ISSN 0853-8697

IMPLEMENTASI METODE *LINEAR ADDITION* UNTUK PERANCANGAN EFEK *MASK LIGHTNING*

Kristoko Dwi Hartomo
Program Studi Teknik Informatika, Fakultas Teknologi Informasi,
Universitas Kristen Satya Wacana
Jalan Diponegoro No. 52-60 Salatiga Jawa Tengah
E-mail : goibmohib@yahoo.com

ABSTRACT

Digital picture development is picture processing using digital computer in order to get different picture or another picture which suits with the needs. One example of picture development is mask lighting effect with linear addition method. The main principle of linear addition method is by adding intensity to every pixel colour by adding every pixel with a variable. This paper focuses on the implementation of linear addition method to give lighting effect on picture.

Keywords : mask lighting, linear addition, pixel.

1. PENDAHULUAN

Dalam kehidupan sehari-hari, pengolahan foto memegang peranan yang cukup penting walaupun tidak disadari secara langsung. Pengolahan foto dimulai sejak ditemukannya fotografi. Hasil dari fotografi tersebut diproses secara manual untuk menghasilkan suatu hasil atau efek yang memperindah foto tersebut. Pengolahan foto saat ini telah banyak digantikan oleh komputer sebagai alat pengolah foto secara digital yang handal. Pengolah foto secara digital mampu menghasilkan suatu foto yang lebih indah dari aslinya. Pengolah foto tersebut memiliki efek-efek yang bermanfaat untuk memanipulasi foto.

Salah satu pemakaian pengolahan foto yang berkembang pesat akhir-akhir ini adalah dalam bidang fotografi yaitu pemberian efek pencahayaan pada foto. Variasi dari efek cahaya ini dapat diciptakan dengan merancang *mask lightning* (cahaya yang berbentuk) yang unik dan beraneka ragam untuk memperindah foto.

Melihat uraian di atas maka akan dirancang dan direalisasikan beberapa *mask lightning*, untuk memberikan efek pencahayaan pada foto, *user* (pemakai) tinggal memilih salah satu *mask lightning* yang telah disediakan. Untuk membuat efek *mask lightning* ada beberapa macam metode, yang akan dibahas dalam makalah ini adalah efek pencahayaan dengan metode *linear addition*.

2. LANDASAN TEORI

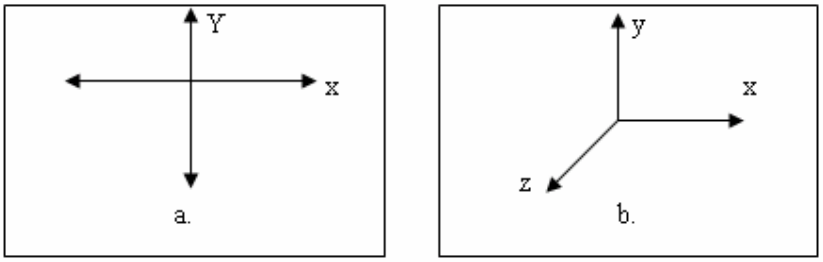
Grafika komputer pada dasarnya adalah suatu bidang ilmu komputer yang mempelajari tentang cara-cara untuk meningkatkan dan memudahkan komunikasi antara manusia dengan mesin (komputer) dengan jalan membangkitkan, menyimpan, dan memanipulasi gambar model suatu obyek menggunakan

komputer. Grafika komputer memungkinkan *user* (pemakai komputer) untuk berkomunikasi lewat gambar-gambar, bagan-bagan, dan diagram-diagram.

2.1 Sistem Koordinat

Suatu grafik, dari bentuk yang paling sederhana, sampai yang paling yang paling rumit sekalipun selalu bisa dipecahkan menjadi elemennya yang paling kecil, yaitu titik. Dalam layar penampil elemen terkecil dari grafik adalah piksel. Letak suatu titik atau piksel, bisa dibedakan dengan titik lain berdasarkan lokasi titik-titik tersebut. Dalam dunia nyata, dikenal adanya sistem koordinat *cartesian* yang dipakai untuk membedakan lokasi atau posisi sembarang titik atau obyek lain. Dalam sistem koordinat tersebut dikenal sistem koordinat *cartesian* dua dimensi dan tiga dimensi.

