PERANCANGAN PROGRAM APLIKASI DETEKSI IRIS MATA UNTUK ABSENSI KARYAWAN MENGGUNAKAN METODE GABOR WAVELET

Zahedi; Eka Janitra

Jurusan Matematika, Fakultas Sains dan Teknologi, Binus University
Jl. KH. Syahdan No. 9, Palmerah, Jakarta Barat 11480.
zahedizahedi@binus.ac.id

ABSTRACT

To take the employees' absences in a company, a variety of invented tools or technologies are and already used, for example, attendance cards, fingerprints attendance tool, and attendance tool that uses facial recognition. Attendance tool with a facial recognition uses iris recognition of those employees. Iris code has many properties that hard to forge. Thus, the detection and iris recognition is one of the most secure and accurate for identification.

***Keywords**: absences, iris, code*

ABSTRAK

Untuk mengabsen karyawan di suatu perusahaan, berbagai macam alat atau teknologi yang sudah ditemukan telah digunakan, misalnya kartu absensi, alat absensi sidik jari, dan alat absensi yang menggunakan pengenalan wajah. Alat absensi dengan pengenalan wajah ini menggunakan pengenalan iris mata karyawan yang bersangkutan. Kode iris mata memiliki banyak properti yang sulit dipalsukan. Dengan demikian, pendeteksian dan pengenalan iris mata merupakan salah satu cara yang paling aman dan akurat untuk identifikasi.

***Kata kunci**: absensi, iris mata, kode*

PENDAHULUAN

Untuk mengabsen karyawan di suatu perusahaan berbagai macam alat atau teknologi sudah ditemukan dan digunakan, misalnya absensi karyawan menggunakan kartu absensi, menggunakan pengenalan sidik jari, dan yang menggunakan pengenalan wajah.

Cara mengabsen karyawan menggunakan kartu absensi adalah dengan memasukan kartu tersebut ke sebuah alat yang kemudian akan mencatat waktu masuk dan keluar seorang karyawan. Penggunaan kartu absensi ini sekarang ini paling banyak digunakan. Hal ini disebabkan murahnya alat yang digunakan. Namun, penggunan kartu ini memiliki kelemahan, yaitu seorang karyawan bisa mengabsenkan karyawan lainnya, dengan kata lain karyawan lain tersebut bisa tidak masuk kerja, namun tidak ada yang tahu. Hal ini bisa mengurangi produktivitas dari perusahaan.

Cara lain adalah dengan mengabsen karyawan menggunakan pengenalan sidik jari. Cara yang digunakan adalah dengan menyentuhkan jari (biasanya ibu jari) ke suatu alat yang kemudian akan membaca sidik jari tersebut, mengidentifikasi karyawan yang memiliki sidik jari tersebut, dan memasukkannya data-data yang diperlukan ke dalam *database* komputer. Penggunaan sidik jari ini juga memiliki kelemahan, yaitu bahwa seseorang bisa mengambil sidik jari seorang karyawan dengan atau tanpa sepengetahuannya dan menggunakannya. Hal ini juga memungkinkan untuk mengurangi produktivitas perusahaan tempat karyawan tersebut bekerja.

Absensi karyawan dengan menggunakan pengenalan wajah pun memiliki kelemahan yang mirip. Meskipun hampir tidak akan ada karyawan yang akan mengabsenkan seorang karyawan lain, karena akan memerlukan sebuah topeng, namun seseorang yang lain yang memiliki maksud dan tujuan tertentu yang lain daripada mengabsenkan seorang karyawan akan bisa menembus kelemahan sistem ini. Untuk mengatasi masalah tersebut dibutuhkan sistem lain yang lebih baik. Untuk saat seperti inilah absensi karyawan dengan menggunakan pengenalan iris mata dibutuhkan. Kode iris mata memiliki banyak properti yang sulit dipalsukan. Dengan demikian, pendeteksian dan pengenalan iris mata merupakan salah satu cara yang paling aman dan akurat untuk identifikasi.

METODE

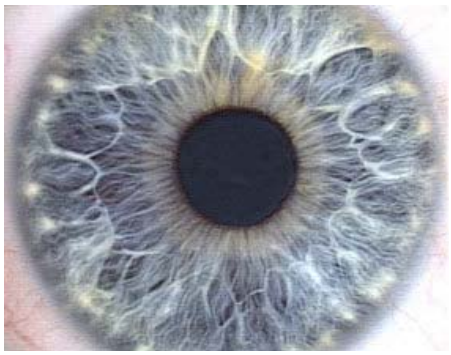
Kode Iris Mata

Iris merupakan suatu bagian mata yang memiliki pigmen. Pigmen pada iris ini berfungsi untuk memberikan warna pada mata seseorang seperti warna mata biru atau hijau pada orang Barat, dan warna hitam atau coklat pada orang Asia. Selain untuk memberikan warna pada mata, iris juga berfungsi untuk mengatur banyaknya sinar yang masuk ke dalam mata. Apabila kita menatap benda-benda yang memancarkan cahaya seperti lampu atau sinar matahari, maka otot-otot iris mengendur sehingga pupil akan mengecil dan cahaya yang masuk lebih sedikit. Sedangkan ketika kita dalam sebuah ruang gelap dengan sinar yang sedikit, otot-otot iris akan berkontraksi sehingga pupil akan membesar dan cahaya yang diterima lebih banyak. Perilaku iris yang seperti ini analog dengan diafragma iris pada kamera.

