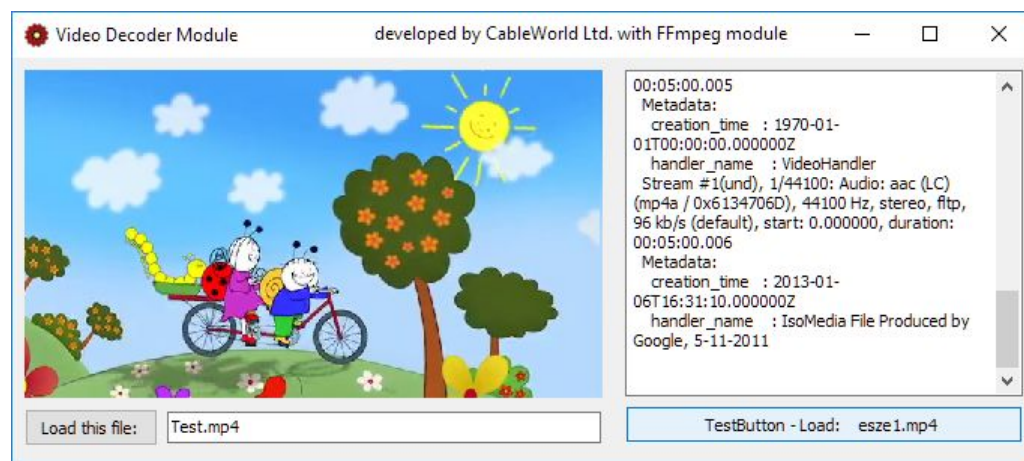


A tömörített video- és hangadatfolyamok mélyebb szintű elemzése számunkra is új terület, de kísérletet teszünk arra, hogy közérthető módon összefoglaljuk, amit a szabványokból megtudtunk.



A tartalomról:

- HDTV vs. UHDTV
A kép minősége nem csak a képpontok számától függ
- A video- és audioadatfolyamok felépítése
Áttérünk a bites szemléletre
- TS packet – Tömörített adatfolyam – Konténer - mp4
Út az mp4 felé
- A tömörített videoadatfolyamok dekódolása
Az FFmpeg fejlesztői környezet bemutatása
- A hangerő mérése és beállítása objektív módszerekkel
Nem csak nálunk probléma a műsorok eltérő hangossága
- Hanggenerátor a PST-ben
Előzetes fejlesztőink asztaláról

CableWorld

hírek

A CableWorld Kft. technikai magazinja
2017. október

Számunk fő témája:



A video- és audioadatfolyamok

66.

HDTV vs. UHDTV

Érdemes váltani?

A 4K és az UHD kifejezések más és más jelentéstartalommal rendelkeznek attól függően, hogy milyen környezetben használjuk őket, illetve magukban foglalnak számos olyan új technológiát és eljárást, amelyek túlmutatnak a standard Full HD videoformátum lehetőségein.

A 4K kifejezés a HD formátumhoz képest négyszeres pixelszámmra, azaz kétszeres felbontásra utal. Ez a moziban 4096×2160, a televíziótechnikában pedig 3840×2160 képpontot jelent. Az UHD (Ultra-High-Definition) egy tágabb fogalom, ami a felhasználói élmény fokozásának érdekében egyéb eljárásokat is magában foglal, úgy mint

- HDR (High Dynamic Range), azaz fényesebb fehérek és sötétebb feketék,
- WCG (Wide Color Gamut), többféle színárnyalat a valós élmény visszaadásához, és
- HFR (High Frame Rate), több képkocka másodpercenként az elmosódások ellen.

A fotósok körében régóta ismert probléma, hogy különböző megvilágítású témák fényképezésekor a képen túl világos vagy teljesen sötét területek jelennek meg, mert a kamerák szenzorai képtelenek visszaadni azt a kontrasztarányt, amit a valóságban látunk.

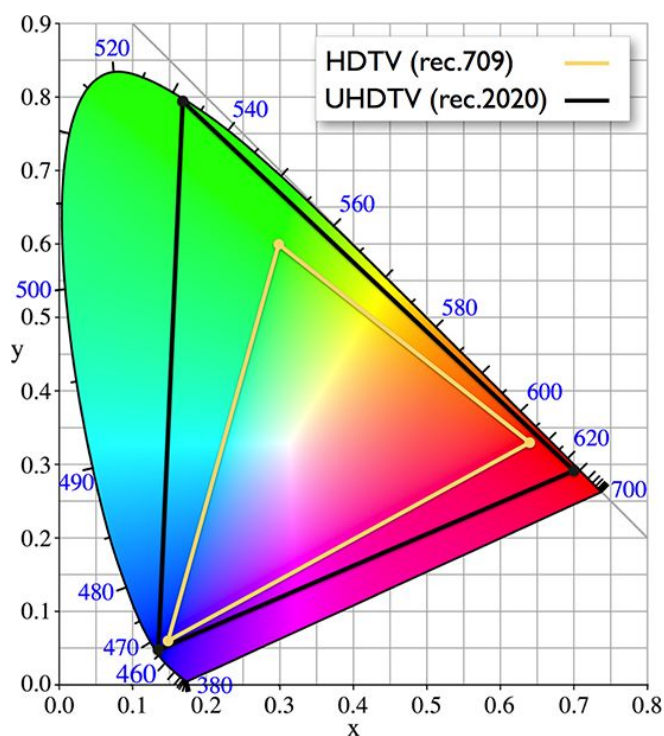
A HDR erre a problémára kínál egy olyan szoftveres megoldást, amellyel a dinamikataromány kiterjeszthető. A funkció bekapcsolásával a fényképezőgép több, eltérő expozíciójú felvételt készít, ezekből végül egy olyan képet generál, amelyen a jól megvilágított területek nem égnek ki, a gyengén megvilágított részek pedig nem tűnnek el a sötétség homályában.

Mozgóképek esetén egyszerűen nincs idő különböző expozíciójú felvételek készítésére és azok összegzésére. A dinamikataromány kiterjesztését a televíziótechnikában, bár ugyanúgy HDR-nek hívják, teljesen más módszerekkel valósítják meg.

Először is speciális, a megszokottnál jóval nagyobb fényerejű kijelzőket használnak, amelyek a megjelenítendő kép fekete területeit nem maszkolással, hanem a fényforrás lekapcsolásával teszik igazán élethűvé. Ezen kívül a három alapszint nem 8, hanem 10 vagy 12 biten ábrázolják, azaz nem 256, hanem 1024 vagy 4096 különböző színárnyalat megjelenítését teszik lehetővé. Az 1. ábra jól érzékelteti, hogy a 36 bites színmélység mennyivel nagyobb területet fed le az emberi szem által érzékelhető színskálából, mint 24 bites.

Ezzel együtt fontos tudni, hogy a kép minőségét alólóképeknél és lassú mozgásoknál elsősorban a felbontás, gyors mozgásoknál pedig inkább a videotömörítés mértéke, valamint a képfrissítés határozza meg.

A moziban annak idején a 24 képkocka/másodperc képfrissítésnél maradtak, mert ennél nagyobb sebességnél rendszerint elszakadt a filmszalag, a televíziótechnikában pedig a hálózati váltófeszültség frekvenciájához igazodva az 50 és a 60 fps (frame per second) vált szabvánnyá. Gyors mozgásoknál az említett képfrissítések mellett még érzékelhető némi elmosódás, ami indokoltta teszi a dupla értékek (100, illetve 120 fps) használatát.



