BAB II
TINJAUAN PUSTAKA

Pada bab ini dibahas mengenai teori-teori yang menjadi dasar dari pembuatan Tugas Akhir.

2.1 Gaussian Mixture Models

Gaussian mixture models adalah salah satu metode yang digunakan untuk mendeteksi gerak. Didefinisikan piksel pada *frame* yang sedang diproses sebagai X dengan nilai sebelumnya dari *frame* 1 sampai *frame* t. Model dapat dilihat pada Persamaan (2.1).

$$\{X_1, X_1, \ldots, X_i, \ldots, X_t\}\ 1 \leq i \leq t \tag{2.1}$$

Probabilitas piksel X_t dapat didefinisikan menggunakan Persamaan (2.2).

$$P(X_t) = \sum_{i=1}^{K} w_{i,t} * \eta(X_t, \mu_{i,t}, cov_{i,t}) \tag{2.2}$$

Dimana K adalah bilangan gaussian (antara 3 hingga 5), cov adalah *covariance* yang didapatkan dari $variance\,.matriks\ identitas$, w adalah *weight*, η adalah *gaussian probability density function*. *Gaussian probability density function* dapat dilihat pada Persamaan (2.3).

$$\eta(x, \mu, cov) = \frac{1}{(2\pi)^{\frac{n}{2}}|cov|^{\frac{1}{2}}}\, e^{-\frac{1}{2}(x-\mu)^T cov^{-1}(x-\mu)} \tag{2.3}$$

Distribusi K diurutkan secara *descending* berdasarkan nilai w_k/σ_k , nilai distribusi B akan digunakan sebagai model *background*. Nilai B didapatkan menggunakan Persamaan (2.4).

$$B = argmin(\sum_{j=1}^{b} w_j > T) \quad (2.4)$$

Dimana nilai T adalah nilai minimum dari *background model*. Selanjutnya dilakukan *background subtraction* dengan menghitung nilai piksel dengan distribusi B. Jika nilai piksel lebih dari 2.5 dari nilai standar deviasi distribusi tersebut (*match*), maka komponen distribusi tersebut dilakukan *update* dan piksel tersebut dianggap sebagai *foreground*. Untuk *update* nilai komponen gaussian dilakukan dengan Persamaan (2.5)-(2.9) [2].

$$w_k^{N+1} = (1-\alpha)w_k^N + \alpha p(w_k|x_{n+1}) \quad (2.5)$$

$$\mu_k^{N+1} = (1-\alpha)\mu_k^N + \rho x_{n+1} \quad (2.6)$$

$$\sigma_k^{N+1} = (1-\alpha)\sigma_k^N + \rho(x_{N+1} - \mu_k^{N+1})(x_{N+1} - \mu_k^{N+1})^T \quad (2.7)$$

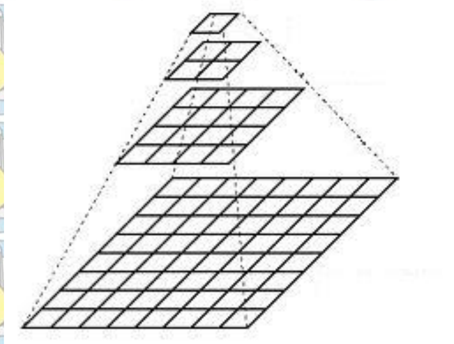
$$\rho = \alpha\, \eta(X_{N+1}, \mu_k^N, cov_k^{N+1}) \quad (2.8)$$

$$p(w_k|x_{n+1}) = \begin{cases} 1, & w_k\ adalah\ komponen\ gaussian\ yang\ match\ pertama\ kali \\ 0, & selain\ itu \end{cases} \quad (2.9)$$

Jika dari semua distribusi piksel yang dicek tidak memenuhi syarat (*unmatch*) maka komponen gaussian yang terakhir akan dilakukan *update*. *Update* dilakukan dengan mengubah rata-rata (μ) dengan nilai piksel yang sedang dicek dan mengubah nilai variasi (σ^2) dengan nilai tinggi dan nilai *weight* (w) dengan nilai yang rendah.