Metode Edge Detection

Edge detection adalah sebuah masalah kepentingan fundamental dalam analisis gambar (*images*). Dalam suatu gambar (*image*), hal yang kita sebut *edge* memberikan karakteristik batas dari

suatu benda dan maka dari itu berguna untuk segmentasi, registrasi, dan identifikasi suatu benda atau objek dalam suatu kejadian.



Gambar 1. Iris dan pupil mata.
Sumber: Carr et al, 2011

Dalam kasus yang ideal, menggunakan sebuah *edge detector* ke sebuah gambar dapat mengurangi jumlah data yang harus diproses dan memungkinkan menyaring informasi yang dianggap kurang relevan, sambil tetap menyimpan data yang penting dari gambar tersebut. Jika langkah pendeteksian berhasil, tugas untuk mentranslasikan gambar aslinya dapat disederhanakan. Sayangnya,tidak mungkin untuk selalu mendapatkan hasil yang ideal, terutama dari gambar kehidupan nyata yang cukup kompleks. Walaupun seperti itu, *edges* tetap berperan penting dalam banyak aplikasi *image processing.*

Gambar dari kehidupan nyata biasanya dipengaruhi oleh satu atau beberapa efek berikut: (1) fokus yang tidak jelas yang disebabkan *depth-of-field* dan *point spread function* yang terbatas; (2) *Penumbral blur* yang disebabkan bayangan yang diciptakan sumber cahaya dari radius *non-zero*; (3) Bayangan pada batas pinggir objek yang mulus.

Contoh persamaan *edge* sederhana yaitu pada sebuah *image* satu dimensi f yang memiliki tepat satu *edge* yang ditempatkan pada $x = 0$ dapat dimodelkan sebagai:

$$f(x) = \frac{I_r - I_l}{2}\left(erf\left(\frac{x}{\sqrt{2\sigma}} \right) + 1 \right) + I_l$$

di mana
erf = error function
I_l = intensitas pada sisi kiri *edge* ($\lim_{x \to -\infty} f(x)$)
I_r = intensitas pada sisi kanan *edge* ($\lim_{x \to \infty} f(x)$)
σ = skala *blur* dari *edge*.

Ada beberapa metode pendekatan untuk *edge detection*, seperti *canny edge detection, differential edge detection, phase congruency based edge detection*, dan sebagainya. Yang akan dibahas di sini hanya *canny edge detection* karena metode itulah yang akan digunakan.

Transformasi *Hough*

Dalam parameter kartesius, titik-titik kolinier dalam sebuah gambar dengan koordinat (x, y) dihubungkan dengan kemiringan *m* dan konstanta *c* sebagai berikut:

$$y = mx + c \qquad (1)$$

Persamaan ini juga dapat dituliskan dalam bentuk:

$$Ay + Bx + 1 = 0 \quad (2)$$

Di mana $A = -1/c$ dan $B = m/c$.

Dengan demikian, sebuah garis didefinisikan dengan memberikan sepasang nilai (*A*, *B*). Kita juga dapat mengobservasi sebuah simetri dalam definisi pada Persamaan (2). Persamaan ini simetris karena sepasang koordinat (*x*, *y*) juga mendefinisikan sebuah garis dalam ruang yang sama dengan parameter (*A*, *B*). Persamaan (1) dan (2) dapat dilihat sebagai persamaan sebuah garis untuk koordinat tetap (*x*, *y*) atau sebagai persamaan sebuah garis untuk parameter tetap (*A*, *B*). Dengan demikian, keduanya dapat digunakan untuk mendefinisikan titik-titik dan garis-garis secara simultan (Aguado, 2000). TH mengumpulkan keterangan dari titik (*A*, *B*) dengan mempertimbangkan bahwa semua titik-titik (x, y) mendefinisikan semua garis yang sama dalam ruang (*A*, *B*) tersebut. Di mana, jika himpunan titik-titik kolinear $\{(x_i, y_i)\}$ mendefinisikan garis (*A*, *B*), maka

