BAB 2
LANDASAN TEORI

2.1 Karakteristik Video Digital

Video digital pada dasarnya tersusun atas serangkaian *frame*. Rangkaian *frame* tersebut ditampilkan pada layar dengan kecepatan tertentu, bergantung pada laju *frame* yang diberikan (dalam *frame* / detik). Jika laju *frame* cukup tinggi, mata manusia tidak dapat menangkap gambar per *frame*, melainkan menangkapnya sebagai rangkaian yang kontinyu. Masing - masing *frame* merupakan gambar / citra (*image*) *digital*. Suatu *image digital* direpresentasikan dengan sebuah matriks yang masing - masing elemennya merepresentasikan nilai intensitas. Jika I adalah matriks dua dimensi, I (x,y) adalah nilai intensitas yang sesuai pada posisi baris x dan kolom y pada matriks tersebut. Titik - titik dimana *image* di*sampling* disebut sebagai *picture elements* atau sering dikenal sebagai *pixel*. Karakteristik *video digital* ditentukan oleh resolusi (*resolution*) atau dimensi *frame (frame dimension)*, kedalaman *pixel (pixel depth)*, dan laju *frame (frame rate)*. Karakteristik – karakteristik ini yang akan menentukan kualitas *video* dan jumlah *bit* yang dibutuhkan untuk menyimpan atau mentransmisikannya [5].

2.1.1 Resolusi

Resolusi (*resolution*) atau dimensi *frame* (*frame dimension*) adalah ukuran sebuah *frame* pada *video digital*. Resolusi dinyatakan dalam *pixel* x *pixel*. Semakin tinggi resolusi, semakin baik kualitas *video* tersebut, dalam arti bahwa dalam ukuran fisik yang sama, *video* dengan resolusi tinggi akan lebih detil. Namun, resolusi yang tinggi akan mengakibatkan jumlah *bit* yang diperlukan untuk menyimpan atau mentransmisikannya meningkat. Misalnya pada format QCIF (*Quarter Common Intermediate Formats*), resolusinya adalah 144 *pixel* x 176 *pixel*, ukuran ini umum digunakan untuk konferensi *video.*

2.1.2 Kedalaman Bit

Kedalaman *bit* (*bit depth*) menentukan jumlah *bit* yang digunakan untuk merepresentasikan tiap *pixel* pada sebuah *frame*. Kedalaman *bit* dinyatakan dalam *bit* per *pixel*. Semakin banyak jumlah *bit* yang digunakan untuk merepresentasikan sebuah *pixel*, berarti semakin tinggi kedalaman *pixel*nya, maka semakin tinggi pula kualitasnya, mengakibatkan jumlah *bit* yang diperlukan menjadi lebih tinggi. Satu byte (8 *bit*) untuk tiap *pixel*, diperoleh 2^8 atau 256 *level* intensitas. Dengan *level* intensitas sebanyak itu, umumnya mata manusia sudah dapat dipuaskan. Ke dalam *pixel* paling rendah terdapat pada *binary-value image* yang hanya menggunakan 1 *bit* untuk tiap *pixel*, sehingga hanya ada dua kemungkinan bagi tiap *pixel*, yaitu 0 (hitam) atau 1 (putih).

2.1.3 Laju Frame

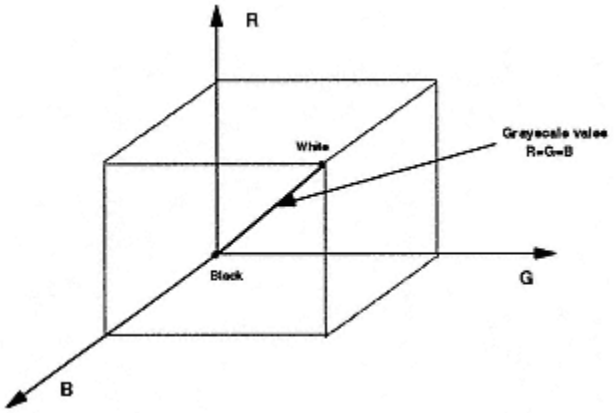
Laju *frame* (*Frame rate)* menunjukkan jumlah *frame* yang digambar tiap detik, dan dinyatakan dengan *frame* per detik. Sehubungan dengan laju *frame* ini, ada dua hal yang perlu diperhatikan, yaitu kehalusan gerakan (*smooth motion*) dan kilatan (*flash*). Kehalusan gerakan ditentukan oleh jumlah *frame* yang berbeda per detik. Untuk mendapatkan gerakan yang halus, *video digital* setidaknya harus menampilkan sedikitnya 25 *frame* per detik. Kilatan ditentukan oleh jumlah berapa kali layar digambar per detik. Dengan 20 *frame* per detik, kilatan sudah dapat dilenyapkan. *Video* yang berkualitas baik akan memiliki laju *frame* yang tinggi, setidaknya sesuai dengan mata manusia, yang berarti membutuhkan jumlah *bit* yang lebih tinggi.

2.1.4 Representasi Warna

Pada *video digital*, umumnya data *video* dipisahkan menjadi komponen – komponen, baik komponen warna maupun komponen kecerahan. Penyajian semacam ini disebut *component video*. Pada *component video*, tiap komponen dipisahkan dengan cara tertentu. Beberapa cara pemisahan komponen tersebut adalah [5]:

1. RGB

Data *video* dapat dipisahkan menjadi komponen – komponen untuk masing – masing warna, yaitu merah (*Red*), hijau (*Green*), dab biru (*Blue*). Warna tiap *pixel* ditentukan oleh kombinasi intensitas dari masing – masing komponen warna. Sebagai contoh, pada RGB 24 *bit*, masing – masing komponen warna dinyatakan dalam 8 *bit* atau 256 *level*.



