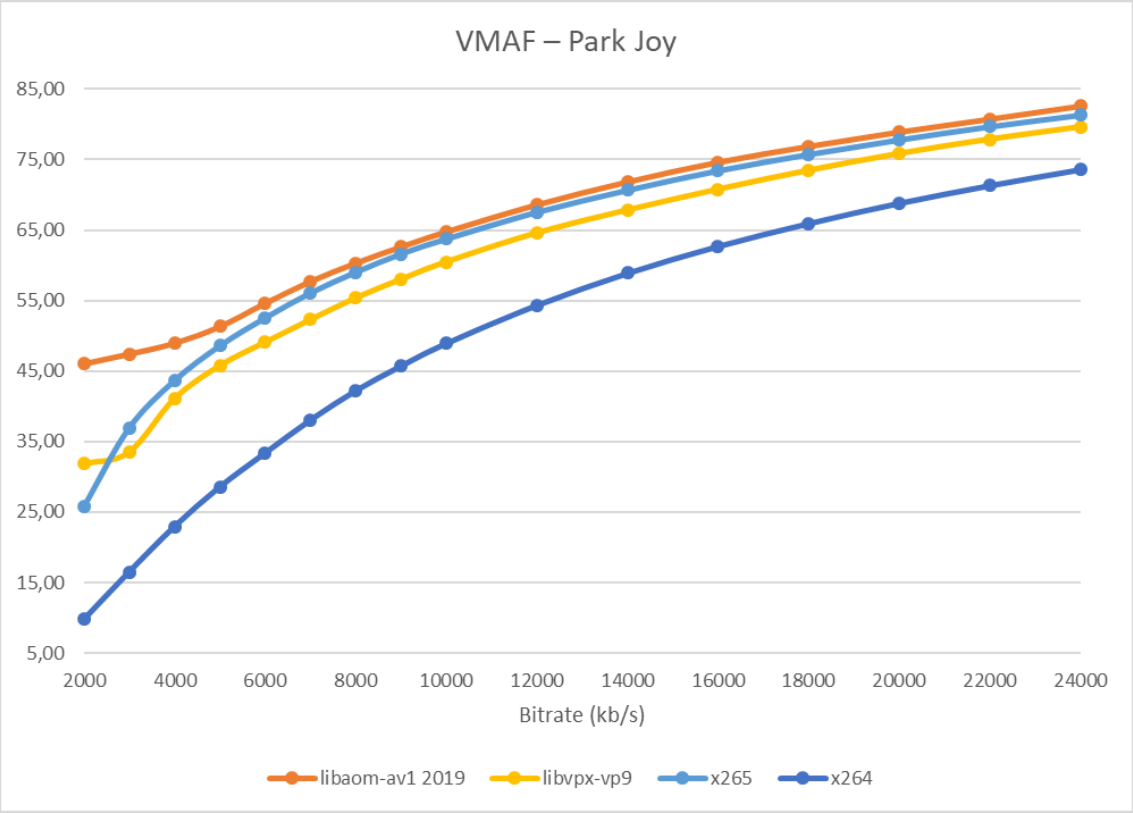
Jelikož bylo potřeba zadat více než 700 příkazů, nebylo možné všechny zadávat do příkazového řádku manuálně. Proto byl použit skript pro shell Bash, který se staral o obsluhu jak samotného kódování, tak hodnocení pomocí nástrojů VQMT a VMAF. Základ tohoto skriptu mi poskytl Jaroslav Svoboda, který ho společně s Michelem Maffeiem vytvořil pro podobný projekt. Za poskytnutí jim tímto děkuji. Skript byl upraven pro potřeby této bakalářské práce, byly doplněny všechny potřebné kodéry a další nástroje sloužící například pro logování konkrétních procesů (délka trvání, využití paměti) nebo zasílání notifikací o průběhu kódování a vyhodnocení e-mailem.

5.3 Objektivní porovnání kvality s ostatními formáty

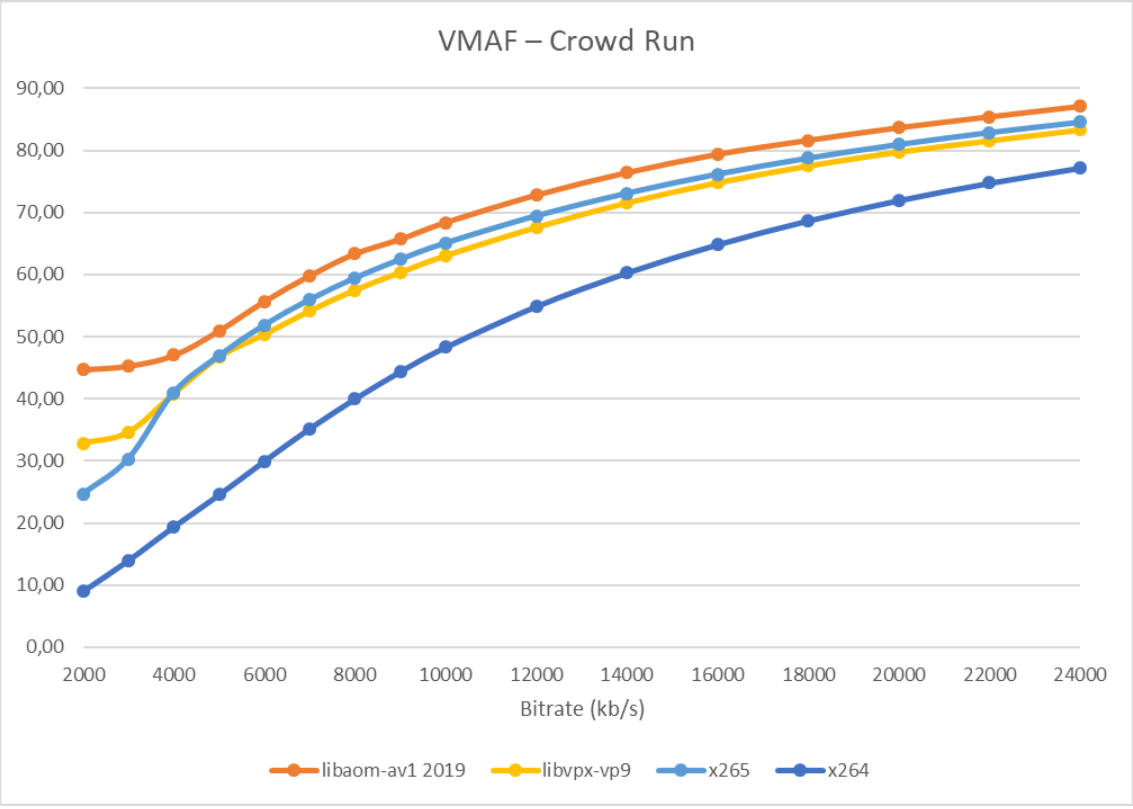
První část objektivního hodnocení kvality videa je zaměřena na porovnání referenční implementace formátu AV1 – libaom s kodéry ostatních formátů.

Všechny hodnoty jsou uvedeny v příloze A.

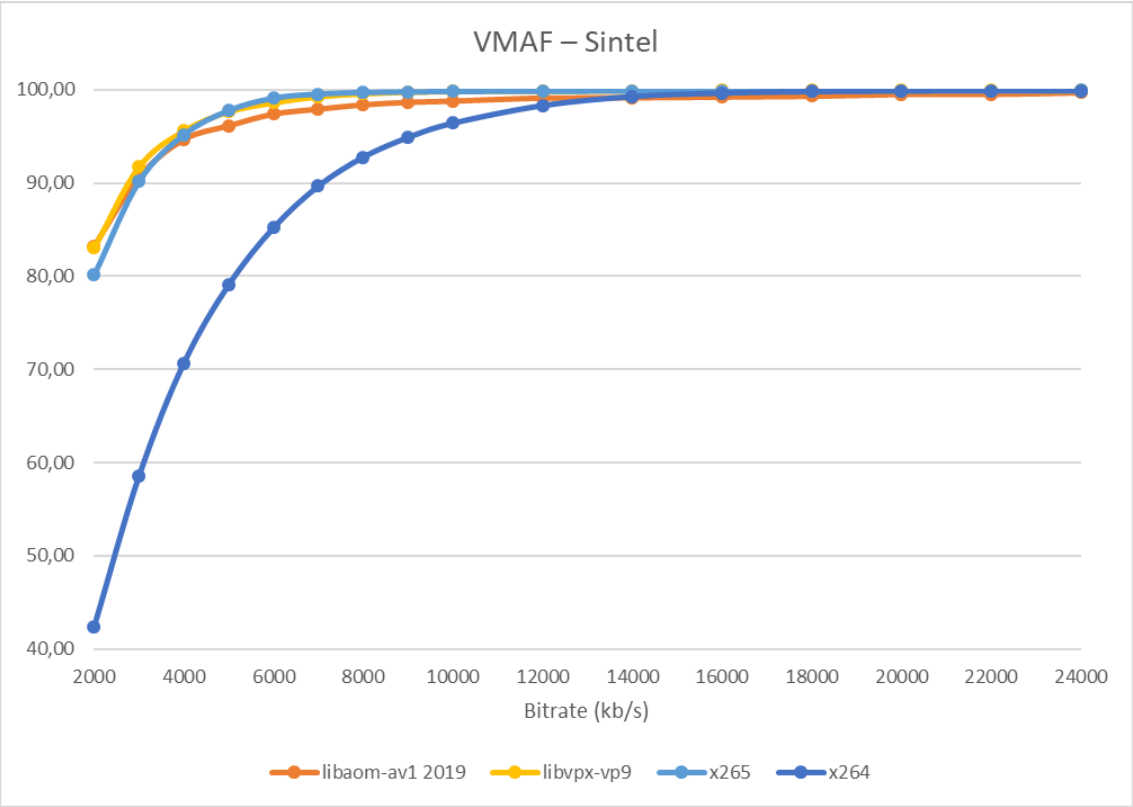
5.3.1 VMAF



Obrázek 9 Výsledek testu VMAF sekvence Park Joy – porovnání s ostatními formáty

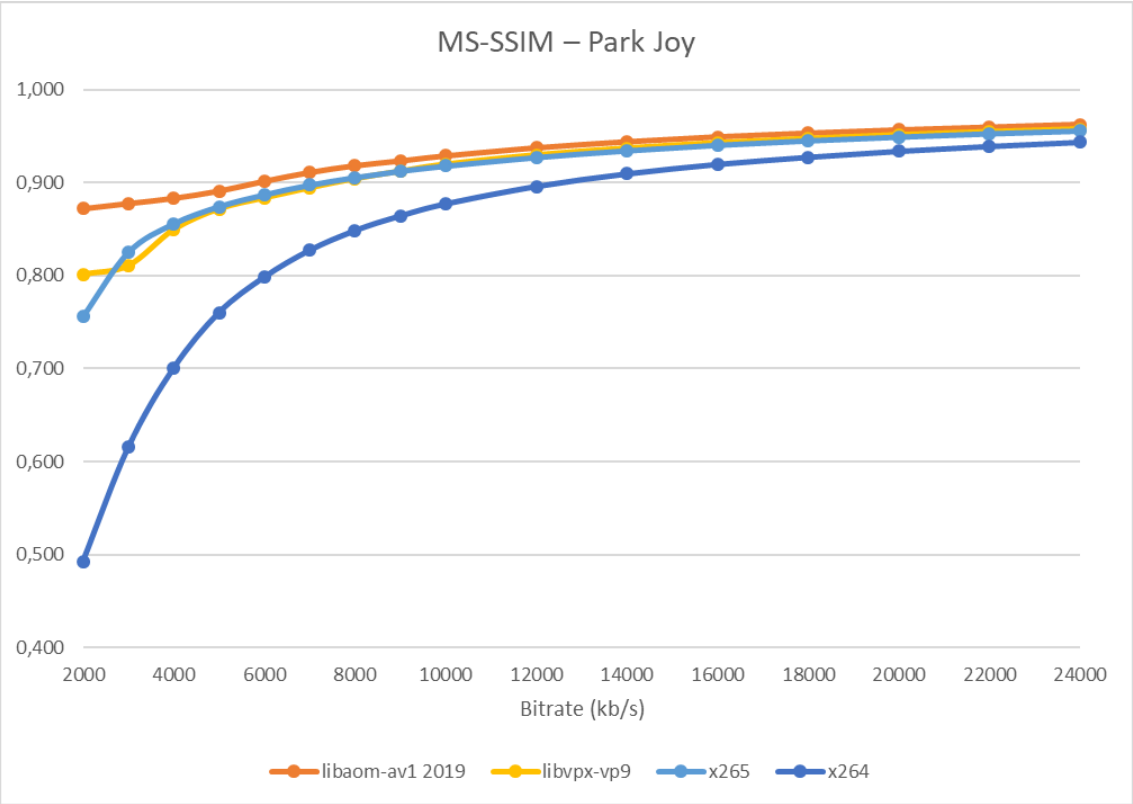


Obrázek 10 Výsledek testu VMAF sekvence Crowd Run – porovnání s ostatními formáty