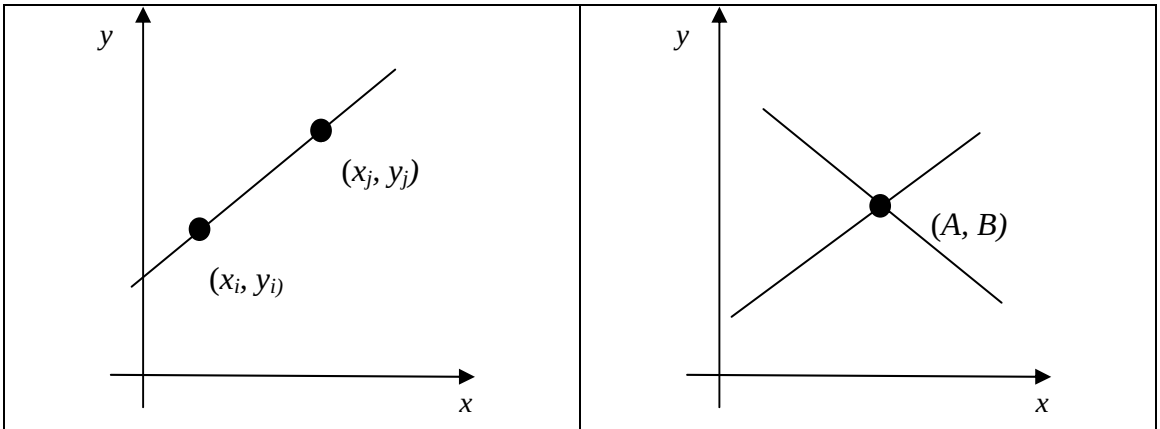
$$Ay_i + Bx_i + 1 = 0 \quad (3)$$

Persamaan ini dapat dilihat sebagai sebuah sistem dan dapat dituliskan secara sederhana dalam parameter kartesius sebagai

$$c = -x_i m + y_i \quad (4)$$

Maka untuk menemukan garisnya kita harus menemukan nilai-nilai dari parameter (*m*, *c*) (atau (*A*, *B*) dalam bentuk yang homogen) yang memenuhi persamaan (3) atau (4).

Hubungan antara sebuah titik (x_i, y_i) dalam sebuah gambar dan garis dalam persamaan (1) sampai (4) digambarkan dalam grafik di bawah ini (Gambar 2).



Gambar 2. Ilustrasi Transformasi Hough untuk garis.

HASIL DAN PEMBAHASAN

Poros dalam ruang ganda mewakili parameter-parameter dari garis. Dalam parameter kartesius *m* memiliki nilai yang tak terbatas, karena garis dapat terbentuk vertikal maupun horizontal. Karena pemilihan dilakukan dalam himpunan *diskrit*, maka hal ini dapat mengakibatkan *bias errors*. Batas pemilihan dapat ditentukan dalam ruang akumulator yang melingkupi semua kemungkinan nilai. Hal

ini berhubungan dengan teknik anti penggandaan dan dapat mengembangkan strategi pengumpulan informasi (Brown, 1983), (Kiryati, 1991).

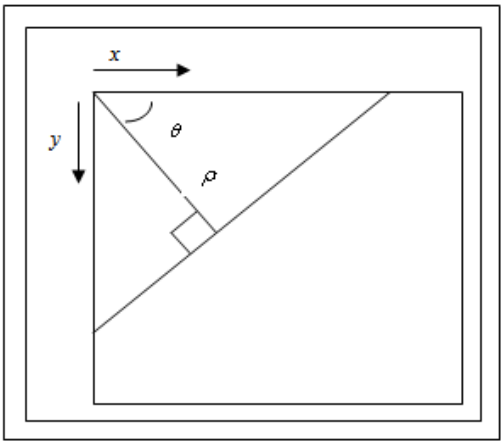
Perlu diperhatikan bahwa persamaan (4) tidak cocok untuk implementasi karena parameter-parameternya memiliki nilai tak terbatas. Untuk mengatasi ketidakterbatasan nilai *c*, kita menggunakan 2 himpunan (*array*). Saat nilai gradien *m* berkisar di antara -45^0 dan 45^0, maka nilai *c* tidak terlalu luas. Untuk nilai *m* di luar kisaran tersebut nilai *c* menjadi sangat luas. Oleh karena itu, dipertimbangkan sebuah akumulator untuk setiap kasus. Dalam kasus yang kedua, digunakan sebuah himpunan yang menyimpan nilai konstanta tersebut dengan sumbu *x*.

Kita dapat melihat bahwa TH memberikan respon yang tepat. Namun, tidak linear dan *diskrit*-nya nilai parameter menghasilkan akumulator-akumulator yang tidak jernih. Masalah utama dalam implementasi TH dasar untuk garis adalah definisi ruang akumulator yang sesuai.

Satu cara untuk mengatasi masalah penggunaan parameter kartesius untuk TH yaitu dengan mendasarkan fungsi pemetaan pada parameter alternatif. Teknik yang paling terbukti ketepatannya dikenal dengan *foot-of-normal parameterisation*. Teknik ini memformulasikan sebuah garis dengan menentukan titik (x, y) sebagai fungsi dari sebuah sudut normal terhadap garis, melewati daerah dari gambar. Ini memberikan bentuk dari TH untuk garis yang dikenal dengan *polar Hough transfom for lines* (Duda, 1972). Titik di mana garis ini berpotongan dengan garis pada gambar diberikan sebagai

$$\rho = x\cos\theta + y\sin\theta \qquad (5)$$

Di mana θ adalah sudut yang dibentuk oleh garis normal dengan garis pada gambar dan ρ adalah jarak antara daerah dan titik di mana garis-garis tersebut berpotongan (Gambar 3).



