Samenvatting Multimedia

D.E.R.P.

2 juni 2019

Inleiding

Deze samenvatting is gemaakt voor De Examen Repo Plaats (DERP), een GitHub-organisatie die te vinden is op `http://github.ugent.be/derp`, die dergelijke samenvattingen bevat. Het doel van DERP is om een plaats te hebben voor de LaTeX-broncode van samenvattingen, zodat deze altijd up to date kunnen gehouden worden. De broncode van deze specifieke samenvatting is te vinden op `http://github.ugent.be/derp/Multimedia`. De gecompileerde PDF is te vinden op `http://ebooks.rxn.be/samenvattingen/Multimedia`

Als je een fout tegenkomt, of bepaalde stukken wil uitbreiden, kan dat altijd door de repo te `forken` en een `pull request` aan te maken met jouw aanpassingen. Na een review kan jouw contributie deel uitmaken van het DERP-project en zo toekomstige studentjes uit de nood helpen.

Deze samenvatting is deels ontstaan op basis van `https://github.ugent.be/stvermas/multimedia_samenvatting`, door Stefaan Vermassen (2011).

Contributors

Deze samenvatting is vrij met dank aan:

- Pieter De Clercq (2016–2017)
- Julie De Meyer (2016–2017)
- Ruben Maes (2016–2017)
- Robbe Vanhaesebroeck (2016–2017)

Opmerking

Hoofdstuk 8 en **Hoofdstuk 9** ontbreken wegens tijdsgebrek.

Hoofdstuk 1

Introduction to Multimedia

1.1 Wat is Multimedia?

1.1.1 Definitie

Veel verschillende uiteenlopende meningen.

- **MediaMarkt-medewerker:** interactieve TV, smartphone, ...
- **Informaticastudent:** applicaties die meerdere modaliteiten gebruiken, zoals tekst, afbeeldingen, tekeningen, geluid, ...

Multimedia combineert verschillende modaliteiten (tekst, audio, afbeeldingen, animatie, video ...) bij elkaar.

Voorbeelden: augmented reality, shapeshifting TV, objecten op naam zoeken in films, kijker kan keuzes maken over verdere verloop van film, ...

1.1.2 Huidige projecten

- Camera die objecten zelf kan volgen
- 3D motion
- Meerdere kijkhoeken
- Interactieve medische oplossingen
- Digitale mode

1.2 Multimedia: verleden en toekomst

1.2.1 Verleden

Kranten, films (1830), geluid (Edisons Fonograaf – 1877), radio (Marconi – 1895), televisie (20ste eeuw)

1.2.2 Computers en multimedia

- World Wide Web (Tim Berners-Lee – 1989), JPEG (1992), DVD (1996), MP3-spelers (1998)
- **Hypertext:** Tekst die niet lineair gelezen moet worden, via links naar andere delen van het document springen.
- **Hypermedia:** Hypertext die niet enkel tekst is, maar ook foto's, audio, video.

1.2.3 Multimedia in jaren 2000

Skype (2003), YouTube (2005), Twitter (2006), iPhone+Android (2007)

1.3 Multimediasoftwaretools: een overzicht

Een overzicht van 1 bekende tool per categorie, aangezien dit hoogstwaarschijnlijk toch niet gevraagd zal worden.

1. **Muzieknotatie:** Sibelius
2. **Digitale audio:** Pro Tools
3. **Graphics en fotobewerking:** Adobe Photoshop
4. **Videobewerking:** Final Cut Pro
5. **Animatie:** API's (OpenGL) – Animatiesoftware (Autodesk 3DS Max)
6. **Authoring:** Complete, interactieve multimedia presentaties. Voorbeeld is het (dode) Adobe Flash.

1.4 Multimedia in de toekomst

Zie Huidige projecten.

Hoofdstuk 2

A Taste of Multimedia

2.1 Taken en Uitdagingen

- **Plaatsbepaling:** Waar is deze foto genomen?
- **Objectklassificatie:** Staat er een luchtballon op deze foto?
- **Objectdetectie:** Waar staat er iets belangrijks op deze foto?
- **Segmentatie:** Tot welk object behoort deze pixel?

2.2 Presentatierichtlijnen

2.2.1 Kleurprincipes

Bepaalde kleuren passen beter bij elkaar dan andere, gebruik in het algemeen niet téveel kleuren.

2.2.2 Lettertypes

Grote lettertypes, niet meer dan 6–8 lijnen per scherm.

2.2.3 Kleurcontrast

Contrast is vaak goede kleur voor achtergrond: $(R, G, B) \Rightarrow (1 - R, 1 - G, 1 - B)$

2.3 Datacompressie

1. Idee geven van ongecomprimeerde bestandsgroottes: 1 uur HD-video = 672 GB.
2. JPEG-compressie: kwaliteitsfactor van 75% verandert vaak niet veel aan de kwaliteit, maar bestandsgroote kan wel meer dan 10 keer kleiner zijn! Kwaliteitsfactor verder verlagen zorgt voor meer JPEG-artefacten.

2.4 Productie

Verschillende mensen; graphisch ontwerper, projectmanager, schrijver, programmeur, UI-ontwerper, audio-ontwerper, animators ...

2.5 Distributie

YouTube

2.6 Bewerkingssoftware

Zie sectie 1.3.

Hoofdstuk 3

Graphics and Image Data Representations

3.1 Datatypes

Enkele bestandsformaten

- **Afbeeldingen:** BMP, GIF, JPG, PNG, PSD ...
- **Audio:** AAC, M4A, MP3, OGG, WMA ...
- **Video:** AVI, MOV, MP4, WMV ...

Container vs codeerformaat

Een container bundelt video, audio, ondertitels en eventuele menu's samen. Deze video, audio en ondertitels mogen in eender welk codeerformaat staan.
Voorbeeld: een MP4-container die MP3-audio bevat en MPEG-2-video.

3.1.1 1-bit-afbeeldingen

- Elke pixel opgeslagen als één enkele bit (0 of 1) ⇒ **binaire afbeelding**
- Geen kleur ⇒ **monochroom**

3.1.2 8-bit-grijswaarden

- Elke pixel opgeslagen als één byte, waardes 0–255. Lagere waarden zijn donker, hoge waarden helder.
- **Bitmap:** 2D array van pixelwaarden die de afbeelding voorstellen.
- **Resolutie:** Aantal pixels in een afbeelding, uitgedrukt in $width \cdot height$
- **Frame buffer:** Hardware waarin bitmap opgeslagen wordt ⇒ grafische kaart

- **Bit-plane:** 8-bit-beeld is verzameling van 1-bit-planes, elke plane is 1-bit representatie van het beeld en voegt zo steeds meer informatie toe om de uiteindelijke 8-bit-representatie te bekomen.
- Elke pixel opgeslagen als 1 byte $\Rightarrow 640 \times 480$ grijswaarden = 307.2kB

Dithering

- **Wat:** Dot-patronen berekenen zodat 8-bit kan worden omgezet naar 1-bit voor bijvoorbeeld een printer.
- **Hoe:** Dit wordt gedaan door een 8-bit-dot op te splitsen in, bijvoorbeeld een 2×2 dither-matrix $\left(\begin{smallmatrix} 0 & 2 \\ 3 & 1 \end{smallmatrix}\right)$. Aantal in te kleuren dots = 256//5.
  1. Herschaal intensiteit van de dot (bv. 255) naar het aantal mogelijkheden van de matrix (bv. [0..4]).
  2. Als intensiteit groter is dan waarde op die plaats in de matrix, plaats een dot, anders doe niets.
- **Gevolgen:** Grotere matrix $\Rightarrow$ Meer soorten grijs $\Rightarrow$ Groter bestand
- **Oplossing:** Ordered dithering: kies een grotere matrix, bijvoorbeeld 4×4 en leg die over de afbeelding. Zet de 1-bit enkel op 1 als de waarde voor die pixel groter is dan het element in de matrix op die positie, en schuif dan de matrix op om zo steeds gebieden van de afbeelding te beschouwen in plaats van elke pixel individueel. $\Rightarrow$ Grootte blijft gelijk.

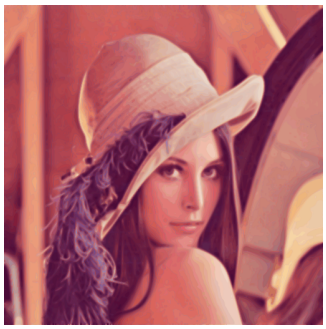
3.1.3 Afbeeldingstypes

Meest voorkomende is 24-bit-kleur en 8-bit-kleur. Meestal is er een vorm van compressie ingebouwd om de bestandsgrootte te beperken. Compressie kan zowel **lossless** (er gaat niets

verloren, enkel redunante info verwijderen) als **lossy** (er gaat wel iets verloren, dus sterkere compressie) zijn.

3.1.4 24-bit kleurenafbeeldingen

- Elke pixel 3 bytes (R,G,B componenten als 0–255).
- $256 \times 256 \times 256 = 16777216$ mogelijke kleuren
- Zonder compressie: 640×480 24-bit-kleurenafbeelding $= 921.6$ kB $\Rightarrow$ compressie is nodig
- **Alphakanaal:** Veel 24-bit-afbeeldingen worden eigenlijk als 32-bit opgeslagen, de extra byte duidt per pixel de transparantie aan.



3.1.5 Afbeeldingen met grotere diepte

Speciale afbeeldingen die meer dan enkel RGB bevatten, bijvoorbeeld ook infrarood, ultraviolet, enz. worden **multispectraal** (meer dan 3) of **hyperspectraal** (bijvoorbeeld 200+ kleuren) genoemd.

3.1.6 8-bit-kleurenafbeeldingen

Kleurinformatie wordt opgeslagen in **lookup tables**. Afbeeldingen slaan dus geen 3 bytes op per pixel, maar wel een index in een tabel met 3-byte-waarden. Hierdoor wordt elke kleur maar één keer opgeslagen en is de bestandsgrootte kleiner (tenzij de afbeelding uit allemaal unieke kleuren zou bestaan).

