Foundations of Multimedia Technologies

Dr. Firtha Gergely

May 16, 2020

Contents

Chapter 1

Representation of colors pixels

Along with the most important properties of human vision, the previous chapter discussed the basics of photometry and colorimetry with introducing the concepts of luminance and the CIE XYZ color space. The present chapter introduces the practical color representation applied in analog and digital video technologies based on the foregoing.

In video applications the direct XYZ representation of color points is scarcely ever used, except for several digital cinema and film archiving applications[1]. However, as the real importance of the XYZ colors space, it allows the investigation and mathematical description of the colors, reproducible in specific cameras and displays, and allows the color conversion between imaging devices. The following section investigates these device-specific color representations, i.e. the **device dependent color spaces**.

1.1 Device-dependent color spaces

Due to the trichromaticity and near-linearity of the human vision the visible colors can be represented in a 3D vector space, allowing the addition of color vectors: The perceived color generated by the mixture of two arbitrary colors can be obtained the addition of the two colors' position vectors (independently of the physical, spectral properties of the involved light rays). On the xy chromaticity diagram the mixture is located along the straight line, connecting the original two color points.

As a consequence, relying on the metamerism of human vision most of the visible colors can be reproduced as the combination of appropriately chosen **primary colors** (or simply **primaries**). Generally speaking this is the basic idea behind color reproduction. Obviously, the reproduction of all the visible colors is not a realistic goal when using a limited number of primaries: in the chromaticity diagram the visible colors are bounded by the continuous, smooth curve of spectral colors, meaning that reproduction all the visible colors—to cover all the color points as a point on the line, connecting

[1] One reason for this is that the XYZ coordinate system can not be used directly for color reproduction, with the X,Y,Z primaries not being real visible colors. On the other hand the digital representation of all the visible colors requires high bit depth unnecessarily, since the set of visible colors only fills a small part of the positive XYZ octant (many digital codes would be assigned to non-visible colors)

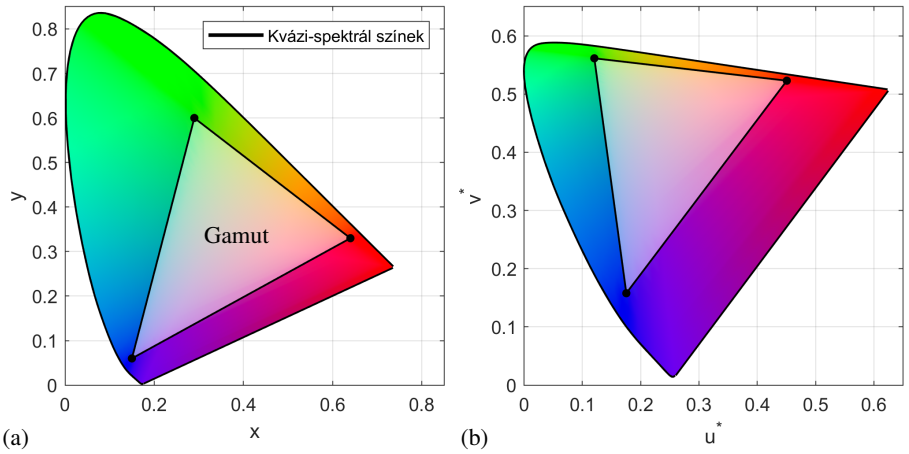


Figure 1.1: Gamut of the SD, HD and sRGB color spaces, all applying the same RGB primaries, illustrated on the xy (a) and the u^*v^* diagram.

two primary colors—theoretically, infinite number of primary colors would be needed, realizing all the spectral colors as primaries. This rises the question, what is the feasible number of primaries used for color reproduction.

In the xy diagram the area of reproducible colors is defined by a polygon with the apex representing the primaries. Most of the xy chromaticity diagram can be covered by a rectangle, meaning 4 primaries allows the reproduction of nearly all the visible colors. On the other hand the $L^*u^*v^*$ chromaticity diagram reflected that the resolution of human vision for shades of green color is low and the perceptually uniform chormaticity diagram resembles to a triangle. Therefore, imaging systems based on additive color mixing almost exclusively apply three properly chosen red, green and blue primary colors, allowing to cover most of the perceptually uniform chromaticity diagram.

The set of colors that can be reproduced with positive primary coordinates (i.e. positive RGB intensities) is termed as a **device dependent color space**, while colorimetric, absolute color spaces like CIE XYZ, $L^*u^*v^*$, $L^*a^*b^*$ are referred to as **device independent color spaces**. The set of the reproducible colors in a device dependent RGB color space are located on the edges and inside a triangle on the xy chromaticy diagram. This triangle is referred to as the **gamut** of the color space. A simple example for the gamut of RGB color spaces is illustrated in Figure 1.6[2]. Colors, located on the apexes and edges of the gamut are the most saturated colors in the RGB color space, being the closest color to spectral colors. These color are termed the **quasi-spectral colors**, with the common property that they contain a maximum of two primaries.

Once an RGB color space is well-defined, for an arbitrary color C the RGB primary

[2]Obviously, device dependent color spaces exist applying different numbers of primaries. A commonly used device dependent space is the CMYK color space of color printing, with the four primaries denoting the color of the inks, used for reproduction. In the following the investigation is limited to RGB color spaces.

intensities can be specified so that their additive mixture coincides with color C (as long as the intensities are positive). These RGB intensities $\mathbf{c}_{RGB} = \begin{bmatrix} R_c \\ G_c \\ B_c \end{bmatrix}$ are referred to as the **RGB coordinates** of color C, describing the position of the color point in the RGB color space. By definition, the RGB coordinates may take values between 0 and 1, therefore, vectors

$$\mathbf{r}_{RGB} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{g}_{RGB} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{b}_{RGB} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{1.1}$$

denote the red, green and blue primary colors with $100\ \%$ intensities.

The following section discusses the practical definition of RGB color spaces.

1.1.1 Definition of device-dependent color spaces

Assume a device dependent color space applying RGB primaries. The three primary colors (red, green, blue) can be represented as three vectors in the XYZ pointing into the actual perceived color of the primaries. The chromaticities (xy coordinates) of the primaries are found at the intersection of the primary vectors and the unit plane. The constellation is illustrated in Figure 1.2. The XYZ-coordinates of the RGB primaries are denoted by

$$\mathbf{r}_{XYZ} = \begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix}, \quad \mathbf{g}_{XYZ} = \begin{bmatrix} X_g \\ Y_g \\ Z_g \end{bmatrix}, \quad \mathbf{b}_{XYZ} = \begin{bmatrix} X_b \\ Y_b \\ Z_b \end{bmatrix}. \tag{1.2}$$

As long as the XYZ-coordinates of the RGB primaries are known, the RGB color space is fully defined: The RGB primary intensities $\mathbf{c}_{RGB}$ can be defined for an arbitrary color point described by its XYZ coordinates $\mathbf{c}_{XYZ}$. The RGB coordinates $\mathbf{c}_{RGB}$, giving the weight of the RGB primaries in the given color point read as

$$\underbrace{\begin{bmatrix} R_c \\ G_c \\ B_c \end{bmatrix}}_{\mathbf{c}_{RGB}} = \mathbf{M}_{XYZ \to RGB} \underbrace{\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}}_{\mathbf{c}_{XYZ}}, \tag{1.3}$$

where $\mathbf{M}_{XYZ \to RGB}$ is a linear transform matrix. Vice versa, if a color point is given by its RGB coordinates, the XYZ vector coordinates can be defined by

$$\underbrace{\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}}_{\mathbf{c}_{XYZ}} = \mathbf{M}_{RGB \to XYZ} \underbrace{\begin{bmatrix} R_c \\ G_c \\ B_c \end{bmatrix}}_{\mathbf{c}_{RGB}}. \tag{1.4}$$

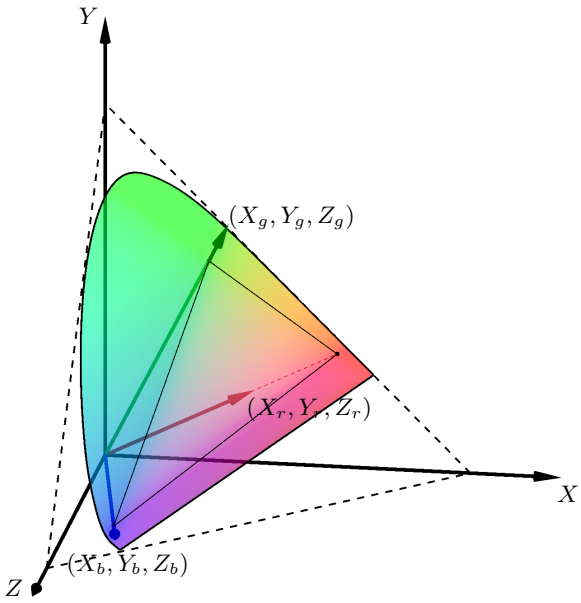


Figure 1.2: RGB primaries and their chromaticities, illustrated in the XYZ color space.

Obviously, for the involved transform matrices $\mathbf{M}_{RGB\to XYZ} = \mathbf{M}^{-1}_{XYZ\to RGB}$ holds.

The latter transform matrix can be simply defined by utilizing the rule for change of basis from elementary linear algebra: The columns of the transform matrix $\mathbf{M}_{RGB\to XYZ}$ are simply containing the bases of the RGB color space in the XYZ space, i.e. the XYZ-coordinates of the RGB primaries[3]

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \underbrace{\left[\begin{array}{c|c|c} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{array}\right]}_{\mathbf{M}_{RGB\to XYZ}} \cdot \begin{bmatrix} R_c \\ G_c \\ B_c \end{bmatrix}. \tag{1.5}$$

These transform matrices allow the conversion between RGB color spaces of specific devices. On the other hand coordinate Y_c of and arbitrary color point gives the luminance, being proportional to the perceived brightness of the given color. Hence, the second row of the transform matrix $\mathbf{M}_{RGB\to XYZ}$ defines how the luminance content can be calculated of an arbitrary color, given in the RGB space.

The set of all reproducible colors in an RGB color space is illustrated in Figure 1.3. Clearly, the set of all vectors that can be written as the linear combination of three primary vectors with positive weights distends a parallelepiped, i.e. an RGB color space can be illustrated in the XYZ coordinate system by a parallelepiped.

[3]The validity of the formula can be simply admitted by substituting e.g. $\mathbf{c}_{RGB} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, describing the red primary in the RGB color space, and substitution to (1.5) returns the first column of the transform matrix.

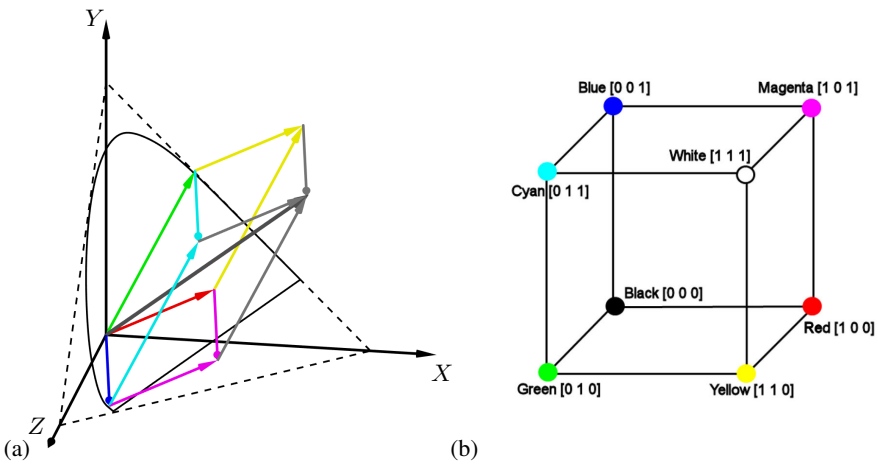


Figure 1.3: Illustration of an RGB color space in the XYZ coordinate system (a) and in the RGB cube (b). The color of the vectors in (a) denotes the color in the endpoints.

Obviously, since the RGB coefficients can take values between 0 and 1, the set of all reproducible colors in the RGB coordinate system fills the unit cube in the positive space octant.[4], with the edges passing through the origin containing the RGB primary colors with different intensities. Therefore, the above transform matrices perform linear transforms, transforming a cube into a parallelepiped and a parallelepiped into a cube.

A relatív fénysűrűség bevezetése:

Egy RGB színtér tehát teljes egészében adott, amennyiben az alapszín-vektorok XYZ koordinátái ismertek. A gyakorlatban azonban egy RGB színtér definiálása során az XYZ koordináták helyett az RGB alapszínek és a fehérpontjának színezetét, azaz xy színkoordinátáit adják meg. Definíció szerint egy adott színtér **fehérpontja** az adott térben elérhető legvilágosabb pont, amelyet az alapszínek egyenlő arányú keverékével érhetünk el. Az adott eszközfüggő színtérben a 100%-os ez alapján (hasonlóan az XYZ-beli fehérhez), definíció szerint

$$\mathbf{w}_{RGB} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \text{és} \quad Y_w = 1, \tag{1.6}$$

ahol Y_w a színpont **relatív fénysűrűsége**, amely tehát 0 és 1 között vehet fel értékeket. A 1.2 ábrán látható példában a fehér szín vektora a paralelepipedon szürkével jelölt főátlója, ezen vonal mentén helyezkednek el a különböző világosságértékű (árnyalatú) fehér színek. A fehér szín színezete, azaz x_w és y_w koordinátái ezen vektor az egységsíkkal vett döféspontja határozza meg.

[4]Hence RGB color spaces are sometimes referred to as RGB cube

A színteret tehát úgy definiáljuk, hogy a három alapszínvektor xy koordinátája (azaz az iránya) mellett megadjuk az alapszínek egyenlő energiájú keverékének a színezetét, (azaz a három bázisvektor összegének irányát), és rögzítjük, hogy az összegvektor Y koordinátája egységnyi. Ebből a 9 adatból meghatározhatók az RGB bázisvektorok tényleges hossza, és így a szükséges transzformációs mátrixok felírhatók.

Az RGB színterek ilyen módú definíciója mögött a motíváció a következő: Láthattuk, hogy az XYZ koordináták a színérzetet létrehozó spektrummal szorosan összefüggnek, az Y koordináta pl. a fényinger fénysűrűségét adja meg ($[\mathrm{cd}/\mathrm{m}^2]$-ben, vagy nit-ben). A gyakorlati alkalmazások során azonban nem szempont egy RGB színtér alapszíneinek—pl. egy RGB kijelző LCD alapszíneinek—fizikai jellemzőinek pontos ismerete (azaz pl. hány nit fénysűrűséget hoz létre az R, G, vagy B pixel-elem). Ennek oka, hogy képi reprodukció során a tényleges, fotometriai abszolút fénysűrűséget szinte soha nem célunk visszaállítani (nem is tudnánk, ha a képernyő maximális létrehozható fénysűrűsége kisebb, mint az eredeti mért fénysűrűség). Ehelyett az adott megjelenítő eszköz által létrehozható legvilágosabb színhez képest reprodukáljuk az adott képpontok relatív fénysűrűségét. Az, hogy ez a legvilágosabb pont ténylegesen hány nit fénysűrűséget hoz létre eszközről eszközre változhat, és a megjelenítők fontos paramétere (ez az általában $[\mathrm{cd}/\mathrm{m}^2]$-ben megadott maximális fényerő paraméter). Az eszközfüggő színterek fenti definíciója tehát azt biztosítja, hogy az Y koordináta az RGB alapszínek fizikai jellemzőitől függetlenül a relatív fénysűrűséget írja le.

A fehér színről általában:
Látható tehát, hogy a fehér szín önmagában szubjektív fogalom: adott környezetben a leginkább akromatikus fényingert nevezzük fehérnek, amelynek spektrális sűrűségfüggvénye minél inkább egységnyi (azaz minél több spektrális komponenst tartalmaz), és ezzel analóg módon RGB színtérben ábrázolva minél közelebb van a csupa-egy vektorhoz. A fehér fogalom egységesítéséhez bevezettek ún. szabványos megvilágításokat (standard illuminants), amelyet szabványosított RGB színterek esetén előírnak, mint fehérpont. Ezeknek a szabványos megvilágításoknak a spektrális sűrűségfüggvénye (és persze az általa keltett színinger xy-koordinátái) adott, jól-definiált. Ilyen szabványos megvilágítások a következők:

- E fehér: egyenlő energiájú fehér, a CIE XYZ színtér fehérpontja. Kolorimetria szempontjából jelentős, videótechnikában kevésbé fontos a szerepe, mivel a gyakorlatban nem fordul elő olyan fényforrás, amely minden hullámhosszon azonos energiával sugároz.

- A fehér: a CIE által szabványosított, egyszerű háztartási wolfram-szálas izzó fényét (azzal azonos színérzetet keltő) fényforrás spektruma és színe, $T_\mathrm{C} = 2856\ \mathrm{K}$ korrelált színhőmérséklettel [5].

[5] A korrelált színhőmérséklet (correlated color temperature, CCT, T_C) azon feketetest sugárzó hőmérsékletét jelzi, amely az emberi szemben a minősítendő fényforrással azonos színérzetet kelt. A feketetest (hőmérsékleti) sugárzó által keltett színingerek az xy színdiagramon az ún. Planck-görbét járják be, amely a 1.4 ábrán látható.

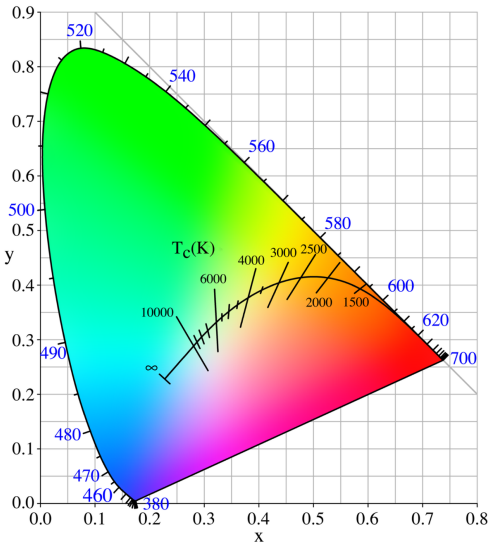


Figure 1.4: Különböző hőmérsékletű feketetest sugárzók által keltett színek összessége, azaz a Planck görbe.

- B és C fehér: Az A fehérből egyszerű szűréssel nyerhető, napfényt szimuláló megvilágítások. A B fehér a déli napsütést modellezi 4874 K színhőmérséklettel, míg a C fehér a teljes napra vett átlagos fény színét (spektrumát) modellezi 6774 K színhőmérséklettel.

- D fehér: szintén a napfény közelítésére alkalmazott megvilágítások sora. Videótechnika szempontjából a legfontosabb a D65 fehér, amely jelenleg is az UHD formátumok színterének szabványos fehérpontja.

1.1.2 Color space conversions

Az eddigiekben látható volt, hogyan definiálható egy eszközfüggő színtér az alapszíneivel. Ahogy az elnevezés is mutatja, ezek a színterek jellegzetesen adott eszközre érvényesek, pl. egy kamera a beépített RGB szenzorok, egy kijelző az alkalmazott RGB kristályok által meghatározott RGB színtérben dolgoznak. Emellett léteznek szabványos RGB színterek amelyek a képi tartalom tárolására, továbbítására szolgálnak egységesített, szabványos módon. A következő szakasz ezeket a szabványos videószíntereket tárgyalja részletesebben. Felmerül tehát a természetes igény az egyes színterek közti átjárásra, amelyet **színtér konverziónak** nevezünk.

A színtérkonverziót az XYZ színtér teszi lehetővé, amely egy eszközfüggetlen, abszolút színtér: egyes színterek közti konverzió a forrás által létrehozott jelek XYZ színtérbe való transzformációjával, majd ezen reprezentáció a nyelő színterébe való transzformációval történik. Az XYZ színtér így tehát színterek közti átjárást biztosít, ún. Profile Connection Space-ként működik (hasonlóan pl. a gyakran azonos célra alkalmazott Lab színtérhez).

Egy tipikus színtér konverziós folyamatot az 1.5 ábra mutat. Tegyük fel, hogy adott egy HD kamera által rögzített képanyag, ahol a kamera színterét RGB_{cam} jelöli. A

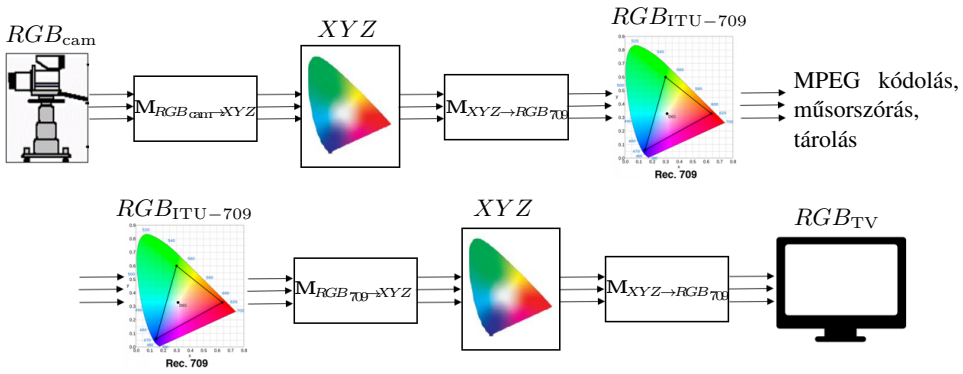


Figure 1.5: Színtér-konverzió folyamatábrája.

HD formátum szabványos színteret alkalmaz, amelyet az ITU-709 ajánlásban rögzítettek (lásd később). A kamera RGB jeleit tehát az esetleges kódolás és tárolás előtt ebbe a HD színtérbe kell konvertálni. Ez a konverzió a kamerajelek XYZ térbe, majd innen az ITU-709 színtérbe való transzformációval oldható meg, amely a megfelelő transzformációs-mátrixszal való szorzással valósítható meg:

$$\begin{bmatrix} R_{\text{ITU}-709} \\ G_{\text{ITU}-709} \\ B_{\text{ITU}-709} \end{bmatrix} = \mathbf{M}_{XYZ\to RGB_{709}} \cdot \left(\mathbf{M}_{RGB_{\text{cam}}\to XYZ} \cdot \begin{bmatrix} R_{\text{cam}} \\ G_{\text{cam}} \\ B_{\text{cam}} \end{bmatrix} \right) \tag{1.7}$$

Természetesen az egymás utáni két mátrixszorzás összevonható, így a két RGB színtér között közvetlen lineáris leképzés határozható meg. Ez a transzformáció jellegzetesen már a kamerán belül megvalósul. Hasonlóképp, megjelenítőoldalon a

$$\begin{bmatrix} R_{\text{cam}} \\ G_{\text{cam}} \\ B_{\text{cam}} \end{bmatrix} = \mathbf{M}_{XYZ\to RGB_{\text{TV}}} \cdot \left(\mathbf{M}_{RGB_{709}\to XYZ} \cdot \begin{bmatrix} R_{\text{ITU}-709} \\ G_{\text{ITU}-709} \\ B_{\text{ITU}-709} \end{bmatrix} \right) \tag{1.8}$$

transzformációt kell elvégezni.

Ez az egyszerű transzformációs módszer lehetővé teszi egy adott színtérben mért színpontok másik színtérben való ábrázolását. Ugyanakkor felmerül a probléma, hogy a nagyobb gamuttal rendelkező színtérből kisebbe való áttérés esetén az új színtérben nem ábrázolható, gamuton kívüli színek negatív és egynél nagyobb RGB koordinátákkal jelennek meg, míg a kisebb gamutú térből való áttérés esetén a nagyobb gamutú tér egy része kihasználatlan marad. A probléma megoldására a fenti transzformációk mellett az egyes színterek gamutját valamilyen nemlineáris leképzés segítségével lehet egymásra illeszteni (expandálással, kompresszálással). Ezek az ún. gamut-mapping technikák.

A következőekben az egyes SD, HD és UHD videóformátumok tárolására és továbbítására alkalmazott eszközfüggő színtereket tárgyaljuk.

1.1.3 Color spaces of video technology

Az NTSC színmérőrendszere:

Az első kodifikált színmérő rendszer az NTSC (National Television System Committee) által 1953-ban szabványosított színes-televíziós műsorszóráshoz alkalmazott NTSC szabvány volt. A színteret a korabeli foszfortechnológiával létrehozható CRT kijelzők (TV vevők) alapszíneik megfelelően írták elő, így színtérkorrekció vevő oldalon nem volt szükség. A színmérő rendszer C fehérponttal dolgozott, alapszíneit pedig a 1.1 táblázat mutatja. Az így kapott gamut az 1.6 ábrán látható. Az alapszínekből és a

Table 1.1: Az NTSC szabvány színmérőrendszere

	x	y
R	0.67	0.33
G	0.21	0.71
B	0.14	0.08
C fehér	0.310	0.316

fehérpontból meghatározható az $RGB_{\mathrm{NTSC}} \rightarrow XYZ$ transzformációs mátrix, amely alakja általánosan

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.60 & 0.17 & 0.2 \\ 0.30 & 0.59 & 0.11 \\ 0 & 0.07 & 1.11 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{\mathrm{NTSC}} \tag{1.9}$$

Az egyenlet második sora kitüntetett szereppel bír: meghatározza, hogy az NTSC színtérben hogyan számítható adott RGB színpont relatív fénysűrűsége (világossága):

$$Y_{\mathrm{NTSC}} = 0.30R + 0.59G + 0.11B. \tag{1.10}$$

A világosságjel számítása egészen a HD formátum megjelenése (azaz közel 50 éven keresztül) a fenti egyenlet szerint történt.

Az foszfortechnológia fejlődésével az újabb megjelenítők egyre inkább feláldozták a széles gamutot (azaz a minél telítettebb alapszínek használatát) a minél nagyobb fényerő érdekében: Az alkalmazott foszforok a nagyobb érzékelt világosság (fénysűrűség) érdekében egyre nagyobb sávszélességben sugároztak, így az alapszínek egyre kevésbé telítettek lettek, a gamut tehát csökkent (más szóval: az alapszínek spektruma a Dirac-impulzus helyett—amely teljesen telített spektrálszín lenne—szélesebb görbe lett, így a görbe alatti terület—és ezzel a szín világossága nőtt—de telítettsége csökkent).

Mivel így a megjelenítő gamutja jelentősen eltért az NTSC szabványtól, ezért ez a képernyőn látható színek torzulását eredményezte. Ennek megoldásául a TV vevőkbe analóg színtérkonverziós áramköröket ültettek, amelyek az NTSC és a megjelenítő saját színtere közti konverziót valósította meg[6]. Ettől a ponttól tehát a műsorszórás szabványos színtere és a megjelenítők színtere különváltak.

[6] Ahogy látni fogjuk a későbbiekben: a vevőkbe már csak a nem-lineárisan Gamma-előtorzított RGB

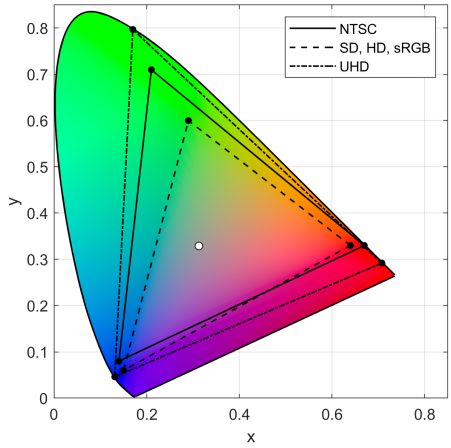


Figure 1.6: Az NTSC, PAL/SD/HD/sRGB és UHD szabványok gamutja az xy-színpatkóban. Az NTSC jóval nagyobb gamuttal dolgozott, mint a ma is használt HD és sRGB formátumok. Ennek oka, hogy a korai CRT megjelenítők ugyan telítettebb, de ugyanakkor kisebb fénysűrűségű és nagy időállandójú foszforokkal dolgoztak, amivel bár nagy színtartományt tudtak megjeleníteni, de kis fényerővel, és mozgó objektumoknál a képernyőn akaratlanul is nyomokat hagyva.

A PAL és az SD színmérőrendszere:

Az európai színes műsorszórás bevezetéséhez az EBU (European Broadcasting Union) 1963-ban szabványosította a PAL (Phase Alternating Line) rendszert, újradefiniálva a színmérőrendszert, új alapszíneket és D65 fehéret alkalmazva: Ez

Table 1.2: A PAL szabvány színmérőrendszere

	x	y
R	0.64	0.33
G	0.29	0.60
B	0.15	0.06
D65 fehér	0.3127	0.3290

matematikailag helyesen a transzformációs mátrix és a világosságjel számításának módjának megváltozását jelentené. Praktikussági szempontokból azonban a PAL rendszer az NTSC-vel azonos módon, (1.18) alapján állítja elő a világosságjelet, mivel a

jelek jutottak, ahol az inverz torzítást maga a kijelző hajtotta végre. Emiatt a színtérkonverziót csak Gamma-torzított $R'G'B'$ jeleken tudták végrehajtani, ami azonban a telített színeknél ismét látható színezet és fénysűrűség-hibát okozott.

gyakorlatban a különbség alig volt látható [7]. Az PAL alapszíneit és a világosságjel számításának módját átvette az első digitális videóformátum, az ITU (International Telecommunication Union) által szabványosított ITU-601-es SD formátum is 1982-ben.

A HD és UHD formátumok színmérőrendszere:

A HD formátumot az 1990-ben szabványosították az ITU-709-es ajánlás formájában. Az ajánlás átvette az PAL rendszer alapszíneit, azonban immáron matematikailag precízen, újraszámította a transzformációs mátrixot és a világosságjel együtthatókat, amely tehát HD esetén

$$Y_{\mathrm{ITU}-709} = 0.2126\,R + 0.7152\,G + 0.0722\,B. \tag{1.11}$$

alapján számítható. Fontos megjegyezni, hogy az ITU-709 szabvány színmérőrendszerét átvette az sRGB szabvány is, ami a mai napig a számítógépes alkalmazások (és operációs rendszerek) alapértelmezett színteréül szolgál.

Az alkalmazott alapszíneket végül számottevően csak az UHD formátum változtatta meg az ITU-2020 számú ajánlásában 2012-ben. Az UHD alkalmazásokra a szabvány egy széles gamutú, spektrál-alapszíneket alkalmazó színteret ajánl a 1.3 táblázatban látható paraméterekkel. A szabvány természetesen újradefiniálta a világosság kom-

Table 1.3: Az ITU-2020 szabvány színmérőrendszere

	x	y
R	0.708	0.292
G	0.17	0.797
B	0.131	0.046
D65 fehér	0.3127	0.3290

ponens számításának a módját is, amely tehát UHD esetben

$$Y_{\mathrm{ITU}-2020} = 0.2627\,R + 0.678\,G + 0.0593\,B \tag{1.12}$$

alapján számítható. A szabvány természetesen nem igényli, hogy az UHD megjelenítők spektrálszíneket legyenek képesek alapszínekként realizálni, a minél szélesebb gamut inkább a jövőbeli technológiák szempontjából ad ajánlást. A mai konzumer megjelenítők az UHD képanyagot megjelenítés előtt a saját színterükben konvertálják, amely jellegzetesen jóval kisebb a szabvány színterénél.

1.1.4 Example for device-dependent color space

Egyszerű példaként az eddig leírtakra vizsgáljuk, hogyan számítható és illusztrálható egy CRT kijelző által megjelenített színek tartománya, röviden rávilágítva a CRT tech-

[7] Ennek oka, hogy a világosságjel átviteltechnológia szempontjából fontos: a kamera és a kijelző is RGB jeleket használ, a világosságjelet, ahogy a következőekben látjuk csak a képanyag átviteléhez számítjuk ki.

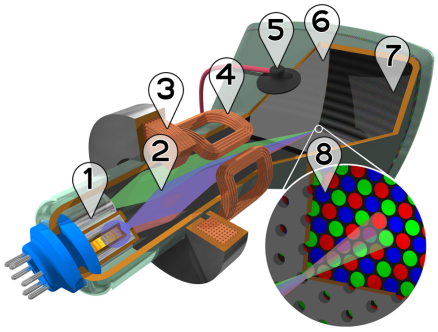


Figure 1.7: CRT megjelenítő felépítése.

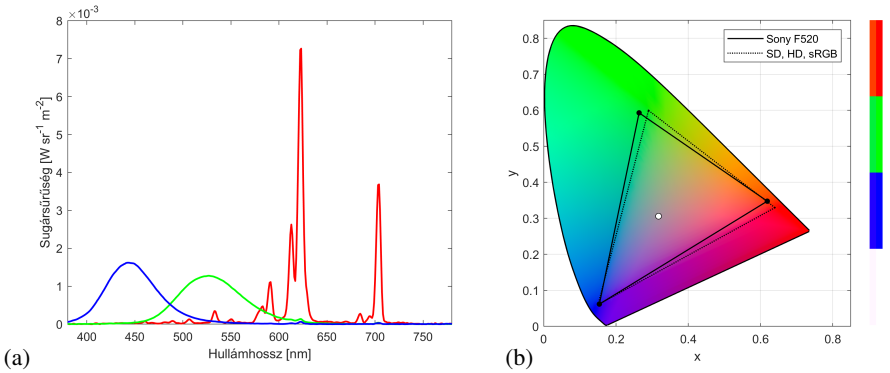
nológia működési elvére is [8]. Bár a CRT technológia kezd egyre inkább eltűnni, néhány évvel ezelőttig a stúdiómonitorok jelentős része még mindig CRT alapon működött köszönhetően a színhű megjelenítésüknek, és a mai LCD megjelenítőkhöz képest is jóval nagyobb statikus kontrasztjuknak.

A katódsugárcsöves (CRT) kijelzők sematikus ábrája az 1.7 ábrán látható. A CRT-k kijelzők működésének alapja három ún. elektronágyú volt, amelyek egy fűtött katódból (1) és egy nagyfeszültségre helyezett anódból állt. A melegítés hatására a katód környezetébe szabad elektronok léptek ki, így egy elektronfelhőt képezve a katód körül. A katód közelébe helyezett nagyfeszültségű (néhány száz Volt) gyorsítóanód hatására a szabad elektronok az anód felé kezdtek mozogni, egy szabad elektronáramot (2) indítva a vákuumban (ugyanezen az elven működtek a vákuum-diódák, triódák, pentódák, stb. is). Elegendően nagy anódfeszültség (és további anódok jelenléte) esetén az elektronok jelentős része nem csapódott be a gyorsítóanódra, hanem továbbhaladt. Ezt az elektronnyalábot elektrosztatikusan és mágnesesen (3) fókuszálták, majd egy vezérelt mágneses eltérítő (4) sorról sorra végigfuttatta azt egy anódfeszültségű-ernyőn (5), azaz a képernyőn. Színes kijelző esetén természetesen három elektronágyú üzemelt párhuzamosan. A képernyő felszínét pixelekre bontva képpontonként három különböző foszforral borították (7-8), amely gerjesztés (becsapódó elektronok) hatására bizonyos ideig adott spektrális sűrűségfüggvényű fényt bocsájtott ki[9], realizálva ezzel az RGB alapszíneket.

Tekintsünk példaként egy Sony F520 CRT kijelzőt: A kijelző RGB foszforjai gerjesztés hatására a 1.8 (a) ábrán látható spektrális sűrűségfüggvényű (sugársűrűségű) fényt bocsájtanak ki egységnyi felületről, egységnyi térszögbe, azaz rendelkezésre állnak a mért $L_e^R(\lambda)$, $L_e^G(\lambda)$ és $L_e^B(\lambda)$ függvények. Fejezzük ki ezek segítségével a kijelző működéséhez szükséges RGB vezérlőjeleket, illetve vizsgáljuk a megjeleníthető

[8] Természetesen az itt leírtak változtatás nélkül alkalmazhatók más technológia alapján működő kijelzőkre is, pl. LCD.

[9] Ellentétben a fluoreszkáló anyagok csak a gerjesztés fennállásának idején bocsájtanak ki fényt. A foszforeszkálás időállandója előnyös, hiszen megfelelően megválasztott foszforok épp egy képidőig bocsájtanak ki fényt, így a kijelzett kép nem fog villogni. Ugyanakkor a korai kijelzők ezen időállandója túl nagy volt, ezért a gyors mozgások elmosódtak a kijelzett képen.



(a) (b)

Figure 1.8: CRT megjelenítő foszforai által kibocsájtott sugárzás spektrális sűrűségfüggvénye (a) a megjelenítő gamutja és az adott spektrumok/alapszínek által keltett színérzet, valamint a színtér fehérpontja (b). A jobb oldali oszlop bal fele a Sony monitor alapszíneit és fehérpontját, a jobb fele az sRGB színtér alapszíneit és fehérpontját szemlélteti.

színek tartományát!

A $\overline{x}(\lambda)$, $\overline{y}(\lambda)$, $\overline{z}(\lambda)$ szabványos XYZ spektrális színösszetevő függvények alkalmazásával a piros (és persze a zöld és kék) alapszín abszolút XYZ színkoordinátái rendre a

$$\begin{aligned}
\overline{X}_R &= K_m \int_{380\text{ nm}}^{780\text{ nm}} L_e^R(\lambda)\cdot\overline{x}(\lambda)\mathrm{d}\lambda = 45.3, \qquad \overline{X}_G = 21.4, \qquad \overline{X}_B = 16.6 \\
\overline{Y}_R &= K_m \int_{380\text{ nm}}^{780\text{ nm}} L_e^R(\lambda)\cdot\overline{y}(\lambda)\mathrm{d}\lambda = 25.5, \qquad \overline{Y}_G = 48, \qquad \overline{Y}_B = 6.7 \\
\overline{Z}_R &= K_m \int_{380\text{ nm}}^{780\text{ nm}} L_e^R(\lambda)\cdot\overline{z}(\lambda)\mathrm{d}\lambda = 2.4, \qquad \overline{Z}_G = 11.6, \qquad \overline{Z}_B = 84.6
\end{aligned} \tag{1.13}$$

integrálok numerikus kiértékelésével számítható, ahol $K_m = 683\ \mathrm{lm/W}$ fényhasznosítási tényező. A színtérben előállítható fehér szín definíció szerint az alapszínvektorok egyenlő súlyú összegeként áll elő, azaz

$$\overline{X}_W = \overline{X}_R + \overline{X}_G + \overline{X}_B, \qquad \overline{Y}_W = \overline{Y}_R + \overline{Y}_G + \overline{Y}_B, \qquad \overline{Z}_W = \overline{Z}_R + \overline{Z}_G + \overline{Z}_B, \tag{1.14}$$

azaz pl. a fehér szín abszolút fénysűrűsége $80.2\ \mathrm{cd/m^2}$. Ez egészen pontosan megegyezik az sRGB szabvány által előírt referenciamonitor fénysűrűségével ($80\ \mathrm{cd/m^2}$).

Természetesen az alapszíneknek nem az abszolút XYZ koordinátái a fontosak, hanem a relatív koordináták, amelyekre teljesül, hogy $Y_W = 1$, és így Y a relatív fénysűrűség. A fenti alapszínvektorok tehát $\overline{Y}_W$ értékével normálandók. Az így kapott relatív alapszínvektorokból már összeállíthatók a színtér alkalmazásához szükséges

transzformáció mátrixok:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \underbrace{\left[\begin{array}{c|c|c} 0.5646 & 0.2665 & 0.2068 \\ 0.3174 & 0.5992 & 0.0834 \\ 0.0302 & 0.1443 & 1.0539 \end{array}\right]}_{\mathbf{M}_{RGB\to XYZ}} \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{\text{F520}} \tag{1.15}$$

$$\mathbf{M}_{XYZ\to RGB} = \mathbf{M}^{-1}_{RGB\to XYZ}$$

Az alapszínek és a fehérpont színezete ezután

$$x_R = \frac{X_R}{X_R + Y_R + Z_R}, \qquad y_R = \frac{Y_R}{X_R + Y_R + Z_R} \tag{1.16}$$

alapján számolható. Az így meghatározott színtér gamutja a 1.8 ábrán látható, az alapértelmezett számítógépes sRGB színtérrel együtt.

Jelen dokumentum sRGB színtérben kerül tárolásra (és megjelenítéskor az sRGB színtér az olvasó kijelzőjének saját színterébe transzformálva), így jelen dokumentumban az XYZ koordinátáival adott alapszínek az sRGB térbe való konverzió után kerülhetnek megjelenítésre (ahogy 1.8 ábrán látható), amely pl. a vörös alapszínre

$$\begin{bmatrix} R_R \\ G_R \\ B_R \end{bmatrix}_{\text{sRGB}} = \mathbf{M}_{XYZ\to RGB_{\text{sRGB}}} \begin{bmatrix} X_R \\ Y_R \\ Z_R \end{bmatrix} = \begin{bmatrix} 1.13 \\ 0.25 \\ -0.02 \end{bmatrix} \tag{1.17}$$

alakú. A Sony megjelenítő alapszíneinek sRGB koordinátáira negatív és 1-nél nagyobb RGB értékek is adódnak. Ez a 1.8 ábrán is látható gamutok közti eltérést tükrözi.

1.2 The Y, $R-Y$, $B-Y$ representation

Az előző szakasz bemutatta egy színes képpont ábrázolásának módját adott RGB eszközfüggő színtérben. Láthattuk, hogy egy színinger leírására a fő érzeti jellemzők a színpont világossága, színezete és telítettsége volt. Felmerül tehát a kérdés, hogy létezik-e hatékonyabb reprezentációja az egyes színpontoknak, ami jobban leírja a fent említett szubjektív jellemzőket, így kevesebb redundáns információt tartalmaz az RGB reprezentációnál.

1.2.1 The color difference signals

A fő oka, hogy a korai TV- és videójeleket nem közvetlenül az RGB jeleknek választották (bár manapság már gyakori a közvetlen RGB ábrázolás) az NTSC bevezetésének idejében a visszafelé kompatibilitás biztosítása volt: A színes műsorszórás kezdetén a korabeli háztartásokban szinte kizárólag fekete-fehér TV-vevők voltak találhatók. Természetes volt az igény a már kiépített fekete-fehér műsorszóró rendszerrel való visszafelé kompatibilitásra színes kép-továbbítás esetén, amelyet a fekete-fehér kép

és a színinformáció külön kezelésével volt elérhető. Természetesen manapság már ez a tradicionális ok nem szempont videójelek megválasztása esetén. Azonban látni a színinformáció külön kezelése lehetővé teszi a színek csökkentett felbontással való tárolását, amely jelentős adattömörítést (analóg esetben sávszélesség-csökkentést) tesz lehetővé.

A fekete-fehér kép egy színes kép világosságinformációjának fogható fel, amely a színpont relatív fénysűrűségével arányos, és így az RGB koordináták lineáris kombinációjaként számítható. Az együtthatók az adott eszközfüggő színtértől függnek, az NTSC alapszínei esetén pl. (1.18) alapján adottak. Ebből kifolyólag színes TV esetén is az változatlanul továbbítandó jelnek a **világosságjelet (luminance)** választották, amely tehát a relatív fénysűrűséggel megegyezik, és így pl. NTSC esetén az RGB jelekből

$$Y_{\mathrm{NTSC}} = 0.30R + 0.59G + 0.11B. \tag{1.18}$$

alapján számítható [10].

Egy színes képpont leírásához 3 komponens szükséges, egy lehetséges és hatékony leírás pl. a képpont világossága, színezete és telítettsége. A világosságjel mellé tehát két független információ kell, amelyek egyértelműen meghatározzák az adott színpont színezetét és telítettségét[11]. Ugyanakkor fontos szempont volt ezen világosságinformáció-mentes, pusztán színinformációt leíró jelek könnyű számíthatósága az RGB komponensekből az egyszerű analóg áramköri megvalósíthatóság érdekében.

A színinformáció/világosságinformáció-szétválasztás legegyszerűbb (de jól működő) megoldásaként egyszerűen vonjuk ki a világosságot az RGB jelekből! Mivel az Y együtthatóinak összege definíció szerint (tetszőleges színtérben) egységnyi, így pl. NTSC esetén (1.18) mindkét oldalából Y-t kivonva igaz a

$$0.30(R-Y) + 0.59(G-Y) + 0.11(B-Y) = 0 \tag{1.19}$$

egyenlőség. Az $(R-Y)$, $(G-Y)$ és $(B-Y)$ a TV-technika ún. **színkülönbségi jelei**, és a következő tulajdonságokkal bírnak:

- Nem függetlenek egymástól, kettőből számítható a harmadik.
- Előjeles mennyiségek.
- Ha két színkülönbségi jel zérus, akkor a harmadik is az. Ekkor $R = G = B = Y$, így tehát a színtér fehérpontjában vagyunk. A fehér színre kapott zérus színkülönbségi jelek azt mutatják, hogy a színinformációt valóban a színkülönbségi jelek jelzik, a fénysűrűség (világosság) pedig tőlük független mennyiség.
- Az adott színkülönbségi jel értéke maximális ha a hozzá tartozó alapszín maximális intenzitású, és vice versa. NTSC rendszerben vörös színkülönbségi jelre

[10] Fontos ismét kihangsúlyozni, hogy a világosság-számítás módja színtérfüggő, az alapszínektől és a fehérponttól függ a már bemutatott módon.

[11] A visszafelé-kompatibilitás biztosításához ezt a két színezetet leíró jelet kellett az NTSC rendszerben a változatlan fekete-fehér jelhez úgy hozzáadni, hogy a meglévő fekete-fehér vevők a világosságjelet demodulálni tudják, és a hozzáadott többletinformáció minimális látható hatással legyen a megjelenített képre.

$R = 1, G = B = 0$ esetén

$$Y = 0.30 \cdot 1 + 0.59 \cdot 0 + 0.11 \cdot 0 \quad \rightarrow \quad R - Y = 0.7, \tag{1.20}$$

és hasonlóan $R = 0, G = B = 1$ esetén

$$Y = 0.30 \cdot 0 + 0.59 \cdot 1 + 0.11 \cdot 1 \quad \rightarrow \quad R - Y = -0.7. \tag{1.21}$$

- A fenti megfontolások alapján a színkülönbségi jelek dinamikatartománya:

$$\begin{gathered} -0.7 \leq R - Y \leq 0.7, \qquad -0.89 \leq G - Y \leq 0.89, \\ -0.41 \leq B - Y \leq 0.41 \end{gathered} \tag{1.22}$$

A három színkülönbségi jelből kettő elegendő a színpont színinformációjának leírásához. Mivel jel/zaj-viszony szempontjából ökölszabályszerűen mindig a nagyobb dinamikatartományú jelet célszerű továbbítani, így a választás a vörös és zöld színkülönbségi jelekre esett.

A videótechnikában tehát egy adott színpont ábrázolása a

$$\begin{gathered} Y : \text{Luminance} \\ \left.\begin{matrix} R-Y \\ B-Y \end{matrix}\right\} : \text{Chrominance} \end{gathered}$$

ún. **luminance-chrominance térben** történik, amely felfogható egy új színmérőrendszernek/színtérnek is az RGB színtérhez képest.

1.2.2 The luminance-chrominance color space

Vizsgáljuk most, hol helyezkednek el az adott RGB eszközfüggő színtérben ábrázolható színek ebben az új, Y, $R - Y$, $B - Y$ térben! Az előzőekben láthattuk, hogy az XYZ térben ez a színhalmaz egy paralelepipedont, az RGB térben egy egységnyi oldalú kockát jelent (lásd 1.2 ábra). Vegyük észre, hogy a Y, $R - Y$, $B - Y$ koordinátákat akár az XYZ, akár az RGB komponensekből egy lineáris transzformációval előállíthatjuk: Jelöljük adott RGB alapszínek esetén a relatív fénysűrűség RGB együtthatóit k_r, k_g, k_b-vel. Ekkor általánosan a színkülönbségi jelek a

$$\begin{bmatrix} Y \\ B-Y \\ R-Y \end{bmatrix} = \begin{bmatrix} k_r & k_g & k_b \\ -k_r & -k_g & 1-k_b \\ 1-k_r & -k_g & -k_b \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1.23}$$

transzformációval számíthatók. Példaképp maradva az NTSC rendszer világosság-együtthatóinál (kiindulva abból, hogy $Y = 0.3R + 0.59G + 0.11B$) a transzformáció alakja

$$\begin{bmatrix} Y \\ B-Y \\ R-Y \end{bmatrix} = \begin{bmatrix} 0.3 & 0.59 & 0.11 \\ -0.3 & -0.59 & 0.89 \\ 0.7 & -0.59 & -0.11 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{\text{NTSC}} . \tag{1.24}$$

A lineáris transzformációt az RGB kockára végrehajtva megkaphatjuk az ábrázolható színek halmazát. Az így kapott test az 1.9 (a) ábrán látható. Láthatjuk, hogy az RGB egységkocka egy paralelepipedonba transzformálódott, ahol a paralelepipedon főátlója az Y világosság tengely. Ennek mentén, az $R-Y = B-Y = 0$ tengelyen helyezkednek el a különböző szürke árnyalatok.

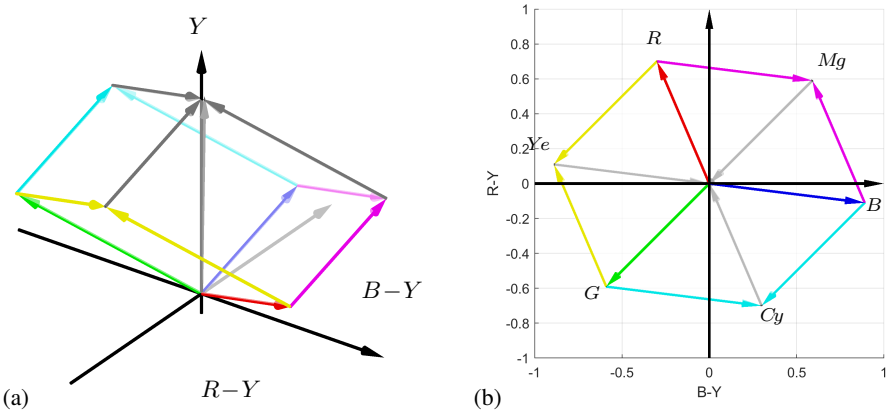


Figure 1.9: Az $Y, R-Y, B-Y$ színtér ábrázolható színeinek halmaza oldalnézetből (a) és felülnézetből (b).

Az eredeti RGB kockához hasonlóan, paralelepipedon főátlón kívüli csúcsaiban (amelyben az $Y = 0$ fekete és az $R = G = B = Y = 1$ fehér található) az eszközfüggő színtér egy, vagy két 100 %-os intenzitású alapszínnel kikeverhető

$$R = \begin{bmatrix}1\\0\\0\end{bmatrix} \quad G = \begin{bmatrix}0\\1\\0\end{bmatrix} \quad B = \begin{bmatrix}0\\0\\1\end{bmatrix} \quad Cy = \begin{bmatrix}0\\1\\1\end{bmatrix} \quad Mg = \begin{bmatrix}1\\1\\0\end{bmatrix} \quad Ye = \begin{bmatrix}1\\1\\0\end{bmatrix} \tag{1.25}$$

vörös, zöld, kék alap- és cián, magenta, sárga ún. komplementer színek találhatók [12].