Gambar 3. Penentuan polar sebuah garis.

Dengan mengetahui bahwa 2 buah garis dinyatakan tegak lurus jika hasil kali dari kemiringannya adalah -1, dan mempertimbangkan bentuk bangun pada gambar di atas, maka didapatkan

$$c = \frac{\rho}{\sin\theta} \quad m = \frac{-1}{\tan\theta} \qquad (6)$$

Dengan substitusi pada persamaan 2 dan 1 diperoleh bentuk polar pada persamaan (5). Hal ini memberikan fungsi pemetaan yang berbeda dalam bentuk sinusoidal. Keuntungan dari pemetaan alternatif ini yaitu nilai dari parameter-parameter θ dan ρ sekarang memiliki batas rentang yang spesifik. Rentang dari ρ adalah 180^0; nilai-nilai kemungkinan ρ tergantung pada ukuran dari

gambar, karena panjang maksimal sebuah garis adalah $\sqrt{2} \times N$, di mana *N* adalah luas permukaan gambar. Rentang nilai-nilainya sekarang telah ditentukan, sehingga teknik ini sekarang dapat digunakan.

Himpunan akumulatornya merupakan himpunan nilai θ dari 0^0 sampai 180^0, dan untuk ρ dari 0 sampai $\sqrt{N^2 + M^2}$, di mana $N \times M$ adalah ukuran gambar. Kemudian, untuk titik-titik yang bernilai lebih besar dari rentang yang telah ditentukan, sudut yang berhubungan dengan himpunan akumulator akan dievaluasi (sebagai radian dengan rentang 0 sampai π) dan kemudian nilai dari ρ dievaluasi dari persamaan 5 dan sel akumulator yang sesuai ditambahkan menurut parameter yang masih berada dalam rentang yang ditentukan. Pada gambar ditunjukkan hasil yang lebih jernih jika dibandingkan dengan parameter kartesius sebelumnya. Hal ini disebabkan karena berkurangnya efek *kediskritan* pada parameter polar sehingga memungkinkan implementasi polar lebih dapat dipraktekkan.

Metode Gabor Wavelet

Gelombang Gabor dinyatakan sebagai

$$G(f) = e^{\left(\frac{-(\log(f/f_0))^2}{2(\log(\sigma/f_0))^2} \right)} \quad (7)$$

di mana *f* adalah frekuensi antara 0 sampai 0.5, f_0 adalah frekuensi modulasi, dan σ merupakan standar deviasi dari Gaussian.

Suatu pendekatan telah menyamaratakan fungsi Gabor ke dalam bentuk 2 dimensi dengan tujuan optimalisasi.

$$G(x, y) = e^{-\pi\left[(x-x_0)^2/\alpha^2 + (y-y_0)^2/\beta^2\right]} e^{-2\pi[u_0(x-x_0) + v_0(y-y_0)]} \quad (8)$$

di mana x_0 dan y_0 menunjukkan posisi. Secara alami, bentuk permukaan yang ditentukan oleh fungsi Gaussian 2 dimensi berupa elips jika berbagai varian digunakan sepanjang sumbu-sumbu *x* dan *y* (frekuensinya juga dapat dimodulasikan dengan berbeda sepanjang masing-masing sumbu).

Fungsi transformasi gelombang adalah untuk menentukan di mana dan bagaimana masing-masing gelombang dispesifikasikan oleh rentang nilai untuk setiap parameter bebas yang terdapat pada gambar. Aplikasi-aplikasi gelombang Gabor antara lain sistem sekuritas dengan menggunakan pengenalan iris mata dan ekstrasi bagian-bagian wajah untuk pengenalan wajah otomatis.

Quadrature 2D Gabor Wavelets dinyatakan sebagai

$$h_{\{\mathrm{Re,Im}\}} = \mathrm{sgn}_{\{\mathrm{Re,Im}\}} \int_\rho \int_\phi I(\rho, \phi) e^{-i\omega(\theta_0 - \phi)} e^{-(r_0 - \rho)^2/\alpha^2} e^{-(\theta_0 - \phi)^2/\beta^2} \rho \partial\rho \partial\phi \quad (9)$$

di mana

$\mathrm{Re}(h) \geq 0.0 \equiv 1$

$\mathrm{Re}(h) < 0.0 \equiv 0$

$\mathrm{Im}(h) \geq 0.0 \equiv 1$

$\mathrm{Im}(h) < 0.0 \equiv 0$

Pendefinisian Citra

'Citra' menurut kamus bahasa Indonesia diartikan sebagai gambar atau rupa. Sehingga dalam dunia teknologi informasi, 'citra' dapat diartikan sebagai gambar atau rupa hasil representasi dari

suatu objek nyata dalam bentuk dua atau tiga dimensi yang dapat dimengerti oleh komputer. Definisi 'citra' lainnya adalah suatu fungsi intensitas warna dua dimensi $f(x, y)$ di mana x dan y mewakili koordinat lokasi suatu titik dan nilai dari suatu fungsi yang merupakan tingkat intensitas warna atau tingkat keabu-abuan dari titik tersebut (Robert J. Schalkoff).

