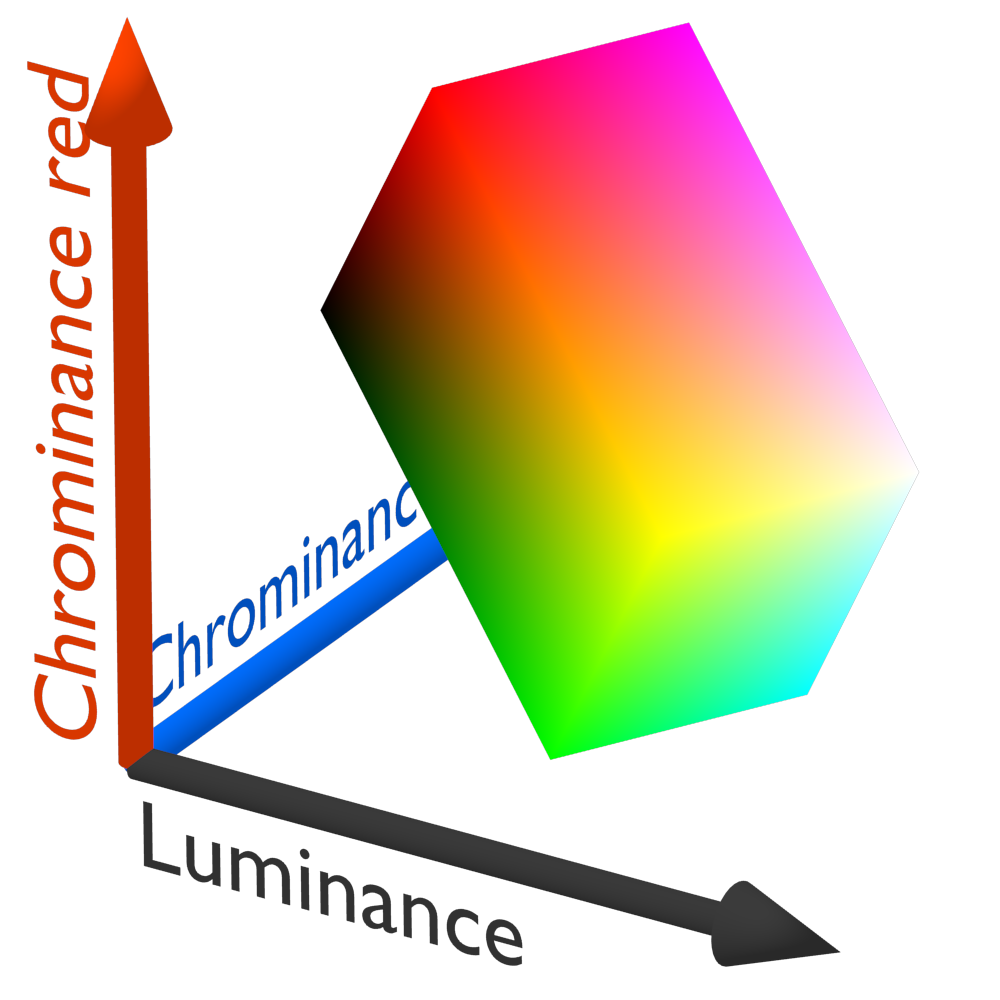
JPEG

JPEG (Joint Photographic Experts Group) er et filformat for lagring av biler. Mens vanlig bildeformat lagrer en verdi for hvert piksel så tar JPEG å komprimerer bildet. Dette resulterer at det tar opp mindre plass enn originalfilen. Det finnes forskjellige versjoner av JPEG. Vi kan få JPEG der bildekvaliteten ikke blir tapt og JPEG der bildekvalitet blir tapt. I utgangspunktet er det ønskelig å bruke JPEG med tap av bildekvalitet. Det er hensiktsmessig fordi litt tap i bildekvalitet kan føre til at man får komprimert bildet betydelig bedre. Den dataen som blir luket ut av bildet har lite eller ikke noe å si på hvordan vi oppfatter slutt produktet.