Dalam sistem koordinat *cartesian* dua dimensi, lokasi setiap titik ditentukan oleh dua besaran. Nilai dua besaran tersebut apabila digambar akan membentuk suatu sumbu koordinat mendatar dan tegak. Sumbu koordinat mendatar sering disebut dengan *absis* (sumbu x), dan sumbu koordinat tegak dinamakan dengan *ordinat* (sumbu y). Dalam sistem koordinat dimensi tiga, selain sumbu x dan sumbu y terdapat satu sumbu lain, yang arahnya tegak lurus dengan sumbu y dan sekaligus tegak lurus sumbu x, sehingga lebih tepat dikatakan bahwa sumbu ini tegak lurus (menembus) bidang gambar, dan disebut sebagai sumbu z. Secara sederhana, sistem koordinat *Cartesian* dua dimensi dan tiga dimensi bisa digambarkan seperti terlihat dalam Gambar 1.a. dan Gambar 1.b.



Gambar 1. Sistem koordinat *cartesian:* (a) koordinat *cartesian* dua dimensi; (b) koordinat *cartesian tiga* dimensi.

Berbeda dengan sistem koordinat *cartesian* yang mengenal sumbu x dan sumbu y (serta sumbu z untuk sistem koordinat tiga dimensi) negatif, maka dalam sistem koordinat layar, yang hanya mengenal sistem koordinat dua dimensi, hanya dikenal sumbu x dan sumbu y positif. Dalam sistem koordinat layar, koordinat (0,0) menunjukkan titik kiri atas layar, dan koordinat (x1,y1) menunjukkan titik kanan bawah. Nilai x1dan y1 menyesuaikan terhadap mode grafik yang digunakan. Sebagai contoh, untuk mode VGA Hi, nilai x1 adalah 639, dan nilai y1 adalah 479.

Seperti dijelaskan di atas, sistem koordinat layar hanya mengenal sumbu x dan sumbu y positif. Meskipun demikian, dengan cara-cara tertentu koordinat

layar bisa dimanipulasi sehingga bisa menampilkan foto seperti halnya pada koordinat *cartesian* dua dimensi. Dengan memperhatikan hal di atas, mungkin ada pendapat yang mengatakan bahwa layar komputer hanya bisa menampilkan foto dua dimensi. Pendapat ini ada benarnya, tetapi juga ada salahnya. Memang tidak bisa menyajikan suatu obyek tiga dimensi pada suatu layar dua dimensi. Tetapi dengan suatu teknik yang disebut dengan proyeksi dan perspektif, layar dua dimensi dapat menampilkan foto obyek tiga dimensi. Apa saja yang bisa dilihat menggunakan mata dan lensa kamera sesungguhnya adalah proyeksi perspektif, dalam proyeksi perspektif ketebalan atau kedalaman bisa ditunjukkan dengan cara memperkecil ukuran dari obyek-obyek yang terletak lebih jauh.

2.2 Mode Warna Red-Green-Blue

Monitor komputer menggunakan tiga buah warna primer yaitu : merah (*red*), hijau (*green*) dan biru (*blue*) yang disebut dengan istilah mode warna RGB [2]. Pada mode warna RGB setiap titik pada layar berisi angka yang bukan menunjukkan intensitas warna dari titik tersebut, melainkan menunjukkan intensitas yang dipilih pada suatu tabel. Jadi pada setiap titik, dapat dipilih salah satu warna dari tabel.

Adapun masing-masing warna dari tabel memiliki tiga buah kombinasi angka yaitu R, G dan B yang menentukan proporsi warna merah (*red*), hijau (*green*) dan biru (*blue*) dari warna tersebut. R, G dan B masing-masing memiliki range antara 0 hingga 255 sehingga jumlah warna ditabel yang dapat dipilih untuk mengisi warna pada sebuah piksel adalah 256 x 256 x 256 = 16,7 juta warna.

Mode warna RGB yang berlaku untuk *grayscale* nilai R, G dan B adalah sama. Jika nilai R adalah 1 maka nilai G dan B adalah 1 juga, demikian seterusnya untuk lebih jelasnya nilai R, G dan B dapat dilihat pada Tabel 1.

