Videótömörítés Linuxon, még hatékonyabban

Linux alatt is készíthetünk jó minőségű, kis helyet foglaló H.264-es tömörítésű videót. A digitális videózásban egyre nagyobb szerepet kap a H.264. Apple és a hozzá hasonló cégek is szívesen használják. Az MPEG-4-es szabvány részeként (part 10) a H.264 bekerült a nagyfelbontású HD-DVD és a Blu-ray szabványokba is. Ez annak köszönhető, hogy a H.264 kis sávszélesség mellett is hihetetlenül jó minőséget képes nyújtani.

■ Az alacsony sávszélesség igény teszi számunkra érdekessé ezt a kódolási eljárást. Szerencsére ***Linux***on megjelent az ***x264***, amely sikeres és hatékony nyílt forrású megvalósítása a ***H.264***-nek, mely ***Advanced Video Codec (AVC)*** néven is ismeretes. Valójában az ***x264*** projekt megnyerte a ***Doom9*** 2005-ös kodek tesztjét is. Az ***x264*** egy jelenleg is aktív projekt, melyet folyamatosan fejlesztenek. A kiemelkedő minőségű ***AVC kodek*** előnyeit számtalan helyen élvezhetjük kezdve a házi videógyűjteményünk biztonsági másolataitól, a webes videótovábbításig, de érdekes lehet csupán kísérletezgetni a legújabb technológia nyújtotta lehetőségekkel.
Jelen cikk célja bemutatni, hogyan készíthetünk egyszerű lépésekkel ***.mp4*** állományt, mely ***H.264***-es videóval és ***AAC*** (***Advanced Audio Codec***, szintén ***MPEG*** szabvány része) hanggal rendelkezik. Azonban egy összetett és átfogó cikk messze meghaladná a rendelkezésre álló terjedelmet, azonban ez remélhetőleg nem riasztja el az Olvasót a téma mélyebb tanulmányozásától.
Minthogy az ***AVC*** és az ***AAC*** is az ***MPEG*** szabvány része, így jó pár segédeszköz (legyen az kereskedelmi vagy másmilyen) támogatja ezeket. Például az ***Apple QuickTime*** programjával is készíthetünk ilyen videókat, illetve a népszerű nyílt forrású ***Mplayer*** is használható ***.mp4*** fájlok lejátszására.

Vágjunk bele

A célként kitűzött videófájl létrehozásához három egyszerű lépést kell végrehajtanunk: a tömörített videó létrehozása, a tömörített hang létrehozása és végül, de nem utolsósorban a két fájlt egyesítenünk kell megfelelő módon. Ehhez az alábbi programok szükségesek:

- ***MPlayer*** (tartalmazza az ***mencoder***-t, ***CVS*** verzióból ***060109*** vagy frissebb szükséges)
- ***faac*** 1.24 vagy frissebb
- ***MP4Box*** (***gpac*** része, 0.4.0 vagy frissebb szükséges)
- ***x264*** (***gpac*** támogatással fordítva)

A célunk: létrehozni egy kis sávszélességű videót, mely a weben könnyen továbbítható. Kis méretű lesz, de reményeink szerint hasonló minőségű, mint egy nagyobb sávszélességű ***XviD***-el tömörített videó. Forrásként egy kilenc másodperces ***raw*** formátumú házi videót (`max.dv`) használunk, melyet a digitális videókameráról mentettünk le.
Kezdjük a hanggal, mely átalakítása igencsak egyszerű művelet. Az ötlet: nyers videóból kinyerhetjük ***Mplayer***rel a hangot:

```
mplayer -ao pcm -vc null -vo
➥null max.dv
```

A művelet végén kapunk egy állományt, melynek neve `audiodump.wav` lesz. A videóval egyelőre nem foglalkozunk. Alakítsuk át ***AAC*** formátumra:

```
faac --mpeg-vers 4
➥audiodump.wav
```

Az `--mpeg-vers` kapcsolóval jelezhetjük az ***MPEG*** verziót. A végeredményt le is ellenőrizhetjük: játsszuk le az `audiodump.aac` fájlt ***Mplayerrel***.
A videó tömörítésekor több megoldás közül választhatunk. A legjobb minőség érdekében többmenetes tömörítést kell alkalmaznunk. A videót legalább kétszer kell feldolgozni ahhoz, hogy a rendelkezésre álló sávszélesség optimálisan legyen elosztva a videón. A többmenetes tömörítéssel hajszálpontosan beállítható a sávszélesség és a fájlméret. Sajnos azonban az ***AVC*** tömörítőprogramok, mint amilyen az ***x264*** is, meglehetősen igénybe veszik a processzort és emiatt hosszú a feldolgozás is. Éppen ezért nem várható el, hogy a többmenetes tömörítésre várjon az Olvasó. Ehelyett lehetőség van egymenetes tömörítésre is. Noha ez is kiváló minőséget ad, sose lesz olyan jó, mint a többmenetes. Az egymenetes tömörítéssel a fájlméret és a sávszélesség pontos beállításának lehetőségét is elvesztettük. Azonban lehetőség van mérlegelni, melyik a fontosabb: a gyorsaság vagy a minőség. Szerencsére az ***x264*** kínál megfelelő középutat is. Egy opcióval beállítható

