Implementasi Algoritma *Canny* dan *Backpropagation* dalam Pengenalan Pola Rumah Adat

Asep Nana Hermana[1], Meikel Sandy Juerman[1]

Jurusan Teknik Informatika, Fakultas Teknologi Industri
Institut Teknologi Nasional Bandung

asep_nana_h@yahoo.com[1], san.jang10@gmail.com[1]

ABSTRAK

Pengenalan pola rumah adat adalah suatu cara yang digunakan untuk membedakan antara satu pola rumah adat dengan pola rumah adat lainnya. Dalam penerapannya digunakan 2 algoritma yaitu algoritma Canny dan Backpropagation. Algoritma Canny digunakan untuk proses pengolahan citra dimana citra rgb akan diubah menjadi citra biner. Sedangkan backpropagation adalah algoritma pembelajaran terawasi yang terdiri dari beberapa layer (multilayer). Citra yang dikenali terlebih dahulu diproses menjadi citra biner, kemudian disimpan dalam bentuk matriks yang digunakan sebagai masukan untuk pelatihan. Hasil pelatihan diuji untuk mengenali pola objek diluar pelatihan. Tingkat keberhasilan pengujian terhadap 125 citra latih sebesar 97.6 %, sedangkan untuk pengujian terhadap 75 citra uji sebesar 50.67 %. Dengan demikian semakin banyak citra latih maka semakin tinggi tingkat keberhasilan dikenali.

Kata Kunci: Multimedia, Pengolahan Citra, Rumah Adat, *Canny* , *Backpropagation*

ABSTRACT

Traditional house pattern recognition is a way to distinguish between a traditional house pattern with the other. Application using two algorithms, Canny algorithm and backpropagation. Canny algorithm used for image processing in which RGB image is converted into a binary image. And backpropagation is a supervised learning algorithm consisting of several layers (multilayer). The image will be recognized first processed into a binary image, then stored in the form of a matrix that will be used as input for training. The results of the training will be tested to identify patterns of objects outside the training. The success rate of 125 train images coached by 97.6%, while for the testing of 75 test images by 50.67%. Therefore the more the image of training the higher the success recognized.

Keywords: Multimedia, *Image processing, Traditional House, Canny, Backpropagation*

Latar Belakang

Rumah adat merupakan salah satu kebudayaan dari setiap daerah di Indonesia. Setiap rumah adat ini memiliki keunikan dan keindahan tersendiri. Sehingga sangat disayangkan apabila kebudayaan ini dilupakan oleh masyarakat di Indonesia.

Rumah adat memiliki bentuk pola yang sangat kompleks. Disamping itu juga rumah adat memiliki modifikasi yang beragam untuk satu rumah adat, sehingga untuk membandingkan antar 1 macam rumah adat pun sangat sulit.

Dibutuhkan suatu metode untuk mereduksi kompleksitas dari pola rumah adat. Metode yang digunakan adalah deteksi tepian. Salah satu algoritma deteksi tepi yaitu algoritma *canny*. Algoritma *canny* adalah suatu algoritma deteksi tepi yang dilakukan dengan pendekatan konvolusi terhadap fungsi matriks gambar dan operator *Gaussian.* Kemudian untuk membandingkan pola yang tidak beraturan digunakan suatu jaringan syaraf tiruan. Salah satu model jaringan syaraf tiruan adalah *backpropagation.* Algoritma *backpropagation* digunakan karena memiliki jaringan terdiri dari fungsi aktivasi dan banyak *layer* sehingga mempermudah proses pengenalan.

Melihat permasalahan tersebut, dirancanglah sebuah aplikasi menggunakan suatu metode deteksi tepi *canny* dan metode *backpropagation*.

Rumusan Masalah

Rumusan masalah dalam pembuatan aplikasi deteksi rumah adat, yaitu :

a. Bagaimana memanfaatkan deteksi tepi agar dapat mengolah dan mengenali tepi suatu objek.
b. Bagaimana menerapkan metode backpropagation dalam menentukan pola suatu objek.
c. Bagaimana menggabungkan metode *Canny* dan *Backpropagation*.

Tujuan

Tujuan dari penelitian ini adalah menggabungkan algoritma *Canny* dan algoritma *Backpropagation*.

Batasan Masalah

Batasan masalah yang dipakai dalam pembuatan aplikasi deteksi rumah adat adalah sebagai berikut :

a. Rumah adat yang dideteksi adalah bagian tampak depan.
b. Sampel dibatasi pada 5 rumah adat di Indonesia yaitu Tongkonan, Sasak, Honai, Rumah Gadang, Minahasa. Setiap sampel terdiri dari 5 macam gambar.
c. Format gambar jpg.
d. Aplikasi yang digunakan berbasis desktop.
e. Pengujian dilakukan berdasarkan rotasi atau besar sudut objek.