Tabel 1. Nomor warna skala abu-abu

No.	Red	Green	Blue
0	0	0	0
1	1	1	1
2	2	2	2
…	…	…	…
…	…	…	…
254	254	254	254
255	255	255	255

2.3 Pembangkitan Piksel

Piksel dibangkitkan berdasarkan data digital yang tersimpan dalam pengingat digital, nilai 0 berarti piksel dalam keadaan mati, dan nilai 1 menunjukkan bahwa piksel pada suatu lokasi dalam keadaan hidup. Dengan cara penyimpanan ini hanya dikenal 2 warna saja, yakni hitam untuk piksel mati dan putih untuk piksel hidup. Proses pembangkitan piksel berdasarkan pola data digital yang tersimpan dalam pengingat digital bisa dijelaskan secara sederhana sebagai berikut. Pada pengingat digital dan layar tampilan seolah-olah terdapat

suatu piranti yang disebut sebagai *scan line*. *Scan line* ini akan "membaca" pola digital baris demi baris. Setiap kali *scan line* "membaca" satu baris pola digital, maka pengolah tampilan akan menterjemahkannya menjadi pola piksel yang terlihat dalam layar tampilan. Dengan demikian *scan line* pada layar tampilan bergerak sinkron terhadap *scan line* dalam pengingat digital. Di dalam pengingat digital setiap piksel diwakili dengan sebuah bit, tetapi jika diinginkan sejumlah warna atau tingkat intensitas yang berbeda muncul pada layar tampilan, maka satu bit untuk satu piksel tidak mencukupi.

Di dalam rangkaian elektronis yang digunakan untuk mengendalikan elektron pada layar tampilan, terutama jika digunakan untuk layar tampilan warna, terdapat tiga buah pembangkit elektron yang disebut sebagai *red gun*, *green gun*, dan *blue gun*. Ketiga *gun* ini dikendalikan oleh kombinasi bit yang tersimpan di dalam pengingat digital. Dengan mengkombinasikan bit-bit yang mengendalikan masing-masing *gun* di atas, akan diperoleh warna yang berbeda satu dengan yang lain.

Untuk sembarang x bit, bisa disusun 2^x buah kombinasi dari x bit tersebut, sehingga apabila x bit tersebut digunakan untuk mengendalikan sebuah *gun,* akan diperoleh 2^x warna atau intensitas yang berbeda. Selanjutnya, jika masing-masing *gun* dikendalikan oleh x bit, akan diperoleh 2^{3x} warna yang berbeda. Sebagai contoh, jika x adalah 2 maka akan diperoleh 2^6 atau 64 buah warna yang berbeda. Dengan setiap *gun* dikendalikan oleh 2 bit, dikatakan bahwa setiap piksel tersaji dalam 6 bit, karena terdapat 3 buah *gun,* sehingga pengingat digitalnya disebut pengingat digital 6 bit/piksel, atau dapat dikatakan bahwa terdapat 6 bidang bit pengingat di dalam pengingat digital. Jika setiap *gun* dikendalikan oleh 2 bit akan diperoleh 4 buah kombinasi intensitas dari *gun* tersebut. Tabel 2 menunjukkan contoh kombinasi intensitas *gun* yang dikendalikan oleh 2 buah bit.

Tabel 2. Contoh intensitas *gun* yang dikendalikan oleh 2 bit

Nilai bit	Intensitas relatif
00	0 (gelap)
01	1/3 (redup)
10	2/3 (terang)
11	1 (paling terang)

Kombinasi seperti tersaji dalam tabel di atas digunakan untuk mengendalikan tegangan dari DAC (*Digital to Analog Converter*). Keluaran dari DAC yang berupa tegangan digunakan untuk mengendalikan *gun* yang berfungsi untuk memancarkan elektron untuk dipendarkan oleh layar tampilan.

2.4 Format gambar BMP(*Bitmap*)

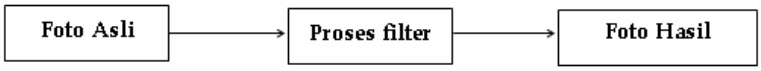
Format gambar BMP merupakan format gambar yang digunakan oleh sistem operasi Windows dan IBM OS/2.Format BMP ini mendukung penggunaan warna mulai dari monokrom hingga true color (16,7 juta warna). Dalam file BMP terdapat struktur seperti terlihat dalam Tabel 3.

