BAB II
TEORI PENUNJANG

Teori-teori penunjang untuk merancang program pengolahan citra pada sistem wahana *VTOL* yang bertujuan membuat sistem pendeteksian objek pada saat penanggulangan bencana, adapun teori yang akan dipaparkan sebagai berikut.

2.1 Gaussian Blur

Gaussian Blur adalah *filter blur* yang menempatkan warna transisi yang signifikan dalam sebuah gambar, kemudian membuat warna-warna pertengahan untuk menciptakan efek lembut pada sisi-sisi sebuah gambar. *Gaussian blur* adalah salah satu *filter blur* yang menggunakan rumus matematika untuk menciptakan efek *autofokus* untuk mengurangi derau dan menciptakan efek berkabut. *Gaussian* adalah istilah matematika yang diambil dari nama seorang matematikawan Jerman, *Karl Friedrich Gauss*[1]. Berikut adalah persamaan matematika untuk *filter gaussian*:

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \qquad (II.1)$$

Dimana:

1. σ adalah nilai deviasi standar distribusi normal yang digunakan. Semakin besar nilai σ, maka semakin banyak titik tetangga yang diikutkan dalam perhitungan.
2. x dan y adalah posisi koordinat *filter* dimana koordinat (0, 0) adalah posisi titik tengah dari *filter* yang mempunyai nilai paling besar atau paling tinggi.
3. π adalah konstanta dengan nilai 3,14.
4. e adalah konstanta bilangan natural dengan nilai 2,718281828.[8]

Filter Gaussian adalah salah satu *filter linear* dengan nilai pembobotan untuk setiap anggotanya dipilih berdasarkan bentuk fungsi *Gaussian*. *Filter* ini digunakan untuk menghilangkan *noise* yang bersifat sebaran normal.

Salah satu *Filter Gaussian* berukuran 7x7 yang umum digunakan untuk proses menghilangkan *noise* adalah sebagai berikut:

$$\begin{pmatrix} 1 & 1 & 2 & 2 & 2 & 1 & 1 \\ 1 & 2 & 2 & 4 & 2 & 2 & 1 \\ 2 & 2 & 4 & 8 & 4 & 4 & 2 \\ 2 & 4 & 8 & 16 & 8 & 4 & 2 \\ 2 & 2 & 4 & 8 & 4 & 4 & 2 \\ 1 & 2 & 2 & 4 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 1 & 1 \end{pmatrix}$$

2.2 Metode Thresholding

Thresholding adalah metode yang paling sederhana untuk melakukan segmentasi citra. Contoh aplikasinya adalah memisahkan suatu daerah dari suatu gambar sesuai dengan objek yang ingin dianalisis. Pemisahan ini didasarkan pada variasi intensitas antara *pixel* objek dan *pixel* objek lainnya. Untuk membedakan *pixel-pixel* yang akan dianalisa, dilakukanlah suatu perbandingan dari setiap nilai intensitas *pixel* menurut nilai ambang yang telah ditentukan.[2] Karena gambar *output* merupakan array bertipe *unsigned integer* maka nilai tertinggi ialah 255 dimana semua bit berlogika 1 yang di representasikan dengan warna putih dan nilai terendahnya ialah 0 dimana semua bit berlogika 0 yang di representasika dengan warna hitam. Setelah itu dipisahkan antara *pixel* penting (putih) dan *pixel* yang akan dibuang (hitam).[16] Persamaan proses *thresholding* dapat dilihat pada persamaan dibawah:

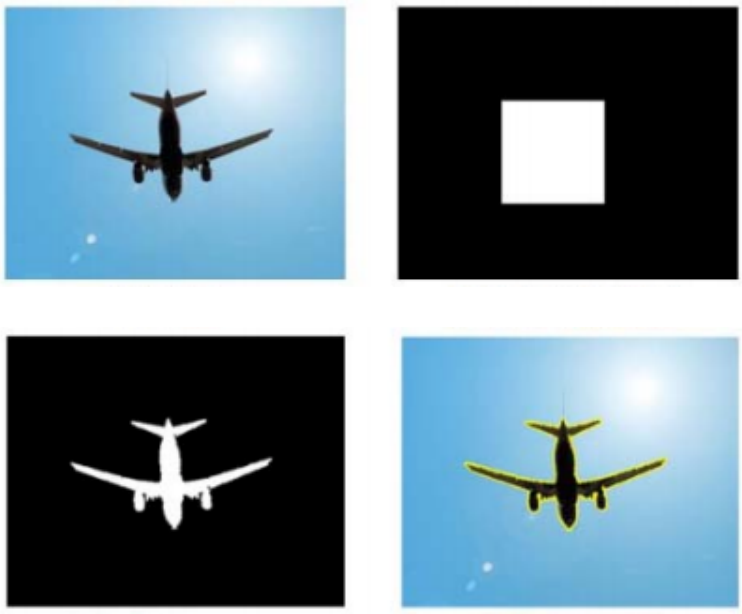
$$f_H = \begin{cases} 1, & jika\ batasBawah(x,y)_H \leq src(x,y)_H \leq batasAtas(x,y)_H \\ 0, & jika\ tidak \end{cases} \quad (II.2)$$

$$f_S = \begin{cases} 1, & jika\ batasBawah(x,y)_S \leq src(x,y)_S \leq batasAtas(x,y)_S \\ 0, & jika\ tidak \end{cases} \quad (II.3)$$

$$f_V = \begin{cases} 1, & jika\ batasBawah(x,y)_V \leq src(x,y)_V \leq batasAtas(x,y)_V \\ 0, & jika\ tidak \end{cases} \quad (II.4)$$

$$f(x,y) = \begin{cases} 255, & jika\ f_H = 1 \wedge f_S = 1 \wedge f_V = 1 \\ 0, & jika\ tidak \end{cases} \quad (II.5)$$

Contoh gambar hasil proses *thresholding* dapat dilihat pada Gambar II.1.



Gambar II.1 Contoh gambar asli dan hasil thresholding[7]

2.3 Pengolahan Warna

Pada pengolahan warna gambar, ada bermacam-macam model warna. Model *RGB* (*Red, Green, Blue*) merupakan model yang banyak digunakan, salah satunya adalah monitor. Pada model ini untuk mempresentasikan gambar menggunakan 3 buah komponen warna tersebut. Selain model *RGB* terdapat juga model *HSV* dimana model ini terdapat 3 komponen yaitu, *Hue*, *Saturation*, *Value*. *Hue* adalah suatu ukuran panjang gelombang yang terdapat pada warna dominan yang diterima

oleh penglihatan sedangkan *Saturation* adalah ukuran banyaknya cahaya putih yang bercampur pada *Hue*.[3]

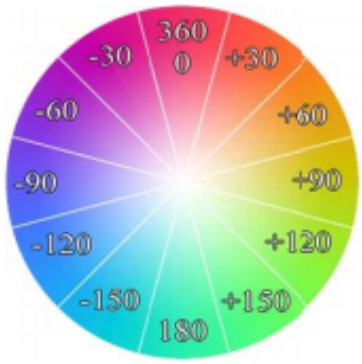
2.4 Model Warna *HSV*

Selain *RGB*, warna juga dapat dimodelkan berdasarkan atribut warnanya. Setiap warna memiliki 3 buah atribut, yaitu *Hue, Saturation,* dan *Value*.[13]

a. *Hue*

Menyatakan warna sebenarnya, seperti merah, violet, dan kuning. *Hue* digunakan untuk membedakan warna-warna dan menentukan kemerahan (*redness*), kehijauan (*greeness*), dan sebagainya dari cahaya. *Hue* berasosiasi dengan panjang gelombang cahaya.

Hue dikuantisasi dengan nilai dari 0 sampai 255; 0 menyatakan merah, lalu memutar nilai-nilai spektrum tersebut kembali lagi ke 0 untuk menyatakan merah lagi. Ini dapat dipandang sebagai sudut dari 0° sampai 360°. Ilustrasi lingkaran elemen warna *Hue* dapat dilihat pada Gambar II.2.



Gambar II.2 Lingkaran elemen warna Hue[9]

b. *Saturation*

Menyatakan tingkat kemurnian warna cahaya, yaitu mengindikasikan seberapa banyak warna putih diberikan pada warna. Jika *Hue* menyatakan warna sebenarnya, maka *Saturation* menyatakan seberapa dalam warna tersebut.