2.2 Gaussian Pyramid

Gaussian pyramid digunakan untuk melakukan reduksi resolusi citra. Citra yang tereduksi resolusinya akan berkurang menjadi seperempat dari resolusi awal. Hal ini dilakukan untuk mempercepat proses perhitungan. Ilustrasi *gaussian pyramid* dapat dilihat pada Gambar 2.1.



Gambar 2.1 Ilustrasi *Gaussian Pyramid* [3]

Gaussian pyramid dilakukan dengan dua operasi, yaitu *smoothing* dan *down sampling*. *Smoothing* dilakukan dengan menggunakan filter 5x5. *Smoothing* dilakukan dengan menggunakan Persamaan (2.10).

$$g_1 = w * g_0 \tag{2.10}$$

Dimana w adalah filter 5x5. Detail persamaan setiap piksel dilakukan menggunakan Persamaan (2.11).

$$g_1(i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) \, . \, g_0\,(i+m, j+n) \tag{2.11}$$

Setelah mendapatkan citra yang telah di *smoothing*, langkah selanjutnya adalah melakukan *down sampling*. *Down* *sampling* dilakukan untuk mengubah resolusi citra asli menjadi resolusi yang lebih kecil. *Down sampling* dilakukan dengan Persamaan (2.12).

$$g_2(i,j) = g_1(2i,2j) \tag{2.12}$$

Untuk melakukan reduksi, dilakukan menggunakan dua proses tersebut, yaitu *smoothing* dan *down sampling*. Perhitungan dapat dilakukan dengan menggabungkan ke dua persamaan tersebut menjadi Persamaan (2.13) [4].

$$g_1(i,j) = \sum_{m=-2}^{2}\sum_{n=-2}^{2} w(m,n) \,.\, g_0(2i+m, 2j+n) \tag{2.13}$$

2.3 Probabilitas Distribusi Gaussian

Probabilitas distribusi gaussian adalah sebuah metode untuk menghitung probabilitas dari suatu data. Probabilitas dilakukan dengan menghitung nilai rata-rata dan standar deviasi dari suatu data. Persamaan umum probabilitas distribusi gausian dapat dilihat pada Persamaan (2.14) [5].

$$p = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{2.14}$$

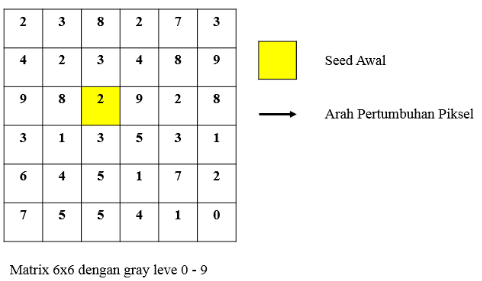
Pada kasus tugas akhir ini, probabilitas gaussian digunakan untuk mendapatkan probabilitas warna piksel. Setiap piksel dihitung nilai probabilitas R,G,B. Setelah dilakukan perhitungan probabilitas R,G,B selanjunya probabilitas tersebut dikalikan sehingga mendapatkan nilai probabilitas piksel. Perhitungan probabilitas setiap piksel dapat dilihat pada Persamaan (2.15).

$$p(I(x,y)) = \prod_{i\,\in\{R,G,B\}} p_i(I_i(x,y)) \tag{2.15}$$

2.4 Region Growing

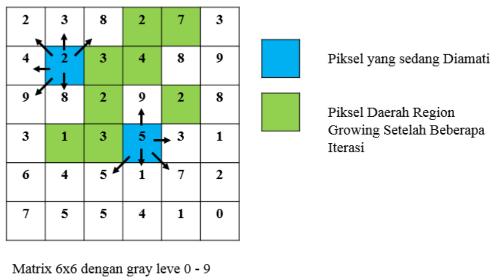
Region Growing adalah metode untuk melakukan segmentasi citra. Pendekatan dasar adalah dengan memulai titik yang sudah diinisialisasi (*seed*) dan dari titik tersebut dilakukan menumbuhkan daerah dengan cara menambahkan tetangga piksel dari *seed* yang mempunyai kesamaan dengan *seed* [6].