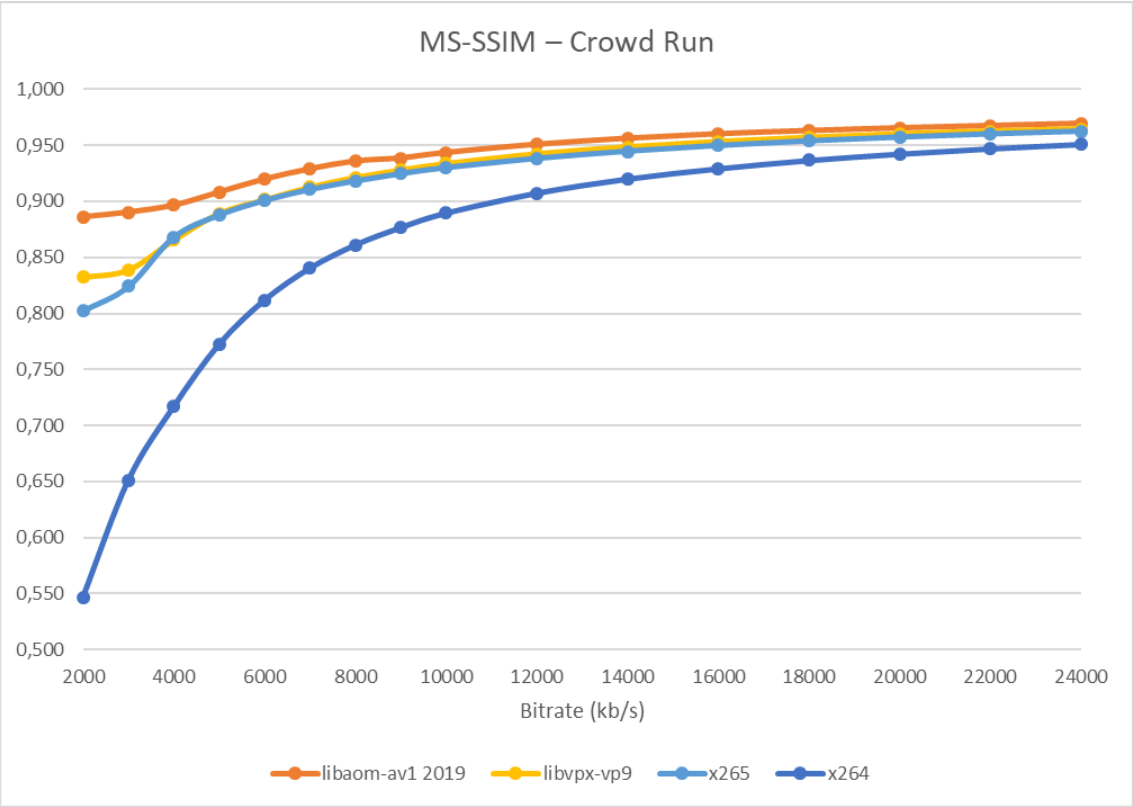


Obrázek 11 Výsledek testu VMAF sekvence Sintel – porovnání s ostatními formáty

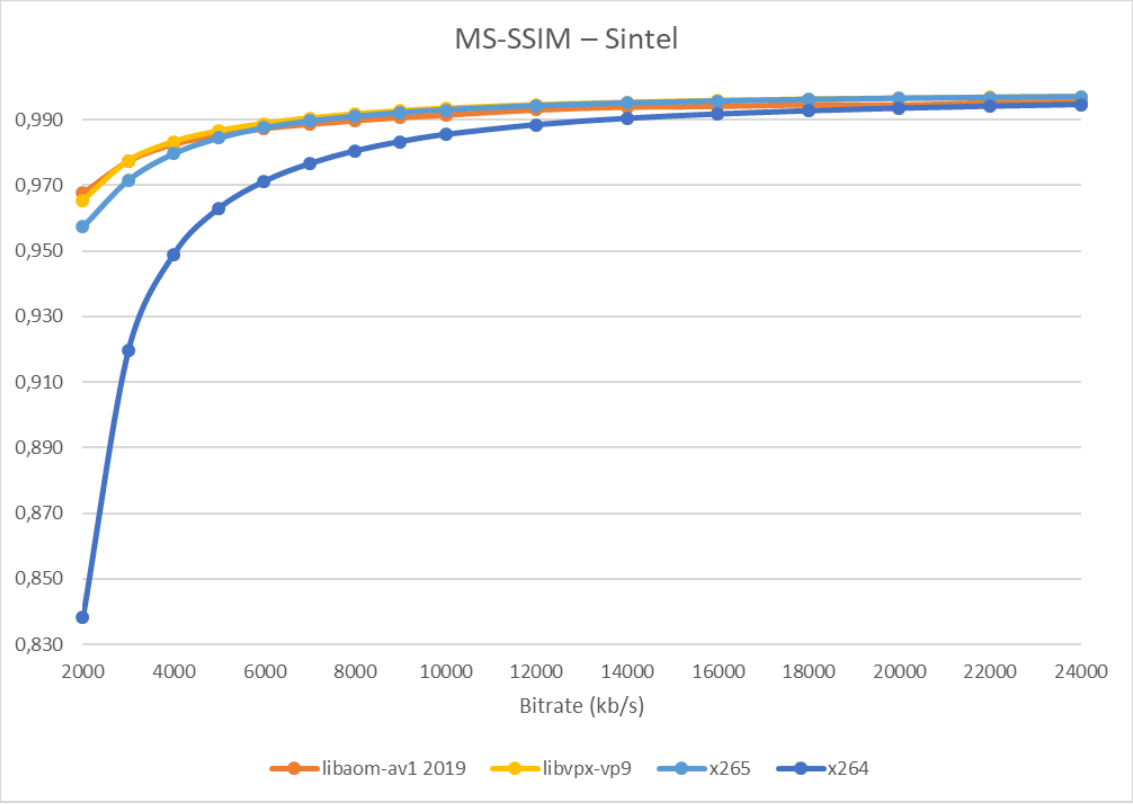
5.3.2 MS-SSIM



Obrázek 12 Výsledek testu MS-SSIM sekvence Park Joy – porovnání s ostatními formáty

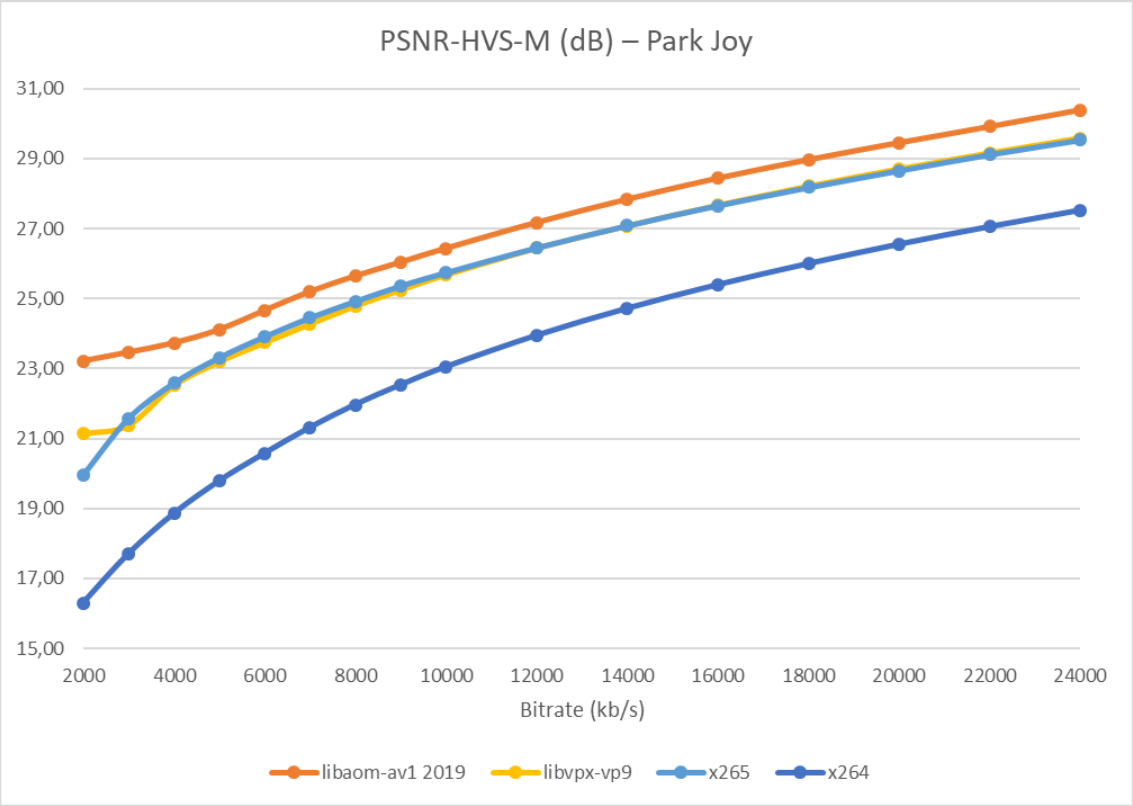


Obrázek 13 Výsledek testu MS-SSIM sekvence Crowd Run – porovnání s ostatními formáty

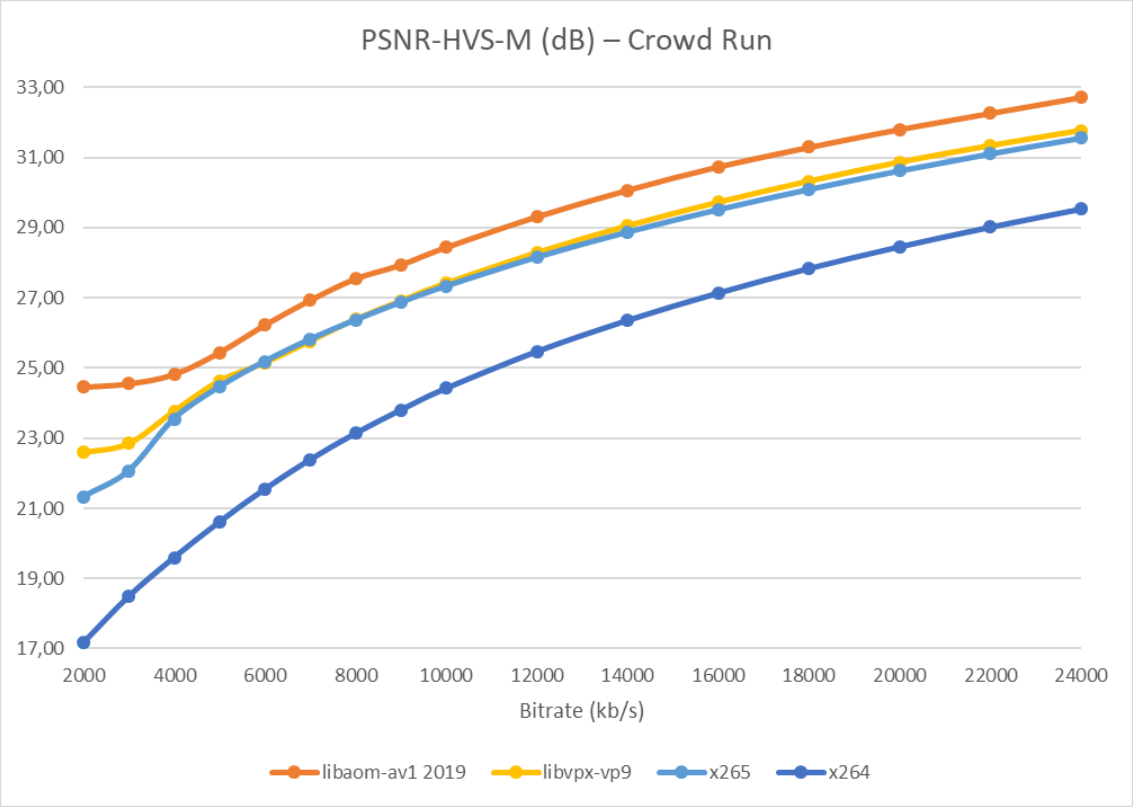


Obrázek 14 Výsledek testu MS-SSIM sekvence Sintel – porovnání s ostatními formáty

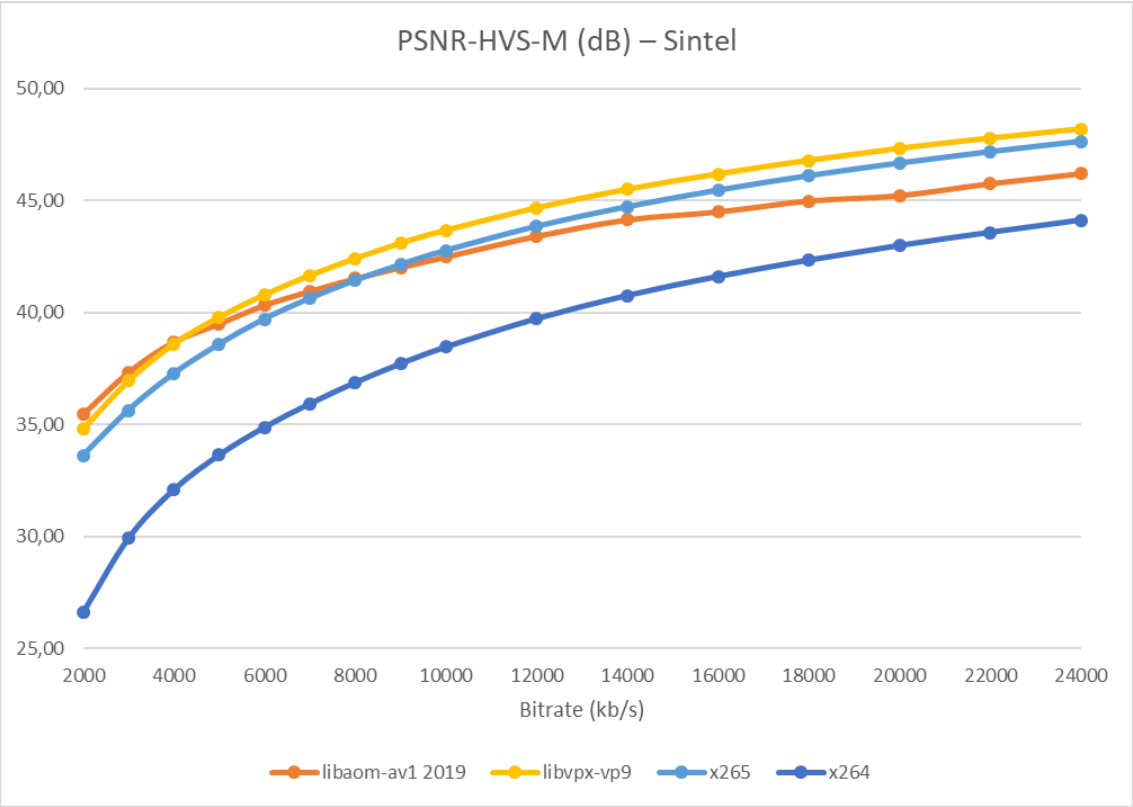
5.3.3 PSNR-HVS-M



Obrázek 15 Výsledek testu PSNR-HVS-M sekvence Park Joy – porovnání s ostatními formáty



Obrázek 16 Výsledek testu PSNR-HVS-M sekvence Crowd Run – porovnání s ostatními formáty



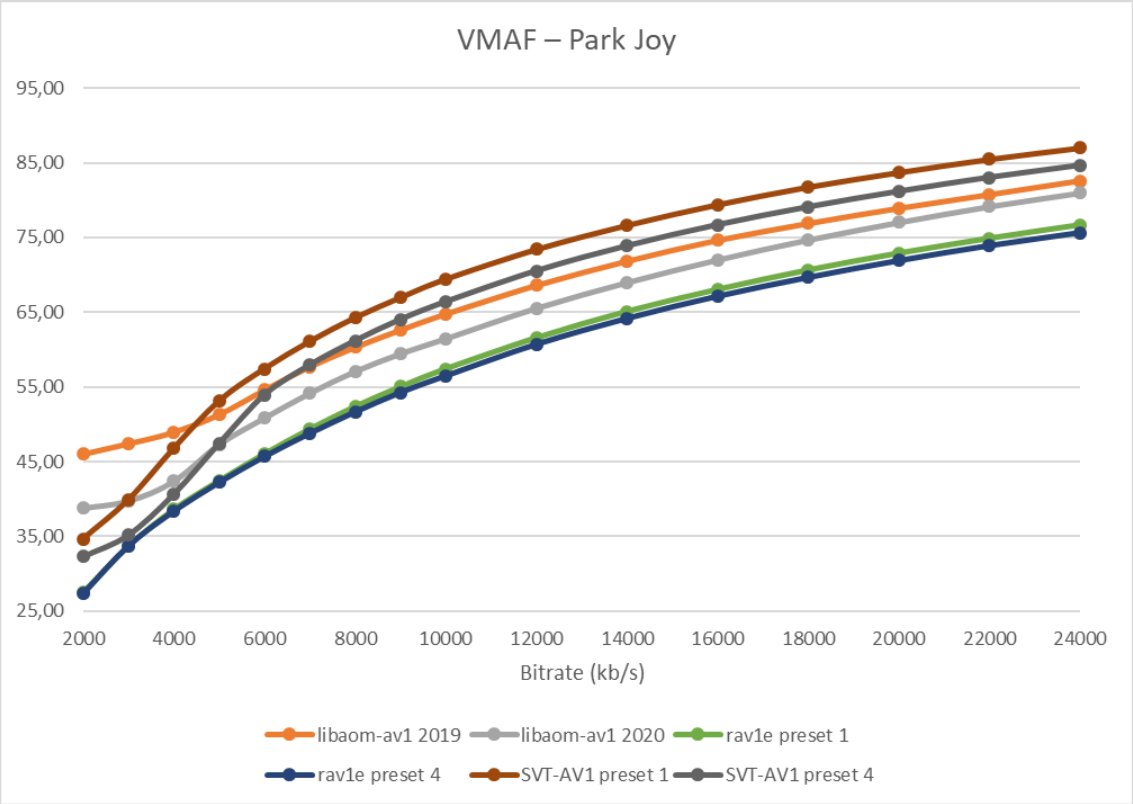
Obrázek 17 Výsledek testu PSNR-HVS-M sekvence Sintel – porovnání s ostatními formáty