1. ábra

A HDTV és az UHD színskála

Persze vegyük azt is figyelembe, hogy a szemgolyó hátsó részében elhelyezkedő retina csap- és pálcika-sejtjeinek struktúrájából adódóan az emberi éleslátásnak is vannak korlátai. A képernyőtől való ideális távolság a képernyő méretétől és a megjelenített kép felbontásától függ. Bonyolult képletek helyett a gyakorlatban számoljunk az alábbi összefüggésekkel.

- SD: a képernyő magassága × 6
- HD: a képernyő magassága × 3
- 4K: a képernyő magassága × 1,5
- 8K: a képernyő magassága × 0,75

Ezek után joggal vetődik fel a kérdés, hogy vajon fog-e valaki egy 50"-os tévét fél méterről nézni? Nem tudjuk. Az UHD készülékek eladási statisztikái mindenesetre biztatóak. Sőt! Bátor előrejelzések szerint 2020-ra 335 millió háztartásban lesz 4K televízió.

Baranyai Zoltán

A video és audio adatfolyamok felépítése

Áttérünk a bites szemléletre

Az elmúlt években a digitális televíziótechnikát a műsorszórás, az adástechnika oldaláról szemléltük. TS packetekkel, táblákkal dolgoztunk. A video és audio adatfolyamokat packetizált elementary streamként láttuk. A következőkben kísérletet teszünk arra, hogy olvasóinknak bemutassuk ezen adatfolyamok belső szerkezetét is.

A tömörítéssel foglalkozó leírások a kép kockákra bontásával és a képpontok transzformálásával stb. kezdik a téma elemzését. Mivel az irodalomban számos ilyen cikk található, mi rendhagyó módon visszafelé fogunk haladni ezen az úton.

1. A TS packet és a Pack

Bizonyára nem szükséges részletesebben beszélni arról, hogy a tömörített video adatfolyam nem más, mint bájtok egymás utáni „végtelen” sorozata. Természetesen a tömörített hang ugyanilyen bájt-sorozat, csak ott időegység alatt kevesebb bájt érkezik.

A rendet és szabályos formátumokat igénylő televíziótechnika ezeket az adatfolyamokat packetekre darabolva továbbítja. Visszaemlékezve a TS packet szerkezetére, tudjuk, hogy az egy 4 bájtos fejlécből és a hozzá kapcsolt 184 bájt hosszú adattartalomból áll. A fejlécben elhelyezett PID mutatja nekünk, hogy melyek az összetartozó adatfolyamdarabok. A fejlécben a második bájt második bitje a Payload Unit Start Indicator, ez jelöli számunkra a különböző tartalmak kezdetét. Az eredeti adatfolyam kinyeréséhez ennél többre nincs is szükség, az 1. ábrán szemléltetjük az adatfolyam visszaállításának menetét.

Fejléc	Hasznos adattartalom
1	184
0	368
0	552
0	736

PUSI

Visszaállított adatfolyam

1	184	185	368	369	552	553	736
---	-----	-----	-----	-----	-----	-----	-----

1. ábra

A video- és hangadatfolyam visszaállításának menete TS packetekből

2. Bájt-sorozat és bitsorozat

Eddig a pontig bájtokról és bájt-sorozatokról beszéltünk, de tudomásul kell vennünk, hogy a tömörítéssel foglalkozó szakemberek bit szemléletűek.

A TS packetek adatfolyamában a szabályos rendszerben 188 bájt-ként érkező 0x47 értékű szinkron bájt biztosította számunkra az egyértelmű eligazodást. Vizsgálódásunk bitsorozataiban a következő bitkombináció fogja ezt biztosítani:

0000 0000 0000 0000 0000 0001

Bájt-szemléletünkben ez a sorozat a 0x001 hexadecimális szám. Mivel egyéb megkötés nincs a sorozattal kapcsolatban, ez a kombináció bármikor érkezhetsen, illetve biztosítani kell, hogy adatként ne fordulhasson elő az adattartalomban.

Az elementary streamek legnagyobb egysége elméletileg a „Pack”, amely a 0x001BA kóddal kezdődik (bocsánat, hogy nem bitsorozattal adjuk meg), és a 0x001B9-cel végződik. A mi gyakorlatunkban ezzel nagyon ritkán lehet találkozni, ezért felépítését nem részletezzük.

A korábbi években mi sem foglalkoztunk részletesen a video- és audioadatfolyamok felépítésével, ezért mi is elsőként arra a kérdésre szerettünk volna választ kapni, hogy hogyan kell megkeresni a számunkra érdekes jellemzőket. Mintát véve ezekből az adatfolyamokból látható, hogy sok-sok ilyen start kód van bennük. A start kódot követő bájt neve „Stream ID”, és ez az, ami további eligazítást ad az adatok kiértékeléséhez. A 256 lehetséges változat közül két érdekesebbnek tűnő változat jelzése (visszatérve a bites szemléletre):

110x xxxx -audio stream
1110 xxxx -video stream

A pillanatnyi öröm – ami szerint értékes adatra bukkantunk – után elemezve a bitek jelentését, olyan nehezen értelmezhető adatokkal találkozhatunk, mint a PTS (Presentation Time Stamp), DTS (Decoding Time Stamp), ESCR (Elementary Stream Clock Reference) stb.

3. Audio elementary stream

Az egyszerűbb kialakításban reménykedve forduljunk a hangadatfolyamok felé, ahol a fejléc mindig négy bájt-ból áll és a következő szinkron szóval kezdődik:

1111 1111 1111

A 13. bit egy azonosító, amelynek jelentése:

- 0 – ISO/IEC 13818-3 alapján kódolt adatfolyam
- 1 – ISO/IEC 11172-3 alapján kódolt adatfolyam

A 14. és 15. bit jelentése:

- 11 – Layer I
- 10 – Layer II
- 01 – Layer III

A következő négy bit egy táblázat segítségével a Layer számától függően adja meg az adatsebességet. A táblázat 48 adata például a 11172-3-ban látható.

A 20. és 21. bit mutatja a mintavételi frekvenciát a következők szerint:

- 00 – 44,1 kHz
- 01 – 48,0 kHz
- 10 – 32,0 kHz
- 11 – fenntartott

Két bitet átgörva (Padding bit, Private bit), a 24. és a 25. bit jelzi az üzemmódot.

- 00 – stereo
- 01 – joint stereo
- 10 – dual channel
- 11 – single channel

A további két bővítő bájtot is átlépve a szerzői jogok védelmét és az eredeti vagy másolat tulajdonságot jelző biteket követi az emphasis beállítást jelző két befejező bit. Jelentésük:

- 00 – no emphasis
- 01 – 50 / 15 μ s
- 10 – fenntartott
- 11 – CCITT J.17

Akit ennél részletesebben érdekel a téma, például aki a dekódolással is foglalkozni kíván, az a PES Header-ben találja az adatmező hosszát. Az adatok elemzését a PST modulja a PID megadása után egy kattintással tárja elénk, így minden figyelmünket a tanulmányozásra összpontosíthatjuk.

4. MPEG-2 video

Az MPEG-2 vagy H.262 videoadatfolyamok tanulmányozása előtt célszerű, ha tudjuk, hogy ez a tömörítési eljárás az elsők között készült, az elsők között került dokumentálásra és szabványosításra. Ennek ellenére számos helyen látható az utólagos kiegészítés, bővítés. Aki a tömörítési eljárásokat kezdi tanulmányozni, annak célszerű az MPEG-2 megismerését előre venni.

A start kódot követő „Stream ID”-mutatja a videotartalom elemzéséhez szükséges értelmezési kódokat is. Kibővítve, és visszatérve a bájtos formára:

0x00	Picture Start Code
0x01...0xAF	Slice
0xB3	Sequence Header
0xB5	Extension
0xB8	Group of Picture
110x xxxx	-audio stream (0xC0...0xDF)
1110 xxxx	-video stream (0xE0...0xEF)

Nem részletezve, de látható, hogy a DVB szabvány készítői a 0xBC azonosító alatt küldött PMT-t innen emelték ki és csináltak belőle táblát.