Untuk metode ini, dibutuhkan aturan yang mengatur mekanisme tumbuhnya *seed* dan suatu aturan lain yang menguji kehomogenan dari *region* setelah satu tahap tumbuh selesai. Pertumbuhan *region* dimulai dari *seed* awal dengan menambahkan tetangga piksel (menggunakan 8-tetangga) yang serupa untuk menumbuhkan *region*. Ketika suatu pertumbuhan *region* selesai, langkah selanjutnya adalah memilih *seed* baru dan melakukan *region growing* kembali. Proses tersebut dilakukan hingga semua piksel berhasil dikelompokkan dalam beberapa *region*. Ilustrasi *region growing* dapat dilihat pada Gambar 2.2 dan Gambar 2.3.



2	3	8	2	7	3
4	2	3	4	8	9
9	8	2	9	2	8
3	1	3	5	3	1
6	4	5	1	7	2
7	5	5	4	1	0

Gambar 2.2 *Seed* Awal *Region Growing*



Gambar 2.3 Piksel yang diamati dan *Region*

2.5 Daubachies 4 Wavelet

Wavelet adalah fungsi matematika yang membagi data menjadi beberapa komponen frekuensi yang berbeda-beda dan menganalisis setiap komponen tersebut dengan menggunakan resolusi yang sesuai dengan skalanya. Transformasi *wavelet* mendekomposisi signal kedalam *frequency bands* dengan memproyeksikan signal kedalam set fungsi dasar [7]. Pada citra, transformasi *wavelet* adalah transformasi yang melakukan filter terhadap suatu masukan. Filter yang digunakan dalam *wavelet* adalah *high pass filter* dan *low pass filter*. Pada daubachies 4, terdapat dua koefisien yang digunakan untuk melakukan *filter*, yaitu *scaling function coefficient* dan *wavelet function coefficient*. Dimana *scaling function coeficient* adalah koefisien yang digunakan dalam melakukan *low pass filter*, sedangkan *wavelet function coeficient* digunakan dalam melakukan *high pass filter* . Persamaan *scaling function coeficient* dapat dilihat pada Persamaan (2.16).

$$
\begin{aligned}
h_0 &= \frac{1+\sqrt{3}}{4\sqrt{2}} \\
h_1 &= \frac{3+\sqrt{3}}{4\sqrt{2}} \\
h_2 &= \frac{3-\sqrt{3}}{4\sqrt{2}} \\
h_3 &= \frac{1-\sqrt{3}}{4\sqrt{2}}
\end{aligned} \tag{2.16}
$$

Wavelet function coeficient dapat dilihat pada Persamaan (2.17).

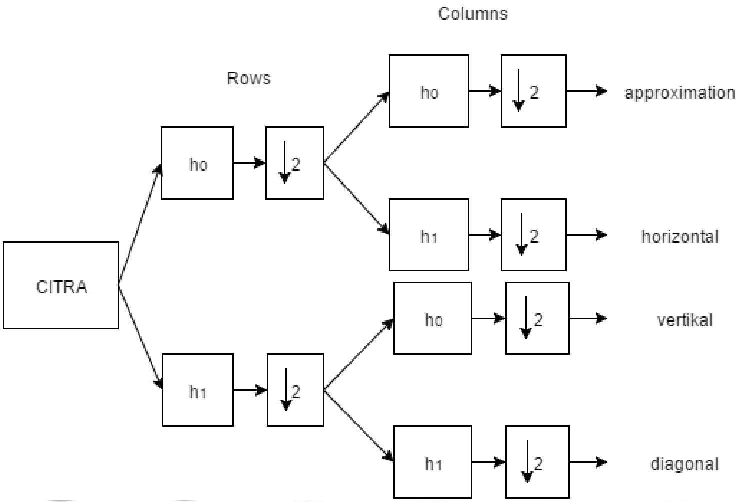
$$
\begin{aligned}
g_0 &= \frac{1-\sqrt{3}}{4\sqrt{2}} \\
g_1 &= \frac{-3+\sqrt{3}}{4\sqrt{2}} \\
g_2 &= \frac{3+\sqrt{3}}{4\sqrt{2}} \\
g_3 &= \frac{-1-\sqrt{3}}{4\sqrt{2}}
\end{aligned} \tag{2.17}
$$