3.1.7 Color Look-up Tables

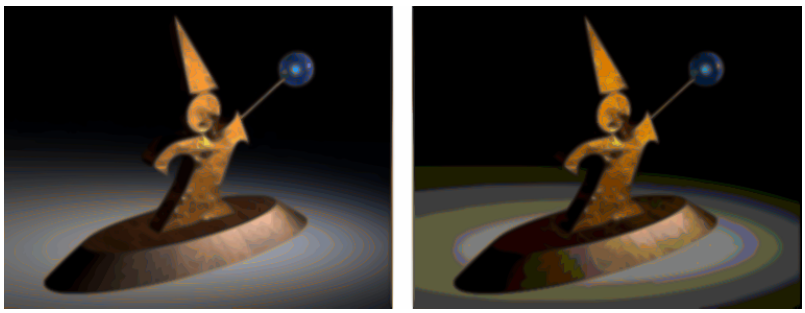
- **Idee:** Tabel maken met per rij 3 bytes (R,G,B). Op elke pixel waar deze kleur voorkomt, sla niet de RGB-bytes op maar sla wel het rijnummer van de tabel op.
- **Colorpicker:** Eigenlijk een lookup table met typisch 255 kleuren, zodanig dat als je op een plaats klikt je de RGB-bytes krijgt die op die index zitten.
- **Color cycling:** De kleurtabel in de picker vervangen door een andere, met 255 nieuwe random kleuren.

Color Look-up Table opstellen

Uniforme kwantisatie

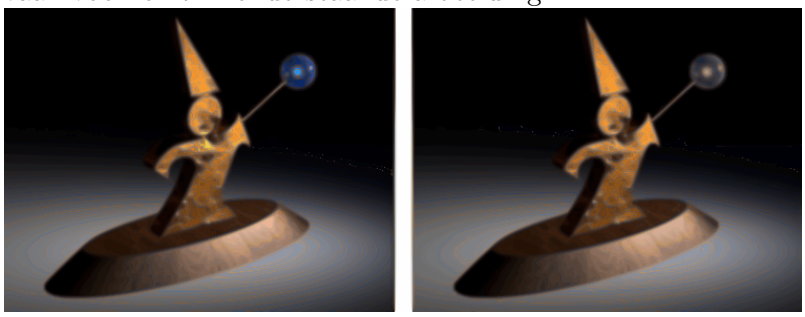
Menselijk oog is gevoeliger voor rood en groen dan voor blauw ⇒ verdeel RGB-kleurenruimte in 3 delen, hiermee rekening houdend. Een probleem hiermee is natuurlijk wel dat er (veel) informatie verloren gaat door de omzetting van 8-bit naar 3-bit.

- R: 8-bit ⇒ 3-bit (0 tot 7). $R_{lut} = \frac{R_{orig}}{256/8} = \frac{R_{orig}}{32}$
- G: 8-bit ⇒ 3-bit (0 tot 7). $G_{lut} = \frac{G_{orig}}{256/8} = \frac{G_{orig}}{32}$
- B: 8-bit ⇒ 2-bit (0 tot 3). $B_{lut} = \frac{B_{orig}}{256/4} = \frac{B_{orig}}{64}$



Popularity-based kwantisatie

De 256 kleuren die het vaakst voorkomen in de afbeelding worden opgeslagen in de look-up table. Als een kleur van een pixel niet voorkomt in de look-up table, wordt de dichtstbijzijnde kleur gekozen die er wel in zit. De blauwe bol wordt dus niet ingekleurd omdat blauw niet vaak voorkomt in onderstaande afbeelding.



Median-cut kwantisatie

- **Idee:** Als er een bepaalde kleur in een afbeelding domineert, bijvoorbeeld rood, plaats dan veel meer rode tinten in de look-up table.
- **Werking:**
  1. Steek alle kleuren in een bucket.

2. Bepaal welke kleur het grootste bereik heeft, bijvoorbeeld rood, en sorteer alle kleuren op basis hiervan.
3. Bepaal de mediaan en maak op basis hiervan 2 buckets (1 met alle kleuren boven de mediaan, 1 met alle kleuren eronder).
4. Herhaal stap 2 en 3 een paar keer tot het gewenste aantal buckets, bijvoorbeeld 256.
5. Neem de gemiddelde kleur van elke bucket en steek deze kleur in de look-up table.

3.2 Populaire bestandsformaten

3.2.1 GIF: Graphics Interchange Format

8-bit-kleurenafbeeldingen (256 kleuren), best geschikt voor afbeeldingen met weinig kleuren zoals tekeningen. Animaties toegevoegd in GIF89.

Structuur GIF87

1. GIF-signature
2. Screen descriptor
   (a) Raster breedte
   (b) Raster hoogte
   (c) Achtergrondkleur (haal uit globale LUT)
3. Globale LUT
4. Voor elke afbeelding in dit bestand
   (a) Image descriptor
      i. Links pixels start
      ii. Top pixels start
      iii. Breedte
      iv. Hoogte
   (b) Lokale LUT
   (c) Raster
5. GIF Terminator

Interlacing

Ondersteund in GIF, met een 4-pass-sequentie. Rijen van de afbeelding worden in vier stukken opgeslagen, zodat er na het ontvangen van één stuk al een grof geschetst beeld van de volledige afbeelding zichtbaar is. De rest van de afbeelding wordt dan per stuk verder zichtbaar, dit is ontwikkeld toen het doorsturen van een afbeelding nog zeer lang duurde.

3.2.2 JPEG: Joint Photographic Experts Group

Maakt gebruik van beperkingen in menselijk zicht voor zeer sterke compressie. Compressieratio (kwaliteit) kan zelf gekozen worden door gebruiker.

3.2.3 PNG: Portable Network Graphics

- Tot 48-bits-kleurinformatie (16-bits per kanaal)
- Kan een alfakanaal bevatten, of informatie voor gammacorrectie
- Progressief weergegeven door 7 *passes*. Hierdoor is de afbeelding al van in het begin volledig zichtbaar, maar lijkt ze scherper te worden naarmate er meerdere passes geladen worden.

3.2.4 TIFF: Tagged Image File Format

- Verschillende types; 1-bit, 8-bit kleur, grijswaarden, 24-bit ...
- Lossless

3.2.5 EXIF: Exchange Image File

- Voor digitale camera's
- Informatie over de camera en omstandigheden worden ook opgeslagen (witbalans, flits ...)

3.2.6 PS: PostScript

- Vectorgebaseerd, niet pixelgebaseerd
- Bevat tekst en vectors/gestructureerde afbeeldingen
- Geen compressie
- Gerelateerd: **PDF: Portable Document Format**

3.2.7 Andere bestandsformaten

- WMF: Windows Meta File
- BMP: Bitmapstandaard voor Windows, bevat veel varianten
- Netbpm formaat: Linux/Unix-georiënteerd
- PTM: Polynomial Texture Mapping: meerdere representaties van eenzelfde beeld, maar met bijvoorbeeld een verschillende lichtinval. ⇒ **Light Field Cameras**

3.3 3D graphics

3.3.1 Game-architectuur

Gamescripts ⇔ game-engine (AI, scènemanager, fysische verschijnselen, geluid, renderer) ⇔ Grafische kaart

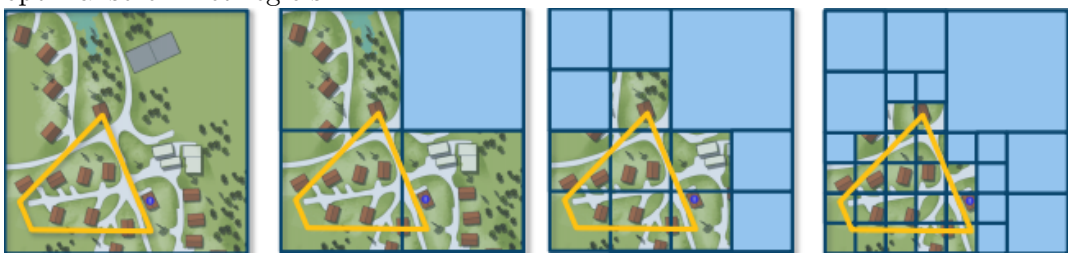
Scènestructuren

Verschillende delen van de engine moeten allemaal aan “de wereld” kunnen. Lineair zoeken zou veel te traag zijn dus wordt de wereld opgedeeld in verschillende blokken met hash/boom-structuren ⇒ sub-lineair.

3.3.2 Renderer

Culling

Bepalen welke delen er volgens de huidige camerahoek zichtbaar zijn en welke niet. Weer optimaliseren met regio’s:

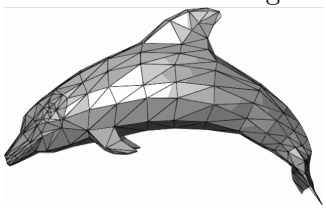


Het resultaat hiervan wordt verder verwerkt en met behulp van een grafische bibliotheek, zoals DirectX of OpenGL naar de grafische kaart gestuurd.

3.3.3 Grafische kaart

Vertex setup

Transformatie van **object space** naar **world space** naar **camera space** naar **screen space**. Hieruit worden triangulaties gegenereerd.



Rasterisatie

Triangulaties worden omgezet naar schermpixels (fragmenten). De kleur hiervan wordt later bepaald.

Pixel shading

Kleur van de fragmenten, gegenereerd door de rasterizer, wordt bepaald. Dit gebeurt parallel voor alle pixels.

Framebufferoperaties

Alle fragmenten worden op elkaar geplakt om zo het uiteindelijke beeld te bekomen.