Gambar 2. 1 Representasi RGB pada warna gambar

2. YUV

Pemisahan komponen tidak hanya dilakukan dengan pemisahan warna, namun dapat juga dilakukan dengan memisahkan menurut komponen kecerahan (*luminance*) dan komponen warna (*crominance*). Pada format PAL (*Phase Alternating Line*), sinyal kecerahan dinyatakan dengan Y, sedangkan dua sinyal warna dinyatakan dengan U dan V. Representasi YUV lebih alami untuk kompresi gambar dan video. Transformasi tepat dari RGB representasi YUV, ditentukan oleh standar CCIR 601, dengan persamaan berikut :

$$Y = 0.299R + 0.587G + 0.114B \qquad \textit{(Pers. 2. 1)}$$

$$U = 0.564(B - Y)$$

$$V = 0.713(B - Y)$$

Dimana Y adalah komponen kecerahan (luminance), sedangkan U dan V adalah dua komponen warna (chrominance).

Perkiraan RGB ke transformasi YUV yang diberikan :

$Y = 0.3R + 0.6G + 0.1B$ *(Pers. 2. 2)*

$U = B - Y$

$V = R - Y$

Transformasi ini lebih baik, ketika R+G+B, maka Y=R=G=B, dan U=V=O.

3. YIQ

Pemisahan sinyal *video* menjadi komponen kecerahan dan komponen warna dapat dilakukan juga sesuai dengan format NTSC (*National Television System Committee*), komponen kecerahan dinyatakan dengan Y, dan dua komponen warna dinyatakan dengan I dan Q. Karena persepsi mata manusia lebih peka pada kecerahan dari pada warna, maka *crominance* cukup di-*sampling* separuh dari *luminance.*

2.2 Kompresi Data

Kompresi data merupakan cabang ilmu komputer yang bersumber dari Teori Informasi. Teori informasi sendiri adalah salah satu cabang Matematika yang berkembang sekitar akhir dekade 1940-an. Tokoh utama dari Teori Informasi adalah Claude Shannon dari Bell Laboratory. Teori Informasi memfokuskan pada berbagai metode tentang informasi termasuk penyimpanan dan pemrosesan pesan. Teori Informasi mempelajari pula tentang redudancy (informasi tak berguna) pada pesan. Semakin banyak redudancy semakin besar pula ukuran pesan, upaya mengurangi redudancy inilah yang akhirnya melahirkan subyek ilmu tentang Kompresi Data.

2.2.1 Teknik Kompresi Data

Teknik kompresi data dapat dibagi menjadi dua kategori besar, yaitu [14]:

1. Lossy Compression

   Lossy compression menyebabkan adanya perubahan data dibandingkan sebelum dilakukan proses kompresi. Sebagai gantinya *lossy compression* memberikan derajat kompresi lebih tinggi. Tipe ini cocok untuk kompresi file suara digital dan gambar digital. File suara dan gambar secara alamiah masih bisa digunakan walaupun tidak berada pada kondisi yang sama sebelum dilakukan kompresi.
2. Lossless Compression

   Sebaliknya *Lossless Compression* memiliki derajat kompresi yang lebih rendah tetapi dengan akurasi data yang terjaga antara sebelum dan sesudah proses kompresi. Kompresi ini cocok untuk basis data, dokumen atau *spreadsheet*. Pada *lossless compression* ini tidak diijinkan ada bit yang hilang dari data pada proses kompresi.

2.2.2 Manfaat Kompresi

Beberapa manfaat kompresi adalah [14]:

1. Waktu pengiriman data pada saluran komunikasi data menjadi lebih singkat. Contohnya pengiriman gambar dari *faximile, video confrencing, handphone, download* dari internet pengiriman data medis, pengiriman dari satelit, dan lain-lain.
2. Membutuhkan ruang memori dalam *storage* yang lebih sedikit dibandingkan dengan data yang tidak dimampatkan.
3. Efisiensi penggunaan *bandwidth* pada jaringan telekomunikasi.

2.2.3 Kriteria Kompresi

Kriteria yang digunakan untuk mengukur kompresi adalah [14]:

1. Waktu kompresi dan waktu dekompresi

Proses kompresi merupakan proses mengkodekan data atau citra sehingga diperoleh data dengan representasi kebutuhan memori yang minimum. Data terkompresi disimpan dalam file dengan format tertentu, misalnya JPEG atau MPEG. Sedangkan proses dekompresi adalah proses untuk menguraikan citra atau data yang dimampatkan untuk dikembalikan lagi menjadi citra yang tidak mampat. Algoritma pemampatan yang paling baik adalah algoritma yang membutuhkan waktu untuk kompresi dan dekompresi yang paling sedikit.

2. Kebutuhan memori

   Metode kompresi yang baik adalah metode kompresi yang mampu mengompresi file menjadi ukuran yang paling minimal.Algoritma pemampatan yang baik akan menghasilkan memori yang dibutuhkan untuk menyimpan hasil kompresi yang berkurang secara berarti. Biasanya semakin besar persentase pemampatan, semakin kecil kebutuhan memori yang diperlukan sehingga kualitas citra makin berkurang. Dan, sebaliknya, semakin kecil persentase citra yang dimampatkan, semakin bagus kualitas hasil pemampatan tersebut.

