

Dasar Pengolahan Citra (2)

3.1. Tujuan:

1. Mahasiswa dapat membuat program untuk merubah citra warna RGB menjadi Gray-Scale
2. Mahasiswa dapat membuat program thresholding atau mengatur jumlah derajat keabuan yang ada pada citra

3.2. Dasar Teori:

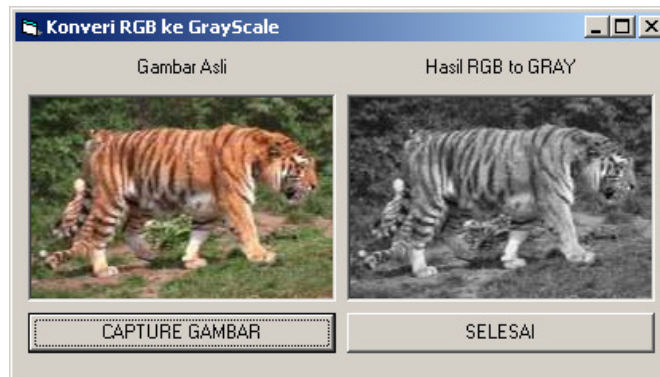
3.2.1. Mengubah Citra Berwarna Menjadi Gray-Scale

Proses awal yang banyak dilakukan dalam *image processing* adalah mengubah citra berwarna menjadi citra gray-scale, hal ini digunakan untuk menyederhanakan model citra. Seperti telah dijelaskan di depan, citra berwarna terdiri dari 3 layer matrik yaitu R-layer, G-layer dan B-layer. Sehingga untuk melakukan proses-proses selanjutnya tetap diperhatikan tiga layer di atas. Bila setiap proses perhitungan dilakukan menggunakan tiga layer, berarti dilakukan tiga perhitungan yang sama. Sehingga konsep itu diubah dengan mengubah 3 layer di atas menjadi 1 layer matrik gray-scale dan hasilnya adalah citra gray-scale. Dalam citra ini tidak ada lagi warna, yang ada adalah derajat keabuan.

Untuk mengubah citra berwarna yang mempunyai nilai matrik masing-masing r , g dan b menjadi citra gray scale dengan nilai s , maka konversi dapat dilakukan dengan mengambil rata-rata dari nilai r , g dan b sehingga dapat dituliskan menjadi:

$$s = \frac{r + g + b}{3}$$

Untuk mencoba proses konversi citra berwarna menjadi citra gray-scale ini dapat dibuat program seperti gambar 3.1



Gambar 3.1. Contoh form untuk menangkap citra.

3.2.2. Thresholding

Thresholding digunakan untuk mengatur jumlah derajat keabuan yang ada pada citra. Dengan menggunakan thresholding maka derajat keabuan bisa diubah sesuai keinginan, misalkan diinginkan menggunakan derajat keabuan 16, maka tinggal membagi nilai derajat keabuan dengan 16. Proses thresholding ini pada dasarnya adalah proses pengubahan kuantisasi pada citra, sehingga untuk melakukan thresholding dengan derajat keabuan dapat digunakan rumus:

$$x = b.\text{int}\left(\frac{w}{b}\right)$$

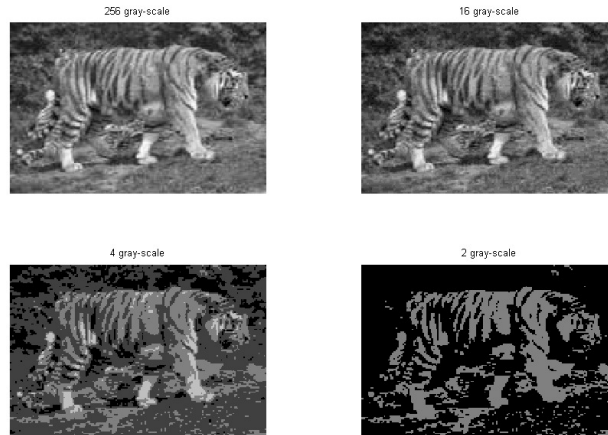
dimana :

w adalah nilai derajat keabuan sebelum thresholding

x adalah nilai derajat keabuan setelah thresholding

$$b = \text{int}\left(\frac{256}{a}\right)$$

Berikut ini contoh thresholding mulai di 256, 16, 4 dan 2.



Gambar 3.2. Contoh thresholding

Untuk mencoba melakukan proses thresholding, perlu dibuat program untuk dapat mengubah-ubah nilai thresholding sesuai keinginan. Sehingga perlu ditampilkan dua citra, yaitu citra asli (gray-scale) dan hasil thresholdingnya dengan nilai thresholding yang ditentukan melalui input seperti terlihat pada gambar 3.2.