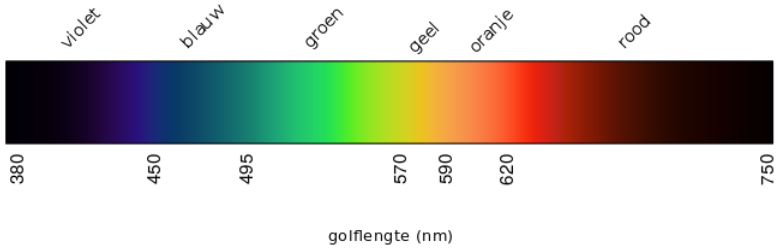
Hoofdstuk 4

Color in Image and Video

4.1 Kleurwetenschap

4.1.1 Licht en spectra

- **Licht:** elektromagnetische golf. Kleur bepaald door golflengte. Kort = blauwe tint, lang = rood. Golflengte zichtbaar licht: 400 nm–700 nm.
- **Spectrofotometer:** zichtbaar licht meten door licht te reflecteren naar een oppervlak dat de straal opsplitst in verschillende golflengtes.



4.1.2 Menselijk zicht

Ooglens focust beeld op het netvlies (ondersteboven en gespiegeld). Netvlies bestaat uit heel veel staafjes (licht-donkergrijswaarden) en kegels (3 soorten: R,G,B). Hersenen gebruiken waarschijnlijk de verschillen in kleuren $R-G$, $G-B$, $B-R$ alsook een achromatisch kanaal waarin R,G,B gecombineerd zijn.

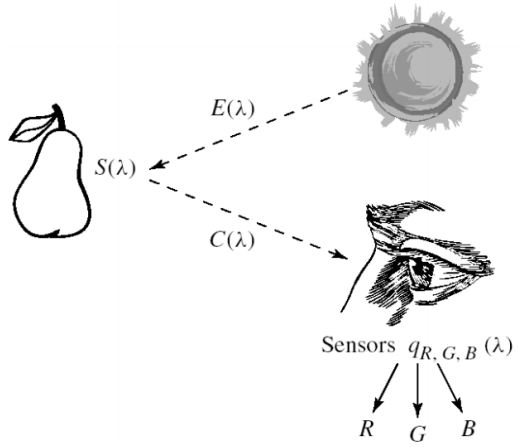
4.1.3 Spectrale gevoeligheid van het oog

Meest gevoelig voor licht in het midden van zichtbaar spectrum. Blauwe receptor is veel kleiner, mens is minder gevoelig voor blauw. Achromatisch kanaal bepaald door: $A = 2 \cdot R + G + \frac{B}{20}$
SPD: Spectral Power Distribution = spectrum, curve E(λ) met op de x-as de golflengte
Metameren: Twee lichten die er visueel hetzelfde uitzien maar mogelijks verschillende **SPD** hebben.

4.1.4 Beeldvorming

Lichtstralen worden door een oppervlak weerkaatst op verschillende golflengten (donkere oppervlakken reflecteren minder).
$S(\lambda)$: Spectralereflectiefunctie van het oppervlak. Kleursignaal C: $C(\lambda) = E(\lambda) \cdot S(\lambda)$
Uiteindelijke beeldvorming gedefinieerd door:

- $R = \int E(\lambda) \cdot S(\lambda) \cdot q_R(\lambda) \cdot d\lambda$
- $G = \int E(\lambda) \cdot S(\lambda) \cdot q_G(\lambda) \cdot d\lambda$
- $B = \int E(\lambda) \cdot S(\lambda) \cdot q_B(\lambda) \cdot d\lambda$



4.1.5 Camerasystemen

Drie analoge signalen worden op gelijkaardige manier omgezet naar integers. Bij een precisie van 8-bit liggen de waarden van R, G, B tussen 0 en 255. Computerscherm is een lichtbron die rechtstreeks naar het oog gaat (geen reflecterend oppervlak).

4.1.6 Gammacorrectie

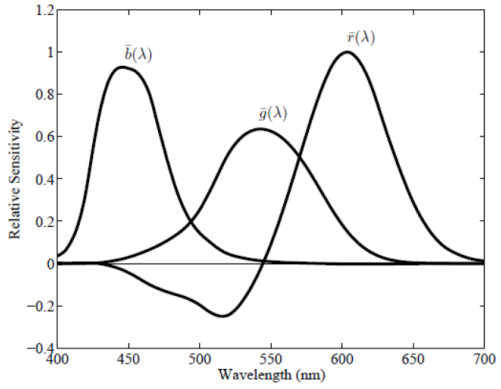
In CRT-monitor (cathode ray tube, kathodestraalbuis) worden de RGB-waarden van elke pixel omgezet naar analoge spanning om de elektronenkanonnen aan te sturen (de spanning bepaalt de intensiteit van het licht).

De gegenereerde lichtintensiteit is lineair ten opzichte van de spanning die CRT aanstuurt **tot de macht γ** (gamma), dus het bekomen beeld op de CRT wijkt af (kleurtinten veel donkerder). Waarde van γ ligt meestal tussen 2,2–2,5.

Gegenereerde signaal: V^γ met V spanning, γ de gammafout.
Gammacorrectie: $V^{\frac{1}{\gamma}}$

4.1.7 Kleurmatching

Colorimeter-experiment: Primaire set lichtstralen (kies drie kleuren, bijvoorbeeld RGB) richten op een witte achtergrond. Testpersoon moet de helderheid van deze drie stralen zodanig afregelen tot de resulterende lichtstraal dezelfde kleur heeft als een andere gekleurde lichtstraal die daarnaast wordt afgebeeld. Hieruit volgen de relatieve kleurmatching-functies: $\overline{r}(\lambda), \overline{g}(\lambda), \overline{b}(\lambda)$. Verklaring voor negatieve rood: het resultaat kon niet worden gevonden uit een lineaire combinatie van drie lichtstralen, dus op het resultaat werd wat rood toegevoegd zodat het wel gevonden kon worden.



4.1.8 CIE-chromaticiteitsdiagram

De $\overline{r}(\lambda)$-curve heeft een negatief deel, dus verzin een fictieve primaire set $\overline{x}(\lambda), \overline{y}(\lambda), \overline{z}(\lambda)$ (resultaat van een matrixtransformatie van de $\overline{r}(\lambda), \overline{g}(\lambda), \overline{b}(\lambda)$-curves). Transformatiematrix zodanig gekozen dat $\overline{y}(\lambda)$ overeenkomt met luminantie-efficiëntiecurve $V(\lambda)$.

Voor gegeven lichtbron met SPD $E(\lambda)$ wordt kleurinformatie vastgelegd door volgende functies:

- $X = \int E(\lambda) \cdot \overline{x}(\lambda) \cdot d\lambda$
- $Y = \int E(\lambda) \cdot \overline{y}(\lambda) \cdot d\lambda$
- $Z = \int E(\lambda) \cdot \overline{z}(\lambda) \cdot d\lambda$

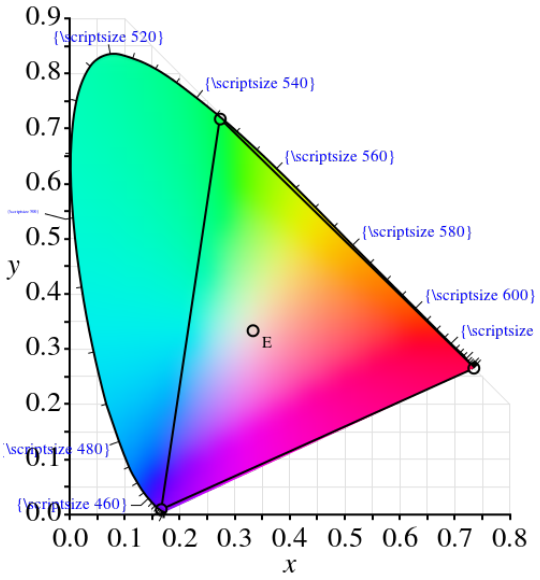
3D-data kan moeilijk worden weergegeven dus bereken een 2D-weergave op volgende manier:

- $x = X/(X + Y + Z)$
- $y = Y/(X + Y + Z)$
- $z = Z/(X + Y + Z)$

z kan worden geschrapt omdat uit bovenstaand stelsel volgt dat $z = 1 - x - y$.

Bekomen CIE[1]-chromaticiteitsdiagram:

[1]Commission internationale de l'éclairage, International Commission on Illumination



- **Gamut:** (driehoek) alle kleuren die gemaakt kunnen worden met de 3 primaire kleuren.
- Witpunt op $(\frac{1}{3}, \frac{1}{3})$
- Zwart is afwezigheid van kleur
- Gesatureerde kleuren: kleuren die op de curve liggen (spectrum**locus**)
- Minder gesatureerde kleuren naarmate je meer naar witpunt gaat

4.1.9 Kleurmonitor

Witpunt: chromaticiteit als alle RGB-elektronenkanonnen op hoogste waarde staan (1.0).

4.1.10 Kleuren buiten het gamut

Wanneer een kleur buiten het gamut van een apparaat valt, zijn er twee mogelijke oplossingen.

1. Neem de dichtsbijzijnde kleur die wel in het gamut ligt.
2. Neem de dichtste complementaire kleur.

Wet van Grassman: Additieve kleurmatching is lineair: met twee gegeven kleuren kan je, door middel van lineaire combinaties, alle kleuren maken die op de lijn tussen die twee kleuren liggen.

4.1.11 Witpuntcorrectie

Probleem

Witte punten exact wit maken, want dit verschilt per apparaat. Er worden drie correctiefactoren berekend waarmee de X,Y,Z omgezet kunnen worden naar apparaatspecifieke RGB-waarden.

4.1.12 XYZ⇒RGB-transformatie

Via matrixtransformaties.

4.1.13 Transformeren met gammacorrectie

Eerst lineair transformeren, dan gammacorrectie toepassen.

4.1.14 L*a*b (CIELAB) kleurmodel

Wet van Weber: Als menselijk zintuig signalen ontvangt met sterktes: $1 \ldots a \ldots a^2 \ldots a^3$, dan zal de menselijke waarneming dit vertalen in gewaarwordingen met intensiteit: $x \ \ldots \ x + b \ \ldots \ x + 2b \ \ldots \ x + 3b$, gelijke verhoudingen in signaalsterkte worden dus ervaren als gelijke verschillen.