Pengolahan Citra

Pengolahan citra (*image processing*) merupakan bidang yang berhubungan dengan proses transformasi citra (*image*) yang bertujuan untuk mendapatkan kualitas citra yang lebih baik (Michael J. Fairhurst). Langkah-langkah utama dalam pengolahan citra (Rafael C. Gonzales) yang dapat dilihat pada Gambar 1, dijelaskan sebagai berikut.

Pertama, *image acquisition, yaitu* merupakan langkah untuk pengambilan citra digital. Langkah ini membutuhkan sebuah sensor pencitraan di mana mengubah sinyal yang diterima menjadi suatu citra digital. Sensor pencitraan berupa kamera berwarna atau hitam putih yang menghasilkan citra setiap 1/30 detik. Jika *output* dari kamera atau sensor pencitraan lainnya bukan dalam bentuk *digital*, maka sebuah *analog-to-digital converter* akan mengubahnya ke dalam bentuk *digital*.

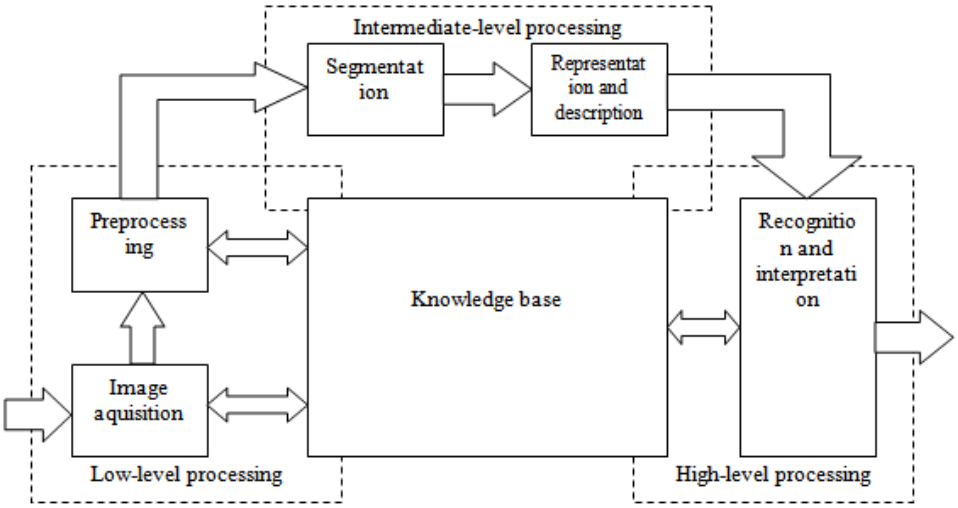
Kedua, *preprocessing*. Fungsi utama dari preprocessing adalah memperbaiki citra dengan cara-cara tertentu di mana meningkatkan kemungkinan sukses bagi proses lainnya. *Preprocessing* berhubungan dengan teknik untuk memperbaiki kontras, menghilangkan *noise*, dan mengisolasi area yang berisi suatu informasi.

Ketiga, *segmentation,* yaitu membagi citra masukan menjadi bagian-bagian penting atau disebut juga objek. Pada umumnya, *segmentation* merupakan tugas yang sulit dari pengolahan citra *digital*. Jika prosedur dari *segmentation* ini kompleks maka akan terjadi proses yang lama untuk mencapai solusi yang sukses dari permasalahan citra. Jika prosedur dari *segmentation* ini lemah atau tidak konsisten maka akan menghasilkan kegagalan. Dalam istilah pengenalan pola, peranan penting dari *segmentation* adalah mengekstrak karakter dan kata individu dari latar belakang.

Keempat, *representation and description*. Keluaran dari tahap *segmentation* biasanya merupakan data yang *pixel*-nya masih kasar, yang merupakan batas dari area ataupun dari semua titik yang terdapat dalam area tersebut. Oleh karena itu, harus dilakukan pengubahan data menjadi bentuk yang sesuai untuk diolah komputer. Keputusan pertama yang harus dibuat adalah memutuskan apakah data dibuat sebagai batas atau area yang penuh. Representasi batas berfokus pada karakteristik eksternal sedangkan representasi area berfokus pada properti internal yang berupa tekstur atau bentuk kerangka. Memilih representasi yang digunakan merupakan bagian dari pencarian solusi untuk mentransformasi data kasar menjadi bentuk yang sesuai untuk tahapan pengolahan selanjutnya. Dan sebuah metode harus dispesifikasikan sehingga dapat mendeskripsikan data.

Kelima, *description*, atau disebut juga sebagai pemilihan *feature*, berhubungan dengan mengekstrak *feature* yang merupakan dasar dalam membedakan satu kelas dengan kelas yang lainnya.