Výsledky všech tří metod objektivního porovnání byly konzistentní. Všechny metody prokázaly, že formát H.264, potažmo jeho implementace x264 je, dle očekávání, nejhorší. To je vidět zejména při nižších datových tocích. Při vyšších datových tocích se rozdíly mezi jednotlivými kodeky zmenšují. U sekvencí Park Joy a Crowd Run byl, dle všech tři metod, formát AV1 a kodek libaom znatelně lepší než ostatní. U sekvence Sintel byly formáty AV1, H.265 a VP9 velice vyrovnané, a AV1 dokonce v některých případech vykazovalo mírně horší výsledky než zbylé dva. H.264 byl i v případě této sekvence nejhorší, ale metody VMAF a MS-SSIM poukázaly na to, že jakmile bitrate v případě této sekvence dosáhne přibližně 16 Mb/s, dosahují všechny kodeky, včetně x264, téměř totožných výsledků. Metoda PSNR-HVS-M však kodek x264 vyhodnotila mnohem hůře, a dle této metody tento kodek ani v nejvyšších měřených datových tocích nedosahuje takových kvalit, jako ostatní kodeky. Zajímavé je sledovat, že dle této metody je i formát AV1 a jeho kodek libaom horší, než VP9 (libvpx) a H.265 (x265), které jsou dle této metody velice podobné.

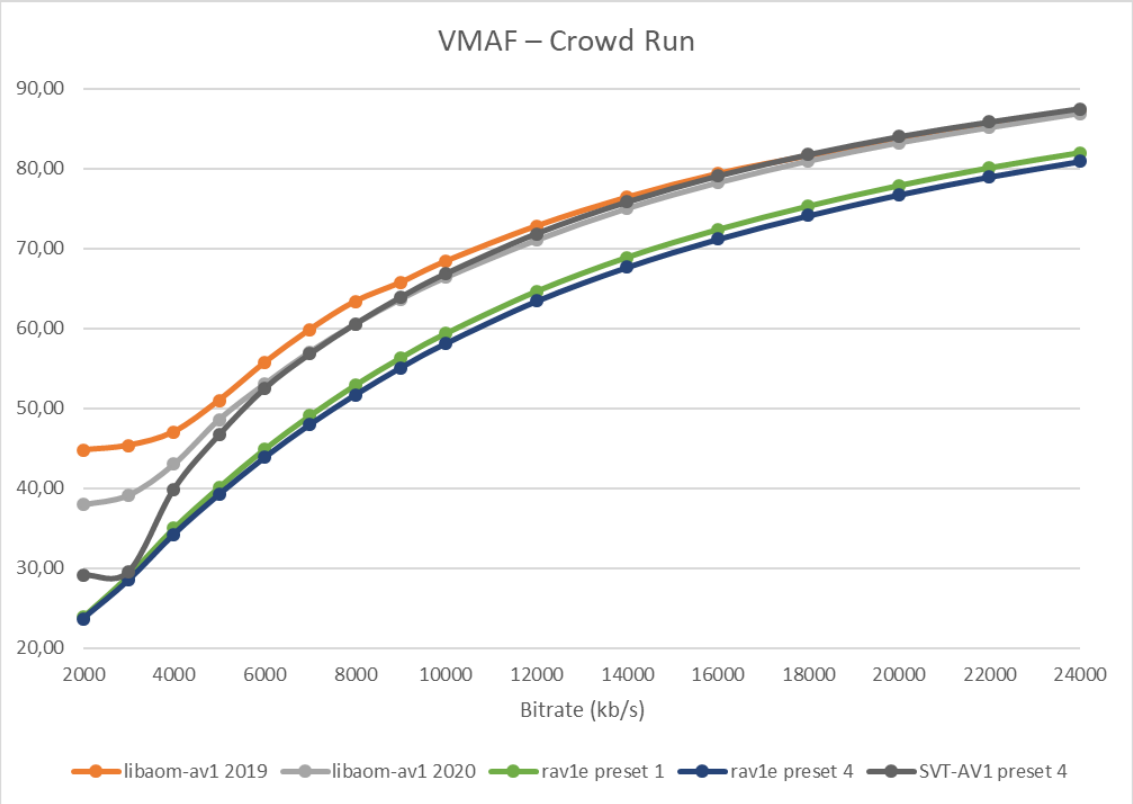
5.4 Objektivní porovnání kvality AV1 kodérů

Druhá část objektivního hodnocení kvality se zaměřuje na porovnání kvality různých implementací formátu AV1 – libaom, rav1e a SVT-AV1.