A paralelepipedonra az Y-tengely irányából ránézve (1.9 (b) ábra) láthatjuk a világosságjeltől függetlenül, adott színtérben kikeverhető színek összességét. Az $R-Y, B-Y, Y$ térben gyakori adott Y világosság mellett a színek ezen $R-Y, B-Y$ síkon való ábrázolása. Minthogy az $R-Y, B-Y$ jelek meghatározzák adott színpont színezetét és telítettségét, így az ábra azt jelzi, hogy a különböző színezetű és telítettségű színek egy szabályos hatszöget töltenek ki. A hatszög csúcsai a színtér alap- és komplementerszínei. Természetesen adott Y érték mellett az ábrázolható színek nem tölti ki teljesen ezt a hatszöget: adott világosságérték mellett az ábrázolható színek halmaza a $Y, R-Y, B-Y$ paralelepipedon egy adott Y magasságban húzott síkkal vett metszeteként képzelhető el, azaz tetszőleges $0 \leq Y \leq 1$ esetén rajzolható egy

[12] Ezen komplementer színek tulajdonsága, hogy az egyes RGB alapszínekkel RGB kockában átlósan helyezkednek el, így a színtérben a lehető legmesszebb elhelyezkedő színpárokat alkotják. Ennek megfelelően egymás mellé vetítve a komplementer színpárok (vörös-cián, sárga-kék, zöld-magenta) váltják ki a legnagyobb érzékelt kontrasztot.

$R-Y, B-Y$ diagram. Az így rajzolható diagramokra példákat a 1.10 ábra mutat. Nyilván rögzített Y mellett nem biztos, hogy minden szín $100\ \%$-os telítettséggel van jelen a $B-Y, R-Y$ diagramon. Például: teljesen telített kékre $Y = 0.11$, azaz a $100\ \%$ intenzitású kék alapszín ezen magasságban vett diagramon található. Más magasságban vett $B-Y, R-Y$ diagramon csak fehérrel higított kék található, azaz nem teljesen telített kék található.

A vizsgált diagramokból leszűrhető, hogy a világosságjel valóban független a színinformációtól, adott színpont színezetét és telítettségét pusztán az $R-Y$ és $B-Y$ diagramokon vett helye meghatározza. Vizsgáljuk most, hogyan definiálhatóak ezen érzeti jellemzők, azaz a színezet és telítettség a TV technika $Y, R-Y, B-Y$ színterében!

1.2.3 Hue and saturation in device-dependent color spaces

A könnyebb elképzelhetőség kedvéért ábrázoljuk az $R-Y, B-Y$ koordinátákhoz tartozó színeket, az adott színponthoz tartozó olyan világosságérték mellett, amely esetén pontonként teljesül, hogy $X+Y+Z = 1$: ezzel gyakorlatilag az adott RGB színtér xy-színpatkón felvett színét képezzük le az $R-Y, B-Y$ diagramra. Az így kapott színhalmaz, amely felfogható az adott alapszínek mellett a luminance-chrominance tér gamutjának is, a 1.11 ábrán látható.

Színezet: Megfigyelhető, hogy a diagramon az origóból kiinduló félegyenesen azok a színek vannak, amelyek egymásból kinyerhetők fehér szín hozzáadásával. Tehát az origóból kiinduló félegyenesen az azonos színezetű, de eltérő telítettségű színek vannak. Azaz tetszőleges színpontot vizsgálva, a $B-Y, R-Y$ diagramon a színpontba mutató helyvektor iránya egyértelműen meghatározza az adott pont színezetét. Ennek megfelelően a TV technikában a színezetet a $B-Y, R-Y$ diagramon a színpont helyvektorának irányszögeként definiáljuk:

$$\text{színezet}_{\text{TV}} = \alpha = \arctan \frac{R-Y}{B-Y} \tag{1.26}$$

a 1.11 ábrán látható jelölés alkalmazásával.

Telítettség: A telítettség kifejezése már kevésbé egyértelmű, több definíció bevezethető rá. Általánosan, a telítettség azt fejezi ki, mennyi fehér hozzáadásával keverhető ki egy adott szín a színezetét meghatározó teljesen telített alapszínből. Az XYZ-térben bevezettük a telítettségre a színtartalmat, illetve színsűrűséget. Felmerül a kérdés, hogyan terjeszthető ki a telítettség fogalma eszközfüggő RGB színterekre. Láthattuk, hogy az adott RGB színtérben előállítható legtelítettebb színek a gamut határán elhelyezkedő kvázi-spektrál színek, amelyek a legközelebb vannak az azonos színezetű valódi spektrálszínhez. A bevezetendő telítettség-mennyiség célszerűen a kvázi-spektrál színekre tehát maximális, egységnyi értékű.

A telítettség ezek után a következő módokon definiálható.

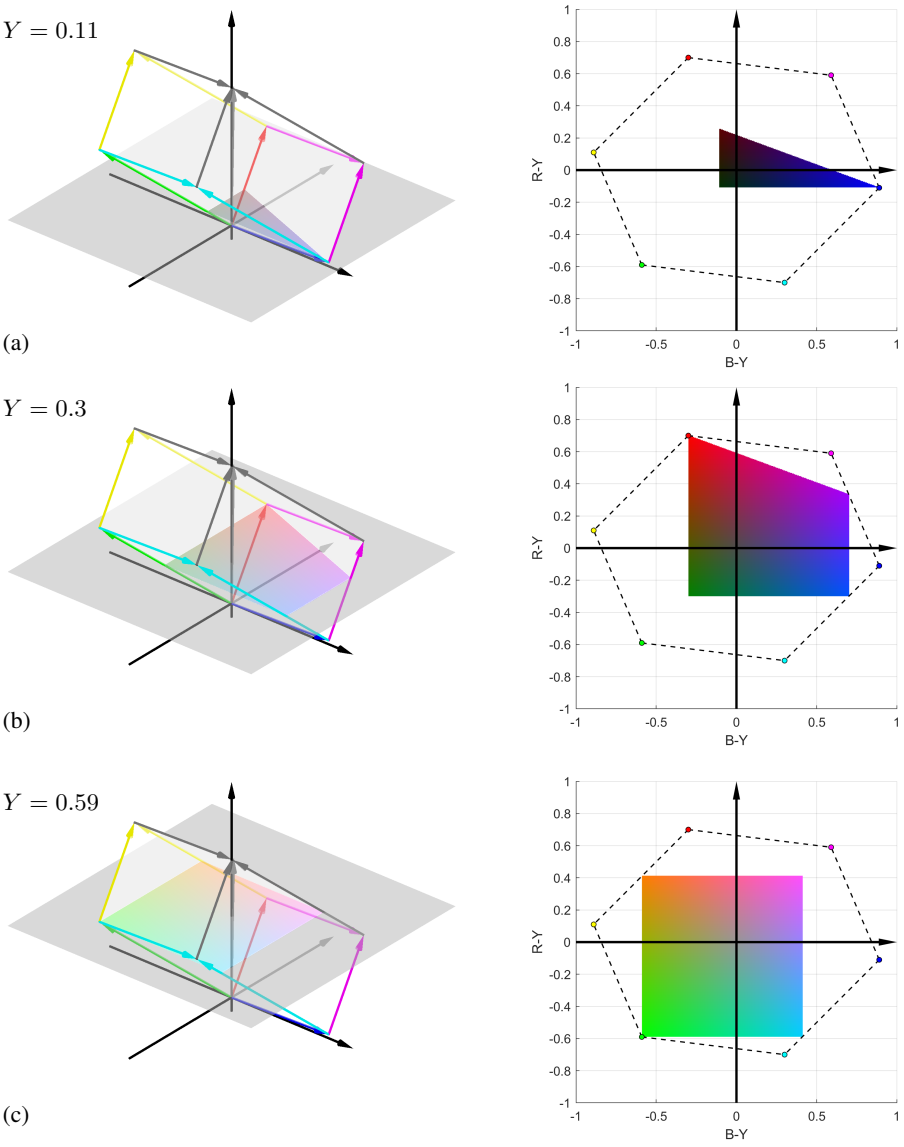


Figure 1.10: Különböző Y értékek mellett rajzolható $B-Y, R-Y$ diagramok.

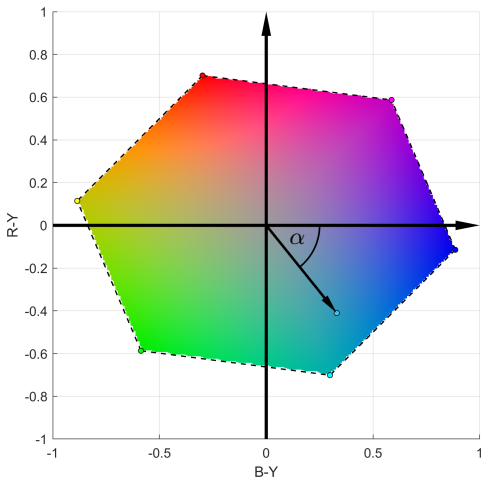


Figure 1.11: Adott $Y, R-Y, B-Y$ térben ábrázolható színek gamutja.

- Minthogy egy tetszőleges színnek a fehér színtől, azaz az origótól vett távolsága arányos a szín fehér-tartalmával, így legegyszerűbb módon a telítettség közelíthető a

$$\text{telítettség}_{\text{TV},1} = \sqrt{(R-Y)^2 + (B-Y)^2} \tag{1.27}$$

távolsággal. Később tárgyalt okok miatt az analóg időkben TV technikusok körében ez a definíció volt érvényben. Az így számolt telítettség valóban 0 a fehér színre, azonban a kvázi-spektrálszínek telítettsége így nem egységnyi.

- A matematikailag korrekt telítettség-definíció bevezetéséhez kiterjeszthetjük a korábban megismert színsűrűséget eszközfüggő színterekre[13]. Ennek egyszerűbb értelmezéséhez ábrázoljuk adott színpont paramétereit ún. területdiagramon! A területdiagram a következő módon rajzolható fel egy tetszőleges

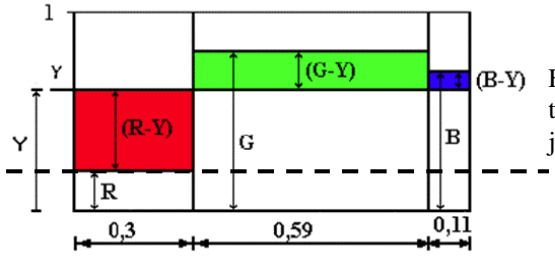


Figure 1.12: Tetszőlegesen választott R, G, B koordináták esetén rajzolható területdiagram.

RGB koordinátáival adott szín esetén: A vízszintes tengelyt osszuk fel az Y fénysűrűség RGB együtthatóinak megfelelően, majd az egyes RGB komponenseket ábrázoljuk az intenzitásuknak megfelelő magasságú oszlopokkal.

[13]Ismétlésként: az XYZ térben adott pont színsűrűsége $p_c = \frac{Y_d}{Y}$, ahol Y_d az adott színhez tartozó domináns hullámhosszú szín fénysűrűsége, Y a vizsgált szín saját fénysűrűsége.

Ekkor egy Y magasságban húzott vonal alatt és fölött a színkülönbségi jeleknek megfelelő magasságú oszlopok alakulnak ki, amely oszlopok előjelesen vett területeinek összege (1.19) alapján zérus.

Válasszuk ki ezután a legkisebb RGB komponenst (a 1.12 ábrán látható példában az R) és húzzunk egy vízszintes vonalat ennek magasságában! Ekkor a vizsgált színt két részre osztottuk: egy fehér színre (amelyre $R = G = B$) és egy kvázi-spektrálszínre, amelynek az egyik RGB komponense zérus, és amelynek fénysűrűsége $Y_d = \min(R, G, B) - Y$. A domináns hullámhosszú spektrálszín szerepét erre a kvázi-spektrálszínre cserélve kiterjeszthetjük a színsűrűséget az adott eszközfüggő színtérre, amely alapján a telítettség definíciója

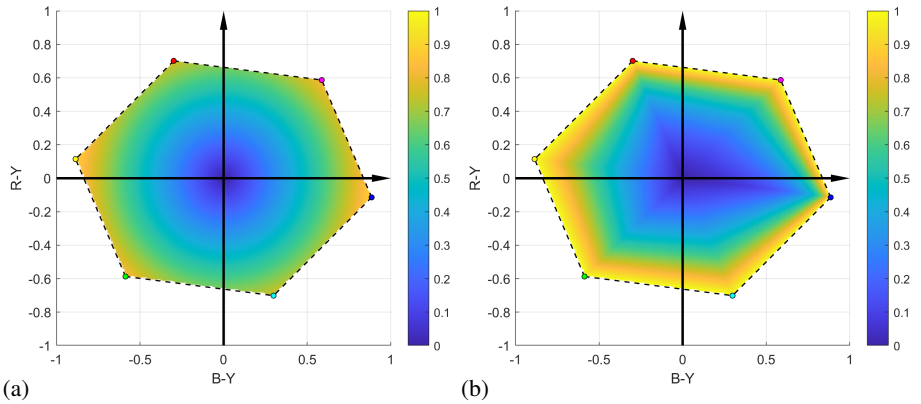
$$\text{telítettség}_{\text{TV},2} = \frac{|\min(R, G, B) - Y|}{Y}. \tag{1.28}$$

Könnyen belátható, hogy az $R = G = B = Y$ fehérpontokra a telítettség definíció szerint 0, míg kvázi-spektrálszínekre ($\min(R, G, B) = 0$) a telítettség azonosan 1.

A fent tárgyalt két telítettség-definíció alkalmazásával a 1.11 ábrán látható színek telítettségét az 1.13 ábra szemlélteti, megerősítve az eddig elmondottakat.

1.3 The Y', $R'-Y'$, $B'-Y'$ components

Az előző szakasz bemutatta, hogyan választható legegyszerűbben szét a világosság és színezet/telítettség információ. A tényleges videójelek ezen $Y, R-Y, B-Y$ jelekkel rokonmennyiségek, azonban történelmi okokból a feldolgozási lánc egy nem-lineáris transzformációt is tartalmaz, az ún. **gamma-korrekciót**.



(a) (b)

Figure 1.13: Az $R-Y, B-Y$ térben ábrázolt színek telítettsége (1.27) (a) és (1.28) (b) alapján számolva

(a)  (b) 

Figure 1.14: RGB kép megjelenítése Gamma-korrekcióval (a) és Gamma-korrekció hiányában (b). Utóbbi esetben az R, G, B komponensek egy 2.4 exponensű hatványfüggvénnyel előtorzítottak.

1.3.1 The role of Gamma-correction

A gamma-korrekció bevezetése történeti okokra vezethető vissza. A CRT megjelenítők elektron-ágyúja erős nem-lineáris karakterisztikával rendelkezik, azaz a képernyő pontjain létrehozott fénysűrűség az anódfeszültség nemlineáris függvénye[14]. Ez a karakterisztika jól közelíthető egy

$$L_{R,G,B} \sim U^{\gamma} \tag{1.29}$$

hatványfüggvénnyel, ahol a legtöbb korabeli kijelzőre az exponens $\gamma \approx 2.5$, $L_{R,G,B}$ az egyes RGB pixelek fénysűrűsége és U a pixelek vezérlőfeszültsége. Ez a nemlineáris átvitel természetesen jól látható hatással lenne a megjelenített képre: Az alacsony RGB szintek kompresszálódnak, míg a világos árnyalatok expandálódnak, ennek hatására a telített színek túltelítődnek, illetve a sötét árnyalatok még sötétebbé válnak. A nemkívánatos torzulás az 1.14 ábrán figyelhető meg.

A torzítás korrekciója kézenfekvő: Az RGB komponensek megjelenítés előtti inverz hatványfüggvénnyel való előtorzítása esetén az előtorzítás és a CRT kijelző torzítása együttesen az RGB jelek lineáris megjelenítését teszi lehetővé $(U^{\gamma})^{\frac{1}{\gamma}} = U$ alapján. Ez a nemlineáris előtorzítás az ún. **gamma-korrekció**.

A korrekció természetesen a megjelenítés előtt bárhol elvégezhető a videófeldolgozási lánc során, azonban a lehető legegyszerűbb felépítésű TV vevők érdekében az előtorzítást az RGB forrás-oldalon célszerű elvégezni[15]. Ennek megfelelően a gamma-korrekció már kamera oldalon megvalósul (akár analóg, akár digitális módon) az RGB jelek közvetlen gamma-korrigálásával. A következőkben tehát

$$R' = R^{\frac{1}{\gamma}}, \qquad G' = G^{\frac{1}{\gamma}}, \qquad B' = B^{\frac{1}{\gamma}}$$

[14] Ez a nemlinearitás az anód-katód feszültség-áram karakterisztikájából származik főleg. A megjelenítésért felelős foszforok már jó közelítéssel lineárisan viselkednek, azaz a gerjesztéssel egyenesen arányos a létrehozott fénysűrűségük.

[15] Természetesen ez a korai TV vevők esetén volt fontos szempont, amikor a gamma-korrekciót drága/komplex analóg áramkörökkel kellett megvalósítani

a Gamma-előtorzított RGB összetevőket jelölik, ahol $\frac{1}{\gamma} \approx 0.4 - 0.6$ szabványtól függően (ld. később).

Fontos leszögezni, hogy ugyan a Gamma-korrekciót a CRT képernyők nemlinearitásának kompenzációjára vezették be, a gamma-korrekció rendszertechnikája manapság is változatlan annak ellenére, hogy a CRT kijelzők alkalmazását szinte teljesen felváltotta az LCD és LED technológia. A gamma-korrekció fennmaradásának oka, hogy a videójel digitalizálása során perceptuális kvantálást valósít meg, ahogyan az a következő fejezetben láthatjuk.

1.3.2 The luma and chroma components

A Gamma-korrekció ismeretében bevezethetjük a mai videórendszerekben is alkalmazott tárolt és továbbított videójel-komponenseket: A videókomponensek előállításának rendszertechnikája a 1.16 ábrán látható, az egyszerűség kedvéért most a kamerából ITU szabványba, ITU szabványból megjelenítő saját színterébe való színtérkonverziókat figyelmen kívül hagyva.

- A gamma-korrekció a kamera RGB-jelein hajtódik végre, SD, illetve HD esetében egy kb. 0.5 kitevőjű hatványfüggvény szerint. A pontos gamma-korrekciós görbéket a következőekben fogjuk tárgyalni.

- Az gamma-torzított R', G', B' jelekből ezután az adott színtér előírt világosságegyütthatói alapján előállíthatók az $Y', R'-Y', B'-Y'$ jelek. Továbbra is példaként az NTSC rendszer együtthatóinál maradva ezek alakja

$$\begin{aligned} Y' &= 0.3\,R' + 0.59\,G' + 0.11\,B' \\ R'-Y' &= 0.7\,R' - 0.59\,G' - 0.11\,B' \\ B'-Y' &= -0.3\,R' - 0.59\,G' - 0.89\,B' \end{aligned} \tag{1.30}$$

  Ezek tehát az alapvető videójel-komponensek, amelyek végül ténylegesen tárolásra, tömörítésre, továbbításra (pl. műsorszórás) kerülnek.

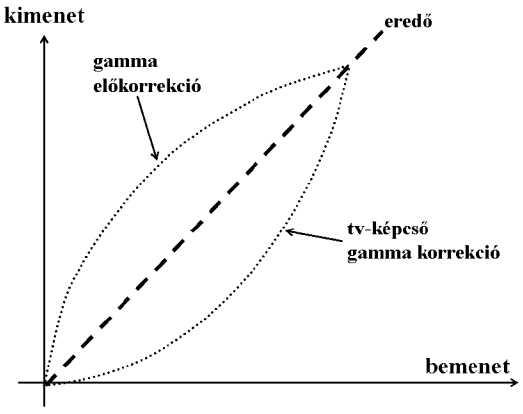


Figure 1.15: A Gamma-korrekció alapelve az RGB jelek előtorzításával.

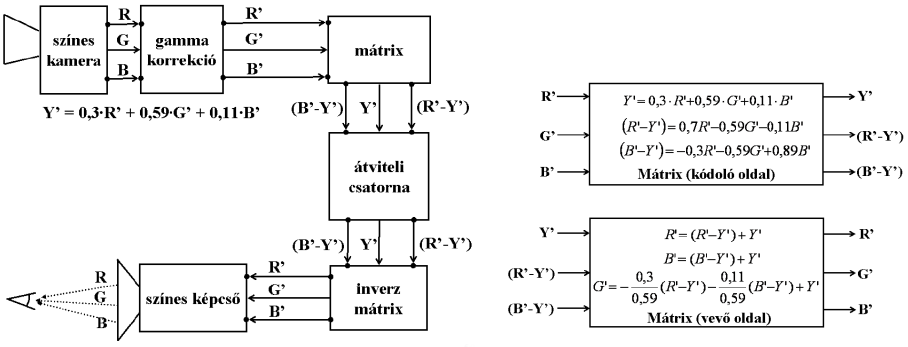


Figure 1.16: A Gamma-korrekció rendszertechnikája és a videójel-komponensek.

- Megjelenítő oldalon a fenti videójelekből a megfelelő inverz-mátrixolással az R', G', B' jelek visszaszámíthatóak. Megjelenítés során a megjelenítő gamma-torzításának hatására a kameraoldalon mért RGB komponensekkel lineárisan arányos fénysűrűségű RGB pixelek jelennek meg a kijelzőn.

Az így létrehozott $Y', R'-Y', B'-Y'$ jelek kitüntetett szereppel bírnak a videótechnikában. Az eddigieket összegezve: ezek adják meg egy színes képpont ábrázolásának módját. A komponensek neve:

- Y': **luma jel**
- $R'-Y', B'-Y'$: **chroma jel**.

A luma és chroma jelek fizikai tartalma:

Fontos észrevenni, hogy a luma jel nem egyszerűen a gamma-korrigált relatív fénysűrűség, hanem a gamma-korrigált RGB jelekből az eredeti Y együtthatókkal számított videójel, azaz

$$Y' = 0.3R^{\frac{1}{\gamma}} + 0.11G^{\frac{1}{\gamma}} + 0.59B^{\frac{1}{\gamma}} \neq Y^{\frac{1}{\gamma}} = (0.3R + 0.59G + 0.11B)^{\frac{1}{\gamma}} \quad (1.31)$$

A luma jel fizikai tartalma emiatt nehezen kezelhető: Legszorosabban az adott színpont világosságával függ össze, fehér szín speciális esetén pl. ahol $R = G = B = Y_W$

$$Y'_W = (0.3 + 0.59 + 0.11)\, Y_W^{\frac{1}{\gamma}} = Y_W^{\frac{1}{\gamma}} \quad (1.32)$$

az egyenlőtlenség egyenlőségbe megy át, azaz a luma megegyezik a gamma-korrigált világosságjellel. Általánosan azonban a luma jel színinformációt is hordoz magában. Hasonlóan, a chroma jelek nem szimplán a gamma-korrigált színkülönbségi jelek (de hasonlóan, fehér esetében azonosan nullák), és így világosságinformációt is hordoznak magukban.

Adott luma értékek mellett az ábrázolható színek halmaza a 1.17 ábrán látható. Megfigyelhető, hogy a luminance-chrominance térrel azonosan az ábrázolható színek egy hatszöget feszítenek ki, és a 100%-osan telített színek helye nem változik (hiszen

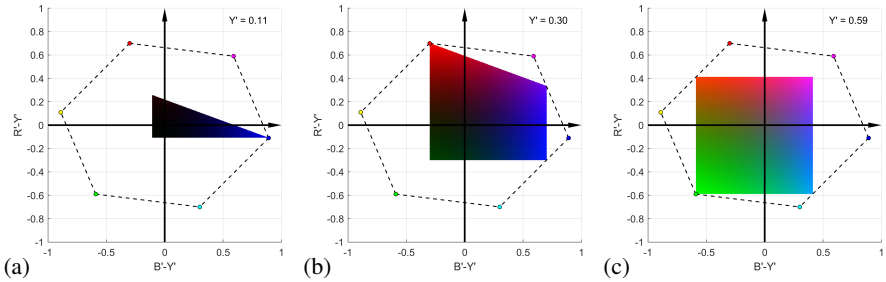


Figure 1.17: A chroma térben ábrázolható színek halmaza fix Y' értékek mellett vizsgálva.

a 0 és 1 értékeken nem változtat a gamma-korrekció), ennek megfelelően az egyes pontok színezete a chroma térben változatlan. Az ábrákon azonban egyértelműen látható, hogy adott Y' értékek mellett is az ábrázolt színek világossága változik, tehát a chroma jelek világosságinformációt is tartalmaznak. Látható, hogy a gamma-torzítás hatására—ahogy 1.14 ábrán is megfigyelhető—adott Y' mellett a telítetlen (fehérhez közeli) színek sötétebbé válnak, míg a telítettebb színek még telítettebbé válnak.

1.4 The $Y'P_BP_R$ color space

A luma és chroma központi szerepet játszanak videótechnikában, a leggyakrabban ezek a jelek a színes képpont ábrázolásának alapja mind komponens, mind kompozit (több komponens kombinációjaként létrehozott videó) formátumok esetén. Utóbbi formátum létrehozásával a következő fejezet foglalkozik részletesen. Analóg, komponens videótechnikában egy színes képpont luma-chroma térben való leírását az $Y'P_BP_R$ **színtérben** való ábrázolásnak nevezzük (az ezekből képzett $Y'P_BP_R$ videójeleket a következő fejezet részletezi).

Az $Y'P_BP_R$ színtér Y' jele maga a luma komponens, míg a P'_B, P'_R jelek szimplán az átskálázott chroma komponensek, a skálafaktort úgy megválasztva, hogy dinamikatartományuk ± 0.5 legyen.

Jelölje az adott RGB színtérben a relatív fénysűrűség együtthatóit k_r, k_g és k_b. Minthogy az $R' - Y$ és $B' - Y'$ komponensek dinamikatartományra rendre $1 - k_r$ és $1 - k_b$, ezért általános az $Y'P_BP_R$ jelek az luma-chroma jelekből a

$$\begin{aligned} Y' &= k_r\, R' + k_g\, G' + k_b\, B', \\ P_R &= k_1\ (R' - Y') = \frac{1}{2}\frac{1}{1-k_r}\ (R' - Y') \\ P_B &= k_2\ (B' - Y') = \frac{1}{2}\frac{1}{1-k_b}\ (B' - Y') \end{aligned} \tag{1.33}$$

összefüggés alapján számítható. Az egyenletekben $R', G', B' \in \{0, 1\}$ a Gamma-korrigált színkoordinátái az ábrázolt színpontnak adott eszközfüggő

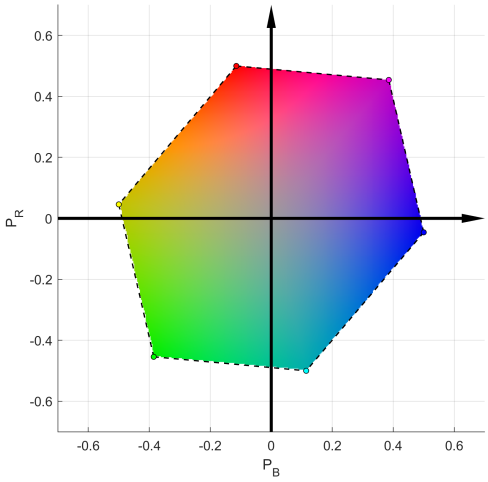


Figure 1.18: Az $Y'P_{\mathrm{B}}P_{\mathrm{R}}$térben ábrázolható színek gamutja.

Hasonlóan meghatározhatjuk általános R', G', B' komponensekre az $Y'P_{\mathrm{B}}P_{\mathrm{R}}$jelek kiszámításához szükséges transzformációs mátrixot

$$\begin{bmatrix} Y' \\ P_{\mathrm{B}} \\ P_{\mathrm{R}} \end{bmatrix} = \begin{bmatrix} k_r & k_g & k_b \\ -\frac{1}{2}\frac{k_r}{1-k_b} & -\frac{1}{2}\frac{k_g}{1-k_b} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2}\frac{k_g}{1-k_r} & -\frac{1}{2}\frac{k_b}{1-k_r} \end{bmatrix} \cdot \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}, \tag{1.34}$$

míg az inverz-transzformációt

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2-2\cdot k_r \\ 1 & -\frac{k_b}{k_g}\cdot(2-2k_b) & -\frac{k_r}{k_g}\cdot(2-2k_r) \\ 1 & 2-2\cdot k_b & 0 \end{bmatrix} \cdot \begin{bmatrix} Y' \\ P_{\mathrm{B}} \\ P_{\mathrm{R}} \end{bmatrix} \tag{1.35}$$

írja le.

Egyszerű példaként a HD szabvány színterében

$$k_r = 0.2126, \qquad k_g = 0.7152, \qquad k_b = 0.0722 \tag{1.36}$$

Az adott alapszínek mellett az ábrázolható színek tartománya a 1.18 ábrán látható.

1.5 Digital representation of color information

So far, the current chapter has introduced the color representation of video technologies by assuming continuous RGB, luma and chroma values. The digital representation of color pixels can be obtained by the direct digitization of the $R'G'B'$, or more often the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$components. The digital representation of the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$signals have its own terminology: it is termed as the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$color space [16].

[16]The $Y'C_{\mathrm{B}}C_{\mathrm{R}}$signals are sometimes incorrectly referred to as $Y'U'VI$ signals (e.g. in VLC player), which term was originally used for the components of the PAL composite video format.

The $Y'C_{\mathrm{B}}C_{\mathrm{R}}$digital color space can be obtained by the quantization of the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$components, with representing the originally continuous values at discrete levels. It is therefore obvious that $Y'C_{\mathrm{B}}C_{\mathrm{R}}$is a device dependent representation, depending on the RGB primaries and its gamut coincides with the gamut of the color gamut of the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$color space, depicted in Figure 1.18 (with of course only discrete number of the reproducible colors due to digitization). In the following the current chapter deals with the questions arising at the quantization of the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$color space.

1.5.1 Perceptual quantization and bit depth

First the optimal quantizer transfer characteristics is investigated, in order to achieve bit-efficient digital representation. As a result, the real role of gamma correction is highlighted.

For the sake of simplicity first it is assumed that the signal-to-quantize is the Y component, i.e. the linear relative luminance signal (without gamma correction). The starting point for defining an appropriate quantizer transfer characteristics is given by the perceptual properties of the human visual system: As a rule of thumb it can be stated that in case of image reproduction, the HVS can not discern luminance levels below $1\ \%$ of the maximal luminance on the given scene. Loosely speaking relative luminances below $\frac{1}{100}$ just appear black for the human observer, therefore, the dynamic range of luminance levels to be reproduced is 100:1.

Within this dynamic range the lightness perception of the HVS is approximately logarithmic function of luminance with the contrast sensitivity being $1\ \%$. This means that two luminance levels can be distinguished only if their relative difference is larger than 1.01. Later the relative luminance-percepted lightness characteristics ($L(Y)$) was given more accurately by the CIE L^* function, describing the lightness as the power function of luminance with the exponent being approximately 0.4.

These properties of human vision establishes the following requirements for quantizing the luminance signal without visible quantization noise:

- The ratio of the largest and the smallest quantized luminance levels should be at least 100:1
- The ratio of the adjacent quantized luminance levels should be at most 1.01, i.e. their relative difference should be less than or equal to $1\ \%$

In the following, as a counterexample for the appropriate quantization strategy the problem with linear quantization is investigated. In this case the digital signal levels are assigned to the relative luminance levels within the dynamic range of $Y \in \{Y_0, 100Y_0\}$ linearly. The quantization can be, therefore, performed by simple rounding to the nearest integer. In case of representing the digital samples at N bits the mapping is given by

$$q = \lfloor \left(2^N - 1\right) \cdot \frac{Y - Y_0}{Y_1 - Y_0} \rceil, \tag{1.37}$$

where $\lfloor \rceil$ is the rounding operation, and $Y_1 = 100Y_0$ is the maximal quantized luminance value. Similarly, the inverse mapping, i.e. the luminance levels of the q-th digital

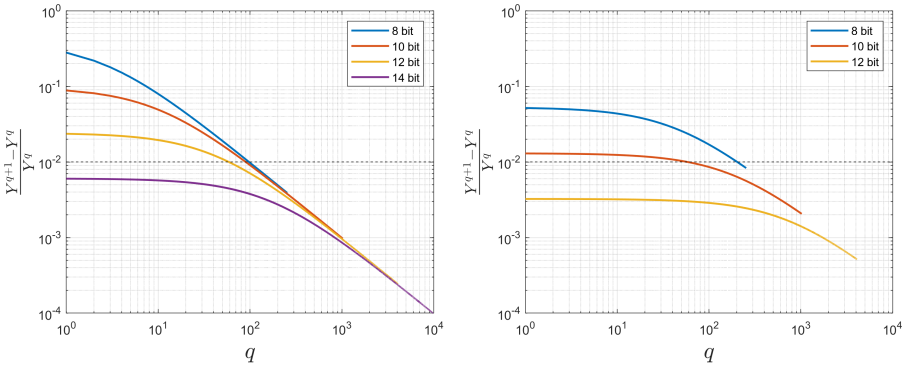


Figure 1.19: Relative difference of adjacent digital codes in case of linear (a) and perceptual (b) quantization.

code is given by

$$Y^q = q \cdot \frac{Y_1 - Y_0}{2^N - 1} + Y_0. \tag{1.38}$$

The relative difference of the adjacent digital codes is then given as

$$\frac{Y^{q+1} - Y^q}{Y^q} = \frac{1}{q + \frac{2^N - 1}{99}}. \tag{1.39}$$

As an example of quantization with $N = 8$ bits, the relative difference between the 101-st and 100-th code (with $q = 100$) the relative difference is

$$\frac{Y^{101} - Y^{100}}{Y^{100}} \approx 0.01 = 1\ \%,$$

meaning that the luminance levels of the adjacent codes are just noticeable. For smaller, or larger codes (e.g. 20 and 21, or 200 and 201) the relative difference is given by

$$\frac{Y^{21} - Y^{20}}{Y^{20}} \approx 0.05 = 5\ \%, \qquad \frac{Y^{201} - Y^{200}}{Y^{200}} \approx 0.005 = 0.5\ \%.$$

Obviously, the luminance levels of codes under 100 are easily distinguishable, meaning that quantization noise at these code levels is clearly visible. As a consequence, in case of the linear quantization of the luminance signal for dark shades the boundary of the different quantization levels would be easily noticeable, leading to so-called banding artifact.

Straightforwardly, by increasing the bit depth (N) banding could be avoided: It is clear that the largest relative difference is between codes 0 and 1. By setting $q = 0$ the smallest bit depth for which (1.39) is larger than 1 %, according to

$$\frac{99}{2^N - 1} \leq 0.01 \qquad N \geq 13.27 \tag{1.40}$$

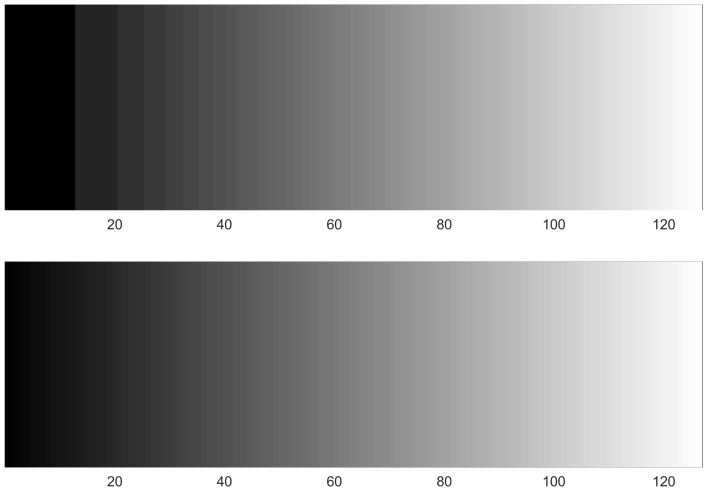


Figure 1.20: Quantized luminance ramp by applying linear (a) and perceptual (b) quantization with $N = 7$ bits. In case of linear quantization the just noticeable quantization is found at $q \approx 99$ (based on $1/q + \frac{2^7-1}{99} = 0.01$).

is given by $N = 14$ bits. This means that linear quantization ensures unnoticeable qunatization noise by applying the bit depth of 14[17]. On the other hand (1.5.1) reflects that codes above 100 have decreasing perceptual utility: luminance at these regimes is quantized with ineffectively fine resolution.

A straightforward strategy in order to avoid the problem with linear quantization would be to quantize the percepted lightness (L) instead of the luminance information, resulting in uniform perceptual difference between adjacent codes. By employing the lightness definition of the CIE (L^*) this **perceptual quantization** can be achieved by the distortion of the relative luminance with the power function of 0.4 before quantization.

The quantization mapping and the inverse mapping in this perceptual case are given by

$$q = \lfloor \left(2^N - 1\right) \cdot \frac{Y^{0.4} - Y_0^{0.4}}{Y_1^{0.4} - Y_0^{0.4}} \rceil \qquad Y^q = \left(q \cdot \frac{Y_1^{0.4} - Y_0^{0.4}}{(2^N - 1)} + Y_0^{0.4} \right)^{\frac{1}{0.4}}. \quad (1.41)$$

Based on these formulae the relative difference of adjacent codes can be expressed for perceptual quantization. The result is depicted in Figure 1.19 (b). It is verified that the relative difference is approximately constant over the entire dynamic range, and even in case of $N = 10$ quantization, it is only slightly higher than 1 %.

As a consequence, in the field of video technologies in studio standards the luminance is quantized perceptually, representing the luminance in 10 bits, while in most

[17] As a consequence digital cameras often digitize the pixel levels by using 14 bits linear quantization, which is requantized to the final bit depth after digital gamma correction.

consumer electronics (e.g. JPEG image compression, MPEG video compression and video broadcasting) representation with $N = 8$ is satisfactory[18]. Therefore, SD and HD studio standards (Recommendations ITU-601 and ITU-709) include the digital representation applying $N = 8$ or $N = 10$ bits, with a rigorous definition of the implementation of quantization and the non-linear pre-distortion curve. This curve is investigated in thew following section in details.

1.5.2 Gamma correction: goal and implementation

The previous section introduced the basic principle of perceptual quantization: pre-distortioning the luminance values by a power function with the exponent being approximately 0.4 the quantization noise can be uniformly distributed over the entire dynamic range. As a result the visible banding of dark shades can be avoided, as it is illustrated in Figure 1.20. In the following the actual implementation of perceptual quantization within the image/video processing chain is discussed.

The optimal signal processing scheme would be the following, as shown on Figure 1.21 (a): At the source of the RGB signals perceptual quantization is achieved by the direct quantization of the perceived lightness ($L^* \sim Y^{0.4}$), obtained from the luminance Y, that can be calculated as the linear combination of the RGB coordinates. At the receiver (e.g. display, TV receiver) following D/A conversion the original luminance value is regained by the inverse distorting the perceived lightness. Finally, from the luminance inforamtion the RGB values to be displayed are obtained by the corresponding inverse transformation.

Note that so far linear RGB values and luminance levels were assumed, without taking the gamma correction into consideration: As a consequence, although perceptual quantization could be achieved, the non-linear transfer characteristics of the CRT display would still result in distorted dynamics of the displayed image. The required neutralization of the CRT distortion would introduce a further non-linear transfer function—as depicted in Figure 1.21 (b)—overcomplicating the signal processing chain (as well as making it more expensive).

However, as a lucky coincidence the luminance-lightness characteristics of the human vision exactly coincides with the required CRT gamma compensation function, both described by $\sim x^{0.4}$, allowing the simplification of the signal processing. As an engineering approximation, both at the source and the receiver side the order of the non-linear mapping and the linear transformation $P : RGB \rightarrow Y, R-Y, B-Y$ is interchanged. This interchange of operations will have two consequences:

- By leaving the principle of strict perceptual quantization, on the source side the quantized signal is not the perceived lightness $L^* = Y^{0.4}$, but the **luma signal** Y', obtained from the gamma corrected color-coordinates, $R^{0.4}, G^{0.4}, B^{0.4}$. As it was already declared, for white shades ($R = G = B$) the luma signal is the gamma corrected luminance signal, i.e. $Y_W^{0.4} = Y'_W = L^*_W$ holds, while

[18] As a third option logarithmic quantization could be performed by setting the ratio of the luminance levels of the adjacent codes to 1.01. In this case according to $1.01^q \geq 100$ the number of required codes is $q = 463$, which can be represented in $N = 9$ bits. Due to historical reasons (due to gamma correction) the presented, power function-based perceptual quantization was introduced in the video standards.

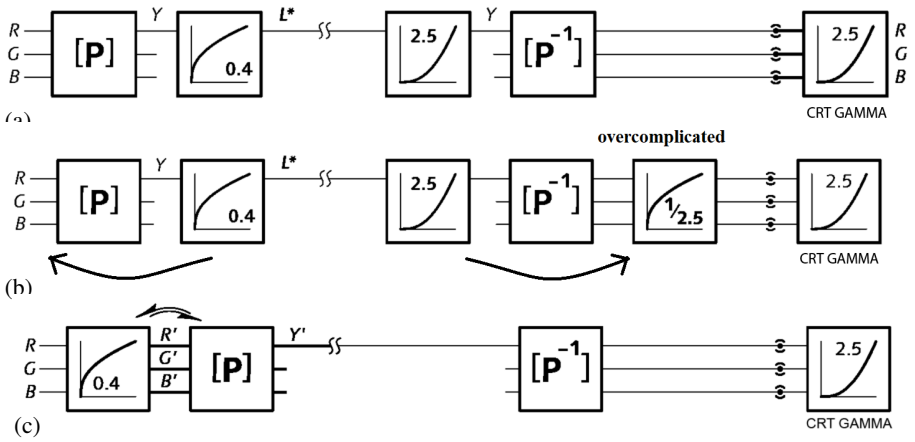


Figure 1.21: Signal processing scheme of gamma correction: Figure (a) realizes true perceptual quantization, however, with the CRT gamma distortion still leading to visible artifacts in the reproduced image. Figure (b) shows a possible, but over-complicated solution for this problem. As an engineering approximation the order of linear and non-linear operations are interchanged on Figure (c), giving a mathematically imprecise solution, but allowing the considerable simplification of the block diagram.

for other colors the luma signal carries color information as well. Therefore, the presented signal processing chain realizes perceptual quantization of white shades, and only approximates it for any other color.

- On the receiver side the CRT correction function exactly neautralizes the inverse quantization function $L^* \rightarrow Y$, resulting in linear resultant transfer characteristics.

The signal processing chain obtained consists only a single non-linear transfer block, with the entire system resulting in the gaama correction technology, discussed already in the previous sections.

Hence, the actual present role of gamma correction is highlighted: Although the role of CRT imaging systems has been almost entirely superseded by LCD and LED based displays, gamma correction is applied in the video processing chain in a completely unchanged manner. However, its real role nowadays is **not** the CRT transfer compensation, but it achieves and approximation of perceptual quantization, adapting the quantization characteristics to the properties of human vision.

Obviously, nowadays the implementation of non-linear transfer functions is computationally inexpensive. Similarly to CRTs, the currently used displays also exhibit highly non-linear driving voltage-display luminance characteristics, which has to be compensated prior to reproducing the input image. Therefore, in current display first the gamma distortion (due to perceptual quantization) has to be neutralized, afterwards

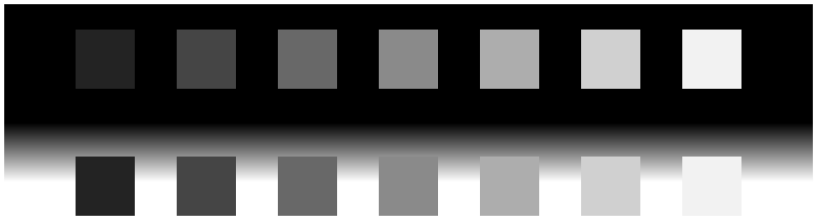


Figure 1.22: Illustration of the Bartleson-Breneman effect.

the actual display characteristics has to be compensated. These non-linear operations are usually implemented before D/A conversion, based on lookup tables (LUTs).

The actual form of gamma correction that has to be implemented before A/D conversion is rigorously codified in SD and HD recommendations. Within these standards the non-linear distortion is referred to as the **opto-electronic transfer function (OETF)**. The actual choice of the OETF is depends on two aspects:

- to achieve approximately perceptual quantization
- to compensate the effects of the viewing environment

On the other hand the standards also codify the non-linear transfer function at the receiver side, which neutralizes the effect of OETF. This receiver side non-linear compensation is termed as the **electro-optical transfer function (EOTF)**.

Compensation of the viewing environment:

So far it was inherently assumed that the exponent of the gamma correction curve is 0.4. However, in actual reproduction systems the exponent of the standardized OETF is usually a higher value, with the actual choice depending on the supposed average illumination level in the reproduction environment.

It was illustrated in Figure 1.14 that the non-linear distortion of the RGB components with the exponent being larger than 1 will result in the compression of deep shades (increase in contrast) and in the saturation of colors. This fact allows the compensation of loss of contrast and saturation due to dim viewing environments: According to the Stevens (Bartleson-Breneman) and Hunt effects in a dark viewing environment the ability of discerning dark shades, the perceived contrast of the image and the perceived saturation of colors decreases.

The Bartleson-Breneman effect is illustrated in Figure 1.22. It is verified that within a displayed image, the image contrast increases with the luminance of surround lighting:

- with bright surround luminance the image contrast (the perceived ratio of the brightest and darkest shades) is larger, than in case of a dark background.

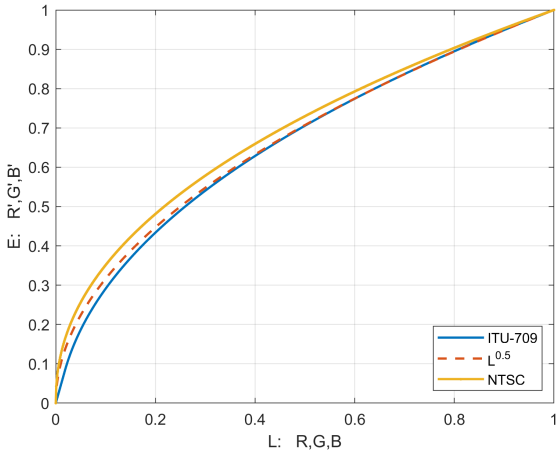


Figure 1.23: The opto-electronic transfer function of the ITU-709 standard and the NTSC standard.

- also the entire luminance-lightness characteristic curve changes depending on the surround luminance: in case of bright surround the perceived contrast between dark shades increases, while bright shades can be distinguished less efficiently, and vice versa.

This means that instead of the well-known $L^* \sim Y^{0.4}$ characteristics, the exponent slightly increases for bright surround luminance levels, and decreases in dark backgrounds.

As a consequence if the environment in image reproduction is dark (e.g. a cinema), in order to avoid the loss of contrast and saturation the overall non-linear transfer function of the signal processing chain should have an exponent of about 1.2-1.5 instead of the linear transfer. This can be achieved with choosing the source gamma correction, i.e. the OETF to be higher than the original value of 0.4, which was originally chosen to compensate the effect of CRT distortion.

Based on these considerations as an example, the ITU-709 HDTV recommendation defines the following OETF

$$E = \begin{cases} 4.500L, & \text{ha } L < 0.018 \\ 1.099L^{0.45} - 0.099, & \text{ha } L \geq 0.018, \end{cases} \tag{1.42}$$

where $L \in \{R, G, B\}$. The entire curve consists of a power function and a linear segment. This linear segment is required in order to avoid the infinite slope of the power function around the origin, that would result in infinite gain of noise around the black level. The entire curve can be well-approximated with a power function of 0.5 (i.e. a square root function), as depicted in 1.23.

The HD standard assumes the receiver gamma distortion to be $\gamma_D \approx 2.5$ (i.e. the EOTF is assumed to be a power function with the exponent being 0.4). Along with the

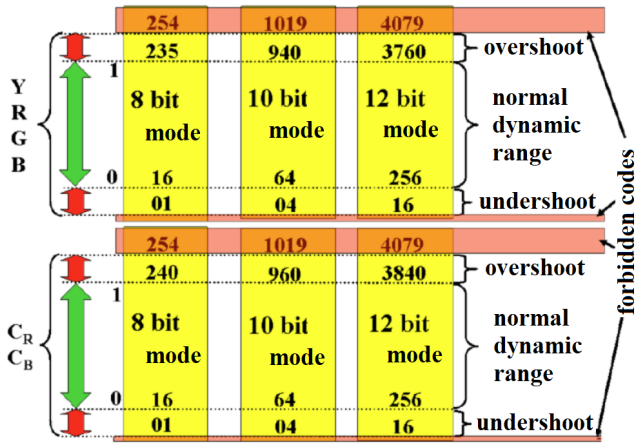


Figure 1.24: The dynamic range of the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$ codes.

prescribed gamma correction this results in a resultant transfer function of $0.5 \cdot 2.5 = 1.25$, which ensures the desired contrast and saturation in an average living room at daylight.

As an other example: the current most widespread digital cinema standard, called the DCI-P3, with the correctly chosen OETF and EOTF achieves a resultant non-linear transfer with the exponent being 1.5. [19]. In practice, in case of current computer and TV displays the actual value of the EOTF, i.e. the display gamma can be freely adjusted in order to achieve the desired contrast.

1.5.3 Dynamic range of $Y'C_{\mathrm{B}}C_{\mathrm{R}}$ representation

In the foregoing it was presented how the gamma correction allows the representation of SD and HD image content as low as 8 (for consumer) or 10 (for studio and professional processing) bits. It seems to be straightforward to utilize the entire possible dynamic range in order to represent the video data, e.g. to let the Y' luma data to cover the $\{0, 255\}$ code range in case of 8 bit representation. This so called **full range** approach is however only applied in the JPEG encoder, and several image editor softwares, operating directly in the RGB color space.

In the field of video technologies in case of $Y'C_{\mathrm{B}}C_{\mathrm{R}}$ representation the image/video data is usually stored and transmitted with a **narrow range** approach. In this case the valid video data is allowed to fill only the part of the available digital dynamic range: In high-quality video, it is necessary to preserve transient signal undershoots below black, and overshoots above white, that are liable to result from processing by digital and analog filters without clipping the video data. Studio video standards provide **foot-**

[19] Actually, in case of e.g. the HD standard, each element of the production and reproduction chain is rigorously defined. The content is produced in such a manner that it would be reproduced with the desired aesthetic properties in a standardized viewing environment (defined by ITU-R BT.2035) displayed on a standardized display apparatus (standardized by ITU-R BT.1886).

room below reference black, and **headroom** above reference white. This code ranges are only containing video data during video processing, and their content is discarded during video storing and transmission.