Pengolahan Citra Digital[1]

Citra digital adalah citra $f(x,y)$ yang telah dilakukan digitalisasi baik koordinat area maupun *brightness* level. Nilai f di koordinat (x,y) menunjukkan *brightness* atau *grayness* level dari citra pada titik tersebut.

Citra digital diwakili oleh sebuah matrik yang terdiri dari M kolom dan N baris dimana perpotongan antara kolom dan baris disebut piksel (pixel = picture element), yaitu elemen terkecil dari sebuah citra. Pengolahan citra digital merupakan proses yang bertujuan untuk menganalisis citra dengan bantuan komputer. Atau dapat juga diartikan sebagai proses memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau komputer.

Grayscale[2]

Grayscale merupakan proses pengolahan citra dengan mengubah nilai-nilai piksel awal citra menjadi sebuah citra keabuan. Citra keabuan adalah citra yang setiap pikselnya mengandung satu layer dimana nilai intensitasnya berada pada interval 0-255, sehingga nilai-nilai piksel

pada citra keabuan tersebut dapat direpresentasikan dalam sebuah matriks yang dapat memudahkan proses perhitungan pada operasi berikutnya. Rumus menghitung *grayscale* :

$$Gray = (R + G + B)/3 \quad \text{.....(1)}$$

Keterangan : R = Red
G = Green
B = Blue

Deteksi Tepi Canny[3]

Salah satu operator deteksi tepi modern adalah deteksi tepi dengan operator Canny. Deteksi tepi *canny* ditemukan oleh Marr dan Hildreth yang meneliti pemodelan persepsi visual manusia. Ada beberapa kriteria pendeteksi tepian paling optimum yang dapat dipenuhi oleh operator *canny* :

a. Mendeteksi dengan baik (kriteria deteksi)
   Kemampuan untuk meletakkan dan menandai semua tepi yang ada sesuai dengan pemilihan parameter-parameter konvolusi yang dilakukan. Sekaligus juga memberikan fleksibilitas yang sangat tinggi dalam hal menentukan tingkat deteksi ketebalan tepi sesuai yang diinginkan.
b. Melokalisasi dengan baik (kriteria lokalisasi)
   Dengan Canny dimungkinkan dihasilkan jarak yang minimum antara tepi yang dideteksi dengan tepi yang asli.
c. Respon yang jelas (kriteria respon)
   Hanya ada satu respon untuk tiap tepi, sehingga mudah dideteksi dan tidak menimbulkan kerancuan pada pengolahan citra selanjutnya. Pemilihan parameter deteksi tepi Canny sangat mempengaruhi hasil dari tepian yang dihasilkan. Beberapa parameter tersebut adalah nilai standart deviasi Gaussian dan nilai ambang (*threshold*).

Langkah-langkah dalam melakukan deteksi tepi :

1. *Smoothing* merupakan proses mengaburkan gambar untuk menghilangkan *noise*. Pada tahap ini digunakan *Gaussian filter* dengan standar deviasi σ = n. Filter harus dirancang terlebih dahulu berdasarkan pada ordo matriks dan nilai standar deviasi. Semakin besar nilai standar deviasi maka semakin halus pula efek yang dihasilkan dari pemfilteran. Persamaan Gaussian :

$$G(i,j) = \frac{1}{2\pi\sigma^2} . e^{-\frac{(i-u)^2+(j-v)^2}{2\sigma^2}} \quad \text{...(2)}$$

   Keterangan :
   e = 2.71 (konstanta euler)
   σ = standar deviasi (sigma)
   π = 3.14 (pi)

2. Tepian harus ditandai pada gambar yang memiliki gradient yang besar. Untuk itu digunakan salah satu operator seperti operator Robert, Prewit atau Sobel dengan melakukan pencarian secara horizontal (G_x) dan vertikal (G_y). Gambar 1 menggunakan operator Sobel :

$$\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \quad \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

G_x G_y

(a) (b)

Gambar 1. mask horizontal(a), mask vertikal(b)

Hasil dari kedua operator digabungkan untuk mendapatkan hasil gabungan tepi vertikal dan horizontal dengan rumus :

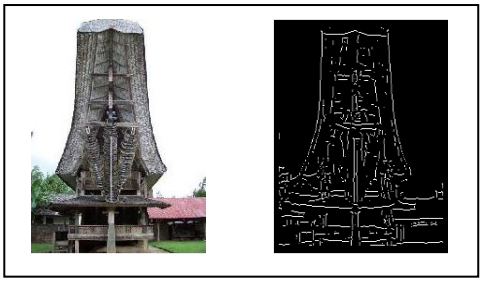
$$G = \sqrt{{G_x}^2 + {G_y}^2} \quad \text{.....(3)}$$