a fix sávszélesség (vagy fix minőség), így az ***x264*** számításba veszi a mozgalmas és a kevésbé mozgalmas részek közötti különbséget. Minthogy az emberi szem a gyors mozgásoknál nem képes a részletekre figyelni, így a tömörítés során a mozgalmas részeken nyert bitek átcsoportosíthatóak máshova. Így összességében élvezetesebb lesz a végeredmény. Az opció alkalmazásával érhető el a legjobb minőség anélkül, hogy az időrabló többmenetes módot kellene használnunk.
Ennek a megoldásnak azonban az ára, hogy nem tudjuk előre megmondani a végleges fájl méretét és sávszélességét. Természetesen kétmenetes módban ez továbbra is lehetséges, de ezzel megduplázzuk a feldolgozáshoz szükséges időt. A példánkban tehát maradjunk az egymenetes feldolgozásnál, azonban állítsuk be a ***Constant Rate Factor*** (`--crf`) lehetőséget a megfelelő minőségért. ***18*** és ***26*** közötti ***Constant Rate Factor*** érték már általában megfelelő (a kisebb érték jobb minőséget, de nagyobb fájlt eredményez).
A megfelelő érték természetesen függ a kívánt mérettől, minőségtől és a rendelkezésre álló időtől. Többmenetes tömörítés esetén azonban nagyobb szabadságot kapunk.
Az ***x264*** csak nyers ***YUV 4:2:0*** bemenetet fogad, ehhez egyszerűen irányítsuk az ***mencoder*** kimenetét az ***x264*** bemenetére egy egyszerű csővezetékkel (pipe):

```
mkfifo tmp.fifo.yuv
mencoder -vf format=i420
➥-nosound -ovc raw -of
➥rawvideo \
    -ofps 23.976 -o
    ➥tmp.fifo.yuv max.dv
    ➥2>&1 > /dev/null &
x264 -o max-video.mp4 --fps
➥23.976 --crf 26 --progress \
    tmp.fifo.yuv 720x480
rm tmp.fifo.yuv
```

Amint látható, meg kell adnunk, hogy hány képkockát szeretnénk egy másodperc alatt lejátszani (`--fps`), ugyanis az ***x264*** nem fogja magától kitalálni. Ugyanígy meg kell adnunk a videó felbontását is. Jelen példánkban az ***x264*** alapértelmezett beállításaival tömörítettünk, ami elég jó minőséget ad, azonban tovább finomíthatunk rajta. Jelen esetben kicsit módosítva a tömörítési stratégiát, javíthatunk a dolgon anélkül, hogy ez lényegesen több időt venne igénybe. Az ***x264*** rengeteg paraméterrel bír, mellyel tovább javíthatunk a minőségen ilyen vagy olyan módon. Természetesen némely opció alkalmazása költségesebb, időigényesebb, mint másoké. Végül pedig pár opció alkalmazásával elfordulhat az is, hogy bizonyos lejátszók nem fogják tudni lejátszani a videót, nevezetes például a ***QuickTime***.
A kompatibilitás érdekében ezeket tartsuk szem előtt.