For the case of 8 bit representation the luma (Y') and (in case of RGB representation) and the R', G', B' components have a headroom of 15 and a footroom of 19 codes, thus the dynamic range of the components is $16 \leq Y' \leq 235$ (codes 0 and 255 are reserved for synchronization in digital interfaces). The asymmetry of the headroom and footroom has no important reason.

For the C'_{B} and C'_{R} chroma components the zero level is the middle of the dynamic range, (i.e. code 128 digitally), while the headroom and footroom are symmetrically 15 codes, i.e. $16 \leq C'_{\mathrm{B}}, C'_{\mathrm{R}} \leq 240$ holds.

For higher bit depths the width of headroom and footroom increases proportionally. As a summary the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$digital levels can be obtained from the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$analog signal levels according to

$$\begin{bmatrix} Y' \\ C'_{\mathrm{B}} \\ C'_{\mathrm{R}} \end{bmatrix} = D \cdot \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + D \cdot \begin{bmatrix} 219Y' \\ 224P'_{\mathrm{B}} \\ 224P'_{\mathrm{R}} \end{bmatrix}, \tag{1.43}$$

where $D = 2^{N-8}$ holds, with N denoting the bit depth.

1.5.4 Chroma subsampling

The luma-chroma representation of image and video content has two great advantages over the direct RGB representation: On one the transmission of the separated luminance information and the color information allowed the color TV transmission system to be fully backward compatible with the then-existing black-and-white TV receivers, as it is discussed in the next chapter. On the other hand it is more adapted to the properties of human vision[20]: The perceptual spatial resolution of the HVS (i.e. the visual acuity) is much lower for spatial change of color information than for that of the luminance. Therefore, the separate transmission of color information allows the bandwidth reduction of the chroma signals, i.e. their transmission with lower spatial resolution. In case of digital representation the reduction of chroma resolution is performed by **chroma subsampling**.

Notation of subsampling schemes

The content of the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$components in case of a simple test image is illustrated in Figure 1.25. Clearly, the color information carries minor high frequency details—and even where details are present they are hardly noticeable for the human vision—thus

[20]The conversion of light to stimulus is ensured by the frequency-selective photoreceptors of the retina, with the three types of cone cells being sensitive to mainly red, green and blue lights. However, the transmission of the stimulus towards the brain on three types of optic nerves, one carrying black and white and two carrying color information.

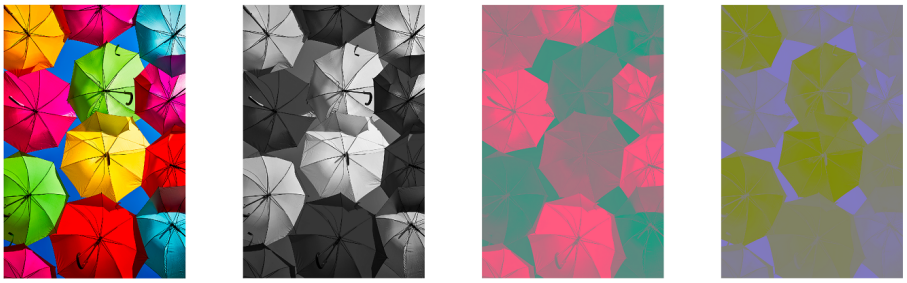


Figure 1.25: The content of the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$components for a simple test image.

the transmission of color information with lower resolution is satisfactory. The resolution of chroma components can be decreased both in the horizontal and vertical dimensions, usually to the half, or quarter of the luma resolution, therefore, different types of chroma subsampling schemes can be implemented. The rate of decimation along the horizontal/vertical dimension are usually denoted by using the following notation:

$$J : a : b : \alpha, \tag{1.44}$$

where the digits denote the following properties:

- J: the first digit stands for the horizontal sampling reference of the luma component. Originally (in the NTSC and SD systems) the first digit indicated the sampling frequency of the luma signal a the multiple of the color subcarrier frequency, via $f_s^{Y'} = J \cdot 3\,\frac{3}{8}$ MHz. In case of HD formats based on this interpretation often $J = 22$ and higher values should be used, so for the sake of simplicity the first digit is usually fixed to the value 4, serving as a reference number for the following digits.

- a: is the C_{B} and C_{R} horizontal decimation factor, given relatively to the first digit. more informally it gives the number of chroma samples in the first row of J pixels. As an example: $J = a = 4 : 2$ means that the horizontal resolution of the color information is half of the luma resolution.

- b: is the number of changes of chroma samples between first and second row of J pixels. If $b = a$, then no vertical subsampling of the chroma samples is performed. If $b = 0$, then the vertical resolution of chroma is half of that of the luma samples.

- α: indicates the presence of alpha channel (e.g. chroma keying). May be omitted, if alpha component is not present, and is equal to J when present.

The chroma subsampling process can be interpreted as a simple lossy compression technique, allowing to reduce the amount of video or image data during storing or transmission. Before reproducing the original RGB signals on the display side, obviously, the discarded chroma samples has to be reconstructed by some interpolation technique.

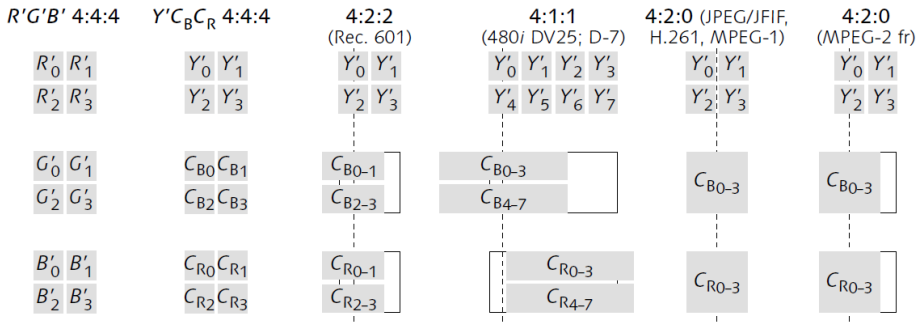


Figure 1.26: Illustration of frequently used chroma subsampling schemes.

Frequently used subsampling schemes

The most common chroma subsampling schemes are summarized in Figure 1.28:

- **4:4:4**: In this case no subsampling is performed neither in the horizontal, nor in the vertical dimension. The spatial resolution of the chroma data is the same as the luma's. If no subsampling is applied, the direct $R'G'B'$ representation of color pixels is often used instead of the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$space. The scheme is, however, scarcely applied in consumer electronics, but mainly used in the field of film archivation, CGI and movie production. The first video studio-standard to support 4:4:4 sampling was the UHD format, published in the ITU-2020 recommendation. The representation of one pixel requires $3 \cdot 8 \text{ bit} = 24 \text{ bits}$ in case of the bit depth of 8.

- **4:2:2**: The horizontal resolution of the chroma samples is the half of the luma resolution, while in the vertical dimension no subsampling is performed. The chroma samples are **cosited** with every second luma samples horizontally, meaning that the horizontal center of the luma samples coincide with the center of every second luma samples. 4:2:2 is the default chroma sampling scheme of SD, HD and UHD studio standards, with also many high-end digital video formats and interfaces using this scheme. Since two pixels consists of two luma, one C_{B} and one C_{R} sample, therefore the representation of one pixel requires$(\frac{2+1+1}{2}) \cdot 8 \text{ bit} = 16 \text{ bit}$, meaning that the compression factor from 4:4:4 to 4:2:2 is $\frac{2}{3}$.

- **4:1:1**: The horizontal resolution of chroma is quarter of that of the luma samples, and in the vertical dimension no subsampling is performed. Originally it was the default sampling scheme of low-end consumer digital electronics, e.g. handheld DVCAMs, but nowadays it is rarely used. Since every 4 pixels contains 4 luma and 1-1 chroma samples, therefore the representation of one pixel requires $(\frac{4+1+1}{4}) \cdot 8 \text{ bit} = 12 \text{ bits}$ and the compression factor from 4:4:4 to 4:1:1 is $\frac{1}{2}$.

- **4:2:0**: Both the horizontal and the vertical resolution of the chroma samples are half of the luma resolution. This is the most commonly used subsampling scheme, used for digital video storing, local playback and broadcasting. Similarly to 4:1:1, 4 pixels contain 4 luma and 1-1 chroma samples, therefore the representation of one pixel requires $(\frac{4+1+1}{4}) \cdot 8 \text{ bit} = 12 \text{ bits}$ and the compression factor from 4:4:4 to 4:2:0 is $\frac{1}{2}$.

  There are two main variants of 4:2:0 schemes, having different horizontal siting (different position of the subsampled chroma samples)

  - In JPEG, H.261, and MPEG-1, C_{B} and C_{R} are sited **interstitially**, halfway between alternate luma samples.
  - In MPEG-2, C_{B} and C_{R} are **cosited** horizontally and are sited between pixels in the vertical direction (interstitially).

The question may arise, how the siting of the chroma samples can be interpreted. In order to answer this the basic steps of chroma subsampling process has to be investigated in more details.

Signal processing questions of chroma subsampling

The practical realization of the chroma subsampling concept requires two basic digital signal processing steps:

- on the source side (e.g. camera) the chroma samples—obtained from the $R'G'B'$ components—has to be sampled with a reduced sampling frequency. Digitally speaking, if the digital chroma samples are directly available then the chroma samples has to be **decimated** (resampled with reduced sampling frequency) according to the current subsampling scheme. The subsampled chroma samples can be stored, transmitted between digital equipment, or broadcasted.
- On the receiver side (e.g. at a display) the $R'G'B'$ components has to be calculated from the luma-chroma representation. In order to do so the discarded chroma samples has to be approximated based on the received chroma data, i.e. the missing chroma samples have to be **interpolated** (resampled with increased sampling frequency) .

Decimation of the chroma samples: As a well-known fact, the frequency content of a discrete signal contains the spectrum of the underlying continuous signal, repeating on the multiples of the sampling frequency. Therefore if the bandwidth of the underlying continous signal is larger than the half of the sampling frequency (the so-called **Nyquist frequency**), after discretization the repeating spectra will overlap, leading to **aliasing** artifacts, and the original continuous signal can not be reproduced from its discrete samples. In order to avoid the spectral overlaping, the signal has to be bandlimited to the half of the sampling frequency with an **antialiasing filter**, being a low pass filter with the cut-off frequency being the Nyquist frequency.

In the field of image reproduction appropriate antialiasing filtering is crucial: aliasing manifests in clearly visible low-frequency patterns on the sampled image. As these

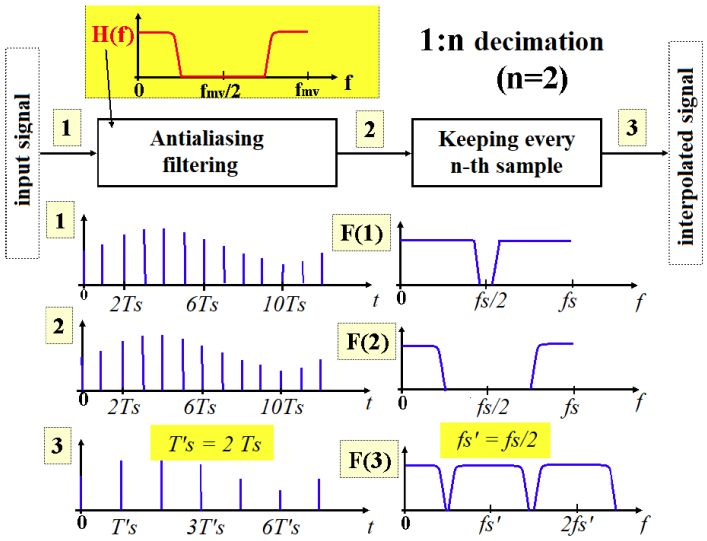


Figure 1.27: Signal processing chain of decimation (subsampling): (1) is the input signal and $F(1)$ the input spectrum. (2) is the output of the antialiasing filter and $F(2)$ is its spectrum. (3) is the output signal.

aliasing patterns, termed as **Moiré patterns** are the result of the spatial sampling of continuous images the type of aliasing is termed as **spatial aliasing**.

Generally speaking, decimation is the reduction of the sampling frequency of an arbitrary discrete input signal. The new sampling frequency is lower than the initial one, and after decimation the spectrum of the input signal will repeat on the multiples of the new sampling frequency. Therefore, prior to setting the new sampling frequency by discarding e.g. every second input samples, the original input signal has to be antialiasing filtered below the half of the new sampling frequency. The process is illustrated in Figure 1.27.

Subsampling from e.g. 4:4:4 to 4:2:2 means the reduction of the chroma samples' horizontal sampling frequency by discarding every second samples in a line of samples, i.e. decimating it horizontally with the ratio of 2:1. Therefore, without antialiasing filtering prior to decimation, the spectrum of the resampled chroma lines would overlap and Moiré patterns would appear in the displayed image. In the present example the horizontal frequency content of the chroma signals has to be bandlimited to the half of the original bandwidth by applying an appropriate **horizontal (spatial) low pass filter**, or horizontal antialiasing filter. In case of converting from 4:4:4 $\rightarrow$ 4:2:0 also a further, **vertical low pass filtering** is required.

The effect of omitting the antialiasing filter prior to decimation is illustrated in Figure 1.28 via the example of 4:1 decimation along the horizontal direction, i.e. in the case of the subsampling scheme conversion from 4:4:4 to 4:1:1. Figure 1.28 (a) depicts

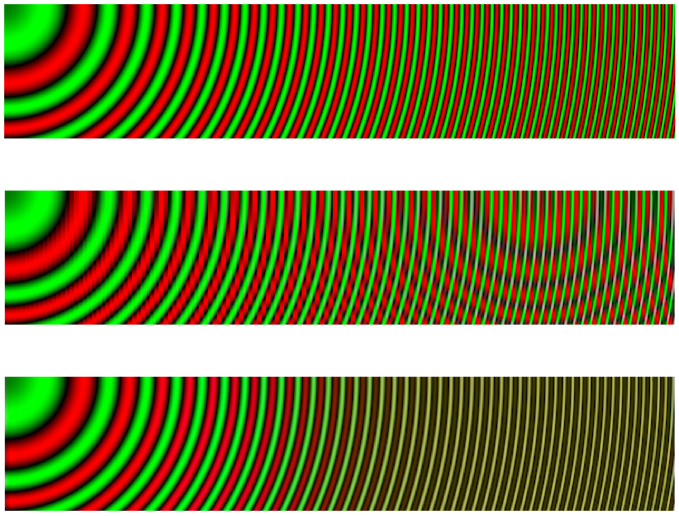


Figure 1.28: Example for the aliasing of the chroma samples. The original 4:4:4 test image (a) contains a sine signal with its frequency increasing along both the horizontal and vertical dimensions, oscillating between the red and green primaries. Figure (b) shows the result of chroma subsampling conversion 4:4:4 $\rightarrow$ 4:1:1, with the interpolation performed by simply repeating the nearest sample ($\mathbf{h}_H = [1\,1\,1\,1]^{\mathrm{T}}$). Figure (c) shows the result of ideal low pass filtering for both antialiasing and interpolation filtering.

the 4:4:4 representation of the image. In Figure 1.28 the chroma samples are subsampled and reconstructed without any antialiasing filter applied. The resulting aliasing Moiré patterns are cleary visible, seriously degrading the image quality. Loosely speaking, the color information of the image contains high-frequency components (small details) that can not be represented in the reduced sampling grid, therefore the Nyquist sampling condition is violated, resulting in the visible patterns[21]. Hence, the application of appropriate antialiasing filtering is crucial. Obviously, since high frequency components in the spectrum represent small details in the image, therefore, spatial low pass filtering can be interpreted as „smoothing" the image by blurring the small details.

Without going deep into signal processing details: the blurring of the small details in a signal can be most easily performed by weighted averaging of the adjacent samples (pixels), i.e. with FIR filtering. Assume that the image is filtered in both the horizontal and vertical dimensions, meaning that both the horizontally and vertically adjacent samples are averaged. The weights of the horizontal samples in the calculated sample is given by vector $\mathbf{h}_H = h_H(n)$. Similarly, the vertical weight factors are given by $\mathbf{h}_V = h_V(n)$. Let $x(m,n)$ denote the intensity of the input sample in the m-th row and n-th column, e.g. in this case being either C_{B} or C_{R} samples. The intensity of the

[21] Aliasing images/Moiré patterns are even more enhanced in case of spatially periodical images, since the aliasing components are also periodical, like the one in the present example.

filtered (or averaged) output sample can be expressed as

$$y(m,n) = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} x(k,l)\, h_V(m-k)\, h_H(n-l), \tag{1.45}$$

describing consequent horizontal and vertical 1D convolutions. Vectors $h_H(n)$ and $h_V(n)$ are the horizontal and vertical filter coefficients, or filter kernels (impulse responses).

As the simplest filtering approach the output is generated as the average of two adjacent samples both horizontally and vertically. This simple averaging process is described by convolution with the filter coefficients

$$\mathbf{h}_H = \mathbf{h}_V = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}. \tag{1.46}$$

As a result, the output sample is obtained as the simple sum of 4 adjacent samples, therefore, the average sample is located between the input samples both horizontally and vertically, i.e. it is sited interstitially. This is the antialiasing filtering approach, applied by the JPEG and MPEG-1 encoders during conversion to 4:2:0 subsampling scheme.

With increasing the computational complexity of filtering—i.e. by applying filters with longer impulse response/involving more input samples to the averaging process—the accuracy of the filtering can be improved, with higher achieved attenuation factor above the filter's cutoff frequency. As an example: MPEG-2 encoder applies the same vertical filter coefficients as the MEPG-1, but in the horizontal direction every output sample is obtained from 3 adjacent input samples, resulting in improved antialiasing performance. The filter coefficients for MPEG-2 are given by

$$\mathbf{h}_V = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}, \qquad \mathbf{h}_H = \begin{bmatrix} 1/4 \\ 1/2 \\ 1/4 \end{bmatrix} \tag{1.47}$$

Since along the horizontal dimension 3 adjacent samples are symmetrically averaged, therefore, the output sample will be located coinciding with the input sample at the center position, i.e. they are cosited.

Interpolation of the chroma samples: The inverse operation of decimation is the increasing of the sampling frequency, by approximating the samples of the input signal in intermediate sampling positions. Interpolation of the missing chroma samples has to be performed in the receiver of the video signal in order to restore the chroma resolution prior to calculating the RGB signal to be displayed.

The signal processing chain of interpolation is illustrated in Figure 1.29: In order to increase the sampling frequency the original signal is stuffed with zeroes in the new sampling positions. This zero stuffing leaves the input spectrum unchanged, since a zero-only spectrum is added to the input spectrum, but with an increased sampling frequency. The approximation of the intermediate samples can be interpreted as filtering

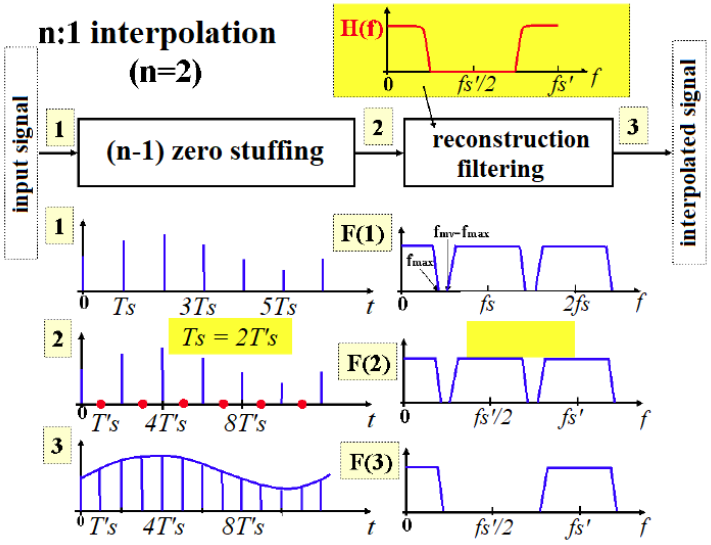


Figure 1.29: Interpolation process, realized by simple linear filtering: (1) is the input signal, and $F(1)$ illustrating its spectrum. (2) zero stuffed input signal and its spectrum $F(2)$. (3) output of the interpolation filter.

out the image spectrum, i.e. the repeating baseband spectrum on the original sampling frequency. Hence, interpolation max be performed by simple low pass filtering of the zero-stuffed signal with an appropriate **reconstruction filter**.

Similarly to decimation, low pass filtering for interpolation can be performed by calculating the weighted average of the adjacent samples. For chroma interpolation in case of 4:2:0 scheme interpolation must be performed both in the horizontal and vertical dimensions. The simplest horizontal and vertical reconstruction filters are given by the coefficients

$$\mathbf{h}_H = \mathbf{h}_V = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{1.48}$$

Obviously, the filter coefficients above realize the repetition of the nearest input samples in the interpolation position. By increasing the computational cost, the missing samples can be approximated more accurately. As an example: Filtering with the horizontal and vertical coefficients

$$\mathbf{h}_H = \mathbf{h}_V = \begin{bmatrix} 1/2 \\ 1 \\ 1/2 \end{bmatrix} \tag{1.49}$$

realize linear interpolation (in case of the interpolation ratio 2:1).

Also, interpolation can be carried out by more sophisticated methods, besides linear filtering, e.g. based on fitting higher order polynomials to the input data (bicubic interpolation).

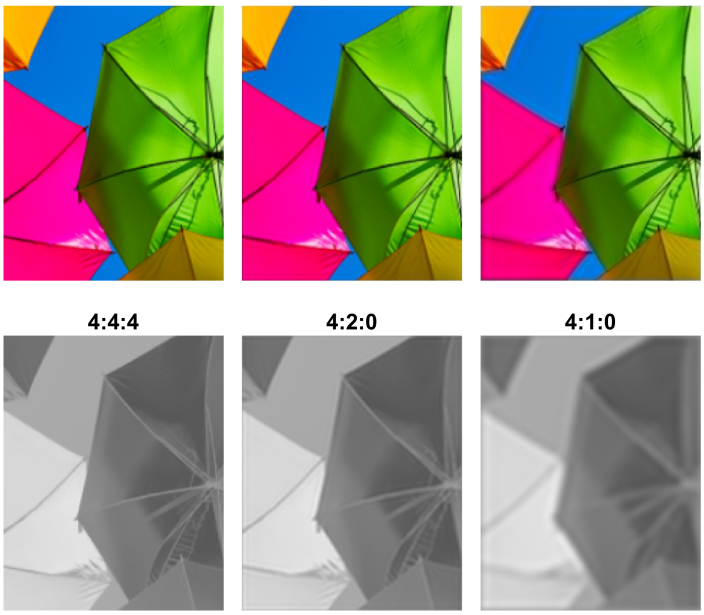


Figure 1.30: Simple example for the chroma subsampling of natural images. The upper row presents the chroma subsampled image after reconstruction, the lower row presents the chroma information only.

Clearly, the introduced antialiasing and reconstruction filter coefficients coincide up to a constant. However, the sum of the antialiasing filtering coefficients has to be equal to 1, otherwise the averaging process would change the intensity of the input signal (e.g. the saturation of the colors). On the other hand, the reconstruction filter in case of $N : 1$ interpolation should have the total energy N, in order to keep the signal energy unchanged.

Figure 1.29 (c) illustrates the effect of chroma subsampling and reconstruction by applying ideal low pass filters for both antialiasing and reconstruction (meaning that its frequency transfer characteristics is a rectangular window). It is clearly shown that on those parts of the image where the rapid change of color information can not be represented in the subsampled grid, the blurring the color information results in an average yellow hue instead of the oscillation between red and green. In practice, in case of natural images the chroma component rarely contains alternating high frequency components, which would result in such clearly visible artifacts after filtering.

Figure 1.30 depicts the chroma subsampling and reconstruction process of a more natural test image. As it is demonstrated, even in the case of a theoretical 4:1:0 subsampling scheme—in which case the chroma resolution is the quarter of the luma information both in the horizontal and vertical dimensions—the final quality of the reproduced image is only slightly degraded. The subsampling scheme 4:2:0, on the other hand,

ensures the image quality, approximately indistinguishable from the original.

End-of-Chapter Questions

- What are the most commonly used chroma subsampling schemes? What is the compression factor of the 4:2:0 scheme, compared to the 4:2:2 scheme?
- What is the difference between the subsampling scheme of MPEG-1 and MPEG-2?

Chapter 2

Video formats

The previous chapter introduced the representation of color pixels in analog and digital systems. The current chapter presents how analog and digital video signals can be formed by using these pixel representations and discusses how the parameters of the resulting video formats were chosen.

2.1 Structure and properties of the video signal

The discussion starts with the questions that arose at the establishment of the early analog TV broadcast systems, beginning with the format parameters elaborated for the NTSC and PAL standards and later adopted by the SD digital format. These analog systems are, of course, already superseded by digital transmission and broadcasting standards. However, their discussion is still of great importance, on one hand because the principles of their actual parameter choices hold unchanged for introducing modern systems as well. On the other hand, a large number of parameters, used in HD and UHD formats are the legacy of these early systems.

2.1.1 Structure of the analog video signal

Before discussing the actual video parameters the structure of the video signals is introduced. The term video signal refers to the signal, carrying video information either from a broadcasting video source, or between local electronic devices (e.g. from set-top-box to the TV on a HDMI interface, or from a computer/laptop to an external display through VGA interface). Even today, the structure of the digital video signal is completely identical with the analog representation, that's structure is the result of the operation principle of early CRT displays.

In the previous chapter 1.1.4 gave a detailed explanation on the principle of cathode-ray tube based imaging devices: The realization of the RGB primaries was achieved by phosphores covering the screen, emitting visible light when excited. The excitation was ensured by an electron gun, producing a narrow electron beam, with the

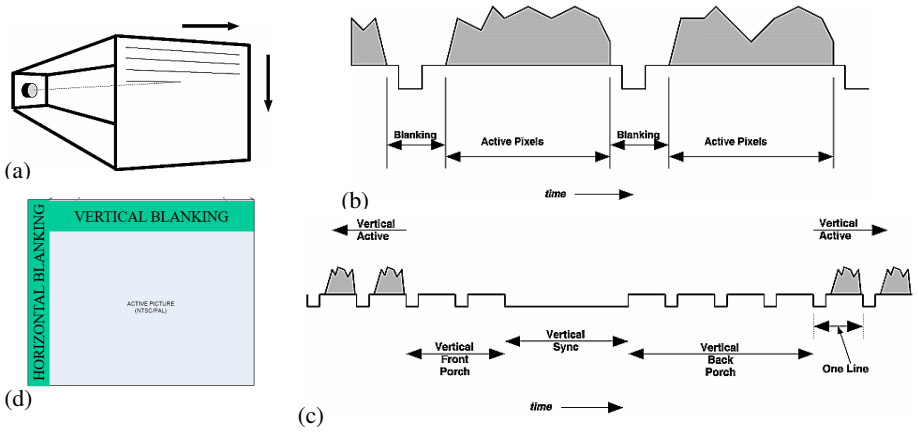


Figure 2.1: General structure of analog video signal

current density being approximately proportional to the 2.2 power of the input voltage (hence the original need for gamma correction).

The electron beams (one in case of black and white, three beams in case of colored display), driven with the input voltage of the video signal, are focused and steered by properly driven magnetic coils, and scan the display screen line by line, along **scan lines** in a predefined **raster scanning pattern**[1]. As the electron beam reaches the end of a given scan line, it retraces horizontally to the beginning of the next line. Similarly, by reaching the lower end of the screen the beam retraces vertically to the beginning of the next frame.

An important principle of CRT displays is that the video data is received and displayed real-time, instantaneously (when early TV receivers were introduced circuits existed for storing video data). Hence, the analog video signal is basically the driving voltage of the CRT electron guns, containing the video data line by line: In case of a black and white display, the video signal consists of the continuous luma information Y', while in color receivers the receiver either receives directly the $R'G'B'$ signals (not used in video broadcasting), or demodulates them from the luma-chroma representation. The principle of raster scanning is depicted in Figure 2.1 (a).

Obviously, it takes a finite time for the electron beam to retrace from the end of a line to the beginning of the next one, and similarly to retrace from the end of the screen to the beginning. During these **retracing intervals** the electron beam is turned off, i.e. **blanked**, otherwise undesired traces would appear on the screen. In practice this means that during these **blanking intervals** the video signal is zero, or negative valued, i.e. it takes black level, or synch level. In the blanking interval synchronizing pulses are added to the video signal that ensure that the oscillators in the receiver remain locked with the transmitted signal, so that the image can be reconstructed on the receiver screen.

[1]The choice of scanning the screen linewise is not absolutely obvious, while creating the early black and white TV standards other solution, e.g. columnwise or back-and-fortha linewise scanning types were also investigated.

Based on the foregoing the linewise video signal contains three distinct time intervals:

- The **active video content**, containing the actual video information of one line. In analog formats the length of the active video interval is expressed in μs, and in px-s in the digital case.
- The **horizontal blanking interval**, the time it takes for the electron beam to retrace from the end of the actual scan line to the beginning of the next scan line. The horizontal blanking interval contains the horizontal sync pulse (**HSYNC**) with negative signal level (i.e. „blacker than black"), serving as the trigger pulse for the horizontal retracing circuit. The sync pulse is separated from the active video data with black level intervals before and after the sync pulse, termed as the front porch and black porch. The length of horizontal blanking is expressed in μs in the analog, and in px-s in the digital case.
- The **vertical blanking interval (VBI)**, the time it takes for the electron beam to retrace from the end of the actual frame to the beginning of the next frame. The vertical blanking interval contains the vertical sync pulses (**VSYNC**) at negative signal level, allowing the vertical synchronization of the display frames on the screen. The length of the VBI is expressed in lines (number of the inactive lines).

The video signal, therefore, contains in one entire line period time (T_{H}, for horizontal) both **active video interval** and **inactive interval**, carrying no video information. The structure of video lines is depicted in Figure 2.1 (b). Similarly, the entire period time of one frame (T_{V}, as vertical) contains **active lines** and **inactive lines**, containing no actual video data. The structure of active and inactive lines is depicted in Figure 2.1 (c).

In the following it is revealed what principles lead to the actual timing parameters (e.g. the lengths of the above intervals, line frequency, frame frequency) and to the standard definition resolution parameters.

2.1.2 Parameters of analog video formats

Aspect ratio and display size

First it is explained what display size should be the optimal format parameters chosen for.

It was already discussed in **??** that the human color vision is ensured by the macula, containing mostly cones, located around the center of the retina. In the center of the macula the fovea is responsible for sharp central vision (also called foveal vision). Due to the size of the fovea, the human central vision with high visual acuity covers about $10 - 15^{\circ}$ from the entire field of view of $\approx 200^{\circ}$ horizontally. During the creation of standard definition television standard the basic goal was to fill only the central vision with content, therefore, the SD television should cover about 10° from the horizontal field of view (i.e. the peripheral vision does not contribute to imaging). Obviously, the actual display size then depends on the viewing distance, as it will be discussed later in this chapter.

Another important spatial attribute of video formats is the ratio of the horizontal and vertical screen size, i.e. the **aspect ratio** (a_r). The basis of all the color TV formats was the NTSC standard and its black and white predecessor, introduced in the 1940s. As an obvious endeavor the introduced broadcasting format was engineered in order to be compatible with the then-existing video sources, e.g. movie films. Until the 1950s during the entire silent film era movies were almost exclusively captured with the aspect ratio of 4:3, i.e. the ratio of the horizontal and vertical screen size was $1.3\dot{3}$ [2]. Although by the 1950s first widescreen movie formats have already emerged, the NTSC standard adopted the **4:3** aspect ratio, which remained the standard TV and video aspect ratio until the introduction of the HD format.

Refresh rate and frame rate

The next parameter to investigate is the temporal sampling frequency of the video data, i.e. the number of frame fed to the display device per second. First three closely related terms are introduced:

- The **refresh rate** ($\mathbf{f}_\mathrm{r}$) refers to the number of times in a second that a display „flashes", i.e. redraws its content, expressed usually in Hz.
- The **frame rate** ($\mathbf{f}_\mathrm{V}$) express the number of time in a second that the content of the display changes, i.e. the number of frames, contained by the video signal per second, usually expressed in fps (frame per second) or in Hz.
- The **field rate** ($\mathbf{f}_{\frac{\mathrm{V}}{2}}$) can be defined for interlaced video data, denoting the number of fields (half frames) per second in the video data, generally expressed in fps. Universally $f_{\frac{\mathrm{V}}{2}} = 2 \cdot f_\mathrm{V}$ holds.

The key difference between refresh rate and frame rate is that refresh rate is the property of the display device (e.g. LCD display) and includes the repeated drawing of identical frames, while frame rate measures how often a video source can feed an entire frame of new data to a display.

In order to arrive at practical frame rate and refresh rate choice for video data two perceptual aspects have to be taken into consideration:

- On one hand when reproducing objects under motion it is crucial to display sufficient number of motion phases in order to ensure that the observer perceives a smooth, continuous motion. This requirement constitutes a lower limit for the applicable frame rate.
- On the other hand the refresh rate has to be chosen high enough in order to avoid **flickering** and to reduce eye strain.

[2] The introduction of the aspect ratio of 4:3 is connected to the work of Thomas Alva Edison, who defined the standard image exposure length on $35\ \mathrm{mm}$ film for movies is four perforations ($19\ \mathrm{mm}$) per frame along both edges. From the available film width between perforations ($25.375\ \mathrm{mm}$) the active are has an aspect ratio of 4:3. This 4-perf negative pulldown became the official standard in 1909, allowing the emerging of standard movie cameras, movie projectors, and hence emerging of cinema technologies.

Due to the **beta movement** phenomenon the frame rate can be significantly lower, than the refresh rate: Beta movement is an optical illusion whereby viewing a rapidly changing series of static images creates the illusion of a smoothly flowing scene, occurring if the frame rate is greater than 10 to 12 fps. Therefore, due to the beta movement the frame rate should satisfy

$$f_{\text{frame}} >\sim 20 \text{ Hz}, \tag{2.1}$$

[3] however, applying the same refresh rate would lead to serious perceived flickering artifacts.

In over to avoid flickering the refresh rate has to be higher than the **flicker fusion threshold** [4]. For the purposes of presenting moving images, the human flicker fusion threshold is usually taken between 60 and 90 Hz: in the central vision, dominated by the cones the response time is high, and the flickering fusion threshold is around 50 Hz. The peripheral vision is dominated by the rods, with a much lower response time and the flickering fusion threshold is higher.

In case of analog TV formats the goal was to fill the central vision of the viewer with, therefore, with a refresh rate of

$$f_{\text{r}} > 50 - 60 \text{ Hz} \tag{2.2}$$

flickering can be avoided[5]. The choice of the actual refresh rate was, however, a consequence of the CRT technology's imperfection: a trick in order to avoid the effect of the supply voltage ripple.

Effect of the mains frequency: Voltage ripple is the residual periodic variation of the DC voltage in a power supply due to the imperfect rectification of the alternating means AC voltage, as illustrated in 2.2. The ripple frequency coincides with the means frequency in case of single-way rectification, or with its double in the two-way case. As in a CRT display the anode is directly connected to the supply voltage, therefore, any perturbation in the DC voltage is directly added to the video signal and is displayed on the screen.

[3] It is worth noting that the above frame rate only allows the perception of motion instead of distinct static images, higher frame rates)(usually 60 fps) still ensure much smoother motion reproduction. In order to increase the effective frame rate of the video stream, modern displays and software allow temporal interpolation by estimating the intermediate frames, based on some motion prediction algorithm, similarly to MPEG encoding. However, the average viewer already adapted to the frame rate of 24 fps, being the standard frame rate of movie films, therefore the increased frame rate often generates an unpleasant, unnatural effect. This is termed as the **soap opera effect**, originating from the fact that usually low cost TV shows are recorded directly to digital video cameras (being much cheaper than capturing to film), allowing higher frame rates, usually set to 60fps.

[4] The flickering fusion threshold is the frequency at which an intermittent light stimulus appears to be completely steady to the average human observer. For surfaces with temporally alternating luminances above the flickering threshold the average luminance is perceived. The threshold depends on numerous factors: Depends on the average illumination intensity, the adaptation state, the color of vibration (above $15 - 20$ Hz fluctuation of hue information can not be perceived) of and the position on the retina at which the stimulation occurs

[5] This is true only for the central vision. The flickering of CRTs can be easily observed with a display watched from the peripheral vision

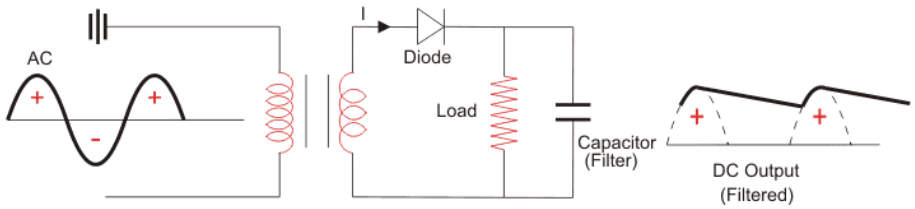


Figure 2.2: Source of the supply voltage ripple in a single-way rectifier

Assume a screen consisting of N_{V} horizontal scan lines, with the refresh rate denoted by f_{r}. The number of scan lines displayed in a second, i.e. the line frequency is given by

$$f_{\mathrm{H}} = N_{\mathrm{V}} \cdot f_{\mathrm{r}}. \tag{2.3}$$

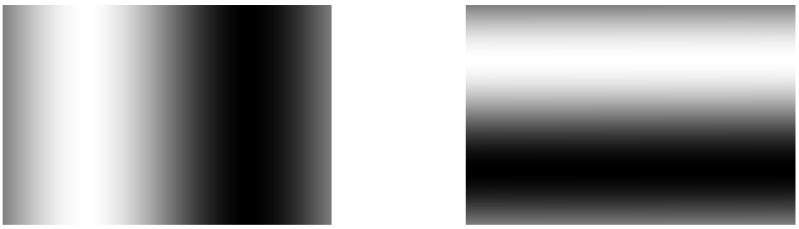
In order to investigate the effect of supply ripple, as a generalization, assume a periodic black and white video signal, oscillating between 0 and 1, described by

$$Y(t) = \frac{1}{2}\left(1 + \sin 2\pi f t\right), \tag{2.4}$$

with f being the signal frequency. For the sake of simplicity the blanking intervals are assumed to be of zero length. In this case the sine wave is displayed on the screen line by line. Depending on the signal frequency f the content of the screen can be the following:

- If $f = f_{\mathrm{H}}$ each scan line consists of exactly one period of the sine wave, resulting in a horizontal sine on the screen, as depicted in Figure 2.3 (a).
- If $f > f_{\mathrm{H}}$ each scan line consists of less then one period of the sine wave, with the initial phase in the beginning of the scan lines increasing line by line and frame by frame. Therefore, the horizontal sine on the screen is slightly steered, and moves towards left. Similarly, with $f < f_{\mathrm{H}}$ the image moves slowly towards right.
- If $f = f_{\mathrm{r}}$ each frame consists of a single period of the sine wave. Assuming $N_{\mathrm{V}} \gg 1$ the value of the sine function changes insignificantly over one scan line, therefore, the displayed image is a vertical sine.
- If $f > f_{\mathrm{r}}$ the initial phase of the vertical sine increases frame by frame, thus, the sine wave moves slowly upwards. Similarly, for $f < f_{\mathrm{r}}$ the vertical sine moves downwards.

Based on the foregoing, with the appropriate synchronization of the signal frequency and the refresh rate a periodic video signal can be displayed as a still, non-moving image. The effect of voltage ripple acts as such a periodic noise signal, with the frequency determined by the mains voltage of the given region and displayed inevitably on the screen. Early tests with black and white TV receiver suggested the visible effect of



(a) (b)

Figure 2.3: Periodic video signal with $f = f_{\mathrm{H}}$ (a) and $f = f_V$ (b) as displayed line by line

this periodic noise of less disturbing if the noise image is still. Therefore, both in the American, and later in the European systems refresh rate was chosen to coincide the mains frequency [6].

$$f_{\mathrm{r,USA}} = 60 \text{ Hz}, \qquad f_{\mathrm{r,Eu}} = 50 \text{ Hz} \tag{2.5}$$

ensuring that supply ripple only manifests in a non-moving vertical noise image on the screen.

Having found the refresh rate for early TV systems, which has been also in use by modern CRTs and LCD displays, still, the actual frame rate is an open question. An obvious choice for the frame rate would be the refresh rate itself. Due to bandwidth efficiency, however, instead a special raster scan order was introduced.

Raster scan orders

The raster scan order is the rectangular pattern of image capture and reconstruction in television. In the receiver side it defines the systematic process how the electron beam scans the entire screen over a given time interval. The following section introduces two frequently used raster scan order, leading to the definition of frame rate and field rate for.

Obviously, for currently used LCD and OLED displays the raster scan order can not be interpreted as the scanning pattern of the display, since the entire display content is updated in the same time instant. Still, scan order can be understood also as the storing and transmission order of video data, hence scan order is interpretable in modern video systems as well.

Progressive scan: As the most straightforward scanning order, **progressive scan** is a format of displaying, storing, or transmitting moving images in which all the lines of

[6] In the American NTSC system with the introduction of color information the choice of refresh rate became more complicated, as the frequency of the color subcarrier could not be correctly chosen. Without more details: as a consequence both the refresh rate (and the field rate) and the line frequency had to be decreased by 0.1 %. Therefore, in the American system the refresh rate is $f_r = 60 \cdot \frac{1000}{1001} = 59.94$ Hz. Due to the presence of vertical and horizontal sync pulses, this change did not had and effect on the then-existing TV receivers.

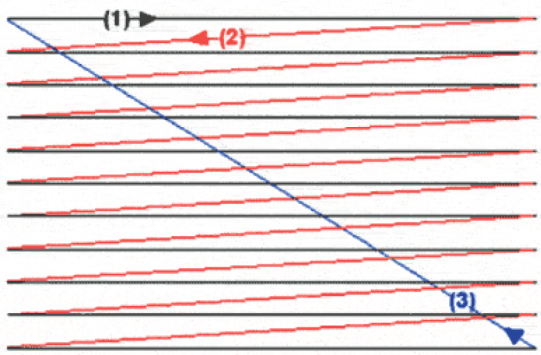


Figure 2.4: Illustration of progressive scanning (for the sake of simplicity with the exemplary number of 11 lines), with the active line content (1), the vertical retrace/blanking interval (2) and the horizontal retrace/blanking interval (3).

each frame are drawn in sequence. In the receiver side it means that the electron beam scans and updates the content of the entire screen over one frame time ($T_{\mathrm{V}} = 1/f_{\mathrm{V}}$), line by line. Progressive scanning is illustrated in Figure 2.4.

In the aspect of transmission in case of video data with progressive scanning the content of the entire frames has to be transmitted over the frame time via the given interface (e.g. HDMI) sequentially. Progressive scan is usually denoted by letter „p" in the format designation.

Although progressive scan seems to be the most simple and obvious scan order, still, until the emergence of the UHD format progressive scan was only occasionally utilized due to the reasons discussed in the following.

Interlaced scan: The previous section have already discussed that in order to avoid flickering the display refresh rate should be risen above 50 Hz, while for ensuring continuous motion the video frame rate of around 20 Hz may be sufficient. This fact suggests that frame rate should be handled independently by choosing a high refresh rate and a efficiently low frame rate, allowing the reduction of video data and the required bandwith for transmission.

This concept could be easily implemented in cinematic techniques: As a heritage of the early silent movie era (where the frame rate varied between $16 - 24$ Hz) movie films are commonly captured with the frame rate of 24 frames per second. In order to avoid flickering the film projectors are equipped with special two or three blade shutters, rotating in front of the projector beam and flashing each frame two or three times before the film would travel to the next frame. With the simple trick of presenting the same frame multiple times the effective refresh rate can be increased to 48fps or 72 fps, and flickering may be avoided.

Also, nowadays frame rate and refresh rate can be handled independently in digital systems (e.g. in a video card and a VGA monitor), where a **display buffer** is present, containing the data of an entire frame, generating and feeding the video signals towards the display. Modern LCD displays are usually built with the LED backlight panel flashing at around 200 Hz[7]. Still, as a rule of thumb even in the presence of a video

[7]Obviously, the LED panel could be built with continuous driving voltage of well, instead of flashing,

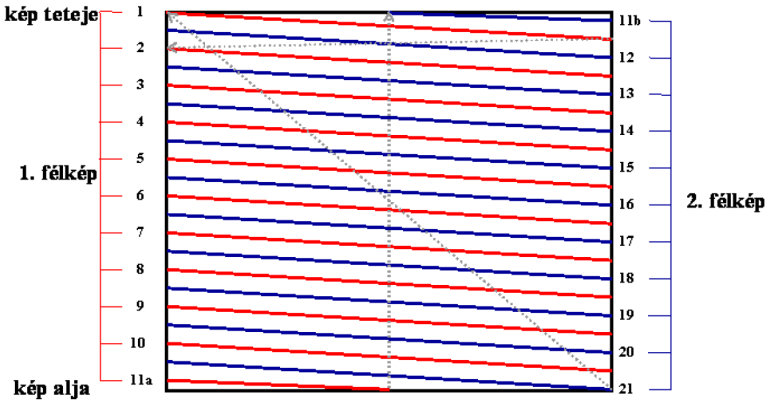


Figure 2.5: Illustration of interlaced scanning (for the sake of simplicity with the exemplary number of 21 lines) Assuming that scanning starts with the beginning of the first scan line, in order to cover the entire screen the first field has to end in a half-line, and the second field has to start with a half-line. This requirement can be satisfied only by applying odd number of scan lines (each field consists of $N_{\frac{V}{2}} + \frac{1}{2}$ lines, thus the total line number is given by $2N_{\frac{V}{2}} + 1$, being an odd number by definition).

buffer the content of the video buffer may change during the vertical blanking of the display in order to avoid screen tearing. As a consequence, generally $f_{\mathrm{r}} = \mathrm{n} \cdot f_{\mathrm{V}}$ holds, where $\mathrm{n} \in \{1, 2, 3, ...\}$.

In analog TV receiver no frame buffer could be present due to the lack of appropriate technology, the transmitted signal had to be displayed by the receiver on the fly. Obviously, transmitting the same frames multiple times requires a significant increase of bandwidth. Therefore, a more simple engineering workaround was needed for the bandwidth efficient video transmission, which was eventually provided by the concept of **interlace scanning**.

The basic concept of interlaced scanning is illustrated in Figure 2.5: Instead of scanning each line of the display within one frame period time, the screen is divided into two **fields**

- the **odd field**, containing every odd scan line
- the **even field**, containing every even scan numbered scan line

The interlaced signal contains the two fields of a video frame captured consecutively (i.e. the even and odd fields are captured in consequent time instants). Similarly, in the receiver the electron beam scans first all the odd lines, displaying the content of the odd field, then draws the content of the even field into all the even lines. Interlaced scan is usually denoted by letter „i” in the format designation.

By applying interlaced scanning the content of the display is redrawn with the frequency of the fields, thus, for interlaced video the refresh rate is given by the **field**

but the dynamic adjustment of its lightness (based on the video content) can be much cost-efficiently solved with PWM modulation.

rate ($f_{\frac{\mathrm{V}}{2}}$). Therefore, the field rate is determined by the mains frequency of the given region, being 50 Hz for the European and 60 Hz (more precisely 59.94 Hz) for the American electric network. Furthermore, the period time of an entire frame, consisting both even and odd fields is obviously twice the field period time, thus the frame rate is half of the field rate. As a result in the frame rate is 25 Hz in the European and 30 Hz (29.97 Hz) in the American systems, being high enough to ensure the perception of continuous motion, while the refresh rate is sufficient in order to avoid flickering. As a summary for the foregoing in the European and American system

$$\begin{aligned} f_{\mathrm{r,USA}} = f_{\frac{\mathrm{V}}{2},\mathrm{USA}} = 2 \cdot f_{\mathrm{V,USA}} = 60 \text{ Hz}, \qquad f_{\mathrm{V,USA}} = 30 \text{ Hz} \\ f_{\mathrm{r,Eu}} = f_{\frac{\mathrm{V}}{2},\mathrm{Eu}} = 2 \cdot f_{\mathrm{V,Eu}} = 50 \text{ Hz}, \qquad f_{\mathrm{V,Eu}} = 25 \text{ Hz} \end{aligned} \tag{2.6}$$

hold.

Opposed to the solution of cinema, i.e. displaying the same frame multiple times the even and odd fields in interlaced video are taken in consequent time instants. Therefore, in interlaced video the temporal resolution is effectively doubled at the cost of halving the vertical spatial resolution of the video data. This means the following properties of interlaced video

- Compared to progressive video with the same refresh rate, interlaced scanning achieves the compression factor of 2:1, resulting in halved analog bandwidth
- In case of still, and slowly moving scenes the vertical resolution coincides with the progressive resolution (since the even and odd fields complete the same frame)
- In case of rapid motion the vertical resolution if half of the progressive resolution

Generally speaking for video content with slowly changing content, e.g. movies, interlaced scanning ensures a sufficiently high vertical resolution with improved motion reproduction besides feasible signal bandwidth. In case of fast motion, e.g. camera movements in sport programs the artifacts due to the reduced vertical resolution become visible.

As bandwidth efficiency was of primary importance during the introduction of early analog television broadcasting, all the analog and digital standard definition video formats adopted exclusively interlaced scanning. Later the high definition video standard introduced progressive video formats for the first time, while the latest ultra high definition format supports progressive scanning mode only.

Questions of interlaced scanning: Besides its obvious advantages, the application of interlaced video rises a number of questions and challenges.

As an example, it introduces the phenomenon of **interline twitter**: The previous chapter explained that the violation of the sampling theorem—i.e. sampling frequency components above half the sampling frequency—leads to spatial aliasing phenomena.

Figure 2.6: Displaying interlaced video on a progressive display without deinterlacing.

In the field of image processing it manifests in visible Moiré pattern on spatially periodic textures (e.g. a brick wall, or squared shirts). In case of interlaced video the vertical resolution is halved, compared to progressing scanning, therefore, vertical aliasing artifacts are more likely to emerge. Since the even and odd fields are displayed alternatively, therefore also the potential Moiré patterns alternate from field to field. This may produce a shimmering effect, termed as twittering, even when a still image is reproduced. Interline twittering is the main reason, why television professionals avoid wearing clothing with fine striped patterns, while professional video cameras apply a low-pass filter to the vertical resolution of the signal to prevent interline twitter.