A 0x00-as azonosítóval küldött Picture Start Code szakaszt elemezve a Picture Coding Type jellemző tűnik érdekesnek, s valóban az is. A 3 bites adat jelentése bites formára visszatérve:

- 001 – I kép
- 010 – P kép
- 011 – B kép
- 100 – D kép

A szakasz többi adatának feldolgozásáról az ismerkedés első fázisában célszerű eltekinteni. A másik számunkra érdekes adatokat tartalmazó szakasz a 0xB3-mal azonosított Sequence Header. A vízszintes és függőleges méret, a képarány és a képfrekvencia olyan adatok, amit mindenki könnyen tud értelmezni.

Részletesebben tanulmányozva az MPEG-2 kódolási rendszert, látszik, hogy valaki két évtizeddel ezelőtt megvalósított egy eljárást, amellyel bizonyította, hogy a blokkokra bontás, a Diszkrét Koszinusz Transzformáció (DTC) és a hasonló műveletek megfelelőek az adatok tömörítésére. Ezt követően jött egy sor „okos” ember, aki szükségesnek tartotta az eljárás kiterjesztését a kisebb és nagyobb méretű képekre, a még kisebb adatsebességre stb., azonban az alaphoz már senki sem mert (vagy már nem lehetett) hozzányúlani, így minden kiegészítésként került beépítésre. Maga az MPEG-2 kódolási rendszer még könnyen megvalósíthatónak látszik, azonban a kiegészítések, bővítmények rendszere oly bonyolult, hogy szinte lehetetlennek látszik ezek hibátlan megvalósítása.

Az MPEG-2 kódolási rendszer leírása az ISO/IEC 13818-2 és az ICO/IEC 11172-2 szabványokból ollózható össze. A téma sokkal könnyebben érthető, ha a logikus mérnöki tervezési folyamat helyett a kialakulási folyamat szerint próbáljuk értelmezni a megoldásokat. Az MPEG-2 tömörítési eljárást a CD lemezekben alkalmazták nagyobb terjedelemben, így az interneten ehhez kapcsolódva található a legtöbb cikk.

5. MPEG-4 video

Az MPEG-2 tömörítési eljárás megvalósítása és gyakorlati alkalmazása után folyamatosan bővült a témával foglalkozó szakemberek köre és egyre több tapasztalat gyűlt össze egy jobb tömörítési eljárás kifejlesztéséhez. Ezzel szemben nehezen megoldható problémaként jelentkezett, hogy a szabványok készítői annak idején nem gondoltak arra, hogy lesz következő tömörítési eljárás is, így az azonosítók kiosztásánál nem hagytak helyet további új megoldások számára. Ebből adódik, hogy ma egy videoadatfolyamot

elemezve nehezen állapítható meg az, hogy melyik tömörítési eljárással készült. A packetizált továbbítás az MPEG-4-nél a bites szemléletet a bájtos felé tolta.

A H.264/AVC és a HEVC (MPEG-4) tömörítési eljárás kimeneti adatfolyama NAL (Network Abstraction Layer) egységekből épül fel. A leírások szerint ennek alkalmazása nagymértékben javította a tömörítési eljárás rugalmasságát a korábbiakhoz képest, így könnyebb a kisebb-nagyobb sebességű vagy kisebb-nagyobb felbontású igények kielégítése.

A NAL mindig egész számú bájtot tartalmaz és a fejléc kezdő bájtja jelzi, hogy milyen típusú adatokat tartalmaz. A NAL-okat két csoportra osztjuk. A VCL NAL tartalmazza a tényleges képadatokat, a Non-VCL NAL tartalmazza a járulékos (kiegészítő) adatokat. Ezen belül a paraméterkészlet is kétféle. A Sequence Parameter Set (SPS) a pillanatnyi dekódoláshoz szükséges adatokat tartalmazza, a Picture Parameter Set (PPS) a ritkán változó képadatokat tartalmazza.

A NAL egységek azon halmazát, amelyek egy komplett kép adatait tartalmazzák, Access Unit-nak nevezzük. Az olyan egységet, amelyben a kép előállításához nincs szükség előzetes képek adataira, Instantaneous Decoding Refresh (IDR) egységnek nevezzük.

A NAL azonosító bájtjának (első bájt) értelmezése:

első bit	0 – helyes, 1 – szintaktikai hiba
következő két bit 00	– eldobható, nem kell a képhez
(nal_ref_idc)	>0 – fel kell dolgozni a képhez
további 5 bit	a NAL tartalmának azonosítója
(nal_unit_type)	
0	foglalt
1-23	NAL unit
24	STAP-A
25	STAP-B
26	MTAP16
27	MTAP24
28	FU-A
29	FU-B
30-31	foglalt (további részletek: RFC 6184-ben)

A TS packetekben történő továbbítás már nem igényelné, de annak érdekében, hogy a dekóderek a más formában (pl. fájlból) érkező adatfolyamokat is fel tudják dolgozni, a NAL egységek elé beépítették az MPEG-2 rendszernél használt 0x000001 start kódot. A start kód beépítése szükségessé tette egy olyan eljárás beépítését, ami biztosítja, hogy ez a kombináció az adatfolyamban ne fordulhasson elő.

Következő lépésként kezdhethetnénk elemezni a NAL egységek tartalmát, azonban egyelőre piros jelzésünk van, azaz nem mehetünk be erre a területre. A tömörítési eljárás fejlesztésénél néhány tudományosan kidolgozott módszert is igénybe vettek az adatok

mennyiségének csökkentésére. Aki ezeket nem ismeri, nem képes az őt érdeklő adatok kinyerésére.

6. Golomb kód

A Golomb kódolási eljárás Solomon W Golomb úrtól származik, aki a '60-as években dolgozta ki módszerét a továbbítandó adatok mennyiségének csökkentésére. A H.264-ben használt eljárás lényege röviden:

- példaként legyen a kódolandó szám a decimális 20,
- adj hozzá egyet, azaz a továbbiakban ez már 21 lesz,
- ábrázold a számot binárisan, ami így 10101 lesz,
- számold meg a számjegyek darabszámát, ami most 5,
- írd a szám elé ennél eggyel kevesebb 0-t és megkapod a kódolt értéket, amely a mi esetünkben: 000010101.

A dekódolás a fentiek fordított sorrendben történő végrehajtása. Tudományos alapon bizonyított, hogy kis számok esetében az így továbbított adat mennyisége kisebb, mintha a 32 vagy 64 bites formában továbbítottuk volna.



Solomon W Golomb
matematikus
Született: 1932. május 30.
Elhunyt: 2016. május 1.

Mint látható, már megint visszatértünk, a bitsorozatos alakra. Talán éppen itt célszerű megemlíteni, hogy a NAL bitsorozatát mindig egy 1 értékű bit hozzáadásával kell lezárni, majd annyi nullát kell utána írni, hogy a hossz nyolccal osztható legyen, azaz bájtos formában szemlélve is értelmes legyen.

7. A NAL feldolgozása

Első lépésként a 0x00 00 01 Start Code minták megkeresésével daraboljuk fel az adatfolyamot NAL egységekre. Említettük, hogy a NAL-on belül ugyanez a bájtármas nem fordulhat elő, mert ha előfordulna nem tudnánk meghatározni a NAL határait. Ennek teljesítése érdekében ha a NAL-ban mégis szükség lenne a nemkívánatos bájt kombináció továbbítására, akkor azt módosítják. Második lépésben e módosításokat el kell távolítani, ami azt jelenti, hogy

- a 0x00 00 **03** 00 bájt kombinációt 0x00 00 00-ra,
- a 0x00 00 **03** 01 bájt kombinációt 0x00 00 01-ra,
- a 0x00 00 **03** 02 bájt kombinációt 0x00 00 02-ra,
- a 0x00 00 **03** 03 bájt kombinációt 0x00 00 03-ra,

kell cserélni, azaz a módosításként beírt 0x03 bájtot ki kell dobni. A NAL valódi tartalmát kinyerve kezdhető meg az első bájt értelmezése a cikk előző pontjában (lásd bal oldali hasáb) leírtak szerint.