3. Kualitas pemampatan

   Metode kompresi yang baik adalah metode kompresi yang mampu mengembalikan citra hasil kompresi menjadi citra semula tanpa kehilangan informasi apapun. Kalaupun ada informasi yang hilang akbat pemampatan, sebaiknya hal itu ditekan seminimal mungkin. Semakin berkualitas hasil pemampatan, semakin besar memori yang dibutuhkan. Sebaliknya, semakin jelek kualitas citra hasil pemampatan, semakin kecil kebutuhan memori yang harus disediakan.

   Kualitas citra hasil pemampatan dapat diukur secara kuantitatif menggunakan besaran PSNR(*Peak Signal to Noise Ratio*). Semakin besar nilai PSNR maka citra hasil pemampatan semakin mendekati citra aslinya, dengan kata lain semakin bagus kualitas citra hasil pemampatan tersebut.

Sebaliknya, semakin kecil nilai PSNR, semakin jelek kualitas citra hasil pemampatan.

2.2.4 Kriteria Penilaian Kualitas Citra

Pada bagian ini dibahas mengenai kriteria-kriteria penilaian baik-buruknya kualitas citra, yaitu kriteria penilaian secara obyektif dengan menggunakan suatu ukuran tertentu dan kriteria penilaian secara subyektif dengan berdasarkan pengamatan mata manusia.

2.2.4.1 Penilaian Secara Obyektif

Kriteria penilaian secara obyektif berdasarkan pengukuran *error* pada citra yang diuji. Parameter-parameter yang digunakan sebagai kriteria penilaian obyektif adalah sebagai berikut :

1. ***Mean Square Error (MSE)***

   Rumus MSE dapat dituliskan sebagai berikut :

$$MSE = \frac{1}{M.N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left(f(x,y) - g(x,y) \right)^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(Pers. 2. 3)}$$

2. ***Peak Signal to Noise Ratio (PSNR)***

   Peak Signal to Noise Ratio (PSNR) adalah perbandingan antara nilai maksimum dari sinyal yang diukur dengan besarnya derau yang berpengaruh pada sinyal tersebut. PSNR merupakan parameter standar untuk menilai kualitas suatu citra secara obyektif dengan membandingkan *noise* terhadap sinyal puncak.

   Nilai PSNR yang baik biasanya berkisar antara 20dB sampai dengan 40dB. Pada umumnya disajikan dengan angka desimal yaitu dua angka dibelakang koma. Nilai nyata (*actual value*) tidak sepenuhnya berarti,

tetapi perbandingan dua nilai untuk citra rekonstruksi yang berbeda memberikan satu nilai mutu [17].

PSNR biasanya diukur dalam satuan desibel.

Rumus PSNR adalah sebagai berikut :

$$PSNR = 20.log\ 10 \left(\frac{255}{\sqrt{MSE}} \right) \quad \text{(Pers. 2. 4)}$$

2.2.4.2 Penilaian Secara Subyektif

Penentuan penilaian ini berdasarkan atas karakteristik visual manusia yang mampu melakukan pendeteksian secara langsung. Dengan menggunakan kriteria ini, baik buruknya citra hasil pengolahan ditentukan oleh pengamat sendiri sehingga bisa terjadi dua buah citra yang mempunyai nilai yang sama pada kriteria penilaian obyektif namun dapat berbeda kualitas subyektifnya tergantung dari persepsi visual pengamat. Penilaian ini diberikan oleh 30 koresponden supaya memiliki nilai yang valid.

Adapun kriteria penilaian secara subyektif yang dapat digunakan adalah sebagai berikut :

1. *Unuseable* (skor penilaian 0)
   Citra yang diamati memiliki kualitas yang sangat rendah, sehingga sudah tidak dapat diamati lagi.

2. *Inferior* (skor penilaian 1)
   Citra yang diamati memiliki kualitas sangat rendah tapi masih bisa diamati dengan keberadaan interferensi yang benar-benar mengganggu.

3. *Marginal* (skor penilaian 2)
   Citra yang diamati memiliki kualitas yang rendah, sehingga diinginkan dapat diperbaiki dan interferensi masih terasa cukup mengganggu.

4. *Passable* (skor penilaian 3)

Citra yang diamati memiliki kualitas yang cukup tinggi, dimana interferensi terasa agak mengganggu.

5. *Fine* (skor penilaian 4)
   Citra yang diamati memiliki kualitas yang tinggi, enak dilihat, dimana interferensi tidak terasa mengganggu.

6. *Excellent* (skor penilaian 5)
   Citra yang dihasilkan memiliki kualitas yang sangat tinggi, sebaik-baiknya kualitas sebagaimana yang diinginkan.

2.2.5 Rasio Kompresi

Rasio kompresi adalah ukuran persentase citra yang telah berhasil dimampatkan. Secara matematis rasio pemampatan citra dituliskan sebagai berikut :

$$Rasio = 100\% - (\frac{HasilKompresi}{CitraAsli} \times 100\%) \qquad (\text{Pers. 2. 5})$$

2.3 Standar H.265/HEVC

HEVC adalah standar baru untuk kompresi video yang memiliki potensi untuk memberikan kinerja yang lebih baik daripada standar sebelumnya seperti H.264/AVC. Sumber video, terdiri dari serangkaian video *frame*, dikodekan atau dikompresi oleh *encoder* video HEVC untuk membuat *bitstream* kompresi video. *Bitsream* kompresi disimpan atau ditransmisikan untuk membuat urutan *decoded frame* [6].