Further questions arise in case of the conversion between interlaced and progressive formats. The progressive to interlaced conversion can be straightforwardly solved by dividing the entire frame to odd and even lines. Interlaced to progressive conversion emerges more frequently in practice: e.g. in case of the playback of a DVD disc on a usually progressive computer monitor. As the simplest solution, two adjacent field can be combined to form a single frame, however, this naive approach leads to the „combing" of moving objects on the screen, as illustrated in Figure 2.6. Generally speaking techniques for the interpolation of the content of intermediate scan lines are termed as **deinterlacing** methods. Deinterlacing is an often emerging problem even today, as broadcasting most often transmits interlaced HD format (generally using format 1080i), while modern LCD displays do not support native interlaced scanning anymore.

2.2 Analog video formats

Based on the foregoing, the main properties of the video format used for analog broadcasting can be introduced. Historically, three analog video formats were introduced with the advent of color television broadcasting:

- The first color TV broadcast system was the **NTSC** format, introduced by the Federal Communications Commission (FCC) in the United States in 1954 and have been adopted by western America and Japan. For vertical resolution NTSC

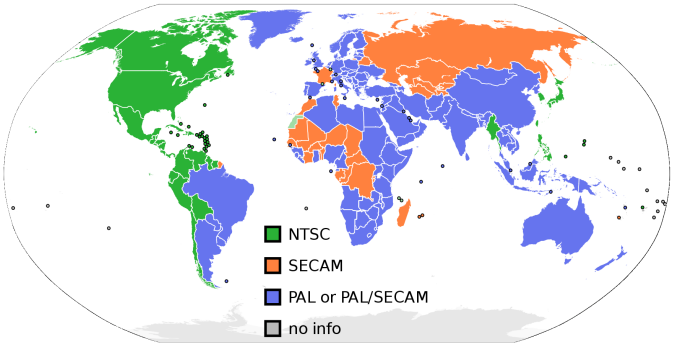


Figure 2.7: Distribution of analog television formats by nation

selected $N_V = 525$ scan lines[8] with interlaced scanning and the field rate of $f_{\frac{V}{2}} = 60$ Hz in agreement with the mains frequency.

- As an improved version of NTSC, the **PAL** system was introduced by the European Broadcasting Union (EBU) in 1967 and besides Europe it was adopted in Australia, South America, Africa and a part of Asia. The PAL format applied interlaced scanning with the number of scan lines of $N_V = 625$ along with the field rate of 50 Hz.

- Also as an improvement on NTSC, emerging from France the **SECAM** system was introduced in France and in the Soviet Union. The SECAM system will not be discussed in details in the present book.

The main parameters of the NTSC and PAL formats are summarized in Table 2.1. As a summary it can be stated that NTSC provides a higher temporal resolution (i.e. higher refresh rate/field rate) with lower spatial resolution, while PAL ensures a higher spatial resolution due to the increased number of scan lines at the cost of lower field rate.

The structure of an entire frame (i.e. two consecutive fields) in the PAL system is illustrated in Figure 2.8. Obviously, the figure depicts only a single video component, the content of the components is discussed in the following. The figure depicts the horizontal and vertical blanking intervals, with the VSYNC (denoted by green color) and

[8]The choice of the vertical line number was a compromise between the existing black and white systems, e.g. as used by RCA's NBC TV network, and the intention of manufactures, desiring to increase the line number to 600-800. The actual value originates from a limitation of the CRT technology of the day: In early TV receivers a master oscillator ran at twice the line-frequency, and this frequency ($2 \cdot 15750$ Hz) was divided down by the number of lines used (in this case $N_V = 525$) to give the field frequency ($f_{\frac{V}{2}} = 60$ Hz). This frequency was compared with the mains frequency (60 Hz in America) and the master oscillator's frequency was corrected by the discrepancy in order to avoid a moving noise image due to supply ripple. At the time, frequency division was performed use of a chain of multivibrators, the overall division ratio being the mathematical product of the division ratios of the chain. Since the line number has to be odd an odd number (see previous section), therefore it can be factorized only to odd numbers as well, which had do be relatively small in order to avoid thermal drift of the oscillator. The closest sequence up to about 500 lines that meets all these requirements is $525 = 3 \cdot 5 \cdot 5 \cdot 7$, giving the number of scan lines.

Table 2.1: Parameters of the NTSC and PAL analog formats

	NTSC	PAL
Total line number (N_{V}):	525	625
Number of active lines ($N_{\mathrm{V,A}}$):	480	576
Frame rate (f_{V}):	30 Hz (29.97 Hz)	25 Hz
Field rate ($f_{\frac{\mathrm{V}}{2}}$):	60 Hz (59.94 Hz)	30 Hz
Line frequency (f_{H}):	$525 \cdot 30 = 15750$ Hz (15734 Hz)	15625 Hz
Line period time (T_{H}):	63.49 μs (63.55 μs)	64 μs
Active line time ($T_{\mathrm{H,a}}$):	$\approx$ 52 μs (63.55 μs)	52 μs

the HSYNC (yellow interval) sync pulses. These pulses ensure the synchronization of the display to the received signal by triggering the vertical and horizontal retrace of the electron beam, therefore, it is ensured that the individual „pixel" values are displayed on the correct position. With the loss of these sync pulses the displayed image is misaligned vertically (in case of lost VSYNC), resulting in the **jitter** or horizontally (in case of lost HSYNC) causing the **rolling** of the video signal.

Although the characteristics of the video signal—with containing vertical and horizontal blanking intervals—originates from the operation principle of CRT displays, even in modern digital systems (e.g. in case of HDMI or SDI interfaces) the structure of the video signal is identical with the presented analog one. Obviously, modern LCD displays do not need the presence of blanking intervals at all, since they update the content of all the pixels quasi-instantaneously, still, the blanking intervals are present in digital video signals as well. One reason for this is that digital video signal is the legacy of the CRT era (and even today, professional CRT studio monitors are often employed), with the newly introduced standards all based on the previous ones. On the other hand the blanking intervals allow the transmission of auxiliary data, including teletext, captions, subtitles, and in case of digital standards, **audio streams** of the media content as well. The actual location of the digital data is codified in ITU-R BT.1364 and SMPTE 291M. As and example, the audio streams accompanying video data are positioned in the vertical blanking intervals in the HDMI standards, i.e. between the content of active lines [9].

[9] As a simple example for audio transmission via HDMI interface: In case of a HD format of 1080p (with

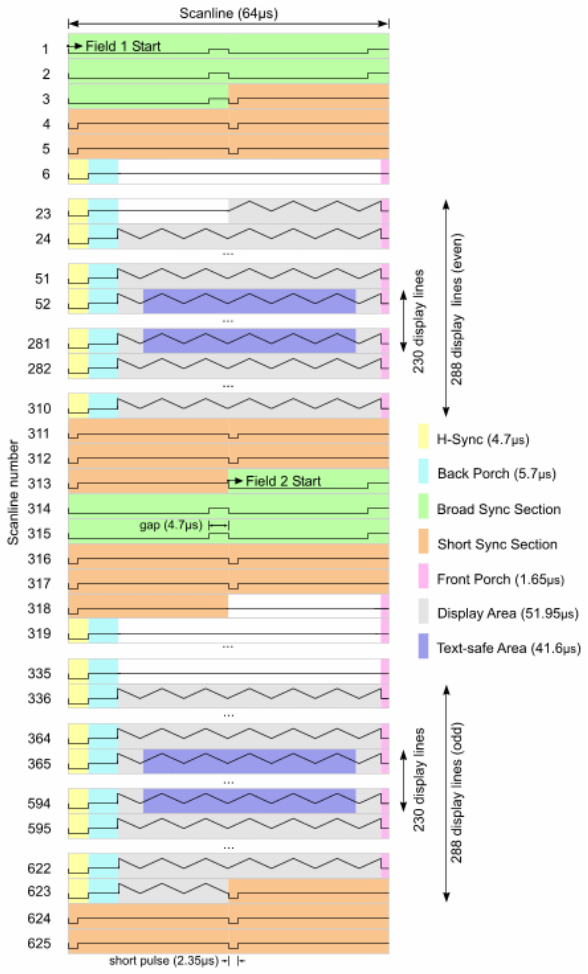


Figure 2.8: The line structure of an entire frame in case of interlaced scan, illustrated for a single video component:

- active line interval: grey
- horizontal blanking interval: magenta, cyan and yellow
- vertical blanking interval: green, orange and white

So far only the general structure of the video signals was discussed, without the investigation of the actual data in the active intervals, e.g. in case of analog video, without investigating what the actual video signals are. Obviously, the representation of one color pixel requires the transmission of three individual video components, which in the field of video transmission is most often the luma-chroma representation.

Based on the number of actual individual transmitted components video signals can be categorized to two main formats

- **Component video** transmits the video signal on three individual signal paths, i.e. on three individual cable pairs.

the total number of lines 1125), with the frame rate of 60 Hz, the line frequency is given by $f_V = 67.5$ kHz. Assuming an audio stream with 8 channels, sampled at $f_s = 192$ kHz the data to be transmitted over one entire frame is $\frac{8 \cdot 192000}{60} = 25600$ samples. This means the transmission of 23 samples transmitted with one video line, which results in the maximum bandwidth of the HDMI 1.0 standard.

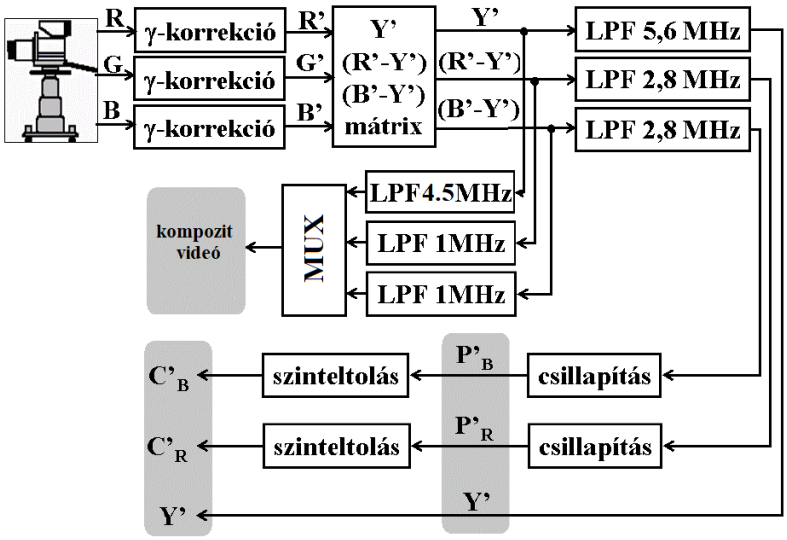


Figure 2.9: Block diagram of the production of composite and component video signals

- **Composite video** transmits the three video signals multiplexed into one single signal, transmitted over a single path, or physically speaking on one pair of cables.

The signal processing scheme of composite and component video is illustrated in Figure 2.9. As it is depicted, in both cases the basis of video representation is given by the luma-chroma separation of the RGB signals[10]. In the following the actual steps of this production chain is investigated.

Bandwidth of the video signals: Obviously, both the luma and chroma signals are transmitted with finite bandwidth.

As has been already discussed, in case of CRT displays the time domain video signal is drawn on the screen real-time, therefore, the bandwidth of the video signal (defined in Hz) limits the actual horizontal detailedness of the displayed on image. Thus, the decrease of temporal bandwidth results in the reduction of the spatial, horizontal resolution. Besides bandwidth efficiency at transmission, the actual choice of video bandwidth has a direct, practical reasons.

The screen of the CRT display is divided both horizontally and vertically to individual RGB pixels. Therefore, even in case of an continuous CRT driving voltage, the screen itself samples the video signals, that may lead to spatial aliasing artifacts, manifesting in visible Moiré patterns on the screen. In order to avoid aliasing the video

[10] Although in case of component video the direct transmission of RGB signals is also allowed in case of UHD video.

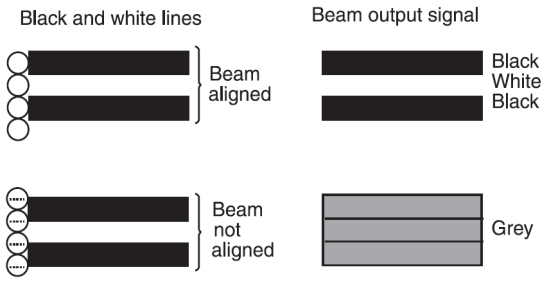


Figure 2.10: Illustration of the Kell effect.

signal has to be temporally bandlimited **at least** to the half of the spatial sampling frequency, in accordance with the sampling theorem.

The required temporal bandwidth is investigated for the case of the NTSC format: From table 2.1 the number of active lines in the format is $N_{V,A} = 480$, while the active line time is $t_{H,A} \approx 52\ \mu$s. As the aspect ratio is 4:3 the number of horizontal pixels is approximately $N_{H,A} = \frac{4}{3}N_{V,A} = 640$ (assuming square pixels). According to the sampling theorem the sine signal with the largest spatial frequency that can be represented with this resolution contains $N_{H,A}$ periods in one line (this is the consecutive „black-white-black-white..." content). The period time and the frequency of this signal can be calculated as

$$T_{\max} = \frac{t_H}{N_{H,A}/2}, \qquad f_{\max} = \frac{N_{H,A}}{2t_H} = 6.15\ \text{MHz} \tag{2.7}$$

which is the theoretical upper frequency limit, based on the Nyquist criterion.

However, experimental results showed that even this theoretical maximal frequency can not be displayed without artifacts, due to the **Kell effect**. The Kell effect is caused by the finite size of the CRT's electron beam: instead of the theoretical infinitely narrow beam, which is assumed when applying the sampling theorem, the electron beam has a **point spread function (PSF)** (i.e. the intensity profile, when projected to a single point on the screen) described approximately by a Gaussian distribution.

A simple example is illustrated in Figure 2.10 for the effect of the beam PSF: The continuous image to be displayed oscillates vertically at the Nyquist frequency, with alternating black and white lines. In case the lines of the image exactly coincides with the sampled lines on the screen (i.e. they are aligned) the image can be reconstructed perfectly. However, if the lines of the image are located between two actual scan lines, due to the non-zero extension of the electron beam PSF the CRT will display the average of the two lines, i.e. a homogeneous grey image is displayed.

As a conclusion, the theoretical maximal spatial frequency can not be displayed on the screen without artifact. The ratio of the experimental, subjective highest displayable spatial frequency and the theoretical limit is given by the **Kell factor**, being approximately 0.7 for the NTSC video parameters.

In order to avoid spatial aliasing and by taking the Kell effect into consideration the largest horizontal frequency that can be displayed on the screen without visible artifacts is given by

$$f_{\max}^{\text{NTSC}} = \frac{N_{H,A} \cdot K}{2t_H} = 4.3\ \text{MHz}, \quad f_{\max}^{\text{PAL}} = \frac{N_{H,A} \cdot K}{2t_H} \approx 5.2\ \text{MHz} \tag{2.8}$$

in the NTSC and PAL systems respectively, where the Kell factor is $K = 0.7$. Therfore, the luma signal (similarly to the early black and white TV signal) is bandlimited to 4.2 MHz in the NTSC and to 5 MHz in the PAL system.

As discussed earlier, the visual acuity (i.e. the spatial resolution) of the human visual system is less than the half for color information than for luminance. This property of human vision can be exploited in order to achieve significant video compression. For the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$ representation this served as the starting point for the subsampling of chroma signals. In the analog case this phenomena allows the bandwidth reduction of chroma signals, therefore, in component systems the chroma signals are bandlimited to the half of the luma signal (resulting in a halved horizontal color resolution), while in composite video the color information is transmitted with even more reduced bandwidth.

2.2.1 Composite video signal

Analóg átviteltechnika szempontjából a legegyszerűbb megoldás a videójel továbbítására a 3 videókomponens egyetlen érpáron való átvitele. Ebben az esetben a luma és chroma komponensekből egyetlen ún. **kompozit** jelet kell képzeni, hogy a vevő oldalon az eredeti három komponens különválasztható. A feladat megoldására három—alapgondolatában azonos—módszer létezik, az NTSC, PAL és SECAM megoldások. A rendszerek pontos működésétől eltekintve a következő bekezdés az NTSC és PAL kompozitjelek képzésének alapelvét mutatja be.

A kompozit formátum az NTSC rendszer bevezetésével került kidolgozásra a létező fekete-fehér TV-vevőkkel kompatibilis analóg színes műsorszórás megvalósítására. A feladat a már létező műsorszóró rendszerben alapsávban továbbított luma jelhez (azaz a fekete-fehér jelhez) a színinformáció olyan módú hozzáadása volt, hogy a létező monokróm vevőkben a többletinformáció minimális látható hatást okozzon, míg a színes vevő megfelelően külön tudja választani a luma és chroma jeleket. Tehát más szóval a visszafele-kompatibilitás miatt az új színes rendszerben a luma jelet változatlanul kellett átvinni. Minthogy az átvitelhez használt RF spektrum jelentős részét már elfoglalták a frekvenciaosztásban küldött egyes TV csatornák (a képinformáció, és az FM modulált hanginformáció), így a luma és chroma komponensek csak ugyanabban a frekvenciasávban kerülhetnek továbbításra.

Az alapsávi fekete-fehér TV jel felépítése egyszerű a már bemutatott 2.8 ábrán látható felépítéssel megegyező: Egymás után, soronként tartalmazza a CRT elektronágyú vezérlőfeszültségének időtörténetét, amely tehát így a műsor vételével teljesen valós időben rajzolja soronként a kijelző képernyőjére az $Y'(t)$ luma jel tartalmát. Az egyes sorok és képek kijelzése között az elektron-ágyú kikapcsolt állapotban véges idő alatt fut vissza a következő sor, illetve kép elejére. Egy fekete-fehér TV sor felépítése az 2.11 ábrán látható.

A valós idejű átvitel/kijelzés elvéből látható, hogy a színinformáció átvitele időosztásban sem lehetséges, tehát a chroma jeleket a luma jelekkel azonos frekvenciasávban és időben szükséges átvinni. A megoldás tárgyalása előtt vizsgáljuk külön a chroma jelek továbbításának módját.

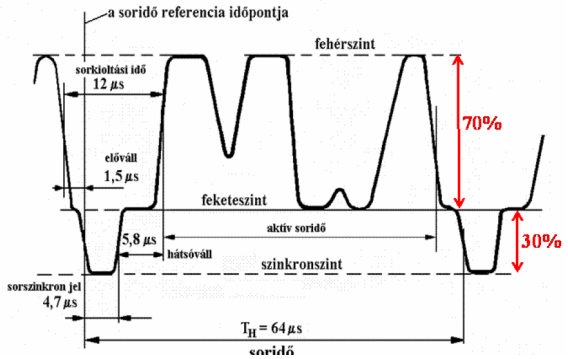


Figure 2.11: Egyetlen TV sor luma jele és szinkron jelei a PAL rendszer időzítései mellett. Az NTSC esetében a TV sor felépítse jellegere teljesen azonos, a PAL-tól eltérő időzítésekkel.

A színsegédvivő bevezetése: A színformációt hordozó két chroma jel $(Y'(t) - R'(t), Y'(t) - B'(t))$ egyidőben történő átvitele során alapvető feladat a két analóg jel egyetlen jellé való átalakítása. Erre az kvadratúra amplitúdómoduláció ad lehetőséget, amely egy olyan modulációs eljárás, ahol az információt részben a vivőhullám amplitúdójának változtatásával, részben annak fázisváltoztatásával kódoljuk (ezzel tehát két független jel vihető át egyszerre). Mind PAL, mind NTSC rendszer esetében az emberi látás színekre vett alacsony felbontását kihasználva a chroma jeleket erősen (PAL esetében pl. a luma jel ötödére, 1 MHz-re) sávkorlátozzák, ezzel az apró, nagyfrekvencián reprezentált részleteket kisimítják. Ezután a kvadratúramodulált chroma jeleket pl. PAL esetén

$$c^{\mathrm{PAL}}(t) = \underbrace{U'(t)}_{(B'-Y')/2.03} \cdot \sin\omega_c t + \underbrace{V'(t)}_{(R'-Y')/1.14} \cdot \cos\omega_c t \tag{2.9}$$

alakban állíthatjuk elő, ahol $\sin\omega_c t$ az ún. **színsegédvivő**, ω_c a színsegédvivő frekvencia, $U'(t)$ az ún. fázisban lévő, $V'(t)$ pedig a kvadratúrakomponens. A kvadratúramodulált színjelek tehát egyszerűen az átskálázott színkülönbségi jelek fázisban és kvadratúrában lévő színsegédvívővel való modulációjával állítható elő.

A színjelek demodulációja koherens (fázishelyes) vevővel egyszerű alapsávba való lekeveréssel és aluláteresztő szűréssel valósítható meg:

$$\sin x \cdot \sin x = \frac{1-\cos 2x}{2}, \qquad \cos x \cdot \cos x = \frac{1+\cos 2x}{2}$$
$$\sin x \cdot \cos x = \frac{1}{2}\sin 2x \tag{2.10}$$

trigonometrikus azonosságok alapján $U'(t)$ demodulációja

$$c^{\mathrm{PAL}}_{\mathrm{QAM}}(t) \cdot \sin\omega_c t = U'(t) \cdot \sin\omega_c t \cdot \sin\omega_c t + V'(t) \cdot \cos\omega_c t \cdot \sin\omega_c t =$$
$$\frac{1}{2}U'(t) - \underbrace{\cancel{\frac{1}{2}U'(t)\cos 2\omega_c t + V'(t) \cdot \frac{1}{2}\sin 2\omega_c t}}_{\text{aluláteresztő szűrés}} \tag{2.11}$$

szerint történik, míg $V'(t)$ demodulálása hasonlóan $\sin\omega_c t$ lekeverés szerint. A megfelelő demodulációhoz tehát a vevőben a színsegédvivő fázishelye, koherens előállítása elengedhetetlen.

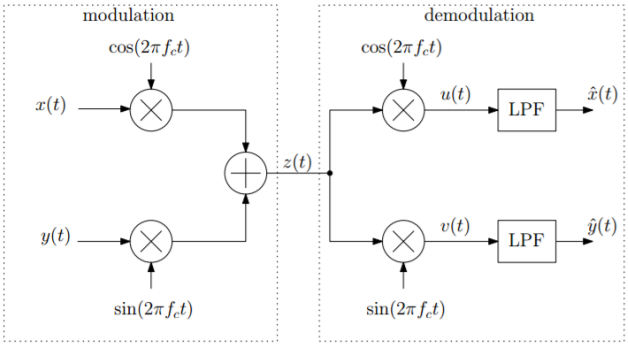


Figure 2.12: QAM moduláció és demoduláció folyamatábrája

Az NTSC rendszerben a PAL-hoz hasonlóan a színjelek

$$c_{\mathrm{QAM}}^{\mathrm{NTSC}}(t) = I'(t) \cdot \sin \omega_c t + Q'(t) \cdot \cos \omega_c t \tag{2.12}$$

alakban kerültek átvitelre, ahol az in-phase és kvadratúra komponensek rendre

$$\begin{aligned} I'(t) &= k_1(R' - Y') + k_2(B' - Y), \\ Q'(t) &= k_3(R' - Y') + k_4(B' - Y). \end{aligned} \tag{2.13}$$

A k_{1-4} konstansokat úgy választották meg, hogy az in-phase és kvadratúra modulált jelek nem közvetlenül a kék és piros merőleges bázisvektorok (ld. 1.11 ábra), hanem ezek kb. $+20^\circ$ elforgatottja. Az így kapott új tengelyek a magenta-zöld és türkiz-narancssárga tengelyek a közvetlen modulálójelek. Ennek oka, hogy úgy találták, az emberi látás felbontása jóval nagyobb türkiz-narancssárga közti változásokra, mint a magenta-zöld között. Ezt kihasználva a magenta-zöld $Q'(t)$ színjeleket az $I'(t)$ jelhez képest is jobban sávkorlátozták, sávszélesség-takarékosság céljából. A PAL rendszer bevezetésének idejére azonban kiderült, hogy ez rendszer felesleges túlbonyolítása, így az új rendszerben megmaradtak az eredeti színkülönbségi jelek modulációjánál.

Vizsgáljuk végül a modulált színjel fizikai jelentését, az egyszerűség kedvéért $c^{\mathrm{PAL}}(t)$ esetére (PAL rendszerben)! Az (2.9) egyenlet egyszerű trigonometrikus azonosságok alapján átírható a

$$c_{\mathrm{QAM}}^{\mathrm{PAL}}(t) = \sqrt{U'(t)^2 + V'(t)^2} \, \sin\left(\omega_c t + \arctan \frac{V'(t)}{U'(t)}\right) \tag{2.14}$$

polár alakra. Minthogy a moduláló U', V' jelek a színkülönbségi jelekkel arányosak, így a fenti kifejezést (1.27) és (1.26)-val összehasonlítva megállapítható, hogy a QAM modulált jel egy olyan szinuszos vivő, amelynek pillanatnyi amplitúdója a továbbított színpont telítettségét, pillanatnyi fázisa a színpont színezetét adja meg.

A színsegédvivő amplitúdójának és fázisának egyszerű értelmezhetősége miatt az NTSC és PAL jeleket gyakran vizsgálták ún. vektorszkóp segítségével jól meghatározott vizsgálóábrák megjelenítése mellett. A vektorszkóp kijelzője gyakorlatilag a 1.11

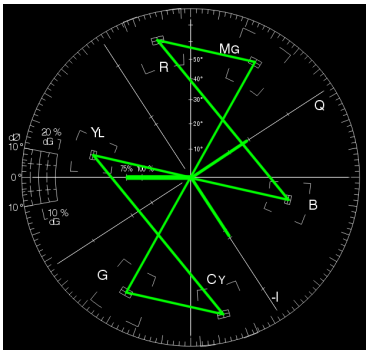
(a)  (b) 

Figure 2.13: Egy gyakran alkalmazott vizsgálókép (SMPTE color bar) (a) és vektorszkóppal ábrázolva (b).

ábrán is látható $B' - Y'$, $B' - Y'$ térben jeleníti meg a teljes képtartalom (azaz egyszerre az összes képpont) chroma jeleit, Y'-tól függetlenül a demodulált chroma-jelek megjelenítésével. A vektorszkóp gyakorlatilag egy olyan oszcilloszkóp, amelynek x kitérését a demodulált $B' - Y'$, y-kitérést a demodulált $R' - Y'$ jel vezérli, így a teljes képtartalom színezetét szinte egyszerre jeleníti meg az előre felrajzol vizsgálati rácson. Egy tipikus vizsgáló ábra és annak vektorszkópos képe látható a 2.13 ábrákon. A vektorszkóp alkalmazásának előnye, hogy az esetleges amplitúdó és fázishibából származó telítettség és színezethibák jól láthatóvá válnak a kijelzőn az egyes felvetített pontok "összeszűkülése/tágulása", illetve a teljes konstelláció elfordulásaként. Megjegyezhető, hogy a mai digitális videojeleket is gyakran ábrázolják szoftveres vektorszkópon az egyes pixelek színezetének vizsgálatához.

A színsegédvivő frekvencia: Vizsgáljuk most, hogyan választható meg a színsegédvivő ω_c vivőfrekvenciája úgy, hogy a QAM modulált $c^{\mathrm{PAL}}(t)$ jelet a luma jelhez hozzáadva a vevő oldalon lehetséges legyen a vett $c^{\mathrm{PAL}}(t) + Y'(t)$ jelből az eredeti chroma és luma jelek szétválasztása!

A jelek vevőoldali szétválasztására a luma és chroma jelek spektruma ad lehetőséget: Láthattuk, hogy a videójel az egyes TV sorokban megjelenítendő világosság és színinformáció sorfolytonos időtörténeteként fogható fel. Természetes képeken a képtartalom sorról sorra csak lassan változik (természetesen a képtartalomban jelenlévő vízszintes éleket leszámítva), így mind a luma, mind a chroma jelek ún. kvázi-periodikusak, azaz közel periodikusak. Jel- és rendszerelméleti ismereteink alapján tudjuk, hogy egy periodikus jel spektruma vonalas, a jelfrekvencia egész számú többszörösein tartalmaz csak komponenseket. Ennek megfelelően mind a luma, mind a chroma jelek spektruma közel vonalas: az energiájuk a sorfrekvencia egész számú többszörösein csomósodik. Természetesen a luma jel az alapsávban helyezkedik el (0 Hz környezetében), kb. 5.6 MHz sávszélességben[11]. A QAM modulált chroma jel spektruma a sávkorlátozás miatt keskenyebb (1 MHz), és középpontját ω_c vivőfrekvencia határozza meg.

[11] Ez a sávszélesség eredményezi az azonos horizontális és vertikális képfelbontást.

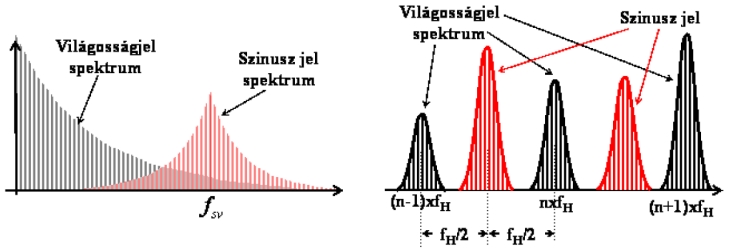


Figure 2.14: A luma és chroma jelek spektrális közbeszövésének alapelve a teljes spektrumokat ábrázolva (a) és a spektrális csomókat felnagyítva (b)

A luma-chroma jel összegzése ennek ismeretében egyszerű: Az ω_c vivőfrekvencia megfelelő megválasztásával elérhető, hogy a chroma jel spektrumvonalai (spektrumcsomói) éppen a luma jel spektrumvonalai közé essen, azaz a spektrumukat átlapolódás nélkül közbeszőhetjük. Az eljárás alapötletét 2.14 ábra illusztrálja f_{H}-val a sorfrekvenciát jelölve. A szétválaszthatóság feltétele ekkor

$$f_c = f_{\mathrm{H}} \cdot \left(\mathrm{n} + \frac{1}{2}\right), \qquad \mathrm{n} \in \mathcal{N} \tag{2.15}$$

azaz a színsegédvivő frekvenciája a sorfrekvencia felének egész szűmú többszörösének kell, hogy legyen [12].

A CVBS kompozit videójel és luma-chroma szétválasztás: Ennek ismeretében végül a teljes kompozitjel a

$$\mathrm{CVBS}(t) = \mathrm{Sy}(t) + Y'(t) + c_{\mathrm{QAM}}(t) \tag{2.16}$$

alakban áll elő, ahol Y' a luma jel, c_{QAM} a QAM modulált chroma jelek és $\mathrm{Sy}(t)$ a kioltási időben jelen lévő sorszinkron és képszinkron jelek. A CVBS elnevezés gyakori szinoníma a kompozit videójelre, jelentése C: color, V: video (luma), B: blanking (azaz kioltás) és S: sync (azaz szinkronjelek).

Az így létrehozott videójel a fekete-fehér képhez képest csak a modulált színsegédvivőt tartalmazza többletinformációnak. Egyszerű fekete-fehér vevőn a CVBS jelet megjelenítve a színinformáció nagyfrekvenciás zajként, pontozódásként (ún. chroma dots) jelenik csak meg a kijelzőn, így a visszafelé kompatibilitás biztosítva volt. Színes vevőkben a CVBS jelből a luma és chromajel elméletileg fésűszűréssel szeparálható a sorfrekvencia felének egész számú többszöröseit elnyomva. Ez ideálisan egy soridejű késleltetést igényel [13]. A fésűszűrős luma-chroma szeparáció lehetősége már az NTSC fejlesztésének idején ismert volt, azonban a szükséges soridejű késleltető nem

[12] Megjegyezhető, hogy PAL esetében az előre adott sorfrekvenciához egyszerű volt a színsegédvivőfrekvencia megválasztása, míg NTSC esetén bizonyos okok miatt a sorfrekvencia és ebből következően a képfrekvencia megváltoztatására volt szükség. Innen származnak a ma is használatos 59.94 és 29.97 Hz képfrekvenciák, amelyeket a következő fejezet tárgyal részletesen.

[13] A bizonyításhoz vizsgáljuk $h(t) = \delta(t) + \delta(t - t_{\mathrm{H}})$ szűrő átviteli karakterisztikáját, amely szűrő a jelből kivonja t_H-val késleltetett önmagát!

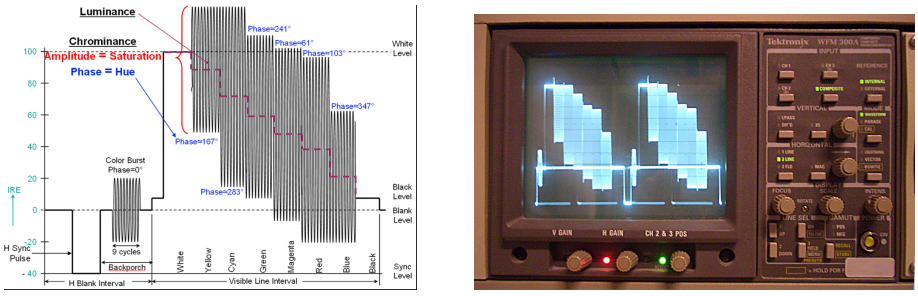


Figure 2.15: Az SMPTE color bar vizsgáló ábrának egy, illetve két sorának hullámformája sematikusan (a), és egy hulláforma monitoron (b) vizsgálva

állt rendelkezésre, ezért a korai NTSC vevők egyszerű alul/felüláteresztő szűrőkkel, vagy egyszerű chroma jelre állított lyukszűrőkkel szeparálták a luma-chroma jeleket. Ennek eredményeképp még a színes vevőkben is a chroma jelen kisfrekvenciás tartalomként jelen lehetett a világosságinformáció látható hatással a megjelenített képre. A megfelelő analóg PAL fésűszűrő-tervezés még a 90-es években is aktív K+F alatt álló terület volt.

Az elmondottak alapján az NTSC rendszerben a 2.13 ábrán látható vizsgálóábrának egy sorának kompozit ábrázolását a 2.15 mutatja be jellegre helyesen, és egy konkrét hullámforma monitoron mérve. Az ábrán megfigyelhető az egyes oszlopokhoz tartozó hullámalak: látható, hogy a csökkenő világosságú oszlopokra (amelyek világosságát szaggatott vonal jelzi) hogyan ültették rá a QAM modulált chroma jeleket. Az első és utolsó fehér, illetve fekete oszlop esetén a chroma jelek amplitúdója zérus (fehérpont), egyéb esetekben a szinuszos színsegédvivő amplitúdója az oszlopok színének telítettségével, fázisa a színezetükkel arányos. Megjegyezzük, hogy a tényleges hullámforma már átskálázott chroma jeleket ábrázol, amely átskálázás épp azért történik, hogy a teljes CVBS jel beleférjen a fizikai interface dinamikatartományába (ez természetesen a nagy telítettségű színek esetén okozna problémát). Ez magyarázza tehát az eddig figyelmen kívül hagyott 2.03 és 1.14 skálafaktorokat pl. (2.9) esetében.

Az NTSC jel felépítése alapján egyértelmű, hogy a megfelelő színek helyreállításához a vevőben a színsegédvivő fázisának nagyon pontos ismerete szükséges. Ahhoz, hogy ez biztosítva legyen a sorkioltási időben az ún. hátsó vállra (ld. 2.11 ábra) beültetésre került néhány periódusnyi (9) képtartalom nélküli referenciavivő, az ún. color burst, vagy burst jel. Ez a burst jel megfigyelhető a 2.15 ábrán is.

Ennek ellenére az NTSC rendszer továbbra is fázisérzékeny volt, hiszen fázishibát a vevőben is bármelyik alkatrész okozhatott. A QAM moduláció jelege miatt már a legkisebb fázishiba is látható színezetváltozást okozott a megjelenített képen. A PAL rendszer tervezésének egyik fő célja épp ezért a rendszer fázishibára vett érzékenységének csökkentése volt

A PAL rendszer: Míg az egyszerű NTSC rendszer már 1953-ban bevezetésre került Amerikában, addig Európában egészen az 1960-as éveikg vártak a színes műsorszórás bevezetésére. Ennek oka, hogy az eltérő hálózati frekvencia miatt az NTSC-t nem

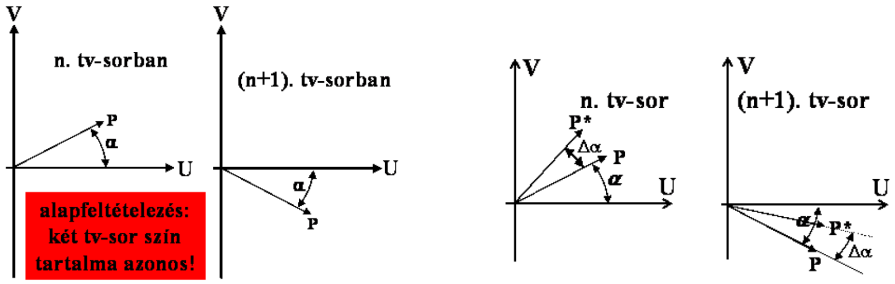


Figure 2.16: Az SMPTE color bar vizsgáló ábrának egy, illetve két sorának hullámformája sematikusan (a), és egy hulláforma monitoron (b) vizsgálva

lehetett egy az egyben átemelni Európába (ld. később). Mire az európai rendszert kifejlesztették, az NTSC rendszer jó néhány gyengeségére fény derült, így az újonnan kifejlesztett PAL (Phase Alternate Lines) ezek kijavítását célozta főként meg. Ennek eredményeképp a PAL rendszer más QAM modulációval dolgozik (a chroma jelek közvetlenül a modulálójelek), eltérő a színsegédvivő frekvencia, és legfontosabb újításként: egy egyszerű megoldással szinte érzéketlen a fázishibára.

Láthattuk, hogy a vevő oldalán bármilyen fázishiba a színezet jól látható torzulását okozza. Mivel a fázishiba gyakran elkerülhetetlen, ezért hatásának kiküszöbölésére a PAL rendszer a következő egyszerű megoldást alkalmazza:

- Az adó oldalon (a PAL jel létrehozása során) képezzük QAM moduláció során a V' chromajel előjelét minden második TV-sorban negáljuk meg, azaz sorról sorra fordított előjellel vigyük át (ez ekvivalens a sorról sorra változó $\pm\cos\omega_c t$ vivővel való modulációval)! Az eljárás szemléltetésére tegyük fel, hogy két egymás utána sorban minden horizontális pozícióban a színinformáció azonos. Ekkor egy adott pontra az n. és (n+1). sorban átvitt U', V' jeleket a 2.16 (a) ábra szemlélteti pl egy lila képpont átvitele esetén.

- Tegyük fel, hogy a vevő oldalon a vett jelhez $\Delta\alpha$ fázishiba adódik az átvitel és demoduláció során. Természetesen a fázishiba hatására az így vett színvektor mind az n., mind az (n+1). sorban azonos irányba fordul az $U'-V'$ konstellációs diagramon (azaz a $R'-Y', B'-Y'$ síkon), ahogy az a 2.16 (b) ábrán látható.

- A vevő oldalán forgassuk vissza minden második sorban a vett V' komponens előjelét és képezzük az (n+1). sor és az n. sor átlagát. Ezzel természetesen a színjelek vertikális felbontását csökkentjük (az átlagképzés az apró részleteket elsimítja), azonban ennek eredménye az emberi szem színezetre vett felbontása eredményeképp az információveszteség nem látható (a horizontális felbontás már egyébként is jelentősen lecsökkent az egyszerű sávkorlátozás hatására). Könnyen belátható, hogy a két vektor átlagát képezve éppen az eredeti, hibamentes színvektort kapjuk eredményül. Két sor esetén azonos sortartalom esetén tehát ezzel az egyszerű trükkel a fázishiba hatása teljesen kiküszöbölhető, míg levezethető, hogy változó sortartalom esetén a fázishiba az átlagvektor hosszá-

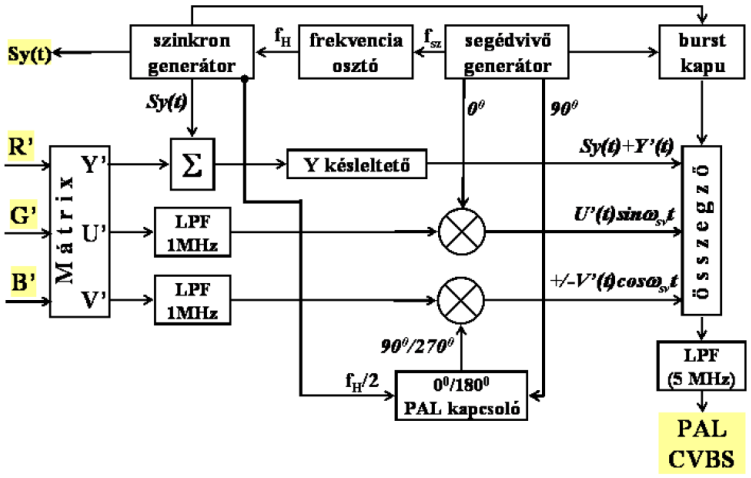


Figure 2.17: A PAL kódoló felépítése

nak csökkenését okozza, tehát színezetváltozás helyett csak telítettségváltozást okoz.

A bemutatott módosított modulációs módszerrel még aránylag nagy fázishibák hatása is minimális hatással van a megjelenített képre. Az ok, hogy mégis több, mint egy évtizedet kellett várni a PAL rendszer bevezetésére az volt, hogy a módszer alkalmazásához (az átlagolás elvégzéséhez) a videójel soridejű késleltetésére volt szükség. Ez az 50-es években analóg módon nem megoldható probléma volt amely a PAL implementálását hátráltatta.

A PAL bevezetését végül az olcsón tömeggyártható ún akusztikus művonalak megjelenése tette lehetővé. Ez az akusztikus művonal, vagy PAL delay line egy egyszerű üvegtömb, amelyre egy piezo aktuátor és piezo vevő csatlakozik. Az adó a TV chroma jelével arányos mechanikai rezgéseket (ultrahang) hoz létre, amely többszörös visszaverődések után épp egy soridőnyi késleltetést szenvedve ér a vevő elektródához. Az ultrahang alapú késleltetővonalak egészen a 90-es évek végéig a PAL dekóderek részét képezték.

Az egyszerű PAL kódoló felépítése az eddig elmondottak alapján a 2.17 ábrán látható. Röviden összefoglalva, mind a PAL, mind NTSC esetén a kompozitjel létrehozása során a feladat a Gamma-torzított R', G', B' jelekből az Y', U', V (PAL) és Y', I', Q' (NTSC) jelek létrehozása, majd az U', V' és $I'Q'$ jelek megfelelő QAM modulációja. Az így létrehozott jeleket összeadva és a kioltási időben továbbított szinkronjelekkel ellátva előáll a CVBS kompozit jel.

A kompozit videójel fizikai interface megvalósítása szabványról szabványra változó. Konzumer felhasználás (pl. kézikamerák, videólejátszók, DVD lejátszók) szempontjából a legelterjedtebb csatlakozó a sárga jelölésű RCA végződés, amely az esetleges kísérő hangtól szigetelve, külön érpáron továbbítja a kompozit videójelet.

A kompozit és komponens jelek közti kompromisszumként az S-video formátum

Figure 2.18: Konzumer alkalmazásokhoz használt sárga jelölésű RCA csatlakozó (a) és a luma-QAM chroma jeleket külön érpáron átvivő S-videó csatlakozó (b)

a luma és chroma jeleket külön érpáron viszi át. Ezt leszámítva az interface jele teljesen a kompozit videóval azonosak, továbbíthat akár NTSC, akár PAL (akár SECAM) videókomponenseket: A luma tehát változatlanul alapsávban, míg a chroma a színsegédvivővel modulálva kerül átvitelre. A chroma jelek modulációja elkerülhetetlen, hiszen a két független színkülönbségi jel egy érpárra való ültetéséhez azokat legalább a sávszélességükkel megegyező frekvenciájú vivőjellel való moduláció szükséges az átlapolódás elkerüléséhez. Az S-video szabvány csatlakozója a 2.18 (b) ábrán látható.

2.2.2 Component video signal

The idea of transmitting the video signal by separated components is straightforward, still, it was allowed by technology only decades after composite video was introduced: While interfaces for device-to-device video transmission could be achieved even with analog data, the broadcasting of component video could be only resolved with the introduction of digital video standards. In case of component video the transmitted signals are directly the luma and the chroma signals (or less often the $R'G'B'$ signals), or more precisely the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$ components.

The $Y'P_{\mathrm{B}}P_{\mathrm{R}}$ representation consists of the three following signals:

- Y': the luma component, calculated by the luma coefficients defined by the RGB primaries. The required sync pulses are added to the luma component. Therefore, a device with composite video output can be connected to the Y' interface of a display with component input. As a result the black and white image is displayed with the modulated chroma signal resulting in high frequency noise („chroma dotting") on the screen.

- P'_{B}, P'_{R}: the $B'-Y'$, $R'-Y'$ chroma components, rescaled to the actual physical interface, usually requiring the dynamic range of ± 0.5 V. Notation P is the legacy of the composite systems, where color information is carried by the **P**hase of the subcarrier.

Similarly to composite video, the actual parameters of the component formats depend on the region of application: components formats can be considered as the transmission of NTSC and PAL video formats without the QAM modulation of the chroma signals. Therefore, both 625 and 525 scan line interlaced component formats exist with 60 Hz and 50 Hz field rate. A simple example is depicted for the separately trans-

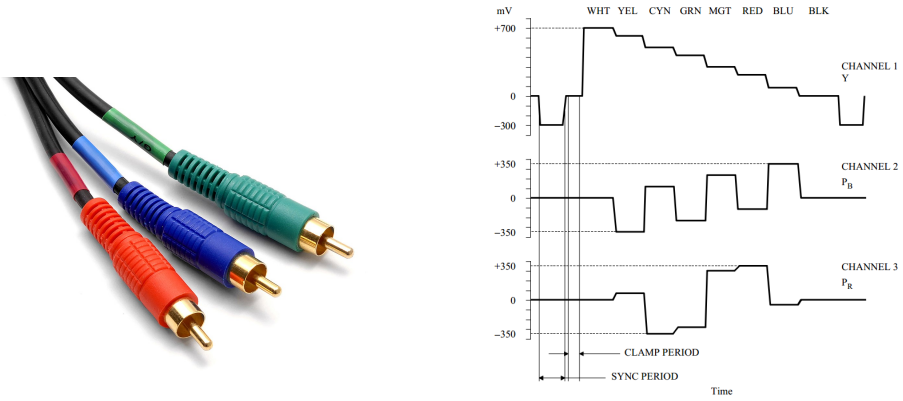


Figure 2.19: RCA connectors, applied frequently for the physical interface of component video (a) and the component video signals of the bar test pattern (b)

mitted $Y'P_BP_R$ signals in case of the SMPTE color bar test pattern is depicted in Figure 2.19 (b).

In consumer electronic the component video interface is equipped with RCA connectors, with the red and blue cables carrying the red and blue chroma signals and the green connector transmitting the luma information.

Finally, from the numerous analog video interfaces two most widespread are mentioned here:

- The SCART connector (or EuroSCART, shorthand for Syndicat des Constructeurs d'Appareils Radiorécepteurs et Téléviseurs) was designed for the transmission of bidirectional composite, S-video and RGB components in the same time, with also carrying stereo audio and digital signing signals in the same time. Before the advent of the HDMI interface SCART connectors also allowed the transmission of high definition 1080p format via $Y'P_BP_R$ component signals. The typical 21-pin SCART connector is depicted in Figure 2.20 (a).

- The VGA (Video Graphics Array) connector is still a commonly used analog component interface between video cards and displays (external monitors, projectors, etc.). The VGA interface transmits analog RGB components (in the color space of the video card) along with dedicate vertical and horizontal sync (VSYNC and HSYNC) cables. Nowadays the VGA interface is superseded with DVI, HDMI and DisplayPort digital interfaces.

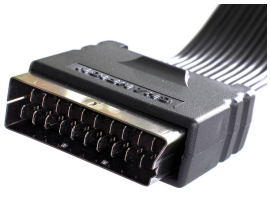


Figure 2.20: The analog composite/component SCART (a) and RGB component VGA (a) connectors

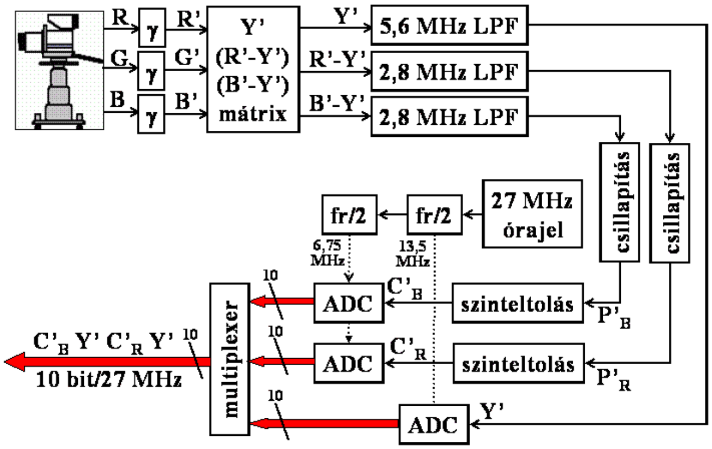


Figure 2.21: Production of SD video format by the digitization of $Y'P_{\mathrm{B}}P_{\mathrm{R}}$components

2.3 Digital video formats

2.3.1 The SD format

The first digital video format was developed by the ITU (International Telecommunication Union) and published in Rec. 601 (or ITU-601) in 1982[14].

The SD format is basically the digital representation of the component analog formats, discussed in the foregoing: The structure of digital video is obtained by the sampling of the analog video signal, as depicted in Figure 2.8, containing all the blanking intervals besides the active video content. Therefore, digital video is the direct digitized version of the $Y'P_{\mathrm{B}}P_{\mathrm{R}}$signals. As already discussed in the previous chapter, the components of the video format are the digital representation of the color pixels, referred to as the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$components. The signal processing scheme of digital video representation is illustrated in Figure 2.21.

The digitization of the analog video signal consists of two main steps:

- sampling of the continuous video signal with a-priori antialiasing filtering
- quantization of the continuous signal levels to discrete codes

[14]The CCIR (the predecessor of ITU) received the 1982–83 Technology and Engineering Emmy Award for its development

The questions of quantization, i.e. digital representation of the individual pixels has been already discussed in the previous chapter. Still, the parameters of sampling—or more specifically the sampling frequency—needs to be specified, defining the number of pixels per line, which, along with the number of lines gives the spatial resolution of the digital format.