A különböző számokkal azonosított NAL egységek közül elsőként a hetes tartalmát célszerű analizálni. A hetes NAL egység neve: Sequence Parameter Set, felépítése a T-REC-H.264.pdf fájlban található.

A NAL egységben található adattartalmat az irodalomban RBSP (Row Byte Sequence Payload – a hasznos adattartalom nyers bájt sorozata) névvel jelölik, azaz feladatunk a 7-es NAL egységben található RBSP kibontása. A szabvány 7.3.2.1.1. táblázata szerint (nagy mérete miatt nem másoljuk ide) az első három bájt közvetlenül olvasható. Az első bájt a Profile Idc, amely a profile, a harmadik a Level Idc, amely a level meghatározásához szükséges. A második bájt ezekhez kiegészítő flageket tartalmaz. A szabvány „A” melléklete 20 oldalon keresztül részletezi e jellemzők értelmezését. Példaként a DVB-T adásban a Duna World 1201-es PID értékén ez a NAL a következők szerint kezdődik (az első három bitet maszkolva a 0x67 érték 0x07-re adódik):

```
67 64 00 1E AD 90 A4 70 80 32 0B 8D 06 00 E8 3A 40 ..
```

Az első bájt a 0x64, azaz a Profile Idc=100. A harmadik bájt a 0x1E, így a Level Idc=30. A középső flag byte itt 0x00 értékű. Az említett három bájtot követő adat már Golomb kódolt, a megfejtéshez a bitsorozatot szemléltetjük:

```
1010 1101 1001 0000 1010 0100 0111 0000 1000 000
```

Mivel a sorozat előtti nullák száma nulla, a hossz 1, innen az érték 1, amiből 1-et levonva az adat 0. Fontos megjegyezni, hogy 1 bitet használtunk el a Sequence Parameter Set Id=0 meghatározásához, tehát a következő adatot a második bittől kezdve kell olvasni. Aki jól csinálja, az a Chroma Format Id=1 értéket fogja kapni. Aki téveszt a pillanatnyi pozíció meghatározásánál, az ebből a NAL-ból már nem tud több helyes adatot kiolvasni.

Természetesen előjeles számok ábrázolására is ki kellett terjeszteni a Golomb kódot. Egyszerűen megfogalmazva: negatív számoknál és a nullánál a $2x$, pozitív számoknál a $2x-1$ értéket kódoljuk, ha x az ábrázolandó szám abszolút értéke. A teljesség érdekében megemlítjük, hogy a két kódolt adat mellett még egy vagy több bit kiolvasására kell felkészülni a bitsorozat olvasásához.

Első olvasásra lehet, hogy nem mindenki számára könnyen értelmezhetők a fent leírtak, ezért másként is megfogalmazzuk az olvasás menetét. A H.264 adatfolyamban elhelyezett NAL egységek szerkezetét (az adatok sorrendjét) a fent említett szabvány mutatja. Ennek táblázataiból lehet megtudni, hogy mikor, mit, és hogyan kell olvasni. Példaként az olvasás és értelmezés menetét így is meg lehet fogalmazni:

Olvass ki három bájtot, ez lesz az A adat, majd olvass ki 5 bitet, ez lesz a B adat. Olvass ki egy pozitív Golomb számot, ez lesz a C adat. Olvass ki egy negatív Golomb számot, ez lesz az D adat, és így tovább. A kiolvasás menetét jelentősen megkönnyíti, hogy a kiolvasott adatok számos elágazást befolyásolnak, azaz, a kiolvasott értéktől függően másként kell folytatni az olvasást. Nagyon fontos látni, hogy téveszteni nem lehet, mert a hibától kezdve a NAL további adatai nem értelmezhetők, mivel egyetlen fix támpont sincs benne a továbbiak értelmezéséhez.

Igaza van annak, aki úgy véli, hogy a H.264 szerint kódolt kép bitsorozatának olvasása meglehetősen nehéz feladat, és kifejezetten nagy figyelmet és körtekintést igényel.

Az ilyen adatfolyamok értelmezését is elvégzi a PST modulja, ha MPEG-4 formátumot állítunk be a lenyíló listán. Talán megbocsájtható, de modulunk egyelőre csak a fontosabbnak vélt NAL egységeket analizálja részletesen.

8. A H.265

Akinek sikerült elsajátítani a H.264 adatfolyamok olvasásának menetét, annak a H.265 változattal sem lesz problémája. A részletek az ITU-T H.265 v4 (12/216) szabványban találhatóak. Talán helyes az a megállapítás, ha azt mondjuk, hogy a H.265 kódolási rendszer a H.264 továbbfejlesztett változata. Szerintünk itt nincs akkora különbség, mint amit a H.262 irányában láttunk.

9. Összefoglaló

Cikkünket igyekeztünk úgy összeállítani, hogy aki elolvassa, annak legalább közelítőleg fogalma legyen a környezetünkben nap mint nap használt tömörítési eljárásokról. Igaza van annak, aki úgy véli, hogy üzemeltetői szinten nincs szükség ennél részletesebb ismeretekre, azonban mi úgy véljük, hogy nem árt ha legalább „újságírói szinten” tisztában vagyunk az alapokkal.

A cikk megírására nem véletlenül került sor. A háttérben folyamatban van a PST video- és audioanalizátor moduljainak fejlesztése. A Video Info modul már most képes arra, hogy paketizált tömörített video adatfolyamról megállapítsa annak H.262 vagy H.264. vagy H.265 szerinti kódoltságát. A tömörítés típusának megállapítása után képes a fő jellemzők (képméret, arányok, képfrekvencia stb.) meghatározására. Fejlesztésünk jelenleg a képek kinyerésén és megjelenítésén dolgozik, de mivel vélelmezzük, hogy e téma kiemelt érdeklődésre számíthat, a részletekről következő újságunkban fogunk beszámolni.

Veres Péter és Zigó József

TS packet – Tömörített adatfolyam – Konténer - mp4

Út az mp4 felé

Olvasóink többsége a TS packetek birodalmában jártas. Újságunk első cikkében megmutattuk, hogyan lehet visszatérni a tömörített adatfolyamhoz, majd azt is megmutattuk, hogyan lehet ezt olvasni, pontosabban hogyan kell értelmezni az adatfolyamot.

A tömörített adatfolyam átvitelének egy másik módja az interneten, vagy különböző adathordozókon keresztül történő továbbítás. Sok-sok cégek közötti huzavona, versengés és szabadalmi vita után az mp4 formátum kezd az élre törni, megelőzve az ogg, mpg, webm stb. néven ismert, szinte megszámlálhatatlanul sok versenytársát.

Cikkünkben az mp4 formátumot kiemelve ezekről adunk némi tájékoztatást úgy, hogy közben a kapcsolódó fejlesztéseinkre is kitérünk.