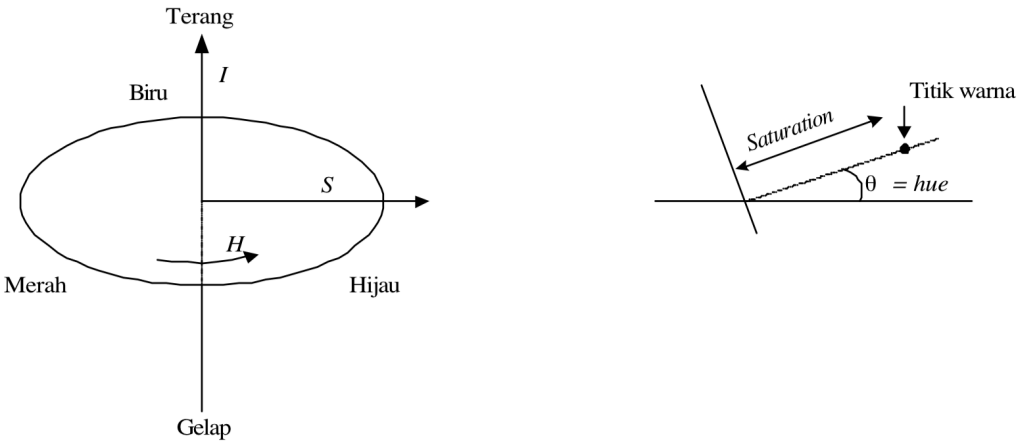
Jika suatu warna mempunyai *saturation* = 0, maka warna tersebut tanpa *hue*, yaitu dibuat dari warna putih saja. Jika *saturation* = 255, maka tidak ada warna putih yang ditambahkan pada warna tersebut. *Saturation* dapat digambarkan sebagai panjang garis dari titik pusat lingkaran ke titik warna.

c. *Value*

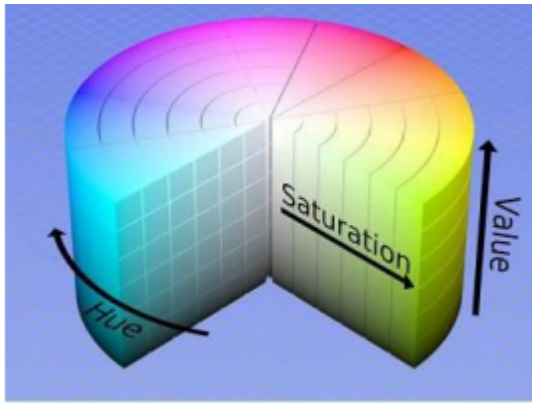
Atribut yang menyatakan banyaknya cahaya yang diterima oleh mata tanpa mempedulikan warna. Kisaran nilainya adalah antara gelap (hitam) dan terang (putih).

Nilai dari gelap sampai terang dalam praktek, gelap = 0, terang = 255; *Value* dapat digambarkan sebagai garis vertikal yang menembus pusat lingkaran.[5]

Ketiga atribut warna (*H, S, dan V*) digambarkan dalam model *HSV* yang diperlihatkan pada Gambar II.2 dan di ilustrasikan pada Gambar II.3[11].



Gambar II.3 Model HSV



Gambar II.4 Ilustrasi Model Warna HSV

Untuk mempermudah proses pengolahan citra, gambar yang semulanya memiliki model warna *RGB* di ubah ke model warna *HSV* dengan persamaan berikut[16]:

$$M = \max(R, G, B) \quad \text{(II.6)}$$

$$m = \min(R, G, B) \quad \text{(II.7)}$$

$$C = M - m \quad \text{(II.8)}$$

$$H' = \begin{cases} tidak\ terdefinisi, & jika\ C = 0 \\ \frac{G-B}{C}\ mod\ 6, & jika\ M = R \\ \frac{B-R}{C} + 2, & jika\ M = G \\ \frac{R-G}{C} + 4, & jika\ M = B \end{cases} \quad \text{(II.9)}$$

$$H = 60\,°\,x\,H' \quad \text{(II.10)}$$

$$S = \begin{cases} 0, & jika\ V = 0 \\ \frac{C}{V}, & jika\ tidak \end{cases} \quad \text{(II.11)}$$