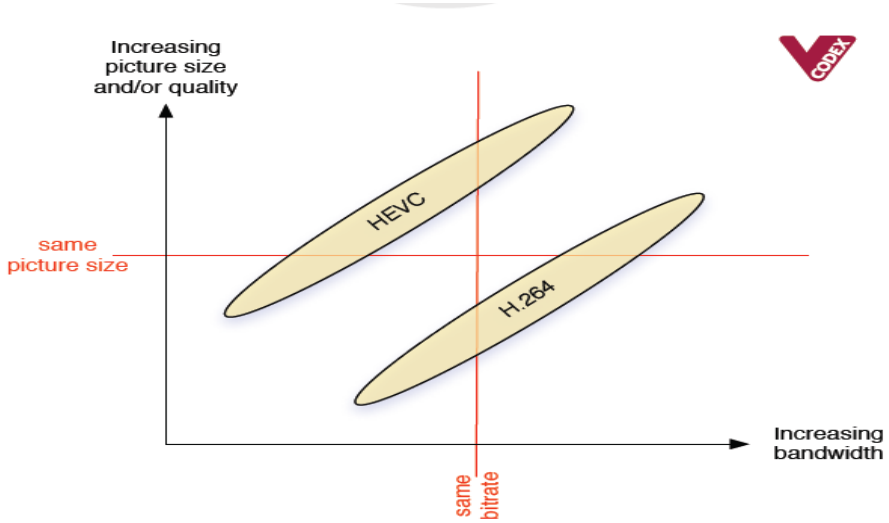
HEVC memiliki struktur dasar yang sama seperti standar sebelumnya seperti video MPEG-2 dan H.264/AVC. Namun, HEVC berisi banyak perbaikan seperti :

1. Partisi lebih fleksibel, dari ukuran partisi yang besar hingga kecil.
2. Fleksibilitas yang lebih besar dalam mode prediksi dan mengubah ukuran blok.
3. Interpolasi dan *deblocking* lebih canggih.

4. Prediksi dan mode signal dan gerakan vektor lebih canggih.
5. Fitur untuk mendukung pemrosesan efisien pararel.

Hasilnya adalah pengkodean standar yang dapat memungkinkan kompresi yang lebih baik, pada biaya peningkatan kekuatan pemrosesan video. Dengan HEVC, harus memungkinkan untuk menyimpan atau mengirimkan video yang lebih efisien dibandingkan dengan teknologi sebelumnya. Ini berarti:

1. Pada ukuran gambar dan kualitas yang sama, sebuah *video sequence* HEVC harus menempati sedikit kapasitas panyimpanan atau kapasitas transmisi daripada setara dengan H.264 *video sequence*.
2. Pada penyimpanan atau transmisi *bandwith* yang sama, kualitas dan atau resolusi sebuah *video sequence* HEVC harus lebih tinggi dari *video sequenc*e H.264.



Gambar 2. 2 Potensi keuntungan dari HEVC vs H.264 (tanpa skala)

(Sumber : http://vcodex.com)

2.4 Cara Kerja *HEVC (High Eficiency Video Coding)*

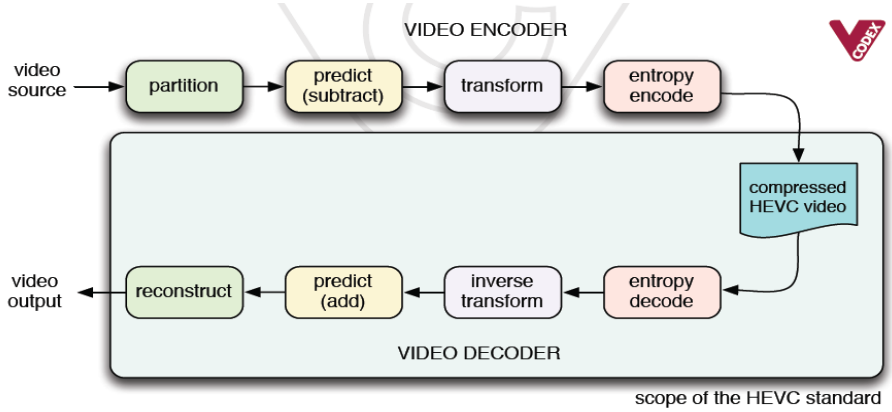
Langkah-langkah yang dilakukan oleh video *encoder* meliputi :

1. Partisi setiap gambar ke beberapa unit.

2. Prediksi setiap unit menggunakan inter atau intra prediksi, dan mengurangi prediksi dari unit.
3. *Transform* dan kuantisasi sisa (perbedaan antara unit gambar asli dan prediksi)
4. *Entropy coding*, prediksi informasi, modus informasi dan *header*.

Langkah-langkah Video *decoder* kebalikan dari video *encoder* yaitu :

1. *Entropy encoding* dan mengekstraksi unsur-unsur dari urutan kode.
2. *Rescaling* dan membalik tahap *transform*.
3. Prediksi setiap unit dan menambahkan prediksi *output* pengubah *invers*.
4. Merekonstruksi gambar *video decoded*.



Gambar 2. 3 Struktur sebuah HEVC encoder dan decoder

(Sumber : http://vcodex.com)

Keuntungan atau kelebihan dari HEVC/H.265 yaitu :

1. Mendukung tayangan UHDTV/8K dengan resolusi 8192 x 4320 pixel.
2. Warna 12 bit, menayangkan gambar lebih berwarna dibanding kompresi video generasi sebelumnya.
3. Chroma sub sampling 4:4:4 dan 4:2:2.
4. Maksimum kecepatan video sampai 300 fps, untuk versi awal masih 59,94 fps.