Az említett multimédiás konténerformátumokról elsőként annyit kell tudni, hogy ezek mindegyike a mi TS packetünkhöz hasonlóan egy keretrendszer, amelybe a megadott módon kell az adatokat beépíteni. Az mp4 konténer hierarchikusan elhelyezkedő egységekből, ún. atomokból épül fel. Az atomok további atomokat, vagy az adatok egy jól meghatározott részét tartalmazhatják. Az adatok értelmezésének első lépéseként tudni kell, hogy az atom adatfolyama mindig a 4 bájtos hossz adattal kezdődik, amit a név 4 karaktere követ. Példaként a **00 00 00 20 6D 64 61 74** hexa bájtsorozat a 32 bájts hosszú (MSB-LSB) és „mdat” elnevezésű dobozt



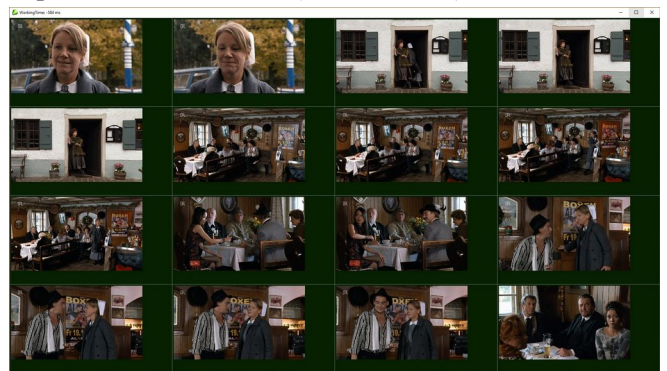
jelöli. Nagyon fontos megjegyezni, hogy amíg a DVB rendszerben a hossz mindig az adat után következő bájtok számát jelenti, addig itt a hosszadat 4 bájttal is beleértendő a méretbe. Mp4 esetén a médiaadatok (kódolt video- és audio- egységek) és a metaadatok (stream típus,

időbélyegek, mintahossz, kódolási eljárás, képfelbontás, IDR képek pozíciója stb.) szétválasztva tárolódnak. Az elsöre bonyolultnak tűnő szerkezet

ellenére az mp4 fájlok szoros kapcsolatban állnak a transport streamekben található kódolt elementary streamekkel, megfelelő algoritmusokkal átalakíthatók egymásba a hang- és képlelek újrakódolása nélkül.

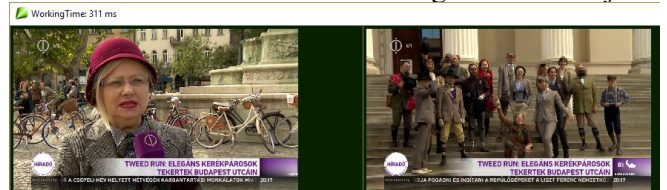
A webes világ az elmúlt hónapokban véglegesen a H.264 szerinti tömörítés, az mp4, webm és ogg formátumok, valamint a HTML5 videomegjelenítő mellett tette le a voksát, és megkezdődött minden más formátum és lejátszó kiirtása. Minket ez annyiban érint, hogy jövőben webes környezetben csak az mp4-be csomagolt, és H.264 szerint tömörített adatfolyamok képeinek megjelenítésére kapunk lehetőséget.

Fejlesztésünk munkájából előzetesként annyi, hogy a PST új modulja a csatorna számát és a PID értékét meghatározó parancs után elkezd egy I képpel kezdődő videointát gyűjteni az MPEG-2, H.264 vagy H.265 kódolású streamből. Némi gyűjtési idő biztosítása után a VID01T.mpg fájlt kiolvastva megkapjuk az első képet a dekódoláshoz szükséges információkkal együtt. A fájl nevében a 01 helyére nagyobb (pl. 08) számot írva a következő különbségi képek egy sorozatát is megkapjuk. A műveletet ciklusba téve például a következő képen látható képsorozat állítható elő (T = kb. 1 sec):



A képeket egy műholdas adásból, az elmúlt hónapban vásárolt FFmpeg dekóder modullal írt szoftverünk állította elő Windows környezetben.

Fejlesztésünk másik ága azon dolgozik, hogy ezt az adatfolyamot mp4 konténerbe csomagolja a webes környezetben történő képmegjelenítéshez. A VIM01T.mpg parancssal kiolvasott HD képre mutat példát a befejező kép. E bevezető után a témakör részleteivel februári számunkban foglalkozunk majd.



dr Zigó Tamás

A tömörített video-adatfolyamok dekódolása

Az FFMPEG fejlesztői környezet bemutatása

A legtöbb felhasználót csak a tömörített video-adatfolyam külső jellemzői érdeklik. Megjeleníthető-e, akadás mentesen, kockásodik-e. Ezek ellenőrzéséhez a legelterjedtebben használt szoftver a VLC Media Player.

Mint azt az előző cikkben is próbáltuk érzékeltetni, a tömörített videoadatfolyamok dekódolása nagyon bonyolult feladat. A problémát valójában nem a dekódolási folyamat megvalósítása, hanem a különböző cégek és szabadalmi érdekek által „rástoppolt” kiegészítések kibontása okozza.

A téma bonyolultságából adódóan nem sok cég vállalkozik arra, hogy saját dekóderszoftvert fejlesszen termékéhez. Még a legnagyobb műszergyártók is az ismert műhelyek egyikéből választanak készülékükbe dekóderszoftvert. Például a Rohde & Schwarz is előszeretettel építi készülékeibe a VLC-t.

Fejlesztéseink során talákoztunk az FFMPEG multimédiás céggel (ffmpeg.org), amely talán kevésbé ismert olvasóink körében, azonban a tömörítési eljárások vizsgálatához sokkal több segítséget nyújt, mint más cégek szoftverei. Cikkünkben azok számára kívánunk segítséget nyújtani, akik e cég szoftverein keresztül kívánnak további ismereteket szerezni a tömörítési folyamatokról.

Az előző cikk bevezetőjében láttuk, hogyan kell a transport streamben használt paketizált adatfolyamból visszaállítani azt a bájtfolyamot, amelyet a tömörítés végző encoderek szolgáltatnak. A cikkből azt is megtudtuk, hogyan kell ezt az adatfolyamot olvasni, hogyan lehet a benne tárolt információk egy részét kiemelni. A következőkben a képek kinyerésére teszünk kísérletet, remélve, hogy e szemléletesebb leírás kevésbé riasztja el az olvasót a témakör tanulmányozásától.

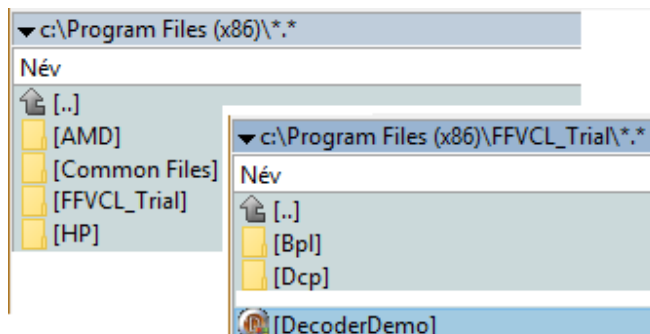
Windows operációs rendszer alatt futó számítógépünkkel látogassunk el a www.delphiffmpeg.com honlapra. Anélkül, hogy komolyabban elmélyednénk a részletekben, töltsük le az 1. ábrán látható exe fájlt.



1. ábra

A Delphi FFmpeg cég korlátozottan használható ingyenes szoftver verziójának telepítője

Ma már mindenki fél a különböző szoftverek telepítésétől, mégis aki komolyabban szeretne foglalkozni a videós témával, annak azt javasoljuk, hogy bátran futtassa a letöltött exe fájlt, amely egy szoftvert telepít számítógépünkre. A telepítést követően a 2. ábrán látható módon egy FVCL_Trial könyvtár jelenik meg számítógépünkön.