Sampling frequency of the video signal:
The sampling frequency of the analog video components was chosen based on the following considerations:

- For several decades analog formats varied between from region to region due to the co-existence of NTSC, PAL and SECAM systems. As a straightforward endeavour, a basic goal was to find a common sampling frequency, compatible with all the existing analog formats. Furthermore, orthogonal sampling was an obvious need, meaning that in each system a single scan line should contain an integer number of samples (pixels). Mathematically these requirements can be formulated, as the line period time has to be dividable by the sampling period, and equivalently the sampling frequency has to be the multiple of the line frequency both in the PAL and the NTSC system, satisfying

$$f_s = n \cdot f_H^{\mathrm{PAL}} = m \cdot f_H^{\mathrm{NTSC}}, \tag{2.17}$$

  where n, m are integers. The line frequencies in the two systems are given by

$$f_H^{\mathrm{PAL}} = 25 \cdot 625 = 15625 \text{ Hz} \tag{2.18}$$

$$f_H^{\mathrm{NTSC}} = 30 \cdot \frac{1000}{1001} \cdot 525 = 15734.2 \text{ Hz}, \tag{2.19}$$

  with the smallest common multiple being

$$144 \cdot f_H^{\mathrm{PAL}} = 143 \cdot f_H^{\mathrm{NTSC}} = 2.25 \text{ MHz}. \tag{2.20}$$

  Therefore, the sampling frequency must be chosen as the multiple of 2.25 MHz.

- On the other hand, according to the sampling theory, the sampling frequency has to be at least twice the bandwidth of the sampled signal in order to avoid aliasing artifacts. In the foregoing it was discussed that—by taking also the Kell effect into consideration—the bandwidth of the luma signal is around 5.6 MHz, with the chroma bandwidth being the half of it.

The smallest frequency satisfying both requirements is given by

$$f_s^{\mathrm{Y,SD}} = 13.5 \text{ MHz}, \tag{2.21}$$

being chosen as the sampling frequency of the luma signal of standard definition video. As the chroma signals are bandlimited to the half of luma signal due to the lower visual acuity of the HVS, therefore, the chroma signals are sampled with halved sampling frequency ($f_s^{\mathrm{Ch,SD}} = 6.75$ MHz).

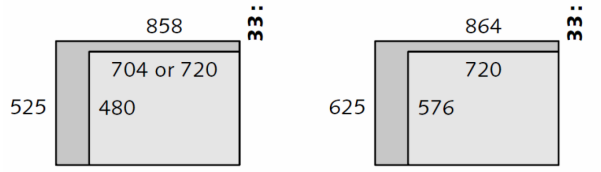


Figure 2.22: The SD video format in the American and European systems

The number of pixels in a scan line—given by the line period time divided by the sampling interval—is $N_{H,t} = \frac{T_H}{1/f_s} = 858$ pixels in the NTSC and 864 in the PAL system, including the horizontal blanking intervals as well. For the sake of further unification of the standard definition format the number of active pixels in a line was chosen to be 720 pixels in both the American and European systems. Note that the active line period is 52 μs in both systems, from which the number of active pixels is $N_{H,a} = T_{\mathrm{H,a}} \cdot f_s = 702$. Hence, the 720 pixels contains a small interval from the blanking time as well, due to the uncertainty in the position of the actual analog video signal, distortions and smears near the edges, originating from analog processing. The actual digital video data is taken as 704 pixels from the center of the nominal 720 pixels. Obviously, the number of the vertical total and active pixels are simply given by the number of total and active scan lines.

The spatial resolution/pixel number of the SD formats is illustrated in 2.22. The two formats are named after the number of the active lines, resulting the the two SD formats being **480i** and **576i**, with „i" referring to the fact that for both formats only interlaced scan is defined.

As it has been already discussed, the aspect ratio of the reproduced SD images should have the aspect ratio of 4:3, which is termed as the **display aspect ratio (DAR)**. However, the ratio of the number of the horizontal and vertical pixel numbers—termed as the **storage aspect ratio (SAR)** does not coincide this ratio in neither formats. The target display aspect ratio, therefore, can be achieved only by applying non-square pixels on the display side, described by the **pixel aspect ratio (PAR)**, defined as the ratio of the horizontal and vertical pixel size. Therefore, format 480i has a pixel aspect ratio of 10:11, and 576i has the PAR of 12:11 with the DAR of 4:3 for both cases. In contrary, computer displays are employing squared pixels with the PAR of 1:1, and the standard VGA resolution being 640x480 pixels.

As a short summary of the foregoing the properties of the SD formats are the following:

- The chromaticity of the primaries and the gamut of the device RGB color space is illustrated in Figure 2.23 (a). The white point of the colors space is D65 white.

Table 2.2: Spatial and temporal parameters of SD video formats

	480i	576i
active pixel number:	704 x 480	704 x 576
total pixel number:	858 x 525	864 x 625
display aspect ratio:	4:3	4:3
pixel aspect ratio	10:11	12:11
field rate:	59.94 Hz	50 Hz
frame rate:	29.97 Hz	25 Hz

The luma is calculated from the nonlinear RGB components as [15]

$$Y' = 0.299R' + 0.587G' + 0.112B'. \tag{2.22}$$

- In order to achieve perceptual quantization the source RGB signals are predistorted with the opto-electronic transfer function, given by

$$E = \begin{cases} 4.500L, & \text{ha } L < 0.018 \\ 1.099L^{0.45} - 0.099, & \text{ha } L \geq 0.018, \end{cases} \tag{2.23}$$

  where $L \in \{R, G, B\}$. The entire curve can be approximated by $L^{0.5}$.

- The original SD display aspect ratio was 4:3. After the introduction of the HD format also the SD format was extended with the aspect ratio of 16:9.

- SD format exclusively supports interlaced scanning.

- ITU-601 originally only allowed the chroma subsampling scheme 4:2:2. Later it was extended with the scheme 4:2:0 for consumer applications.

- The SD format applies the sampling frequency of $f_s = 13.5$ MHz. The format strictly prescribes the antialiasing low-pass filters, bandlimiting the luma signal to 5.75 MHz as the upper frequency limit and 6.75 MHz as the lower limit of the stopband. The chroma is sampled with halved sampling frequency, and prefiltered with corresponding antialiasing filters.

[15] It is noted here that the above coefficients—unmathematically—do not coincide with the second row of the $XYZ \rightarrow RGB$ transformation matrix (which can be calculated from the primaries and the white point). This means that strictly speaking the Y component—calculated from the RGB values with the above coefficients—does not equal to the relative luminance. Instead, for the sake of simplicity the luma coefficients of the NTSC system were used. These mathematical inconsistencies are usual in videotechnologies.

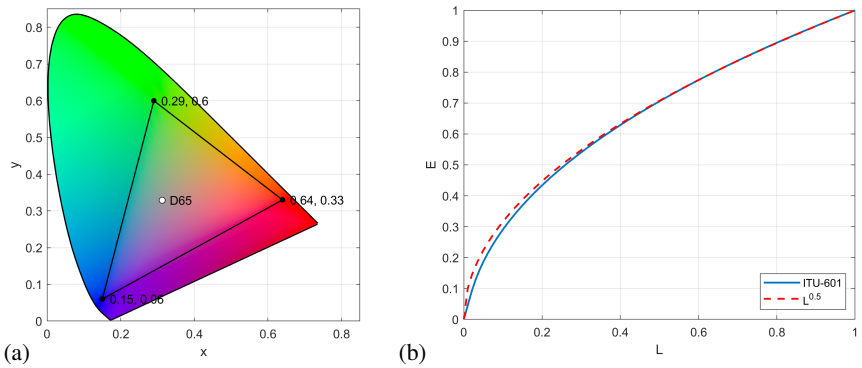


Figure 2.23: Gamut (a) and OETF (b) of the ITU-601 SD format

- The sampled luma and chroma sampled are quantized with 8 (for consumer electronics and broadcasting applications) or 10 bits (in studio applications) of bit depths.

2.3.2 The HD format

The starting point for the introduction of the analog video formats, leading directly to SD digital format was to cover approximately about 10° of the central vision from the viewer's field of view. Obviously, this requirement does not directly define the display size, or the spatial resolution. Instead at a given resolution and display size the optimal viewing distance can be derived. In the following, first this optimal viewing distance is investigated, highlighting the basic motivation behind the introduction of the HD and UHD formats.

The optimal viewing distance

Generally speaking the goal of pixel based image reproduction is to ensure that the light rays, arriving from adjacent pixels include an angle below the spatial resolution of the human vision. Once it is achieved it is inherently ensured that the pixel structure of the image is not visible, and allows RGB based color reproduction, since instead of perceiving the individual RGB primaries, only the mixture of the primaries is perceived. In the foregoing it has been discussed that the visual acuity of human vision is $\frac{1}{60}^\circ$ (at least for luminance, for colors the resolution is significantly lower), from which for a given pixel size the minimal viewing distance can be derived. In practice instead of the pixel size, the dimensions and the horizontal and vertical resolution of displays are given, hence it is useful to express the minimal viewing distance as the function of these quantities.

In the following, the geometry depicted in Figure 2.24 is investigated in case of a display with given height H and number of active lines N_V, with the display located at a distance D from the viewer. The vertical pixel size is then $\frac{H}{N_V}$.

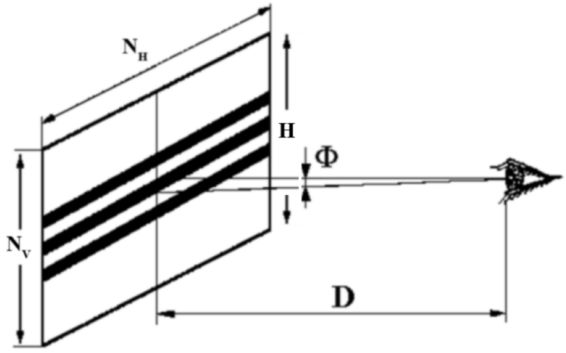


Figure 2.24: Geometry for deriving the optimal viewing distance

The perceived angle between the adjacent pixels ate the observer position is then given as

$$\tan\frac{\Phi}{2} = \frac{H}{2N_V D}. \tag{2.24}$$

For the sake of simplicity the small argument approximation of the tangent function can be applied, i.e. $\tan x \approx x$, ha $x \ll 1$, leading to

$$\Phi = \frac{H}{N_V D} \quad \rightarrow \quad D = \frac{H}{N_V \Phi}. \tag{2.25}$$

By substituting the visual activity of the HVS ($\frac{1}{60} \cdot \frac{\pi}{180}$ rad $= 2.9 \cdot 10^{-4}$) for a display with given vertical resolution the optimal (minimal) viewing distance is given by

$$D = H\frac{1}{N_V\, 2.9 \cdot 10^{-4}}. \tag{2.26}$$

This is the so-called **Lechner distance**, formulating the optimal viewing distance, taken into consideration during the deigning of a display with given size and resolution.

In case that also the display aspect ratio (denoted by a_r) is given (for SD 4:3 and for HD 16:9) the Lechner distance can be expressed as the function of the horizontal dimension (for the sake of simplicity assuming square pixels):

$$D = \frac{W}{a_r}\frac{1}{N_V\, 2.9 \cdot 10^{-4}}, \tag{2.27}$$

where W is the horizontal dimension (width) of the display. Furthermore, if the display is observed from the optimal viewing distance, the horizontal field of view angle can be expressed, included by the display:

$$\tan\frac{\Phi_H}{2} = \frac{W}{2D} \qquad \rightarrow \qquad D = \frac{W}{2\tan\frac{\Phi_H}{2}}, \tag{2.28}$$

Table 2.3: Optimal viewing distance of common SD and HD formats and the covered observer field of view

	SD, 480i	SD, 576i	HDTV
Active lines:	480	576	1080
Viewing distance:	7 x height	6 x height	3 x height
Viewing distance:	4.25 x diameter	3.6 x diameter	1.5 x diameter
Horizontal field of view	$\approx 11^\circ$	$\approx 13^\circ$	$\approx 32^\circ$

and the field of view is written as

$$\Phi_H = 2 \arctan\left(\frac{a_r \, N_V \, 2.9 \cdot 10^{-4}}{2}\right). \tag{2.29}$$

The evaluation of these results for the discussed SD formats and the upcoming HD formats is summarized in Table 2.4. As a result it is obtained that displays with SD resolution should be observed from a distance 6-7 times the height of the screen. In this case, it is verified that the display covers about 10-13 degrees from the observer's field of view horizontally.

The table already foreshadows the main motivation behind the introduction of HD video format: increasing the perceived reality of the reproduced scene, by filling an increased field of view with video content, compared to SDTV. Therefore, the basic goal was not to squeeze six times the number of pixels into the same visual angle opposed to the popular misbelief. Instead the angular subtense of a single pixel should be maintained, and the entire image shoud occupy a much larger area of the viewer's visual field.

Short HD history

Although today the term high definition format is associated with digital video, even in the era of early analog TV system increased resolution, high definition initiations existed. Years before the introduction of color television, in 1949 France started analog monochromatic transmission with a high resolution standard at 819 lines (with 737 active lines), a system that should have been high definition even by today's standards, discontinued only in 1983. In 1958, the Soviet Union developed the Transfomator, the first high-resolution television system capable of producing an image composed of 1125 lines of resolution aimed at providing teleconferencing for military command. It was a research project and the system was never deployed by either the military or consumer broadcasting.

In 1979, the Japanese public broadcaster NHK (Japan Broadcasting Corporation) developed consumer high definition television with 1125 scan lines and a 5:3 display aspect ratio. The system, known as Hi-Vision or MUSE required about twice the bandwidth of the existing NTSC system but provided about four times the resolution, applied for analog regular HD broadcasting beginning in 1994. The limited standardization of analog HDTV in the 1990 did not lead to global HDTV adoption as technical and economic constraints did not permit HDTV to use bandwidths greater than normal television.

The introduction of the HD format was finally motivated by the emerging of digital compression methods (mainly of MPEG-1 and MPEG-2). The HD format was codified in the ITU-709 (Rec. 709) standard, published in 1990.

HD parameters

The ITU-709 recommendation prescribed the following properties for HD video format:

- **Aspect ratio:** The first standardized parameter for the new video format was its aspect ration, being chosen to 16:9. The choice is not obvious, since at the time of the standardization process no video content was captured with this aspect ratio: TV movies were shot with the aspect ratio of SD format (4:3), while movie films used widescreen formats (most commonly the 1.85:1 and 2.2:1 widescreen, or 2.35:1 anamorphic format). It has been found that rectangles with the side ratios of the above popular aspect ratios and with equal areas exactly fit within an outer rectangle with the aspect ratio of 16:9. Furthermore, the intersection of all these rectangles is an inner rectangle with the aspect ratio of 16:9. Therefore, the aspect ratio of 16:9 ensured the compatibility with most of the then-existing formats. The geometry is shown in Figure 2.25.

- **Raster scan, field and frame rate:** In the HD system besides interlaced scanning the possibility of progressive scan was introduced. As the legacy of the SD system, numerous field rate and field rates are specified by the format, being 24 Hz, 25 Hz, 30 Hz, 50 Hz and 60 Hz field/frame rates, along with fractional rates, having the above values multiplied by $\frac{1000}{1001}$.

- **Sampling frequency:** The HD sampling frequency can be derived from the SD sampling rate: One of the main goals of the HD format was to double both the horizontal ($\times 2$) and vertical ($\times 2$) spatial resolution, along with the aspect ratio changed to 16:9 ($4:3 \times 4/3$), which the total pixel number being at least $2 \times 2 \times \frac{4}{3} = 5.33$ times that of the SD format. In order to ensure orthogonal sampling (only full pixels in a line) for most field and frame rates the optimal choice for the sampling frequency was $5.5 \cdot 13.5 = 74.25$ MHz and the fractional sampling frequency $\frac{1000}{1001} \cdot 74.25$ MHz for fractional field/frame rates. In case of progressive scanning for frame rates $50 - 60$ Hz the data rate is double compared to the interlaced case, therefore, a doubled sampling frequency of $fs = 148.5$ MHz is required.

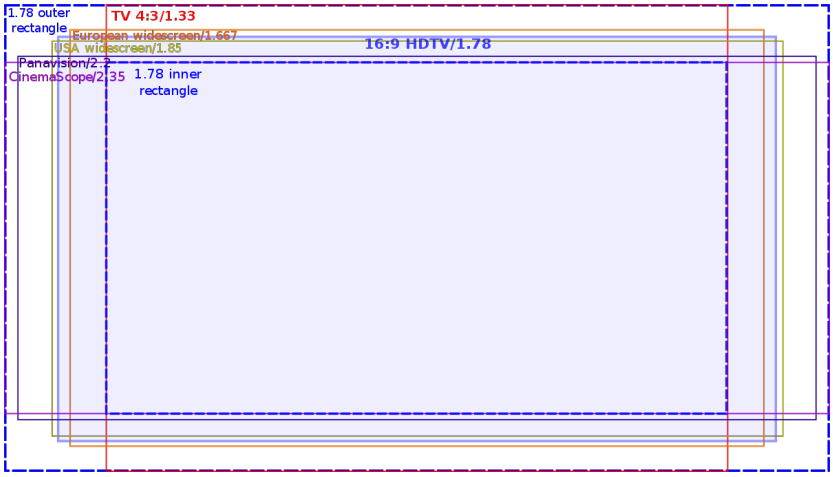


Figure 2.25: The 16:9 aspect ratio as the outer and inner rectangle of rectangles with common aspect ratios

- **Spatial resolution:** After a lengthy debate [16] the number of active line has been codified to 1080 and the total number of lines to 1125, including the vertical blanking with both progressive and interlaced scan modes (this raster format was also codified in SMPTE 274M). The horizontal number of active pixels has been chosen to a fixed number of 1920 regardless of the frame/field rate, resulting in the active resolution of 1920×1080. Therefore, the HD formats with different field and frame rates only differ in the number of total horizontal number of pixels (i.e. in the length of the horizontal blanking time). The different HD formats are denoted using the following convention:

  - Active resolution in pixels, denoting often only the vertical resolution
  - Scanning mode: p for progressive and i for interlaced
  - frame rate. For interlaced video often instead of frame frate—incorrectly—field rate is marked.

  As an example: $1080i25$ denotes interlaced video with 1080 active lines, and the frame rate of 25 Hz (i.e. field rate is 50 Hz).

  Due to the large data rate of progressive HD formats the ITU-709 has been extended with a lower spatial resolution format, containing 720 active lines and 1280 active pixels in a line, with exclusively progressive scanning mode. [17]. It is denoted by: **720p**. Two examples for the two basic HD formats can be seen in 2.26.

[16] The first version of the standard specified 1035 active lines, which was later withdrawn.

[17] As already discussed, the basic goal of 1080 lined format was to double the number of scan lines. The line number of the 720p format—as an intermediate format between SD and HD—applies $\frac{3}{2} \cdot 480 = 720$ scan lines, with the horizontal number of pixels obtained from the aspect ratio of 16:9

Table 2.4: Sampling frequency, total number of lines and columns and active resolution of commonly used HD formats. The total number of pixels in a line can be calculated according to $N_{\mathrm{H}} = f_s \times \frac{f_{\mathrm{V}}}{N_{\mathrm{V}}}$, with f_{V} being the frame rate.

Format	f_s [MHz]	N_{H}	N_{V}	Active resolution
$720p50$	74.25 MHz	1980	750	1280×720
$720p59.94$	$74.25 \cdot \frac{1000}{1001}$ MHz	1650	750	1280×720
$1080i25$	74.25 MHz	2640	1125	1920×1080
$1080i30$	74.25 MHz	2200	1125	1920×1080
$1080p50$	148.5 MHz	2640	1125	1920×1080
$1080p59.94$	$148.5 \cdot \frac{1000}{1001}$ MHz	2200	1125	1920×1080

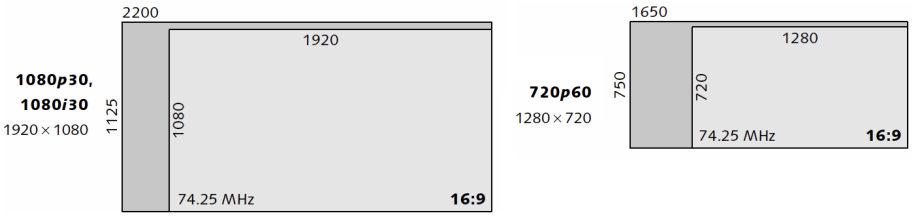


Figure 2.26: Az 1080 soros és 720 soros HD formátum szemléltetése

- The primaries of the HD format coincide with that of the ITU-601 SD format, resulting in the same gamut. The luma (and the relative luminance) is calculated as

$$Y' = 0.2126R' + 0.7152G' + 0.0722B'. \tag{2.30}$$

  Opposed to the SD formats, in this case the luma coefficients are mathematically correct, obtained from the relative luminance coefficients of the chosen primaries and its white point, i.e. the coefficients correspond to the second row of the $RGB \rightarrow XYZ$ transformation matrix.

- The opto-electronic transfer function coincides with the SD's OETF:

$$E = \begin{cases} 4.500L, & \text{ha } L < 0.018 \\ 1.099L^{0.45} - 0.099, & \text{ha } L \geq 0.018, \end{cases} \tag{2.31}$$

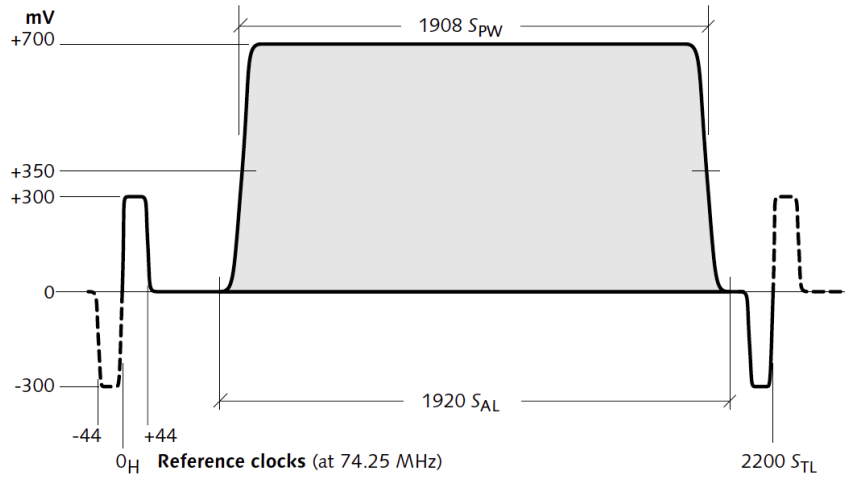


Figure 2.27: Structure of one line of the 1080 lined HD format.

where $L \in \{R, G, B\}$.

- The bit depth is 8 bits for consumer and 10 bits for studio applications.
- The studio standard codifies the chroma subsampling scheme of 4:2:2.

The structure of the resulting standardized HD video signal is the same as the SD video signal, discussed in the foregoing, illustrated in Figure 2.27. The sole difference is the application of tri-state sync pulses, with the physical dynamic range usually being $0, \pm 300$ mV.

Uncompressed data rate of HD formats

In the following the uncompressed data rate of different video formats is discussed. With denoting the active and total pixel dimensions the notations showed in Figure 2.28 the total data rate can be calculated as

$$BR_T = \underbrace{N_{\mathrm{H}} \cdot N_{\mathrm{V}} \cdot f_{\mathrm{V}} \cdot n_{\mathrm{bit}}}_{\text{bitrate per sample}} \cdot n_{\mathrm{CS}}, \tag{2.32}$$

with denoting f_V the frame rate, n_{bit} the bit depth and n_{CS} the number of components, depending on the chroma subsampling scheme applied. The value of the latter is $n_{\mathrm{CS}} = 3$ for 4:4:4, $n_{\mathrm{CS}} = 2$ for 4:2:2 and $n_{\mathrm{CS}} = 1.5$ for 4:2:0 schemes.

Similarly the active data rate is calculated as

$$BR_A = N_{\mathrm{H,A}} \cdot N_{\mathrm{V,A}} \cdot f_f \cdot n_{\mathrm{bit}} \cdot n_{\mathrm{CS}} \tag{2.33}$$

.

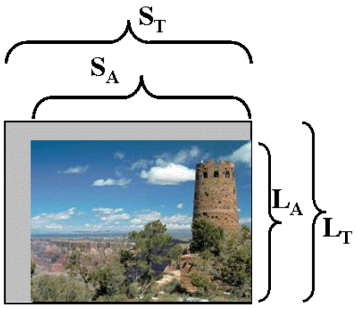


Figure 2.28: Notation-convention for calculating the data rate of video

- N_{H}: Total pixel number/line
- $N_{\mathrm{H,A}}$: Active pixel number/line
- N_{V}: Total number of lines/frame
- $N_{\mathrm{V,A}}$: Number of active lines/frame

The active and total bit rates of several frequently used video formats are summarized in Table **??**[18]. The uncompressed bit rate approximately equals for $720p$ and

Table 2.5: The total and active bitrate of frequently used video formats

Format	**Sampling frequency**	**Total bitrate 4:2:2**	**Active bitrate 4:2:2**	**Active bitrate 4:4:4**
$576p50$	13.5 MHz	0.54 Gbit/s	0.41 Gbit/s	0.62 Gbit/s
$720p60$	74.25 MHz	1.49 Gbit/s	1.11 Gbit/s	1.67 Gbit/s
$1080i30$	74.25 MHz	1.49 Gbit/s	1.24 Gbit/s	1.86 Gbit/s
$1080p60$	148.5 MHz	2.97 Gbit/s	2.49 Gbit/s	3.73 Gbit/s
$2160p60$	297 MHz	11.88 Gbit/s	9.96 Gbit/s	14.93 Gbit/s

$1080i$ formats. An advantage of the progressive format is it can be more efficiently compressed than interlaced video. On the other hand interlaced format ensures high vertical resolution for still and slowly moving video content, although with the resolution degrading for moving reproduced objects. Therefore, broadcasting operators with sport content in their main profile traditionally chose $720p50/60$ format, while operators, broadcasting mainly news and movies usually apply $1080i$ video format.

[18]The vertical blanking interval of UHD format is fixed to 90 lines, with the horizontal blanking depending on the frame rate of the video. For a video with the frame rate of 60 Hz the horizontal blanking interval is 560 samples.

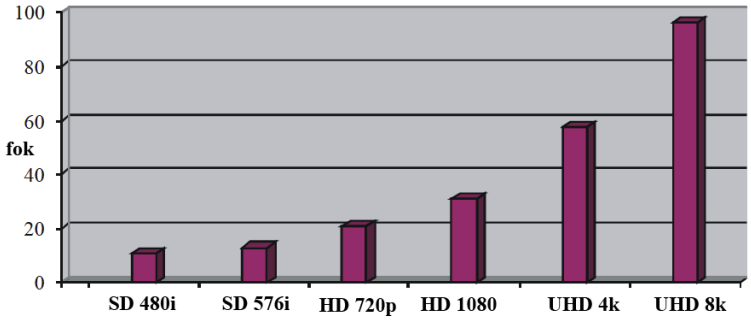


Figure 2.29: Visual angle ensured by SD, HD and UHD formats, with the screen being watched from the optimal viewing distance.

2.3.3 The UHD format

The original intention behind the introduction of HD format was to enhance the visual experience by increasing the listener's visual angle filled with video content. The ultra high definition format aims at the further improvement of the reproduction quality by applying even larger display sizes—covering an even larger part of the field of view—and by increasing the displayed image quality.

Similarly to the HD format, the first notable initiation of UHD technology is connected to the Japanese NHK, capturing video data at the resolution of 7680×4096 by using an array of 16 HDTV recorders and four CCD sensors with the resolution 3840×2048, as early as 2003. The first UHD standard was published in 2007 (SMPTE 2036), while the currently accepted UHD codification was introduced in 2012, entitled the **ITU-R BT. 2020**. The standard specifies two UHD formats, the 4k with its spatial resolution being the double of the 1080p format both horizontally and vertically (meaning 4 times larger total pixel size), and the 8k, with a further doubled resolution, compared to 4k.

Theoretically, the goal of the 4k and 8k formats was to fill the visual angle of approximately 58° and 96° with content, respectively. The visual angle of the different SD, HD and UHD formats are compared in Figure 2.29.

In order to ensure high quality video reproduction over a large visual angle the ITU-2020 improved the reproduction parameters in numerous aspect compared to HD:

- **Spatial resolution:** The standard specifies two resolution types: 3840×2160 termed as 4k and 7680×4320 as 8k resolutions. Both 4k and 8k employs squared pixels, i.e. both the display and storage aspect ratios are 16:9.

- **Scanning type:** Unlike HD, ITU-2020 exclusively allows progressive scanning mode (i.e. the p/i designation is no longer used).

- **Frame rate:** With the increased visual angle the UHD content already fills the peripheral vision of the observer with content. Since the peripheral vision is dominated by rods with a much quicker response time

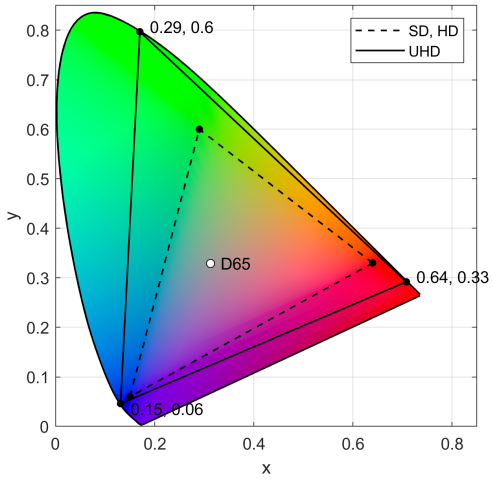


Figure 2.30: Gamut of the ITU-2020 UHD color space compared with the gamut of SD and HD color representations.

than the cones of the central vision, thus, in order to ensure continuous motion and avoid flickering significantly higher frame rates are supported than the HD frame rates. The standard allows the frame rates of $120, 119.88, 100, 60, 59.94, 50, 30, 29.97, 25, 24, 23.976$ Hz.

- **Color space:** For the first time since the introduction of NTSC the ITU-2020 UHD standard applies new RGB primaries for color representation for the sake of an enlarged gamut. The resulting gamut is shown in Figure 2.30: As the figure verifies it, the UHD color space applies spectral colors for the RGB primaries, described by the wavelengths of $\lambda_R = 630$ nm, $\lambda_G = 532$ nm and $\lambda_B = 467$ nm. [19]. The ITU-2020 covers the $75.8\% of$ the CIE chromaticity diagram and the entire Pointer's gamut of real surface colors [20]. The relative luminance (and the luma) can be calculated from the RGB coordinates as

$$Y = 0.2627\,R + 0.6780\,G + 0.0593\,B. \tag{2.34}$$

- **Bit depth:** The increase of the range of the described colors requires increasing

[19] Obviously, this does not mean that UHD displays use spectral colors as RGB primaries, instead the color pixels of UHD video content are stored and transmitted in terms of spectral primary colors. Monitors and TVs display colors by using the actual applied LCD or LED primary colors, first converting the ITU-2020 content into the RGB color space of the display. The gamut of these color spaces are obviously smaller than the gamut of the UHD standard: as an example, JDI introduced a studio reference monitor with the diameter of $14.3''$ in 2018, allowing the reproduction of 97% of the ITU-2020 color space.

[20] The Pointer's gamut is the result of the series of measurements, containing the chromaticites (hues) of colors of reflective surfaces, occurring in the natural environment—opposed to the colors that can be produced by emissive surfaces, e.g. neon or colors of LED/LCD light sources. The Pointer's gamut is based on 4089 measurement samples, with the database published in 1980. Since then, the Pointer's gamut is a de facto standard of qualifying color spaces.

the bit depth of representation as well, in order to ensure the same quantization precision. Since a larger color space increases the difference between colors an increase of 1-bit per sample is needed for Rec. 2020 to equal or exceed the color precision of ITU-709. Thus, Rec. 2020 defines a bit depth of either 10 bits per sample for consumer or 12 bits per sample for studio applications.

- **The opto-electronic transfer function:** The OETF of the ITU-2020 coincides with that of the SD and HD standards:

$$E = \begin{cases} 4.500L, & \text{ha } L < \beta \\ \alpha L^{0.45} - (\alpha - 1), & \text{ha } L \geq \beta, \end{cases} \tag{2.35}$$

where $\alpha = 1.09929682680944$ and $\beta = 0.018053968510807$. The only difference is the precision of the coefficients: for 12 bits bit depth the above coefficients should be evaluated with 5 digits precision.

- **Chroma subsampling:** The standard specifies the subsampling schemes of 4:2:0, 4:2:2 and 4:4:4. In the latter case instead of luma-chroma representation direct $R'G'B'$ representation is allows. In case of 4:2:0 or 4:2:2 sampling, besides $Y'C_{\mathrm{B}}C_{\mathrm{R}}$representation the so-called **contant luminance mode** is supported, resulting in the $YC_{\mathrm{bc}}C_{\mathrm{rc}}$ components. This constant luminance mode may be used when the top priority is the most accurate retention of luminance information. The luma component in $YC_{\mathrm{bc}}C_{\mathrm{rc}}$ is calculated using the same coefficient values as for $Y'C_{\mathrm{B}}C_{\mathrm{R}}$, but it is calculated from linear RGB and then gamma corrected, rather than being calculated from gamma-corrected $R'G'B'$. As a result Y' in constant luminance mode is mathematically accurately describes the gamma-corrected relative luminance component (therefore, the resulting luma contains no color information and the resulting chroma contains no luminance information). The prevents artifacts that arise in case of $Y'C_{\mathrm{B}}C_{\mathrm{R}}$representation due to the subsampling of minor luminance information, present in the chroma components.

2.3.4 Digital interfaces

The most common physical interface for SD, HD and UHD video transmission in video studiotechnologies is the SDI (Serial Digital Interface), and HDMI (High-Definition Multimedia Interface) in consumer applications.

The bit rates of most common video formats were summarized in Table 2.5. As a comparison, the maximal digital bandwidth of the different HDMI interface versions are the following

- HDMI 1.0-1.2: 4.95 Gbit/s (3.96 Gbit/s effective)[21]
- HDMI 2.0: 18 Gbit/s (14.4 Gbit/s effective)

[21] Due to undiscussed reasons the HDMI interface applies a so-called 8b/10b channel coding, transmitting 8 bits of data in 10 bit sequences. Therefore, only the part of $\frac{8}{10}$ of the total bandwith can used for effective data transmission.

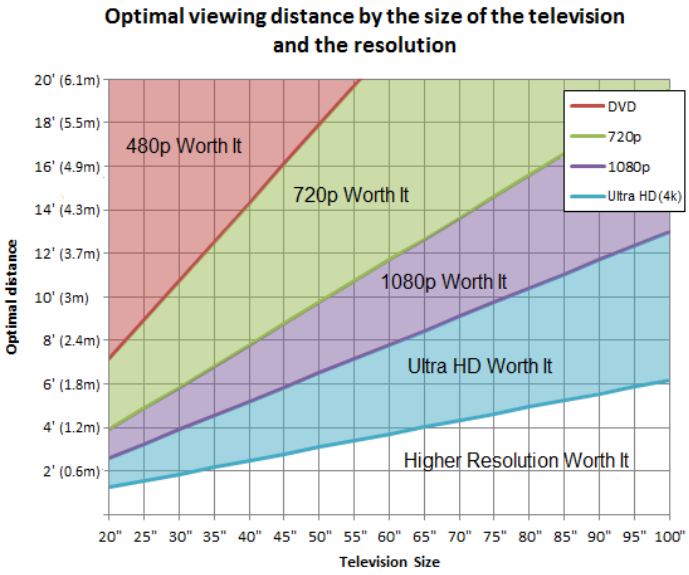


Figure 2.31: Optimal viewing distance as the function of display diameter.

- HDMI 2.1: 48 Gbit/s (38.4 Gbit/s effective)

Obviously, in order to find the HDMI version for the transmission of a given video format, the total number of lines and sample per lines have to be taken into consideration, since HDMI signals contains the horizontal and vertical blanking intervals as well. As discussed earlier: the vertical blanking interval carries multichannel audio streams and other auxiliary data, presented simultaneously with the video data.

It can be concluded that HDMI 1.0 version was created mainly for the transmission of 1080p video format with 4:2:2 chroma subsampling scheme, and the bit depth of 10 or 12 bits (4:4:4 video can be only transmitted with this version represented on 8 bits). On the other hand, HDMI 2.0 was developed for 4k, and the 2.1 version for 8k video transmission.

2.3.5 Optimal choice of display resolution

As the conclusion of the present chapter the Lechner distance is revisited in order to arrive at the the optimal viewing distance for a given display size and for the optimal display size for a given observer distance. Besides the Lechner distance, several other recommendations exist for the optimum viewing distance, including manufacturers, retail and THX recommendations.

- SMPTE 30: is the standardized codification of the Lechner distance, recommending the viewing distance of 1.6 times the diameter in case of a HD display with 1080 lines, resulting in a field of view of 30°. This recommendation is

very popular with the home theater enthusiast community, appearing in books on home theater design,

- The recommendation of manufacturers, retails and several publications suggests the viewing distance of 2.5 times the diameter in case of a HD display, resulting in the field of view of 20°.
- THX recommends that the „best seat-to-screen distance" is one where the view angle approximates 40°, which according to the THX approximates the cinema experience the most. This is achieved in case of HD format with the viewing distance being 1.2 times the display diameter.

Although there are slight discrepancies between these recommendations, they all agree in that for the viewing distance of the HD displays „the closer the better".

The graph of optimal viewing distances can be visualized for the different video formats, as depicted in Figure 2.31. At a given display size, of course the viewing distance can be increased, the pixel structure of the display does not become visible. Thus, at a fixed display size above a given line in the graph the display is applicable. The graph, therefore, can be divided into regions, indicating the optimal display resolution for a given viewing distance and display size.

Based on statistics, conducted by Bernard J. Lechner the average domestic TV viewing distance is approximately $2.7\ \mathrm{m}$. At this distance displays above the diameter of 50" should have the resolution of 1080p, while in order to exploit the advantages and quality of 4k resolution the display should have a diameter of at least 75" ($\sim 1.9\ \mathrm{m}$). Displays with the diameter of 2 meters are extremely rarely applied in domestic use even today, suggesting that the capabilities of even 4k displays are not completely utilized nowadays. However, the introduction of consumer 8k displays is already trending, with also experimental 8k broadcasting initiations beginning in the recent years. As an example, the first dedicated satellite for broadcasting 8k content has been lunched (BSAT-4a), which was planned to broadcast the 2020 summer olympics, which has been however postponed to 2021 due to the outbreak of the COVID-19 pandemic.

End-of-Chapter Questions

- What were the reasons behind the introduction of the interlaced format? What is the main idea of interlaced scanning?
- How was the sampling frequency of the SD format chosen? How was it extended in order to choose the sampling frequency of the HD format?
- Calculate the optimal viewing distance for a 4k (2160p) display with the diameter of 65" (the aspect ratio is 16:9)!

- List some improvements of the UHD standard compared to the HD format!
- Define the number of total and active resolution of the $2160p60$ format! The number of inactive lines is 90 and the sampling frequency is $297\ \mathrm{MHz}$.
- Define the total bitrate of the video format of the previous example in case of a chroma subsampling scheme of 4:2:2, with 12 bits per sample representation! Pick the minimal HDMI version which is capable of transmitting the exemplary video stream, if the HDMI transmits 8 bit data in 10 bits sequences (i.e. the effective bandwidth is $\frac{8}{10}$ times the total bandwidth), and the total bandwidth of the HDMI interface versions are
  - HDMI 1.0-1.2: $4.95\ \mathrm{Gbit/s}$
  - HDMI 1.3-1.4: $10.2\ \mathrm{Gbit/s}$
  - HDMI 2.0-1.2: $18\ \mathrm{Gbit/s}$
  - HDMI 2.1: $48\ \mathrm{Gbit/s}$

Chapter 3

Basics of image and video compression

The previous chapter introduced the basic properties of consumer and professional, studio video parameters.

The active spatial resolution and the resulting bit rates of frequently used digital video formats are summarized in Table 3.1.

Table 3.1: The active bitrate of frequently used video formats along with the size, required for storing 1 hour of video stream

Format	**Active resolution**	**Active bitrate 4:2:2**	**Active bitrate 4:2:0**	**Size of 1 hour video**
SIF ($i59.54$)	352×240	40.6 Mbit/s	30.4 Mbit/s	13.7 Gbyte
CIF ($i59.54$)	352×288	48.6 Mbit/s	36.5 Mbit/s	16.4 Gbyte
$576i50$	576×720	199 Mbit/s	149.1 Mbit/s	67.1 Gbyte
$720p60$	1280×720	883 Mbit/s	662.8 Mbit/s	298.3 Gbyte
$1080i30$	1920×1080	994 Mbit/s	745.8 Mbit/s	335.6 Gbyte
$1080p60$	1920×1080	1.99 Gbit/s	1.49 Gbit/s	671.2 Gbyte
$2160p60$ (10 bits)	3840×2160	9.95 Gbit/s	7.47 Gbit/s	3.36 Tbyte
$4320p60$ (10 bits)	7680×4320	39.8 Gbit/s	29.9 Gbit/s	13.44 Tbyte

In the table SIF and CIF abbreviate Source Input Format and Common Intermediate Format respectively. Both formats were introduced for the consumer digital representation of NTSC and PAL videos—with CIF being the default video format of the H.261 encoder and SIF being that for the MPEG-1 standard— with a halved vertical resolution when compared to the professional ITU-601 studio standard.

As the table verifies it, the generated data rate of video formats—and thus the required storage space—grows exponentially with higher spatial and temporal resolution. Modern studio and consumer interfaces—variants of the SDI interface for studio applications and HDMI or DisplayPort for consumer use—allow the transmission of the data rates of uncompressed video over short ranges, e.g. between local devices. However, the storage and broadcasting of such high data rates is virtually impossible: the compression of digital video data is indispensable.

Fortunately, real-life sequence of images contain significant amount of redundant information: Statistically speaking within single frames the neighboring pixels are usually highly correlated. Similarly, consequent frames are usually very similar to each other, even if they contain objects under motion. In video signals, the redundancy can be classified as spatial, temporal, coding and psychovisual redundancies:

- Spatial redundancy (or intraframe/interpixel redundancy) is present in areas of images or video frames where pixel values vary only by small amounts.

- Temporal redundancy (or interframe redundancy) is present in video signals when there is significant similarity between successive video frames.

- Coding Redundancy is present if the symbols produced by the video encoder are inefficiently mapped to a binary bitstream. Typically, entropy coding techniques can be used in order to exploit the statistics of the output video data where some symbols occur with greater probability than others.

- Psychovisual redundancy is present either in a video signal or a still image containing perceptually unimportant information: The eye and the brain do not respond to all visual information with same sensitivity, some information is neglected during the processing by the brain. Elimination of this information does not affect the interpretation of the image by the brain and may lead to a significant compression. Psychovisual redundancy is usually removed by appropriate requantization of the video data, so that the quantization noise remains under the threshold of visibility.

In order to achieve a high compression ratio, all the above redundancy types should be eliminated, being the basic goal of a **source encoder**.

Generally speaking, the aim of source encoding is reducing the source redundancy by keeping only the relevant information, based on the properties of the source and the sink. The source in this case is the video (or possibly audio) sequence, and the sink is the human visual system (or the auditory system for audio info). The general structure of a source encoder, valid both for video or audio inputs is depicted in Figure 3.1. The reduction of the different types of redundancy is performed by the following steps:

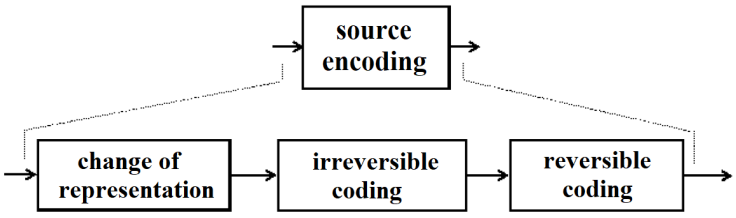


Figure 3.1: Block scheme of a general video/audio source encoder.

- Change of representation: in order to reduce spatial and temporal the input data is represented in a new data space containing less redundancy. The change of representation can be performed by
  - Differential coding (DPCM: Differential Pulse Code Modulation)
  - Transformation coding
  - Sub-band coding
- Irreversible coding: the accuracy of representation is reduced by removing irrelevant information, hence, eliminating psychovisual redundancy. Irreversible coding is achieved by
  - requantization of the data
  - spatial and temporal subsampling
- Reversible coding: an efficient code-assignment is established reducing statistical redundancy. Types of reversible entropy coding applied often in video, image and audio processing are
  - Variable Length Coding (VLC)
  - Run-Length Coding (RLC)

In the following this chapter introduces the basic concepts of compression methods, based on differential coding and transformation coding. The basic concepts are introduced for the generalized case of arbitrary one and two dimensional input signals, and later specialized to video signal inputs.

3.1 Predictive coding

Predictive coding, or differential quantization is a compression technique, utilizing linear prediction along with the requantization of the predicted data (i.e. performing both a change of representation and irreversible coding): instead of the direct quantization and transmission of the input signal, the actual input sample is predicted with an appropriately chosen prediction algorithm, and only the discrepancy between the actual and the estimated sample is further processed. In the receiver the same prediction is

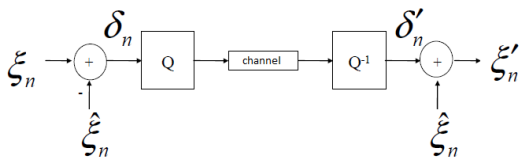


Figure 3.2: Block scheme of a general differential encoder and decoder.

performed as in the source side, and the output sample is obtained as the sum of the estimated signal and the error of estimation.

The signal processing steps in a differential encoder and decoder are shown in Figure 3.2 with the following notation:

- $\xi(n)$ is the input source sample
- $\hat{\xi}(n)$ is the predicted input sample
- $\delta(n)$ is the error of prediction/differential signal
- Q is the quantization of the signal
- Q^{-1} is the inverse quantization
- $\delta'(n)$ is the quantized differential signal
- $\xi'(n)$ is the quantized, reconstructed input sample

In the block diagram quantization is performed by rescaling the input signal to match the dynamic range of the quantizer, followed by the rounding of the signal level to the nearest integer. Inverse quantizer, on the other hand scales back the quantized signal to the original dynamic range (obviously, information loss can not be reversed).

The basic idea behind differential quantization is the following: Assuming an efficient prediction the dynamic range of the differential signal is significantly smaller than that of the original input signal. Therefore, discretizing the error signal means the division of a smaller dynamic range to the same number of intervals (2^N in case of N bits representation) than in case of quantizing the input signal directly, resulting in an increased resolution, or mathematically speaking, in an increased signal-to-noise ratio. Alternatively, the same signal-to-noise ratio may be achieved by using lower bit depths utilizing differential quantization.

In order to give a mathematical description on differential quantization and quantify the introduced quantities, first a brief summary of stochastic processes is given.

3.1.1 Basic stochastic concepts

A stochastic process is any process describing the evolution in time or space of a random phenomenon, given by an indexed sequence of random samples. Each sample is a random variable with a given probability distribution, and with the probability usually depending on the previous samples. For the sake of simplicity it is implied here that the

process evolves over time, but all the following can be easily extended for e.g. spatially dependent processes.

Let ξ denote a stochastic process, and the sample index denoted by n, hence for each index $\xi(n)$ is a random variable. A stochastic process is fully described by its joint distribution function, which is, however, rarely available either by measurement or analytically. Instead, more often stochastic processes are characterized in a simplified manner by their **moments** (being the **mean value** its first and the **variance** its second moment) and the **autocorrelation function**.

Wide-sense stationary processes: In the following only **stationary processes** are investigated, that's statistical properties do not change over time. Strict stationary requires the entire joint distribution function of the process to be time invariant. In most applications it is sufficient to require the process to be **wide-sense stationary (WSS)**, defined by the following properties:

- The mean/expected value of a WSS process is constant, invariant of n:

$$m_\xi(n) = m_\xi \tag{3.1}$$

  Once the above relation holds, the expected values of the process can be approximated as the average of a realization of length N according to

$$m_\xi = \mathbb{E}(\xi(n)) = \frac{1}{N}\sum_{n=1}^{N}\xi(n) \tag{3.2}$$

- For a general process the autocorrelation function can be defined for two distinct samples, i.e. it is a two-dimensional function

$$r_\xi(n_1, n_2) = \mathbb{E}(\xi(n_1)\cdot\xi(n_2)), \tag{3.3}$$

  loosely speaking measuring the linear dependence between samples $\xi(n_1)$ and $\xi(n_2)$. If two samples are uncorrelated—i.e. $r_\xi(n_1, n_2) = 0$—it implies that no linear relation exists between them, however, higher order dependence may be present. Therefore, uncorrelatedness does not imply independence (while independence strictly ensures uncorrelatednes).

  For a WSS process this linear dependence is translation invariant

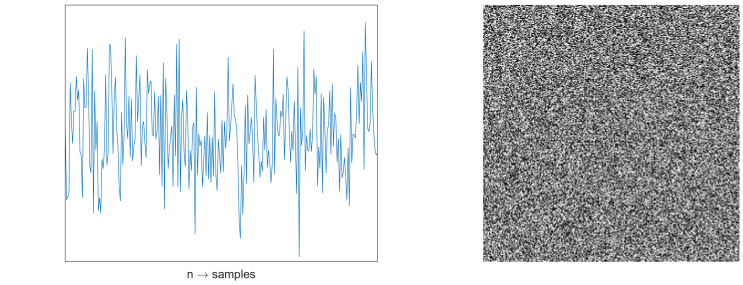
$$r_\xi(n_1, n_2) = r_\xi(n_1 + d, n_2 + d), \qquad \forall d \in \mathcal{N} \tag{3.4}$$

  therefore autocorrelation depends only on the distance of the two samples (denoted now by d)

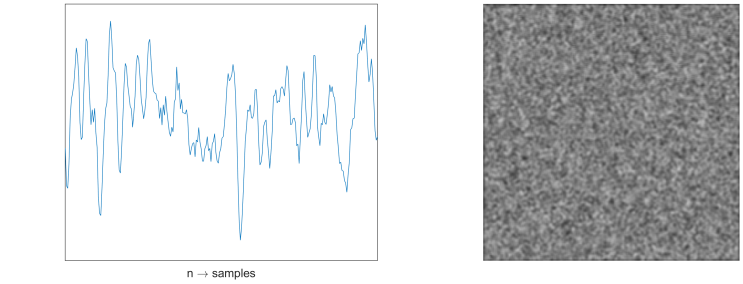
$$r_\xi(n_1 - n_2) = r_\xi(d). \tag{3.5}$$

  If the above relation holds, autocorrelation can be statistically approximated from a realization of the process as

$$r_\xi(d) = \mathbb{E}(\xi(n)\cdot\tilde{\xi}(n+d)) = \frac{1}{N}\sum_{n=0}^{N}\xi(n)\xi(n+d) \tag{3.6}$$



(a)



(b)

Figure 3.3: One dimensional and two dimensional white noise process (a) and correlated noise process (b).