Tahap keenam sekaligus terakhir yaitu *recognition* dan *interpretation*. *Recognition* merupakan proses yang memberikan "label" terhadap suatu objek berdasarkan informasi yang disediakan oleh deskriptornya. Sebagai contoh, untuk mendefinisi sebuah karakter, misalnya 'c', membutuhkan deskriptor yang berhubungan yaitu karakter dengan label 'c'. *Interpretation* melibatkan pemberian arti kepada sekumpulan objek yang dikenali. Sebagai contoh, *string* yang terdiri dari lima karakter angka merupakan lambang dari kode pos.



Gambar 4. Langkah-langkah utama pengolahan citra.

PENUTUP

Kesimpulan

Berdasarkan pembahasan sebelumnya dan dari aplikasi deteksi iris mata untuk absensi karyawan menggunakan Gabor *Wavelet* yang telah dirancang dapat disimpulkan bahwa: (1) Iris mata terbukti unik, karena dapat dikenali dengan akurat, keakuratannya dalam pengenalan mencapai 90%; (2) Kualitas citra yang diambil sangat mempengaruhi output dari proses, sehingga sebisa mungkin ambil citra yang tidak memiliki banyak gangguan dan jangan terlalu terang maupun terlalu gelap. Dan meskipun pada citra ada gangguan atau pencahayaannya kurang, kode iris mata tetap dihasilkan, walau hasilnya jadi kurang baik; (3) Gabor *Wavelet* dapat digunakan untuk mengidentifikasi iris mata dengan baik dan dalam pembuatan aplikasi absensi karyawan.

Saran

Saran-saran yang diusulkan adalah sebagai berikut: (1) Algoritma pemisahan antara pupil dan iris mata dapat dikembangkan lebih lanjut; (2) Dapat ditambahkan pula algoritma lain untuk menghilangkan gangguan pada citra, sehingga tidak perlu mengambil ulang citra yang akan dijadikan ke dalam kode digital.

DAFTAR PUSTAKA

Anonymous. http://cnx.org/content/col10256/latest/. Agustus 2008.

Anonymous. http://library.wolfram.com/examples/edgedetection/. Agustus 2008.

Anonymous. http://planetmath.org/encyclopedia/HoughTransform.html. Desember 2008.

Anonymous. www.argus-solutions.com/how_iris_recognition_works.htm. Agustus 2008.

Anonymous. www.biometrics.gov/Documents/IrisRec.pdf. Agustus 2008.

Anonymous.http://www.pages.drexel.edu/~pyo22/students/designTeams/kite2001WorkFolder/cannyEdgeDetector.pdf. Agustus 2008.

Forsyth, David A. (2003). *Computer Vision : A Modern Approach*. Prentice Hall, New Jersey.

Gonzales, Rafael C.;Woods, Richard E. (2008). *Digital Image Processing, (3rd ed.)*. Pearson Prentice Hall, New Jersey.

J. Daugman. (2001). *Statistical Richness of Visual Phase Information: Update on Recognizing Persons by Iris Patterns*. *International Journal on Computer Vision*. Vol 45(1), pp25-38.

J. Daugman. (2003). *The importance of being random: Statistical principles of iris recognition*. *Pattern Recognition*. Vol 36(2), pp279-291.

Lee, Kwanyong; Lim, Shinyoung; Byeon, Okhwan; Kim, Taiyun. (2001). *Efficient Iris Recognition through Improvement of Feature Vector and Classifier*. *ETRI Journal*. Vol 23, pp61-70.

Mehtora, Hunny; Majhi, Banshidar; Gupta, Phalguni. (2008). *Multi-algorithmic Iris Authentication System*. *Proceeding of World Academy of Science, Engineering and Technology*. Vol 34, pp148-152.

Nixon, Mark S.; Aguado; Albero S. (2002). *Feature Extraction and Image Processing*. Newnes.

Poursaberi, Ahmad; Araabi, Babak N. (2005). *A Novel Iris Recognition System Using Morphological Edge Detector and Wavelet Phase Features*. *ICGST-GVIP Journal*. Vol 5, pp 9-15.

LAMPIRAN

Listing Program Menu Utama

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace ProjectIris2
{
   public partial class frmMain : Form
   {
      public frmMain()
      {
         InitializeComponent();
      }

      private void btnInit_Click(object sender, EventArgs e)
      {
```

```
            frmVeri veri = new frmVeri();
            veri.Show();
            this.Hide();
        }

        private void btnAtt_Click(object sender, EventArgs e)
        {
            frmAtt att = new frmAtt();
            att.Show();
            this.Hide();
        }
    }
}
```

Listing Program Layar Verifikasi

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace ProjectIris2
{
    public partial class frmVeri : Form
    {
        public frmVeri()
        {
            InitializeComponent();
        }

        private void btnCancel_Click(object sender, EventArgs e)
        {
            frmMain main = new frmMain();
            main.Show();
            this.Hide();
        }

        private void btnOK_Click(object sender, EventArgs e)
        {
            frmInit init = new frmInit();

            if (txtPass.Text == "finldrax")
            {
                init.Show();
                this.Hide();
            }
            else
            {
                MessageBox.Show("Password yang anda masukkan salah", "Peringatan!");
                txtPass.Text = "";
```

```
        }

    }

  }
}
```