Kemudian menentukan arah tepian yang ditemukan dengan menggunakan rumus :

$$\theta = arctan\left(\frac{G_y}{G_x}\right) \quad \text{.....(4)}$$

Selanjutnya membagi ke dalam 4 warna sehingga garis dengan arah yang berbeda memiliki warna yang berbeda. Derajat 0 – 22,5 dan 157,5 – 180 berwarna kuning. Derajat 22,5 – 67,5 berwarna hijau. Derajat 67,5 – 157,5 berwarna merah.

3. Memperkecil garis tepi yang muncul dengan menerapkan *non maximum suppression* sehingga menghasilkan garis tepian yang lebih ramping.
4. Langkah terakhir adalah binerisasi dengan menerapkan dua buah *thresholding* yaitu *high threshold* dan *low threshold*. Gambar 2 menunjukkan bentuk citra sebelum pemprosesan (a) dan sesudah pemprosesan dengan nilai standar deviasi 2 dan kernel Gaussian 7x7.



(a) (b)

Gambar 2. Citra asli(a), citra hasil *Canny*(b)

Jaringan Syaraf Tiruan[4]

Jaringan Syaraf Tiruan dibuat pertama kali pada tahun 1943 oleh neurophysiologist Waren McCulloch dan logician Walter Pits, namun teknologi yang tersedia pada saat itu belum memungkinkan mereka berbuat lebih jauh.

Jaringan Syaraf Tiruan adalah paradigma pemrosesan suatu informasi yang terinspirasi oleh sistim sel syaraf biologi, sama seperti otak yang memproses suatu informasi.

Jaringan syaraf tiruan ditentukan oleh 3 hal :

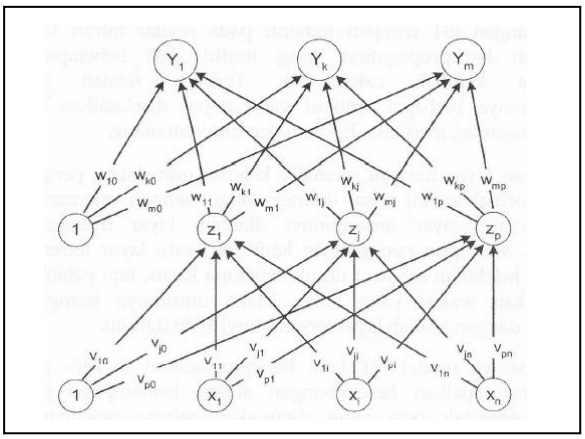
1. Pola hubungan antar neuron (disebut arsitektur jaringan)
2. Metode untuk menentukan bobot penghubung (disebut metode learning/training).
3. Fungsi aktivasi.

Backpropagation[5]

Perambatan galat mundur (*Backpropagation*) adalah sebuah metode sistematik untuk pelatihan *multiplayer* jaringan saraf tiruan. Metode ini memiliki dasar matematis yang kuat, obyektif dan algoritma ini mendapatkan bentuk persamaan dan nilai koefisien dalam formula dengan meminimalkan jumlah kuadrat galat error melalui model yang dikembangkan (*training set*).

a. Arsitektur Backpropagation

Backpropagation memiliki beberapa unit yang ada dalam satu atau lebih layer tersembunyi. Gambar 3 adalah arsitektur backpropagation dengan n buah masukan (ditambah sebuah bias), sebuah layer tersembunyi yang terdiri dari p unit (ditambah sebuah bias), serta m buah unit keluaran.



Gambar 3. Arsitektur Backpropagation

V_{ji} merupakan bobot garis dari unit masukan x_i ke unit layer tersembunyi z_j (v_{j0} merupakan bobot garis yang menghubungkan bias di unit masukan

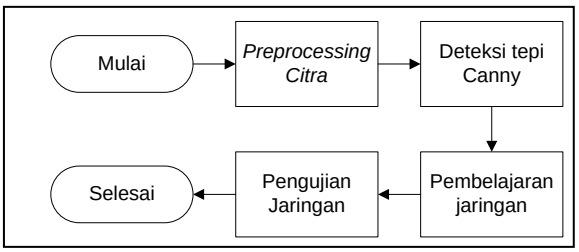
ke unit layer tersembunyi z_j). W_{kj} merupakan bobot dari unit layer tersembunyi z_j ke unit keluaran y_k (w_{k0} merupakan bobot dari bias di layer tersembunyi ke unit keluaran z_k).