4.1.15 Meer kleurenschema's

- CMY: Cyan, Magenta, Yellow (4.2.4)
- HSL: Hue, Saturation, Lightness
- HSV: Hue, Saturation, Value
- HSI: Hue, Saturation, Intensity
- HCI: Hue, Chroma, Intensity
- HVC: Hue, Value, Chroma
- HSD: Hue, Saturation, Darkness

Nog meer kleurmodellen bij Kleurmodellen in video's.

4.2 Kleurmodellen in afbeeldingen

4.2.1 RGB-kleurmodel voor monitors

4.2.2 Multi-sensor-cameras

Kan meer zien dan enkel RGB, bijvoorbeeld ook bijna-infrarood.

4.2.3 Camera-afhankelijke kleur

Naast RGB wordt vaak ook **HSV** en **sRGB** gebruikt. sRGB is een standaard voor RGB waarop andere apparaten zich baseren, dit is ook op het web geïmplementeerd.

4.2.4 Subtractieve kleuren: CMY-kleurmodel

- **Additief:** twee lichtstralen schijnen op hetzelfde zwarte oppervlak en hun kleuren worden bij elkaar opgeteld.
- **Subtractief:** bv. op papier, gele inkt doet blauw weg maar weerkaatst rood en groen, dus het resultaat is geel. We willen hier geen kleuren toevoegen, maar wel kleuren wegnemen. Zwart is dus niet $(0, 0, 0)$, maar $(1, 1, 1)$.

4.2.5 RGB $\Rightarrow$ CMY transformatie

$(C, M, Y) = (1 - R,\ 1 - G,\ 1 - B)$

4.2.6 Onderkleur verwijderen: CMYK

Scherpere en goedkopere printerkleuren: bereken het stuk van CMY dat zwart zou zijn, verwijder dat, en vervang het met het echte zwarte inktpatroon.
$K = \min\{C, M, Y\}$
$(C - K, M - K, Y - K)$

4.2.7 Printergamuts

Door twee factoren beïnvloed:

1. **Transmittantie:** Inkt moet een bepaalde kleur blokkeren en de andere doorlaten, maar de perfecte inkt bestaat niet. Cyane inkt blokkeert namelijk rood maar ook een klein deel groen, dit wordt crosstalk genoemd.
2. **Drager:** Verschil tussen glanzend papier en krantenpapier.

4.3 Kleurmodellen in video's

4.3.1 Videokleurtransformaties

Analoge TV-signalen worden in elk continent met een verschillende matrixtransformatie verzonden. Digitale video gebruikt meestal YCbCr (gerelateerd aan YUV).

4.3.2 YUV-kleurmodel

Gebruikt in PAL (bv. Europa). Chrominantiewaarden U en V zijn het verschil tussen de kleur en een wit referentievlak, met dezelfde lichtintensiteit (Y). Grijze pixels zorgen voor $(U, V) = (0, 0)$.

4.3.3 YIQ-kleurmodel

Gebruikt in NTSC (bv. Noord-Amerika). Zelfde principe als YUV, maar I en Q zijn de chrominanties. Chrominantiesignalen zijn geroteerde U- en V-signalen omdat er geen rekening wordt gehouden met de kleurgevoeligheid van het menselijk oog.

4.3.4 YCbCr-kleurmodel

Gebruikt in JPEG- en MPEG-compressie, transformatie werd empirisch bepaald.

Hoofdstuk 5

Fundamental concepts in video

5.1 Analoge video

Analoog signaal $f(t)$ samplet een beeld dat varieert in de tijd, twee manieren.

Progressieve scanning

Alle lijnen worden in volgorde uitgeschreven.

Interlaced scanning

Beeld opgedeeld in twee velden: oneven en even lijnen. Eerst worden alle oneven lijnen uitgeschreven, dan alle even lijnen. Menselijk oog merkt hier door de snelheid niets van.

- **Horizontale retrace:** Tijd die nodig is om van rechterkant van scherm terug naar links (het begin) te gaan bij het schrijven van een nieuwe lijn.
- **Verticale retrace:** Tijd die nodig is om van onderkant van scherm terug naar boven (het begin) te gaan bij het schrijven van een nieuw veld.

De kleur wordt bepaald door de spanning. Bij een retrace wordt een synchronisatiepuls gegeven, bijvoorbeeld door een negatief voltage.

5.1.1 NTSC-video

- National Television System Committee
- Vooral USA en Japan
- 525 scanlijnen, 30 fps, 4:3 breedte/hoogte-ratio
- YIQ-kleurmodel
- Bandbreedte: 6 MHz

Framerate is eigenlijk 29.97 fps in plaats van 30, want het audiosignaal bevindt zich te dicht bij het kleursignaal en dit zou voor interferentie zorgen.

5.1.2 PAL-video

- Phase Alternating Line
- West-Europa, China, India, nog veel plaatsen
- 625 scanlijnen, 25 fps, 4:3 breedte/hoogte-ratio
- YUV-kleurmodel
- Bandbreedte: 8 MHz

Om de kwaliteit van de beelden te verbeteren kunnen de chromasignalen afwisselende tekens hebben (bvb $+U$ en $-U$), vandaar de naam PAL.

5.1.3 SECAM-video

- Système Electronique Couleur Avec Mémoire
- Frankrijk, Afrika, Rusland
- $\approx$ PAL, ook YUV maar verschilt in kleurcoderingsschema

Verschil met PAL: Kleurschema werkt anders, de U- en V-signalen worden op een andere frequentie gemoduleerd en per scan lijn wordt ofwel de U ofwel de V verzonden, niet allebei samen.

5.2 Digitale Video

Overzicht

Video kan op elk moment bewerkt worden, geïntegreerd worden in applicaties, niet-lineaire toegang, encryptie.

Soorten frames

- **I-frames:** Onafhankelijk van andere frames.
- **P-frames:** Houdt enkel de verschillen bij tussen het huidige frame en het vorige frame. Bijvoorbeeld wanneer een auto voorbij een stilstaande achtergrond rijdt, zouden enkel de bewegingen van de auto worden opgeslagen.
- **B-frames:** Houdt nog minder bij dan P-frames, door ook rekening te houden met nog eerdere frames en zelfs volgende frames. P-frames kijken namelijk enkel naar het frame er juist voor.

Een sequentie van I, B, P beelden wordt een **GOP** genoemd (Group of Pictures). Een GOP begint logischerwijs altijd met een I-beeld.

Resoluties

- HDTV: 1920×1080
- Digital Cinema (4K): 4096×2160
- Super Hi-Vision (8K): 7680×4320

Bij elke resolutie hoort een bijhorende afstand (**visual acuity**) en **viewing angle**, een hoek waaronder het scherm moet worden bekeken voor de optimale kwaliteit. Vaak heeft een hogere resolutie ook een groter gamut, dus betere kwaliteit op alle vlakken.

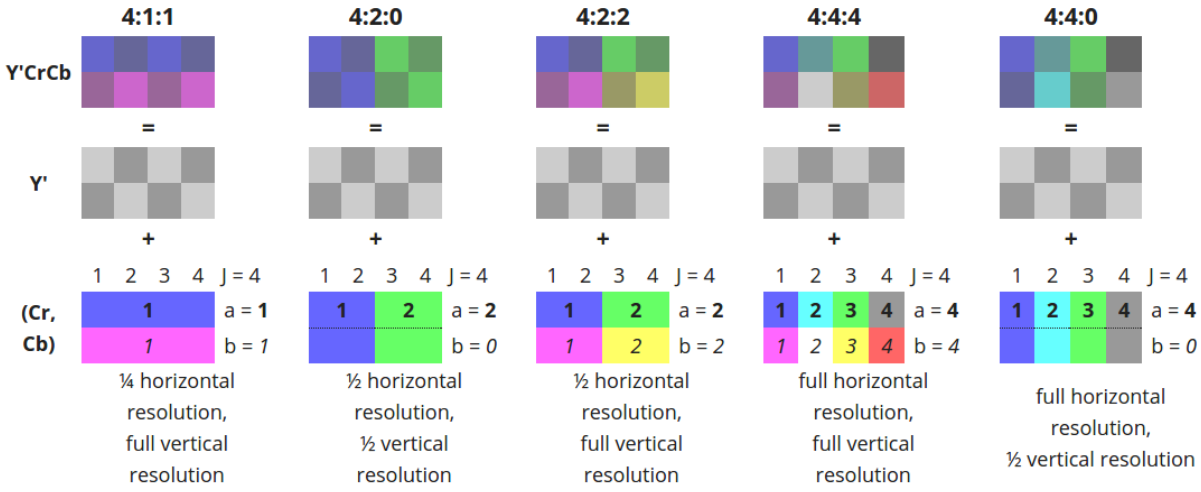
Interlacing-problemen

1. **Combing:** Bijvoorbeeld bij een wiel van een rijdende auto. Het autowiel is dan zichtbaar op twee plaatsen door de snelheid.
2. **Twittering:** Bijvoorbeeld als een persoon een T-shirt aanheeft met zeer fijne horizontale strepen. Door de interlacing kan het lijken alsof deze strepen bewegen, dit wordt opgelost met een low-passfilter.

5.2.1 Chromaonderbemonstering

Mensen kunnen minder makkelijk kleuren van elkaar onderscheiden dan zwart en wit van elkaar, dus gebruik minder kleurinformatie. Dit heet **chromaonderbemonstering** (Eng.: chroma subsampling). Volgend schema duidt aan hoeveel pixels er per vier originele pixels worden verzonden:

- **4:4:4:** Geen chromaonderbemonstering, per 4 originele pixels hebben er 4 Y-, 4 Cb-, en 4 Cr-informatie.
- **4:2:2:** Horizontale onderbemonstering met factor 2. Per 4 originele pixels hebben er 4 Y-, 2 Cb-, en 2 Cr-informatie.
- **4:1:1:** Horizontale onderbemonstering met factor 4. Per 4 originele pixels hebben er 4 Y-, 1 Cb-, en 1 Cr-informatie.
- **4:2:0:** Horizontale en verticale subsampling met factor 2. Vaak gebruikt in JPEG/MPEG.