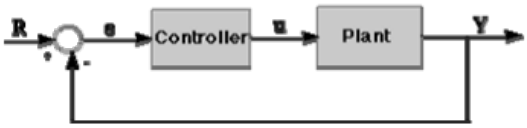
$$V = M \quad \text{(II.12)}$$

2.5 Cahaya

Cahaya merupakan pancaran energi yang merangsang retina manusia dengan menimbulkan sensasi visual sedangkan jumlah cahaya yang jatuh pada sebuah permukaan disebut sebagai pencahayaan, pencahayaan itu sendiri memiliki satuan lux (lm/m^2) dimana lm adalah lumens dan m^2 adalah satuan dari luas permukaan.[19]

2.6 Kendali *PID*

Kendali *PID* (*Proportional-Integral-Derivative*) digunakan dalam sebuah sistem *loop* tertutup yang melibatkan umpan balik dari *output* sistem guna mencapai respon yang diinginkan.[10] Sistem *PID* dapat mengendalikan variabel *input* dengan memanipulasi variabel *output* sehingga diperoleh variabel *input* baru agar menghasilkan *output system* yang sesuai[14]. Contoh dapat dilihat dari blok diagram yang ditunjukkan pada gambar II.3.



Gambar II.5 Blok diagram sistem kontrol closed-loop[4]

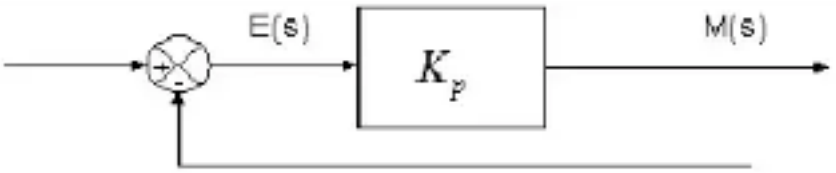
Blok diagram diagram diatas merupakan sistem kontrol *closed-loop* dimana kontroler bekerja sebagai penggerak *plant* (objek fisik yang digerakkan dalam sistem) dan mengontrol sifat *plant*. Sistem *PID* (*Proportional-Integral-Derivative*) sebagai kontroler akan bekerja untuk menggerakkan *plant* sebagaimana ia seharusnya menghasilkan respon yang diinginkan. Yang dikontrol oleh sistem *PID* adalah variabel *output* sistem yaitu Y. Agar diperoleh variabel Y yang sesuai maka sistem *PID* akan memanipulasi variabel *input* R. Variabel yang dimanipulasi (R baru) merupakan hasil komputasi dari variabel R, Y (*feedback*) dan sinyal *error* (e). Sinyal *error* ini dihasilkan oleh *output* Y yang dibawa dalam komponen *feedback* untuk dikirim ke kontroler *PID* sehingga dapat dijadikan pengukuran *error output*. Dari variabel manipulasi ini, diperoleh *output* yang sesuai dengan *error* yang minimum. [4]

Kendali *PID* memiliki tiga komponen yang terdiri dari kendali *Proportional*, *Integral*, *Derivative*. Ketiga komponen ini memiliki peran yang saling terkait satu dengan yang lainnya.[12]

2.7 Kendali *Proportional*

Kendali *proportional* memiliki keluaran yang sebanding dengan besar nya sinyal *error*. Secara sederhana, keluaran kendali *proportional* merupakan perkalian

antara konstanta *proportional* dengan masukannya. Gambar II.4 menunjukan blok diagram yang menggambarkan hubungan antara besaran *setting*, besaran aktual dengan besaran keluaran kendali *proportional*.



Gambar II.6 Blok diagram kendali proportional

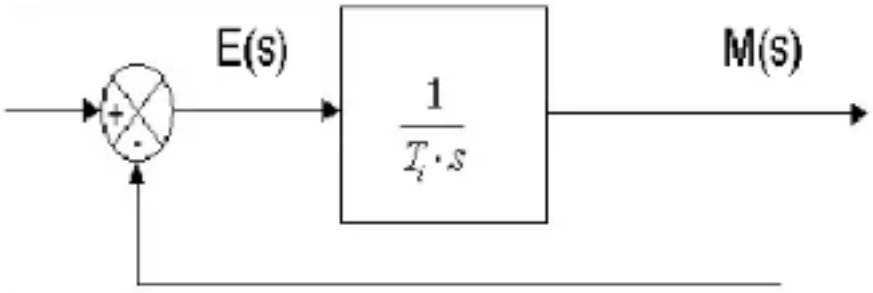
Ciri-ciri kendali *porportional* harus diperhatikan ketika kendali tersebut diterapkan pada suatu sistem. Berikut ciri-ciri kendali *proportional*:

1. Jika nilai Kp kecil, kendali *proportional* hanya mampu melakukan koreksi kesalahan yang kecil, sehingga akan menghasilkan respon sistem yang lambat.
2. Jika nilai Kp dinaikkan, respon sistem akan semakin cepat mencapai keadaan idealnya.

Namun jika nilai Kp diperbesar secara berlebihan, makan akan mengakibatkan sistem bekerja secara tidak stabil, atau respon sistem akan berosilasi.[6]

2.8 Kendali *Integral*

Kendali *integral* berfungsi menghasilkan respon sistem yang memiliki *error* keadaan ideal nol. Jika sebuah *plant* tidak memiliki unsur *integrator* (1/s), kendali *proportional* tidak akan mampu menjamin keluaran sistem dengan *error* keadaan idealnya nol. Dengan kendali *integral*, respon sistem dapat diperbaiki sehingga mempunyai *error* keadaan idealnya nol. Gambar II.5 menunjukkan blok diagram antara besaran *error* dengan keluaran suatu kendali *integral*.



Gambar II.7 Blok diagram hubungan besaran error dengan kendali integral

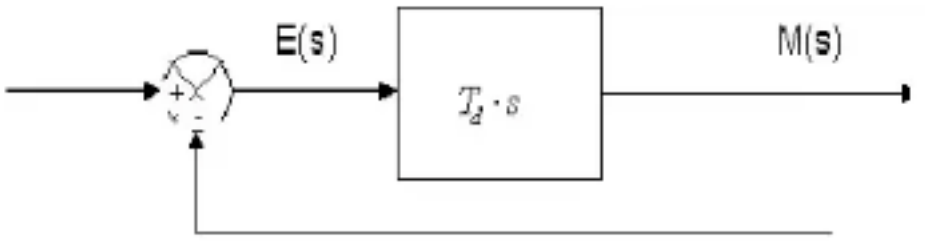
Ketika digunakan, kendali *integral* mempunyai beberapa karakteristik sebagai berikut:

1. Keluaran kendali membutuhkan selang waktu tertentu, sehingga kendali *intergral* cenderung memperlambat respon.
2. Ketika sinyal *error* bernilai nol, maka keluaran kendali akan bertahan pada nilai sebelumnya.
3. Jika sinyal *error* tidak bernilai nol, maka keluaran akan menunjukkan kenaikan atau penurunan yang dipengaruhi oleh besarnya sinyal *error* dan nilai Ki.

Konstanta *integral* Ki yang bernilai besar akan mempercepat hilangnya *offset*. Tetapi semakin besar nilai konstanta Ki akan mengakibatkan peningkatan osilasi dari sinyal keluaran kendali. [6]

2.9 Kendali *Derivative*

Keluaran kendali diferensial memiliki sifat seperti halnya suatu operasi *derivative*. Perubahan yang mendadak pada masukan kenadil akan mengakibatkan perubahan yang sangat besar dan cepat. Gambar II.6 menunjukkan blok diagram yang menggambarkan hubungan antara sinyal *error* dengan keluaran kendali.



Gambar II.8 Blog diagram kendali derivative

Karakteristik kendali diferensial adalah sebagai berikut:

1. Kendali ini tidak dapat menghasilkan keluaran bila tidak ada perubahan pada masukannya.
2. Jika sinyal *error* berubah terhadap waktu, maka keluaran yang dihasilkan kendali tergantung pada nilai Kd dan laju perubahan sinyal *error*.
3. Kendali diferensial mempunyai suatu karakter untuk mendahului, sehingga kendali ini dapat menghasilkan koreksi yang signifikan sebelum pembangkit *error* menjadi sangat besar.

Kerja kendali *derivative* hanya efektif pada lingkup yang sempit, yaitu pada periode peralihan. Oleh sebab itu kendali *derivative* tidak pernah digunakan tanpa ada kendali lain pada sebuah sistem. [6]