b. Algoritma Backpropagation

Pelatihan suatu jaringan dengan algoritma *backpropagation* meliputi dua tahap : perambatan maju dan perambatan mundur. Selama perambatan maju, tiap unit masukan (*xi*) menerima sebuah masukan sinyal ini ke tiap-tiap lapisan tersembunyi z1,…..,zp. Tiap unit tersembunyi ini kemudian menghitung aktivasinya dan mengirimkan sinyalnya (*zj*) ke tiap unit keluaran. Tiap unit keluaran (*yk*) menghitung aktivasinya (*yk*) untuk membentuk respon pada jaringan untuk memberikan pola masukan. Selama pelatihan, tiap unit keluaran membandingkan perhitungan aktivasinya *yk* dengan nilai targetnya *tk* untuk menentukan kesalahan pola tersebut dengan unit itu. Berdasarkan kesalahan ini, faktor δ*k (k = 1,..,m) dihitung.* δ*k* digunakan untuk menyebarkan kesalahan pada unit keluaran *yk* kembali ke semua unit pada lapisan sebelumnya (unit-unit tersembunyi yang dihubungkan ke *yk*). Juga digunakan (nantinya) untuk meng*update* bobot-bobot antara keluaran dan lapisan tersembunyi. Dengan cara yang sama, faktor (*j = 1,…,p*) dihitung untuk tiap unit tersembunyi *zj*. Tidak perlu untuk menyebarkan kesalahan kembali ke lapisan masukan, tetapi δ*j* digunakan untuk mengupdate bobot-bobot antara lapisan tersembunyi dan lapisan masukan. Setelah seluruh faktor δ ditentukan, bobot untuk semua lapisan diatur secara serentak. Pengaturan bobot w*jk* (dari unit tersembunyi *zj* ke unit keluaran y*k*) didasarkan pada faktor *δk* dan aktivasi *zj* dari unit tersembunyi *zj*. didasarkan pada faktor *δj* dan dan aktivasi *xi* unit masukan.

Perancangan Sistem

Pada perancangan sistem ada beberapa proses yang dijalankan yaitu *preprocessing* citra, deteksi tepi *canny*, proses pembelajaran, sampai proses pengujian jaringan seperti yang ditunjukkan pada Gambar 4.

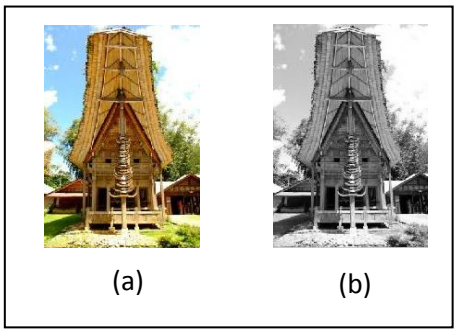


Gambar 4. Alur Sistem Keseluruhan

Input yang digunakan berupa citra rumah adat. Citra rumah adat adalah citra RGB. Pada proses *preprocessing*, citra RGB diubah menjadi citra keabuan oleh proses *grayscale* sebagai masukan untuk proses *canny*. Prose Prose *grayscale* diselesaikan menggunakan persamaan (1). Nilai R, G, B dari setiap piksel citra terlebih dahulu diekstraksi kemudian dibagi 3. Misalkan nilai R(1,1) = 137, nilai G(1,1) = 190, nilai B (1,1) = 255.

Grayscale (1,1) = 137 + 190 + 255 / 3
= 194

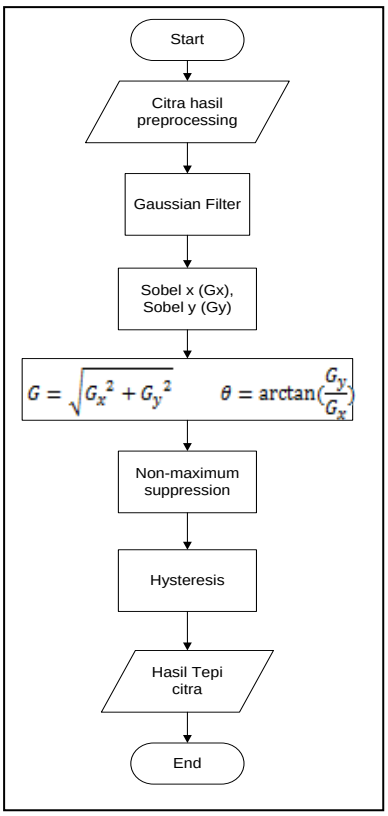
Maka didapatkanlah nilai keabuan dari citra (1,1) sebesar 194. Demikian seterusnya untuk seluruh piksel citra. Sehingga menghasilkan citra *grayscale* seperti yang ditunjukkan pada Gambar 5.