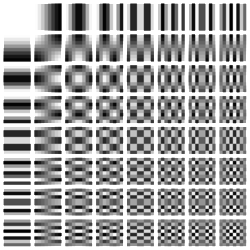
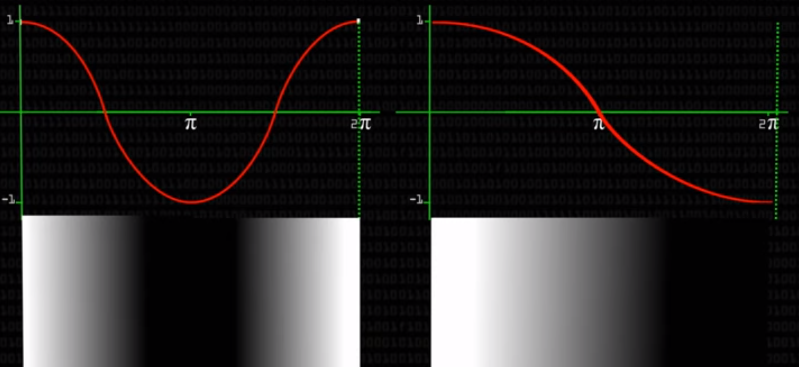
Først blir pikslene gjort om fra RGB til noe som heter YCbCr. YCbCr er en fargemodell slik som RGB. Istedenfor å representere fargen gjennom en fargekombinasjon av rød, grønn og blå så kan YCbCr representere det samme fargespekteret gjennom Y, Cb og Cr. Y er luminans, det er et mål på hvor lyst et område er. Cb og Cr er rødnyanser og blånyanser til bildet. Gjennom disse tre komponentene får vi det samme fargespekteret som RGB gir oss.



Kilde: <https://en.wikipedia.org/wiki/YCbCr> 02.10.17

Grunnen til at vi gjør om fra RGB til YCrCb er fordi at vi kan redusere data fra den sistnevnte uten at det endrer hvordan vi oppfatter det ferdige resultatet. Øynene våre er relativt dårlige på å se forskjell mellom fargenyanser. Dette gjør av vi kan kutte ut en del data fra Cb og Cr uten at det reduserer vårt inntrykk av hvordan vi oppfatter bildet. Ved å fjerne data fra Cr og Cb kutter vi litt av kvaliteten på bildet. Dette er noe av bildekvalitetstapet som vi har med JPEG. Luminans oppfattes av oss mennesker som gråtoner. Vi mennesker mye bedre på å se nyanser av gråtoner enn nyanser av farger. Det blir som å se midt på natta når det er mørkt. Du kan skimte hva som er foran deg, men du klarer ikke å se farger. I YCrCb får vi avgrenset luminanskomponenten, slik at vi får redusert de andre komponentene uten å påvirke luminansen i bildet.

Deretter deler JPEG bildet opp i kvadrater på 8x8 piksler der hver fargekomponent er adskilt. Hver av disse komponentene som er delt opp i 8x8 blokker kan bli representert av Discrete Cosine Transformation (DCT), Diskré Cosinus Transformasjon. Diskré Cosinus Transformasjon er basert på Joseph Fourier sin Fourierrekke. I forskjell fra Fourierrekke bruker DCT kun cosinusledd, cosinus dette er for å slippe å ta stilling til både sinus og cosinus ledd og dermed redusere den totale dataen enda mer. Vi kan se på DCT som følgende matrise (til venstre);

Kilde: <https://youtu.be/Q2aEzeMDHMA?t=4m35s> 02.10.17

Denne matrisen er bygd opp av cosinusledd med forskjellige frekvenser i x planet (horisontalt) og y planet (vertikalt).