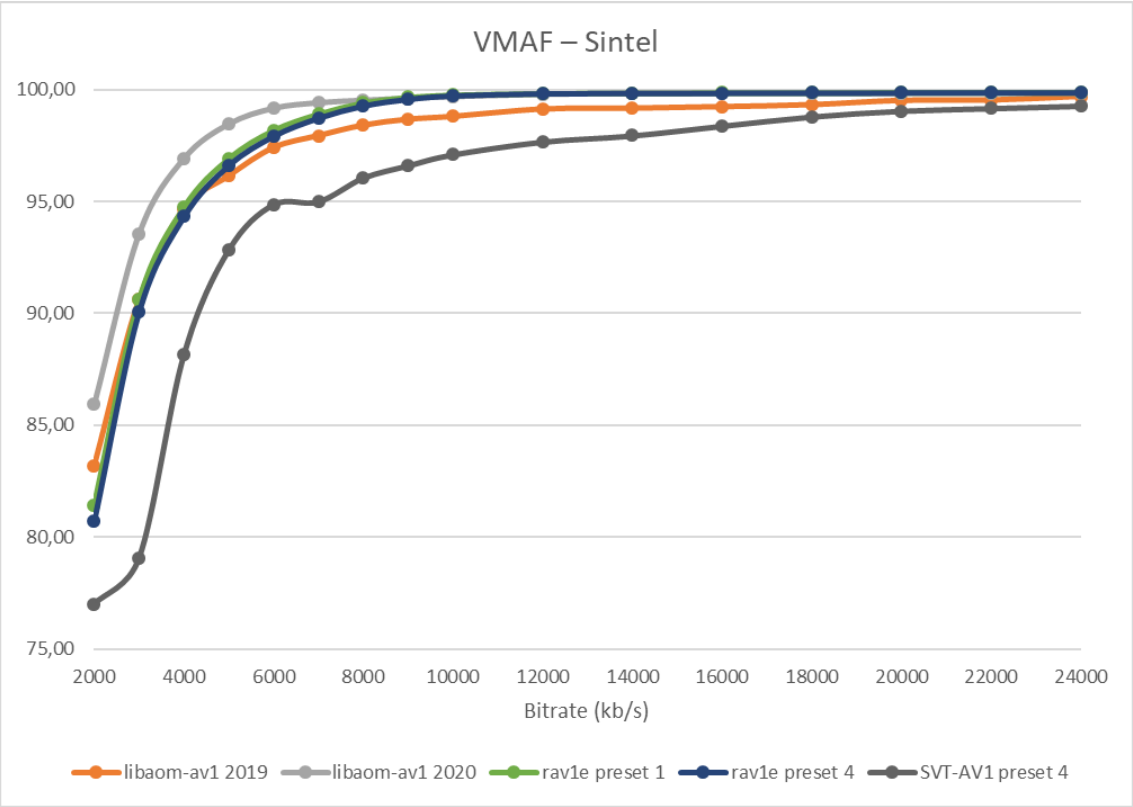
5.4.1 VMAF



Obrázek 18 Výsledek testu VMAF sekvence Park Joy – porovnání kodérů AV1

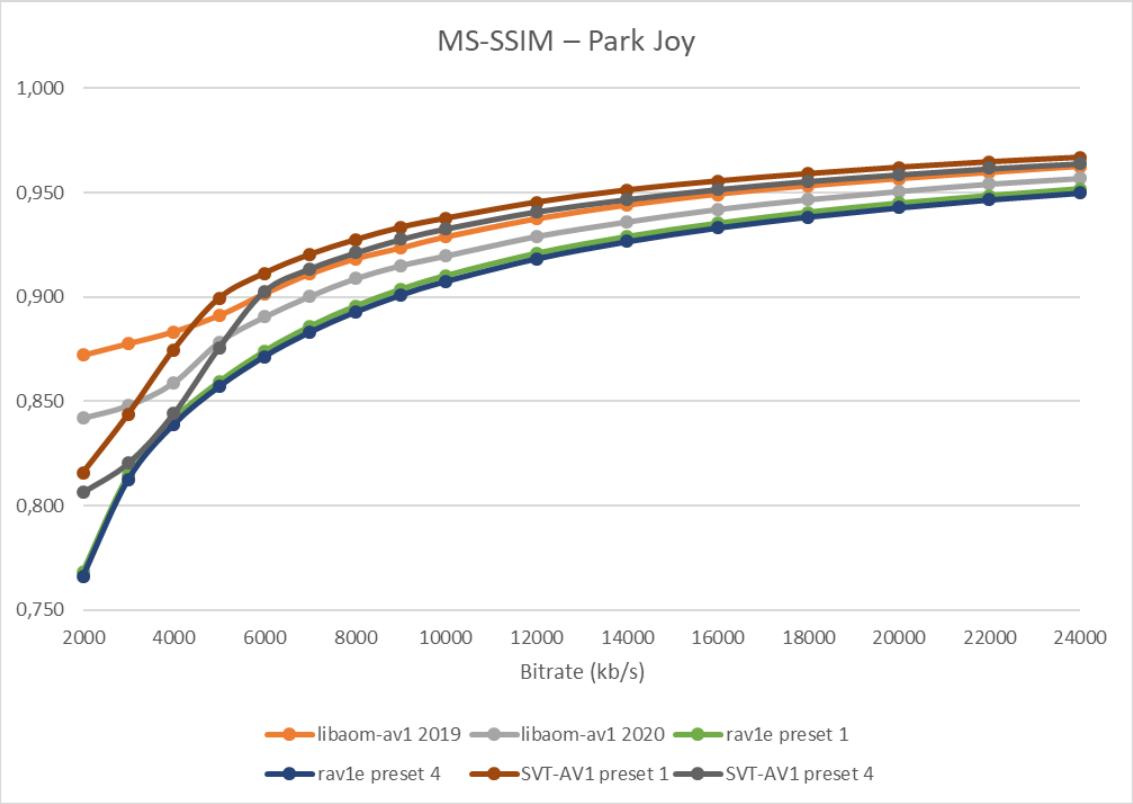


Obrázek 19 Výsledek testu VMAF sekvence Crowd Run – porovnání kodérů AV1

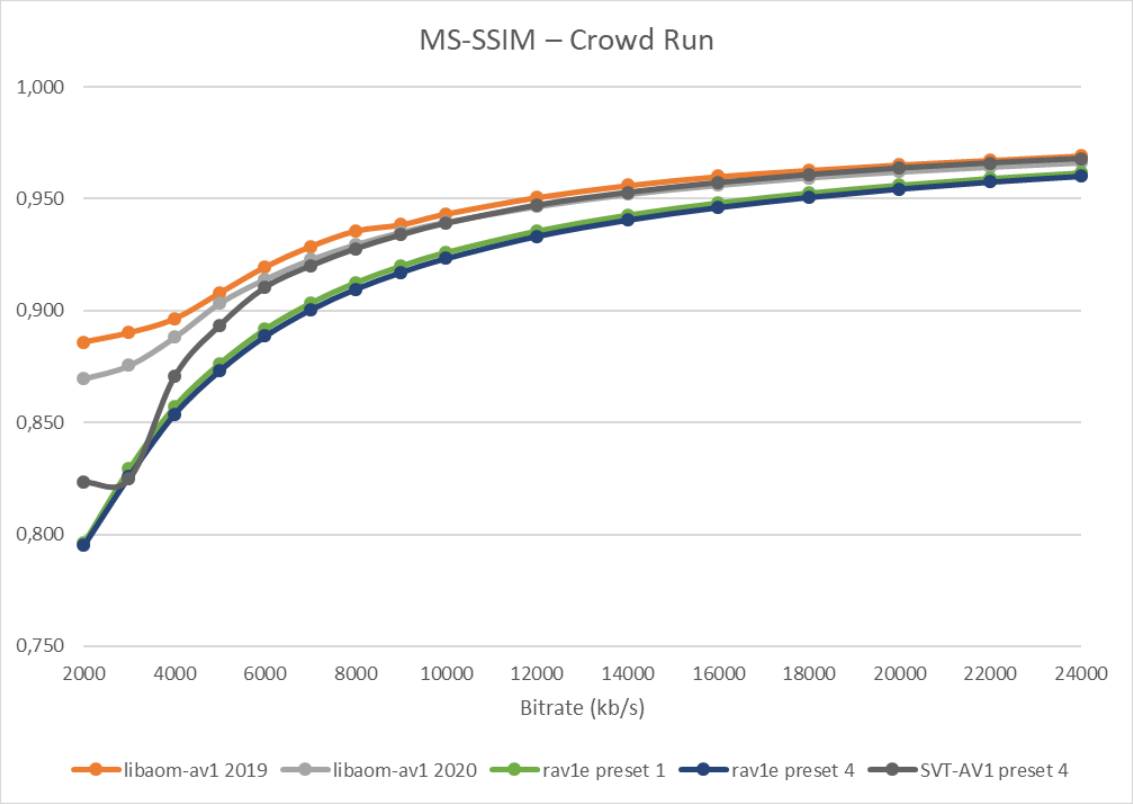


Obrázek 20 Výsledek testu VMAF sekvence Sintel – porovnání kodérů AV1

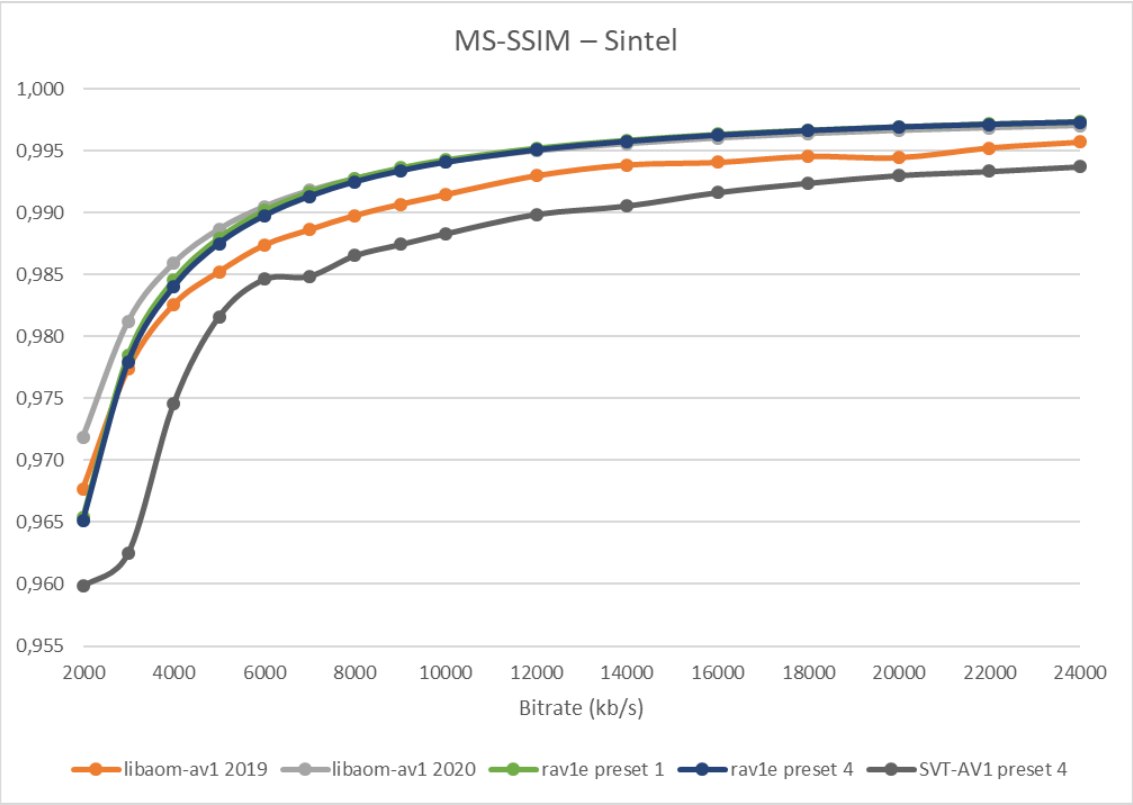
5.4.2 MS-SSIM



Obrázek 21 Výsledek testu MS-SSIM sekvence Park Joy – porovnání kodérů AV1

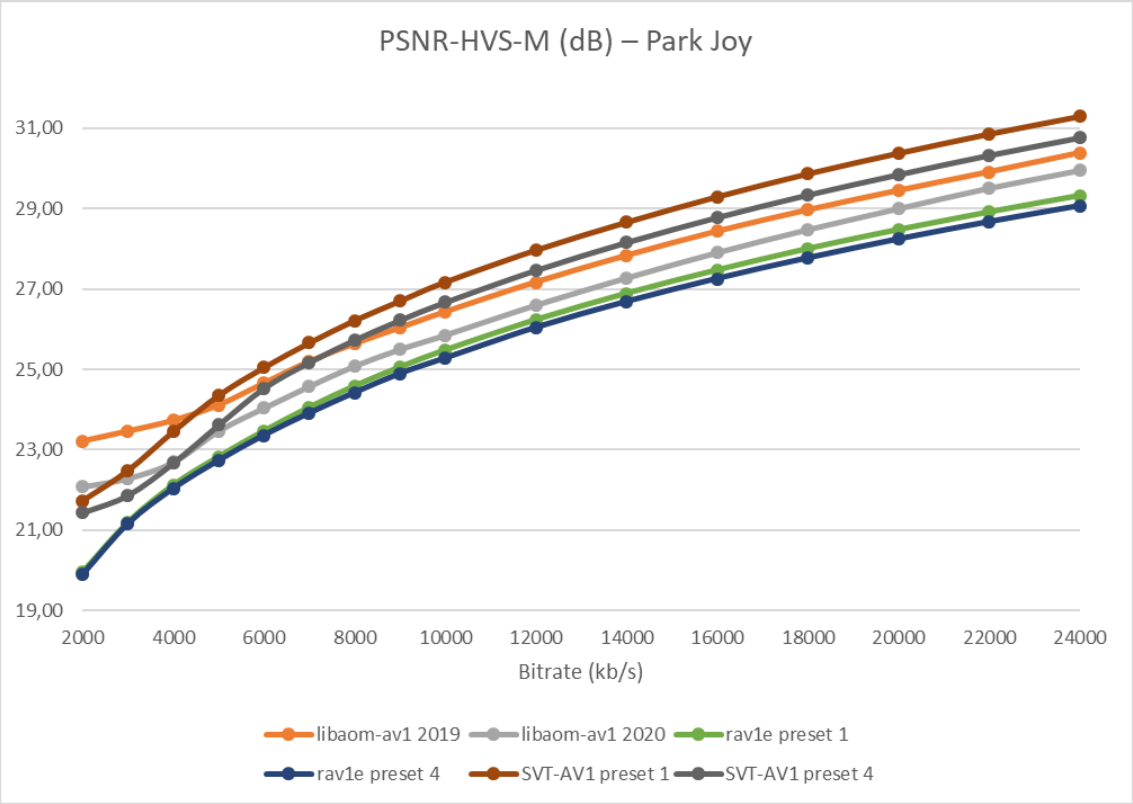


Obrázek 22 Výsledek testu MS-SSIM sekvence Crowd Run – porovnání kodérů AV1

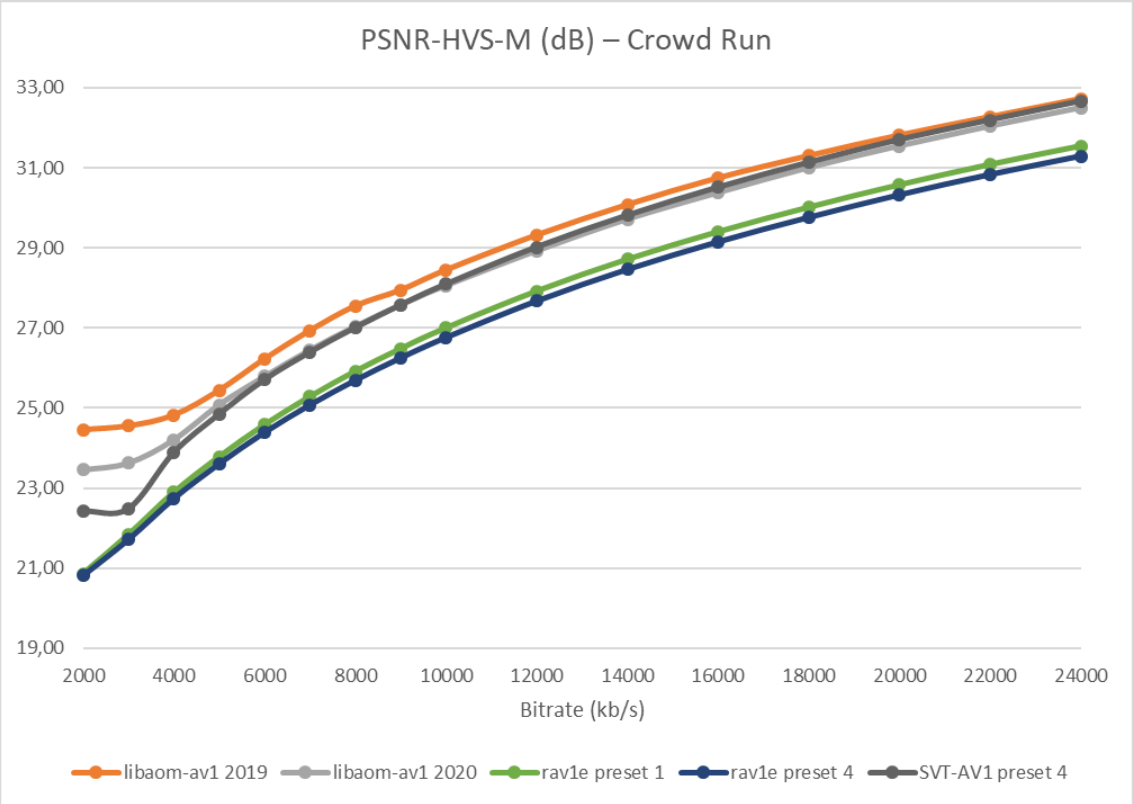


Obrázek 23 Výsledek testu MS-SSIM sekvence Sintel – porovnání kodérů AV1

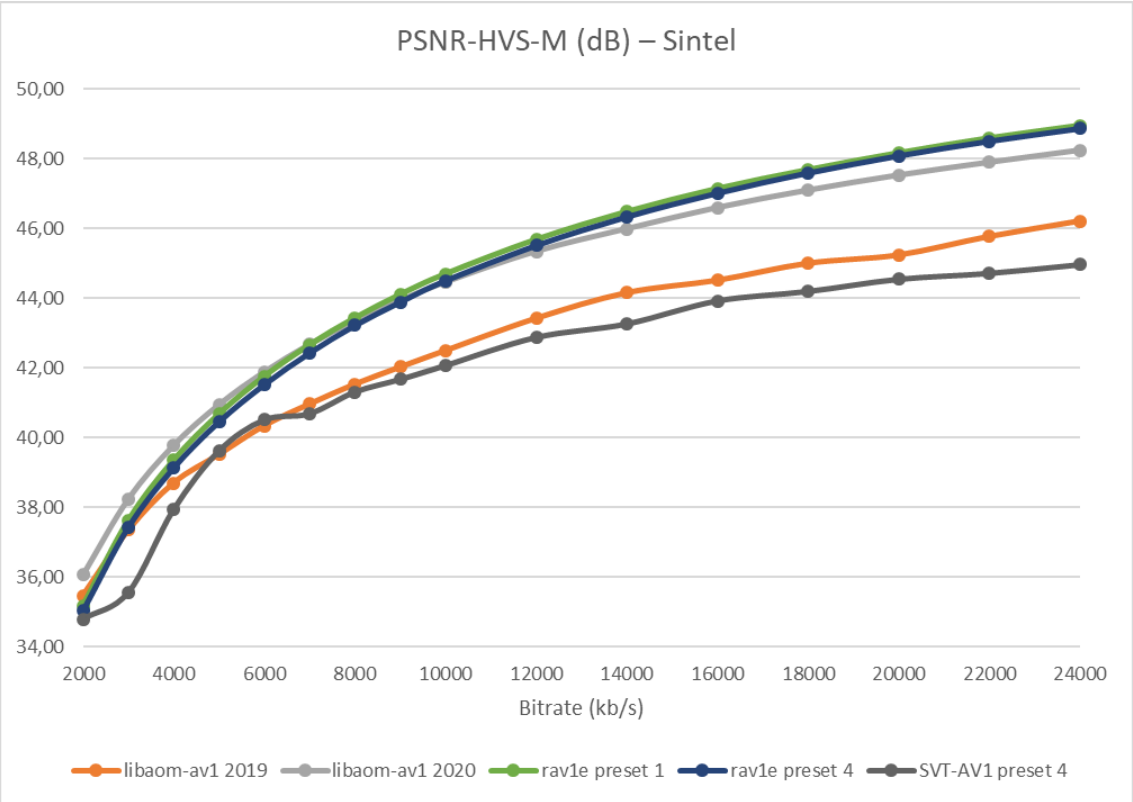
5.4.3 PSNR-HVS-M



Obrázek 24 Výsledek testu PSNR-HVS-M sekvence Park Joy – porovnání kodérů AV1



Obrázek 25 Výsledek testu PSNR-HVS-M sekvence Crowd Run – porovnání kodérů AV1



Obrázek 26 Výsledek testu PSNR-HVS-M sekvence Sintel – porovnání kodérů AV1

Při porovnání objektivní kvality jednotlivých kodérů formátu AV1 lze dospět k velice zajímavým poznatkům. Největším překvapením je to, že verze kodeku libaom z března 2019 dosahuje v případě testovacích sekvencí Park Joy a Crowd Run lepších výsledků než jeho o rok novější verze. V případě sekvence Sintel je výsledek opačný a novější verze kodeku je dle měření znatelně lepší. Podobný jev, jako u nové verze libaomu, lze pozorovat i v případě kodéru rav1e, který v případě prvních dvou zmíněných sekvencí dosahuje nejhorší kvality v porovnání s ostatními kodéry, ale v sekvenci Sintel je dle všech metod velice podobný kodeku libaom ve verzi z března 2020 a dosahuje tak společně s ním nejlepších výsledků z testovaného spektra.

Kodér SVT-AV1 docílil v sekvencích Park Joy a Crowd Run poměrně dobrých výsledků. Při nižších bitrate je sice horší než obě testované verze kodeku libaom, ale jakmile dosáhne rychlost přibližně 6 Mb/s, kvalita je dle měření podobná, jako libaom. V případě klipu Park Joy při vyšších přenosových rychlostech dokonce obě verze libaomu překonává. V případě testovacího videa Sintel má kodér SVT-AV1 dle všech tří metod nejhorší kvalitu.

Mezi rychlostním presetem 1 a 4 kodéru rav1e není ve výsledné objektivní kvalitě markantní rozdíl. Mezi presety 1 a 4 kodéru SVT-AV1 je rozdíl v kvalitě znatelnější. To je ovšem vykoupeno velice nízkou rychlostí presetu 1, která je dle měření až 14× nižší než v případě presetu 4, viz kapitola 5.5.