Listing Program Menu Inisialisasi Karyawan

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using AForge.Imaging;
using AForge.Imaging.Filters;
using ProjectIris2.Core;

namespace ProjectIris2
{
  public partial class frmInit : Form
  {

    private Bitmap sourceImage;
    private Bitmap houghCirlceImage;
    private FiltersSequence filter = new FiltersSequence(
      new GrayscaleBT709(),
      new Threshold()
    );

    HoughCircleTransformation circleTransform = new HoughCircleTransformation(35);

    public frmInit()
    {
      InitializeComponent();
    }

    Int32[,] valueI = new Int32[240, 20];
    Int32[,] valueN = new Int32[240, 20];

    private void frmInit_Load(object sender, EventArgs e)
    {

    }

    Double[,] eyevalue = new Double[320, 280];

    private void btnPrc_Click(object sender, EventArgs e)
    {
      Bitmap img = new Bitmap(label2.Text);
      ///resize image
```

```
            Size size = new Size(320, 280);
            img = (Bitmap)ImgPro.resizeImage(img, size);
            ///grayscale image
            img = (Bitmap)ImgPro.MakeGrayscale(img);
            ///convert to matrix
            eyevalue = ImgPro.GetEyeValue(img);
            ///generate canny edge
            Bitmap img2 = new Bitmap(img);
            img2 = (Bitmap)ImgPro.EdgeDetect(img2);
            ///Hough transformation to find pupil and iris
            AForge.Imaging.Image.FormatImage( ref img );
            sourceImage = filter.Apply(img);
            circleTransform.ProcessImage(sourceImage);
            houghCirlceImage = circleTransform.ToBitmap();
            HoughCircle[] circles = circleTransform.GetCirclesByRelativeIntensity(0.5);
            foreach (HoughCircle circle in circles)
            {
               string s = string.Format("X = {0}, Y = {1}, I = {2} ({3})", circle.X, circle.Y,
circle.Intensity, circle.RelativeIntensity);
               System.Diagnostics.Debug.WriteLine(s);
            }
            System.Diagnostics.Debug.WriteLine("Found circles: " + circleTransform.CirclesCount);
            System.Diagnostics.Debug.WriteLine("Max intensity: " + circleTransform.MaxIntensity);
            ///normalize -- dari bentuk donat jd persegi panjang
            ImgPro.normalize(img, 20, 240);
            ///extraction -- using gabor + create iris matrix*/
            valueI = ImgPro.GetValue();
            valueN = ImgPro.GetValue();
            MessageBox.Show("Kode digital berhasil didapatkan");
         }

         private void btnBrowse_Click(object sender, EventArgs e)
         {
            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
               label2.Text = openFileDialog1.FileName;
               Bitmap img = new Bitmap(label2.Text);
               pictEye.Image = img;
            }
         }

         private void btnMain_Click(object sender, EventArgs e)
         {
            frmMain main = new frmMain();
            main.Show();
            this.Hide();
         }

         private void btnIns_Click(object sender, EventArgs e)
         {
            if (txtID.Text == "" || txtNama.Text == ""|| label2.Text == "" )
            {
               MessageBox.Show("Data kurang lengkap");
```

```
            }
            else
            {
                DBConnect DBInit = new DBConnect();
                try
                {
                    DBInit.InsInitialize(txtID.Text.ToString(), txtNama.Text.ToString(),
label2.Text.ToString(), valueI, valueN);
                    MessageBox.Show("Data berhasil ditambahkan ke dalam database dan akan direset dari
layar");
                    pictEye.Image = null;
                    txtID.Text = null;
                    valueI = valueN = null;
                    txtNama.Text = null;
                    label2.Text = null;
                    eyevalue = null;
                    MessageBox.Show("Data berhasil direset dari layar");
                }
                catch
                {
                    MessageBox.Show("Salah satu dari ID atau iris sudah terpakai, tidak boleh ada dua ID
atau iris yang sama");
                    txtID.Text = null;
                    pictEye.Image = null;
                    valueI = valueN = null;
                    label2.Text = null;
                    eyevalue = null;
                }
            }
        }