Tabel 3. Struktur file BMP

Byte#	*Ukuran*	*Nama*	*Keterangan*
1	2 byte	BmpType	Tipe file BMP
3	4 byte	BmpSize	Besar file BMP dalam *byte* atau *word*
7	2 byte	XHotSpot	X hot spot untuk kursor (pointer)
9	2 byte	YHotSpot	Y hot spot untuk kursor (pointer)
11	4 byte	OffBits	Offset kepada awal data bitmap dalam byte
1	4 byte	HdrSize	Ukuran header dalam byte
5	4 byte	Width	Lebar bitmap dalam piksel
9	4 byte	Height	Tinggi bitmap dalam piksel
13	4 byte	Planes	Jumlah plane (hampir selalu 1)
15	2 byte	BitCount	Jumlah bit per piksel
17	2 byte	Compression	Jenis kompresi (0 = tidak terkompresi)
21	4 byte	ImgSize	Ukuran bitmap dalam *byte*
25	4 byte	HorzRes	Resolusi horisontal
29	4 byte	VertRes	Resolusi vertikal
33	4 byte	ClrUsed	Jumlah warna yang digunakan
37	4 byte	ClrImportant	Jumlah warna yang penting
41	4 byte	Units	Satuan pengukuran yang digunakan
43	2 byte	Reserved	Tidak digunakan
45	2 byte	Recording	Algoritma perekaman
47	2 byte	Rendering	Algoritma *halftoning*
49	2 byte	Size 1	Nilai ukuran 1
53	4 byte	Size 2	Nilai ukuran 2
57	4 byte	ClrEncoding	Pengkodean warna
61	4 byte	Indentifier	Kode untuk digunakan aplikasi

2.5 Pengolahan Gambar

Dalam perkembangannya grafika komputer tidak terlepas dari pengolahan foto secara digital *(Digital Image Processing)*. Pengolahan foto secara digital adalah pemrosesan foto dengan menggunakan komputer digital sehingga menghasilkan foto lain yang lebih sesuai dengan keinginan. Ini dilakukan dengan cara memanipulasi foto-foto yang diambil dengan menggunakan *scanner*, kamera digital dan peralatan input yang lain untuk diubah ke dalam bentuk data digital (digitisasi). Foto yang telah didigitisasi ini kemudian bisa diproses dan ditampilkan di monitor. Pengolahan foto dapat digambarkan dengan diagram yang terdapat dalam Gambar 2.



Gambar 2. Proses pengolahan foto

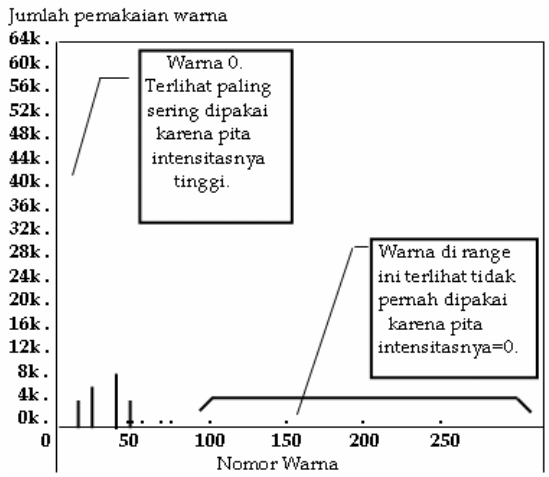
Filter adalah jenis pengolahan gambar yang dilakukan pada foto asli untuk mendapatkan foto hasil. Dengan menggunakan pengolahan gambar ini komputer bisa melakukan hal yang sulit atau tidak mungkin dilakukan pada film.

2.6 Penambahan dan Pengurangan Intensitas Foto

Setiap foto dapat dianalisis dengan melihat *image histogram*-nya. *Image histogram* ini ialah grafik yang menunjukkan *frequency* (jumlah digunakannya) suatu nomor warna. Sumbu horisontal menunjukkan nomor warna yang dipakai (mulai dari 0 hingga 255). Sedangkan sumbu vertikal menunjukkan jumlah titik yang mempergunakan warna yang bersangkutan. Nomor warna 0 adalah warna paling gelap dan nomor warna selanjutnya, semakin mendekati nomor 255 semakin mendekati putih terang. Di bawah ini adalah pengertian dari beberapa istilah yang akan digunakan dalam pembahasan :

- Noise: titik-titik pada foto yang sebenarnya bukan merupakan bagian dari foto melainkan ikut tercampur pada foto karena suatu sebab.
- Pita intensitas: garis vertikal pada *image histogram* yang menunjukkan berapa kali sebuah intensitas digunakan dalam foto.