Gambar 5. (a) Citra RGB, (b) citra *grayscale*

Canny

Penerapan operasi deteksi tepi *Canny* dimaksudkan agar menghasilkan tepi dari objek dalam suatu citra digital. Proses ini dilakukan bertujuan untuk menandai bagian detail dari objek dan memperbaiki detail citra yang kabur karena error atau adanya efek akuisisi. Sehingga mempermudah dalam proses Jaringan Syaraf Tiruan. Tahapan dalam mendeteksi tepi *canny* yaitu pada Gambar 6.



Gambar 6. Tahap proses algoritma *Canny*

Pada *canny* citra pertama akan dihaluskan dengan metode *Gaussian filter*. Dalam proses penghalusan terlebih dahulu kernel dirancang berdasarkan pada ordo matriks m x n dan nilai standar deviasi σ^2 menggunakan persamaan (2) kemudian dikonvolusi dengan matriks citra. Contoh pembuatan kernel sebagai berikut.

Diketahui $\sigma^2 = 1$, $e = 2.718281828459$

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3

= elemen matriks posisi (i,j)

= indeks tengah dari matriks kernel (u,v)

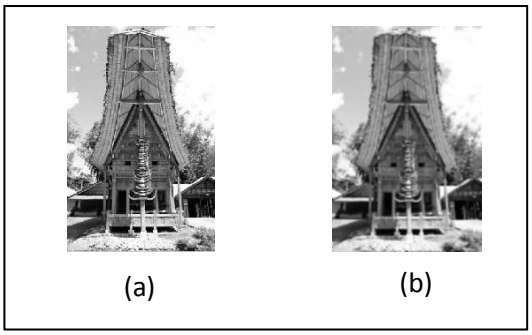
Untuk posisi i,j = 1,1 :

$$G(1,1) = \frac{1}{2 * 3.14 * 1^2} . 2.718281828459^{-\frac{(1-2)^2+(1-2)^2}{2*1^2}}$$

$$G(1,1) = \frac{1}{6.28} . 2.718281828459^{-1}$$

$G(1,1)$ = 0.0585

Contoh hasil proses penghalusan dari citra rumah adat Toraja menggunakan *Gaussian Filter* diperlihatkan pada Gambar 7.



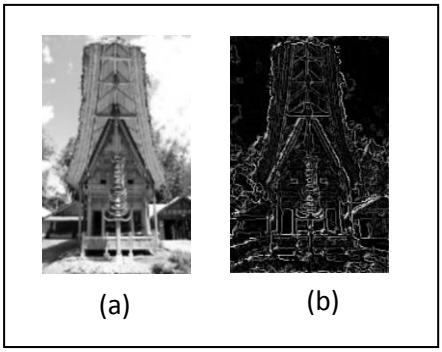
Gambar 7. (a) Citra *grayscale*, (b) citra hasil penghalusan

Citra yang telah dihaluskan selanjutnya dilakukan pencarian gradient terhadap arah horizontal (Gx) dan vertikal (Gy) menggunakan operator Sobel. Hasil dari operasi sobel ditunjukkan pada Gambar 8. Contoh dari gambar rumah adat Toraja yang diproses Sobel menggunakan persamaan (3) berikut.

G_x = (182)(-1) + (182)(-2) + (183)(-1) + (182)(1) + (182)(2) + (183)(1) = 0

G_y = (182)(1) + (182)(2) + (182)(1) + (183)(-1) + (183)(-2) + (183)(-1) = - 4

$G = \sqrt{0^2 + -4^2}$ = |0|+|-4| = 4



Gambar 8. (a) Citra hasil penghalusan, (b) citra hasil operasi Sobel

Setelah itu, menerapkan *Non maximum suppression* untuk menghasilkan garis tepian yang ramping. Berdasarkan penentuan arah pada Sobel pada Gambar 27 (b), diketahui $\theta = 0^o$ dengan contoh matriks sebagai berikut :

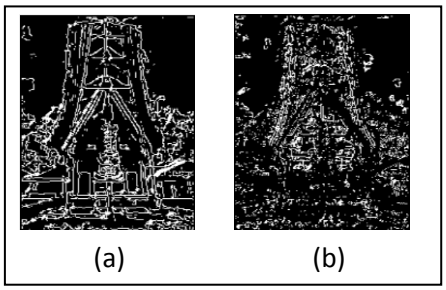
4	5	6	4	5
10	12	14	10	13
4	4	3	5	5

Arah gradien

Keterangan :
6 = magnitudo (i_1,j_1)
14 = magnitude (i,j)
3 = magnitude (i_2,j_2)
Untuk setiap piksel (x,y) *do*:
Jika magnitude(i,j) < magnitude(i_1,j_1) atau magnitude(i,j) < magnitude(i_2,j_2)
Maka $G_n(i,j) = 0$
Else $G_n(i,j)$ = magnitude(i,j)
14 > 4 or 3 jadi 14 merupakan tepi asli.