- As a further property for WSS process the auto-correlation function at zero lag ($d = 0$) gives the mean value of the squared samples, i.e. the mean energy of the process, being obviously also time invariant

$$r_\xi(0) = E_\xi = \mathbb{E}(\xi(n)^2) = \frac{1}{N}\sum_{n=0}^{N}\xi(n)^2. \tag{3.7}$$

Noise processes: As the most simple stochastic example, an uncorrelated random process is considered, meaning that linear relation exists between neighboring samples. For such a process the autocorrelation is zero valued everywhere, except for zero lag ($d = 0$), where the autocorrelation value is the energy of the random process. The autocorrelation, therefore, is a Kronecker delta (discrete Dirac delta) function at the origin, given by

$$r_\xi(n) = E_\xi \cdot \delta(n) = \begin{cases} 0, & \text{if } n = 0 \\ E_\xi, & \text{elsewhere.} \end{cases} \tag{3.8}$$

Such a stochastic process is called **white noise**. The distribution of the individual samples is arbitrary, most often the samples are drawn from uniform or Gaussian normal distribution.

The terminology originates from the **power spectral density**, defined as the Fourier transform of the autocorrelation function , describing the frequency content of the stochastic process. For white noise the spectral density function is constant, similarly to the spectrum of white light containing all lights with all the visible wavelengths equally. A simple example realization of white noise process is depicted in Figure 3.3 (a) in one and two dimensions.

A correlated process can be most easily generated from white noise by linear filtering (e.g. FIR filtering): since after filtering each output sample is produced as the linear combination of the previous samples, therefore neighboring samples become correlated, and the autocorrelation is described by the filtering coefficients themselves. Correlated noise, obtained by filtering of the exemplary white noise realization is depicted in Figure 3.3.

3.1.2 The goal of differential quantization

Having introduced basic stochastic concepts differential quantization can be discussed mathematically.

In the model applied the input signal $\xi(n)$ is assumed to be a wide sense stationary process. The effect of quantization can be most easily modeled as an additive noise $\epsilon(n)$, added to the quantized signal. Efficiency of quantization is usually described by the signal-to-quantization-noise ratio, defined as the ratio of the energy of the quantized signal and the quantization noise, written as

$$\mathrm{SQNR} = \frac{\mathbb{E}(\xi(n)^2)}{\mathbb{E}(\epsilon(n)^2)}, \tag{3.9}$$

assuming that the quantized signal is the input signal directly.

In an ideal case where the quantization error is uniformly distributed and the signal has a uniform distribution covering all quantization levels the quantization noise can be calculated as

$$\mathrm{SQNR} = 20\log_{10}2^N, \tag{3.10}$$

where N is the bit depth. In case that differential quantization is applied, two statements can be made

- Assuming that in the receiver side the input signal can be regenerated from the quantized differential signal the final signal-to-noise ratio can be calculated as

$$\mathrm{SNR} = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\epsilon(n)^2)}, \tag{3.11}$$

- However, instead of the input signal, the differential signal is quantized, setting the quantization SNR to

$$\mathrm{SQNR} = \frac{\mathbb{E}(\delta(n)^2)}{\mathbb{E}(\epsilon(n)^2)} = 20\log_{10}2^N. \tag{3.12}$$

Rewriting the above equations results in the total SNR of

$$\mathrm{SNR} = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\epsilon(n)^2)} = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\delta(n)^2)} \cdot \underbrace{\frac{\mathbb{E}(\delta(n)^2)}{\mathbb{E}(\epsilon(n)^2)}}_{20\log_{10}2^N}, \tag{3.13}$$

revealing that compared to the direct quantization of the input signal the signal-to-noise ratio is increased by a factor of

$$\mathrm{G}_p = \frac{\mathbb{E}(\tilde{\xi}(n)^2)}{\mathbb{E}(\delta(n)^2)} \tag{3.14}$$

termed as the **prediction gain**, being a large number, assuming that the input signal can be predicted precisely. This arises the question, how the actual input sample can be estimated based on the previous samples only.

3.1.3 The optimal prediction coefficients

As the most simple approach the actual input sample $\xi(n)$ can be predicted as the linear combination of the previous N number of samples, written in the form of

$$\tilde{\xi}(n) = \sum_{m=1}^{N} p(m)\xi(n-m) = \mathbf{p}^{\mathrm{T}}\xi_{n-1}, \tag{3.15}$$

written in a vectorial form. In the expression vector $\mathbf{p} = [p(1), p(2), ..., p(N)]^{\mathrm{T}}$ contains the weights of the previous input samples used for prediction, and vector $\xi_{n-1} = [\xi(n-1), \xi(n-2), ..., \xi(n-N)]^{\mathrm{T}}$ contains the previous N number of the input samples.

The goal is to minimize the expected energy of the difference between the actual input sample $\xi(n)$ and the prediction $\hat{\xi}(n)$ by optimizing the prediction weights $\mathbf{p}^{\mathrm{T}}$ so that

$$\arg\min_{\mathbf{p}} : \mathbb{E}\left(|\xi(n) - \tilde{\xi}(n)|^2\right) = \arg\min_{\mathbf{p}} : \mathbb{E}\left(|\xi(n) - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) \tag{3.16}$$

holds. The quadratic expression can be expounded to

$$\begin{aligned}\mathbb{E}\left(|\xi(n)^2 - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) = \mathbb{E}\left(\xi(n) - 2\xi(n)\mathbf{p}^{\mathrm{T}}\xi_{n-1} + \mathbf{p}^{\mathrm{T}}\xi_{n-1}\xi_{n-1}^{\mathrm{T}}\mathbf{p}\right) = \\ \mathbb{E}\left(\xi(n)^2\right) - 2\mathbf{p}^{\mathrm{T}}\,\mathbb{E}\left(\xi(n)\xi_{n-1}\right) + \mathbf{p}^{\mathrm{T}}\,\mathbb{E}\left(\xi_{n-1}\xi_{n-1}^{\mathrm{T}}\right)\mathbf{p}\end{aligned} \tag{3.17}$$

with exploiting the linearity of expected value operator and collecting non-stochastic quantities outside of it. The expected value of the scalar-vector product and the dyadic product terms of the expression can be recognized as the autocorrelation values of the input signals, rewritten in a matrix form as

$$\mathbb{E}\left(|\xi(n)^2 - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) = r_\xi(0) - 2\mathbf{p}^{\mathrm{T}}\mathbf{r}_\xi + \mathbf{p}^{\mathrm{T}}\mathbf{R}_\xi\mathbf{p} \tag{3.18}$$

with denoting the signal energy, and the autocorrelation vector and matrix as

$$r_\xi(0) = \mathbb{E}\left(\xi(n)^2\right), \qquad \mathbf{r}_\xi = \begin{bmatrix} r_\xi(1) \\ r_\xi(2) \\ \dots \\ r_\xi(N) \end{bmatrix},$$
$$\mathbf{R}_\xi = \begin{bmatrix} r_\xi(0) & r_\xi(1) & \dots & r_\xi(N-1) \\ r_\xi(1) & r_\xi(0) & .... & r_\xi(N-2) \\ \dots & & & \\ r_\xi(N-1) & r_\xi(N-2) & \dots & r_\xi(0) \end{bmatrix}. \tag{3.19}$$

Expression 3.18 has to be minimized with respect to vector $\mathbf{p}^{\mathrm{T}}$. The minimization can be performed by finding the zero of the derivative of the expression with respect to vector $\mathbf{p}^{\mathrm{T}}$, reading

$$\frac{\partial}{\partial \mathbf{p}^{\mathrm{T}}} \mathbb{E}\left(|\xi(n)^2 - \mathbf{p}^{\mathrm{T}}\xi_{n-1}|^2\right) = -2\mathbf{r}_\xi + 2\mathbf{R}_\xi\mathbf{p} = 0. \tag{3.20}$$

Finally, from the above equation the optimal prediction coefficient vector can be expressed as

$$\mathbf{p}^{\mathrm{T}} = \mathbf{R}_\xi^{-1}\mathbf{r}_\xi. \tag{3.21}$$

The above coefficients are the so-called **Wiener filter** coefficients for the estimation of a stationary stochastic process.

From the form of the optimal prediction coefficients it is clear that the signal estimation is based on the measured correlation of the previous source samples, therefore prediction is efficient as long as neighboring samples are linearly related. Hence the optimal prediction is often termed as **linear prediction**. The above Wiener filter is, therefore, capable of the estimation of the correlated part of the input signal.

3.1.4 Prediction as FIR filtering

It should be noted that linear prediction (3.15) describes the discrete linear convolution of vectors $\mathbf{p}$ and ξ_{n-1}. This means that the estimation of the actual sample can be obtained by the simple FIR filtering[1] of the input stream with the coefficient vector $\mathbf{p}$. The result of estimation is subtracted from the input sample, generating the differential signal, which, therefore, can be written as

$$\delta(n) = \xi(n) - \sum_{m=1}^{N} p(m)\xi(n-m), \tag{3.22}$$

[1]The term FIR (Finite Impulse Response) filtering refers to the fact that the applied filter contains no feedback, thus it is ensured that to an excitation with finite extent the filter output is of finite extent. The actual filter impulse response is described by the coefficient vector itself.

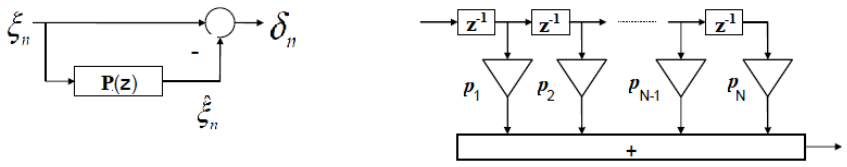


Figure 3.4: One dimensional and two dimensional white noise process (a) and correlated noise process (b).

or, transforming the equation to the z-transform domain—by exploiting that delay by one sample is a multiplication by z^{-1} in the z-domain—as

$$\delta(z) = \xi(z)\left(1 - \sum_{m=1}^{N} p_m(z)\, z^{-m}\right) = \xi(z)\,(1 - P(z))\,. \tag{3.23}$$

The realization of linear prediction with FIR is depicted in Figure 3.4. The structure of FIR filtering is depicted in Figure 3.4 (b). The prediction filter can be also interpreted as an accumulator, or memory, containing the previous samples, added with different weights to the output.

It is important to note that the prediction filter $P(z)$ is capable of identifying linear tendencies in the input signal and the actual sample is estimated based on the assumed linear relationship between previous samples Once all the correlated part of the input signal is removed, by definition, in the remaining differential signal each sample is uncorrelated from the previous samples. Thus, with theoretical optimal prediction, filter $(1 - P(z))$ **decorrelates** the input signal, and the differential signal is a **white noise process**: The filter is often referred to as whitening filter, since in the optimal case it transforms the input signal into white noise.

Figure 3.5 illustrates the whitening process in case of a simple 1D input signal, in the present example being an audio stream. The correlation of the signal can be estimated based on the input stream according to (3.6) from which the linear prediction coefficients are obtained (from (3.21)). These coefficients are applied to perform whitening filtering process described by 3.23.

Figure 3.5 (a) compares the time histories of the input and the differential signals. As it is illustrated the dynamics range of the input signal is significantly reduced by subtracting the predicted signal. Generally speaking, periodic signals can be predicted efficiently by measuring correlation, therefore, harmonic signals are almost entirely removed from the input. Obviously, transients can not be predicted based on previous samples, therefore, they are still present in the differential signals.

Figure 3.5 (b) verifies that as the effect of filtering the autocorrelation of the differential signal became approximately a delta function, hence the output stream is nearly uncorrelated. This is also verified by comparing the input and output spectral density functions, with the output spectrum being nearly constant as depicted in Figure 3.5 (c).

Note that since the input signal is typically of low-pass filtered characteristics,

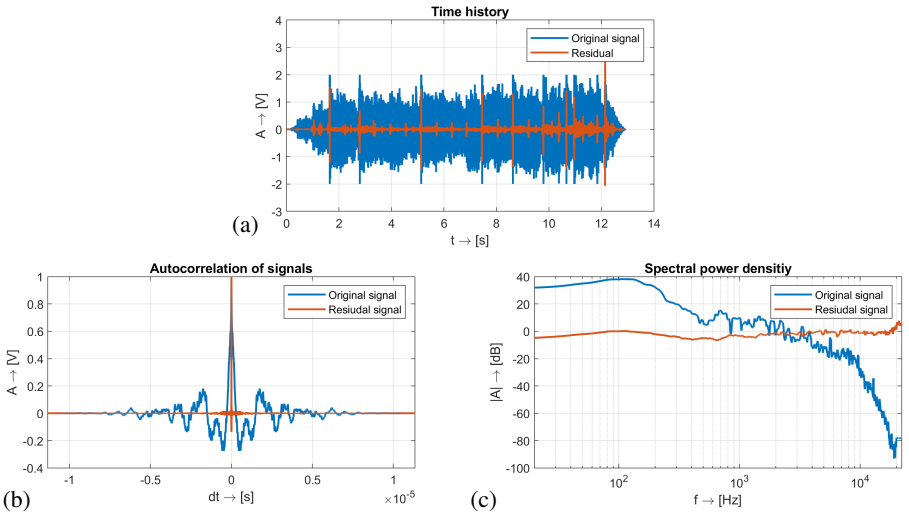


Figure 3.5: Example for the optimal linear prediction of an audio input stream.

therefore, the whitening filter has to be of high-pass characteristics in order to flatten out the output spectrum. This statement can be generalized to typical audio and video signals: since natural signals are usually dominated by low-frequency content, therefore, in practical applications the differential signal (i.e. prediction) is obtained as the high-pass filtered version of the input signal.

It has been highlighted that once the autocorrelation of the input signal can be estimated (e.g. by measurement) nearly optimal prediction coefficients can be defined. Hence, the prediction filter $P(z)$ depends on the actual input signal. Obviously, the decoding of the differential signal also requires the knowledge of coefficients $P(z)$ in the decoder side. In order to avoid the transmission of the prediction filter coefficients, in practice as a sub-optimal solution fixed prediction coefficients are applied for high-pass filtering. In the following only this fix-coefficient approach is considered.

3.1.5 Problem of feedforward prediction

Although being a very simple approach, the direct FIR filtering scheme, presented in Figure 3.4 (a) is never used directly in practice, due to the following reason:

So far only the prediction and generation of the differential signal have been discussed in details, without taking the receiver side into consideration. In the receiver—with assuming that the prediction filter coefficients are known—the original samples are reconstructed by adding the previously accumulated, weighted samples to the residual signal. With denoting the decoded signal by $\tilde{\xi}$ it can be written as

$$\tilde{\xi}(z) = \delta'(z) + \sum_{m=1}^{N} p_m(z)\tilde{\xi}(z)z^{-m} \quad \rightarrow \quad \frac{\tilde{\xi}(z)}{\tilde{\delta}(z)} = \frac{1}{1-P(z)}, \tag{3.24}$$

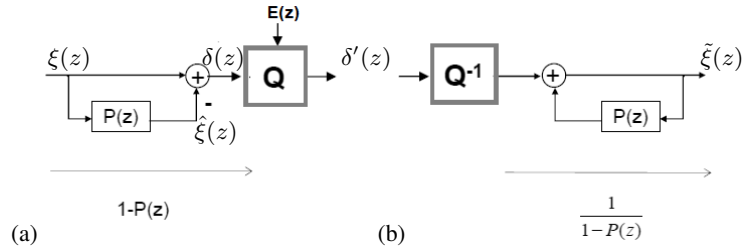


Figure 3.6: Example for the optimal linear prediction of an audio input stream.

hence, in the receiver the original signal can be reconstructed by **inverse filtering** and the transfer function of the inverse filter is the reciprocal of the forward filter. Obviously, $\frac{1}{1-P(z)}$ describes an IIR (Infinite Impulse Response) filter, since the reconstruction in the receiver is performed by feedbacking the output signal to the input of the receiver. Hence, the zeros of the forward FIR filter are mapped to the poles of the inverse filter, leading to stability problems in the receiver.

These stability problems causes serious artifacts due to the presence of the quantizer: As discussed earlier the presence of quantizer can be modeled as introducing quantization noise, added directly to the differential signal. Hence, with denoting the z-transform of the additive quantization noise by $\epsilon(z)$ and taking both the encoding and decoding steps into consideration the decoded signal can be written as

$$\tilde{\xi}(z) = \left(\underbrace{\xi(z)\,(1-P(z))}_{\delta(z)} + \epsilon(z) \right) \cdot \frac{1}{1-P(z)} = \xi(z) + \epsilon(z)\frac{1}{1-P(z)}. \qquad (3.25)$$

The decoded signal, therefore, consists of the original input signal and the quantization noise, filtered with the potentially unstable inverse filter.

In order to highlight the resulting effect as a simple example DPCM encoding is investigated.

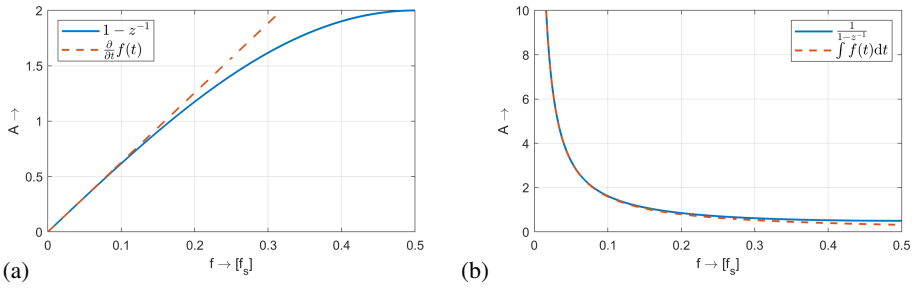
DPCM coding with feedforward prediction:
As the simplest approach for differential coding the basis of prediction is simply the previous sample of the input signal, hence

$$\begin{aligned} \delta(n) &= \xi(n) - \xi(n-1) \\ P(z) &= z^{-1} \quad \rightarrow \quad 1 - P(z) = 1 - z^{-1}. \end{aligned} \qquad (3.26)$$

The differential sample is simply given as the difference of the actual and the previous input samples. The approach is termed as **Differential Pulse-Code Modulation**.

Generally speaking, the derivative of the a continuous function most simply can be approximated numerically by the finite difference

$$\frac{\partial}{\partial t} f(t) \approx \frac{f(t) - f(t-T)}{T} = \frac{1}{T}\left(f(n) - f(n-1)\right), \qquad (3.27)$$



(a) (b)

Figure 3.7: Example for the optimal linear prediction of an audio input stream.

termed as the backward Euler scheme, with T being the sampling interval. Comparison of this expression with (3.26) clearly reveals that differential coding realizes the approximate differentiation of the input signal prior to quantization. The frequency response of this simple differentiation is obtained analytically by evaluation the z-transform along the unit circle, i.e. by the substitution $z = \mathrm{e}^{\mathrm{j}2\pi f/f_s}$, with f_s being the sampling frequency

$$|1 - z^{-1}| = |1 - \mathrm{e}^{-\mathrm{j}2\pi f/f_s}| = 2\left|\sin\frac{\pi f}{f_s}\right|. \tag{3.28}$$

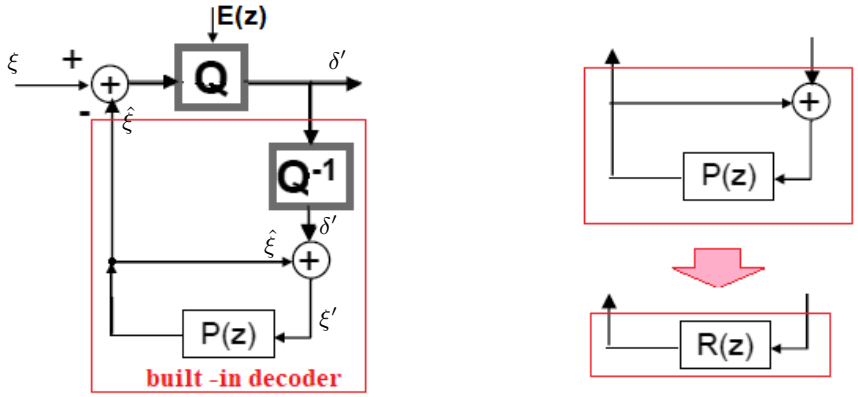
The frequency response of the filter is shown in Figure 3.7 (a) along with the frequency response of an ideal differentiator. The filter has a zero at $z = 1$, at zero frequency (as the derivative of a constant signal is zero), and approximates an ideal differentiator perfectly in the low frequency region.

Obviously, the inverse operation of differentiation is integration, i.e. the inverse filter $\frac{1}{1-P(z)}$ describes the numerical approximation of integration. This is verified by Figure 3.7 (b), depicting the frequency response of the inverse filter and that of an ideal integrator, with a pole at zero frequency (the integral of constant signal tends to infinity).

As a conclusion, the drawback of the simple feedforward processing scheme—where the encoder performs prediction directly from the input signal—is the following: According to (3.25) the quantization noise is reproduced in the output of the decoder filtered with the inverse filter $\frac{1}{1-z^{-1}}$. Since this filter response describes the integration of the filtered signal, therefore, the quantization noise is integrated, **accumulated** in the decoder. Less formally speaking: the source of accumulation of quantization error is the fact that due to the presence of quantization the basis of prediction is different in the encoder and the decoder side. While the encoder predicts the next sample from the original signal, in the decoder side only the quantized, decoded previous samples are available for the next prediction.

3.1.6 Feedback prediction loop

In order to overcome the error of feedforward prediction and to avoid the accumulation of the quantization error it has to be ensured that the encoder and the decoder uses



(a) (b)

Figure 3.8: One dimensional and two dimensional white noise process (a) and correlated noise process (b).

the same past samples for prediction. This can be achieved only by ensuring that the encoder predicts from the quantized input samples as well. Hence, in the encoder side the quantized differential signal has to be decoded, which decoded signal will serve as the basis for the next prediction: the decoder has to be built in the encoder in a feedback loop.

The concept of differential quantizer with built-in decoder stage is depicted in Figure 3.8 (a): the accumulator/filter $P(z)$ contains the previous quantized samples (in the previous example only the previous, quantized sample) and produces the actual estimation $\hat{\xi}'$ as their weighted sum. In the actual time step the estimation is subtracted from the actual input signal, and the difference is quantized, producing the quantized differential output signal. Besides transmission to the receiver, the quantized differential signal is added to the estimated sample, producing the decoded, quantized signal ξ', which is pushed into the accumulator and will serve as the basis of the prediction in the next time step. Hence, it is ensured that the basis of prediction is the quantized signal, similarly to the receiver side.

Mathematically the entire built in decoder can be modeled as a single transfer function: The input of the decoder block is the quantized differential signal δ', while the output is the estimation $\hat{\xi}$, i.e.

$$R(z) = \frac{\hat{\xi}(z)}{\delta'(z)} = \frac{P(z)\left(\hat{\xi}'(z) + \delta'(z)\right)}{\delta'(z)} \quad \rightarrow \quad R(z) = \frac{P(z)}{1 - P(z)} \tag{3.29}$$

holds. The transmitted differential signal can be written with taking the quantization error into consideration as

$$\begin{aligned} \delta'(z) &= \xi(z) - R(z)\delta'(z) + \epsilon(z) \quad \rightarrow \quad (1 + R(z))\,\delta'(z) = \xi(z) + \epsilon(z) \\ \delta'(z) &= \frac{1}{1 - R(z)}\left(\xi(z) + \epsilon(z)\right) = (1 - P(z))\left(\xi(z) + \epsilon(z)\right) \end{aligned} \tag{3.30}$$

and finally after decoding by applying the unchanged decoder with the transfer function of $\frac{1}{1-P(z)}$ the decoded signal reads

$$\tilde{\xi}(z) = \frac{1}{1-P(z)}(1-P(z))\left(\xi(z)+\epsilon(z)\right) = \xi(z)+\epsilon(z). \tag{3.31}$$

Therefore, without the accumulation or the error, besides the original input the decoded signal contains only the actual quantization noise. Due to this reason in differential coding exclusively the above feedback structure is applied with the encoder containing the built-in decoder stage.

3.1.7 Application of predictive coding in video technologies

Predictive coding is extensively applied in the field of image and video compression. Prediction can be applied either between pixels or blocks of the image, performing spatial prediction, or between consequent images, applying temporal prediction:

- **Temporal prediction:** MPEG video encoders apply block based temporal DPCM prediction, meaning that each block of a frame is predicted from a block from the previous frame. Hence, the prediction filter/accumulator contains a single block of the previous image, which is subtracted from the actual input block. Newer compression methods, like H.264, allows prediction using multiple reference blocks with arbitrary weights, therefore, the length of the prediction filter may be larger than one, approximating the optimal prediction case, discussed in the foregoing. The block in the prediction filter, termed as the **reference block**, does not necessarily located in the same position as the actual input block, but its location is found where the the reference block resembles the most to the actual input block. Finding the position of the reference block is the basic task of **motion estimation.** Motion estimation and temporal prediction of the MPEG encoder is discussed in details in the following chapter.

- **Spatial prediction:** Alternatively, within an image each pixel may be predicted based on the neighboring pixel values, followed by requantization of the difference pixel. Assuming that pixel values change slowly over the image, storing only the difference from the previous pixels may result in significant data compression. Pixel based predictive coding is applied by the PNG image encoder as discussed in the following, while modern video compression methods, e.g. H.264 allows more sophisticated block based intra-image prediction.

  Furthermore, all JPEG and MPEG block-based video encoders stores the average luminance/color of the actual block predictivly compared to the previous block in a DPCM loop.

  As a rule of thumb, since in the decoder side only the requantized difference pixels/blocks are available, in order to avoid the accumulation quantization error predictive coding is in all cases performed within a feedback prediction loop.

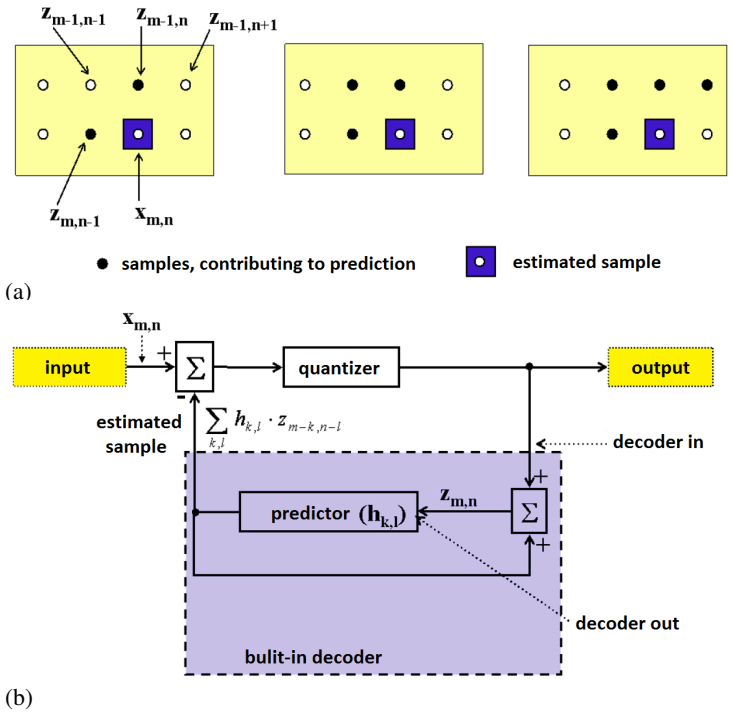


Figure 3.9: Prediction schemes (a) and block diagram (b) of PNG encoder.

3.1.7.1 The PNG encoder

As a simple example for spatial prediction the PNG encoder is discussed briefly. PNG (Portable Network Graphics), being one of the simplest image compression methods apply a lossless spatial prediction along with lossless entropy coding, with the following properties:

- Since PNG is a lossless compression, therefore, no quantization is present in the encoder.
- As PNG is lossless, luma/chroma separation is not beneficiary, but direct RGB coordinates are encoded.
- Compression is achieved by the fact that the result of spatial prediction contains mainly small pixel intensities, clustered around 0, which can be efficiently entropy coded with a properly chosen variable-length coding. The entropy coder of PNG is termed as DEFLATE coding, being originally introduced for the early version of ZIP comression.
- Before DEFLATE compression the actual pixel, denoted in Figure 3.9 by $x_{m,n}$ is predicted from the neighboring pixels. Obiviously, only previously encoded and transmitted pixels can be the reference of prediction, denoted by $z_{m,n}$. The

number of reference pixels can be chosen by the encoder from the prediction schemes, shown in Figure 3.9. Therefore, in this case linear prediction is based on a 2D FIR filter, containing 2, 3 or 4 values in the prediction accumulator, denoted by $h_{k,l}$.

- Note that in this exclusive case—since no quantization is performed—feedback prediction may be interchangeable with feedforward structure.

3.2 Transform coding

Alternatively to linear prediction, potential linear dependence between samples can be eliminated by representing the input data in a new representation space as the linear combination of properly chosen basis vectors (or matrices). The change of basis can be performed by applying a linear transform to the input. With an optimal choice of the linear transform the weights of the basis vectors/matrices—i.e. the representation of the data in the new basis—are less correlated, with reduced spatial redundancy, and allows more efficient quantization adapted to the properties of human vision.

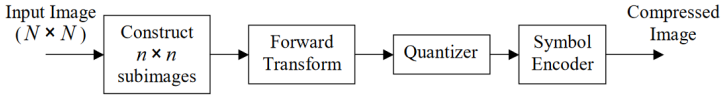


Figure 3.10: General block scheme of image transform encoder.

The general processing scheme for image coding is depicted in Figure 3.10. In image processing applications usually block-based coding is applied, with the input image segmented to $N \times N$ sized pixel blocks. The input blocks are transformed by a properly chosen 2D linear transform, representing the pixel block as the weighted sum of basis images. Similarly to prediction, the linear transform itself is theoretically reversible. The irreversible, lossy coding is usually the requantization of the transformed data, followed by an efficient reversible entropy coding method.

3.2.1 1D and 2D linear transforms

Before discussing the 2D linear transforms applied frequently in image coding, the general theoretical basics of linear transforms are revisited.

3.2.1.1 One-Dimensional Transforms

Let $x(n) = [x(0),\, x(1),\, ...,\, x(N-1)]^{\mathrm{T}}$ denote the **input vector** $\mathbf{x}$ of N samples and $y(k) = [y(0),\, y(1),\, ...,\, y(N-1)]^{\mathrm{T}}$ the **transform vector** $\mathbf{y}$ with the same size.

The connection of the input and transform vectors is established by

$$y(k) = \sum_{n=0}^{N-1} x(n)\, A(k,n), \qquad \text{for } k = 0, 1, ..., N-1, \tag{3.32}$$

or in matrix notation

$$\mathbf{y} = \mathbf{A}\mathbf{x}, \tag{3.33}$$

where $A(k, n)$ is the forward transformation kernel, forming the **A forward transform matrix**, and $y(k)$ are termed as the transform coefficients. The inverse transform that recovers the input sequence is given by

$$x(n) = \sum_{k=0}^{N-1} y(k)\, B(k, n), \qquad \text{for } n = 0, 1, ..., N-1, \tag{3.34}$$

or in matrix notation

$$\mathbf{x} = \mathbf{B}\mathbf{y}, \tag{3.35}$$

where $\mathbf{B}$ is the **inverse transform matrix**, satisfying $\mathbf{B} = \mathbf{A}^{-1}$.

Let $\mathbf{b}_k$ denote the k-th basis vector of the new representation space. The inverse transform (3.34) can be rewritten with a slightly modified notation

$$x(n) = \sum_{k=0}^{N-1} y(k)\, b_k(n), \qquad \text{for } n = 0, 1, ..., N-1, \tag{3.36}$$

clearly reflecting the fact that the inverse transform reproduces the original input vector as the weighted sum of the basis vectors, and the weights are given by the transform coefficients $y(k)$. Hence, linear inverse transform can be interpreted as the series expansion of the input signal. The series expansion can be written in a matrix form as

$$\begin{bmatrix} \\ \mathbf{y} \\ \\ \end{bmatrix} = \underbrace{\left[\begin{array}{c|c|c|c} & & & \\ \mathbf{b}_0 & \mathbf{b}_1 & ... & \mathbf{b}_{N-1} \\ & & & \end{array}\right]}_{\mathbf{B}} \cdot \begin{bmatrix} \\ \mathbf{x} \\ \\ \end{bmatrix}. \tag{3.37}$$

The above expression shows that generally speaking, the columns of the inverse transform matrix are the basis vectors of the new representations space.

In the aspect of image compression the class of **orthogonal transform**—or in case of complex valued basis vectors the **unitary transforms**—are of special importance. In this case the basis vectors form an orthonormal set, their scalar product satisfying

$$\mathbf{b}_i^{*\mathrm{T}} \cdot \mathbf{b}_j = \sum_{n=0}^{N-1} b_i^*(n) \cdot b_j(n) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \tag{3.38}$$

with $*$ denoting complex conjugate. Therefore, in case of a real orthogonal transform the rows and the columns of the inverse transform matrix are orthonormal, and

$$\mathbf{B}^{*\mathrm{T}}\mathbf{B} = \mathbf{E} \quad \rightarrow \quad \mathbf{B}^{*\mathrm{T}} = \mathbf{B}^{-1} = \mathbf{A} \tag{3.39}$$

holds, where $\mathbf{E}$ is the identity matrix.

Therefore, in case of a real, orthogonal transform the rows of the forward transform matrix contain the basis vectors:

$$\begin{bmatrix} \\ \mathbf{x} \\ \\ \end{bmatrix} = \underbrace{\begin{bmatrix} \\ \mathbf{b}_0^{\mathrm{T}} \\ \mathbf{b}_1^{\mathrm{T}} \\ \dots \\ \mathbf{b}_{N-1}^{\mathrm{T}} \\ \\ \end{bmatrix}}_{\mathbf{A}} \cdot \begin{bmatrix} \\ \mathbf{y} \\ \\ \end{bmatrix}, \tag{3.40}$$

and the k-th element of the coefficient vector is calculated as

$$y(k) = \sum_{k=0}^{N-1} x(n)\, b_k(n), \qquad \text{for } k = 0, 1, ..., N-1, \tag{3.41}$$

describing the scalar product of the k-th basis vector and the input vector. Hence, in case of a orthogonal (or unitary) transform the transform coefficients are obtained by projecting the input vector to the basis vectors.

Orthogonality is clearly a necessary property for basis vectors that are used to decompose an input into uncorrelated components in an N-dimensional space, otherwise linear relationship exist between the basis vectors. Orthonormality of basis vector is a stronger property ensuring the preservation of signal energy in the transform space following

$$|\mathbf{y}|^2 = \mathbf{y}^{\mathrm{T}}\mathbf{y} = (\mathbf{A}\mathbf{x})^{\mathrm{T}}\,\mathbf{A}\mathbf{x} = \mathbf{x}^{\mathrm{T}}\underbrace{\mathbf{A}^{\mathrm{T}}\mathbf{A}}_{\mathbf{E}}\mathbf{x} = \mathbf{x}^{\mathrm{T}}\mathbf{x} = |\mathbf{x}|^2. \tag{3.42}$$

This is the Parseval-theorem.

3.2.1.2 Two-Dimensional Transforms

One-dimensional transforms can be easily generalized towards the two-dimensional case. The forward and inverse transforms generalized as

$$\begin{aligned} Y(k,l) &= \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} X(n,m)\, A(k,l,m,n), \qquad \text{for } k,l = 0,1,...,N-1, \\ X(m,n) &= \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} Y(k,l)\, B(k,l,m,n), \qquad \text{for } m,n = 0,1,...,N-1, \end{aligned} \tag{3.43}$$

with $X(m,n)$ being the $N \times N$ sized input matrix, representing e.g. non-overlapping pixel intensities of an input image, $Y(k,l)$ the coefficient matrix, and $A(k,l,m,n)$ and $B(k,l,m,n)$ being the forward and inverse transform hyper-matrices. By using notation $A(k,l,m,n) \rightarrow A_{k,l}(m,n)$ the orthogonal 2D transform and inverse transform is defined as

$$\begin{aligned} Y(k,l) &= \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} X(n,m)\, A_{k,l}(m,n), \\ X(m,n) &= \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} Y(k,l)\, A_{k,l}^{\mathrm{T}}(m,n). \end{aligned} \tag{3.44}$$

Clearly, $A^{\mathrm{T}}_{k,l}(m,n)$ describes the k,l-th **basis matrix**, with the forward transform representing the projection of the input block to the actual basis matrix, while the inverse transform describes the expansion of the input matrix as the weighted sum of basis matrices. The coefficient matrix $Y(k,l)$ gives the weight of the k,l-th basis matrix in the input image.

The evaluation of the 2D transforms of (3.43) may be computationally expensive. In practice **separable transforms are applied** in which case the transform hypermatrices can be written as a product of two matrices

$$A(k,l,m,n) = A_1(k,n) \cdot A_2(l,m), \tag{3.45}$$

meaning that the basis matrices $A_{k,l}(m,n)$ can be also written

$$A_{k,l}(m,n) = A_k(m) \cdot A_l(n) \tag{3.46}$$

as the product of a row and column vector (i.e. as a dyadic product).

In this case the forward transform takes the form

$$\begin{aligned} Y(k,l) &= \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} X(n,m)\, A_1(k,n) \cdot A_2(l,m) = \\ &= \sum_{m=0}^{N-1} A_2(l,m) \cdot \left(\sum_{n=0}^{N-1} A_1(k,n) X(n,m) \right), \end{aligned} \tag{3.47}$$

or written in a matrix form

$$\mathbf{Y} = \left(\mathbf{A}_2 \left(\mathbf{A}_1 \mathbf{X} \right)^{\mathrm{T}} \right)^{\mathrm{T}} = \left(\mathbf{A}_2 \mathbf{X}^{\mathrm{T}} \mathbf{A}_1^{\mathrm{T}} \right)^{\mathrm{T}} = \mathbf{A}_1 \mathbf{X} \mathbf{A}_2^{\mathrm{T}}. \tag{3.48}$$

Obviously, left-multiplication by $\mathbf{A}_1$ describes the 1D linear transform of each column of $\mathbf{X}$ by the transform described by $\mathbf{A}_1$, while right-multiplication describes the 1D linear transform of each rows of the matrix, described by $\mathbf{A}_2$. Hence, a separable transform can be written as two consequent vertical and horizontal 1D transforms, allowing efficient numerical evaluation. Similarly, the orthogonal inverse transform can be written as

$$\mathbf{X} = \left(\mathbf{A}_2^{\mathrm{T}} \left(\mathbf{A}_1^{\mathrm{T}} \mathbf{Y} \right)^{\mathrm{T}} \right)^{\mathrm{T}} = \left(\mathbf{A}_2^{\mathrm{T}} \mathbf{Y}^{\mathrm{T}} \mathbf{A}_1 \right)^{\mathrm{T}} = \mathbf{A}_1^{\mathrm{T}} \mathbf{Y} \mathbf{A}_2. \tag{3.49}$$

In most cases the vertical and horizontal transforms are identical ($\mathbf{A}_1 = \mathbf{A}_2 = \mathbf{A}$). In this case the 2D transform can be evaluated by

$$\mathbf{Y} = \mathbf{A}\mathbf{X}\mathbf{A}^{\mathrm{T}}, \qquad \mathbf{X} = \mathbf{A}^{\mathrm{T}}\mathbf{Y}\mathbf{A}. \tag{3.50}$$

In image processing the basis matrix $A^{\mathrm{T}}_{k,l}(m,n)$ is usually referred to as **basis images**, referring to the fact that inverse transform describes the original image block as the weighted sum of the basis images. Similarly, forward transform is the decomposition of the input image to basis images.

3.2.2 Transforms coding of images

The linear transform in itself is lossless: the consequent forward and inverse transform return the original image block. Irreversible coding is applied in the transform space by the quantization of the transform coefficients. The Parseval theorem (3.42) ensures that the total quantization noise energy remains the same in the spatial domain after the corresponding inverse transform, however, the spatial distribution of the noise depends on the actual form of inverse transform.

The basic goal of representing input image blocks in the transform space is to find a representation which can be quantized more efficiently than the direct quantization of the image. The efficiency of a linear transform can be qualified as follows:

- The linear transform is usually followed by requantization the transformed coefficients. However, while in the spatial domain each pixel matrix element contributes approximately equally to the total image, in the transform domain usually most of the energy is concentrated to several coefficients. If the transformed representation is **compact**—meaning most of the input energy is distributed amongst a few coefficients—then basis vectors with low energy can be either quantized more coarsely, or can be completely omitted in the inverse transform by applying a truncated series expression. This can lead to significant compression. Due to the Parseval theorem, the energy of the quantization error is, of course, the same in the spatial domain, but quantization noise will be distributed among the pixels uniformly.

  Hence, the efficiency of the transform applied is described by the compactness of the representation [2].

- Alternatively, the same requirement can be formulated via correlation in the transform domain: In order to reduce spatial redundancy, i.e. correlation of the pixel values, the transformed coefficient should be uncorrelated. It can be verified that the compactness of representation is completely equivalent with being uncorrelated (so the above two requirements are equivalent).

Defining a measure for the compactness of the transform is not straightforward. On the other hand, the correlation of the transformed coefficients can be easily defined, allowing the derivation of an optimal linear transform: the Karhunen-Loeve transform

3.2.3 The Karhunen-Loeve transform

Being the optimal solution the Karhunen-Loeve transform (KLT) is defined so that in the transform space the coefficients are completely correlated. The goal is then, with a known input signal the linear transform has to be defined, which decorrelates the input samples. Note that the basic problem is very similar to the optimal linear prediction approach, which results in filter coefficients, completely estimating the correlated content of an input signal, resulting in a completely decorrelated differential signal.

[2] As a simple example the Fourier transform of a constant signal contains only a DC coefficient, hence, the spectral representation is much more compact than the spatial one.

3.2.3.1 1D KLT:

In order to arrive at the KLT solution assume that the input vector is a realization of a wide-sense stationary stochastic process, denoted by $x(n)$, with the *mean value being zero*. Let $\mathbf{R}_x$ denote the correlation matrix of the input signal, with the elements given as

$$R_x(m,n) = \mathbb{E}(x(m) \cdot x(n)) \quad \rightarrow \quad \mathbf{R}_x = \begin{bmatrix} r_x(0) & r_x(1) & ... & r_x(N-1) \\ r_x(1) & r_x(0) & .... & r_x(N-2) \\ ... & & & \\ r_x(N-1) & r_x(N-2) & ... & r_x(0) \end{bmatrix} \tag{3.51}$$

giving the estimated correlation between the elements of the input registry (similar to the concept used in optimal prediction theory). The correlation matrix is therefore a symmetric (in case of complex input a hermitian) matrix. Furthermore, assume that the eigendecomposition of the correlation matrix is known[3] so that it can be written in the form

$$\mathbf{R}_x = \mathbf{U}_x \mathbf{D}_x \mathbf{U}_x^{\mathrm{T}}, \text{ with } \mathbf{D}_x = \begin{bmatrix} \lambda_0 & 0 & ... & 0 \\ 0 & \lambda_1 & .... & 0 \\ ... & & & \\ 0 & 0 & ... & \lambda_{N-1} \end{bmatrix}, \quad \mathbf{U}_x = \left[\begin{array}{c|c|c|c} \mathbf{v}_0 & \mathbf{v}_1 & ... & \mathbf{v}_{N-1} \end{array}\right]. \tag{3.52}$$

In the decomposition the diagonal matrix $\mathbf{D}_x$ contains the eigenvalues of $\mathbf{R}_x$ in its diagonal, and matrix $\mathbf{U}_x$ contains the eigenvectors of $\mathbf{R}_x$ in its columns.

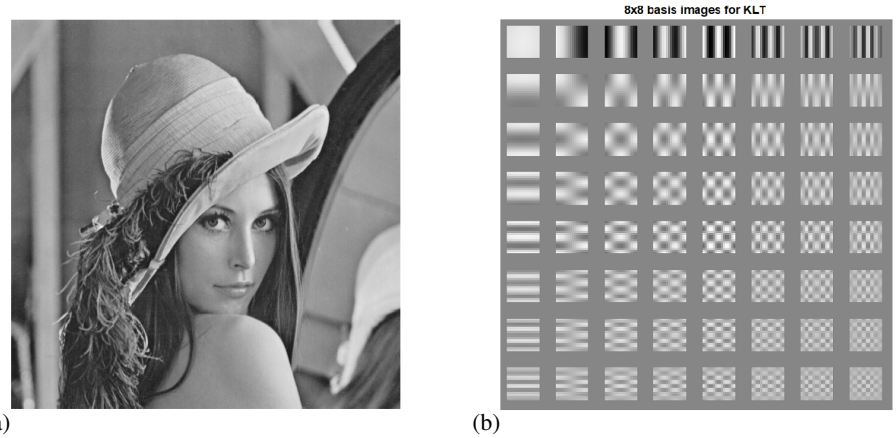
By definition, a process is uncorrelated only if its correlation matrix is diagonal (all the samples are correlated only with themselves). Hence, the optimal transform is found for a given input vector, so that the correlation matrix of the result of transformed vector is diagonal, being the basic idea behind KLT. The correlation matrix of the transformed vector $\mathbf{y}$ is obtained as

$$\begin{aligned} R_y(m,n) &= \mathbb{E}(y(m) \cdot y(n)) = \\ &\mathbb{E}(\sum_{m=0}^{N-1} A(k,m)\, x(m) \cdot \sum_{n=0}^{N-1} A(l,n)\, x(n)) = \\ &\sum_{m=0}^{N-1} A(k,m) \sum_{n=0}^{N-1} A(l,n)\ \mathbb{E}(x(m) \cdot x(n)) = \\ &\sum_{m=0}^{N-1} A(k,m) \sum_{n=0}^{N-1} A(l,n) R_x(m,n), \end{aligned} \tag{3.53}$$

which can be written in a matrix form as

$$\mathbf{R}_y = \mathbf{A}^{\mathrm{T}} \mathbf{R}_x \mathbf{A} \tag{3.54}$$

[3] For a symmetric matrix the eigendecomposition is guaranteed to exist with the eigenvectors forming a full orthonormal basis



(a) (b)

Figure 3.11: Test image (a) and basis images (b) obtained from 2D Karhunen-Loeve transform.

As and educated guess the transform matrix is chosen as $\mathbf{A} = \mathbf{U}_x$, and substituting the eigendecomposition (3.53) into $\mathbf{R}_x$ the correlation matrix reads as

$$\mathbf{R}_y = \underbrace{\mathbf{U}_x^{\mathrm{T}}\mathbf{U}_x}_{\mathbf{E}}\mathbf{D}_x\underbrace{\mathbf{U}_x^{\mathrm{T}}\mathbf{U}_x}_{\mathbf{E}} = \mathbf{D}_x \tag{3.55}$$

The autocorrelation of the transform vector is, thus, a diagonal matrix.

The optimal solution, therefore, constructs the transform matrix as its columns being the eigenvectors of the input signal's autocorrelation matrix. As a result in the transform domain the coefficients are completely decorrelated. This statement is equivalent to that the KLT transform of the input signal gives the most compact representation in a sense that it reduces the total mean-square error resulting of the truncation of the inverse transform series expansion.

It should be noted that for processes with non-zero mean value instead of the correlation matrix, the **covariance matrix** $\mathbf{K}_x$ must be used, which is related to the correlation matrix for a WSS process as

$$\mathbf{K}_x = \mathbf{R}_x - m_x^2, \tag{3.56}$$

where m_x is the expected value of the process, hence, the covariance equals the correlation biased to zero mean value. In image processing the luminance signal takes it values between 0 and 1, therefore, it has a non-zero mean value. Thus, in image processing instead of the correlation matrix the covariance should be applied.

3.2.3.2 2D KLT:

The Karhunen-Love transform can be easily extended towards 2D input matrices. In this case the a 4D hyper-matrix gives the correlation between the samples of the 2D

input matrix. The optimal transform hyper-matrix is then obtained from the eigenmatrices of the correlation hyper-matrix in the same manner.

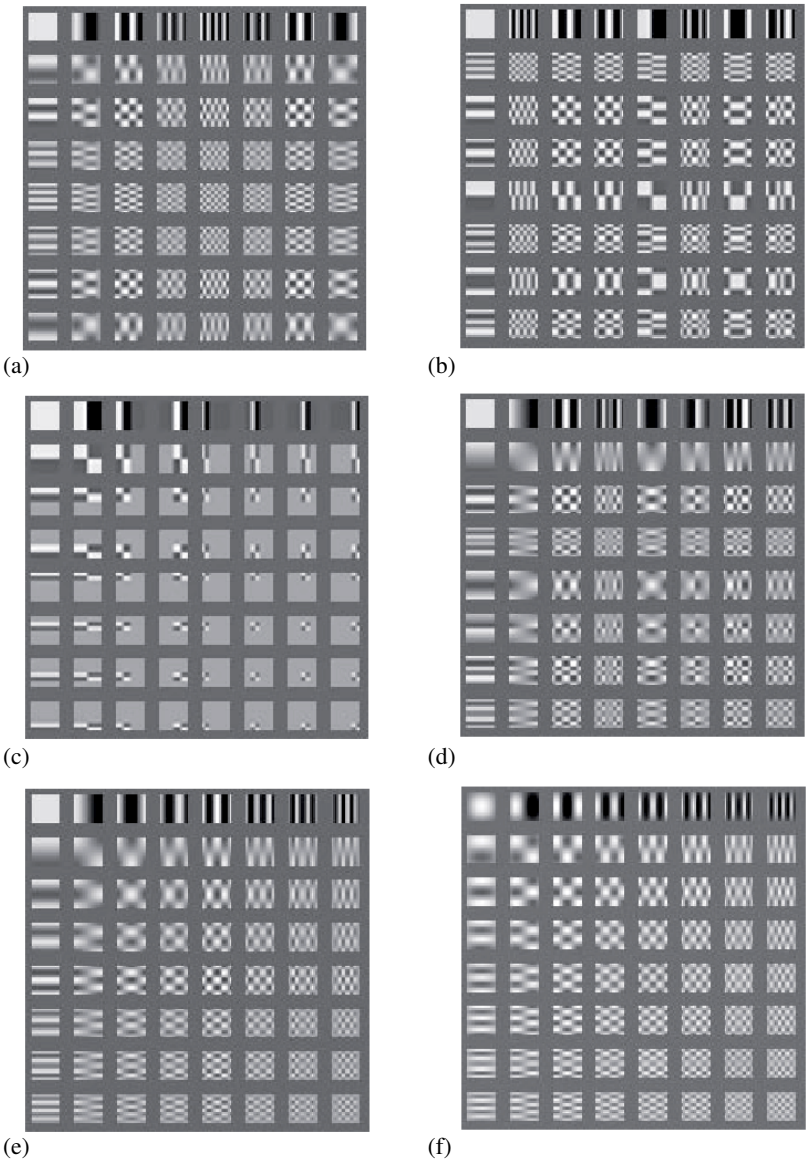


Figure 3.12: Basis images of frequently used orthogonal transforms: Hartley (a), Hadamarad (b), Haar (c), Slant (d), Discrete Cosine (e) and Discrete Sine (f) transforms.