Sebagai contoh, gambar 3 berikut ini ialah *image histogram* dari sebuah foto. Titik yang mewakili nomor warna sekitar 35, dipakai sejumlah 8k kali. 8k berarti 8000 kali. Sedangkan warna nomor 50, dipakai sekitar 1k = 1000 kali. Dan nomor warna 0 dipakai sekitar 45000 kali. Hal ini disebabkan karena warna 0 (paling gelap) paling banyak dipakai.



Gambar 3. Image histogram sebelum dilakukan proses *linear addition*.

Sekarang perhatikan *image histogram* di atas, pada *image histogram* ini menunjukkan bahwa nomor warna 75 hingga 255 tidak pernah dipakai sama sekali. Dengan kata lain, foto cenderung menggunakan warna 0 hingga 75 di mana warna tersebut masih termasuk warna putih gelap.

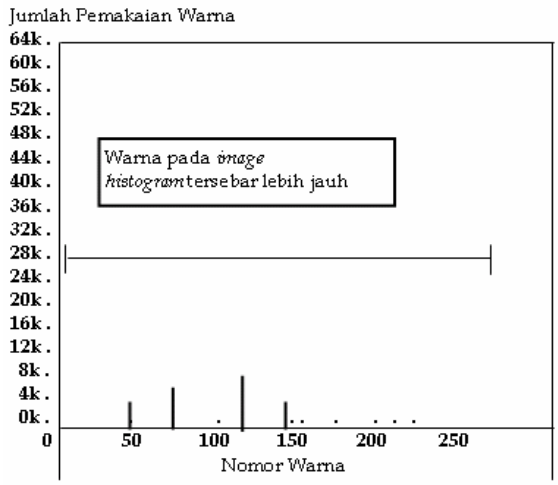
2.7 Metode Linear Addition

Prinsip utama dari teknik ini ialah melakukan penambahan intensitas pada setiap warna titik dengan cara menambahkan setiap titik dengan suatu variabel.

Langkah-langkah yang akan dikerjakan dalam metode *linear addition* adalah sebagai berikut :

1. Melakukan *looping* untuk seluruh titik di layar.
2. Mengambil nomor warna titik yang hendak diubah.
3. Tambahkan dengan variabel penambah.
4. Jika hasil penambahan melebihi warna 255 maka ambil angka 255 sebagai hasilnya.
5. Letakkan titik hasil proses pada layar output.

Perhatikan gambar 4, karena penambahan intensitas dari setiap titik tidak sama, semakin terang suatu titik semakin banyak penambahan intensitas yang didapat maka jarak *image histogram* yang dihasilkan cenderung lebih lebar. Setelah dilakukan proses *linear addition*, *image histogram* tidak bergeser ke kanan melainkan diperlebar ke kanan.



Gambar 4. *Image histogram* setelah dilakukan proses *linear addition.*

Secara grafik dapat disimpulkan perbedaan intensitas dari titik yang ada di layar menjadi lebih besar sehingga mata lebih mudah membedakan antara titik yang satu dengan yang lain, dengan demikian foto akan terlihat jelas.

3. ANALISIS DAN PERANCANGAN

3.1 Analisis

Cahaya yang jatuh pada permukaan suatu obyek (gambar) akan menimbulkan bagian dari obyek tersebut terlihat menjadi terang. Adapun bagian mana dari obyek yang terlihat lebih terang ditentukan dari jarak cahaya, bentuk dari lampu yang menyorotkan cahaya, dan kekuatan cahaya. Untuk membuat efek cahaya seperti di atas, caranya adalah dengan membuat sebuah *mask* (bentuk) yang menentukan bentuk dari cahaya yang akan dijatuhkan pada permukaan dari gambar, isi *mask* ini ialah titik-titik yang intensitas warnanya mewakili kekuatan

cahaya dari titik tersebut. Semakin kuat cahaya pada titik tersebut, semakin banyak penambahan intensitas yang akan dilakukan.

3.2 Perancangan

3.2.1 Perancangan Proses

Proses yang berlangsung untuk makalah ini dapat digambarkan dalam algoritma yang menjelaskan secara berurutan langkah demi langkah program ini. Program diimplementasikan menggunakan bahasa pemrograman Borland Delphi.