5. Targetnya memberikan tayangan gambar lebih baik tapi kompresi video hanya separuh dari H.264.

2.4.1 *Partitioning*

HEVC mendukung partisi yang sangat fleksibel dari video *sequence*. Setiap kode *video frame*, atau gambar, dibagi kedalam *Tiles* dan atau *Slices*, yang selanjutnya dibagi ke *Coding Tree Units* (CTUs). CTU adalah unit dasar *coding*, analog dengan *Macroblock* standar sebelumnya, dan dapat menjadi hingga 64x64 pixel dalam satu ukuran. CTU dapat dibagi menjadi persegi yang dikenal sebagai *Coding Units* (CUs) menggunakan struktur *quadtree* [7].



Gambar 2. 4 Video frame yang menunjukkan Slices dan CTUs

(source : Parabola search)

2.4.2 *Prediction*

CUs dipartisi ke satu atau lebih *Prediction Units* (PUs), masing-masing yang diprediksi menggunakan Intra atau Inter Prediksi.

a. *Kompresi Intraframe* : Kompresi *Intra* Prediksi dilakukan dengan memanfaatkan redundansi *spasial* yang terdapat dalam suatu *frame*. Redundansi ini disebabkan karena adanya kesamaan antara sebuah *pixel* dengan *pixel* disekitarnya.

Kompresi *intraframe* terdiri dari proses transformasi dan kuantisasi, dalam proses transformasi digunakan 3D-DCT untuk melakukan proses transformasi dari domain waktu ke domain ruang. Kuantisasi digunakan untuk memotong hasil transformasi, proses selanjutnya adalah pengkodean dengan menggunakan CABAC.

Tahap paling awal pada kompresi *intraframe* adalah persiapan blok, yaitu suatu *frame* dibagi menjadi blok-blok yang tidak saling menindih. Pembagian blok ini diperlukan agar proses kompresi menjadi efisien, karena proses akan dilakukan pada blok-blok yang relatif kecil.

b. *Kompresi Interframe* : kompresi *interframe* dilakukan dengan memanfaatkan redudansi *temporal* yang terdapat antar *frame*. Redudansi *temporal* disebabkan adanya pixel-pixel yang berkorelasi di antara *frame-frame* tersebut, terutama dikarenakan banyak bagian *frame* yang berubah dibanding *frame* sebelum atau sesudahnya.

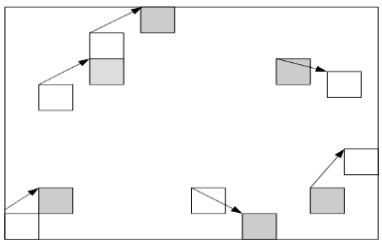
   Proses yang digunakan dalam kompresi *interframe* adalah estimasi gerak (*motion compensation*) dengan teknik pencocokan blok (*matching block*) untuk mendapatkan vektor gerak (*motion vector*).

1. Estimasi dan Kompensasi Gerak

Estimasi gerak (*motion estimation)* merupakan teknik kompresi *interframe* yang memprediksi sebuah *frame* dari *frame* sebelumnya (*reference frame),* dengan mengestimasi gerakan blok – blok antar *frame* tersebut. *Frame* dibagi menjadi blok – blok yang tidak *overlap*. Tiap blok dibandingkan dengan blok – blok berukuran sama, pada *frame* sebelumnya dengan melakukan pencocokan blok (*block matching).*

Dalam melakukan pencocokan tersebut, lokasi dari blok yang paling mirip atau *match* pada *frame* referensi (*reference frame*) berbeda dari lokasi blok

target (*target block*). Perbedaan relatif posisi ini disebut vektor gerak (*motion vector*), seperti ditunjukkan pada Gambar 2. *5*.



Gambar 2. 5 Vektor Gerak

Jika posisi blok target dan blok yang *match* sama, maka vektor geraknya adalah nol. Vektor gerak inilah yang menunjukkan pergeseran blok – blok antar *frame*. Ketika mengkodekan tiap blok dari *frame* yang diprediksi, vektor gerak yang menunjukkan posisi blok yang *match* pada *frame* referensi, dikodekan pada posisi blok yang *match* pada *frame* referensi, dikodekan pada posisi target blok itu sendiri, maka terjadi kompresi, karena jumlah *bit* yang diperlukan untuk mengkodekan vektor gerak lebih sedikit daripada untuk mengkodekan suatu blok secara utuh.

Pada *dekompresi*, *decoder* menggunakan vektor gerak untuk menemukan *matching block* pada *frame* referensi dan menyalin *matching block* tersebut ke posisi yang sesuai pada *frame* yang sedang diprediksi. Dengan demikian, suatu *frame* prediksi tersusun atas blok – blok dari *frame* sebelumnya.

Keefektifan teknik kompresi menggunakan kompensasi gerak berbasis blok ini bergantung pada beberapa kondisi berikut :

1. Objek bergerak pada bidang datar. Efek dari *zoom* dan rotasi tidak dapat ditangani dengan metode ini.
2. Pencahayaan harus konstan dan seragam. Objek yang mengalami perubahan pencahayaan tidak dapat dikenali
3. Objek yang dilewati objek lain tidak dapat ditangani dengan metode ini.