The result of 2D KLT is depicted in 3.11. The figure shows the basis images of KLT decomposition, obtained as follows: The input image is divided into 8×8 sized blocks,

resulting obviously in 8×8 sized basis images and each image block is represented in the transform domain as 8×8 sized coefficient matrices. The covariance hyper-matrix is calculated of the image blocks. The eigenmatrices of this hyper-matrix gives the above basis images, from which each input block can be obtained by weighting the basis images with the coefficient matrix of the given block.

Similarly to optimal prediction, the KLT transform depends on the actual input registry, and in order to decode (inverse transform) the original image the transform matrix has to be transmitted along with the transformed signal. Therefore, in image compression KLT is rarely used, instead it is frequently applied in image analysis, e.g. in machine image recognition. Instead, in image compression sub-optimal, fixed-basis transforms are used. The most frequently used sub-optimal linear transforms are the non-harmonic Welsh-Hadamard, Hartley, Haar, Slant transforms and the harmonic Discrete Fourier, Sine and Discrete Cosine transforms (DCT). The basis functions of the above transforms are depicted in Figure 3.12.

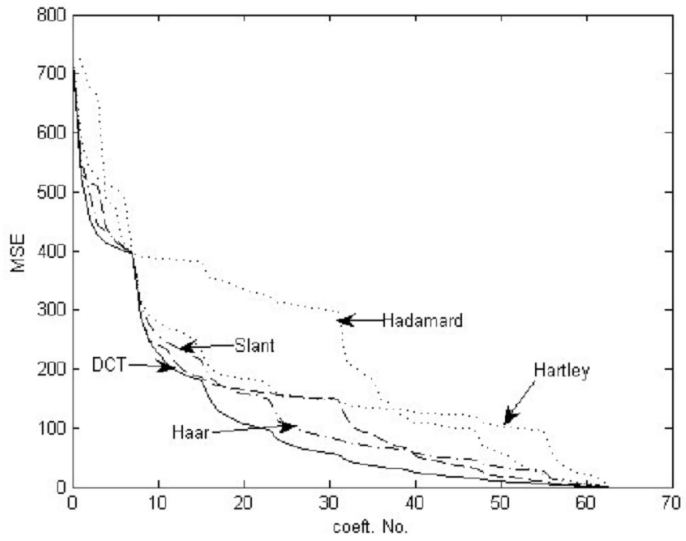


Figure 3.13: Energy of error with approximating the original image as the truncated series of basis functions.

3.2.4 Discrete Cosine Transform (DCT)

Investigating the basis images 3.11 (b) obtained from the KLT of an exemplary test image reveals that independently of the image content the resulting basis images resemble very closely to harmonic, sinusoidal functions. Hence, applying a linear transform with fixed basis images max decorrelate the input images sufficiently, giving a good approximation for the optimal KLT solution.

The performance of commonly used linear transforms is depicted in Figure 3.13. The figure illustrates the mean squared error between the reconstructed image and the original image in case black-based 8×8 sized linear transforms. The reconstructed image is obtained from a truncated inverse transform of the coefficient matrix: Instead of using all the coefficients—and the corresponding basis images—only a number of basis images contribute to the inverse transform. The figure depicts the introduced error in the reconstructed image as the function of the number of coefficients in the inverse transform. Clearly, most simple transforms, e.g. Hadamard and Hartely transforms introduce significant error even in the case of omitting a small number of coefficients in the in series expansion, meaning that these transform does not give a sufficiently compact representation. On the other hand, DCT achieves a relatively small error even in case of low order expansions. Therefore, in the field of image and video processing the most widely applied linear transform is the Discrete Cosine Transform (DCT), applying harmonic basis images.

The DCT have several important advantages making it feasible for practical applications:

- DCT is a real valued transform, resulting in real valued coefficients (opposed to e.g. the Discrete Fourier Transform, which generates twice the input data due to the complex transform)
- The DCT decorrelates natural images almost perfectly, being a nearly optimal transform
- Most of the signal energy is concentrated to low frequency coefficients
- DCT can be calculated effectively, with numerous fast implementations existing.

1D DCT

The definition of the Discrete Cosine Transform of the input vector $\mathbf{x}$ is given as

$$y(k) = \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} \alpha(n)\, x(n)\, \cos\left(\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right) \tag{3.57}$$

and the corresponding inverse transform as

$$x(n) = \sqrt{\frac{2}{N}} \sum_{k=0}^{N-1} \alpha(k)\, y(k)\, \cos\left(\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right), \tag{3.58}$$

with

$$\alpha(k) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } k = 0 \\ 1, & \text{if } k \neq 0. \end{cases} \tag{3.59}$$

From the inverse transform (3.58) the basis vectors can be directly read as

$$\mathbf{b}_k = \sqrt{\frac{2}{N}}\alpha(k)\, \cos\left(\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right), \qquad \text{for } n = 0, 1, ..., N-1. \tag{3.60}$$

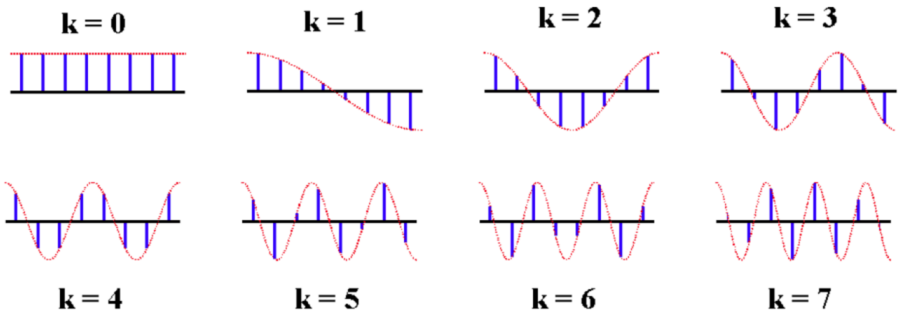


Figure 3.14: Basis vectors of 1D Discrete Cosine Transform.

The inverse transform, thus, describes and arbitrary input vector as the weighted sum of sampled cosine functions with various frequency. The spectral index k describes the frequency of the cosine function, and the spectral coefficient $y(k)$ describes the weight of the cosine $\mathbf{b}_k$ in the input signal. Since the DCT is orthogonal (the inner product of cosines with different frequency is zero) the spectral coefficients—i.e. the forward transform—are obtained by projecting the input signal to the k-th basis vector.

Similarly to the Fourier transform, since $k = 0$ describes a constant vector, the zeroth spectral coefficient $y(0)$ is referred to as the DC coefficients, and $y(1), y(2), ..., y(N-1)$ are the AC coefficients. The DC coefficient gives the mean value of the input vector, low frequency coefficients represent slowly changing input vector content, while high frequency AC components are dominated by fine resolution input content.

2D DCT

The two-dimensional Discrete Cosine transform is the extension of the DCT for input matrices as the consequent DCT transforms of each column and row.

Formally, the forward and inverse 2D DCT are defined as

$$Y(k,l) = \frac{2}{N}\alpha(k)\alpha(l)\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} X(m,n)\,\cos\left(\frac{(2m+1)k\pi}{2N}\right)\cos\left(\frac{(2n+1)l\pi}{2N}\right)$$

$$X(m,n) = \frac{2}{N}\sum_{l=0}^{N-1}\sum_{k=0}^{N-1} \alpha(k)\alpha(l)Y(k,l)\,\cos\left(\frac{(2m+1)k\pi}{2N}\right)\cos\left(\frac{(2n+1)l\pi}{2N}\right).$$

Again, the inverse transform can be rewritten in a shorter form of

$$X(m,n) = \sum_{k,l=0}^{N-1} B_{kl}(m,n)Y(k,l), \tag{3.61}$$

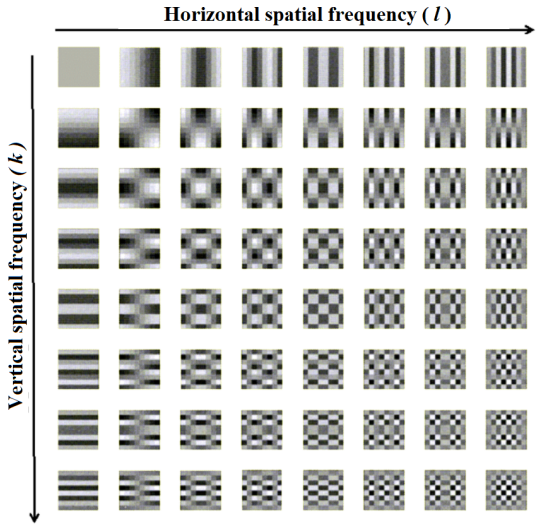


Figure 3.15: Basis images of 8×8 sized 2D Discrete Cosine Transform.

where $B_{kl}(m,n)$ are the basis matrices

$$\mathbf{B}_{kl} = B_{kl}(m,n) = \frac{2}{N}\alpha(k)\alpha(l),\ \cos\left(\frac{\pi}{N}\left(m+\frac{1}{2}\right)k\right)\cos\left(\frac{\pi}{N}\left(n+\frac{1}{2}\right)l\right). \tag{3.62}$$

Clearly, the basis matrices—i.e. the basis images—are obtained as the product of a horizontal and vertical cosine function with their frequencies given by l and k, respectively. Hence, l and k are referred to as the horizontal and vertical spatial frequencies. The basis images are illustrated for $N = 8$ in Figure 3.15, from which an arbitrary input block can be produced as their linear combination. Since the transform is orthogonal the transform coefficients are obtained as the projection of the input block to the basis images.

The coefficient matrix $Y(k,l)$—being the spectrum of the input image—defines the weight of the individual basis images in the input image block. The zero frequency coefficient $Y(0,0)$ gives the average value of the input matrix (e.g. if the input matrix contains the luminance samples of the input block, $Y(0,0)$ defines the average luminance), therefore, it is referred to as the DC coefficient. On the other hand, low frequency AC components correspond to slowly changing image content, while high frequency AC components represent fine details.

As a conclusion, generally speaking, the 2D DCT transforms the spatial domain representation of the $N \times N$ sized input block into the spatial frequency domain.

By definition the 2D Discrete Cosine Transform is a separable transform, since the basis images—and the forward and inverse transform matrices—can be written as the product of merely horizontal and vertical terms. Thus, in practice the 2D DCT is always evaluated by the consequent 1D DCT of each row and column of the input matrix.

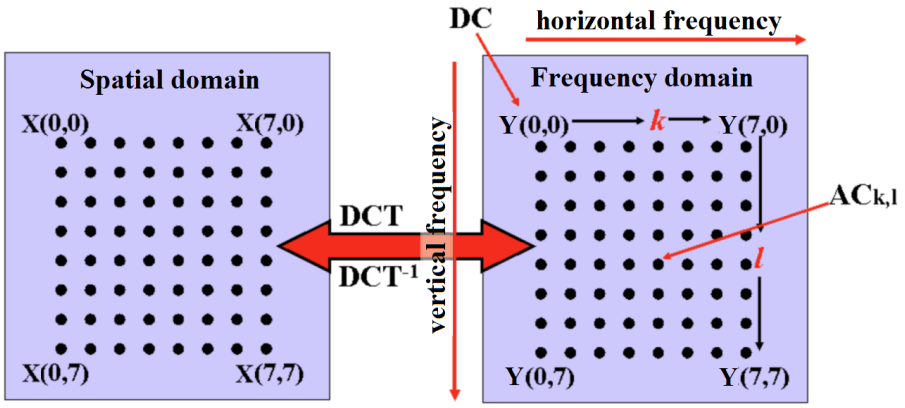


Figure 3.16: Illustration of 2D DCT transform, leading from the spatial domain to the spatial frequency domain.

By denoting the 1D DCT transform matrix by

$$\mathbf{A} = A(k,n) = \sqrt{\frac{2}{N}}\alpha(n)\,\cos\left(\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right), \qquad \alpha(k) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } k = 0 \\ 1, & \text{if } k \neq 0. \end{cases} \tag{3.63}$$

and the input matrix by $\mathbf{X} = X(m,n)$ the column-wise and row-wise 1D transforms of the input matrix are written as

$$\mathbf{Y}_c = \mathbf{A}\mathbf{X}, \qquad \mathbf{Y}_r = \left(\mathbf{A}\mathbf{X}^{\mathrm{T}}\right)^{\mathrm{T}} = \mathbf{X}\mathbf{A}^{\mathrm{T}} \tag{3.64}$$

respectively. The 2D DCT and IDCT can be calculated then as

$$\mathbf{Y} = \mathbf{A}\mathbf{X}\mathbf{A}^{\mathrm{T}}, \qquad \mathbf{X} = \mathbf{A}^{\mathrm{T}}\mathbf{Y}\mathbf{A}, \tag{3.65}$$

being in agreement with (3.50).

Example of 2D transform

As a simple example, the 2D DCT of the 8×8 sized block, illustrated in Figure 3.17 is investigated. In the current example the input matrix contains the black-and-white pixel intensities, i.e. only the luma information. The samples are stored on 8 bits with full swing representation, i.e. code 0 denotes black level, and code 255 denotes white.

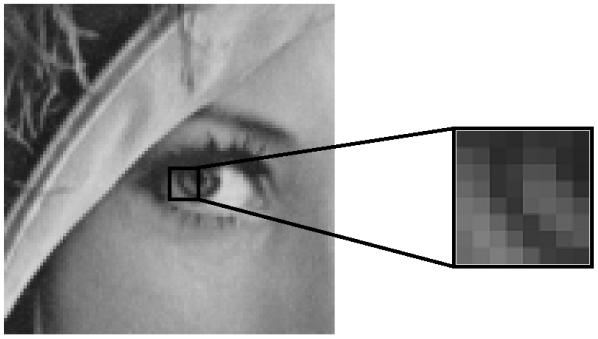


Figure 3.17: Simple 8×8 test block for the 2D Discrete Cosine Transform.

The pixel intensities are given in the current example by

$$\mathbf{X} = \begin{bmatrix} 48 & 47 & 55 & 47 & 51 & 48 & 46 & 44 \\ 53 & 53 & 47 & 43 & 54 & 49 & 50 & 40 \\ 77 & 67 & 43 & 47 & 62 & 60 & 45 & 39 \\ 90 & 81 & 50 & 60 & 80 & 79 & 48 & 38 \\ 103 & 90 & 50 & 57 & 94 & 93 & 76 & 48 \\ 115 & 107 & 71 & 52 & 92 & 98 & 90 & 72 \\ 117 & 121 & 99 & 67 & 58 & 84 & 98 & 86 \\ 108 & 127 & 115 & 88 & 59 & 60 & 79 & 87 \end{bmatrix}, \tag{3.66}$$

The 2D DCT coefficient matrix can be calculated by perform consequent vertical and horizontal 1D transforms, according to (3.65). In order to evaluate the 1D transform the required transform matrix is given by (3.63) with $N = 8$, with evaluating the sampled cosine functions at the $k = (0, 1, ..., N-1)$-th row and $n = (0, 1, ..., N-1)$-th column:

$$A(k,n) = \mathbf{A} = \begin{bmatrix} 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 \\ 0.5 & 0.42 & 0.28 & 0.1 & -0.1 & -0.28 & -0.2 & -0.5 \\ 0.46 & 0.19 & -0.19 & -0.46 & -0.46 & -0.19 & 0.19 & 0.46 \\ 0.41 & -0.1 & -0.5 & -0.28 & 0.28 & 0.5 & 0.1 & -0.42 \\ 0.35 & -0.35 & -0.35 & 0.35 & 0.35 & -0.35 & -0.35 & 0.35 \\ 0.28 & -0.5 & 0.1 & 0.42 & -0.42 & -0.1 & 0.5 & -0.28 \\ 0.19 & -0.46 & 0.46 & -0.19 & -0.19 & 0.46 & -0.46 & 0.19 \\ 0.1 & -0.28 & 0.42 & -0.5 & 0.5 & -0.42 & 0.28 & -0.1 \end{bmatrix}, \tag{3.67}$$

and the DCT coefficient matrix of the input is yielded as two consequent matrix multi-

plications

$$Y(k,l) = \mathbf{Y} = \mathbf{AXA}^{\mathrm{T}} = \begin{bmatrix} 563 & 61 & 33 & 48 & -19 & -17 & -10 & 3 \\ -136 & -29 & -41 & 13 & 20 & -0 & 5 & 3 \\ -6 & 2 & 14 & -68 & -12 & 15 & 6 & 3 \\ 15 & -22 & 3 & 23 & -7 & -3 & 1 & 10 \\ -1 & 9 & -18 & -6 & 8 & -0 & -1 & -4 \\ 5 & 5 & 6 & -3 & -4 & 3 & 7 & -5 \\ 2 & 2 & -1 & 3 & 6 & -7 & 1 & 2 \\ -1 & -1 & 5 & 2 & -2 & 5 & -2 & -1 \end{bmatrix}. \tag{3.68}$$

It is clearly highlighted that the coefficient matrix is dominated by the DC and low-frequency coefficients. The DC coefficient gives the average block intensity ($\times N$, i.e. the average luminance is $\frac{563}{8} \approx 70.3$). Furthermore, the input pixel block consists a dominant vertical line (on the edge of the eye's iris). Horizontally invariant, vertical textures can be described as the sum of horizontal cosines with the vertical spatial frequency being zero ($k = 0$). This fact is clearly reflected by the coefficient matrix with dominant horizontal frequency content: the energy is concentrated into small vertical frequency values ($k = 0, 1, 2$), while in the first matrix row high horizontal frequency components still have considerable energy. This is due to the fine structure/details of the sharp vertical line on the input image.

3.2.5 Quantization of the transform coefficients

In a transform based encoder the linear transform is followed by a lossy encoding step, realized by the requantization of the transform coefficients.

As an important advantage, transform coding allows the adaptation of the quantization process to the human visual system, by letting the coarseness of quantization vary as the function of spatial frequency: The human eye is good at seeing small differences in brightness over a relatively large area, but not so good at distinguishing the exact strength of a high frequency brightness variation. In other words, human vision is less sensitive to fine details which are mapped to the high spatial frequency components in the DCT domain, than to slow changes in the images, dominating the DC and low spatial frequencies. Therefore, applying a more rough quantization to high frequency coefficients than in the lower region does not degrade significantly the final image quality after inverse transform.

Frequency variant quantization can be most simply achieved by describing a frequency dependent constant factor, with which the DCT coefficient is divided before rounded to the nearest integer. This leads to the concept of the quantization matrix, defining the quantization resolution of each DCT coefficient. With denoting the elements of the **quantization matrix** by $W(k,l)$, the quantized coefficients are obtained as

$$Y^*(k,l) = \lfloor \frac{Y(k,l)}{W(k,l)} \rceil, \tag{3.69}$$

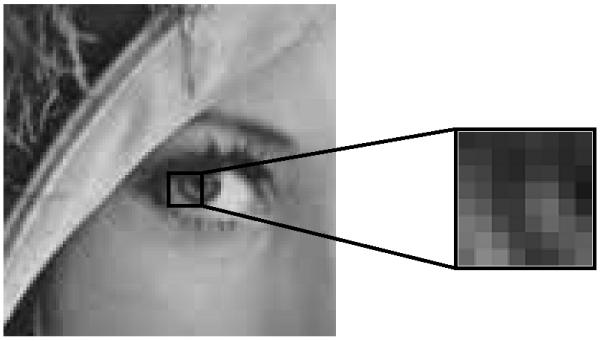


Figure 3.18: Reconstructed image and pixel block after quantization of the DCT coefficient matrix.

where $\lfloor \rceil$ denotes rounding to the nearest integer. Obviously, large entries in $W(k,l)$ result in coarse quantization on the given spatial frequency. A frequently used perceptually based quantization matrix is given as

$$W(k,l) = \begin{bmatrix} 8 & 16 & 19 & 22 & 26 & 27 & 29 & 34 \\ 16 & 16 & 22 & 24 & 27 & 29 & 34 & 37 \\ 19 & 22 & 26 & 27 & 29 & 34 & 34 & 38 \\ 22 & 22 & 26 & 27 & 29 & 34 & 37 & 40 \\ 22 & 26 & 27 & 29 & 32 & 35 & 40 & 48 \\ 26 & 27 & 29 & 32 & 35 & 40 & 48 & 58 \\ 26 & 27 & 29 & 34 & 38 & 46 & 56 & 69 \\ 27 & 29 & 35 & 38 & 46 & 56 & 69 & 83, \end{bmatrix} \tag{3.70}$$

with the larger coefficients ensuring coarser quantization with increasing spatial frequency.

As the continuation of the previous example, by quantizing the elements of matrix (3.68) with the above quantization matrix the quantized coefficient matrix read as

$$Y^*(k,l) = \frac{Y(k,l)}{W(k,l)} = \begin{bmatrix} 70 & 4 & 2 & 2 & -1 & -1 & 0 & 0 \\ -9 & -2 & -2 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -3 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0. \end{bmatrix} \tag{3.71}$$

The coefficient matrix reflects that due to coarse quantization high frequency coefficients are tending to become zero after quantization. Therefore, the transform based

perceptual quantization can be considered as a type of lossy spatial low-pass filtering of the input image. Obviously, increasing all the quantization matrix values would result in larger number of zero entries in the requantized coefficient matrix, requiring less bits for storage. Hence, the total bit rate can be controlled by multiplying the quantization matrix with an appropriately chosen constant **scale factor** at the cost of degrading image quality. This is the basic idea of bit rate controlling in JPEG and MPEG compression standards.

The effect of requantization can be examined in Figure 3.18 via the previous example. The image is obtained by decoding after requantization, i.e. by multiplication by the quantization matrix and performing the inverse DCT ($\mathbf{A}^{\mathrm{T}}\left(\mathbf{W}\cdot\mathbf{Y}^{*}\right)\mathbf{A}$). It is verified that although most of the high frequency content is eliminated, perceptual quantization results in minor degradation of the image quality. Two main artifacts are visible:

- The quantization of low frequency coefficients results in visible discontinuities around the border of the transform blocks, leading to the **blockiness** of the output image.
- As a well-known fact, low-pass filtering with sharp transition above the cut-frequency results in under- and overshoot (Gibbs oscillation) around a jump discontinuity in the input signal. Since the above matrix quantization approach can be interpreted as spatial low-pass filtering, therefore, **ringing artifacts** (over- and undershoots) appear near to edges in the input image.

3.3 The JPEG encoder

As a complex example for transform encoding the JPEG image encoding standard is investigated in details.

JPEG, standing for Joint Photographic Experts Group, was designed for the compression of full-color or gray-scale continuous-tone still images of natural, real-world scenes, like photographs or naturalistic artworks—hence its performance degrades for non-continuous tone images, like simple cartoons or line drawings—. The Joint Photographic Experts Group examined several transform coding techniques and eventually selected the Discrete Cosine Transform (DCT), as it was by far the most efficient practical compression technique. The JPEG standard was published in 1992, specifying the codec, which defines how an image is compressed into a stream of bytes and decompressed back into an image, but not the file format used to contain that stream. The Exif and JFIF standards define the commonly used file formats for interchange of JPEG-compressed images.

The JPEG standard allows the compression of images with the maximal size of 65535×65535 pixels, with the pixels represented either their RGB coordinates in the $Y'C_{\mathrm{B}}C_{\mathrm{R}}$space. The maximum number of components is limited to 255. The RGB color space is not defined by the standard, but arbitrary device dependent color space may be used, with the color space information embedded into the JPEG file. Commonly used color profiles include sRGB and Adobe RGB.

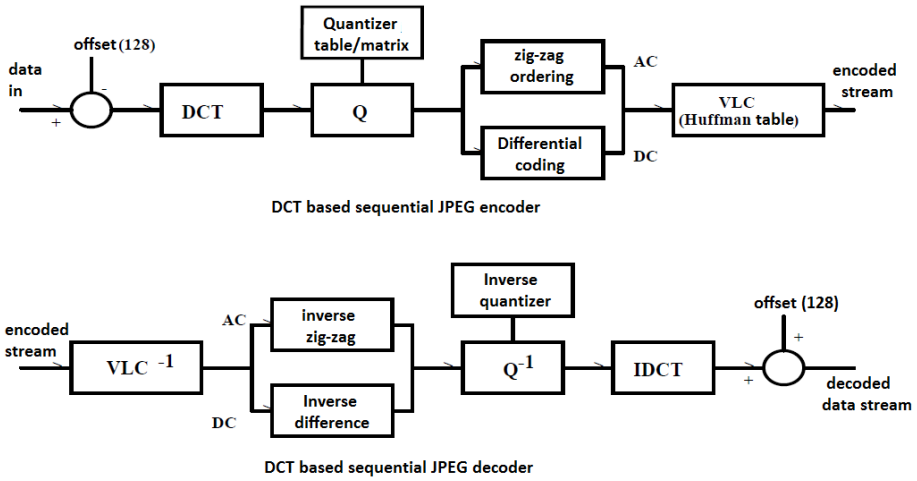


Figure 3.19: Block scheme of the JPEG encoder and decoder for a single image block.

The steps of JPEG encoding are discussed in the following.

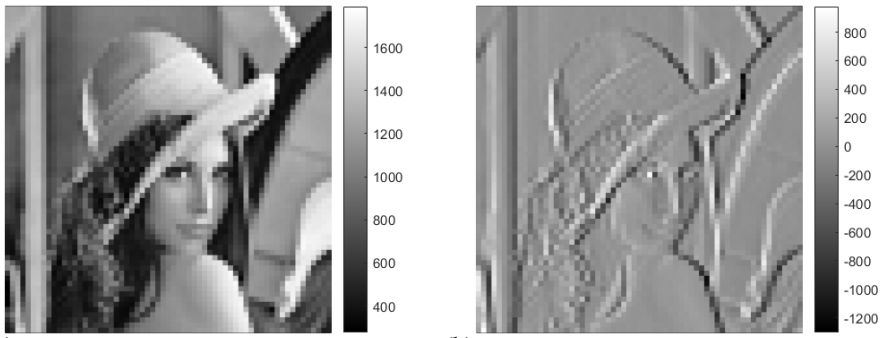
Chroma subsampling and block splitting

Although direct RGB processing is also allowed, by default JPEG encoding represents RGB pixels in the $Y'C_BC_R$ space, in order to achieve significant compression by chroma subsampling. The components of representation are referred to as **channels** in the JPEG standard. The encoder applies the chroma subsampling of 4:2:0 by default, reducing the spatial resolution of chroma components to half of the luma resolution both horizontally and vertically. Besides 4:2:0 scheme, 4:4:4 and 4:2:2 subsampling are also allowed by the standard. The actual anti-aliasing filtering strategy of the JPEG encoder was discussed in chapter **??**.

After subsampling, each component are split into 8×8 blocks, and further processing is performed block-wise, from left-to-right, starting with the upper-left corner. By applying the chroma subsampling scheme 4:2:0 an 8×8 sized chroma block corresponds to 16×16 luma samples, which are encoded into the output stream together, hence, the **Minimum Coded Unit (MCU)** for JPEG is 16×16. In video compression MCUs will be called macroblocks.

If the data for a channel does not represent an integer number of blocks then the encoder must fill the remaining area of the incomplete blocks with some form of dummy data. Filling the edges with a fixed color (for example, black) can create ringing artifacts along the visible part of the border due to the reasons discussed above. Therefore, repeating the edge pixels is a common technique that reduces such artifacts.

From that point the processing scheme of a single block is depicted in Figure 3.19, discussed as follows.



(a) (b)

Figure 3.20: The DC coefficient of all 8×8 sized blocks in an input image (a) and the horizontal difference between DC components of adjacent blocks (b).

3.3.1 DCT and quantization

First, the values of the input block are shifted from the positive range to one centered on zero, in order to reduce the dynamic range requirements in the DCT processing stage that follows. This is achieved by subtracting the mid-point of the representation from each input entries, being the value of 128 in case of 8 bits representation.

The 8×8 sized blocks are DCT transformed and requantized by applying an either default or user-defined quantization matrices, which can be uniquely defined for each input channel. A typical quantizer matrix was presented in (3.70). The actual image quality can be adjusted by multiplying the quantizer matrix with an appropriately chosen constant value, the **scale factor**: by choosing a large scale factor after division more coefficients tend to zero, resulting in a lower output data rate and higher compression ratio.

A typical, exemplary quantized coefficient matrix was shown in (3.71), with its DC component being the average chroma or luma value of the input block, and the AC components describing the horizontal and vertical variations along the block. These DC and AC values are encoded in the following separately.

3.3.2 Encoding the DC coefficients

For natural images both the average luminance and color information changes slowly inside the image. This is verified by Figure 3.20 (a), depicting merely the DC coefficient, i.e. the average of each input block, which is simply the original image with its resolution reduced by a factor of 8 both vertically and horizontally. In order to reduce spatial redundancy, the JPEG encoder applies linear prediction to the DC coefficients: each DC coefficient is encoded differentially, storing only the difference from the previous block's DC value. Hence, JPEG encoder is a hybrid encoder, applying both transform coding and predictive coding.

The difference between the DC component of adjacent blocks is depicted in Figure 3.20, verifying that most differential DC components are nearly zero valued, with the

differential representation allowing efficient compression.

Note that in order to avoid the accumulation of prediction error in the decoder side, linear prediction is performed in a feedback prediction loop: for each block the difference is calculated from the **quantized** DC component of the previous block.

3.3.3 Encoding the AC coefficients

Observing the quantized AC coefficients in (3.71) reflects that for natural images most AC coefficients are likely to become zero after requantization. Obviously, significant compression can be ensured by storing only the values of the non-zero coefficients along with their positions in the DCT matrix. This is achieved by the **zig-zag ordering** of the matrix entries followed by the **run-length coding** of the resulting vector.

Zig-zag ordering means the collection of the matrix entries with increasing spatial frequency order. The zig-zag order of matrix entries is depicted in Figure 3.21. With omitting the DC coefficients—since it is differentially encoded as discussed in

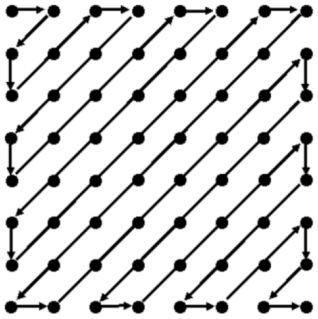


Figure 3.21: Zig-zag ordering of the transform matrix into a N^2 sized zig-zag vector.

the foregoing—the zig-zag ordered vector of the example of (3.71) reads as

$$(4, -9, 0, -2, 2, 2, -2, 0, 1, 0, -1, 1, 1, -1, -1, 1, -3, 0, 0, 0, 0, 0, -1, 1, 0, 0...) \tag{3.72}$$

Run-length coding of the above vector means storing only the non-zero vector values, along with the number of the preceding zeros: From the zig-zag vector **Run-level** pairs are constructed in the form of $[\text{No. of zeros}, \text{Next non-zero coefficient}]$. The last non-zero coefficient is furthermore signed with an End-of-block (EOB) flag. The run-

level representation of (3.72) reads as

$$
\begin{array}{l}
[0, 4] \\
[0, -9] \\
[1, -2] \\
[0, 2] \\
[0, 2] \\
[0, -2] \\
[0, 1] \\
\ldots \\
\text{EOB}
\end{array}
$$

From the above representation the decoder is able to reconstruct the original coefficient matrix.

Finally, both the differential DC coefficients and the run-level AC components are entropy coded by applying mixed **Huffman coding** and **Variable Length Integer** representation.

3.3.4 Entropy coding of DC and AC coefficients

Huffman code is a variable length, optimal prefix code, minimizing the average code length statistically. Huffman coding assigns codes to a set of source symbols based on the a-priori probability of the generation of each symbol, with shorter code assigned to more frequent/probable source symbols. Being a prefix code—meaning that no whole code word is the prefix of any other code—it is inherently ensured that the stream of output codes can be uniquely decoded.

The generation of Huffmann code is illustrated in Figure 3.22. In the example a source generates 4 types of symbols (a_1, a_2, a_3, a_4) with a-priory known probability. The symbol probabilities are given by $P_{a_1} = 0.4, P_{a_2} = 0.35, P_{a_3} = 0.2, P_{a_4} = 0.05$. The Huffman codes are assigned as follows, with the steps illustrated in Figure 3.22:

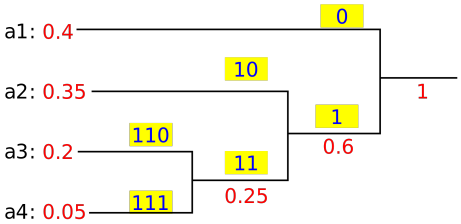


Figure 3.22: Construction of binary tree for Huffman coding.

- A binary tree is built starting as many leaves as symbols there are.
- Symbols with the least probabilities are combined into a single leaf with their probabilities summed.
- The last step is repeated until a full tree is built.

Size	Amplitude	VLI
1	-1, 1	0, 1
2	-3, -2, 2, 3	00, 01, 10, 11
3	-7...-4, 4...7	000...011, 100...111
4	-15...-8 , 8...15	0000..0111, 1000..1111
5	-31...-16 , 16...31	.....
6	-63...-32 , 32...63	.....
7	-127...-64 , 64...127	.....
8	-255...-128 , 128...255	.....
9	-511...-256 , 256...511	.....
10	-1023..511, 511..1023	.....
11	-2047...-1024, 1024...2047	00000000000...01111111111 10000000000...11111111111

Figure 3.23: The Variable Length Integer representation of DC and AC coefficients

- Starting backward from the root assign a bit symbol to each leaf, so that on a given branch the symbol will be the prefix of the next leaf.

As a result Huffman codes are assigned to each symbols with the lengths being inversely proportional with the symbol probability. The resulting codes are summarized in Table 3.2.

Table 3.2: Probabilities and the resulting Huffman codes.

Symbol	Probability	Huffman code
a_1	0.4	0
a_2	0.35	10
a_3	0.2	110
a_4	0.05	111

The JPEG encoder applies Huffman codes indirectly for encoding both the differential DC and the run-level AC components: The actual differential DC coefficient and the amplitude of the non-zero AC components are encoded with **variable length integer (VLI)** representation. VLI is a simple binary representation of the given value with omitting leading zeros, hence, the number of digits of the binary number varies. The VLI representation of the DC and AC amplitudes are summarized in Figure 3.23.

Since VLI is not a prefix codes, therefore, the JPEG encoder utilizes Huffman codes that denote the length of the VLI numbers to ensure decodability. Both the DC and AC coefficients are encoded using two symbols as presented in the following table

- In case of the DC coefficient, symbol 1 is a Huffman code, denoting the length of the following amplitude coefficient in bits, while Symbol 2 is the DC coefficient

	Symbol 1	Symbol 2
Differential DC	(SIZE)	(AMPLITUDE)
Run-level AC	(RUNLENGTH, SIZE)	(AMPLITUDE)

itself in VLI representation. As an example, if the differential DC coefficient is 3, then Symbol 1 is the Huffman code for SIZE = 2, and Symbol 2 is AMPLITUDE = 11 (being the binary representation of 3).

- Similarly, the run-level coded AC coefficients are stored by two symbols, with Symbol 1 being a Huffman code and Symbol 2 being a VLI code. Symbol 1 denotes the run length, along with the size of Symbol 2, with Symbol 2 encoding the actual level value. As an example, if the Run-level pair is (5,-1), being present in the previous example, then Symbol 1 is (RUNLENGTH = 5, SIZE= 1) and Symbol 2 is (AMPLITUDE= 0).

Obviously, since the differential DC coefficients are usually small values, therefore, shorter Huffman codes are assigned to small SIZE symbols.

The JPEG encoder allows the use of either default or user-defined Huffman tables, with the latter maximized to 2 tables per channel per AC or DC component. Along with the user defined quantization matrices, the Huffman tables are transmitted in the JPEG bitstream header. Both the use of non-default quantization matrices and Huffman tables slightly increases the encoders computational need, otherwise the encoder and decoder are symmetric in computational complexity.

3.3.5 Sequential and progressive encoding

So far the processing scheme of the individual blocks has been discussed. The order of the DC and AC components in the output bitstream depends on the encoding mode. Two main encoding modes are supported by the JPEG standard:

- **Sequential mode** encodes coefficients of a single block at a time, and the entire image is encoded and decoded (and displayed) block-wise.

- **Progressive mode** encodes and transmits the entire image with increasing resolution, being useful when the main goal is to display a low quality image as soon as possible and successively improve the image quality (e.g. in web applications). Two approaches exists to successively improve the image quality

  - **Spectral selection** appends first the DC component and first few AC coefficients of all image blocks to the output stream, then gradually transmits the remaining AC coefficients.

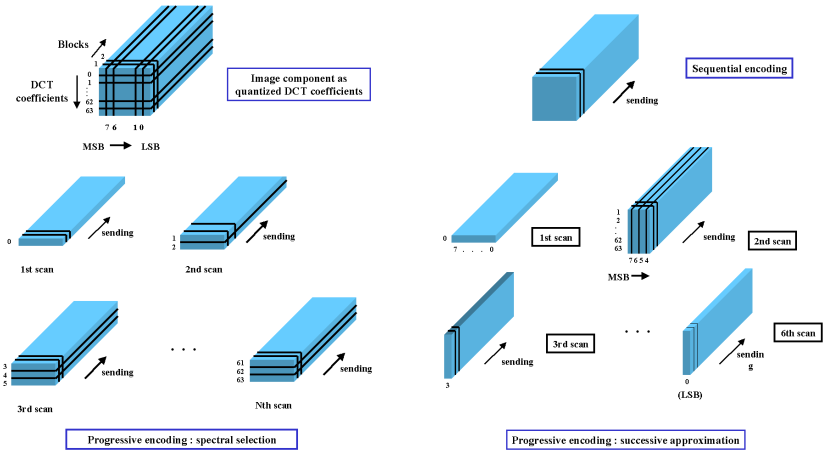


Figure 3.24: The Variable Length Integer representation of DC and AC coefficients

– **Successive approximation** first transmits the bits with increasing significancy of all coefficient of all input blocks, starting with the most significant bit.

The principle of progressive encoding is illustrated in Figure 3.24.

3.3.6 Artifacts of JPEG encoding

Overall, JPEG compression consists of tow lossy processing step: the optional chroma subsampling and the requantization of the DCT coefficients. In the previous chapters it was verified that the subsampling of the color information results in quasi-unnoticeable image quality degradation. The artifacts introduced by requantization were discussed in details in the previous section: blocking due to the quantization of the low-frequency coefficients and ringing due to the rough quantization of the high-frequency components. The amount of artifacts introduced and the global image quality depends on the quantization matrix and the quantizer scale factor, both set for the entire image in the output bitstream header. Therefore, JPEG allows only global image quality and bitrate control. This is opposed to MPEG standards that allow local quality control by adjusting the quantizer scale block from block.

A simple example for the artifacts, introduced by JPEG encoding is presented in Figure 3.25, with clearly highlighting the effects of blocking and ringing. The image was compressed with the quantization matrix (3.70) multiplied by the scale factor of 2 in Figure 3.25 (b) and of 4 in Figure 3.25 (c).

(a) (b) (c)

Figure 3.25: Result of JPEG encoding with the original image (a), with a scale factor of 2 (b) and of 4 (c).

End-of-Chapter Questions

- What

Chapter 4

MPEG video encoding

Relying on the theoretical basics discussed in the foregoing, the present chapter introduces MPEG compression, being the most frequently applied lossy video and audio compression method.

MPEG is the abbreviation for the Motion Picture Expert Group, which was established at 1988 in order to codify the first joint audio and video compression standard, based on the success of the preceding H.261 video encoder of the ITU (International Telecommunication Union). Development of the **MPEG-1** standard began in May 1988. Fourteen video and audio codec proposals were submitted by individual companies and institutions for evaluation, designed to compress VHS-quality raw digital video (SIF quality) and CD audio down to $1.5\ \mathrm{Mbit/s}$. After a more than 4 years progress the MPEG-1 standard was approved in 1992. Although offering a feasible video and audio quality, MPEG-1 achieves efficient compression only for a set of parameters, with low resolution progressive video and stereo audio tracks.

Therefore, in July 1990, before the first draft of the MPEG-1 standard had even been written, work began on a second standard, MPEG-2, intended to extend MPEG-1 technology to provide full broadcast-quality video (as per ITU-601 standard) at high bitrates ($3\tilde{}15\ \mathrm{Mbit/s}$) and support for interlaced video. The **MPEG-2** standard was eventually published in 1994, becoming the default compression for digital television and DVD contents. Due in part to the similarity between the two codecs, the MPEG-2

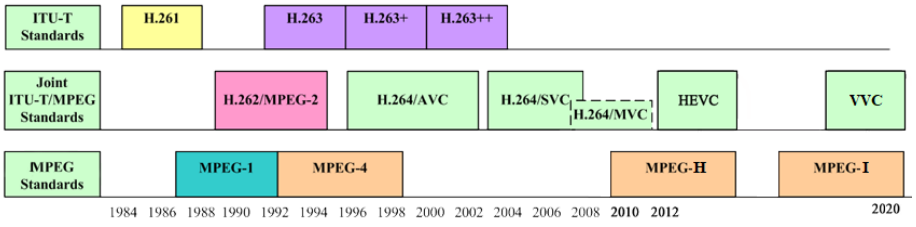


Figure 4.1: Evolution of the MPEG standards.

standard includes full backwards compatibility with MPEG-1 video, so any MPEG-2 decoder can play MPEG-1 videos.

The predecessor of MPEG-2, i.e. the MPEG-4 standard—introduced in 1998—was originally aimed primarily at low bit-rate video communications; however, its scope as a multimedia coding standard was later expanded. MPEG-4 provides a framework for more advanced compression algorithms potentially resulting in higher compression ratios compared to MPEG-2 at the cost of higher computational requirements. The MPEG-4 introduced AAC (Advanced Audio Coding) for audio encoding and the H.263 video codec, for which the most famous implementations are the DivX and Xvid video codecs. Later, in 2003 the standard was extended to include a new video encoding method, referred to as **Advanced Video Coding (AVC)**—or with its ITU name as the **H.264** codec—which is the most widespread encoding method used in the present days for digital video broadcasting and distribution, e.g. in blue ray formats.

With the emergence of ultra high resolution video formats and novel, object-based audio the need for more efficient encoding techniques rose. **MPEG-H** was introduced in 2013 intended for the high efficiency coding and media delivery in heterogeneous environments. The video encoder of the standard—developed by the MPEG and ITU together—is termed as **HEVC (High-Efficiency Video Coding)**, or with its ITU designation as **H.265**, is already a successor of H.264 in consumer 10 bits 4k and 8k UHD applications. As of 2019, HEVC is used by $43\ \%$ of video developers, and is the second most widely used video coding format after AVC. Besides its video encoder, MPEG-H 3D Audio standard can support up to 64 loudspeaker channels and 128 codec core channels, allowing higher order ambisonics (HOA) and object-based audio representation.

Although H.265 has only began to interchange the role of the H.264 encoder, the next generation of video encoding is about to be released: The **MPEG-I** standard is scheduled to be finalized in October 2020, offering a solution for the encoded representation of immersive media. In the future standard MPEG-I Part 3 will codify **Versatile Video Coding (VVC)** as a new generation video compression standard. The new algorithms should have $30 - 50\ \%$ better compression rate for the same perceptual quality, with support for lossless and subjectively lossless compression. It should support resolutions from 4k to 16k as well as 360° videos. VVC should support $Y'C_{\mathrm{B}}C_{\mathrm{R}}$4:4:4, 4:2:2 and 4:2:0 with 10 to 16 bits per component, wide color gamut and high dynamic range (HDR) of more than 16 stops, auxiliary channels (for depth, transparency, etc.), variable and fractional frame rates from 0 to 120 Hz, scalable video coding for temporal (frame rate), spatial (resolution), SNR, color gamut and dynamic range differences, stereo/multiview coding, panoramic formats, and still picture coding. Encoding complexity of several times (up to ten times) that of HEVC is expected, depending on the quality of the encoding algorithm (which is outside the scope of the standard). The decoding complexity is expected to be about twice that of HEVC.

Notably, all the MPEG standards very strictly define the bitstream, and decoder function, but does not define how encoding is to be performed. This means that MPEG coding efficiency can drastically vary depending on the encoder used, and generally means that newer encoders perform significantly better than their predecessors.

Figure 4.2: Adjacent image in video stream (a-b) and the difference of the luma matrices (c).

The MPEG standards consist of **parts**, describing the bit syntax of different aspects of the encoded of multimedia streams. As an example, the MPEG-1 standard includes the following parts:

1. Systems (storage and synchronization of video, audio, and other data together)
2. Video (compressed video content)
3. Audio (compressed audio content)
4. Conformance testing (testing the correctness of implementations of the standard)
5. Reference software (example software showing how to encode and decode according to the standar

In the following this chapter presents the video encoder defined by the MPEG-1 Part 2 standard. First the basic theory of motion compensation, used commonly in all MPEG video encoders is discussed.

4.1 Motion estimation and compensation

As illustrated in Figure 4.2, except for scene changes and rapid motion the video content does not change much from frame to frame, thus, the subsequent frames are strongly correlated. Linear prediction is an efficient tool to reduce this temporal redundancy: by encoding only the difference from the previous frames significant compression may be achieved. Obviously, moving objects in adjacent frames would severely degrade the performance of simple linear prediction. Instead, motion should be taken into consideration, and for a given object in the actual frame the basis of prediction should be those part of the previous frame where the object under consideration was previously located. Finding the corresponding, similar parts in consequent video frames is the basic goal of **motion estimation**. Linear prediction by applying the result of motion estimation is termed as **motion compensation**, or motion compensated prediction.

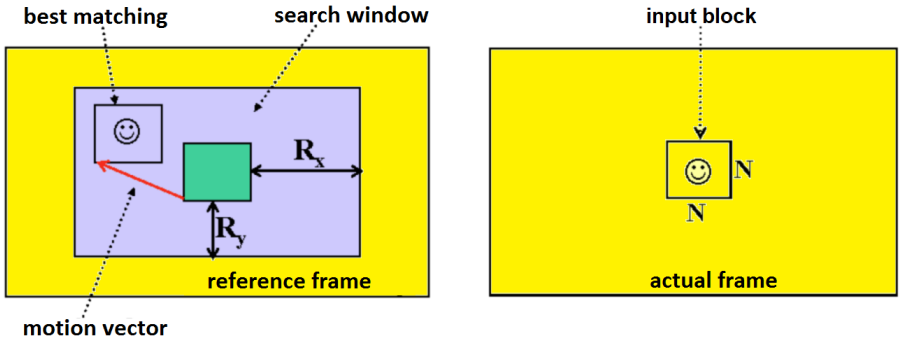


Figure 4.3: Adjacent image in video stream (a-b) and the difference of the luma matrices (c).

4.1.1 The goal of motion estimation and block matching

The motion of objects in consequent frames are described by **motion vectors**, which has to be transmitted in the encoded video stream for the sake of decodability. The definition of the motion vectors is one of the most computationally complex task in the encoder side, affecting significantly the efficiency and the time consumption of the encoding process.

Obviously, calculating the apparent motion of each pixel would be both time consuming and would require unnecessarily high bitrate to transmit all the resulting motion vectors. Instead, MPEG encoding applies **block-based motion estimation**: The input image is segmented into $N \times N$ non-overlapping blocks, fitting well into the block-based transform coding scheme of the following parts of the encoder. The basic goal of motion estimation is to assign a reference block to each block in the actual frame, which will serve as the basis of prediction. The previous frame, containing the reference block is referred to as the **reference frame** for the actual input frame. The vector pointing from the actual coded block to the reference block—i.e. their position difference measured in pixels—is the motion vector of the actual input block. The geometry is illustrated in Figure 4.3.

The reference block is defined as the block in the reference frame, resembling the most to the actual input block. The resemblance is formulated mathematically by minimizing a cost function measuring the difference between the input and the reference blocks. Most often the cost function is the **Mean Squared Difference (MSD)**, or the (**Mean Absolute Difference (MAD)**) between the pixel blocks, reading as

$$\begin{aligned} MSD(u,v) &= \frac{1}{N^2}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\left(B_{\text{act}}(m,n) - B_{\text{ref}}(m-u,n-v)\right)^2, \\ MAD(u,v) &= \frac{1}{N^2}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\left|B_{\text{act}}(m,n) - B_{\text{ref}}(m-u,n-v)\right|, \end{aligned} \tag{4.1}$$

with $B_{\text{act}}(m,n)$ denoting the actual input block and $B_{\text{ref}}(m,n)$ denoting blocks in the reference frame, with the their position difference denoted by (u,v). The most similar reference block, which is **matching** the actual block is then found at those translation value at which the above cost function is minimal. In this position (u,v) denotes the **motion vector** of the given block, defined mathematically as

$$\mathbf{v}_m = (u_m, v_m) = \arg\min_{\mathbf{u},\mathbf{v}} MSD(u,v). \tag{4.2}$$

Obviously, scanning the entire reference frame for the matching reference block is computationally infeasible. Instead, the matching reference block is searched inside a **search window** in the proximity of the actual input block position, denoted by R_x and R_y in Figure 4.3.

Once the best fitting block is found in the reference image, the resulting reference block will be the basis of prediction during encoding.

The performance of the motion estimation algorithms can be quantified as the ratio of the input block energy and the energy of the differential block, described by the **Peak-Signal-to-Noise-Ratio (PSNR)**

$$\text{PSNR} = 10\log_{10}\left(\frac{\max\limits_{m,n} B_{\text{act}}(m,n)^2}{MSD}\right), \tag{4.3}$$

where $\max\limits_{m,n} B_{\text{act}}(m,n)$ is the maximal pixel intensity in the input block. As the efficiency of motion estimation increases with decreasing MSD, higher PSNR indicates more efficient estimation result.

PSNR is most commonly used to measure the quality of reconstruction of lossy compression codecs. The signal in this case is the original data, and the noise is the error introduced by compression. When comparing compression codecs, PSNR is an approximation to human perception of reconstruction quality[1].

4.1.2 Block matching algorithms

Finding the best fitting reference block for the input blocks of the actual image by minimizing the prescribed cost function is termed as **block matching**, being a critical steps in the MPEG encoding scheme: Block matching has to be performed for each single block in each single input frame, often multiple times, if more than one reference frames are used. Therefore, sufficiently fast block matching algorithm has to be implemented which also efficiently finds the best matching block in the search window, i.e. ensures sufficiently high PSNR. If the difference between the coded and the reference block is large, the bitrate of the output MEPG stream increases.