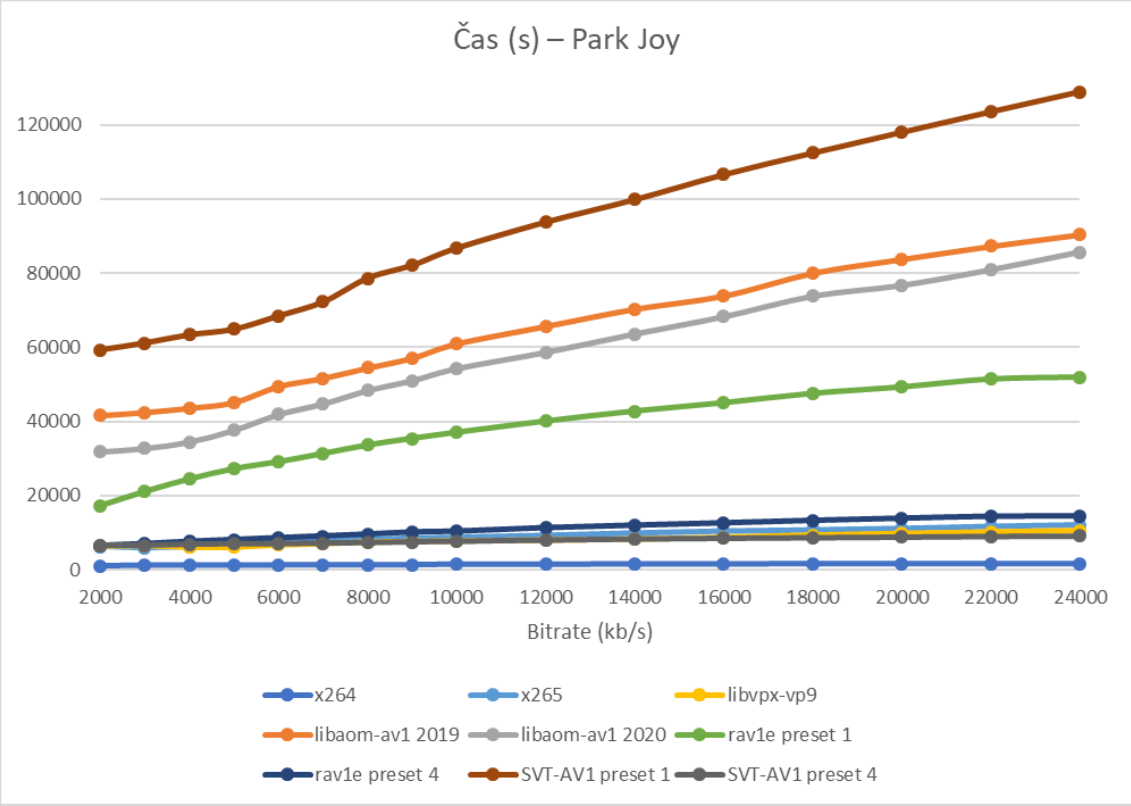
Ve zkratce lze říci, že kodéry libaom ve verzi z března 2019 a SVT-AV1 podávají nejlepší výsledky v případě reálných scén natočených videokamerou, a kodér rav1e podává nejlepší výsledky při kódování animovaného videa. Libaom ve verzi z března 2020 dosahuje v reálných

scénách o něco horších výsledků než jeho starší verze, ale zato exceluje v animovaných klipech, a celkově tak podává vyrovnaný výkon v širokém spektru různých scén.

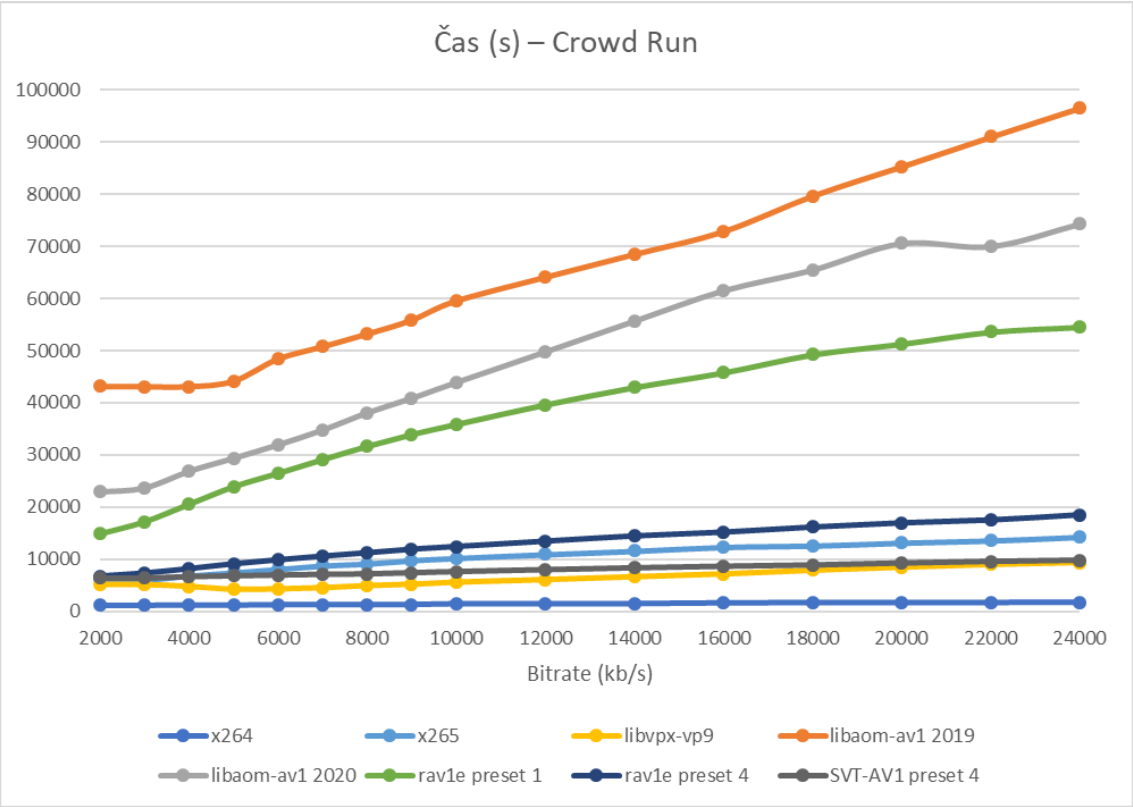
5.5 Objektivní porovnání rychlosti

Kodér libaom-av1 ve verzi z března 2019 plně nepodporuje vícevláknové kódování. Během testování se ukázalo, že v případě aktivace vícevláknové podpory pomocí přepínače `-row-mt 1` v FFmpegu se při použití čtyřjádrového procesoru se čtyřmi vlákny plně využívá pouze jedno jádro a druhé se využívá pouze nárazově. Jako nejefektivnější byla zvolena varianta, při níž se kódovala čtyři videa současně, každé pomocí jednoho jádra (pokud to bylo z důvodu využití paměti možné, více v kapitole 5.6, která rozebírá paměťovou náročnost jednotlivých kodérů). Tím se dosáhlo maximální rychlosti kódování, jelikož se využívala všechna procesorová jádra na 100 %. Aby bylo porovnání rychlosti opravdu objektivní, byly i ostatní kodéry omezeny tak, aby využívaly pouze jedno vlákno procesoru. Při reálném použití na vícejádrovém procesoru a kódování pouze jednoho videa v daný okamžik, by kodeky x264, x265, rav1e a SVT-AV1 dosahovaly rychlejších časů, jelikož dokáží využít všechna jádra na 100 %. Kodek libvpx využívaný pro kódování formátu VP9 plný potenciál čtyřjádrového procesoru nevyužívá, ale přesto je na tom mnohem lépe než libaom.

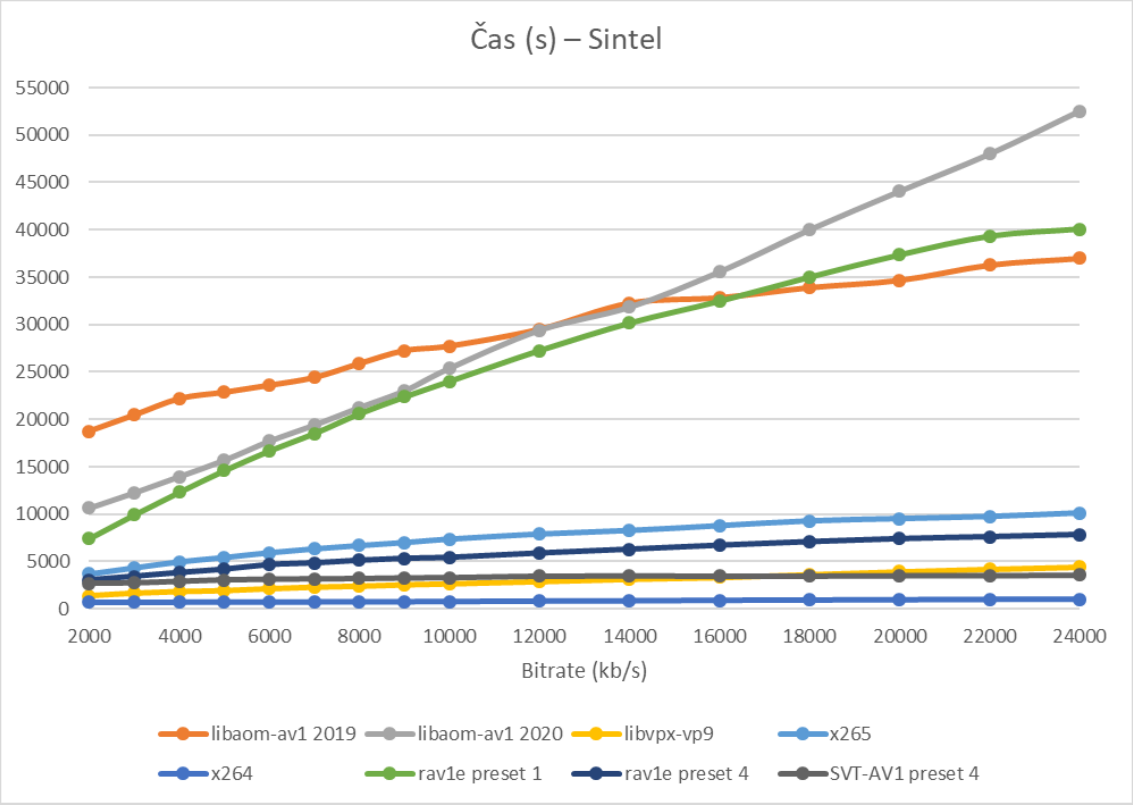
Všechny hodnoty jsou uvedeny v příloze B.



Obrázek 27 Čas potřebný pro kódování sekvence Park Joy



Obrázek 28 Čas potřebný pro kódování sekvence Crowd Run



Obrázek 29 Čas potřebný pro kódování sekvence Sintel

Z provedených testů vyplývá, že je kodér libaom ve verzi z března 2019 přibližně 3,5-8× pomalejší než x265 a přibližně 25-60× pomalejší než x264. Čas potřebný pro kódování VP9 pomocí libvpx byl mezi kodeky x264 a x265. U náročnějších sekvencí (Park Joy, Crowd Run) byl blíže x265, u sekvence z animovaného filmu Sintel byl naopak blíže x264.

Zajímavým jevem je rychlost kódování pomocí kodeku libaom z března 2020. Ten je v případě sekvencí Park Joy a Crowd Run citelně rychlejší než jeho o rok starší varianta, ale u animovaného Sintelu je situace komplikovanější. Do přenosové rychlosti 12 Mb/s je i v tomto případě rychlejší, při vyšších bitrate je oproti starší verzi pomalejší.

Kodér SVT-AV1 je v případě presetu 4 při porovnání s ostatními kodéry formátu AV1 velice rychlý – jeho rychlost je velice podobná libvpx-vp9. Kodér rav1e (preset 4) je v případě přirozených scén v klipech Crowd Run a Park Joy o trochu pomalejší než x265 (preset veryslow), v animovaném videu Sintel je situace opačná.

Rav1e je v případě použití presetu 1 dle očekávání pomalý – v případě klipu Sintel je dokonce na stejné úrovni jako libaom. Dle výsledků objektivního porovnání kvality v kapitole 5.4 není rozdíl v kvalitě mezi presetem 1 a 4 nikterak velký. Z tohoto důvodu nelze pomalý preset pro většinu použití doporučit, je přibližně 2–5× pomalejší.

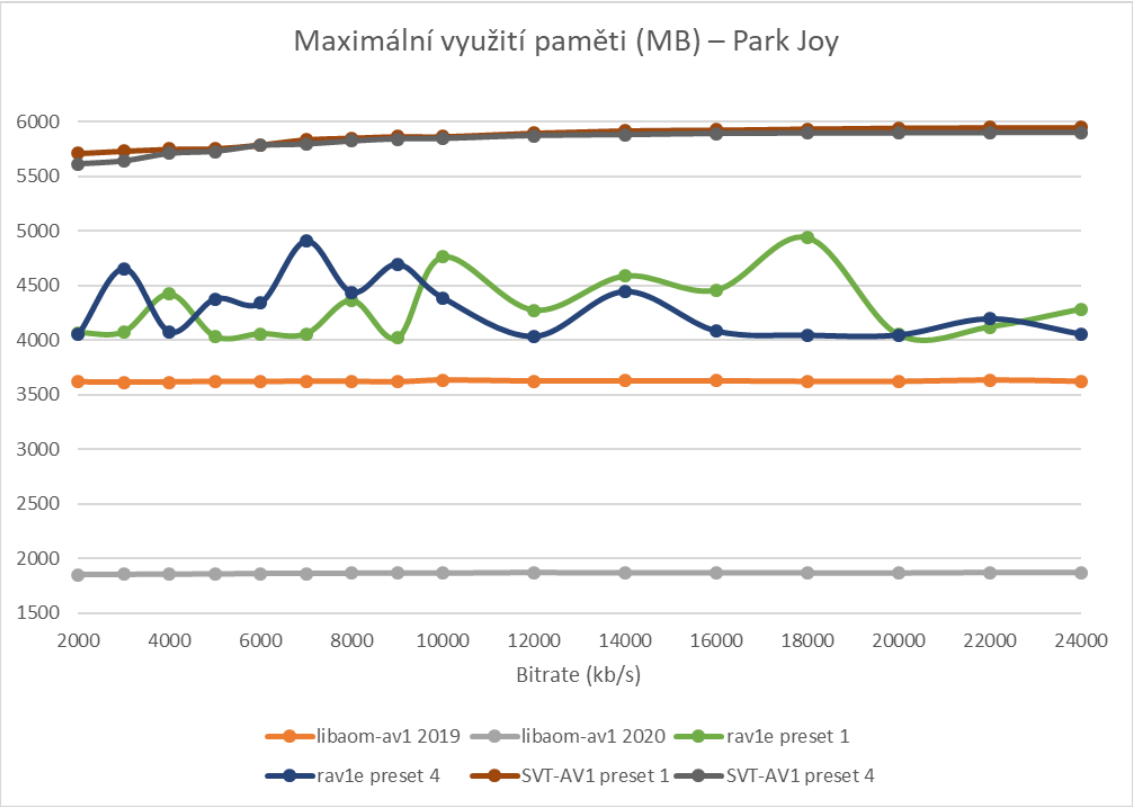
Preset 1 kodéru SVT-AV1 byl použit pouze pro sekvenci Park Joy. Oproti presetu 4 stejného kodéru je až 14× pomalejší, což z něj činí nejpomalejší kodér z celého porovnání. Pro zajímavost lze uvést, že v případě nejvyššího testovaného bitrate je 82× pomalejší než kodér x264. Z důvodu poměrně vysoké paměťové náročnosti popsané v kapitole 5.6 nebylo možné mít spuštěné více jak dva procesy kódování současně. I proto trvalo kódování všech 16 přenosových rychlostí sekvence Park Joy téměř deset dní a ke kódování sekvencí Crowd Run a Sintel nebylo přistoupeno.