3.3. Tugas Pendahuluan:

1. Tuliskan tujuan praktikum
2. Jelaskan cara merubah citra berwarna menjadi Gray-Scale
3. Jelaskan cara mengatur jumlah derajat keabuan pada citra dengan Thresholding

3.4. Percobaan:

3.4.1. Mengubah Citra Berwarna Menjadi Gray-Scale

1. Cara mengubah citra warna menjadi gray-scale
 - Buat aplikasi AppWizard seperti pada praktikum 1 dan beri nama project dengan GrayScale
 - Buat Menu seperti pada praktikum 2 dengan tambahan Test sedangkan submenunya OpenFileDialog dan GrayScale
 - Untuk mengedit isi program tekan tombol Edit Code atau buka file GrayScaleView.cpp

- Tambahkan program untuk mengubah citra warna menjadi gray-scale seperti dibawah ini

```

////////////////////////////////////
// CGrayScaleView message handlers

void CGrayScaleView::OnTestLoadgambar()
{
    // TODO: Add your command handler code here
    static char BASED_CODE szFilter[]="Bitmap Files (*.bmp)|*.bmp|";
    CFileDialog m_IdFile(TRUE, "*.bmp", name,
    OFN_HIDEREADONLY|OFN_OVERWRITEPROMPT, szFilter);
    if(m_IdFile.DoModal()=IDOK)
    {
        name=m_IdFile.GetPathName();
        LoadGambar();
    }
}

// Menampilkan gambar hasil dari open file
void CGrayScaleView::LoadGambar(void)
{
    CDC* pDC = GetDC();
    CDC dcMem;
    HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(), name,
    IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
    if(hBitmap)
    {
        if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
        m_bmpBitmap.Attach(hBitmap);
    }
    dcMem.CreateCompatibleDC(pDC);
    dcMem.SelectObject(&m_bmpBitmap);
    pDC->BitBlt(0,0,150,100,&dcMem,0,0,SRCCOPY);
}

// merubah data pixel ke RGB
void WarnaToRGB(long int warna,int *Red, int *Green, int *Blue)
{
    *Red = warna & 0x000000FF;
    *Green = (warna & 0x0000FF00) >> 8;
    *Blue = (warna & 0x00FF0000) >> 16;
}

//merubah RGB ke data pixel
long int RGBToWarna(int Red, int Green, int Blue)
{
    return(Red+(Green<<8)+(Blue<<16));
}

void CGrayScaleView::OnTestGrayscale()
{
    // TODO: Add your command handler code here
    long int warna;
}

```

```

int j,k,red,green,blue,gray;
CDC* pDC = GetDC();
CDC dcMem;
dcMem.CreateCompatibleDC(pDC);
dcMem.SelectObject(&m_bmpBitmap);
for(j=0;j<100;j++)
    for(k=0;k<150;k++)
    {
        warna=dcMem.GetPixel(k,j);
        // merubah data pixel ke RGB
        WarnaToRGB(warna,&red,&green,&blue);

        // mengubah warna menjadi Gray-Scale
        gray=(red+green+blue)/3;

        //merubah RGB ke data pixel
        warna=RGBToWarna(gray,gray,gray);
        dcMem.SetPixel(k,j,warna);
    }
pDC->BitBlt(0,0,150,100,&dcMem,0,0,SRCCOPY);
}

```

2. Menambah header file

- Buka file GrayScaleView.h
- Tambahkan program seperti dibawah ini

```

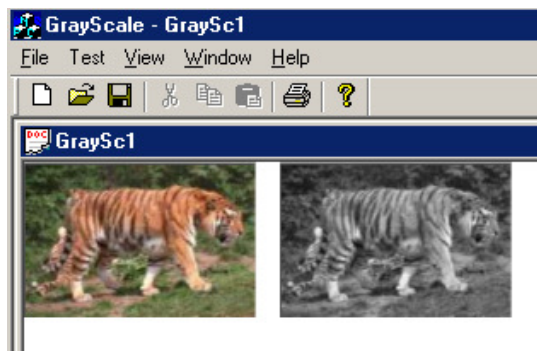
// Attributes
public:
    CGrayScaleDoc* GetDocument();
    CString name;
    CBitmap m_bmpBitmap;

// Operations
public:
    void LoadGambar(void);

```

3. Cara menjalankan program

- Pilih menu : Build->Execute (!)
- Pilih menu : Test->OpenFile -> pilih salah satu gambar misalnya gambar.bmp
- Pilih menu: Test->GrayScale hasilnya seperti gambar 3.3