    }
}
```

Lampiran 4: Listing Program Menu Mulai Absensi Karyawan

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using AForge.Imaging;
using AForge.Imaging.Filters;
using ProjectIris2.Core;

namespace ProjectIris2
{
    public partial class frmAtt : Form
    {
        private Bitmap sourceImage;
        private Bitmap houghCirlceImage;
```

```
private FiltersSequence filter = new FiltersSequence(
   new GrayscaleBT709(),
   new Threshold()
);

HoughCircleTransformation circleTransform = new HoughCircleTransformation(35);

public frmAtt()
{
   InitializeComponent();
}

Int32[,] valueI = new Int32[240, 20];
Int32[,] valueN = new Int32[240, 20];

private void btnBrowse_Click(object sender, EventArgs e)
{
   if (openFileDialog1.ShowDialog() == DialogResult.OK)
   {
      label2.Text = openFileDialog1.FileName;
      Bitmap img = new Bitmap(label2.Text);
      pictEye.Image = img;
   }
}

Double[,] eyevalue = new Double[320, 280];

private void btnMain_Click(object sender, EventArgs e)
{
   frmMain main = new frmMain();
   main.Show();
   this.Hide();
}

private void btnPrc_Click(object sender, EventArgs e)
{
   Bitmap img = new Bitmap(label2.Text);

   ///resize image
   Size size = new Size(320, 280);
   img = (Bitmap)ImgPro.resizeImage(img, size);
   ///grayscale image
   img = (Bitmap)ImgPro.MakeGrayscale(img);
   ///convert to matrix
   eyevalue = ImgPro.GetEyeValue(img);
   ///generate canny edge
   Bitmap img2 = new Bitmap(img);
   img2 = (Bitmap)ImgPro.EdgeDetect(img2);
   ///Hough transformation to find pupil and iris
   AForge.Imaging.Image.FormatImage(ref img);
   sourceImage = filter.Apply(img);
   circleTransform.ProcessImage(sourceImage);
   houghCirlceImage = circleTransform.ToBitmap();
```

```
        HoughCircle[] circles = circleTransform.GetCirclesByRelativeIntensity(0.5);
        foreach (HoughCircle circle in circles)
        {
           string s = string.Format("X = {0}, Y = {1}, I = {2} ({3})", circle.X, circle.Y,
circle.Intensity, circle.RelativeIntensity);
           System.Diagnostics.Debug.WriteLine(s);
        }
        System.Diagnostics.Debug.WriteLine("Found circles: " + circleTransform.CirclesCount);
        System.Diagnostics.Debug.WriteLine("Max intensity: " + circleTransform.MaxIntensity);
        ///normalize -- dari bentuk donat jd persegi panjang
        ImgPro.normalize(img, 20, 240);
        ///extraction -- using gabor + create iris matrix*/
        valueI = ImgPro.GetValue();
        valueN = ImgPro.GetValue();

        DBConnect DBcon = new DBConnect();
        String id = "";
        try
        {
           id = DBcon.SelectIDFromDB(label2.Text);
        }
        catch
        {
           MessageBox.Show("ID tidak dapat ditemukan");
        }

        if (id != "") {
        String name = DBcon.SelectNameFromDB(id);
        String date = DateTime.Now.Date.ToShortDateString();
        String jammasuk = DBcon.SelectJamMasukFromDB(id, date);
        String jamkeluar = DBcon.SelectJamKeluarFromDB(id, date);

        txtID.Text = id;
        txtNama.Text = name;
        txtMasuk.Text = jammasuk;
        txtKeluar.Text = jamkeluar;

        MessageBox.Show("Image berhasil diproses");
        pictEye.Image = null;
        label2.Text = null;
        }
      }

      private void btnIn_Click(object sender, EventArgs e)
      {
        if (txtID.Text == "")
        {
           MessageBox.Show("Data belum lengkap");
        }
        else if (txtMasuk.Text == "")
        {
           txtMasuk.Text = DateTime.Now.ToLongTimeString();
           String date = DateTime.Now.Date.ToShortDateString();
```

```
            DBConnect DBStart = new DBConnect();
            DBStart.InsEnterTime(txtID.Text.ToString(), date.ToString(), txtMasuk.Text.ToString());

            MessageBox.Show("Jam Masuk karyawan telah dimasukkan ke dalam database");
        }
        else if (txtKeluar.Text == "")
        {
            MessageBox.Show("Karyawan ini telah masuk pada hari ini");
        }
        else
        {
            MessageBox.Show("Karyawan ini telah selesai bekerja pada hari ini");
        }
    }

    private void btnOut_Click(object sender, EventArgs e)
    {
        if (txtID.Text == "")
        {
            MessageBox.Show("Data belum lengkap");
        }
        else if (txtKeluar.Text == "")  {
            txtKeluar.Text = DateTime.Now.ToLongTimeString();
            String date = DateTime.Now.ToShortDateString();

            DBConnect DBStart = new DBConnect();
            if (txtMasuk.Text == ""){
                DBStart.InsExitTime(txtID.Text.ToString(), date.ToString(), txtKeluar.Text.ToString());
            }
            else {
                DBStart.InsExitTime2(txtID.Text.ToString(), date.ToString(), txtKeluar.Text.ToString());
            }

            MessageBox.Show("Jam Keluar karyawan telah dimasukkan ke dalam database");
        }
        else{
            MessageBox.Show("Karyawan ini telah selesai bekerja pada hari ini");        }
    }

    private void frmAtt_Load(object sender, EventArgs e)
    {
        lblTime.Text = DateTime.Now.ToString();
    }

    private void timer1_Tick(object sender, EventArgs e)
    {
        lblTime.Text = DateTime.Now.ToString();
    }
  }
}
```