Algoritma proses buka file bitmap

Untuk membuka file foto dan menampilkan di layar digunakan algoritma sebagai berikut :

- Buka kotak dialog untuk memilih nama file bitmap
- Pilih satu foto
- Ambil nama file yang telah dipilih
- Tampilkan foto di layar
- Image1.AutoSize:=True (supaya gambar berukuran asli)

Algoritma siapkan selection

Untuk menyiapkan bidang foto (daerah untuk menampilkan *mask lightning*) supaya selalu dalam kondisi bersih atau belum dikenai proses, algoritmanya adalah :

- Buat bentuk *selection* (proses memilih daerah pada gambar yang akan dikenai proses) di atas bidang foto
- Selecting:=False (tidak terjadi proses *selection*)
- Beri kondisi, if AdaSelection then
- Pada foto buat kotak *selection* dengan sudut kiri atas KoordAwal.x, KoordAwal.y dan sudut kanan bawah KoordSaiki.x, KoordSaiki.y

Algoritma proses pilih area selection

Algoritma di bawah untuk melakukan proses pemilihan lokasi pembuatan *selection* (menentukan titik koordinat sudut *selection*) dan ukurannya dengan *mouse*:

- Pada saat tombol *mouse* ditekan, langkahnya :
  - - Selecting := True (proses pemilihan area dimulai)
  - Mencatat titik KoordAwal.x dan KoordAwal.y
  - Mencatat titik KoordSaiki.x dan KoordSaiki.y
- Pada saat menekan tombol sekaligus menggeser *mouse*, langkahnya :
  - Selecting := True
  - KoordMbiyen.x := KoordSaiki.x
  - KoordMbiyen.y := KoordSaiki.y
  - Buat *selection* (berbentuk kotak) dengan sudut kiri atas *KoordAwal*(x,y) dan sudut kanan bawah K*oordMbiyen*(x,y)
  - Mencatat K*oordSaiki*(x,y)
  - Buat *selection* dengan sudut kiri atas *KoordAwal*(x,y) dan sudut kanan bawah K*oordSaiki*(x,y)

- Pada saat melepas tombol *mouse,* langkahnya :
  - Selecting := False (proses pemilihan area diakhiri)

Algoritma linear addition

Algoritma ini untuk menjelaskan langkah untuk melakukan proses *linear addition*:

- Ambil titik koordinat *selection (*daerah yang telah di pilih) untuk dikenai *mask lightning*
- Tukarkan titik koordinat *selection* jika perlu, hal ini dilakukan agar proses *selection* dapat dilakukan dari semua sudut
- Hitung lebar dan tinggi gambar
- *Copy* gambar *mask lightning* dari *maskform* ke *maskimage* di *mainform*
- Image1HDC := Image1.Canvas.Handle
- MaskHDC := MaskImage.Canvas.Handle
- For I := 0 to Width
- For J := 0 to Height
- Ambil titik *mask lightning*
  - TitikMask := GetPixel (MaskHDC,I,J)
  - Rmask := GetRValue (TitikMask)
  - Gmask := GetGValue (TitikMask)
  - Bmask := GetBValue (TitikMask)
- Ambil titik gambar
  - TitikImage := GetPixel (Image1HDC,I+Letak.Left,J+Letak.Top)
  - Rmask := GetRValue (TitikImage)
  - Gmask := GetGValue (TitikImage)
  - Bmask := GetBValue (TitikImage)
- Lakukan proses *linear addition*
  - GbMerah := (GbMerah + RMask) + Variabel Pengali + NuanMerah
  - GbHijau := (GbHijau + GMask) + Variabel Pengali + NuanHijau
  - GbBiru := (GbBiru + BMask) + Variabel Pengali + NuanBiru
- Tampilkan titik hasil proses di layar monitor di posisi (x,y), nilai RGB (Red, Green, Blue)
- Ulangi I,J

Untuk menentukan warna pada Borland Delphi selain menggunakan fasilitas Borland Delphi yaitu perintah GetRValue, GetGValue, GetBValue dan RGB, nilai masing-masing warna dapat dicari secara manual yaitu dengan perhitungan :

Merah = (warna and $FF)
Biru = (warna and $FF00) div ($FF+1)
Hijau = (warna and $FF0000) div ($FFFF+1)