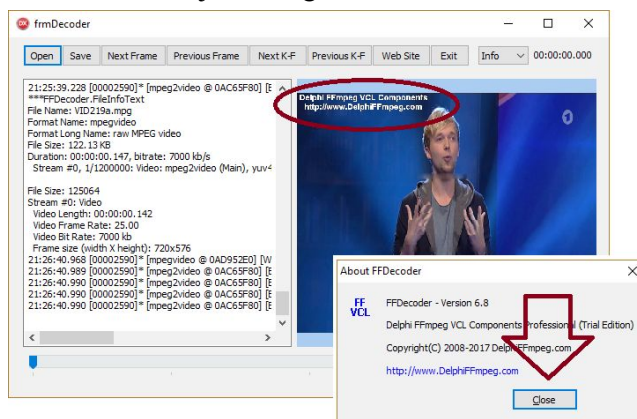
2. ábra

Az FFmpeg trial telepítése után megjelenő könyvtárak

Az FFVCL_Trial könyvtárban több mint 30 különböző alkalmazás demo változata közül választhatunk. Témánkhöz most legközelebb a DecoderDemo.exe futtatása során megjelenő alkalmazás áll, ezért elsőként indítsuk el ezt.

Megjegyzés: ugyanez az alkalmazás telepítés nélkül is is átvihető másik gépre a dll-ek biztosításával.

Esetünkben a trial verzió azt jelenti, hogy a szoftver egy reklámfeliratot helyez el a megjelenített képen, illetve a fájl betöltésekor egy felugró ablakban az OK gombra is kattintanunk kell időnként. A 3. ábrán szemléltetjük e megkötéseket.

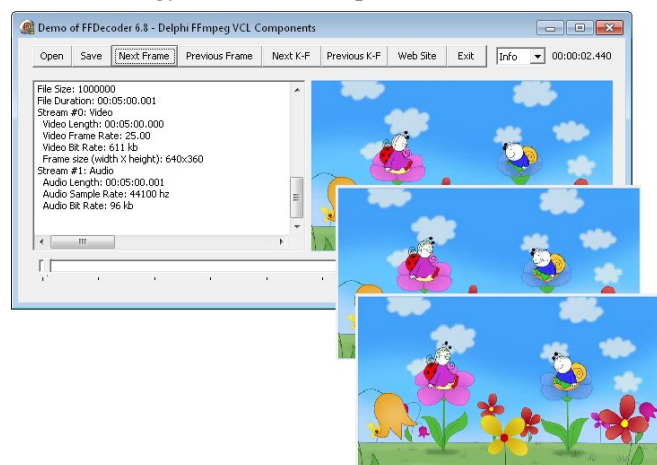


3. ábra

A trial verzió miatt megjelenő korlátozások

A 3. ábra egy műholdas adásból visszaállított MPEG-2 adatfolyam vizsgálata során készült. A packetekből történő visszaállítást a PST végezte.

Aki csak ismerkedés céljából futtatja a DecoderDemo-t, annak először egy *.mp4 fájl betöltését ajánljuk. Az mp4 konténeren belüli MPEG-4 adatfolyam már alkalmas a kódolás tanulmányozására. A DecoderDemo szoftver a betöltést követően az első képet mutatja. A felső sorban elhelyezett gombokkal képenként és I képenként is lépkedhetünk. A 4. ábrán érdekességgént a képenkénti léptetésből mutatunk be részletet. A felvétel készítésénél picit csalunk, a szemléletesség érdekében egynél többet is léptünk.



4. ábra

Az MPEG-4 adatfolyamon képenként lépkedve a rajzfilm készítésénél megszokott látvány tárul elénk

A lépünk nagyobbbat és nézzük meg, milyen komplett, azaz I képek vannak az adatfolyamban. Az 5. ábra felvételén az I képeken lépkedve szemléltetjük a megjelenő képeket.



5. ábra

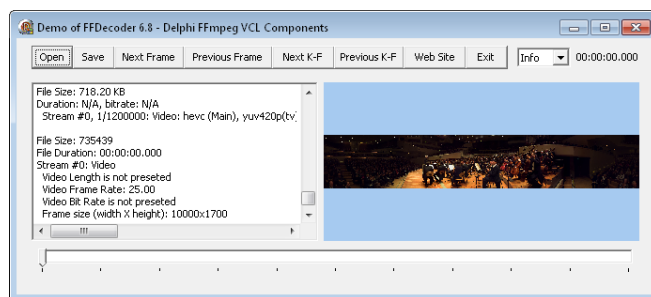
Az I képeken lépkedve megjelenő képek

A cikk készítésénél azért használtunk egy YouTube-ról letöltött rajzfilmet, mert ez szemléletesebb, mintha a bemondók arcvonásait kellene megfigyelni.

A kódolási jellemzők tekintetében érdemes észrevenni, hogy az MPEG-2 adatfolyamban sokkal több referenciakép van, mint az MPEG-4 szerint kódoltakban. Természetesen ez nem véletlen, az adatsebesség csökkentése érdekében az MPEG-4 eljárásban a Golomb kód stb. alkalmazása mellett a nagy méretű referenciaképek számát is csökkentették. A bemutatott felvételen a referenciaképek között esetenként 32, 29, 41 darab különbségi képet is számoltunk.

A különböző módon kódolt adatfolyamokat vizsgálva jusson eszünkbe, hogy az MPEG-2 adatfolyamokban sokkal könnyebb biztos pontokat találni, így sokkal könnyebb információkat kiolvasni, mint az MPEG-4 szerint kódoltakból. A megállapítás arányosan tovább vihető, azaz az MPEG-4 szerint kódolt HD adatfolyamok olvasása is nehezebb, mint az ugyanilyen módon kódolt SD adatfolyamok értelmezése. Pontosítva az olvasás nehézsége alatt itt azt értjük, sok sokkal nagyobb mennyiségű adatot kell átnézni az információ kinyeréséhez, mint a másíknál. Analizáló szoftvereinkben ez úgy jelenik meg, hogy az információ kinyeréséhez egyre több és több idő szükséges, miközben a felhasználó türelmetlen és mindent azonnal tudni szeretne.

Visszatérve az FFMPEG szoftverünkhöz mindenképpen meg kell róla említeni, hogy a leírás szerint elképesztően sokféle módon kódolt adatfolyam felismerésére képes. Minket ezek közül elsősorban a H.265 érdekelt, ezért röviden ezt is megvizsgáltuk egy PST által készített mintával. Mint az 5. ábrán is látható, a szoftver a 10000x1700 pixel felbontású adatfolyammal is könnyedén megbirkózik.



5. ábra

Kép megjelenítése egy H.265 szerint kódolt adatfolyamból

Befejezésül előre jelezzük, hogy újságunk következő számában folytatjuk a video- és audio-adatfolyamok vizsgálatát és a PST kapcsolódó moduljainak bemutatását.

Zigó József

A hangerő mérése és beállítása objektív módszerrel

Nem csak nálunk probléma a műsorok eltérő hangossága

Valószínűleg mindannyian találkoztunk már azzal a jelenséggel, hogy csatornaváltás után a tv-készülék hangszórójából üvölt a zene, vagy éppen egy kukkot sem értünk a kedvenc filmsztárunk szavaiból, mert olyan halk az adás. Ilyenkor bőszen nyomkodjuk a távvezérlő hangerőszabályzó gombjait a megfelelő hangosság beállítása érdekében. Az is gyakran előfordul, hogy a műsort megszakító reklámblokk lényegesen hangosabbnak tűnik, mint maga a műsor. Ezt a hangsáv manipulálásával érik el. A dinamika tartományt lecsökkentve a halkabb hangokat felerősítik. A maximum jel-szint nem változik, viszont a magasabb átlagos hangerő miatt az emberi fül hangosabbnak érzékeli a hirdetéseket.