Blok – blok yang terletak pada pinggiran *frame*, diestimasi dan vektor gerak boleh melebihi batas *frame*. *Pixel – pixel* pinggir digunakan untuk

melakukan kompensasi bila vektor gerak menunjuk ke *pixel* yang berada diluar batas *frame.*

2. Block Matching

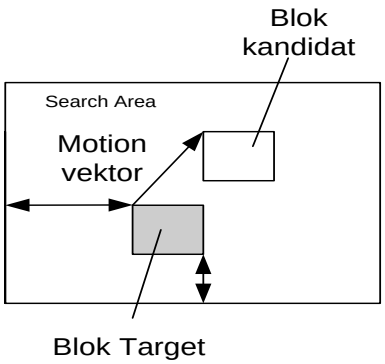
Pencocokan blok atau *block matching* adalah proses pembandingan blok dengan blok – blok pada *frame* sebelumnya, untuk menemukan *matching block*. *Matching block* merupakan proses yang paling banyak menyita waktu selama *encoding*. *Matching block* cukup dilakukan pada komponen kecerahan (*luminance*) dari *frame*. Hal ini dikarenakan mata manusia lebih peka terhadap kecerahan.

Langkah pertama untuk proses ini adalah membagi *frame* menjadi blok – blok berukuran tertentu. Ukuran blok yang besar mengakibatkan sedikit jumlah vektor gerak yang dihasilkan. Namun, akan sulit menemukan blok yang *match* dengannya dan *error* yang dihasilkan perbedaan blok relatif besar. Blok yang dibandingkan dengan blok – blok pada *frame* referensi disebut sebagai blok target.

Langkah selanjutnya adalah menentukan *search area* atau daerah pencarian pada *frame* referensi. Pencarian blok yang *match* dapat dilakukan pada seluruh daerah *frame* referensi. Namun karena perubahan antar *frame* cenderung kecil, daerah pencarian cukup dibatasi pada posisi sekitar blok target pada *frame* referensi, maka ditentukan suatu *maximum displacement* yang membatasi jumlah *pixel* maksimum pada arah *vertikal* dan *horizontal* dari posisi blok target pada *frame* saat ini.

Langkah terakhir adalah menemukan pencocokan blok pada daerah pencarian. Proses ini dilakukan dengan membandingkan target blok dengan blok – blok pada daerah pencarian yang disebut blok kandidat, seperti ditunjukkan pada Gambar 2. *6*. Semakin besar *displacement,* semakin luas daerah pencarian, semakin besar pula peluang untuk mendapatkan pencocokan blok yang bagus. Namun jumlah blok kandidat meningkat secara *kuadratik* sebanding dengan peningkatan *displacement*, sehingga lebih banyak lagi pembandingan blok yang perlu dilakukan.

Pencocokan blok target dengan blok – blok kandidat pada daerah pencarian dilakukan dengan besar *step* tertentu, yang merupakan besar pergeseran dalam pencarian blok. Jumlah blok kandidat, selain ditentukan oleh ukuran daerah pencarian, ditentukan pula oleh besarnya *step.*



Gambar 2. 6 Pencocokan Blok

Setelah diperoleh *matching* blok, maka perbedaan posisinya dengan target blok disebut vektor gerak (*motion vector*), Proses ini menghasilkan vektor gerak pada arah horizontal MVx dan vektor gerak pada arah *vertikal* MVy.

2.4.3 *Transform* dan *Quantization*

Data sisa yang tersisa setelah prediksi adalah mentransformasikan blok yang berdasarkan *Discrete Cosine Transform* (DCT) atau *Discrete Sine Transform (DST).* Satu atau lebih blok diubah dalam ukuran 32x32, 16x16, 8x8, dan 4x4 yang diterapkan sisa data dalam setiap CU [15].

Proses kuantisasi merupakan proses untuk mengurangi jumlah *bit* yang diperlukan untuk menyimpan suatu nilai dengan memperkecilnya. Proses ini diterapkan pada keluaran proses DCT. Kuantisasi dilakukan dengan membagi keluaran proses DCT dengan suatu nilai yang ditetapkan dalam *matriks* kuantisasi.

Quantum adalah matriks kuantisasi. *Matriks* kuantisasi dapat dipilih *uniform* atau *nonuniform*. Pada matriks kuantisasi *uniform*, semua koefisien

mempunyai besar yang sama, sedangkan, pada matriks kuantisasi *nonuniform*, koefisien – koefisien pada matriks meningkat tajam dari titik pusat. Hal ini akan mengakibatkan nilai frekuensi tinggi pada keluaran DCT akan dipotong dengan cepat, sehingga kompresi yang dilakukan lebih efektif [15].

Transformin *Forward integer* HEVC adalah perkiraan DCT yang ditetapkan sebagai perkalian matriks. Integer DCT adalah [8] :

$$Y = CXC^T \qquad \text{(Pers. 2. 6)}$$

Dimana X adalah *residual block* (sisa blok), Y adalah koefisien peubah matriks, dan C adalah inti transformasi matriks. Persamaan tersebut menunjukkan bentuk umum NxN blok diubah oleh integer DCT di HEVC.

2.3.3.1 *Discrete Cosine Transform (DCT)*

Metode DCT (*Discrete Cosine Transform*) yang pertama kali diperkenalkan oleh Ahmed, Natarajan dan Rao pada tahun 1974 dalam makalahnya yang berjudul "*On Image Processing and a Discrete Cosine Transform*".

Discrete Cosine Transform adalah sebuah teknik untuk mengubah sebuah sinyal kedalam komponen frekuensi dasar. *Discrete Cosine Transform* merepresentasikan sebuah citra dari penjumlahan sinusoida dari magnitude dan frekuensi yang berubah-ubah. Sifat dari DCT adalah mengubah informasi citra yang signifikan dikonsentrasikan hanya pada beberapa koefisien DCT.