Tahapan dari proses *filter* adalah melakukan *filter* terhadap baris citra terlebih dahulu. Dilakukan *high pass filter* dan *low pass filter*, dimana hasil dari proses ini adalah dua citra, yaitu citra *high pass* dan citra *low pass*. Ilustrasi tahap *filter* dapat dilihat pada Gambar 2.4.

$$\begin{matrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \end{matrix} \quad . \quad \begin{matrix} s_1 \\ s_0 \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_3 \\ s_2 \end{matrix}$$

Gambar 2.4 Ilustrasi *Low Pass Filter*

Citra yang dilakukan *filter* akan ter-reduksi menjadi setengah. Dilanjutkan dengan melakukan *filter* terhadap kolom citra, dilakukan *high pass filter* dan *low pass filter*. Hasil dari proses ini adalah empat citra dengan satu citra approximasi, dan tiga citra detail. Citra detail yang didapat adalah detail horizontal, vertikal dan diagonal. Gambar 2.5 Adalah lustrasi dari proses *filter* yang dilakukan.



Gambar 2.5 Ilustrasi *Filter* yang dilakukan

Ilustrasi citra yang masukan dan keluar pada proses ini dapat dilihat pada Gambar 2.6 dan Gambar 2.7.

Gambar 2.6 Citra Masukan

Gambar 2.7 Citra Keluaran Hasil *Wavelet*

2.6 Normalisasi Min-Max

Normalisasi adalah sebuah proses untuk mengubah suatu data ke dalam rentang nilai tertentu. Tujuannya adalah untuk menghindari persebaran data yang terlalu jauh sehingga sebuah variabel tidak mendominasi terhadap variabel lain. Salah satu jenis normalisasi adalah normalisasi skala. Pada normalisasi, skala rentang yang umum digunakan yaitu 0 hingga 1. Rumus umum skala adalah sebagai dapat dilihat pada Persamaan (2.18).

$$Y_{new} = \frac{(Y - Y_{min})(Y_{newmax} - Y_{newmin})}{Y_{max} - Y_{min}} + Y_{newmin} \qquad (2.18)$$

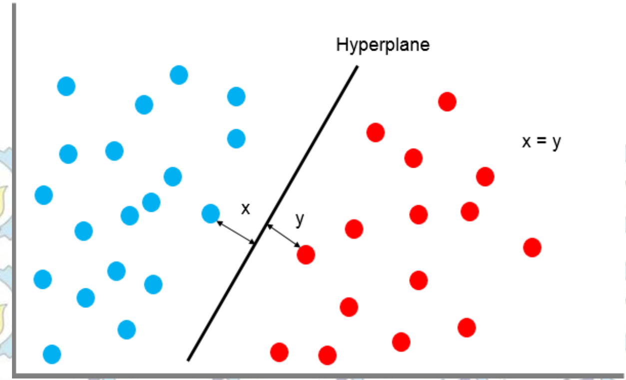
Variabel Y adalah nilai yang akan dinormalisasi. Variabel Y_{min} dan Y_{max} adalah nilai minimum dan maximum pada nilai-nilai atribut dimana Y berada. Variabel Y_{newmin} dan Y_{newmax} adalah nilai minimum dan maximum yang diinginkan. Apabila rentang yang digunakan adalah 0 hingga 1 maka rumus diatas menjadi Persamaan (2.19) [8].