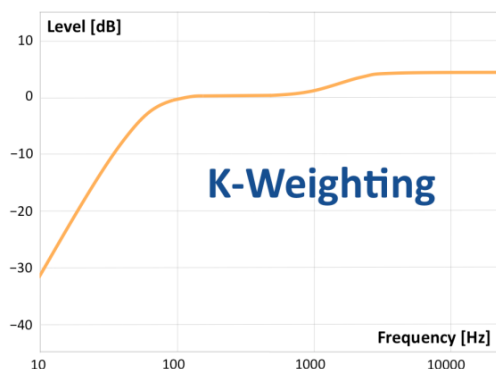
A médiahatóság feladata a törvényben meghatározott hangerő használatának betartatása, de az ilyen trükkök feltárása nem mindig egyszerű feladat.

1. Hogyan mérjük a műsor hangerejét?

Az új eljárások kidolgozásakor számos alapfogalmat vezettek be. Ezek az alábbiak:

K-Weighting (K súlyozás)

A hagyományosan alkalmazott módszer, ahol a hangminták csúcsértékének mérése (sample-peak level) helyett a szakemberek új megközelítést alkalmaznak. Hosszadalmas kísérletek és kutatások bizonyítják, hogy annak ellenére, hogy két hangminta hangossága a sample-peak level módszerrel mérve azonos, az emberi fül azokat nagyon eltérőnek érzékelheti. Számos független szervezet közreműködésével kifejlesztettek egy olyan eljárást, amely a hang szintjét az érzékelt hangosság (loudness) alapján méri. A módszer legfontosabb eleme egy úgynevezett K-weighted felüláteresztő szűrő alkalmazása minden egyes hangcsatornában, amely kapcsolatot teremt a szubjektív érzékelés és az objektív mérés között. A szűrő részletes leírása az ITU-R BS.1770 ajánlásban található.



K súlyozó szűrő karakterisztika (EBU TECH 3343-2016)

LU, LUFS, LKFS

Az LKFS a Loudness K-weighted Full Scale rövidítése, hevenyészett fordításban a K-súlyozott hangosság teljes kivezérlésnél. A fogalmat az ITU-R BS.1770 vezette be, ugyanakkor az EBU a Loudness Units Full Scale (LUFS) terminust használja. Az eltérő név ellenére a két fogalom teljesen azonos. 1 LKFS = 1 LUFS = 1 dB. Az LKFS/LUFS javasolt névleges értéke (ajánlástól függően) -23 LUFS, illetve -24 LKFS.

Az előző abszolút mértékegységek mellett a megszokáshoz igazodóan egy relatív mértékegységet is bevezettek (Loudness Unit, LU), tehát -23 LUFS = 0 LU. 1 LU szintén 1 dB-t jelent.

Loudness Range (Hangosság tartomány)

A Loudness Range (LRA) meghatározza az adott hanganyag hangerőtartományát a leghalkabb értéktől a leghangosabbig, LU egységben. A mindkét irányban esetlegesen előforduló extrém értékek figyelmen kívül hagyása érdekében a tartomány alsó 10 és felső 5 százalékát a mérésből kizárják.

Program Loudness (A program hangossága)

A Program Loudness az adott teljes hanganyag (program) átlagos hangosságát fejezi ki. Időnként integrált hangosságnak (Integrated Loudness) is nevezik. Megadása LUFS vagy LKFS egységekben történik.

Gating (Kapuzás)

A Program Loudness számítása során nem mindig kívánatos bizonyos hatások (pl. hosszú csöndes szakaszok) figyelembevétele. Ilyenkor olyan kapuzási séma használata célszerű, amely leállítja a mérést amennyiben a hang a kapuzatlan esetben mért értékhez képest -10 LU szint alá esik. Ennek az az előnye, hogy így a különböző műfajú és hosszúságú hanganyagokat lényegesen könnyebb hangerő szempontjából illeszteni. Ez a probléma a műsor-szórás területén gyakran jelentkezik (pl. egy kétórás filmet egy 20 másodperces reklámhoz kell illeszteni), amit hatékonyan segít a kapuzás módszere.

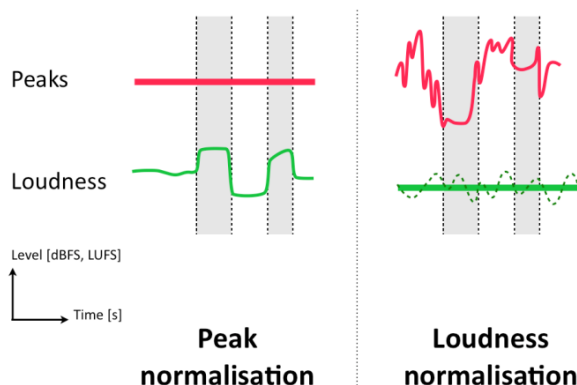
Target Levels (Névleges hangerőszint)

A névleges szintet a különféle szabványok kissé eltérően határozzák meg. Például az ATSC A/85 a -24 dB értéket javasolja és az LKFS fogalmat használja, ezzel szemben az EBU R128 a LUFS-t, és a -23 dB értéket javasolja. A különbség fő oka az előzőekben említett (EBU által javasolt) kapuzási

eljárás, amely segít azonos hangossági értékre állítani az eltérő műfajú vagy tartalmú hanganyagokat.

True-Peak (Valódi csúcserő)

Mivel a hangerőmérés tárgyalt módszere a szubjektív érzékelésből kiinduló algoritmuson alapul, a hanganyag, ami ugyan megfelel a szabványok szerinti LRA és Program Loudness értékeknek, túlzérlést okozhat, ha a normalizálás a hagyományos sample-peak módszerrel történt. Ennek elkerülésére a normalizálás sok műsorszórási szabvány része és a szolgáltatóknak a megfelelő értékek betartásához true-peak mérő használata javasolt.



A hangerő normalizálása a csúcserő (hagyományos eljárás), és az érzékelt hangosság alapján (EBU TECH 3343-2016)

A loudnessmérő eszközök általában tartalmaznak true-peakmérőt is, amely a hagyományos csúcserőmérőtől eltérően speciális algoritmust használ. Ez lehetővé teszi, hogy a mérés ne csak az aktuális mintákat mérje, hanem azokat a csúcsokat is vegye figyelembe, amelyek később, a D/A konverziós szűrőben 0 dB FS értéknél nagyobb jelszintet eredményezve torzítást okoznának (inter sample peaks). A fentiek miatt, ha a hagyományos (sample peak) módon mért érték, pl. -0.2 dB, egy true peak mérő akár +3 dB szintet is mutathat.

Az EBU R128 ajánlás alapján a maximális normalizált jelszint -1 dBTP (dB true-peak) lehet.

2. Szabványok és ajánlások

Az újonnan bevezetett mennyiségek mérésének módszereit számos szabvány és ajánlás részletezi.

ITU-R BS.1770: A témában született legtöbb ajánlás és szabvány alapjául szolgál. Az ajánlás részletesen leírja a műsorszórásban használt hangosság (Broadcast Loudness) és csúcserő (True-peak Level) mérését.

EBU R128: Az EBU a hangszintek normalizálására tesz ajánlást a Programme Loudness, a Loudness Range és a Maximum True Peak Level paraméterek mérésével.

Alapvetően a BS.1770 ajánlásra épül, de további eljárásokkal is kiegészíti azt a mérés pontosabbá és konzisztensebbé tétele érdekében az eltérő műsorfajták méréséhez. Alapvetően négy EBU technikai dokumentumra épül: Tech 3341, 3342, 3343, 3344.