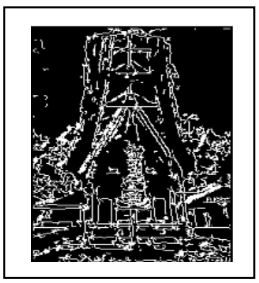
Terakhir menggunakan *Hysteresis by double thresholding* untuk memisahkan tepian yang tidak terhubung dengan tepian kuat. Tepi piksel lebih lemah dari *low threshold* akan ditekan dan tepi piksel antara *high threshold* dan *low threshold* ditandai sebagai piksel lemah.

Misalkan diatur nilai *high threshold* (H) = 80 dan *low threshold* (L) = 20 sehingga menghasilkan tepian seperti pada Gambar 9.



Gambar 9. (a) *Strong edge*, (b) *Weak edge*

Setelah tepian kuat dan lemah dipisahkan, maka didapatlah tepian yang sebenarnya seperti pada Gambar 10.



Gambar 10. Hasil Canny

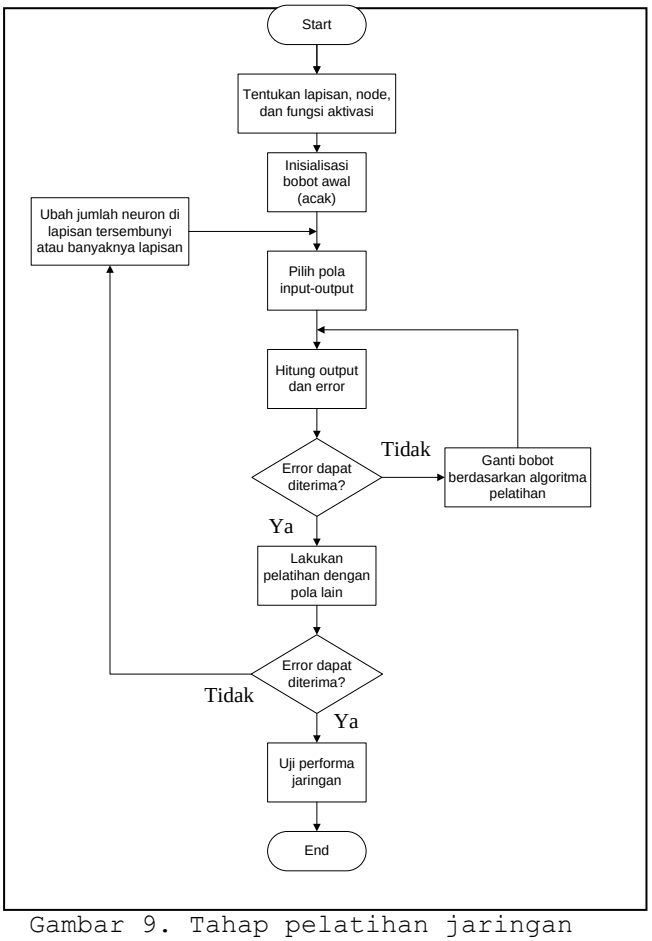
Backpropagation

a. Pembelajaran Jaringan

Proses akan dimulai dengan menentukan arsitektur jaringan yaitu menentukan parameter awal seperti *learning rate, epoch, max error,* dan jumlah lapisan yaitu lapisan input, lapisan tersembunyi serta lapisan output.

Kemudian bobot dan bias awal akan di*set* secara acak. Biasanya bobot awal diinisialisasi dengan nilai antara -0.5 sampai 0.5. Setelah itu, hasil deteksi tepi yang berupa nilai biner dimasukkan sebagai pola latih.

Hitung nilai *output* dan nilai *error*. Jika nilai *error* belum mencapai target maka proses pelatihan dengan mengubah bobot akan terus dilakukan , namun jika telah terpenuhi, maka proses pelatihan akan berhenti.



Gambar 9. Tahap pelatihan jaringan

Jaringan hasil pelatihan siap diuji untuk proses pengenalan. Untuk lebih jelas, perhatikan Gambar 11.