Dále lze z grafů vyčíst značný růst doby potřebné pro kódování s rostoucím bitrate. Zatímco kódování sekvence Crowd Run do AV1 pomocí kodeku libaom ve verzi z března 2019 při použití bitrate 2 Mb/s trvalo 43 133 sekund, tedy přibližně 12 hodin, v případě bitrate 24 Mb/s kódování stejné sekvence trvalo 96 588 sekund, což odpovídá necelým 27 hodinám. Tento trend nevykazuje pouze libaom, ale je patrný u všech testovaných kodeků.

Sekvence Sintel se kódovala mnohem rychleji než zbylé dvě. To je samozřejmě dáno tím, že má pouze 240 snímků a ostatní 500 snímků.

5.6 Porovnání využití paměti jednotlivými AV1 kodéry

Některé kodéry potřebují k chodu více paměti než ostatní. Rozdíly mezi nimi jsou poměrně velké. V případě testovacího počítače, který má 16 GB RAM, viz Tabulka 5, využití paměti v mnoha případech určovalo, kolik paralelně běžících procesů kódování lze mít spuštěno. Princip paralelizace procesů kódování je blíže popsán v předchozí kapitole, maximální počet souběžných kódování je na testovacím hardware 4 (počet jader procesoru).



Obrázek 30 Maximální využití paměti – porovnání kodérů AV1

Z provedených testů vyplývá, že nejvíce paměti potřebuje kodér SVT-AV1, v případě sekvence Park Joy to je necelých 6 GB. Z tohoto důvodu bylo na testovacím hardware možné mít spuštěny maximálně 2 procesy kódování současně. Kodér rav1e ke kódování stejné sekvence potřeboval maximálně 5GB paměti, a bylo tak možné mít v běhu 3 procesy kódování v jeden okamžik. Mezi kodéry libaom – aomenc ve verzích z března 2019 a března 2020 je v tomto srovnání velký rozdíl. V novější verzi došlo k různým optimalizacím a nyní tento kodér vyžaduje pouze přibližně polovinu paměti než jeho o rok starší verze, a nebyl tak problém mít spuštěny 4 paralelní procesy tohoto kodéru. Rozdíly ve využití paměti mezi jednotlivými rychlostními presety SVT-AV1 a rav1e jsou zanedbatelné, nárůst využití paměti při kódování vyšších přenosových rychlostí, oproti nižším, je také minimální.

Závěr

Z provedeného měření je patrné, že komprese formátu AV1 je v současnosti objektivně nejefektivnější, a to jak při porovnání s royalty-free formáty minulých let, tak při porovnání s patentově zatíženými formáty.

Situace okolo licenčních poplatků za užití video formátu HEVC od skupiny MPEG je kvůli vzniku 3 různých patentových fondů dosti nejistá. Leonardo Chiariglione, předseda skupiny MPEG, označil současný stav za tragický, a ačkoliv navrhnul několik možných řešení, pochybuje o tom, že se tuto krizi někdy podaří zastavit. (79) Na stále velice rozšířeném formátu MPEG-4 AVC, u kterého je licenční politika volnější, je již znát vysoký věk. Tyto faktory významně nahrávají potenciálnímu úspěchu iniciativy AOMedia a jejímu bezpoplatkovému video formátu nové generace.

Širokému rozšíření formátu AV1 zatím brání zejména téměř neexistující podpora hardwarového kódování a dekódování a také zatím stále poměrně pomalé softwarové kodéry a dekodéry. Především tyto softwarové implementace však v posledním roce prošly četnými rychlostními optimalizacemi a jejich reálné nasazení je díky rychlým dekodérům, jako je dav1d a SVT-AV1, již nyní v omezené míře možné. Tento fakt dokazují například služby Netflix a YouTube, které v posledních měsících začaly nabízet vybraný obsah v AV1 široké veřejnosti. Značnou nevýhodou softwarového dekódování na zařízeních koncových uživatelů je zvýšené vytížení procesoru, což má mnoho nevýhod, v případě mobilních zařízení například znatelný dopad na výdrž baterie. Než dojde k dostatečné penetraci běžných uživatelů zařízeními s hardwarovou podporou AV1, bude to ještě několik let trvat. V tomto ohledu mají stále výhodu starší formáty.

Tato práce zahrnuje pouze zlomek aspektů, které se pojí s kódováním videa. V další práci by bylo vhodné zaměřit se například na škálovatelnost jednotlivých kodérů na více procesorových vláken. V praktické části provedené práce byly všechny kodéry omezeny pouze na jedno procesorové vlákno z důvodů popsaných v kapitole 5.5. V reálném nasazení je však pravděpodobnější scénář, kde se vláken využívá více. Další věcí, kterou tato práce nezahrnuje, je detailní úprava parametrů kodérů. Každý testovaný kodér umožňuje širokou customizaci různých parametrů. Téměř s jistotou lze říci, že úpravou správných parametrů pro konkrétní testovací sekvenci by bylo možné dosáhnout ještě o něco lepší kvality překódovaného videa. Dále je vhodné zmínit, že praktická část se zaměřovala pouze na kódování formátu AV1 a dekódování nebyla věnována pozornost. V praxi se však většina běžných uživatelů setkává spíše s potřebou video z formátu AV1 dekódovat. Z tohoto důvodu by se příští práce mohla zaměřit také na výkonnost jednotlivých softwarových dekodérů na různých architekturách procesorů. Právě rychlost softwarových dekodérů je tím, co by v následujících letech mohlo určit, jak rychlý nástup formát AV1 bude mít.

Použitá literatura

1. BARNETT, Thomas, Shruti JAIN, Usha ANDRA a Taru KHURANA. *Cisco Visual Networking Index (VNI)* [online]. prosinec 2018. Dostupné z: https://www.cisco.com/c/dam/m/en_us/network-intelligence/service-provider/digital-transformation/knowledge-network-webinars/pdfs/1213-business-services-ckn.pdf

2. NGUYEN, Tung a Detlev MARPE. Future Video Coding Technologies: A Performance Evaluation of AV1, JEM, VP9, and HM. In: *2018 Picture Coding Symposium (PCS)*: *2018 Picture Coding Symposium (PCS)* [online]. 2018, s. 31–35. ISSN 2472-7822. Dostupné z: doi:10.1109/PCS.2018.8456289

3. JVET - JOINT VIDEO EXPERTS TEAM. *Developing a video compression algorithm with capabilities beyond HEVC* [online]. [vid. 2020-05-04]. Dostupné z: https://www.itu.int/en/ITU-T/studygroups/2017-2020/16/Pages/video/jvet.aspx

4. OZER, Jan. What Is VVC? *Streaming Media Magazine* [online]. 14. listopad 2019 [vid. 2019-12-31]. Dostupné z: https://www.streamingmedia.com/Articles/ReadArticle.aspx?ArticleID=135215

5. AOMEDIA. License. *Alliance for Open Media* [online]. 4. únor 2018 [vid. 2019-12-31]. Dostupné z: https://aomedia.org/license/

6. *Theora* [online]. 2020 [vid. 2020-04-19]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Theora&oldid=940627809

7. XIPH.ORG FOUNDATION. *Theora Specification* [online]. 6 2017 [vid. 2020-02-04]. Dostupné z: https://theora.org/doc/Theora.pdf

8. STAFF, S. V. G. Dirac Algorithm Helps BBC Transport HD Olympic Coverage Via SD Circuits. *Sports Video Group* [online]. [vid. 2020-02-04]. Dostupné z: https://www.sportsvideo.org/2008/07/21/dirac-algorithm-helps-bbc-transport-hd-olympic-coverage-via-sd-circuits/

9. BBC R&D. *Dirac Specification* [online]. 23. září 2008 [vid. 2020-02-04]. Dostupné z: https://web.archive.org/web/20150503015104/http://diracvideo.org/download/specification/dirac-spec-latest.pdf

10. *Dirac (video compression format)* [online]. 2019 [vid. 2020-01-14]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Dirac_(video_compression_format)&oldid=928325070

11. BANKOSKI, James, Paul WILKINS a Yaowu XU. *VP8 Data Format and Decoding Guide* [online]. listopad 2011 [vid. 2020-02-04]. Dostupné z: https://tools.ietf.org/html/rfc6386

12. WALKER-MORGAN, DJ. *Google and MPEG LA make a deal over VP8 codec - Update - The H Open: News and Features* [online]. 8. prosinec 2013 [vid. 2020-02-11]. Dostupné z: https://web.archive.org/web/20131208040459/http://www.h-online.com/open/news/item/Google-and-MPEG-LA-make-a-deal-over-VP8-codec-Update-1818785.html

13. LARABEL, Michael. *Intel SVT-VP9 Finally Makes Its First Pre-Release For Speedy VP9 Encoding - Phoronix* [online]. 10. říjen 2019 [vid. 2020-05-04]. Dostupné z: https://www.phoronix.com/scan.php?page=news_item&px=Intel-SVT-VP9-0.1

14. *VP9* [online]. 2019 [vid. 2019-12-31]. Dostupné z: https://en.wikipedia.org/w/index.php?title=VP9&oldid=931361857

15. MUKHERJEE, Debargha, Jingning HAN, Jim BANKOSKI, Ronald BULTJE, Adrian GRANGE, John KOLESZAR, Paul WILKINS a Yaowu XU. A Technical Overview of VP9 – the Latest Open-Source Video Codec. *SMPTE Motion Imaging Journal* [online]. 2015, **124**, 44–54. Dostupné z: doi:10.5594/j18499

16. ROBERTSON, Steven. VP9: Faster, better, buffer-free YouTube videos. *YouTube Engineering and Developers Blog* [online]. 6. duben 2015 [vid. 2020-02-11]. Dostupné z: https://youtube-eng.googleblog.com/2015/04/vp9-faster-better-buffer-free-youtube.html

17. NETFLIX TECHNOLOGY. More Efficient Mobile Encodes for Netflix Downloads. *Medium* [online]. 19. duben 2017 [vid. 2020-02-11]. Dostupné z: https://netflix-techblog.com/more-efficient-mobile-encodes-for-netflix-downloads-625d7b082909

18. IETF – INTERNET ENGINEERING TASK FORCE. *IETF102-NETVC-20180719-1550* [online]. 19. červenec 2018 [vid. 2019-12-31]. Dostupné z: https://www.youtube.com/watch?v=EDX2tFTDH70

19. *Daala* [online]. 2019 [vid. 2019-12-31]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Daala&oldid=917105650

20. ROSENBERG, Jonathan. World, Meet Thor – a Project to Hammer Out a Royalty Free Video Codec. *Cisco Blogs* [online]. 11. srpen 2015 [vid. 2019-12-31]. Dostupné z: https://blogs.cisco.com/collaboration/world-meet-thor-a-project-to-hammer-out-a-royalty-free-video-codec

21. MONTGOMERY, Chris. *next generation video: Introducing Daala* [online]. 20. červen 2013 [vid. 2020-01-21]. Dostupné z: https://people.xiph.org/~xiphmont/demo/daala/demo1.shtml

22. SHANKLAND, Stephen. Tech giants join forces to hasten high-quality online video. *CNET* [online]. 1. září 2015 [vid. 2020-05-04]. Dostupné z: https://www.cnet.com/news/tech-giants-join-forces-to-hasten-high-quality-online-video/

23. AOMEDIA. A Shared Vision for the Future of Video Streaming. *Alliance for Open Media* [online]. [vid. 2019-12-31]. Dostupné z: https://aomedia.org/news/a-shared-vision-for-the-future-of-video-streaming/

24. VÁCLAVÍK, Lukáš. Google, Microsoft, Mozilla a další chtějí vytvořit videoformát budoucnosti. Bezplatný a otevřený. *Cnews.cz* [online]. 2. září 2015 [vid. 2019-04-10]. Dostupné z: https://www.cnews.cz/google-microsoft-mozilla-a-dalsi-chteji-vytvorit-video-format-budoucnosti-bezplatny-a-otevreny/

25. *Internet Engineering Task Force* [online]. 2019 [vid. 2019-12-31]. Dostupné z: https://cs.wikipedia.org/w/index.php?title=Internet_Engineering_Task_Force&oldid=17665715

26. *NETVC* [online]. 2019 [vid. 2019-12-31]. Dostupné z: https://en.wikipedia.org/w/index.php?title=NETVC&oldid=911354328

27. *Alliance for Open Media* [online]. 2019 [vid. 2019-04-10]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Alliance_for_Open_Media&oldid=891813717

28. AOMEDIA. Alliance for Open Media Established to Deliver Next-Generation Open Media Formats. *Alliance for Open Media* [online]. 28. srpen 2015 [vid. 2019-04-10]. Dostupné z: https://aomedia.org/alliance-to-deliver-next-generation-open-media-formats/

29. IETF – INTERNET ENGINEERING TASK FORCE. *IETF101-NETVC-20180319-1550* [online]. 19. březen 2018 [vid. 2019-12-31]. Dostupné z: https://www.youtube.com/watch?v=qS7hfrFWeic

30. AOMEDIA. Members. *Alliance for Open Media* [online]. 25. říjen 2019 [vid. 2019-04-10]. Dostupné z: https://aomedia.org/membership/members/

31. NETFLIX TECHNOLOGY. AVIF for Next-Generation Image Coding. *Medium* [online]. 13. únor 2020 [vid. 2020-04-11]. Dostupné z: https://netflixtechblog.com/avif-for-next-generation-image-coding-b1d75675fe4

32. *AV1* [online]. 2020 [vid. 2020-03-15]. Dostupné z: https://en.wikipedia.org/w/index.php?title=AV1&oldid=945678206

33. OZER, Jan. *Encoding options for H.264 video* [online]. 16. březen 2009 [vid. 2020-05-04]. Dostupné z: https://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html

34. RIVAZ, Peter de a Jack HAUGHTON. *AV1 Bitstream & Decoding Process Specification* [online]. 2018. Dostupné z: https://aomediacodec.github.io/av1-spec/av1-spec.pdf

35. TIMOTHY B. TERRIBERRY. *The AV1 Video Codec* [online]. 24. leden 2019 [vid. 2020-02-12]. Dostupné z: https://www.youtube.com/watch?v=qubPzBcYCTw

36. CHEN, Yue, Debargha MURHERJEE, Jingning HAN, Adrian GRANGE, Yaowu XU, Zoe LIU, Sarah PARKER, Cheng CHEN, Hui SU, Urvang JOSHI, Ching-Han CHIANG,

Yunqing WANG, Paul WILKINS, Jim BANKOSKI, Luc TRUDEAU, Nathan EGGE, Jean-Marc VALIN, Thomas DAVIES, Steinar MIDTSKOGEN, Andrey NORKIN a Peter DE RIVAZ. An Overview of Core Coding Tools in the AV1 Video Codec. In: *2018 Picture Coding Symposium (PCS)*: *2018 Picture Coding Symposium (PCS)* [online]. 2018, s. 41–45. ISSN 2472-7822. Dostupné z: doi:10.1109/PCS.2018.8456249

37. CHEN, Yue. A technial overview of the AV1. In: *@Scale 2018* [online]. B.m. 29. květen 2018 [vid. 2020-04-14]. Dostupné z: https://www.youtube.com/watch?v=04lXWMcwdXA