EBU Tech 3341: Az R128 keretein belül lefektetett hangosság-mérés módszereit tartalmazza. Az „EBU mód” definíció szerinti hangosság-mérést definiálja. Ennek legfontosabb jellemzője, hogy három időskálán történik a mérés: pillanatnyi érték (Momentary), rövid idejű (Short-term), és integrált, vagy program hangosság (Integrated, Program Loudness). Ennek megfelelően bármely EBU módú real-time mérőeszköznek képesnek kell lennie a három érték megjelenítésére, valamint a pillanatnyi hangosságérték maximumának kijelzésére. A pillanatnyi érték mérése 0,4 mp-es időablakban történik, a rövid idejű hangosság pedig 3 mp-esben.

A Program Loudness mérés speciális időkapuzási eljárást használ, ami kizárja azon komponensek hatását, amelyek -10 LU szint alá esnek a kapuzatlan értékhez képest. A fentiekén túl az EBU módú szintmérőnek képesnek kell lennie kijelezni az LRA értékét is.

EBU Tech 3342: A hangszintek normalizálási folyamatát (és a Loudness Range meghatározását) írja le az ITU-R BS.1770 ajánlásban található számítási módszerek felhasználásával.

Az LRA kiszámítása a mérendő hanganyag karakterisztikájának statisztikai eloszlásán alapszik. Ez azt jelenti, hogy egy hosszú időn át számított LRA értéket egy rövid idejű, nagy hangerejű esemény alig befolyásol. Hasonlóan, a hangsáv végén lévő elhalkuló szakasz nem növeli észrevehetően a Loudness Range értékét. Ezt úgy érik el, hogy a tartomány 10% alatti és 95% fölötti részét az algoritmus figyelmen kívül hagyja.

EBU Tech 3343: Ez a dokumentum az EBU R128-ban megfogalmazottak alapján megvalósítási segédletet tartalmaz a különféle hanganyagok előállítói, feldolgozóinak számára. Megtalálható benne a Program Loudness, a Loudness Range, és a True-peak részletes magyarázata, valamint a különféle hangosság-beállító stratégiák megvalósítási lehetőségeinek leírása.

EBU Tech 3344: Az ajánlás meghatározza, hogyan lehet a hanganyagok hangosságát normalizálni a különféle végfelhasználói platformok (rádió, televízió, hordozható eszközök, stb.) felé történő szétosztás esetén.

Úgy gondolom, a fentiek alapján nyilvánvaló, hogy a programok hangerejének korrekt beállítása a megfelelő készülékek segítségével könnyen elvégezhető. Bízunk benne, hogy a műsorkészítők, szolgáltatók meg is teszik ezt.

Veres Péter

Personal Stream Tool **Extra**

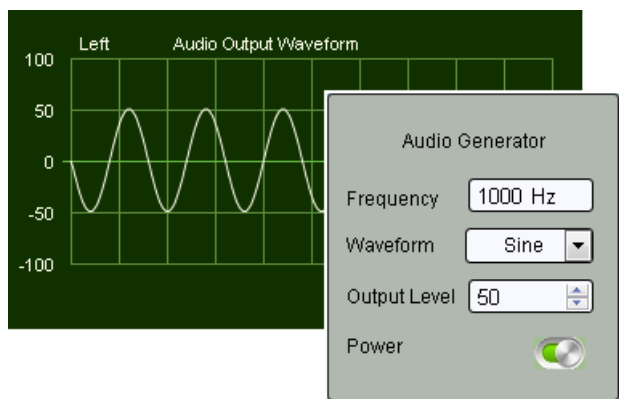
Bizonyára olvasóink is tisztában vannak vele, hogy a készülékek fejlesztése bonyolult és összetett feladat. Egy-egy jó termék kialakításához számos lehetőséget kell megvizsgálni, számos megoldást kell tesztelni. A PST audio moduljának fejlesztése közben bukkantunk rá a böngészők Web Audio API-jára, amely fejlesztőmérnökeinket is izgalomba hozta.

Az internet használat világméretű elterjedésével a webböngészők fejlesztésének két kiemelt iránya

- a vásárlások folyamatának tökéletesítése, és
- a játékok élményének fokozása lett.

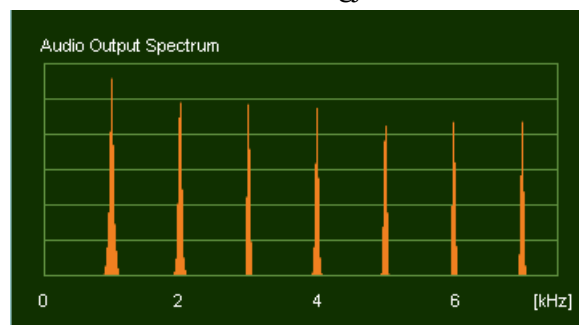
Cikkünkben azt mutatjuk be, hogy a hangeffektusok tökéletesítéséhez kifejlesztett modult hogyan állítottuk a PST felhasználók szolgálatába. A modul a v1.08 változattól mindenki számára elérhető, de szélesebb körben nem publikáltuk, a gépkönyvben még nem említjük.

A Web Audio API a hangok előállításához egy adattárolóból vezérelhető D/A konvertert kínál. Az adattárolót véletlenszerű adatokkal feltöltve zaj vagy zaj csomagok jelennek meg a kimeneten, amelyeket a játékokban „puskalövés” és hasonló effektek előállítására használnak. A PST esetében ebből a modulból egy olyan hanggenerátort alakítottunk ki, amelyik szinusz-, négyszög-, háromszög- és fűrészjelet is elő tud állítani a zaj mellett. A jel frekvenciája elméletileg 20 Hz és 20 kHz között szabadon állítható, de a gyakorlatban csak a 20 Hz ... 10 kHz-es tartomány használható, mivel a hangfrekvenciás jelet számítógépünk hang kimenete szolgáltatja. A szoftver kezdetleges módon a jelalak megjelenítésére is lehetőséget ad, mint az a következő képen is látható.



A modul az Expert/Video-Audio/Audio Analyzer menüben érhető el. Valójában a PST-nek ehhez a szolgáltatáshoz semmi köze, a szoftver PST nélkül fájlból is futtatható.

Az említett API egy „FFT” (Fast Fourier Transformator) függvényt is tartalmaz, ami lehetővé tette számunkra egy spektrumanalizátor beépítését is. Érdekességként az 1 kHz-es fűrészjel spektruma a következők szerint néz ki a megjelenítőn:

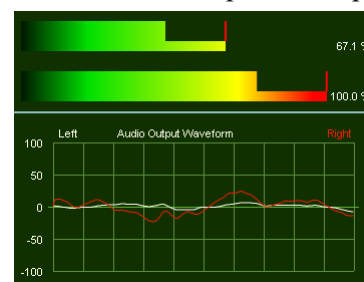


Jogosan vetődik fel a kérdés: mire jó mindez, miért építettük a PST szoftverébe?

A hanggenerátor szolgáltatás bármilyen helyszínen segítség lehet egy kábelszakadás felderítésére vagy az erősítő működőképességének megállapítására stb.

A jelalakok bemutatása és a hozzá kapcsolódó spektrumkép szemléltetése az oktatásban, továbbképzésben lehet hasznos.

A szoftver jelenlegi változata *.wav és *.ogg fájlok lejátszása esetén képes a hullámalak, a spektrumkép, és a kivezérlés mértéké-nek kijelzésére. Egy zenefájl lejátszása esetén megjelenő képre láthatunk példát a következő felvételen.



A valós ok, amiért a kérdéssel foglalkozunk, az a DVB rendszerben sugárzott műsorok hangosságának mérése. A fejlesztés még javában folyik, a fentiek csak részeredmények. Továbbra is várjuk, olvasóink véleményét, megjegyzéseit, különösen a fenti témához kapcsolódóan.

De Vescovi Róbert