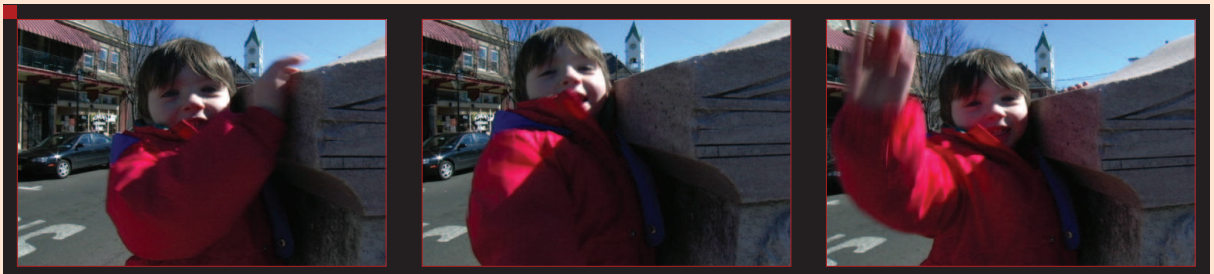
QuickTime és a H.264

A ***QuickTime 7***-es verziója már támogatja a ***H.264***-es videókat. Az ***Apple*** az interneten elérhető mozielőzeteseket például már ***H.264*** szerint tömöríti. Noha ez jó, és elősegíti a kodek terjedését, pár korlát is került a ***QuickTime*** megvalósításába, melyek közül legjelentősebb a ***B-képkockákkal (B-Frames)*** és a ***Profil*** támogatással kapcsolatos. Teszünk egy rövid kitérőt, hogy lássa az Olvasó, mit is jelent ez számunkra.
A ***H.264*** többféle profillal rendelkezik, ilyen a ***Baseline***, a ***Main***, az ***Extended*** és a ***High***. Ezek a profilok leírják, hogy az adott videó lejátszásához milyen képességek szükségesek a lejátszó részéről. Ahogy az sejthető, a ***Baseline*** a legegyszerűbb profil, míg a ***Main***, az ***Extended*** és a ***High*** több számítási kapacitást igényel és technológiailag is bonyolultabb lejátszani. A ***QuickTime 7*** csak a ***Baseline***-t támogatja teljesen, illetve a ***Main***-t részben, az ***Extended***-del és a ***High***-al nem foglalkozik.
A ***B képkocka (B-Frames)*** egy megoldás a videók tárolására. Az ilyen képkockák dekódolásánál létfontosságú a korábbi képkockák hibátlan dekódolása. ***B képkockákat (B-Frames)*** más képkockákkal fésülnek össze (a ***B képkockák*** csupán a változásokat tárolják), ilyenek például az ***I (I-Frames)*** és a ***P képkockák (P-Frames)***. Technikai részlet, de a ***QuickTime 7***-ben alkalmazott ***H.264*** megvalósítás csupán 2 darab ***B képkockát*** enged. Ez nem túl jó, hiszen több ***B képkocka*** alkalmazásával bizonyos körülmények között javítható a képminőség.
Ezt a korlátozást tartsuk észben, hogy ***QuickTime*** kompatibilis maradjon a videónk. Ez a korlátozás természetesen nincs jelentős kihatással a végeredményre. Pár opcióval tovább javíthatunk a dolgon. Ilyen például a pixel elmozdulás becslési pontossága (`--subme`), mellyel beállítható, mennyire legyen pontos pixel elmozdulásának becslése ***x264***-es tömörítés közben. Ezt a 6-os maximumra állítva lényegesen szebb végeredményt kapunk, és noha hosszabb lesz a tömörítés, megéri a többlet idő. A minőség javítása érdekében azt is megadhatjuk, hogy az ***x264*** hogyan elemezze a képkockákat (`--analyse`). Jegyezzük meg: van olyan mód, amely csak ***High*** profil esetén érhető el – ilyen például a ***8x8 DCT*** – , így erről le kell mondanunk a ***QuickTime*** kompatibilitás érdekében. A ***PSNR*** kikapcsolásával gyorsíthatunk a tömörítésen (`--no-psnr`) anélkül, hogy bármi különbség látszana a videónkon.
Mindent összevetve most már mindent tudunk ahhoz, hogy létrehozzunk jó minőségű, kis sávszélességű ***Quicktime*** kompatibilis ***H.264*** kódolású videót:

```
mkfifo tmp.fifo.yuv
mencoder -vf format=i420
➥-nosound -ovc raw -of \
    rawvideo -ofps 23.976 -o
    ➥tmp.fifo.yuv \
    max.dv 2>&1 > /dev/null &
x264 -o max-video.mp4 --fps
➥23.976 --bframes 2 \
    --progress --crf 26
    ➥--subme 6 --analyse \
    p8x8,b8x8,i4x4,p4x4
    ➥--no-psnr tmp.fifo.yuv
    ➥720x480
rm tmp.fifo.yuv
```

Tovább finomíthatjuk a végeredményt. Minthogy a webre szánjuk a videónkat, így például nem hátrány ha kisebbre vesszük a képkockákat és levágjuk a nem kívánt részeket. A képkocka átméretezése például az alábbi paranccsal valósítható meg:

```
mkfifo tmp.fifo.yuv
mencoder -vf scale=480:320,
➥format=i420 -nosound -ovc \
    raw -of rawvideo -ofps
    ➥23.976 -o tmp.fifo.yuv \
    max.dv 2>&1 > /dev/null &
```



1. ábra DV

2. ábra XviD

3. ábra x264

1. táblázat *Eredmények*

Fájl	Fájl méret	Sávszélesség
max.dv	*32Mbyte*	*3MByte/s*
max-xvid.avi	*623Kbyte*	*418kbit/s*
max-x264.mp4	*522Kbyte*	*392kbit/s*

```
x264 -o max-video.mp4 --fps
➥23.976 --bframes 2 \
    --progress --crf 26
    ➥--subme 6 --analyse \
    p8x8,b8x8,i4x4,p4x4
➥--no-psnr tmp.fifo.yuv
➥480x320
rm tmp.fifo.yuv
```

Az *mencoder* 480x320 képpont felbontású videót készít, így az *x264*-nek is meg kell adnunk a helyes méretet. Ez a művelet csökkentette a videó méretét, mely a gyorsabb letöltés miatt előny.