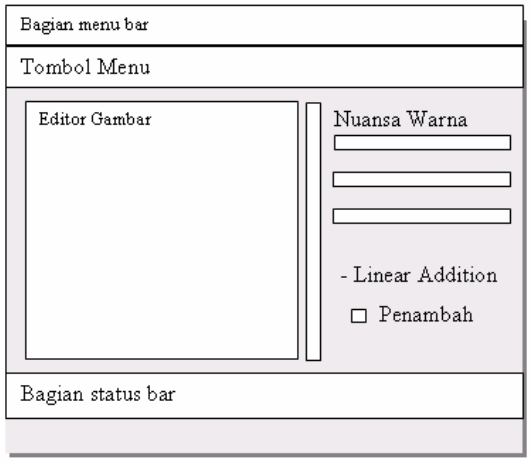
dan perhitungan untuk menggabungkan ketiga warna tersebut:

Warna = (biru + 65536) + (hijau + 256) + merah

Dalam program ini yang digunakan adalah perintah-perintah Borland Delphi di atas.

Perancangan Antar Muka

Tampilan program ini dibuat dalam mode VGAHi, berbentuk *desktop* yang dibagi menjadi bagian judul yang terletak pada bagian atas, bagian menu bar yang terletak di bawah bagian judul untuk menempatkan menu *pulldown*, bagian untuk menempatkan tombol-tombol menu, bagian editor untuk menampilkan gambar, bagian untuk menampilkan pengatur nuansa warna, bagian untuk memasukkan data variabel pengali pada proses *linear addition* dan bagian *statusbar* berisi keterangan program yang terletak pada bagian paling bawah. Perancangan tampilan tersebut dapat dilihat pada Gambar 5.



Gambar 5. Perancangan tampilan

Perancangan Tombol Menu

Sistem tombol menu berisikan menu-menu seperti dirangkum pada Tabel 4.

4. IMPLEMENTASI DAN ANALISIS HASIL KERJA

4.1 Implementasi

Untuk melakukan proses efek pencahayaan *mask lightning*, tekan tombol menu buka, pilih file foto yang akan diberi proses *mask lightning*. Tentukan lokasi tertentu pada gambar yang akan diberi efek. Tekan menu cahaya, pilih *mask lightning* yang dikehendaki. Untuk memberi tambahan efek pewarnaan, efek *mask lightning* bisa diberi tambahan nuansa warna (merah, hijau, biru). Tentukan angka sebagai variabel pengali, untuk melihat hasil akhir pemberian efek tekan tombol proses. Terlihat pada Gambar 6, foto telah berhasil diberi efek *mask lightning*.

4.2 Analisis Hasil Kinerja Progam

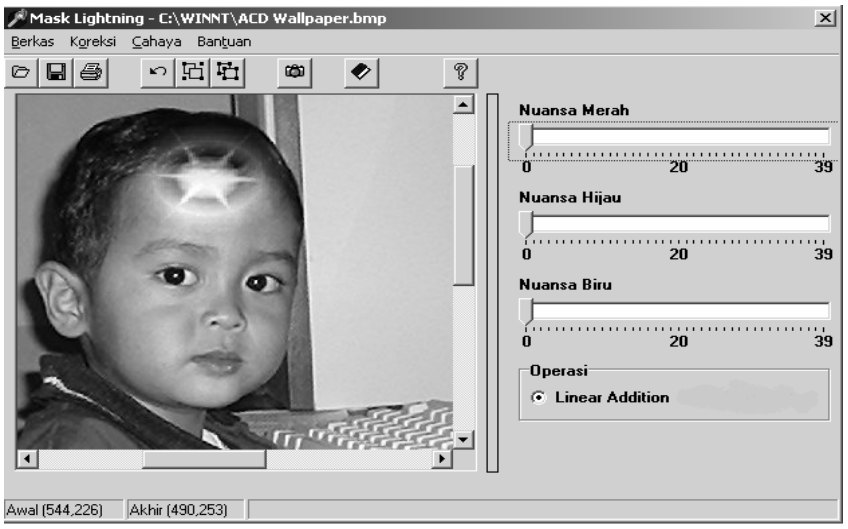
Setelah program dijalankan maka piksel pada data foto akan diproses dengan algoritma *linear addition* dan menjadi sebuah foto baru dengan tambahan efek *mask lightning*.