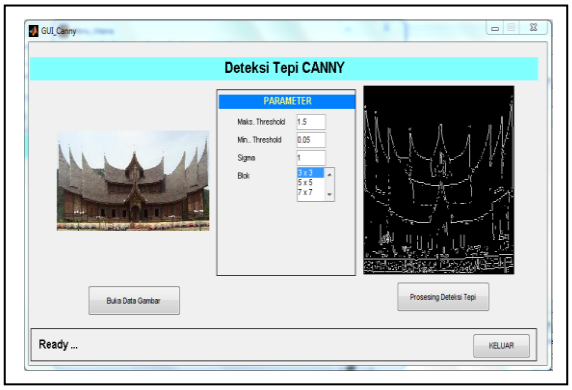
Spesifikasi kebutuhan sistem

Setelah dilakukan perancangan, kemudian dilakukan implementasi sistem. Berikut merupakan spesifikasi dari perangkat keras dan perangkat lunak yang digunakan pada pembangunan sistem :

1. Spesifikasi perangkat keras yang digunakan :
   a. Laptop dengan spesifikasi Processor Intel® Core™2 Duo CPU, Memory : 2048 MB RAM, Harddisk : 320 GB
2. Spesifikasi perangkat lunak yang digunakan :
   a. Microsoft Windows7 32 bit
   b. Matlab R2009a

Implementasi *Canny*

Implementasi untuk deteksi tepi *canny* terdapat pada Gambar 13. Pada *interface canny*, terdapat beberapa parameter yang digunakan untuk mendapatkan hasil deteksi tepian yang diinginkan.



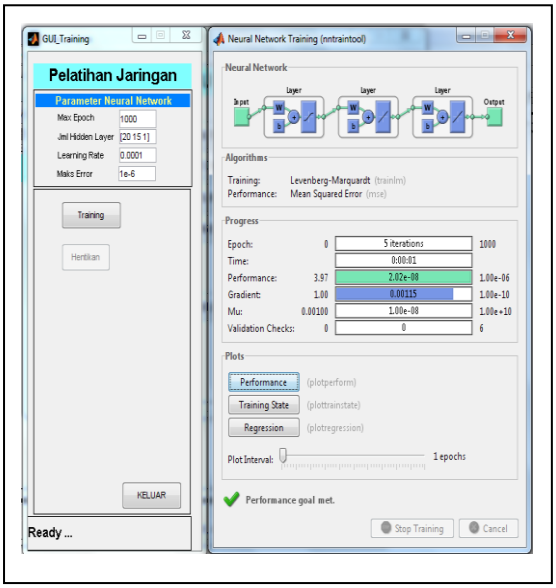
Gambar 13. Tampilan Deteksi Tepi *Canny*

Diantaranya yaitu nilai *high* dan *low threshold*, nilai standar deviasi (sigma), dan ukuran kernel *Gaussian* (blok).

Implementasi Backpropagation

a. Implementasi Pelatihan (*Training*)

Proses pelatihan pada aplikasi ini menggunakan *toolbox nntraintool* yang telah disediakan oleh Matlab. Informasi yang diberikan dari hasil pelatihan yaitu *max epoch* (jumlah iterasi yang dilakukan selama proses pelatihan), jumlah *hidden layer* (jumlah layer tersembunyi yang diberikan untuk proses pelatihan), *learning rate* (laju proses pelatihan), dan *max error* (nilai *error* yang dihasilkan setelah proses pelatihan selesai. Proses pelatihan menggunakan *traintool* terlihat pada Gambar 14.



Gambar 14. Tampilan Proses *Training*

b. Implementasi Pengenalan (Pengujian)

Interface untuk pengenalan terhadap citra uji diperlihatkan pada Gambar 15.

Gambar 15. Tampilan Proses *Pengujian*

Karena citra latih dideteksi tepi sebelum dilatih maka citra uji pun harus di deteksi tepiannya juga. Hasil dari pengenalan sangat dipengaruhi oleh nilai standar deviasi dan kernel Gaussian. Keluaran dari proses pengenalan adalah teks yang menyebutkan nama rumah adat yang dikenali.

Data Citra Latih

Data yang diambil untuk pelatihan terdiri dari 5 macam rumah adat yaitu rumah adat NTB, rumah adat Padang, rumah adat Sulawesi Utara, dan rumah adat Toraja. Total seluruh citra latih yaitu sebanyak 250 citra seperti yang diperlihatkan pada Tabel 2.