Gambar 3.3 Mengubah Citra Berwarna Menjadi Gray-Scale

3.4.2. Thresholding

1. Cara mengubah gambar menjadi Thresholding 2 warna

- Buat aplikasi AppWizard seperti pada praktikum 1 dan beri nama project dengan ThresHold
- Buat Menu seperti pada praktikum 2 dengan tambahan Test sedangkan submenunya OpenFile dan ThresHold
- Untuk mengedit isi program tekan tombol Edit Code atau buka file ThresHoldView.cpp
- Tambahkan program untuk Thresholding seperti dibawah ini

```
////////////////////////////////////  
// CThresHoldView message handlers  
  
void CThresHoldView::OnTestOpenfile()  
{  
    // TODO: Add your command handler code here  
    static char BASED_CODE szFilter[]="Bitmap Files (*.bmp)|*.bmp|";  
    CFileDialog m_IdFile(TRUE, "*.bmp", name,  
        OFN_HIDEREADONLY|OFN_OVERWRITEPROMPT, szFilter);  
    if(m_IdFile.DoModal()==IDOK)  
    {  
        name=m_IdFile.GetPathName();  
        LoadGambar();  
    }  
}  
  
// Menampilkan gambar hasil dari open file  
void CThresHoldView::LoadGambar(void)  
{  
    CDC* pDC = GetDC();  
    CDC dcMem;  
    HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(), name,  
        IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);  
    if(hBitmap)  
    {  
        if(m_bmpBitmap.DeleteObject())  
            m_bmpBitmap.Detach();  
        m_bmpBitmap.Attach(hBitmap);  
    }  
    dcMem.CreateCompatibleDC(pDC);  
    dcMem.SelectObject(&m_bmpBitmap);  
    pDC->BitBlt(0,0,200,200,&dcMem,0,0,SRCCOPY);  
}  
  
// merubah data pixel ke RGB  
void WarnaToRGB(long int warna,int *Red, int *Green, int *Blue)  
{  
    *Red = warna & 0x000000FF;
```

```

        *Green = (warna & 0x0000FF00) >> 8;
        *Blue = (warna & 0x00FF0000) >> 16;
    }

    //merubah RGB ke data pixel
    long int RGBToWarna(int Red, int Green, int Blue)
    {
        return(Red+(Green<<8)+(Blue<<16));
    }

void CThresHoldView::OnTestThreshold()
{
    // TODO: Add your command handler code here
    long int warna;
    int j,k,red,green,blue,gray;
    CDC* pDC = GetDC();
    CDC dcMem;
    dcMem.CreateCompatibleDC(pDC);
    dcMem.SelectObject(&m_bmpBitmap);
    for(j=0;j<100;j++)
        for(k=0;k<150;k++)
        {
            warna=dcMem.GetPixel(k,j);
            // merubah data pixel ke RGB
            WarnaToRGB(warna,&red,&green,&blue);

            // mengubah warna menjadi Gray-Scale
            gray=(red+green+blue)/3;
            if(gray<128)
                warna=RGBToWarna(0,0,0);
            else
                warna=RGBToWarna(255,255,255);
            dcMem.SetPixel(k,j,warna);
        }
    pDC->BitBlt(150,0,300,100,&dcMem,0,0,SRCCOPY);
}

```

2. Menambah header file

- Buka file ThresHoldView.h
- Tambahkan program seperti dibawah ini

```

// Attributes
public:
    CThresHoldDoc* GetDocument();
    CString name;
    CBitmap m_bmpBitmap;

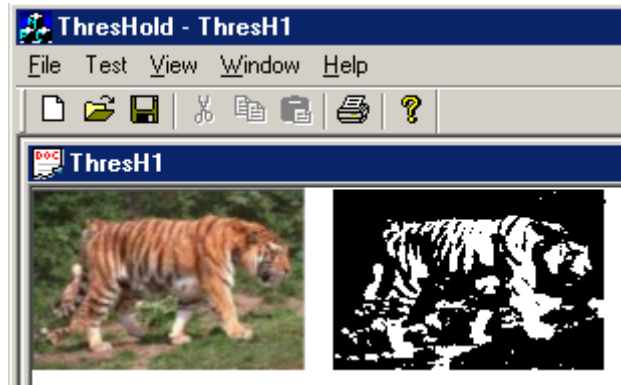
// Operations
public:
    void LoadGambar(void);

```

3. Cara menjalankan program

- Pilih menu : Build->Execute (!)

- Pilih menu : Test->OpenFile -> pilih salah satu gambar misalnya gambar.bmp
- Pilih menu : Test->ThresHolding -> hasilnya seperti gambar 3.4



Gambar 3.4 ThresHolding dengan nilai 128

3.5. Latihan:

1. Sebutkan proses utama pada proses konversi citra berwarna menjadi citra gray-scale? Apa perbedaan antara pemakaian rumus rata-rata $s = (r + g + b) / 3$ dan rumus RGB optimal $x = 0.42r + 0.32g + 0.28b$.
2. Ubahlah program konversi citra berwarna menjadi citra gray scale di atas dengan mengubah nilai gray scale dengan

$$x = 0.5r + 0.2g + 0.3b$$

$$x = 0.2r + 0.5g + 0.3b$$

$$x = 0.2r + 0.2g + 0.5b$$

$$x = 0.5r + 0.5g + 0b$$

$$x = 0.5r + 0g + 0.5b$$

Perhatikan bagaimana perbedaan hasil konversi dengan tiga macam rumus di atas.

3. Buatlah program menggunakan dialog box untuk mengubah-ubah nilai thresholding melalui slider, perhatikan hasilnya?
4. Jelaskan apa pengertian dari thresholding, dan bagaimana prosesnya?

3.6. Laporan Resmi:

Buatlah laporan resmi dari latihan-latihan diatas dengan cara membuat analisa dan kesimpulan.