The speed of motion prediction is usually quantified by the steps, required to find the minimum of the cost function inside the search window, i.e. the number of block

[1] Typical values for the PSNR in lossy image and video compression are between 30 and 50 dB, provided the bit depth is 8 bits, where higher is better. For 16-bit data typical values for the PSNR are between 60 and 80 dB. Acceptable values for wireless transmission quality loss are considered to be about 20 dB to 25 dB.

positions in the search window at which the difference from the input block is evaluated.

In the following several block matching algorithms are discussed.

Exhaustive search

As a brute force approach the difference between the input block and each possible block inside the search window may be calculated. Mathematically this means the evaluation of $MAD(u, v)$ for each possible $u \in [-R_x, R_x]$, $v \in [-R_y, R_y]$ translation value. By assuming a typical square search window of $R_x = R_y = R = 7$ this requires

$$N = (2\,R + 1)^2 = 225 \tag{4.4}$$

steps to arrive at the minimum of the resulting error surface. This **exhaustive search**, or **full search** inherently ensures that the absolute minimum of the error surface is found, ensuring the highest possible PSNR value. However, the algorithm is computationally expensive for real-time applications.

Logarithmic search

As a faster alternative for the exhausting search, logarithmic approaches find the minimum position of the error surface by testing/evaluating the MSD values with large initial resolution and refine the resolution in the proximity of the result. The resolution of the search locations in the search window is termed as **step size**: logarithmic search starts with a relatively large initial step size and decreases it in every search step. Commonly used logarithmic search methods are the following:

Three Step Search:
Three Step Search (TSS) is one of the earliest fast block matching algorithms, originating from the early 1980s. The steps of TSS is illustrated in Figure 4.4 (a) for the typical value of the search window of $R = 7$ which results in the initial step size of $S = 4$:

- First the cost function is evaluated in 9 initial position, with zero translation and with $S = \pm 4$ pixels offset around $(0, 0)$. In the current example the minimum of the cost function is found at $(4, 0)$
- The center of the second search step is the minimum position of the previous step. The step sized is set to $S = 2$ and the cost function is evaluated for 8 positions around the new origin. In the current example the minimum of the cost function in the second step is found at $(6, 2)$.
- Again, the center of the third step is the minimum value, found in the second step. Around this new center the cost function is evaluated with the last step size set to $S = \pm 1$ pixels. The minimum value of the cost function is the final result of the TSS algorithm.

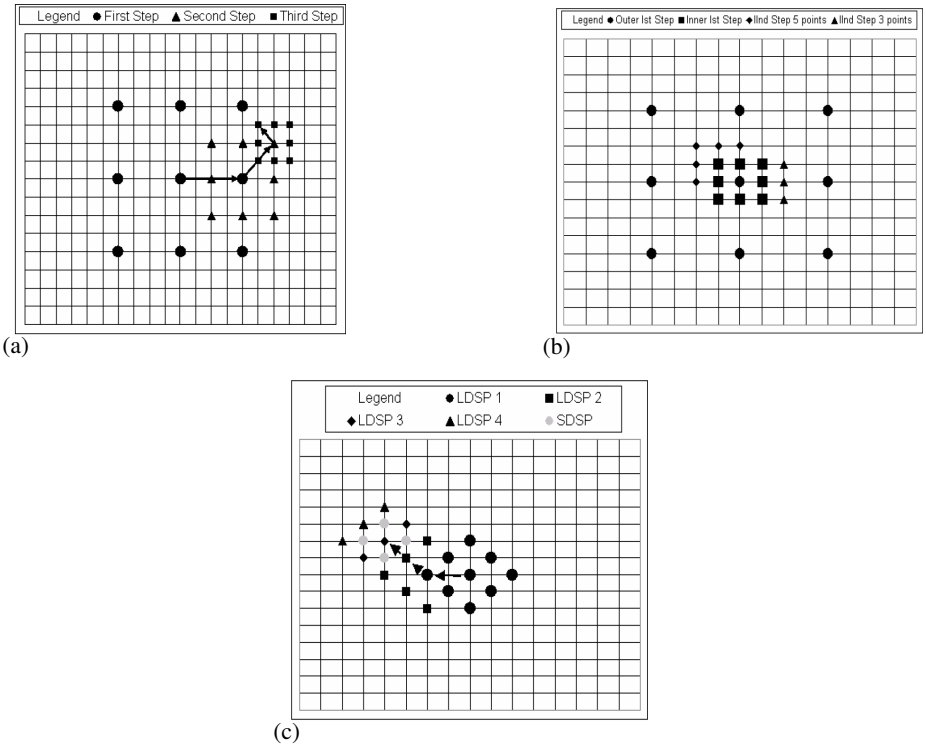


Figure 4.4: Search steps of the Three Step Search (a), New Three Step Search (b) and Diamond Search (c) block matching algorithms.

The TSS algorithm finds its result at the fixed step number of

$$N = 9 + 8 + 8 = 25 \tag{4.5}$$

steps, instead of the 225 steps of the exhaustive search. However, the algorithm only finds the global minimum of the error surface only if the error surface is unimodal (bowl shaped).

New Three Step Search:
New Three Step Search (NTSS) improves on TSS results by providing a center biased searching scheme and having provisions for half way stop to reduce computational cost. It was one of the first widely accepted fast algorithms and frequently used for implementing earlier standards like MPEG 1 and H.261.

The TSS uses a uniformly allocated checking pattern for motion detection and is prone to missing small motions. Therefore, NTSS adds 8 more search positions the first step with the step size of ± 1 pixel:

- If the lowest cost is at the origin then the search is stopped right here and the motion vector is set as $(0, 0)$.

- If the lowest cost is in one of the 8 positions around the origin, then this position is set as the new center and the cost function is evaluated in its the adjacent positions (which are either 3 or 5 more positions).
- If the lowest cost is found in one outer position, then the rest of the algorithm is the same as the TSS.

Depending on the actual scenario, the NTSS has the number of overall search position of 17 as the best-case to 33 as the worst-case scenario.

Diamond Search:
Diamond Search (DS) uses two different types of fixed patterns, one is Large Diamond Search Pattern (LDSP) and the other is Small Diamond Search Pattern (SDSP). These two patterns and the DS procedure are illustrated in Figure 4.4 (c).

Diamond Search uses the large diamond pattern as long as the minimum of the cost function is found in an outer position of the LDSP. Every step with the large pattern requires to check either 3 or 5 positions. As soon as the algorithm finds the best match in the central position the algorithm gets to the last step, in which a small search pattern is used around this final central position.

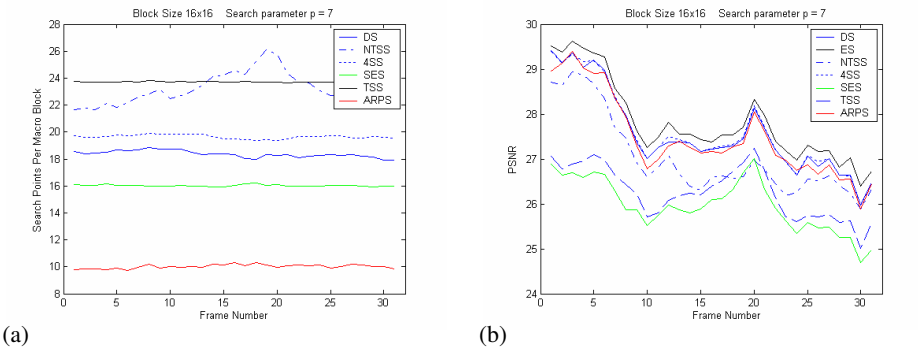
As the search pattern is neither too small nor too big and the fact that there is no limit to the number of steps, this algorithm can find global minimum very accurately. The end result should see a PSNR close to that of ES while computational expense should be significantly less.

Adaptive Rood Pattern Search:
Generally speaking, the motion in a frame is usually coherent, i.e. if the macroblocks around the current macroblock moved in a particular direction then there is a high probability that the current macroblock will also have a similar motion vector. Therefore, Adaptive Rood Pattern Search (ARPS) uses the motion vector of the macroblock to its immediate left to predict its own motion vector: i.e. the initial search position is given by the previous block's motion vector. Otherwise, the ARPS algorithm is identical with the Diamond Search process.

Comparison of block matching algorithms

The performance of the most widely used block matching algorithms is compared in Figure 4.5. Figure (a) depicts the total step number of the algorithms per frame, evaluated for 30 frames of an input video sequence. Clearly, all the algorithms need significantly less steps than the brute-force exhaustive search, requiring 225 steps to fing the global minimum of the MSD error surface. Adaptive search, by using the previous block's motion vector as the initial search position reduces the step number even further.

In the aspect of accuracy, obviously, exhaustive search ensures the highest PSNR value, serving as the reference solution. However, diamond search and its modification (ARPS) achieve PSNR values close to this reference, finding efficiently the real, global minimum of the error surface. Thus modern H.264 encoders implement frequently some modification of the discussed diamond search algorithm.



(a) (b)

Figure 4.5: Search points per block (a) and the mean resulting PSNR (b) of fast block matching algorithms.

4.1.3 Motion compensation in MPEG encoders

As discussed in the foregoing, the MPEG encoder applies linear temporal prediction with motion compensation. The motion estimation is always performed on the luma blocks of the image, and the resulting motion vectors are applied also for encoding the chroma blocks. In MPEG encoders the unit of motion compensation is termed as **macroblock (MB)**, with its size chosen to be fixed to 16×16 pixels in the MPEG-1 encoder.

Instead of encoding the macroblock data directly, predictively coded macroblocks are stored differentially to the reference macroblock, along with the motion vector, describing its position on the reference frame. The differential macroblocks are calculated according to

$$\epsilon(m,n) = B_{\mathrm{act}}(m,n) - \sum_{i=0}^{I} a_i \, B_j(m-u_i, n-v_i), \tag{4.6}$$

with B_j denoting the reference macroblock in the j-th reference frame and $\mathbf{v}_i = (u_i, v_i)$ being its motion vector. Note that the above expression is in correspondence with the linear prediction scheme, discussed in the previous chapter: In this case the prediction filter contains the reference macroblocks, and the actual input macroblock is predicted as their linear combination. Obviously, the more efficient the prediction is, the more noise-like is the differential macroblock. As discussed in the following the simple MPEG-1 and MPEG-2 encoders allow the maximum number of reference frames with fixed prediction weights, while latter codecs (H.264, H.265) usually use 4-5 reference frames with arbitrary weight factors.

As the simplest case the macroblocks are encoded differentially from a single reference macroblock, i.e. $a_i = 1$. The result of prediction in the example presented in Figure 4.2 is depicted in Figure 4.6. As a result of motion compensation the PSNR has increased significantly due to the reduced differential image energy. Figure 4.6 (b) depicts the motion vector space of the input frame: the motion vector of a given macro shows into the position of the individual reference blocks on the reference frame. Since

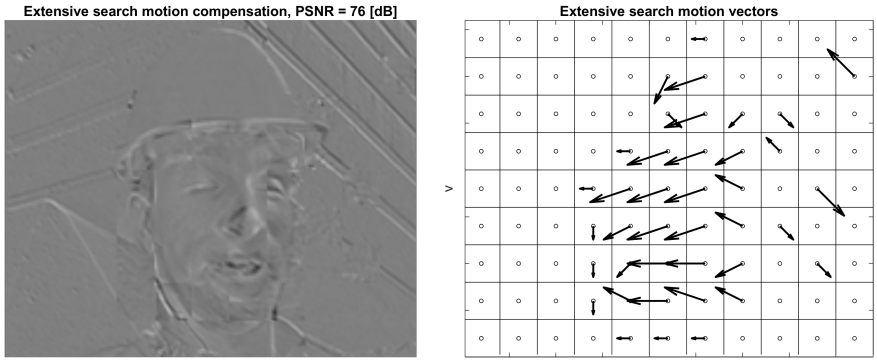


Figure 4.6: The differential image (a) and the motion vector space (b) with one-pixel precision and using exhaustive search.

adjacent block usually move into the same, or similar direction, most of the neighboring motion vectors are pointing in the same direction, as it was exploited for ARPS block matching. Therefore, MPEG codecs transmit the motion vectors differentially, with encoding only the difference of the actual motion vector and the motion vector of the previous macroblock.

So far block matching algorithms have been discussed that allow the estimation of motion vectors with the precision of one pixel. The accuracy of motion estimation, and hence the efficiency of compression can be increased by applying fractional pixel precision motion vector representation. This **sub-pixel accuracy** can be achieved by increasing the resolution of both the input block and the search window by interpolation before motion estimation and the coordinates of the resulting motion vectors are divided by this upsampling factor. Motion estimation is most often performed with doubled ($\times 2$) and quadrupled ($\times 4$) spatial resolution, resulting in half-pixel (often termed as **half-pel**) and **quarter-pixel** (or **quarter-pel/Q-pel**) motion vector precision. Obviously, the interpolation of all the input macroblocks before motion estimation significantly increases the computational cost of the MPEG encoder. As an example, common implementations of H.264/AVC uses a 6-tap filter for half-pixel interpolation and then simple linear interpolation to achieve quarter-pixel precision from the half-pixel data, while HEVC uses separable 7-tap or 8-tap filtering. The result of motion compensation with half-pel accuracy is depicted in Figure 4.7.

Besides the number of reference frames and precision of motion vector representation, also the spatial parameter choice of motion estimation is not straightforward: Both the search area and the macroblock size affects the efficiency of prediction:

- Applying large search area may find fast movements, but at higher computational cost, and the transmission of long motion vectors may require more bits for representation. On the other hand, small search window results in inaccurate motion estimation and, therefore, inefficient compression.

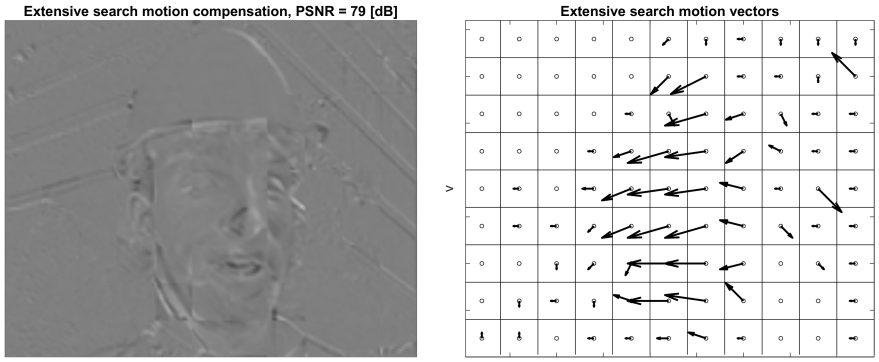


Figure 4.7: The differential image (a) and the motion vector space (b) with half-pixel precision and using exhaustive search.

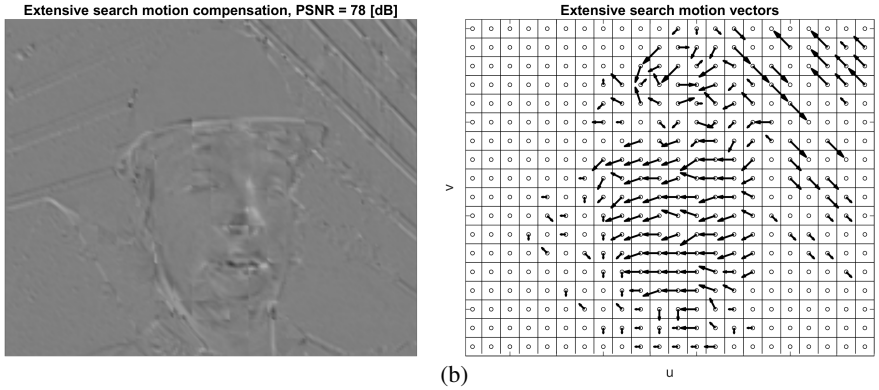


Figure 4.8: The differential image (a) and the motion vector space (b) with 8×8 sized macroblocks.

- Choosing a small macroblock size may significantly increase the accuracy of prediction, as it is presented in Figure 4.8. However, on one hand it results in a higher number of motion vectors. Also, in case of small macroblocks the result of motion estimation often does not reflect the true object motion, but simply a numerical resemblance. This leads to alternating motion vector space, for which differential encoding is not effective anymore.

As a compromise MPEG-1 encoder applies the macroblock size of 16×16 pixels with the search window usually set to ± 7 pixels. Newer encoders, e.g. H.264 and H.265 allow the division of the macroblock into sub-macroblocks, with the size of 4×4, 4×8, 8×8, or 8×16. The macroblock size has to be decided by the encoder based on which MB size ensure the most efficient compression.

4.2 The MPEG-1 video encoder

MPEG-1 video exploits perceptual compression methods to significantly reduce the data rate required by a video stream. It reduces or completely discards information in certain frequencies and areas of the picture that the human eye has limited ability to fully perceive. It also exploits temporal and spatial redundancy common in video to achieve better data compression. The basic principle of MPEG-1 Video is **hybrid coding**, a combination of block-wise motion-compensated prediction and DCT-based coding: The blocks of the input image can be encoded either directly, or predictively. In the direct case the MPEG encoder is basically the simple JPEG encoding of the individual frames. For predictively coded frames the residual blocks are encoded by DCT-based transform encoding.

First the structure of the MPEG video is discussed in details, highlighting the basic features of the MPEG encoding process.

4.2.1 Layers of MPEG-1 video stream

MPEG video is broken up into a hierarchy of layers to help with error handling, random search, editing, and synchronization, for example with an audio bitstream. The layers of the MPEG stream from the top level are the following

- **Sequence layer**: any self-contained bitstream, for example a coded movie or advertisement
- **Group of Pictures (GOP)**: sequence of frames, consisting at least one intra-coded frame. Unit of random access.
- **Picture layer**: The encoded data of one single frame
- **Slice layer**: continuous sequence of raster ordered macroblocks, most often on a row basis in typical video applications. This is the unit of re-synchronization, i.e. it is the lowest layer where decoder is able to revive in case of bit-error.
- **Macroblock layer**: The unit of motion compensation. In MPEG-1 it is fixed to 16×16 and 8×8 sized blocks of luma (Y') and chroma (C_b, C_r) components respectively.
- **Block layer**: The unit of DCT encoding. Similarly to JPEG encoding, it is fixed to 8×8 size block within a MB, resulting in 4 luma blocks and 1-1 chroma blocks inside one macroblock.

Each layer of the input stream is identified with a unique header, containing all the required information for decoding the given layer.

The sequence layer

The sequence layer of an MPEG video is the individual video stream, with the header containing the global resolution and bitrate control type of the video data. The typical MPEG-1 input format is the source input format (SIF), derived from ITU-601, the

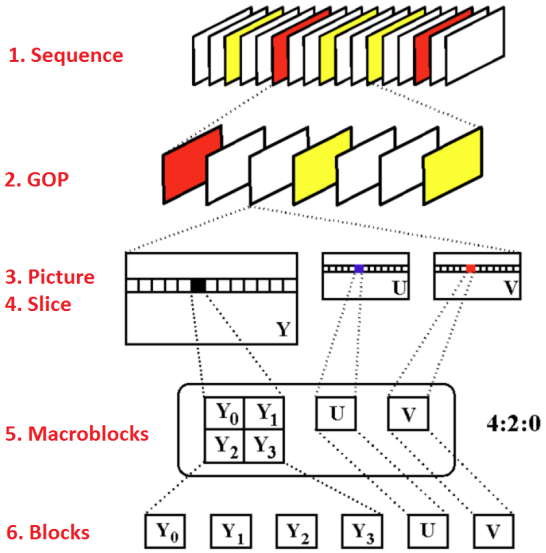


Figure 4.9: Layers of the MPEG video stream.

then-worldwide standard for digital TV studio, with the resolution of 720×480 and 720×576 for the NTSC and PAL systems respectively. Since it turned out to be difficult to compress an ITU-601 video to 1.5 Mbit/s with good video quality, in MPEG-1, typically the source video resolution is decimated to a quarter of the ITU-601 resolution by filtering and subsampling. The resultant format is called **source input format (SIF)**, which has a 360×240 resolution for NTSC and a resolution for 360×288 PAL. In order to fit with the 16×16 macroblock size this is truncated to the resolution of 352×240 and 352×288.

By default the MPEG encoder applies 8-bit $Y'C_{\mathrm{B}}C_{\mathrm{R}}$representation with the chroma subsampling scheme 4:2:0. Therefore, the first step of MPEG encoding is chroma subsampling with the anti-aliasing filter described in the previous chapter. As a result the 16×16 sized luma macroblocks correspond to 8×8 chroma blocks.

The GOP and the picture layer

MPEG encoding allows the use of three basic frame types:

- **I (intra) pictures:** I pictures (intracoded pictures) are coded independently with no reference to other pictures. With using the denotation of equation (4.6) the prediction coefficients are simply $a_i = 0, \forall i$.

  I pictures provide random access points in the compressed video data, since the I pictures can be decoded independently without referencing to other pictures. With I pictures, an MPEG bit stream is more editable. Also, error propagation due to transmission errors in previous pictures will be terminated by an I picture, since the I picture does not have a reference to the previous pictures. Since I pictures use only transform coding without motion compensated predictive coding, it provides only moderate compression, similar to JPEG encoding.

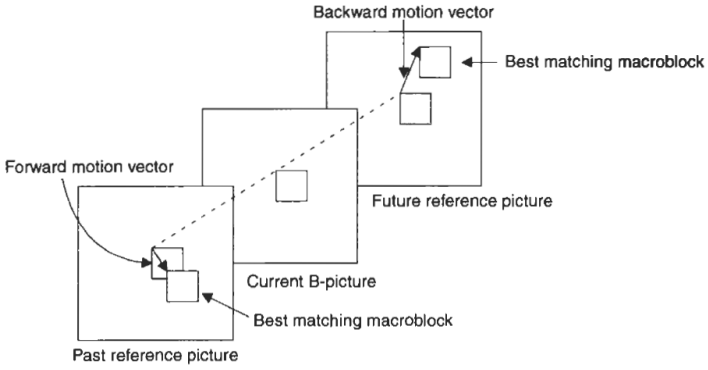


Figure 4.10: Linear prediction of B pictures.

- **P (predictive) pictures:** P pictures store only the difference in image from the frame (either an I-frame or P-frame) immediately preceding it. With using the denotation of equation (4.6) the prediction coefficients are given by $a_1 = 1$.

  P pictures provide more compression than the I pictures by virtue of motion compensated prediction. They also serve as references, or **anchor frames** for B pictures and future P pictures. Transmission errors in the I pictures and P pictures can propagate to the succeeding pictures, because the I pictures and P pictures are used to predict the succeeding pictures.

- **B (bidirectional-coded) pictures:** B pictures are similar to P-frames, except they can make predictions using both the previous and future I or P frames (i.e. two reference frames, or anchor frames). The basis of prediction are given by the average of the past and the future reference macroblocks, meaning that differential macroblocks are computed as

$$\epsilon(m,n) = B_{\text{act}}(m,n) - \frac{B_1(m-u_1, n-v_{-1}) + B_{-1}(m-u_{-1}, n-v_{-1})}{2}, \tag{4.7}$$

  i.e. the filter coefficients of (4.6) are $a_1 = a_{-1} = \frac{1}{2}$. The concept of predicting B pictures are illustrated in Figure 4.10.

  The use of two reference blocks for prediction basically means a longer prediction filter (with the length of 2), therefore, the residual block after prediction is more noise-like, less correlated. This is verified in Figure 4.11 (a), reflecting the increased PSNR value in the previous example. On the other hand, two motion vectors will be assigned to each macroblock which have to be transmitted along with the residual block, increasing the output stream's bit rate.

  To keep the structure simple and since there is no apparent advantage to use B pictures for predicting other B pictures, the B pictures are not used as reference pictures. Hence, B pictures do not propagate errors.

A group of pictures (GOP) is the sequence of several frames, consisting at least

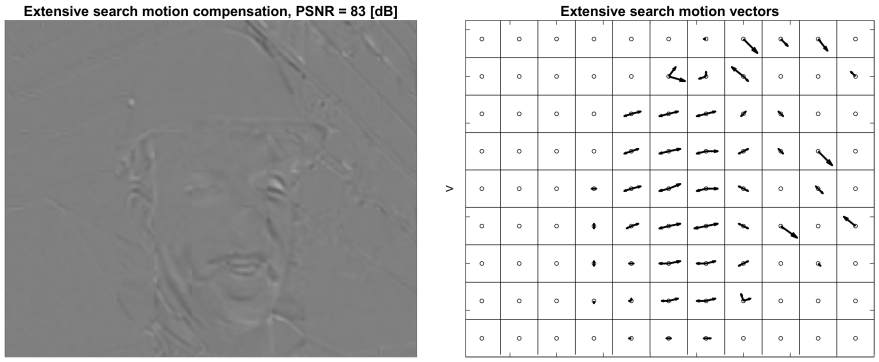


Figure 4.11: The differential image (a) and the motion vector space (b) with 16×16 sized macroblocks and half-pel motion estimation accuracy in case of B pictures.

one I picture. Since an I frame is required to begin the decoding of the following predictively coded frames, therefore, GOP is the unit of random access: The playback of a video stream always starts at the beginning of the actual group of pictures.

The order of the I, P and B pictures in a group of pictures is not specified by the standard, a sequence may consist of merely I, or I and P frames. The encoder is free to choose the optimal coding mode for a given input frame. The application of all frame types has both advantages and disadvantages

- Encoding an I frame requires on average $0.8 - 1.2\ \mathrm{bit/pixel}$. The application of many I pictures allows fast random access for editability, and for the start of decoding. Therefore, studio applications frequently use **I-only** GOP structure, containing exclusively intra coded frames. On the other hand the compression factor with using large number of I pictures may be low.

- Applying high number of P and B pictures increases the compression ratio: P pictures require about $0.3 - 0.5\ \mathrm{bits/pixel}$, while B pictures take $0.1 - 0.3\ \mathrm{bits/pixel}$ on average. The trade- off of having frequent B pictures is that it decreases the correlation between the previous I or P picture and the next reference P or I picture. It also causes coding and decoding delay and increases the encoder complexity

Generally speaking, if the aim is the editability then the application of large number of I frames is favorable, while if high compression factor has to be achieved at the cost of increased hardware cost and latency then the number of B frames should be increased.

As a good compromise, the typical group of pictures structure is given by

sg I B B P B B P B B P B B sg I B ...

termed as the **long GOP**. All the P frames are predicted from the first I pictures, while intermediate B frames are predicted from the closest past and future I or P frame.

Obviously, in order to encode/decode a B frame first its future reference has to be encoded/decoded. Therefore, the encoding, transmission and **decoding order** differs

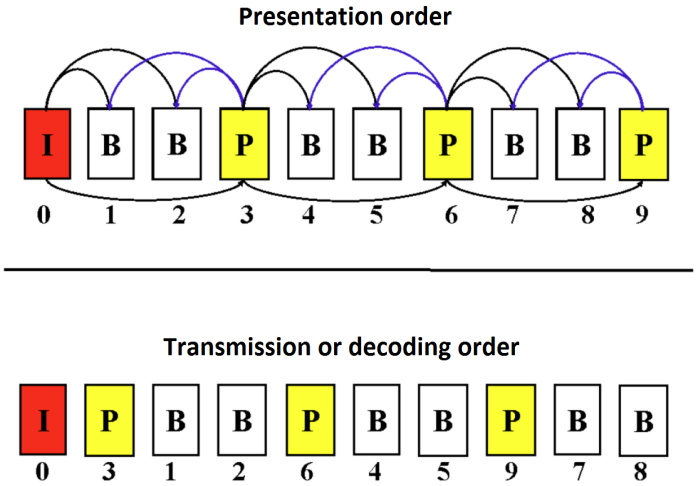


Figure 4.12: The presentation order and the corresponding decoding order in case of long GOP structure.

from the **presentation order**, at wich the decoded images are displayed. As simple example for the decoding and presentation order is shown in Figure 4.12.

The slice, macroblock an block layers

An MPEG picture consists of slices. A slice consists of a contiguous sequence of macroblocks in a raster scan order (from left to right and from top to bottom). In an MPEG coded bit stream, each slice starts with a slice header, which is a clear codeword (a clear codeword is a unique bit pattern that can be identified without decoding the variable-length codes in the bit stream). As a result of the clear-codeword slice header, slices are the lowest level of units that can be accessed in an MPEG coded bit stream without decoding the variable-length codes. If a bit stream contains a bit error, the error may cause error propagation because of the variable-length coding. The decoder can regain synchronization at the start of the next slice.

A macroblock consists of a 16×16 block of luma samples and two 8×8 block of corresponding chroma samples. A macroblock thus consists of four 16×16 Y' blocks, one 8×8 C_b block, and one 8×8 C_r block.

There are four types of macroblocks: **Intra, forward predicted, backward predicted, and bidirectionally coded macroblocks**. Forward and backward predicted macroblocks have only one reference MBs and one motion vectors, while bidirectionally macroblocks are predicted from the average of the past and future reference, as discussed in the foregoing.

Different types of macroblocks can be used mixedly inside a single frame, e.g. macroblocks of a B frame does not necessarily have to be all encoded bidirectionally:

- I frames by definition has to contain exclusively intra coded macroblocks

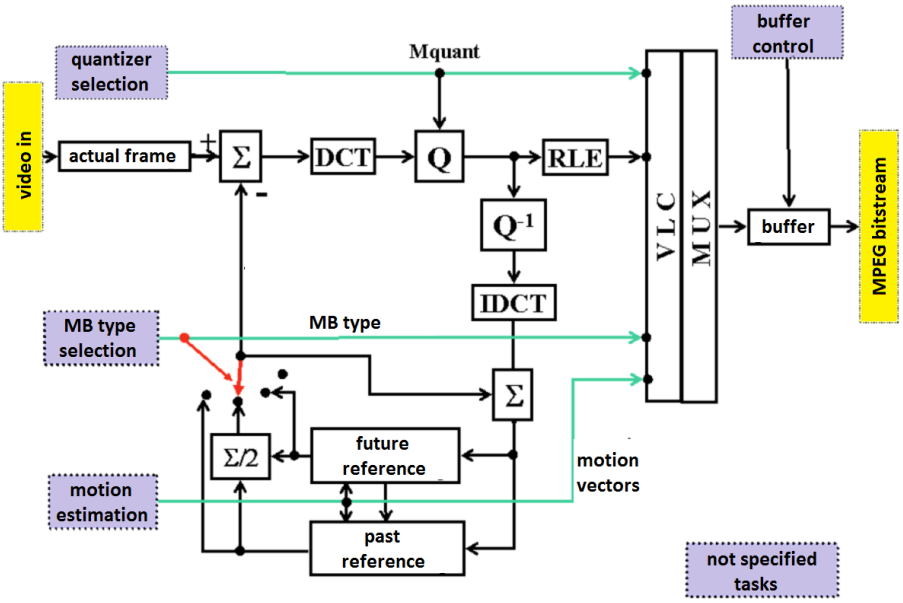


Figure 4.13: The block scheme of the MPEG encoder.

- P frames may contain intra and bacward predicted macroblocks
- B frames may contain all the four MB types.

For predictively coded macroblocks motion compensation can be neglected, in case its motion vector is zero, i.e. the object on the MB did not move compared to the reference frame. Furthermore, in case of predicted macroblocks it often occurs that the entire residual MB is zero. In this case the MB is not encoded, termed as **skipped MB**.

The choice of the MB type inside a P or B frame is the task of the encoder, being the other most time consuming challenge besides motion estimation. Obviously, the brute-force solution would be to encode the actual macroblock with all MB types and choose the encoding type with the least number of output bits. This approach is infeasible in case of real-time applications. Instead, special decision strategies are evaluated, heavily influencing the overall compression efficienvy of the encoder.

4.2.2 The steps of MPEG encoding

As a preprocessing step the MPEG encoder applies the subsampling scheme of 4:2:0 by subsampling the croma information to half resolution both horizontally and vertically. After choosing the appropriate GOP structure (e.g. IBBP... long GOP) the encoding of the individual I, P and B pictures are performed macroblock-wise in the raster scan order from left to right, top to bottom. The processing scheme of one single macroblock is illustrated in Figure 4.13.

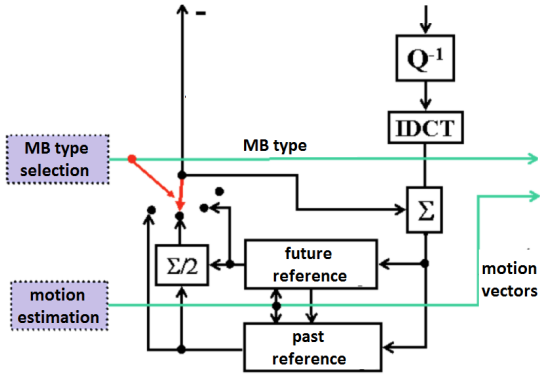


Figure 4.14: The built-in decoder segment of MPEG encoder.

The input of the block diagram is the macroblock of 16×16 luma and 8×8 chroma samples. The block diagram is basically JPEG encoder extended with a feedback prediction loop, or from the opposite point of view, a linear prediction loop extended by an intermediate DCT transforming step.

The feedback prediction loop

The feedback loop contains two reference buffers, storing the reference images for P and B type input macroblocks. These reference buffers can be interpreted as the embedded FIR prediction filter inside the feedback loop with the length of two frames. Each time when a frame arrives at the encoder which can be the reference of following predictively coded frames (i.e. if an I or P frame is decoded) the content of the reference buffers is shifted (the future reference content is shifted to the past reference) and the decoded frame is loaded into the future reference buffer.

The MB type selection is represented by a switch, allowing prediction from past, prediction future reference, bidirectional prediction, or no prediction at all.

- For an intra coded MB the switch is in the uppermost position, and the the following encoding steps are performed directly on the input MB.
- If the current macroblock is encoded predictively, the implemented motion estimation algorithm finds the position of the matching reference MB in the reference frame, as well as appending the resulting motion vectors to the output stream. The resulting reference macroblocks are subtracted from the input MB, an further encoding steps are performed on the residual MB.

The macroblock selection and motion estimation are **non-specified encoding tasks**, meaning that the MPEG standard does not codifies how they should be implemented (but only codifies, how the resulting residual blocks and motion vectors should be embedded into the output stream): the actual MB selection and motion estimation algorithms depends on the actual codec implementation, significantly determining its compression rate and the time consumption.

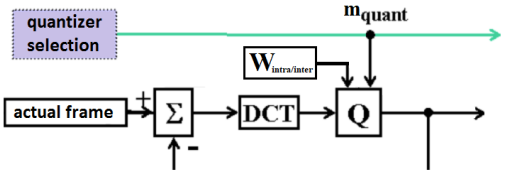


Figure 4.15: The transform coding segment of MPEG encoder.

The content of the reference buffers are obtained by decoding the actual requantized input macroblock, in order to avoid the accumulation of error. Decoding is performed by re-adding the corresponding part of the reference buffers to the quantized residual MB, with the reference MB position given by the motion vectors.

The transform coding step

Each 8×8 input blocks (4 luma and 1-1 chroma blocks) block is encoded by first applying a forward discrete cosine transform and then a quantization process. The DCT process (by itself) is theoretically lossless, and can be reversed by applying an inverse transform to reproduce the original values (in the absence of any quantization and rounding errors). In reality, there are some (sometimes large) rounding errors introduced both by quantization in the encoder and by IDCT approximation error in the decoder. The minimum allowed accuracy of a decoder IDCT approximation is defined by standards.

Similarly to JPEG encoding, irreversible coding is performed by the perceptual based requantization of the DCT coefficients by using a predefined quantization matrix. The MPEG encoder applies different quantization matrices for intra (I) coded frames and predictively coded P and B frames. The default intra quantizer matrix coincides with that of the JPEG encoder. However, the differential, residual macroblocks are already noise-like, with constant spectral density, and equally important coefficients from the perceptual aspect. Therefore, predictively coded blocks are quantized with constant quantization matrix. The default quantization matrices in MPEG-1, MPEG-2 and H.261 standards of intra and precivtive blocks are, therefore,

$$\mathbf{W}_{\text{intra}}=\begin{bmatrix} 8 & 16 & 19 & 22 & 26 & 27 & 29 & 34 \\ 16 & 16 & 22 & 24 & 27 & 29 & 34 & 37 \\ 19 & 22 & 26 & 27 & 29 & 34 & 34 & 38 \\ 22 & 22 & 26 & 27 & 29 & 34 & 37 & 40 \\ 22 & 26 & 27 & 29 & 32 & 35 & 40 & 48 \\ 26 & 27 & 29 & 32 & 35 & 40 & 48 & 58 \\ 26 & 27 & 29 & 34 & 38 & 46 & 56 & 69 \\ 27 & 29 & 35 & 38 & 46 & 56 & 69 & 83, \end{bmatrix} \quad \mathbf{W}_{\text{inter}}=\begin{bmatrix} 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 & 16 & 16 & 16 & 16. \end{bmatrix} \tag{4.8}$$

The bitrate for a given macroblock can be adjusted by choosing the **quantization scale** ($\mathbf{m_{quant}}$) in the same manner as in the JPEG encoder: The quantizer matrix is multiplied with the value of the quantization scale. Hence, the larger quantization

scale results in the higher number of zero coefficients after quantization which can be encoded with less bits at the cost of potential visible artifacts due to compression. The quantization scale, therefore, directly defines the quality of the output video stream along with its bit rate.

The default quantizer scale can be set in each slice header, however, its value can be changed from the default for each macroblock. Therefore—unlike JPEG, which allowed a single quantization scale for the entire image—the MPEG encoder allows quality control on macroblock level. The choice of the actual quantizer scale is again a non-specified encoder task, determining the efficiency of the actual encoder implementation.

The entropy coding

The MPEG encoder applies the same entropy encoding scheme as the JPEG codec, applying mixed Huffmann coding and variable length integer representation. The entropy coded video data are the following:

- For I frames the DC coefficients are encoded differentially, by storing only the difference from the previous block's DC coefficient. The AC coefficients are zig-zag ordered and run-level encoded.
- For P and B frames the DC coefficients does not represent the mean intensity anymore, therefore, both the DC and AC components are zig-zag ordered and run-level coded.
- Due to the statistical similarity of adjacent motion vectors, the motion vectors are also differentially encoded.

Simple encoding example

As a simple example consider the encoding process of a long GOP consisting of IBBPBBP... input pictures. Due to the presence of the B pictures the encoding order differs from the presentation order, given by IPBBPBB.... The steps of encoding of the first three frames is the following:

- First the I picture is encoded. Since being an intra coded image (the switch of 4.13 is in the rightmost position) nothing is subtracted from the input macroblocks. The MBs are DCT transformed and quantized with the given quantizer scale, and the requantized MB is appended to the output data stream. On the same time, the MB is decoded (mutlipled with the quantizer scale and the quantization matrix and inverse transformed) and stored in the future reference buffer.
- After encoding all the I macroblocks of the first I frame, the next encoded image is the P frame. Since the decoded frame will be stored in the reference buffer, first the content of the future reference is shifted to the past reference. The encoder has to decide whether the actual input MB is intra encoded, or predicted from the past. In the latter case the switch is adjusted to the leftmost position, and after motion estimation the resulting reference MB is subtracted from the input MB.

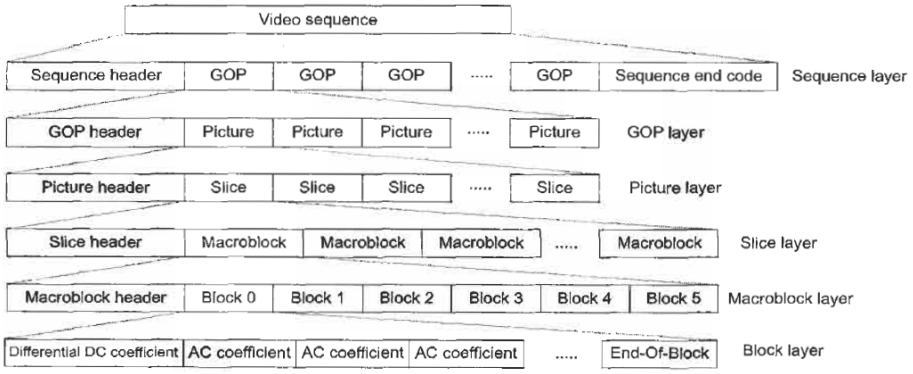


Figure 4.16: MPEG-I bit-stream syntax layers.

The residual MB is transformed, quantized and appended to the input stream. At the same time it is decoded by inverse transforming and re-adding the reference MB. Finally the decoded MB is stored in the future reference buffer.

- The next input image is a bidirectionally coded B image. The content of the reference buffers remain unchanged, with the past reference containing the first I frame and the future reference containing the P frame. For each input macroblock the encoder has to decide, if the MB is intra coded, or predicted from the past, future or both. In the example of the latter case the switch is adjusted into the second-from-the-left position, and after motion estimation the average of the reference MBs is subtracted from the input. The residual is transformed, quantized and appended to the input stream. Since the B picture can not serve as a reference picture, therefore, it is not stored in the reference buffer.

The structure of the resulting video stream Figure 4.16.

4.2.3 Bit-rate control of MPEG encoding

The number of bits generated by the MPEG encoder over unit time highly varies due to several reasons:

- The I, P and B pictures require different number of bits to represent while ensuring the same percepted quality
- Inside a frame the bits for the representation of the individual macroblock vary as the function how much details an actual MB contains.
- Also variable length coding causes a fluctuation in the output bitrate.

The number of generated bits can be directly affected in the MPEG video encoder by adjusting the quantization scale of the individual macroblocks.

Depending on whether the fluctuation of output data rate can be allowed by the transmission channel, two types of encoding modes can be distinguished: constant bitrate and variable bit rate coding.

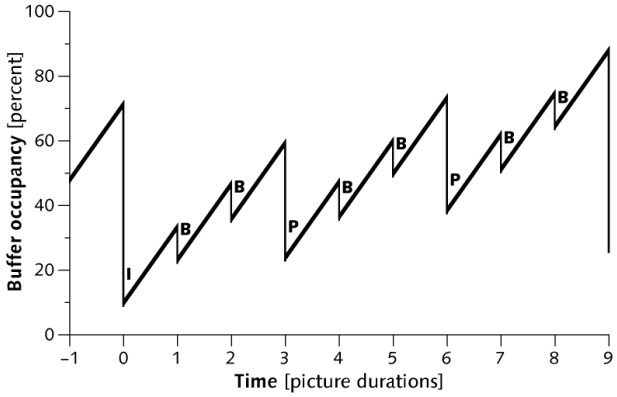


Figure 4.17: The input buffer of an MPEG decoder.

Constant bit-rate

Constant bit rate encoding means that the rate at which a codec's output data should be consumed is constant. CBR is useful for streaming multimedia content on limited capacity channels since it is the maximum bit rate that matters, not the average, so CBR would be used to take advantage of all of the capacity.

The flattening of the MPEG output data rate is achieved by placing an output buffer to the end of the encoder. The encoder loads the encoded video data with the frame period time, while the channel (and the receiver) reads the data from the buffer with a fixed data rate. On the decoder side an input buffer is present, buffering the received video data with constant bitrate, and the decoder reads the data required for the decoding of the individual frames with the frame period time. This process is illustrated in Figure 4.17.

The criterion of decodability is that no underflow or underflow may arise in the buffering process, which could occur if a frame is represented on too much or too few bits. The solution in order to avoid underflow and overflow is the concept of **Video Buffer Verifier (VBV)** of the MPEG encoder.

The VBV is a hypothetical decoder buffer attached to the MPEG encoder, with the operation illustrated in Figure 4.18. The VBV simulates the operation of the decoder side, loaded by the output of the encoder buffer with constant bitrate, R. After a predefined initial buffering time T_D the hypothetical decoder reads out a certain number of bits, needed for decoding one single frame. From this point the decoder reads each frame data after one frame period time. The goal of bitrate control on the encoder side is to follow the VBV occupancy and control the number of the bits for the representation the next frame, so that no overflow or underflow may happen in the VBV buffer. This bitrate control can be achieved by adjusting the quantizer scale during encoding.

As a simple example: assume that the buffer occupancy after the decoding of the $n-1$-th image is given by B_{n-1}, and the maximal buffer capacity is $B_{\max}$. With denoting the frame period time by T and the constant bitrate by R, from simple geometric consideration the minimal (L_n) and maximal (U_n) number of bits for representing the

n-th image is given by

$$L_n = B_{n-1} + 2R \cdot T - B_{\max},$$
$$U_n = B_{n-1} + R \cdot T,$$

In practice the encoder transmits the required decoder buffer size in order to decode the given MPEG video stream.

Variable bitrate

As opposed to constant bitrate, variable bitrate (VBR) streams vary the amount of output data per time segment. VBR allows a higher bitrate (and therefore more storage space) to be allocated to the more complex segments of the video data while less space is allocated to less complex segments. Two basic types of VBR encoding exists:

- **Open loop control** applies no output buffer at all, or the output buffer can be considered infinitely large. This approximately holds for local media playback, i.e. reading video stream from DVD, Blue-ray, or directly from memory. Open loop control can be further subdivided to two types:
  - Constant scale factor encoding, in which case the quantizer scale is constant inside an image (or the entire stream). Obviously, this approach does not ensure optimal video quality.
  - Constant quality encoding, in which case the scale factor is adapted to hold the perceived image quality constant. This approach requires thorough image analysis before choosing the optimal quantizer scale. The approach is widely used for DVD and Blue-ray encoding.
- **Feedback control** allows variable bitrate at the output of the encoder, however, limits the bitrate to a predefined maximal level in order to adapt the video transmission to the transmission channel characteristics. This approach is widely used for video streaming services.

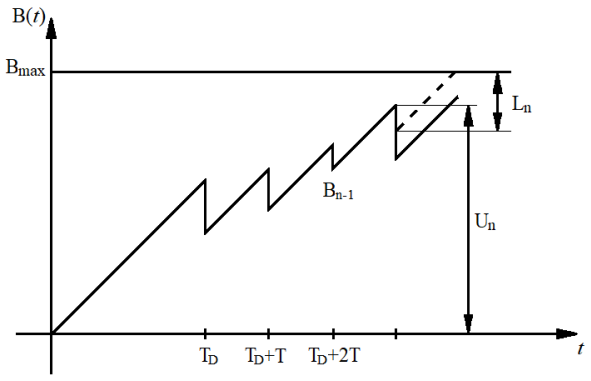


Figure 4.18: The Video Buffer Verifier.

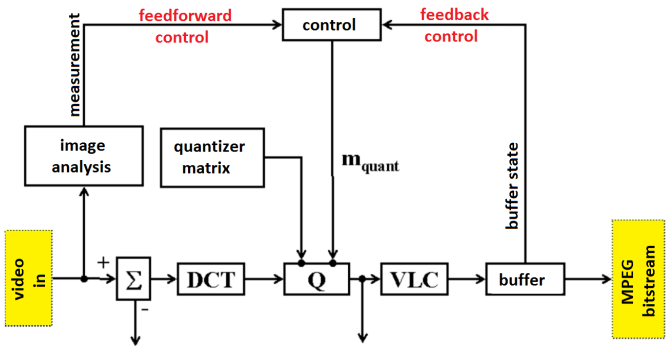


Figure 4.19: Bit-rate control in MPEG encoder

The two VBR encoding types are frequently used in a mixed manner, i.e. with image analysis and output buffering. This approach is presented in Figure 4.19. The steps of bitrate control in this processing schemes are the following:

- First the number of for representing the GOP and the individual I,P and B frames is defined, setting the global video quality.

- By analyzing the actual image the quantizer scales of the individual macroblocks are set, based on the complexity of the macroblocks. This step preserves local quality.

- Finally, the quantizer scale may be modified based on the VBV buffer state.

The advantages of VBR are that it produces a better quality-to-space ratio compared to a CBR file of the same data. The bits available are used more flexibly to encode the sound or video data more accurately, with fewer bits used in less demanding passages and more bits used in difficult-to-encode passages.

The disadvantages are that it may take more time to encode, as the process is more complex, and that some hardware might not be compatible with VBR files. VBR may also pose problems during streaming when the instantaneous bitrate exceeds the data rate of the communications path. These problems can be avoided by limiting the instantaneous bitrate during encoding or (at the cost of increased latency) by enlarging the playout buffer.

In the past, many hardware and software players could not decode variable bitrate files properly, partly because the various VBR encoders used were not well developed, resulting in common use of CBR over VBR for the sake of compatibility. Nowadays, devices that support only CBR encoded files are largely obsolete, as the vast majority of modern devices and software support VBR encoded files.

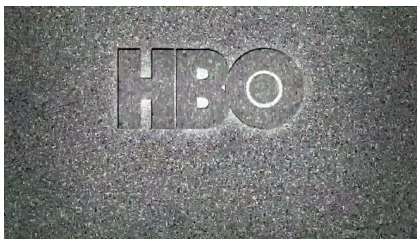
4.2.4 Summary

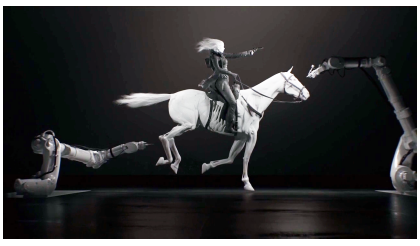
MPEG-1 was mainly developed for early storage media applications. Virtually, the standard allowed the encoding of video sequence up to the image size of 4095×4095 pixels with progressive scanning and the frame rate of 24, 25, 29.97, 30 fps. In practice the MPEG-1 encoding ensures optimal image quality at the target bitrate of $1 - 1.5$ Mbit/s only for a given set of parameters, being 352×288 at 25 fps, or 352×240 at 30 fps. These parameters were later coined as **Constrained Parameter Bitstream (CPB)**.

The MPEG-1 encoder is much more expensive than the decoder because of the large search range, the half-pixel accuracy in motion estimation, and the use of the bidirectional motion estimation. The MPEG-1 syntax can support a variety of frame rates and formats for various storage media applications. Similar to other video coding standards, MPEG-1 does not specify every coding option (motion estimation, rate control, coding modes, quantization, preprocessing, postprocessing, etc.). This allows for continuing technology improvement and product differentiation.

End-of-Chapter Questions

- What is the goal of motion estimation?
- What is the goal of block matching? Name several frequently used block matching algorithms.
- Draw the block diagram of an MPEG encoder! Explain the steps of the encoding process if the applied GOP structure is IBBP!
- Assume two frames from two video sequences: one containing white noise, the other frame contains an arbitrary object in front of a black background, as shown in the following example:





Both frames are encoded as I frames. Which frame can be encoded more efficiently and why? The explanation should contain how the DCT matrices and the run-level pairs typically look like in the two above examples.