5.2.2 CCIR- en ITU-R-standaarden

- **CCIR:** Consultative Commmittee for International Radio. Digitale component-video is CCIR-601, nu ITU-R-601.
- **CIF:** Common Intermediate Format, standaardformaat voor lage bitrates.

5.2.3 High Definition TV (HDTV)

Compressie belangrijk, want ongecomprimeerd past het signaal niet binnen de huidige bandbreedtes. MPEG-2 wordt gebruikt voor videocompressie, AC-3 voor audiocompressie (ondersteuning voor 5.1 Surround Sound).

Verschillen tussen TV en HDTV

- Aspect ratio van HDTV is 16:9 $\Leftrightarrow$ 4:3 voor TV
- HDTV gaat richting progressive scan, want interlacing zorgt voor gekorrelde randen bij bewegende objecten. Hierdoor gaat de framerate ook omhoog richting 60 fps.

5.2.4 Ultra High Definition TV (UHDTV)

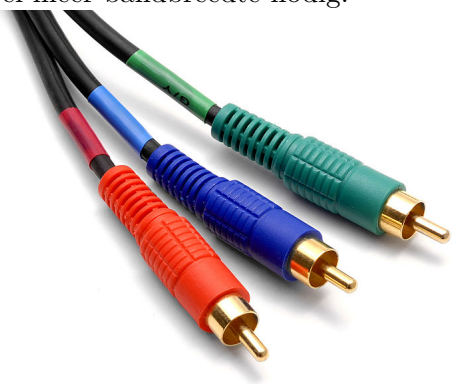
- Resolutie is 4K, 8K
- Aspect-ratio is 16:9, chroma subsampling 4:2:0 of 4:2:2
- 120 fps

5.3 Video Display Interfaces

5.3.1 Analoge interfaces

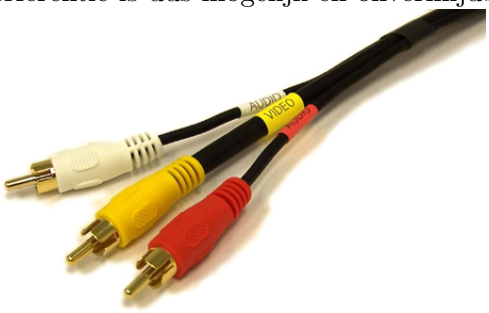
Component Video

3 aparte videosignalen (R,G,B). Beste kleurreproductie want crosstalk is niet mogelijk, maar wel meer bandbreedte nodig.



Composite Video

2 audiokanalen, 1 videokanaal. Chrominantie en luminantie zitten in 1 signaal gemixt. Interferentie is dus mogelijk en onvermijdbaar.



S-video

1 pin wordt gebruikt voor luminantie, een andere pin voor voor chrominantie (composiet), rest zijn aardingspinnen. Minder crosstalk tussen grijswaarden en kleur, dit is goed want mensen zijn gevoeliger voor verschillen tussen zwart/wit dan tussen kleuren.

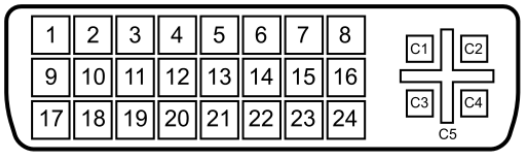
VGA: Video Graphics Array

Veelgebruikt voor computerindustrie. Analoge componenten R, G, B, HSync, VSync. Lange kabels zorgen voor interferenties.

5.3.2 Digitale interfaces

DVI: Digital Visual Interface

Zowel een analoge, digitale, als gecombineerde variant. Vooral gebruikt voor computerschermen, game consoles, DVD-spelers, ...



SDI: Serial Digital Interface

Ongecomprimeerde digitale videosignalen, via BNC stekkers (coax-kabels).

HDMI: High-Definition Multimedia Interface

Ongecomprimeerde digitale data, kan snelheden tot 14 Gbps halen.

DisplayPort

Thunderbolt-gebaseerd, verstuurt data in pakketten (zoals USB/Ethernet). Snelheden tot 26 GBps, kan 8K aan en heeft een ingebouwde (visueel) lossless compressie.

5.4 3D Video en TV

5.4.1 Factoren

Monoculaire factoren

Vervagen, perspectief, relatieve grootte, occlusie, parallax.

- **Occlusie:** Als 2 objecten gedeeltelijk achter elkaar staan, verberg dan het deel van het achterste object dat verborgen wordt door het voorste.

- **Parallax:** Objecten op de voorgrond bewegen sneller dan objecten op de achtergrond, bijvoorbeeld wanneer er met een auto door een landschap wordt gereden.

Binoculaire factoren

Menselijk zicht is binoculair (stereo) want 2 ogen. Linker en rechteroog staan gemiddeld ongeveer 65mm uit elkaar (= **Interoculaire afstand**). Linker en rechteroog zien allebei iets anders, het verschil hiertussen is de **dispariteit** en is afhankelijk van de afstand van een object tot beide ogen, wat zorgt voor een 3D effect.

5.4.2 3D Camera's

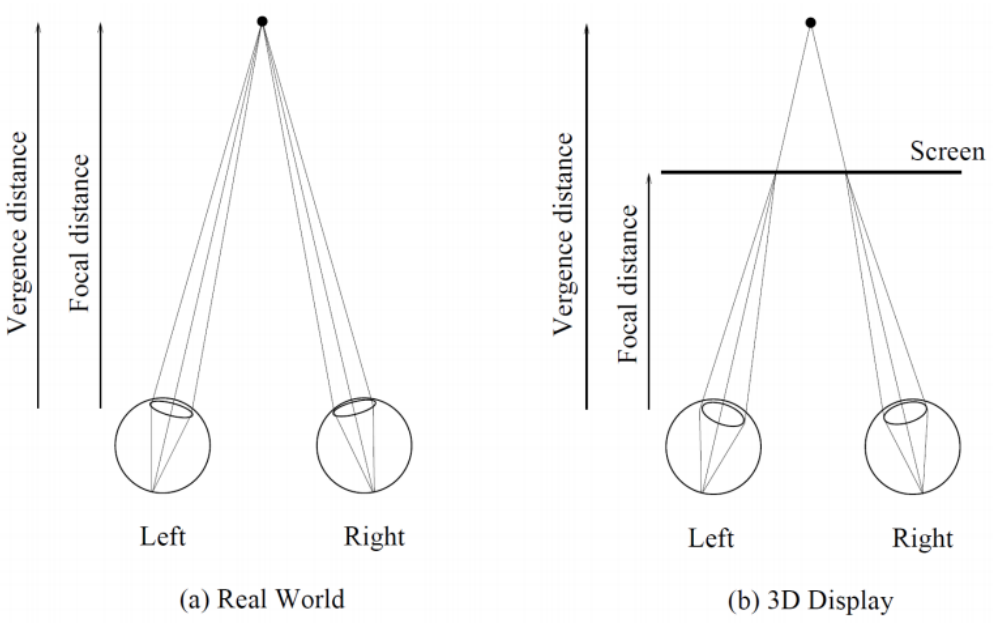
1. **Stereo Camera:** Linker en rechtercamera zijn identiek en staan op eenzelfde as. Door naar hetzelfde object te kijken wordt een horizontale parallax gecreëerd.
2. **Toed-in Stereo Camera:** Lijkt op menselijk oog, camera's staan op dezelfde as en focussen op dezelfde manier als het menselijk oog dat recht voor zich uitkijkt: de linkercamera draait naar rechts en de rechtercamera naar links, om zo 1 enkel binoculair zicht te maken (= **vergence**).

5.4.3 3D Films en TV gebaseerd op Stereo

1. **3D brillen:** Complementaire kleuren, vaak rood-cyaan.
2. **Gepolariseerde brillen:** Laat 1 van de 2 gepolariseerde beelden door terwijl het het ander blokkeert.
3. **Shutter brillen:** Bril wordt gesynchroniseerd met de TV die afwisselend linker en rechterbeelden toont. Door bril onder stroom te zetten wordt de kristallaag op de bril doorzichtig (zoals een camerashutter).

5.4.4 Vergence-Accomodation conflict

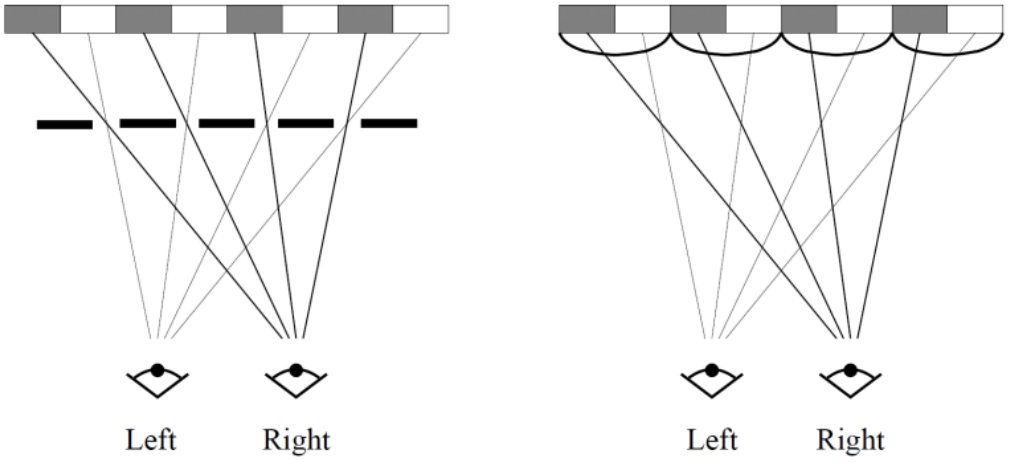
Accomodatie: een scherp beeld behouden wanneer afstand tot object verandert. Bij 3D video is dit niet zo, waardoor misselijkheid kan optreden.



(a) Real World (b) 3D Display

5.4.5 Autostereoscopisch (zonder glazen)

- **Parallax:** Een laag ondoorzichtig materiaal (met doorzichtige gleufjes) wordt voor de LCD geplaatst.
- **Lenticulaire lens:** Vergrootglazen worden voor de LCD geplaatst zodat lichtstralen correct naar linker en rechteroog worden gestuurd.