Dengan metode *linear addition* setiap nilai warna piksel pada foto akan ditambahkan dengan variabel angka yang kita input. Semakin besar variabel yang

dipergunakan untuk mengalikan piksel, semakin jelas terlihat perbedaan antara tiap piksel dan juga foto yang didapat semakin terang. Penambahan intensitas dengan *linear addition* ini juga dapat dipergunakan untuk mengurangi intensitas suatu gambar, yaitu dengan memberi harga N < 1 tetapi N > 0. Jika N berharga 0.5 maka intensitas gambar akan dikurangi menjadi setengah.

Tabel 4. Tabel fungsi tombol-tombol menu

Tombol	*Fungsi Tombol*
	Untuk membuka file grafik dengan ekstension BMP.
	Untuk menyimpan file gambar dengan nama yang sama.
	Untuk mencetak gambar yang ditampilkan di editor.
	Untuk mengembalikan keadaan gambar ke posisi awal saat gambar di buka.
	Untuk memberi efek *mask lightning* pada semua bagian gambar.
	Untuk membatalkan perintah **Pilih Semua**.
	Untuk melakukan proses pemberian efek pencahayaan pada lokasi dalam bidang gambar yang telah dipilih.
	Untuk keluar dari program.
	Untuk menampilkan keterangan cara menggunakan program.



Gambar 6. Tampilan Sistem

Foto yang akan diberi efek *mask lightning* tidak akan mengalami perubahan ukuran karena perubahan warna tidak mempengaruhi ukuran file, yang akan mempengaruhi ukuran file adalah jumlah piksel. Metode *linear addition* berhasil diimplementasikan untuk pemberian efek *mask lightning* pada foto.

5. SIMPULAN DAN SARAN

5.1 Simpulan

Pada akhir dari makalah ini dapat disimpulkan bahwa:

a. Metode *linear addition* dapat diimplementasikan untuk memberi efek *mask lightning* (cahaya berbentuk) pada suatu foto, sehingga membuat bagian foto seolah-olah terkena efek cahaya yang bentuknya beragam.
b. Program ini telah menyediakan beberapa *mask lightning*, sehingga user tinggal memilih salah satu yang diinginkan dan dalam program juga terdapat menu untuk mengambil *mask lightning* dari file lain untuk kemudian ditambahkan ke program sehingga bentuk-bentuk mask dapat dirancang sendiri, untuk pilihan yang kedua diperlukan kreatifitas pengguna program untuk menghasilkan variasi-variasi *mask lightning* yang diinginkan.
c. Program selalu menghasilkan *output* dari masukan yang diberikan.
d. Besar file foto asli dengan foto yang telah diberi efek sama.

5.2 Saran

Hasil penelitian penerapan metode *linear addition* untuk pembuatan efek *mask lightning* masih terdapat beberapa kekurangan. Untuk lebih menyempurnakan program ini dapat ditambah dengan beberapa kemampuan dan fasilitas, antara lain:

a. Program sebaiknya dilengkapi dengan fasilitas untuk membuka format file gambar yang lain (selain BMP) seperti GIF,PCX, JPEG, TIFF, dan lain-lain.
b. Untuk pengembangan lebih lanjut, dalam program ini bisa diberikan fasilitas untuk memberikan efek lain pada gambar selain efek mask lightning seperti efek emboss, efek *blur*, efek *sharpenning*, *edge detection*, dan lain-lain.
c. Untuk lebih mempercepat proses perhitungan metode *linear addition* lebih baik jika menggunakan bahasa *assembly* untuk proses filteringnya.

PUSTAKA

[1] Rogers, F. D., dan Adams, A. J. (2002) *Mathematical Elements for Computer Graphics*, Tata McGraw-Hill, New Delhi
[2] Hearn, D., dan Baker, P. M. (1996) *Computer Graphics C Version*, Prentice-Hall, New Jersey
[3] Santosa, I. (1999) *Grafika Komputer dan Antarmuka Grafis*, Andi Offset, Yogyakarta
[4] Nalwan, A. (1997) *Pengolahan Gambar Secara Digital*, PT Elex Media Komputindo, Jakarta
[5] Pacheco, X., Teixeira, S. (2002) *Borland Delphi 6 Developer's Guide*, Sams Publiser, New York
[6] Kadir, A. (1999) *Borland Delphi Lengkap*, Andi, Yogyakarta.