38. W3C. *PNG Specification: Filter Algorithms* [online]. 1. říjen 1996 [vid. 2020-04-14]. Dostupné z: https://www.w3.org/TR/PNG-Filters.html

39. MASSIMINO, Pascal. AOM - AV1 How does it work? In: *Paris Video Tech* [online]. Paříž. červenec 2017. Dostupné z: https://parisvideotech.com/wp-content/uploads/2017/07/AOM-AV1-Video-Tech-meet-up.pdf

40. COMPUTERPHILE. *JPEG DCT, Discrete Cosine Transform (JPEG Pt2)* [online]. 22. květen 2015 [vid. 2020-04-14]. Dostupné z: https://www.youtube.com/watch?v=Q2aEzeMDHMA

41. MONTGOMERY, Christopher. AV1: next generation video – The Constrained Directional Enhancement Filter – Mozilla Hacks - the Web developer blog. *Mozilla Hacks – the Web developer blog* [online]. [vid. 2020-04-14]. Dostupné z: https://hacks.mozilla.org/2018/06/av1-next-generation-video-the-constrained-directional-enhancement-filter

42. MDN CONTRIBUTORS. Media container formats (file types). *MDN Web Docs* [online]. 12. prosinec 2019 [vid. 2020-03-25]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/Media/Formats/Containers

43. LARABEL, Michael. *Rav1e 0.2 Rust AV1 Encoder Released With 40~70% Improvements Over Previous Release - Phoronix* [online]. 19. prosinec 2019 [vid. 2020-02-12]. Dostupné z: https://www.phoronix.com/scan.php?page=news_item&px=Rav1e-0.2-Released

44. LOZOVSKY, Alexey. Rust vs C++ Comparison. *Apriorit* [online]. 1. únor 2018 [vid. 2020-05-04]. Dostupné z: https://www.apriorit.com/dev-blog/520-rust-vs-c-comparison

45. NETFLIX TECHNOLOGY. SVT-AV1: an open-source AV1 encoder and decoder. *Medium* [online]. 13. březen 2020 [vid. 2020-03-15]. Dostupné z: https://netflixtechblog.com/svt-av1-an-open-source-av1-encoder-and-decoder-ad295d9b5ca2

46. LARABEL, Michael. *Google Has Been Developing „libgav1" As New AV1 Decoder - Phoronix* [online]. 5. říjen 2019 [vid. 2020-03-24]. Dostupné z: https://www.phoronix.com/scan.php?page=news_item&px=Google-libgav1-AV1-Decode

47. LARABEL, Michael. *DAV1D vs. LIBGAV1 Performance - Benchmarking Google's New AV1 Video Decoder - Phoronix* [online]. 5. říjen 2019 [vid. 2020-03-24]. Dostupné z: https://www.phoronix.com/scan.php?page=news_item&px=dav1d-libgav1-AV1-performance

48. GOOGLE. *LICENSE - codecs/libgav1 - Git at Google* [online]. 4. říjen 2019 [vid. 2020-03-24]. Dostupné z: https://chromium.googlesource.com/codecs/libgav1/+/refs/heads/master/LICENSE

49. WAGHRAY, Akshay. Microsoft brings AV1 Video Codec support to Windows 10. *Windows Latest* [online]. 11. listopad 2018 [vid. 2020-03-22]. Dostupné z: https://www.windowslatest.com/2018/11/11/free-av1-video-codec-for-windows-10-now-available-to-download-from-microsoft-store/

50. BEAUFORT, François. Audio/Video Updates in Chrome 70 | Web. *Google Developers* [online]. [vid. 2020-03-24]. Dostupné z: https://developers.google.com/web/updates/2018/09/chrome-70-media-updates?hl=cs

51. HONDA, Hirokazu. *8d7ec57bb900ee6c16a502a1de2e70fbd665261c - chromium/src/media - Git at Google* [online]. 2020 [vid. 2020-03-24]. Dostupné z: https://chromium.googlesource.com/chromium/src/media/+/8d7ec57bb900ee6c16a502a1de2e70fbd665261c

52. BURKE, Dave. Introducing Android Q Beta. *Android Developers Blog* [online]. 13. březen 2019 [vid. 2020-05-04]. Dostupné z: https://android-developers.googleblog.com/2019/03/introducing-android-q-beta.html

53. CAN I USE... *Can I use AV1* [online]. [vid. 2020-03-26]. Dostupné z: https://caniuse.com/#feat=av1

54. VIDEOLAN. *VLC Changelog* [online]. [vid. 2020-03-24]. Dostupné z: https://www.videolan.org/developers/vlc-branch/NEWS

55. FFMPEG. FFmpeg/FFmpeg. *GitHub* [online]. [vid. 2020-03-26]. Dostupné z: https://github.com/FFmpeg/FFmpeg

56. FFMPEG. *Projects – FFmpeg* [online]. 14. leden 2020 [vid. 2020-03-26]. Dostupné z: https://trac.ffmpeg.org/wiki/Projects

57. CHOUDHURY, Sagar. List of Phones with MediaTek Dimensity 1000 5G Processor to Buy in 2020. *DealNTech* [online]. 4. březen 2020 [vid. 2020-03-20]. Dostupné z: https://www.dealntech.com/mediatek-dimensity-1000-5g-processor-phones/

58. BUSINESS WIRE. *Strategy Analytics: Q2 2019 Smartphone Apps Processor Market Share: AI Chips Show Strong Growth* [online]. 14. listopad 2019 [vid. 2020-03-20]. Dostupné z: https://www.businesswire.com/news/home/20191114005625/en/Strategy-Analytics-Q2-2019-Smartphone-Apps-Processor

59. HRUSKA, Joel. Leak: Intel's Rocket Lake To Deliver New 14nm CPU Architecture, PCIe 4.0 - ExtremeTech. *ExtremeTech* [online]. 23. březen 2020 [vid. 2020-04-19]. Dostupné z: https://www.extremetech.com/computing/308049-leak-intels-rocket-lake-to-deliver-new-14nm-cpu-architecture-pcie-4-0

60. PORTER, Jon. LG unveils eight 'Real 8K' OLED and LCD TVs ahead of CES. *The Verge* [online]. 3. leden 2020 [vid. 2020-03-20]. Dostupné z: https://www.theverge.com/2020/1/3/21047876/lg-real-8k-tv-ces-2020-oled-lcd-alexa-google-assistant-homekit-airplay-2-alpha-9-gen-3-features

61. NETFLIX TECHNOLOGY. Netflix Now Streaming AV1 on Android. *Medium* [online]. 5. únor 2020 [vid. 2020-03-26]. Dostupné z: https://netflixtechblog.com/netflix-now-streaming-av1-on-android-d5264a515202

62. LI, Abner. YouTube testing AV1 codec for more efficient video streaming. *9to5Google* [online]. 14. září 2018 [vid. 2020-04-14]. Dostupné z: https://9to5google.com/2018/09/14/youtube-av1-video-beta-chrome/

63. WRIGHT, Arol. YouTube for Android TV adopts AV1 video codec in certain devices. *XDA Developers* [online]. 29. duben 2020 [vid. 2020-05-09]. Dostupné z: https://www.xda-developers.com/youtube-for-android-tv-adopts-av1-video-codec-in-certain-devices/

64. *Image quality* [online]. 2020 [vid. 2020-04-11]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Image_quality&oldid=943633751

65. NETFLIX TECHNOLOGY. Toward A Practical Perceptual Video Quality Metric. *Medium* [online]. 19. duben 2017 [vid. 2020-04-08]. Dostupné z: https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652

66. WANG, Z., E.P. SIMONCELLI a A.C. BOVIK. Multiscale structural similarity for image quality assessment. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*: *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003* [online]. 2003, s. 1398-1402 Vol.2. Dostupné z: doi:10.1109/ACSSC.2003.1292216

67. TONG, Yubing, Hubert KONIK, Faouzi ALAYA CHEIKH a Alain TREMÉAU. Image Quality Assessment Based on Saliency Map Analysis. *Journal of Imaging Science and Technology* [online]. 2010, **54**. Dostupné z: doi:10.2352/J.ImagingSci.Technol.2010.54.3.030503

68. *Peak signal-to-noise ratio* [online]. 2020 [vid. 2020-04-07]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Peak_signal-to-noise_ratio&oldid=945287143

69. PONOMARENKO, Nikolay, Flavia SILVESTRI, Karen EGIAZARIAN, Marco CARLI, J. ASTOLA a Vladimir LUKIN. On between-coefficient contrast masking of DCT basis functions. *Proc of the 3rd Int Workshop on Video Processing and Quality Metrics for Consumer Electronics*. 2007.

70. SHEIKH, H.R. a A.C. BOVIK. Image information and visual quality. *IEEE Transactions on Image Processing* [online]. 2006, **15**(2), 430–444. ISSN 1941-0042. Dostupné z: doi:10.1109/TIP.2005.859378

71. KOTERA, Jan. Image Quality Assessment. In: *ZOI Friday Seminar* [online]. Ústav teorie informace a automatizace AV ČR. 25. duben 2014. Dostupné z: http://zoi.utia.cas.cz/files/seminar_image_quality.pdf

72. NETFLIX TECHNOLOGY. Dynamic optimizer — a perceptual video encoding optimization framework. *Medium* [online]. 5. březen 2018 [vid. 2020-04-08]. Dostupné z: https://netflixtechblog.com/dynamic-optimizer-a-perceptual-video-encoding-optimization-framework-e19f1e3a277f

73. ENCODING.COM. *2019 Global Media Formats Report* [online]. [vid. 2019-12-31]. Dostupné z: https://www.encoding.com/resources/

74. *H.264/MPEG-4 AVC* [online]. 2019 [vid. 2019-04-03]. Dostupné z: https://en.wikipedia.org/w/index.php?title=H.264/MPEG-4_AVC&oldid=890646327

75. XIPH.ORG FOUNDATION. *Xiph.org :: Derf's Test Media Collection* [online]. [vid. 2019-03-20]. Dostupné z: https://media.xiph.org/video/derf/

76. FFMPEG. *Encode/H.264 – FFmpeg* [online]. [vid. 2020-03-10]. Dostupné z: https://trac.ffmpeg.org/wiki/Encode/H.264#FAQ

77. HAHLING, Robin. *VQMT: Video Quality Measurement Tool. Fast implementations of the following objective image quality metrics: PSNR, SSIM, MS-SSIM, VIFp, PSNR-HVS and PSNR-HVS-M.: Rolinh/VQMT* [online]. C++. 2019 [vid. 2019-04-10]. Dostupné z: https://github.com/Rolinh/VQMT

78. NETFLIX, INC. *Perceptual video quality assessment based on multi-method fusion.: Netflix/vmaf* [online]. C++. B.m.: Netflix, Inc., 2019 [vid. 2019-04-10]. Dostupné z: https://github.com/Netflix/vmaf

79. CHIARIGLIONE, Leonardo. *A crisis, the causes and a solution* [online]. [vid. 2020-03-31]. Dostupné z: https://blog.chiariglione.org/a-crisis-the-causes-and-a-solution/

Přílohy

Příloha A: Objektivní porovnání kvality

Hodnoty naměřené pomocí jednotlivých metod popsaných v kapitole 4.

VMAF

Tabulka 7 Naměřené hodnoty pomocí metody VMAF sekvence Park Joy

Bitrate (kb/s)	libaom-av1 2019	libaom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 1	SVT-AV1 preset 4
2000	46,05	38,83	31,88	25,75	9,78	27,51	27,33	34,64	32,29
3000	47,45	39,79	33,55	36,93	16,53	33,72	33,74	39,88	35,17
4000	48,98	42,41	41,12	43,66	22,97	38,64	38,38	46,78	40,64
5000	51,33	47,33	45,79	48,59	28,60	42,46	42,23	53,12	47,43
6000	54,61	50,88	49,10	52,55	33,39	46,07	45,70	57,44	53,94
7000	57,72	54,20	52,30	56,08	38,07	49,39	48,81	61,10	57,94
8000	60,32	57,11	55,38	59,03	42,22	52,38	51,66	64,28	61,18
9000	62,62	59,48	58,02	61,62	45,73	55,03	54,27	66,96	64,02
10000	64,76	61,50	60,48	63,74	48,94	57,43	56,50	69,43	66,40
12000	68,59	65,57	64,59	67,54	54,35	61,59	60,72	73,40	70,52
14000	71,83	69,02	67,83	70,66	58,90	65,12	64,18	76,64	73,88
16000	74,62	72,04	70,77	73,42	62,72	68,05	67,15	79,40	76,69
18000	76,87	74,70	73,44	75,73	65,96	70,64	69,67	81,76	79,07
20000	78,89	77,05	75,82	77,80	68,84	72,89	71,92	83,71	81,16
22000	80,72	79,18	77,86	79,67	71,35	74,88	73,92	85,48	83,03
24000	82,57	80,99	79,60	81,30	73,61	76,67	75,65	87,00	84,64

Tabulka 8 Naměřené hodnoty pomocí metody VMAF sekvence Crowd Run

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	44,73	37,91	32,79	24,65	9,00	23,88	23,66	29,14
3000	45,32	39,12	34,54	30,30	13,97	29,04	28,47	29,56
4000	47,02	42,98	40,73	40,96	19,39	34,98	34,22	39,86
5000	50,97	48,59	46,67	46,92	24,54	40,12	39,23	46,67
6000	55,69	53,00	50,30	51,87	29,96	44,86	43,83	52,46
7000	59,84	57,05	54,11	55,95	35,20	49,06	47,98	56,81
8000	63,37	60,52	57,48	59,44	40,00	52,89	51,68	60,55
9000	65,74	63,63	60,32	62,45	44,38	56,30	55,08	63,89
10000	68,41	66,36	63,02	65,09	48,31	59,36	58,13	66,86
12000	72,83	71,02	67,60	69,46	54,88	64,64	63,41	71,86
14000	76,44	74,99	71,56	73,10	60,32	68,90	67,67	75,82
16000	79,40	78,21	74,80	76,18	64,86	72,37	71,18	79,08
18000	81,64	80,92	77,51	78,76	68,70	75,30	74,14	81,75
20000	83,69	83,22	79,77	80,96	71,97	77,88	76,75	83,98
22000	85,43	85,12	81,58	82,88	74,77	80,10	79,00	85,83
24000	87,11	86,90	83,32	84,55	77,20	81,99	80,94	87,50

Tabulka 9 Naměřené hodnoty pomocí metody VMAF sekvence Sintel

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	83,18	85,92	83,04	80,15	42,38	81,43	80,71	77,00
3000	90,61	93,54	91,73	90,17	58,55	90,60	90,06	79,06
4000	94,66	96,91	95,58	95,17	70,66	94,74	94,32	88,15
5000	96,14	98,45	97,64	97,77	79,05	96,92	96,62	92,83
6000	97,41	99,15	98,61	99,10	85,18	98,19	97,93	94,87
7000	97,94	99,40	99,26	99,54	89,66	98,91	98,74	94,99
8000	98,41	99,51	99,57	99,71	92,76	99,42	99,28	96,06
9000	98,67	99,60	99,74	99,78	94,87	99,66	99,58	96,60
10000	98,80	99,68	99,81	99,81	96,40	99,77	99,74	97,09
12000	99,13	99,78	99,87	99,84	98,30	99,83	99,83	97,66
14000	99,17	99,83	99,89	99,86	99,27	99,85	99,85	97,95
16000	99,23	99,84	99,90	99,88	99,64	99,86	99,86	98,38
18000	99,32	99,84	99,91	99,89	99,80	99,86	99,86	98,79
20000	99,51	99,84	99,91	99,90	99,87	99,87	99,87	99,05
22000	99,53	99,84	99,92	99,90	99,89	99,87	99,87	99,17
24000	99,69	99,84	99,92	99,91	99,90	99,88	99,88	99,27