(a) Parallax Barrier (b) Lenticular Lens

5.4.6 Dispariteitsmanipulatie

- **Bereik:** Originele dispariteiten naar een bereik omzetten dat comfortabel is voor mensen.
- **Gevoeligheid:** Menselijk zicht kan beter dieptes onderscheiden wanneer ze dichterbij zijn, dus onderdruk het verre bereik.
- **Gradiënt:** Menselijk zicht kan niet goed gradiënten zien in de diepte.
- **Snelheid:** Mensen kunnen niet snel zeer grote accomodatie/vergence veranderingen verwerken, dus vertraag die.

Hoofdstuk 6

Basics of digital audio

6.1 Digitalisatie van geluid

6.1.1 Wat is geluid?

Golfverschijnsel: lucht die samengedrukt wordt en uitzet. Bv.: een luidspreker vibreert heen en weer en maakt een longitudinale golf. Een analoog geluidssignaal wordt voorgesteld door een voltage i.f.v. tijd.

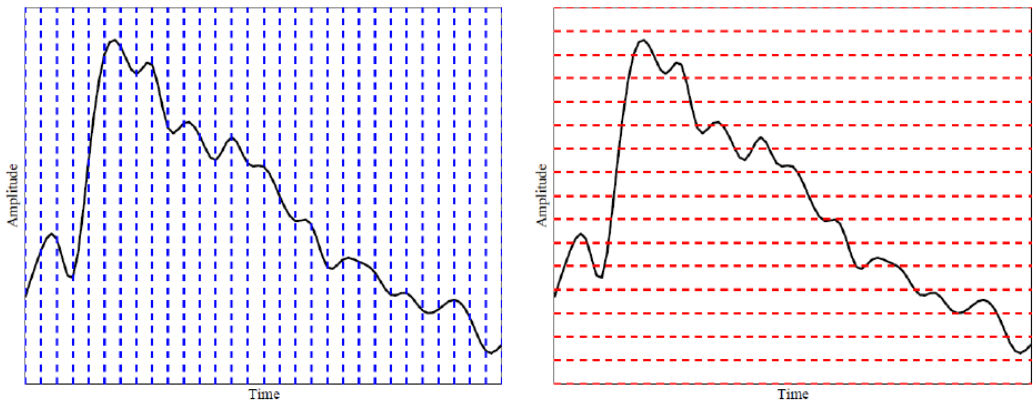
Een signaal kan ontbonden worden in een lineaire combinatie van **sinusfuncties**.

- **Toonhoogte** is een relatieve indruk van frequentie (typische stemming voor la is 440 Hz).
- Een **octaaf** is het interval bij het verdubbelen van de frequentie, in ons muzieksysteem krijg je dan een toon met dezelfde naam (440 Hz is la, 880 Hz ook).
- **Harmonische** tonen: gehele veelvouden van frequentie: (110 Hz, 220 Hz, 330 Hz, 440 Hz).
- **Boventonen** worden bereikt met niet-gehele veelvouden van de grondtoon.

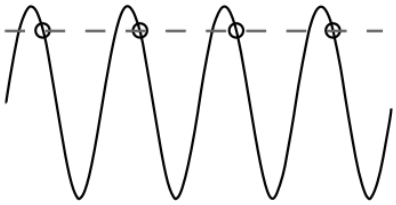
6.1.2 Digitalisatie

Samplen in ...

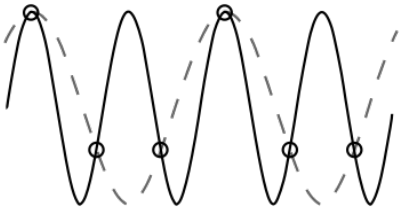
- ... **tijd**. Dit heet **sampling** (tout court) en gebeurt aan een bepaalde samplingfrequentie.
- ... **amplitude**. Dit heet **kwantisatie** en gebeurt met een bepaalde fijnheid. Het kan uniform of niet uniform.



6.1.3 Stelling van Nyquist



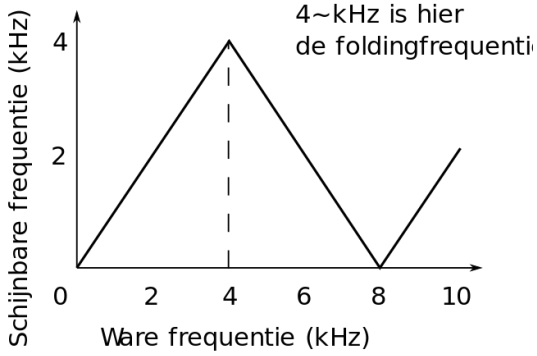
samplen aan invoerfrequentie: constante frequentie



1,5 keer de frequentie: verkeerd resultaat: **aliasing**

Stelling van Nyquist[1]: als een signaal **band-limited** is met minimumfrequentie f_l en maximumfrequentie f_h, dan moet de samplerate ten minste $2 \cdot (f_h - f_l)$ zijn. Dit heet de **Nyquistrate**.

Als een frequentie te hoog wordt, vouwt hij bij de sampling als het ware terug. Dit heet **folding**. Voorbeeld hiervan bij samplen aan 8 kHz:



De **Nyquistfrequentie** is de helft van de Nyquistrate. Hogere frequenties kunnen sowieso niet bereikt worden. Op invoer wordt voor het samplen een **antialiasing-filter** gebruikt die het tot die frequentie beperkt.

De **samplingstelling** zegt: een signaal is uniek bepaald door een sampling $\Longleftrightarrow$ het gaat niet boven de Nyquistfrequentie.

6.1.4 SNR: signal to noise ratio

Signal to noise ratio (SNR) is een maat voor geluidskwaliteit. $SNR = 10 \cdot \log_{10}\left(\frac{V^2_{signaal}}{V^2_{ruis}}\right)$ in dB, waarin V het voltage is.

[1] Lees: *niekvist*, Nyquist was een Zweed.

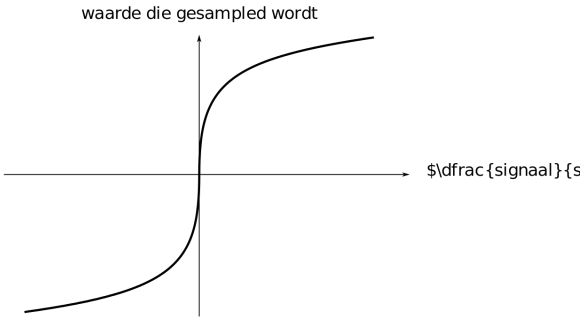
6.1.5 SQNR: signal to quantization noise ratio

Kwantisatieruis is niet echt ruis maar afrondingsfouten in de kwantisatie. De formule voor **signal to quantization noise ratio** (SQNR) is analoog aan SNR maar met $V^2_{kwantisatieruis}$ in de noemer. Als we de piekwaarden (pieksignaal en piekruis) invullen, krijgen we de peak SQNR (**PSQNR**). Bij kwantisatie aan N bits per sample, geldt in het slechtste geval $PSQNR \approx 6 \cdot N(\text{dB})$. Statistisch is het meestal zo dat de $PSQNR \approx 6 \cdot N + 1.75(\text{dB})$.

Dynamisch bereik (Eng.: dynamic range) is $\frac{V_{max}}{V_{min}}$. V_{min} is het kleinste voltage dat niet door ruis gemaskeerd wordt.

6.1.6 Lineaire en niet-lineaire kwantisatie

- **Lineair**: uniforme verdeling van niveaus.
- **Niet-lineair**/niet-uniform: fijnere onderverdeling bij meest gevoelige gebieden in menselijk gehoor. Twee formules hiervoor zijn de μ- en A-wet.



Dit wordt bv. gedaan door uniform aan 16 bit te samplen, dan te transformeren met de μ-wet en dan te downsamplen naar 8 bit.

6.1.7 Audiofiltering

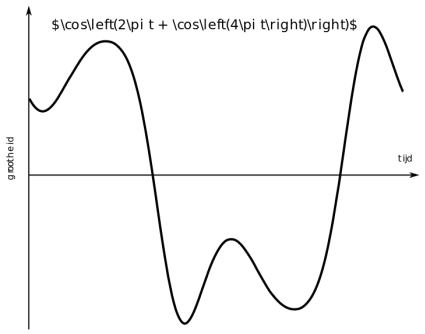
Voor het samplen wordt vaak gefilterd. Wordt behouden:

- spraak: 50 Hz – 10 000 Hz.
- muziek: 20 Hz – 20 000 Hz.

Na digitalisatie ontstaan er terug hoge frequenties, dus bij het decoden wordt een lowpassfilter gebruikt.

6.1.9 Synthetisch geluid

- **Frequentiemodulatie** (FM) is bekend van radio, maar kan ook gebruikt worden om geluid te synthetiseren. Bv. een cosinus in een cosinus:



- **Samplingsynthese**[2] is opnames (samples) van instrumenten – eventueel bewerkt – te combineren.

6.2 MIDI: Musical Instrument Digital Interface

MIDI is een scriptingtaal die gebeurtenissen opslaat die dan gebruikt kunnen worden om geluid te produceren. Gebeurtenissen kunnen bijvoorbeeld per noot toonhoogte, duur en volume bevatten.

6.3 Kwantisatie en transport van audio

6.3.1 Codering van audio

Kwantiseren en transformeren van data heet samen **coderen** (Eng.: coding).