Discrete Cosine Transform berhubungan erat dengan *Discrete Fourier Transform (DFT)*, sehingga menjadikan data direpresentasikan dalam komponen frekuensinya. Demikian pula, dalam aplikasi pemrosesan gambar, DCT tiga dimensi (3D) memetakan sebuah gambar atau sebuah segmen gambar kedalam komponen frekuensi 3D (tiga dimensinya).

Discrete Cosine Transform adalah sebuah skema *lossy compression* dimana NxN blok ditransformasikan dari domain spasial ke domain DCT. DCT

menyusun sinyal tersebut ke frekuensi spasial yang disebut dengan koefisien DCT. Frekuensi koefisien DCT yang lebih rendah muncul pada kiri atas dari sebuah matriks DCT, dan frekuensi koefisien DCT yang lebih tinggi berada pada kanan bawah dari matriks DCT. Sistem penglihatan manusia tidak begitu *sensitive* dengan *error-error* yang ada pada frekuensi tinggi dibanding dengan yang ada pada frekuensi rendah. Karena itu, maka frekuensi yang lebih tinggi tersebut dapat dikuantisasi [5].

Kelebihan kompresi data menggunakan *Discrete Cosine Transform* adalah :

1. DCT menghitung kuantitas bit-bit data gambar dimana pesan tersebut disembunyikan didalamnya. Walaupun gambar yang dikompresi dengan *lossy compression* akan menimbulkan kecurigaan karena perubahan gambar terlihat jelas, pada metode ini hal ini tidak akan terjadi karena metode ini terjadi di domain frekuensi di dalam image, bukan pada domain spasial, sehigga tidak akan ada perubahan yang terlihat pada cover gambar, dan
2. Kokoh terhadap manipulasi pada *stego-object*.

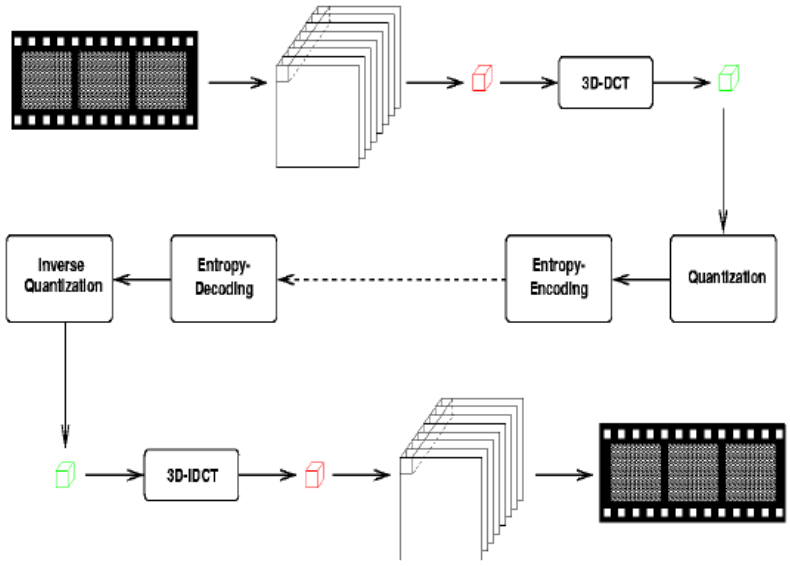
Sedangkan kekurangan kompresi data menggunakan *Discrete Cosine Transform* adalah :

1. Tidak tahan terhadap perubahan suatu objek dikarenakan pesan mudah dihapus karena lokasi penyisipan data dan pembuatan data dengan metode DCT diketahui.
2. Implementasi algoritma yang panjang dan membutuhkan banyak perhitungan.

Pada *dekompresi*, untuk mentransformasikan kembali data dari domain frekuensi ke domain ruang, digunakan *inverse* dari *discrete cosine transform* atau IDCT.

2.3.3.2 *Three Dimesion-Discrete Cosine Transform (3D-DCT)*

Diagram blok 3D-DCT *coder* dari data multidimensi ditunjukkan pada Gambar 2. 7. Pertama, data 2D dibagi menjadi kumpulan kelompok frame (paling sering 8 frame per kelompok). Kemudian masing-masing kelompok frame dibagi menjadi kubus 3D, biasanya berukuran 8x8x8. Kemudian, 3D-DCT diterapkan untuk setiap kubus. Karena dekorelasi dari DCT, sebagian besar dari energi terkonstentrasi di sejumlah kecil koefisien. Kuantisasi digunakan untuk menghapus beberapa yang kurang relevan, untuk mengamati, informasi dan untuk memfasilitasi kompresi lebih jauh. Setelah itu, *entropy encoding* dilakukan. Agar efisien, *entropy encoding* dilakukan secara terpisah untuk berbagai komponen DCT spektrum. Koefisien DC bisa diprediksi efisien dari blok tetangga. Dengan demikian, perbedaannya antara diprediksi dan nilai riil yang dikodekan. [9] .