Tabel 2. Data Citra Latih

No.	Rumah Adat	Citra latih	Sudut rotasi	Jumlah
1.	NTB	Sasak1 Sasak2 Sasak3 Sasak4 Sasak5	70, 80, 90, 100, 110, 225, 315, 237, 270, 293	50
2.	Padang	Padang1 Padang2 Padang3 Padang4 Padang5	70, 80, 90, 100, 110, 225, 315, 237, 270, 293	50
3.	Papua	Papua1 Papua2 Papua3 Papua4 Papua5	70, 80, 90, 100, 110, 225, 315, 237, 270, 293	50
4.	Sulawesi Utara	Minahasa1 Minahasa2 Minahasa3 Minahasa4 Minahasa5	70, 80, 90, 100, 110, 225, 315, 237, 270, 293	50
5.	Toraja	Toraja1 Toraja2 Toraja3 Toraja4 Toraja5	70, 80, 90, 100, 110, 225, 315, 237, 270, 293	50

Pengujian Terhadap Citra Latih

Data yang telah dikumpulkan dilatih secara bertahap berdasarkan penambahan jumlah citra latih untuk mendapatkan jaringan terbaik. Pada tahap pertama digunakan citra latih sebanyak 125 citra, tahap kedua digunakan citra latih sebanyak 175, dan tahap ketiga digunakan citra latih sebanyak 250 citra. Pada proses pelatihan ada 2 parameter yang digunakan sebagai acuan pelatihan yaitu *learning*

rate, dan *MSE* seperti yang terlihat pada Tabel 3.

Tabel 3. Pengujian Citra latih

Jumlah citra latih	Jumlah dikenali	*Learning rate*	*MSE*	% berhasil
125	116	0.01	0.0285	92.8 %
125	110	0.02	0.0249	88 %
125	122	0.03	0.01	97.6 %
175	152	0.01	0.0374	86.9 %
175	164	0.02	0.0243	93.7 %
175	154	0.03	0.0271	88 %
250	193	0.01	0.0568	77.2 %
250	206	0.02	0.0547	82.4 %
250	198	0.03	0.0558	79.2 %

Pengujian Terhadap Citra Uji

Setelah citra dilatih dan didapat jaringan terbaik pada setiap perubahan jumlah citra latih, maka selanjutnya menguji citra uji.

Tabel 4. Perbandingan persentase keberhasilan

Jumlah citra latih	**Jumlah citra uji**	**Jumlah dikenali**	**% berhasil**
125	75	30	40 %
175	75	31	41.33 %
250	75	38	50.67 %

Citra uji yang digunakan adalah citra latih yang dirotasi dengan sudut yang berbeda seperti pada Tabel 4.

Kesimpulan

Kesimpulan yang diperoleh setelah melalui penelitian dan pengujian adalah sebagai berikut :

1. Persentase keberhasilan dalam pengenalan pola rumah adat terhadap citra latih sangat dipengaruhi oleh nilai *MSE*. Dari pelatihan terhadap 125 citra latih, dihasilkan persentase keberhasilan tertinggi sebesar 97.6% dengan nilai *MSE* 0.01. Dari pelatihan terhadap 175 citra latih, dihasilkan persentase keberhasilan tertinggi sebesar 93.7% dengan nilai *MSE* 0.0243. Dari pelatihan terhadap 250 citra latih, dihasilkan persentase keberhasilan tertinggi sebesar 82.4% dengan nilai *MSE* 0.0547.
2. Pengujian terhadap 125 citra latih dengan *learning rate* 0.03, *MSE* 0.01 menghasilkan persentase keberhasilan dikenali 97.6%. Pengujian terhadap 175 citra latih dengan *learning rate* 0.02, *MSE* 0.0243 menghasilkan persentase keberhasilan dikenali 93.7%. Pengujian terhadap 250 citra latih dengan *learning rate* 0.02, *MSE* 0.0547 menghasilkan persentase keberhasilan dikenali 82.4%.
3. Pengujian terhadap 75 citra uji dengan citra latih sebanyak 125 menghasilkan persentase keberhasilan dikenali 40%, pengujian terhadap 75 citra uji dengan citra latih sebanyak 175 menghasilkan persentase keberhasilan dikenali 41.33%, pengujian terhadap 75 citra uji dengan citra latih sebanyak 250 menghasilkan persentase keberhasilan dikenali 50.67% seperti yang ditunjukkan pada Tabel 4.

DAFTAR PUSTAKA

[1] Darma Putra. "*Pengolahan Citra Digital*". 2010. Penertbit ANDI.

[2] Hanif Al Fatta (2007). Konversi Format Citra RGB ke Format Grayscale menggunakan Visual Basic. STIMIK AMIKON Yogyakarta. Diakses pada tanggal 17 Oktober 2014 pada pukul 20.23.

[3] Batra Yudha Pratama. *Pendeteksian Tepi Pengolahan Citra* Digital. 2007

[4] T.Sutojo, S.Si, M.Kom, dkk (2010). Kecerdasan Buatan. ANDI, Yogyakarta.

[5] Muhammad Tonovan (2007). Pengenalan pola Geometri Wajah Menggunakan Jaringan Syaraf Tiruan Backpropagation. Universitas Islam Indonesia, Yogyakarta.