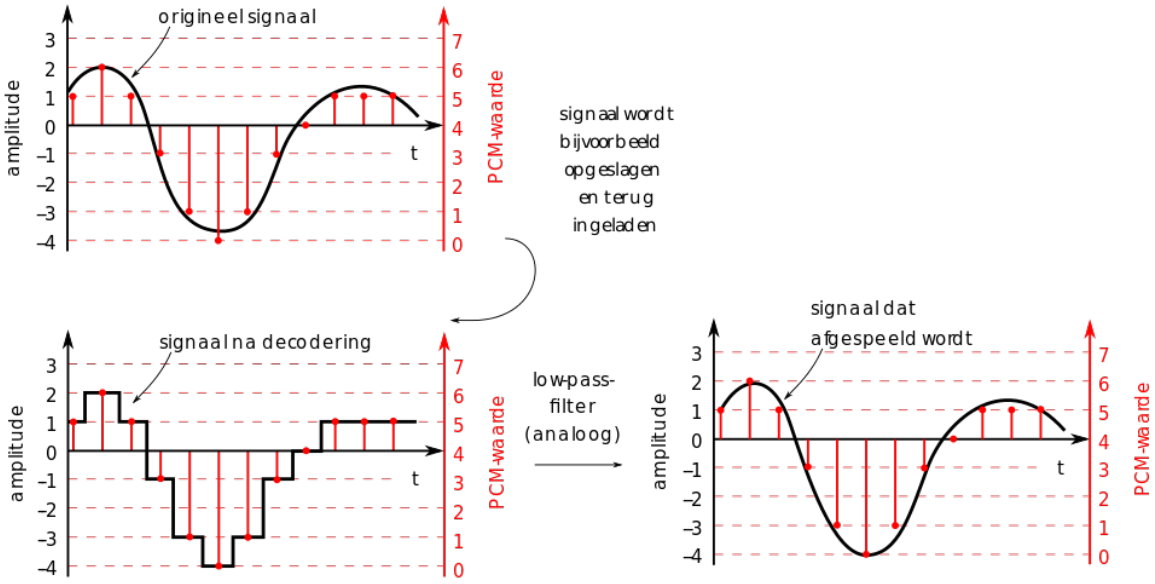
Na digitalisatie willen we de data comprimeren. Elk compressieschema heeft drie fasen:

- **transformatie** naar een gemakkelijker of efficiënter te comprimeren formaat
- informatie **verliezen**. Kwantisatie is de hoofdbron van verlies
- **coderen**: een codewoord toekennen aan elk uitvoerniveau[3]

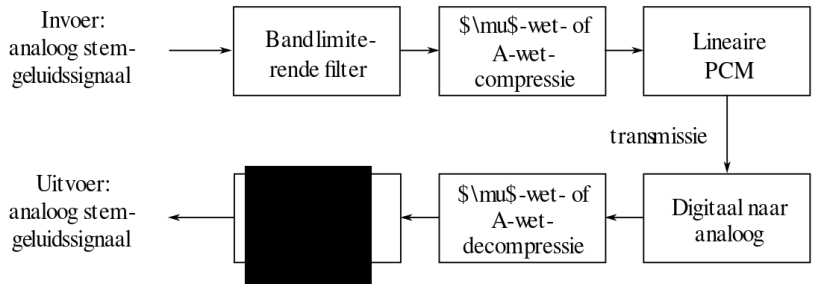
6.3.2 PCM: pulscodemodulatie

Pulscodemodulatie (PCM) is eigenlijk gewoon voor elke sample de waarde als een integer opslaan. Na het demoduleren wordt een analoge low-passfilter toegepast.

[2] Let op, in het boek wordt dit verward met wavetable-synthese! Wikipedia
[3] Ja, hierboven staat een andere definitie voor coderen. Geen idee wat ze bedoelen.



Bij telefonie kan het er bijvoorbeeld zo aan toe gaan:



6.3.3 Differentieel coderen van audio

Differentieel coderen van audio is telkens het verschil met de vorige waarde opslaan, in plaats van de waarde zelf.

6.3.4 Verliesloos voorspellend coderen

Verliesloos voorspellend coderen (Eng.: lossless predictive coding) is een eenvoudige voorspelling van de waarde maken, en enkel het verschil daarmee opslaan. Typisch wordt een lineaire combinatie van enkele voorgaande waarden gebruikt.

Aangezien de waarden die opgeslagen moeten worden vaker klein dan groot zullen zijn, maar we toch grote waarden moeten aankunnen om het dynamisch bereik niet te beperken, kan het volgende systeem gebruikt worden. Voorzie een beperkt aantal codes voor kleine getallen, en twee extra codes, voor Shift-Up en Shift-Down, die de waarde een interval doen verspringen.

6.3.5 DPCM: differentiële PCM

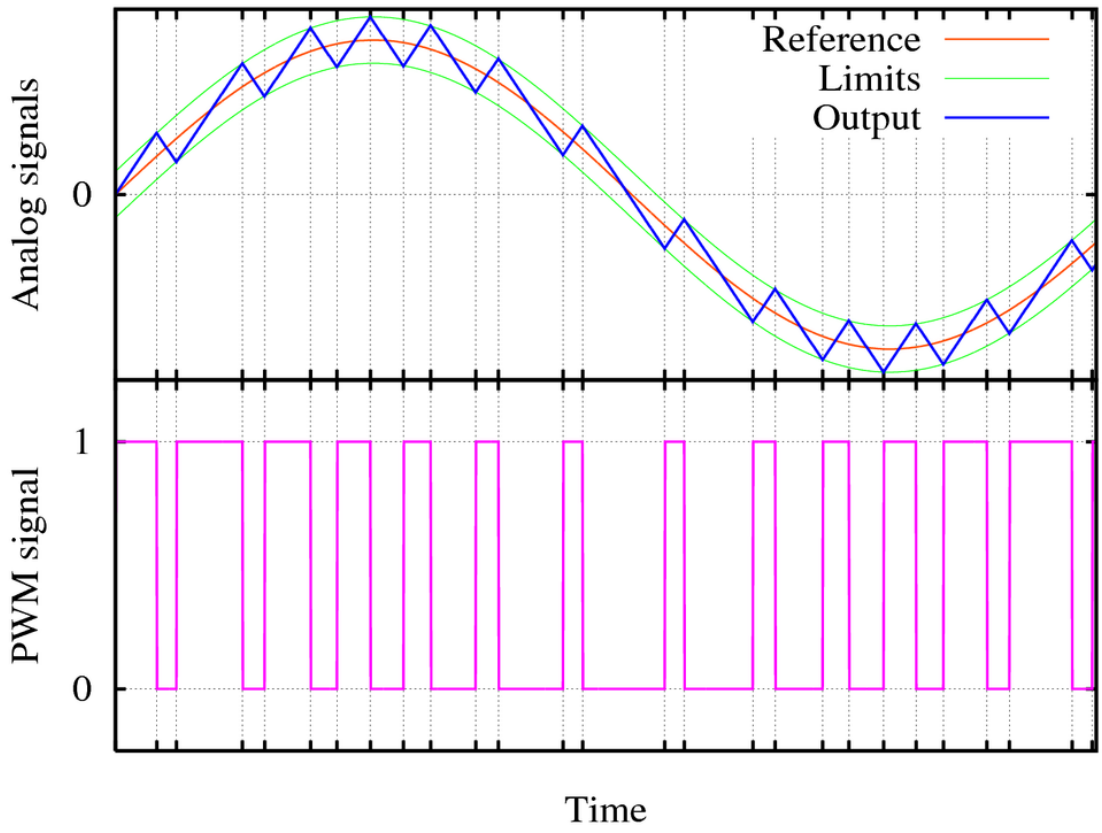
Differentiële PCM (DPCM) is hetzelfde als voorspellend coderen, maar past nog eens kwantisatie toe op het verschil.

Lloyd-Max-kwantisatie is hier beter voor dan lineaire kwantisatie. Het is gebaseerd op een kleinstekwadratenminimalisatie van de foutterm.

De codewoorden worden opgesteld aan de hand van entropiecodering, bv. Huffmancodering (sectie 7.4.2).

6.3.6 DM: deltamodulatie

Deltamodulatie (DM) is een versimpelde versie van DPCM en wordt gebruikt als snelle analoog-digitaalomzetter. Het verschil wordt opgeslagen in één bit. Ze is dus telkens dalend of stijgend, nooit nul. Om een deftige SNR te behalen, moet er oversampled worden.



Adaptieve deltamodulatie (ADM): verander stapgrootte adaptief.

6.3.7 ADPCM: adaptieve differentiële PCM

Bij **adaptieve differentiële PCM** (ADPCM):

- veranderen we de stapgrootte en de beslissingsgrenzen adaptief.
  Dat kan voorwaarts (a.d.h.v. invoersignaal) of achterwaarts (bijsturen als de fouten te groot worden).
- kunnen we ook de voorspellingsfunctie adaptief bijsturen.

Hoofdstuk 7

Lossless Compression Algorithms

7.1 Introductie

- **Compressie:** Het verminderen van het totaal aantal bits dat nodig is om dezelfde informatie voor te stellen.
- **Lossless:** Geen verlies aan informatie.
- **Lossy:** Wel verlies aan informatie, maar vaak ook sterker juist hierdoor.
- **Compressieratio:** $ratio = \frac{bits\ voor\ compressie}{bits\ na\ compressie}$

7.2 Basis van Informatietechnologie

Entropie (η) van een informatiebron met alfabet $S = \{s_1, s_2, \ldots, s_n\}$:

$$\eta = H(S) = \sum_{n=1}^{n} p_i \log_2 \left(\frac{1}{p_i}\right)$$

met p_i de kans dat symbool s_i voorkomt in S.
Voorbeeld: Uniforme verdeling, $\forall i.\ p_i = \frac{1}{256} \Rightarrow \eta = \log_2(256) = 8$

Entropie en codeerlengte

Entropie η is een gewogen som van termen $\log_2 \frac{1}{p_i}$ en betekent de gemiddelde hoeveelheid informatie per symbool in bron S.

Als $\bar{l}$ het gemiddeld aatal bits per codewoord is, dan geldt $\eta \leq \bar{l}$

7.3 RLC: Run-Length Coding

Memoryless: Informatie wordt onafhankelijk behandeld; de waarde van het huidige symbool is niet afhankelijk van eerder gecodeerde symbolen.

Werking: RLC is niet memoryless, maar zoekt integendeel continue groepen van symbolen in de informatie van een gegeven lengte die geëncodeerd worden.