Gambar 2. 7 Blok Diagram 3D-DCT dari Data Multidimensi

(Sumber : Three-Dimensional DCT Video Compression Technique Based On Adaptive Quantizers. *Multimedia Laboratory*)

Nilai koefisien DCT dapat diperoleh dengan rumus berikut :

$$F(u,v,w) = C(u)C(v)C(w) * \sum_{x=0}^{N_1-1}\sum_{y=0}^{N_{2-1}}\sum_{z=0}^{N_3-1} . f(x,y,z) * \frac{cos((2x+1)u\pi)}{2N_1}\frac{cos((2y+1)v\pi)}{2N_2}\frac{cos((2z+1)w\pi)}{2N_3}$$

(Pers. 2. 7)

Dimana :

x,y,z adalah *index pixels* dari ruang pixel (*pixel space)*,

f(x,y,z) adalah nilai sebuah pixel dalam ruang pixel (*pixel space)*,

u,v,w adalah index pixels dalam ruang DCT (*DCT space*),

F(u,v,w) adalah transformasi nilai pixel kedalam ruang DCT (*DCT space*), dan

$$C\ (i,j) = \begin{cases} \frac{1}{\sqrt{N}} & if\ i=0 \\ \sqrt{\frac{2}{N}}\ x\ cos\frac{(2j+1)i\pi}{2N} & if\ i\neq 0 \end{cases}$$

Kuantisasi dikerjakan sesuai dengan persamaan berikut :

$$F_q(u,v,w) = \left\lfloor \frac{F(u,v,w)}{Q(u,v,w)} \right\rfloor \qquad \textit{(Pers. 2. 8)}$$

Dimana :

F(u,v,w) adalah elemen sebelum kuantisasi,

F_q(u,v,w) adalah elemen yang sudah terkuantisasi, dan

Q(u,v,w) adalah elemen dari tabel kuantisasi.

Hasil dari operasi kuantisasi adalah sekumpulan nilai kecil koefisien, dimana nilai terbesarnya adalah 0. Koefisien ini kemudian diubah menjadi urutan biner yang kompak menggunakan *entropy coder*.

2.4.4 *Entropy Coding*

Bitstream kode HEVC terdiri dari *quantized transform coefficients*, prediksi informasi seperti mode prediksi dan gerakan vektor, partisi informasi dan data *header* lainnya. Semua elemen ini dikodekan menggunakan *Context Adaptive Binary Arithmetic Coding (CABAC).* Operasi *entropy coding* dimulai dengan penataan kembali koefisien dalam urutan menurun dari nilai yang diharapkan.

2.4.4.1 *CABAC (Context Adaptive Binary Arithmetic Coding)*

Jika pada standar H.264 bisa menggunakan algoritma CABAC ataupun CAVLC pada proses *entropy coding*, maka lain halnya dengan H.265/HEVC yang hanya bisa menggunakan CABAC pada proses *entropy coding*.

CABAC merupakan sebuah pengkode entropi yang melakukan pengkodean aritmatika biner secara adaptif, berdasarkan *context* dari data yang dikodekan. CABAC merupakan algoritma yang secara dinamis, dapat memprediksi tingkat kemungkinan pemunculan simbol (*symbol probability*), pada sistem pengkodean aritmatika biner. Sehingga kemungkinan pemunculan simbol pada pengkodean aritmatik dapat terus ter-*update* untuk simbol-simbol baru, yang akhirnya dapat menghasilkan kompresi yang lebih baik [16]. CABAC digunakan pada 2 sisi yaitu pada sisi *Encoder* dan sisi *Decoder*.

2.5 Unified Modeling Language (UML)

UML adalah penerus gelombang beriorientasi objek analisis dan desain (OOA & D) metode yang muncul di akhir ’80-an dan 90-an awal. Hal yang paling langsung menyatukan metode Booch, Rumbaugh (OMT), dan Jacobson, tapi jangkauan lebih luas dari itu. UML pergi melalui proses standarisasi dengan OMG (*Object Management Group)* dan sekarang menjadi standar OMG.

UML disebut bahasa pemodelan, bukan metode. Sebagian besar metode terdiri, setidaknya pada prinsipnya, kedua model sebuah bahasa dan proses. Bahasa pemodelan adalah notasi (terutama grafis) bahwa metode digunakan untuk mengekspresikan desain. Proses ini menyatakan mereka apa langkah yang harus diambil dalam melakukan desain.

Bagian proses dalam buku banyak metode yang agak samar. Selain itu, kebanyakan orang, ketika mereka mengatakan bahwa mereka menggunakan metode, menggunakan bahasa pemodelan, tapi mereka jarang mengikuti proses. Jadi dalam banyak hal pemodelan bahasa adalah bagian paling penting untuk komunikasi. Jika anda ingin mendiskusikan desain anda dengan seseorang, itu adalah bahasa pemodelan yang anda berdua perlu pahami, bukan proses yang digunakan untuk sampai ke desain tersebut [10].

2.5.1 Diagram UML

Menggunakan berbagai macam diagram dengan fungsi masing-masing untuk menggambarkan setiap proses dari sistem berorientasi objek. Berikut meruapakan beberapa diagram UML diantaranya [11] :

a. Use Case Diagram

Use Case atau diagram *use case* merupakan pemodelan yang digunakan untuk menggambarkan kelakuan (*behavior*) dari sistem yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *usecase* [11].

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat diluar sistem yang akan dibuat itu

sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.

2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

b. Activity Diagram

Diagram aktivitas atau *activity diagram* adalah sebuah diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Dalam diagram aktivitas yang perlu diperhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem, bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem [10].

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut [11] :

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokkan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memiliki seuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

c. Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi [11].

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

d. Sequence Diagram

Diagram sekuen adalah diagram yang menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Banyaknya diagram sekuen yang harus digambarkan adalah sebanyak pendefinisian *use case* yang memiliki proses itu sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak [11].