$$Y_{new} = \frac{(Y - Y_{min})}{Y_{max} - Y_{min}} \qquad (2.19)$$

2.7 Support Vector Machines

Support vector machines adalah metode klasifikasi yang mengklasifikasikan dua kelas, yaitu kelas +1 dan -1. Pada metode klasifikasi *support vector machines*, dibentuk suatu *hyperplane*. *Hyperplane* adalah garis pemisah yang memisahkan dua kelas yang berbeda. Dalam metode *support vector machines* dikenal istilah *margin*. *Margin* adalah jarak antara kelas +1 dengan kelas -1 yang paling dekat, pada *support vector machines* dicari *margin* terpanjang antara dua kelas tersebut. Ilustrasi *support vector machines* dapat dilihat pada Gambar 2.8.

Diberikan masukan berupa data belajar $(x_{1,}x_{2,}x_3,\ldots,x_n)$ dan masing-masing kelas dianotasikan $y_i \in \{-1,+1\}$ untuk i = 1,2,3,…,l , dimana l adalah banyaknya data. Fungsi *hyperplane* dibuat dengan Persamaan (2.20). Dalam mencari nilai w dan b yang optimal, dilakukan dengan Persamaan (2.21) [9].



Gambar 2.8 Ilustrasi *Support Vector Machines*

$$w \, . \, x + b = 0 \tag{2.20}$$

$$\min \frac{1}{2} \, ||w||^2 + C \sum_{i=1}^{l} t_i \tag{2.21}$$

Persamaan diatas mempunyai variabel C, dimana variabel tersebut adalah konstanta nilai pinalti dari kesalahan klasifikasi. Pencarian nilai *w* dan *b* dilakukan dengan batasan yang ditulis menggnakan Persamaan (2.22).

$$y_i(wx_i + b) + \ t_i \ \geq 1 \tag{2.22}$$

Persamaan (2.21) dilakukan untuk mencari nilai *w* dan *b* yang optimum. Fungsi tujuan Persamaan (2.21) berbentuk kuadrat. Untuk menyelesaikannya, bentuk tersebut ditransformasi kedalam bentuk *dual space*. Persamaan *dual space* dapat ditulis menggunakan Persamaan (2.23).

$$\max \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j {x_i}^T x_j \tag{2.23}$$

Dengan batasan pada Persamaan (2.24).

$$\alpha_i \geq 0 , \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{2.24}$$

Untuk mencari nilai α_i, digunakan *quadratic programming*. Setelah mendapatkan nilai α_i, persamaan *hyperplane* dilakukan dengan Persamaan (2.25).

$$f = w^T z + b = \sum_{i=1}^{s} \alpha_i y_i {x_i}^T z + b \tag{2.25}$$

Dimana z adalah data masukan. Pada banyak kasus, data yang diklasifikasikan tidak bisa langsung dipisahkan dengan garis yang linear. Oleh karena itu, digunakan metode kernel untuk mengatasi permasalahan tersebut. Dengan metode kernel, suatu data x di *input space* dimapping ke fitur *space* F dengan dimensi yang lebih tinggi. Salah satu kernel yang biasa dipakai adalah kernel RBF dan *Polynomial*. Persamaan kernel RBF dan *Polynomial* berurutan dapat dilihat pada Persamaan (2.26) dan Persamaan (2.27).

$$k(x, y) = \exp(-\gamma |x - y|^2) \tag{2.26}$$

$$k(x, y) = (\gamma \langle x^T y \rangle + r)^p \tag{2.27}$$

Penggunaan fungsi kernel mengubah persamaan *training*. Persamaan dapat dilihat pada Persamaan (2.28).

$$\max \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j k(x_i, x_j) \tag{2.28}$$

Persamaan *hyperplane* diubah menjadi Persamaan (2.29).

$$f = \sum_{i=1}^{s} \alpha_i y_i \, k(x_i, z) + b \tag{2.29}$$