Voorbeeld: $ABBBCCDDDD \mapsto A1\ B3\ C2\ D4$

7.4 VLC: Variable-Length Coding

7.4.1 Shannon-Fano

Algoritme

1. Sorteer de symbolen op frequentie van voorkomen (links meest voorkomende, rechts minst voorkomende).
2. Verdeel die lijst in 2 delen en zorg dat de som van de frequenties per deel zo gelijk mogelijk is.
3. Geef het linkerdeel van de lijst het binaire cijfer 0, het rechterdeel 1. Alle codes voor symbolen in het linkerdeel zullen dus starten met een 0, die in het rechterdeel met een 1.
4. Herhaal stappen 2 en 3 recursief tot er een boom ontstaat met allemaal aparte symbolen.

Voorbeeld-resultaat voor HELLO

Symbool	Frequentie	$\log_2$	Code	Aantal bits gebruikt
L	2	1.32	00	4
H	1	2.32	01	2
E	1	2.32	10	2
O	1	2.32	11	2

7.4.2 Huffman Coding

Algoritme

1. Steek alle symbolen in een lijst, gesorteerd op frequentie.
2. Herhaal tot er maar 1 element meer in de lijst zit:
   (a) Neem de 2 elementen uit de lijst met de laagste frequentie en vorm een nieuwe Huffman sub-boom. De wortel van deze boom heeft als waarde de som van de frequenties van zijn 2 kinderen.
   (b) Plaats die Huffman sub-boom in de lijst.
   (c) Verwijder de 2 kinderen van stap 2a uit de lijst.
3. Geef elk blad van de boom een codewoord, op basis van het pad vanaf de wortel. Naar links gaan is een 0 toevoegen, naar rechts een 1 (zie Shannon-Fano).

Voorbeeld voor HELLO - Inhoud van de lijst

1. L, H, E, O
2. L, P1, H
3. L, P2
4. P3

Eigenschappen

1. **Unieke prefix:** De waarden die toegekend zijn aan de symbolen zijn uniek, dit voorkomt dubbelzinnigheden bij het decoderen.
2. **Optimaal:** De gemiddelde codeerlengte voor informatie S bedraagt: $\eta \leq \bar{l} < \eta + 1$

Extended Huffman Coding

Reden: Alle codewoorden in Huffman coding hebben gehele bit lengtes, voor grote strings kan dit ervoor zorgen dat de entropie afneemt.
$\Rightarrow$ **Oplossing:** Sommige symbolen samen groeperen en die groep een codewoord geven.

Extended alfabet: Voor een alfabet $S = s_1, s_2, \ldots, s_n$, als k symbolen gegroepeerd worden dan is het extended alfabet alle herhalingsvariaties van lengte k uit het oorspronkelijke alfabet. De lengte van het nieuwe alfabet $S^{(k)}$ is dus n^k.

7.4.3 Adaptieve Huffman coding

Algoritme

1. Op voorhand wordt aan elk symbool een code toegekend, zonder de frequentie te kennen.
2. Herhaal zolang er symbolen in de lijst zijn:
   (a) Encodeer het symbool volgens de op voorhand afgesproken code.
   (b) Incrementeer de frequentie-teller van dit symbool.
   (c) Herschik de boom.

Opmerking: Tijdens het coderen verandert de code van een symbool, omdat de boom herschikt wordt.

7.5 LZW: Dictionary-gebaseerde coding

Codewoorden met vaste lengte gebruiken om variabele-lengte strings te encoderen. LZW encoder en decoder bouwen dezelfde dictionary dynamisch op als ze de te coderen data ontvangen.

7.5.1 Algoritme

```
1 s = input()
2 dictionary = []
3
4 while input:
5   c = input()
6   if s + c in dictionary:
7     s = s + c
8   else:
```

```
9       print dictionary[s]
10      dictionary[s+c] = new_code()
11      s = c
12  print dictionary[s]
```

7.5.2 Voorbeeld voor ABABBABCABABBA

S	C	Output	Code	String
			1	A
			2	B
			3	C
A	B	1	4	AB
B	A	2	5	BA
A	B			
AB	B	4	6	ABB
B	A			
BA	B	5	7	BAB
B	C	2	8	BC
C	A	3	9	CA
A	B			
AB	A	4	10	ABA
A	B			
AB	B			
ABB	A	6	11	ABBA
A	EOF	1		

Output: 1 2 4 5 2 3 4 6 1. In plaats van 14 karakters moeten er dus maar 9 codes verstuurd worden (compressieratio: $14/9 = 1.56$)
Code lengte l wordt meestal in een interval $[l_0, l_{max}]$ gehouden, wanneer l_{max} wordt bereikt, wordt de dictionary geflusht of wordt een **Least Recently Used** algoritme gebruikt.

7.6 Arithmetische coding

Encodeer geen losse woorden, maar encodeer de volledige boodschap als tot een getal in een interval $[0.0 \leq n < 1.0)$ Grenzen van het interval worden steeds ingekort naarmate de lengte van de boodschap groter wordt. Aan elk symbool uit de boodschap wordt een probabiliteit toegekend waarmee de grenzen worden aangepast.

7.6.1 Algoritme

In dit voorbeeld kunnen 2 letters geëncodeerd worden, namelijk A en B. A krijgt een hogere probabiliteit (60%) dan B (40%).

```
ranges = {"A": [0, 0.6], "B": [0.6, 1.0]}
low = 0.0
high = 1.0
range = 1.0

while input:
  symbol = input()
  low = low + range * ranges[symbol][0]
  high = low + range * ranges[symbol][1]
  range = high - low
print x zodat low <= x < high
```

7.6.2 Incremental coding

Probleem: Voor lange boodschappen is enorm hoge precisie vereist omdat de intervallen te klein worden. Bovendien wordt er pas een codewoord gegenereerd wanneer de volledige boodschap geëncodeerd is.
Opmerking: De meest significante bits van de intervallen zijn altijd gelijk voor zeer kleine intervallen. Volgende intervallen vallen altijd binnen het huidige interval (want het huidige interval verkleint), dus we kunnen de gemeenschappelijke meest significante bits printen en verwijderen van volgende outputs.

E1 en E2 scaling

```
while high <= 0.5 or low >= 0.5:
  if high <= 0.5:
    # E1 scaling
    output 0
    low = 2 * low
    high = 2 * high
  else
    # E2 scaling
    low = 2 * (low - 0.5)
    high = 2 * (high - 0.5)
```

7.6.3 Integer implementatie

Vervang simpelweg het interval $[0, 1.0)$ door een interval $[0, N)$ met N een integer, bijvoorbeeld 255.

7.6.4 Binaire Arithmetische coding

Gebruik enkel de symbolen 0 en 1, verloopt veel sneller.
Adaptive Arithmetische coding: Pas de ranges van de symbolen aan na elk gecodeerd symbool. **Context-Adaptieve Arithmetische coding:** Op basis van conditionele kans.

7.7 Lossless Beeldcompressie

7.7.1 Algemeen

Difference image: Beeld $I(x, y) \Rightarrow d(x, y) = I(x, y) - I(x - 1, y)$
Difference image zal een minder entropie hebben en een kleiner histogram, door de spatiale redundantie in de originele afbeelding.

7.7.2 Lossless JPEG

Predictive manier

1. **Differentiële voorspelling:** Waarde van maximaal 3 aangrenzende pixels combineren tot de voorspelde waarde van de huidige pixel.
2. **Encoding:** Voorspelling vergelijken met de echte waarde van die pixel, en dat met bijvoorbeeld Huffman encoderen.

Appendix: Fourier for dummies

7.8 Inleiding

Dit is niet bedoeld als wiskundige definitie, redenering,... maar eerder om een soort van 'feeling' te krijgen met wat fouriertransformatie is en wat het met signalen doet. Fouriertransformatie is onlosmakelijk verbonden met signalen. Vandaar beginnen we eerst met een definitie van een signaal.

7.9 Signalen

Een signaal kan worden gedefinieerd als een functie van de tijd t naar een waardenverzameling W. In de praktijk is dit vaak $\mathbb{R}$ of $\mathbb{C}$.

$$g : \mathbb{R} \to W, \quad t \to g(t)$$

Deze functie geeft dus voor elke $t \in \mathbb{R}$ aan welke waarde het signaal aanneemt op dat bepaalde tijdstip t. Het idee van Fourier is dat dit niet de enige manier is om het signaal te karakteriseren.

7.10 Fourieranalyse

Het idee van Fourier bestaat erin dat elke functie te ontbinden is als een (oneindige) som van complex exponentiële functies of, equivalent daarmee, sinussen en cosinussen. Neem bijvoorbeeld de sinusfunctie:

$$A \sin(\omega t + \phi)$$

hierbij is A de amplitude, ϕ de fasehoek en ω de hoekfrequentie voor. Uit *omega* kunnen we dan de frequentie f halen als volgt:

$$f = \frac{\omega}{2\pi}$$

Wat een fouriertransformatie doet is simpel gezegd het volgende: het bepaalt de verschillende frequentiecomponenten die in het signaal g aanwezig zijn en hoe sterk die aanwezig zijn. Wat daarmee bedoeld wordt is: stel we hebben het frequentiespectrum van een signaal. Dit is de verzameling van alle frequenties en hun amplitudes die in het signaal voorkomen. Willen we nu het oorspronkelijke signaal construeren door deze frequenties dan nemen we de som van alle sinussen en cosinussen met als frequentie en amplitude deze uit het frequentiespectrum. De fouriergetransformeerde levert dus een alternatieve karakterisatie van het signaal op aan de hand van de frequentiecomponenten.

7.11 Conclusie

Het besluit is eenvoudig. Fouriertransformaties helpen ons om signalen op een andere manier te karakteriseren, die in vele toepassingen gemakkelijker is om mee te werken. Dit komt omdat fouriertransformaties enkele zeer nuttige eigenschappen bezitten die rekenwerk enorm kunnen vereenvoudigen. Omdat dit natuurlijk een puur informele en oppervlakkige beschrijving was van wat fouriertransformaties net zijn, ga ik er niet dieper op in.

Meanwhile...