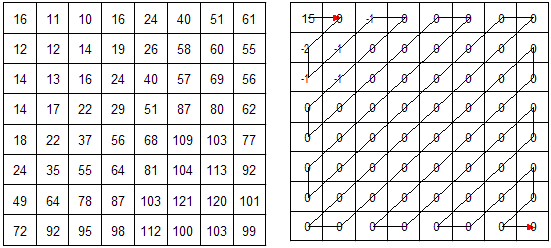
På bildet til høyre ser vi hvordan cosinusleddene representerer matrisen. Amplituden bestemmer nyansen av komplimentene til Y, Cr og Cb. Det er her snakk om komplimentfargene, vi kan se på bildet her at det er luminanskomponenten y. Frekvensen bestemmer hvor ofte det skal byttes mellom komplimentene. Matrisen oppe til venstre kombinerer cosinusleddene i y planet og x planet og gis oss 8 ganger 8 antall kombinasjoner. Disse 64 kombinasjonene legges lagvis for å rekonstruere en 8x8 blokk med bilde. De ulike kombinasjonene trengs i ulik grad for å gjenskape bildet presist. For å bestemme hvor mye hver kombinasjon skal bidra regnes bidraget ut som koeffisienten av fourierrekken i DCT.

DCT matrisen og vektleggingen av kombinasjonene er kun basert på Fourierrekker. Det gjør ved at fourierrekkeformelen a0 +Zigma(an\*cos(n)+bn\*sin(n)) er alt vi trenger for å gjenskape 8x8 blokken. Variablene her er

Videre for å redusere data betraktelig brukes en tabell som heter «Quantized table». Hensikten med kvantisering (quantized) er å fjerne de bidragene vi ikke vil ha, slik som de med veldig høy frekvens og de som har liten betydning for bildet. Dette gjøres fordi vi kan redusere filstørrelsen ved å utelukke disse leddene med uvesentlige detaljer. I tillegg er disse mindre detaljene i bildet vanskelig for mennesker å oppfatte, vårt helhetsinntrykk av bildet vil forbli det samme. En slik kvantiseringstabell er forhåndsbestemt av JPEG og inngår i et regnestykke sammen med koeffisienten som divisor og dividend for å bestemme hvor mye vi velger å vektlegge bidraget. En slik kvantisert tabell ser vi på bildet under til venstre. Legg merke til at verdiene nede til høyre er større enn de oppe til vestre. Dette er fordi det er ønskelig at verdiene nede til høyre veier mindre enn de verdiene oppe til venstre.

Koeffisient / kvantisert = ny data.

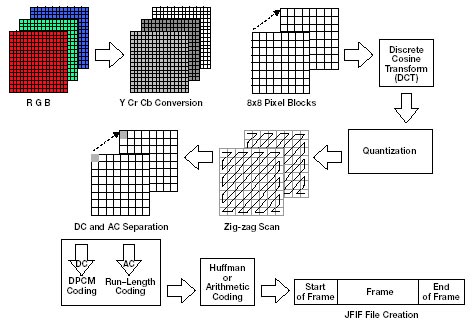
Bidragene etter kalkulasjonen vil være mindre enn de var før. I tillegg vil flesteparten av bidragene ha blitt desimaltall. Disse desimaltallene rundes av til nærmeste heltall. Siden en del av disse bidragene har blitt veldig små så vil de rundes av til null. Det vil bli større representasjon av null nede til høyre enn oppe til venstre ettersom de mindre bidragene i matrisen allerede har blitt dividert med de større kvantiseringsverdiene. Når vi da leser igjennom tabellen sikk-sakk vil de tellende verdiene komme først og etterfulgt av en lang rekke med null. Det er hensiktsmessig å lese igjennom tabellen sikk-sakk ettersom vi får plassert hele rekka med null bakerst og ved siden av hverandre. En slik lang rekke med null kan lett bli komprimert av Huffman koding. Bildet under til høyre viser hvordan dataen til bli lest om til en rekke.



Kilde: <http://www.ece.ucdavis.edu/cerl/reliablejpeg/compression/> 02.10.17

Denne prosessen gjentar seg for alle 8x8 blokker i hele bildet som blir lagret som den nye JPEG filen.

Gange bidraget med kvatiseringsverdien.



Kilde: <http://www.eetimes.com/document.asp?doc_id=1225736> 1.10.17

Generelt: <http://www.techradar.com/news/computing/all-you-need-to-know-about-jpeg-compression-586268/2>

Generelt: <https://www.youtube.com/watch?v=Q2aEzeMDHMA&t=568s>

<https://en.wikipedia.org/wiki/Discrete_cosine_transform>