MS-SSIM

Tabulka 10 Naměřené hodnoty pomocí metody MS-SSIM sekvence Park Joy

Bi-trate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 1	SVT-AV1 preset 4
2000	0,872	0,842	0,801	0,756	0,493	0,768	0,766	0,816	0,806
3000	0,878	0,848	0,810	0,825	0,615	0,814	0,813	0,844	0,820
4000	0,883	0,859	0,849	0,856	0,700	0,841	0,839	0,875	0,844
5000	0,891	0,878	0,871	0,874	0,760	0,859	0,857	0,899	0,876
6000	0,902	0,890	0,883	0,887	0,798	0,874	0,871	0,911	0,903
7000	0,911	0,900	0,894	0,897	0,827	0,886	0,883	0,920	0,914
8000	0,918	0,909	0,904	0,905	0,848	0,895	0,893	0,927	0,921
9000	0,923	0,915	0,912	0,912	0,864	0,903	0,901	0,933	0,928
10000	0,929	0,920	0,919	0,918	0,877	0,910	0,907	0,938	0,933
12000	0,937	0,929	0,930	0,927	0,896	0,921	0,918	0,945	0,941
14000	0,944	0,936	0,937	0,934	0,909	0,929	0,927	0,951	0,947
16000	0,949	0,942	0,943	0,940	0,920	0,935	0,933	0,955	0,952
18000	0,953	0,947	0,947	0,945	0,927	0,941	0,938	0,959	0,956
20000	0,957	0,951	0,951	0,949	0,934	0,945	0,943	0,962	0,959
22000	0,960	0,954	0,955	0,953	0,939	0,949	0,947	0,965	0,962
24000	0,962	0,957	0,957	0,956	0,943	0,952	0,950	0,967	0,964

Tabulka 11 Naměřené hodnoty pomocí metody MS-SSIM sekvence Crowd Run

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	0,886	0,870	0,833	0,802	0,547	0,796	0,795	0,823
3000	0,890	0,876	0,839	0,824	0,651	0,829	0,826	0,825
4000	0,896	0,888	0,866	0,868	0,717	0,857	0,854	0,871
5000	0,908	0,903	0,889	0,888	0,773	0,877	0,873	0,893
6000	0,920	0,914	0,901	0,901	0,812	0,892	0,889	0,910
7000	0,929	0,923	0,913	0,911	0,840	0,903	0,900	0,920
8000	0,936	0,930	0,921	0,918	0,861	0,913	0,910	0,928
9000	0,939	0,935	0,928	0,925	0,877	0,920	0,917	0,934
10000	0,943	0,940	0,934	0,930	0,890	0,926	0,923	0,939
12000	0,951	0,947	0,942	0,938	0,907	0,936	0,933	0,947
14000	0,956	0,952	0,949	0,945	0,920	0,943	0,941	0,953
16000	0,960	0,956	0,953	0,950	0,929	0,948	0,946	0,957
18000	0,963	0,960	0,957	0,954	0,937	0,953	0,951	0,961
20000	0,965	0,962	0,960	0,957	0,942	0,956	0,954	0,964
22000	0,967	0,964	0,963	0,960	0,947	0,959	0,958	0,966
24000	0,969	0,966	0,965	0,962	0,951	0,962	0,960	0,968

Tabulka 12 Naměřené hodnoty pomocí metody MS-SSIM sekvence Sintel

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	0,968	0,972	0,965	0,957	0,838	0,965	0,965	0,960
3000	0,977	0,981	0,977	0,972	0,919	0,979	0,978	0,963
4000	0,983	0,986	0,983	0,980	0,949	0,985	0,984	0,975
5000	0,985	0,989	0,987	0,984	0,963	0,988	0,988	0,982
6000	0,987	0,990	0,989	0,988	0,971	0,990	0,990	0,985
7000	0,989	0,992	0,991	0,990	0,977	0,992	0,991	0,985
8000	0,990	0,993	0,992	0,991	0,981	0,993	0,992	0,987
9000	0,991	0,994	0,993	0,992	0,983	0,994	0,993	0,987
10000	0,991	0,994	0,993	0,993	0,985	0,994	0,994	0,988
12000	0,993	0,995	0,995	0,994	0,988	0,995	0,995	0,990
14000	0,994	0,996	0,995	0,995	0,990	0,996	0,996	0,991
16000	0,994	0,996	0,996	0,996	0,992	0,996	0,996	0,992
18000	0,995	0,996	0,996	0,996	0,993	0,997	0,997	0,992
20000	0,994	0,997	0,997	0,996	0,993	0,997	0,997	0,993
22000	0,995	0,997	0,997	0,997	0,994	0,997	0,997	0,993
24000	0,996	0,997	0,997	0,997	0,995	0,997	0,997	0,994

PSNR-HVS-M

Tabulka 13 Naměřené hodnoty pomocí metody PSNR-HVS-M sekvence Park Joy (dB)

Bi-trate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 1	SVT-AV1 preset 4
2000	23,21	22,08	21,14	19,95	16,29	19,96	19,90	21,74	21,44
3000	23,46	22,29	21,39	21,58	17,70	21,20	21,16	22,49	21,88
4000	23,73	22,69	22,52	22,58	18,85	22,12	22,04	23,46	22,68
5000	24,11	23,47	23,20	23,31	19,78	22,83	22,74	24,35	23,62
6000	24,66	24,04	23,74	23,91	20,57	23,47	23,36	25,05	24,54
7000	25,20	24,58	24,27	24,45	21,31	24,05	23,91	25,66	25,18
8000	25,64	25,09	24,79	24,92	21,96	24,58	24,42	26,22	25,74
9000	26,04	25,50	25,24	25,36	22,53	25,05	24,89	26,70	26,23
10000	26,43	25,86	25,68	25,74	23,05	25,48	25,28	27,17	26,67
12000	27,17	26,60	26,44	26,45	23,94	26,23	26,04	27,97	27,47
14000	27,83	27,28	27,08	27,08	24,72	26,89	26,68	28,66	28,16
16000	28,44	27,91	27,66	27,66	25,39	27,47	27,26	29,29	28,79
18000	28,97	28,48	28,21	28,17	25,99	28,00	27,77	29,87	29,35
20000	29,45	29,00	28,70	28,65	26,55	28,47	28,24	30,38	29,85
22000	29,91	29,51	29,16	29,11	27,05	28,91	28,67	30,86	30,33
24000	30,39	29,95	29,59	29,53	27,52	29,32	29,07	31,29	30,77

Tabulka 14 Naměřené hodnoty pomocí metody PSNR-HVS-M sekvence Crowd Run (dB)

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	24,46	23,45	22,60	21,32	17,17	20,87	20,82	22,44
3000	24,55	23,63	22,86	22,06	18,49	21,84	21,73	22,50
4000	24,82	24,21	23,78	23,54	19,60	22,90	22,75	23,89
5000	25,44	25,08	24,64	24,46	20,62	23,79	23,61	24,85
6000	26,22	25,79	25,16	25,19	21,55	24,58	24,40	25,71
7000	26,93	26,45	25,77	25,82	22,39	25,29	25,08	26,40
8000	27,54	27,03	26,39	26,37	23,14	25,92	25,70	27,01
9000	27,94	27,57	26,91	26,88	23,81	26,48	26,25	27,58
10000	28,44	28,06	27,42	27,34	24,43	27,00	26,76	28,09
12000	29,31	28,92	28,29	28,16	25,48	27,92	27,67	29,02
14000	30,07	29,72	29,05	28,88	26,37	28,71	28,46	29,81
16000	30,74	30,38	29,72	29,52	27,15	29,40	29,15	30,51
18000	31,30	31,00	30,32	30,10	27,85	30,02	29,76	31,13
20000	31,81	31,54	30,86	30,63	28,47	30,58	30,32	31,70
22000	32,27	32,04	31,34	31,11	29,03	31,08	30,83	32,19
24000	32,72	32,50	31,77	31,56	29,54	31,55	31,29	32,66

Tabulka 15 Naměřené hodnoty pomocí metody PSNR-HVS-M sekvence Sintel (dB)

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	35,46	36,06	34,81	33,62	26,62	35,17	35,02	34,78
3000	37,34	38,24	36,96	35,65	29,94	37,62	37,42	35,55
4000	38,69	39,77	38,61	37,28	32,11	39,35	39,12	37,92
5000	39,51	40,94	39,81	38,59	33,66	40,67	40,45	39,61
6000	40,34	41,88	40,81	39,72	34,89	41,75	41,51	40,50
7000	40,96	42,67	41,67	40,64	35,94	42,66	42,42	40,68
8000	41,53	43,34	42,42	41,44	36,89	43,43	43,21	41,30
9000	42,02	43,93	43,11	42,14	37,72	44,10	43,88	41,67
10000	42,49	44,46	43,68	42,77	38,47	44,69	44,50	42,06
12000	43,42	45,34	44,68	43,85	39,73	45,68	45,51	42,87
14000	44,15	45,99	45,51	44,73	40,77	46,48	46,33	43,26
16000	44,51	46,59	46,20	45,46	41,61	47,14	47,01	43,92
18000	45,00	47,09	46,80	46,11	42,34	47,68	47,59	44,19
20000	45,23	47,52	47,33	46,68	43,00	48,16	48,08	44,54
22000	45,77	47,89	47,79	47,18	43,58	48,59	48,50	44,71
24000	46,21	48,24	48,20	47,65	44,13	48,95	48,86	44,96

Příloha B: Objektivní porovnání rychlosti

Doba potřebná pro kódování jednotlivých sekvencí.

Tabulka 16 Čas v sekundách potřebný pro kódování sekvence Park Joy

Bitrate (kb/s)	libaom-av1 2019	libaom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 1	SVT-AV1 preset 4
2000	41616	31797	6369	6276	1006	17191	6475	59278	6479
3000	42458	32724	6554	5880	1179	21023	6988	61151	6537
4000	43586	34482	6066	6264	1180	24406	7572	63344	6786
5000	45137	37614	6087	6721	1197	27171	8065	64851	6950
6000	49451	41865	6718	7442	1235	29062	8556	68448	6986
7000	51646	44744	7018	7862	1239	31299	9022	72213	7133
8000	54442	48319	7481	8164	1265	33588	9531	78634	7259
9000	57088	51007	7563	8470	1293	35369	10085	82107	7476
10000	60958	54250	7754	8714	1409	37020	10321	86721	7652
12000	65689	58642	7998	9236	1416	40049	11238	93747	7978
14000	70324	63594	8252	9888	1454	42761	11907	99814	8270
16000	73883	68249	8713	10397	1495	45019	12542	106555	8474
18000	80014	73787	9343	10725	1531	47547	13219	112406	8654
20000	83750	76696	9779	11129	1534	49281	13781	118000	8812
22000	87275	80906	10158	11694	1542	51435	14297	123512	8980
24000	90401	85682	10548	12110	1572	51918	14431	128917	9082

Tabulka 17 Čas v sekundách potřebný pro kódování sekvence Crowd Run

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	43133	22915	5091	6303	1111	14876	6695	6388
3000	43095	23656	5087	6075	1153	17188	7282	6376
4000	43096	26860	4741	6826	1203	20547	8128	6647
5000	44129	29369	4174	7424	1186	23926	9067	6806
6000	48391	31953	4244	8078	1264	26499	9824	6956
7000	50771	34783	4480	8755	1268	29126	10539	7072
8000	53199	38100	4894	9126	1281	31669	11201	7164
9000	55869	40901	5157	9764	1313	33929	11873	7388
10000	59554	43927	5537	10204	1422	35878	12404	7599
12000	64114	49791	6042	10920	1431	39631	13422	7968
14000	68507	55712	6599	11562	1465	42965	14433	8314
16000	72881	61534	7152	12327	1582	45788	15165	8595
18000	79699	65502	7854	12574	1639	49240	16142	8839
20000	85336	70607	8452	13132	1649	51285	16948	9211
22000	91046	70024	8988	13559	1647	53582	17503	9525
24000	96588	74347	9365	14260	1705	54535	18501	9740

Tabulka 18 Čas v sekundách potřebný pro kódování sekvence Sintel

Bitrate (kb/s)	li-baom-av1 2019	li-baom-av1 2020	libvpx-vp9	x265	x264	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 4
2000	18748	10640	1398	3678	686	7435	3023	2671
3000	20454	12253	1677	4328	704	9936	3417	2723
4000	22190	13963	1842	4957	718	12319	3816	2895
5000	22856	15690	1939	5418	726	14606	4167	3069
6000	23597	17702	2140	5919	733	16680	4658	3139
7000	24421	19398	2307	6330	746	18517	4836	3188
8000	25897	21246	2410	6673	764	20604	5115	3207
9000	27265	23012	2554	6989	765	22368	5325	3288
10000	27748	25387	2662	7345	780	24012	5431	3315
12000	29519	29403	2894	7882	826	27254	5886	3481
14000	32275	31836	3140	8259	852	30149	6294	3517
16000	32846	35556	3328	8768	901	32456	6716	3487
18000	33914	39995	3625	9266	928	34988	7100	3490
20000	34685	44051	3929	9504	956	37334	7417	3532
22000	36288	47994	4192	9734	989	39301	7626	3541
24000	36993	52512	4427	10102	1013	40050	7856	3595

Příloha C: Porovnání využití paměti kodéry formátu AV1

Tabulka 19 Maximální využití paměti – sekvence Park Joy (MB)

Bitrate (kb/s)	libaom-av1 2019	libaom-av1 2020	rav1e preset 1	rav1e preset 4	SVT-AV1 preset 1	SVT-AV1 preset 4
2000	3616	1851	4066	4048	5713	5613
3000	3611	1853	4076	4654	5735	5640
4000	3614	1856	4421	4072	5751	5708
5000	3620	1857	4032	4375	5756	5727
6000	3620	1861	4057	4338	5790	5780
7000	3621	1862	4056	4908	5837	5793
8000	3620	1864	4363	4436	5852	5821
9000	3616	1864	4017	4694	5866	5841
10000	3633	1864	4760	4384	5865	5844
12000	3624	1869	4273	4031	5897	5871
14000	3625	1866	4585	4444	5918	5879
16000	3626	1867	4457	4081	5924	5890
18000	3620	1866	4937	4042	5934	5893
20000	3619	1866	4048	4043	5943	5895
22000	3634	1870	4119	4193	5946	5897
24000	3620	1871	4282	4053	5948	5899