Utolsó lépések

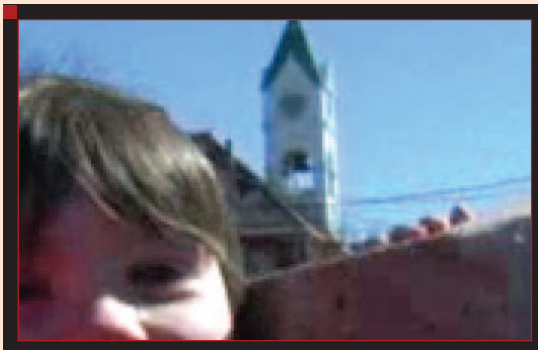
Az *.mp4* formátum (melyhez alapul a *QuickTime* szolgált) többféle médiaformátumot (hang, videó) tárolhat, így ideális számunkra a *H.264*-es videó és *AAC* hang tárolására is. Az *MP4Box* segítségével – mely a *gpac* projekt része – a korábbiakban elkészített két állományt egyesíthetjük:

```
MP4Box -add max-video.mp4 -add
➥audiodump.aac \
   -fps 23.976 max-x264.mp4
```

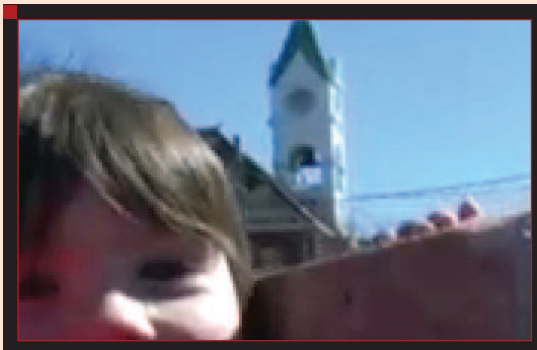
Létrehoztuk a végleges `max-x264.mp4` állományt, melyet le is játszhatunk *Mplayer*rel, vagy *nem linuxos* gépeken az *Apple QuickTime* lejátszójával. A videónk weboldalba is beilleszthető. Ezt például *Linux* alatt *Firefox*ból a telepített *mplayer-plugin* segítségével tekinthetjük meg. Összehasonlítás végett az állományok méretei és sávszélességei nyers *DV*, a tárgyalt *H.264* és az alábbi parancssor generálta *XviD* szerint:

```
mencoder max.dv -vf
➥scale=480:320 -ovc xvid
➥-xvidencopts \
   fixed_quant=7:qpel:nopacked
   ➥-oac mp3lame \
   -ofps 24000/1001 -o max-
   ➥xvid.avi
```

És minden videóból egy-egy képkocka: *1-3. ábra.*



4. ábra XviD részletek



5. ábra x264 részletek

Ahogy látható, a ***H.264*** éppolyan jó minőséget nyújt, mint az ***XviD*** (talán jobbat is), de mindezt kisebb sávszélesség igény és méret mellett. Belátható, hogy ugyanolyan minőséghez kisebb tárhely is elég, vagy ugyanakkora tárhelyen jobb minőséget érhetünk el. Emellett a munkamenet és a beállítások hasonlóak az ***XviD***-hez, viszont jobb minőséget ad a ***H.264***. Tehát ha már tömörített az Olvasó ***XviD***-del, az ***x264*** se fog meglepetést okozni.
Minél többet kísérletezik az ember az ***x264***-el, annál jobban elcsodálkozik, mekkora megtakarítás érhető el a minőség megtartása mellett. A videó tömörítés kétségkívül egyfajta varázslat, hiszen több száz opció sokféle értéke befolyásolja a végeredményt. A videótömörítéshez nincs általánosan használható eljárás vagy ajánlott paraméterek, minden videó más és más. A ***H.264*** technológiai fölénye az ***XviD***-el vagy ***MPEG2***-vel szemben elég jelentős ahhoz, hogy figyelmen kívül hagyjuk. Már ma élvezhetjük az előnyeit. Tekintve, hogy a ***H.264*** kodek ***MPEG*** szabvány, így a befektetett munka végeredménye időtálló és jó minőségű lesz. Jelen cikk talán segített rávilágítani arra, miért lesz a jövő tömörítési formátuma a ***H.264***.

Linux Journal 2006., 150. szám

Dave Berton profi programozó, a mosey@freeshell.org címen érhető el.

KAPCSOLÓDÓ CÍMEK

A cikk forrása:
➲ www.linuxjournal.com/article/9197