BAB III
OUTPUT PRIMITIF

OBJEKTIF :

Pada Bab ini mahasiswa mempelajari tentang :

1. **Primitif Grafis**
2. **Algoritma Pembentukan Garis**
3. **Algoritma Pembentukan Lingkaran**
4. **Algoritma Pembentukan Ellips**

TUJUAN DAN SASARAN:

Setelah mempelajari bab ini mahasiswa diharapkan:

1. **Memahami objek grafis dua dimensi**
2. **Memahami algoritma pembentukan garis**
3. **Memahami Algoritma Pembentukan Lingkaran**
4. **Memahami Algoritma Pembentukan Ellips**
5. **Mengimplementasikan algoritma yang telah dipelajari dengan menggunakan Java**

WAKTU dan TEMPAT

1. **4 (Empat) kali pertemuan**
2. **8 x 50 menit pertemuan di kelas**
3. **16 x 50 menit latihan di rumah**

3.1 Primitif Grafis

Secara umum algoritma grafis memiliki persamaan yaitu bagaimana menampilkan hasil. Primitif grafis yang umum dijelaskan pada tabel berikut :

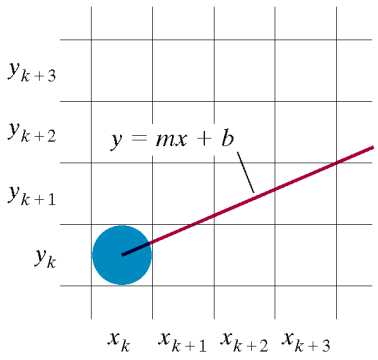
OBJEK GRAFIS	PRIMITIF
Pixel (Dot)	Posisi (x,y), Warna
Garis (Line)	Posisi (x1,y1,x2,y2), Warna, Thickness, Pattern
Lingkaran (Circle)	Pusat(x,y), Radius, Warna, Thickness, Pattern
Ellipse	Pusat(x,y), Radius Horisonal/Vertikal, Warna, Thickness, Pattern
Kurva	Teratur/Tidak teratur (Bezier)
Character	Type, Slanted, Thickness, Color, dll

Pada bab ini akan dibahas beberapa algoritma untuk pembentukan garis, lingkaran dan ellips.

3.2 Algoritma Pembentukan Garis

Garis dibuat dengan menentukan dua endpoint atau posisi titik awal dan akhir dari suatu garis. Kemudian peralatan output membuat garis sesuai posisi titik-titik tersebut. Untuk peralatan analog seperti plotter dan random-scan display garis lurus dapat dihasilkan dengan halus.

Pada peralatan digital garis lurus dihasilkan dengan menetapkan titik diskrit antara titik awal dan akhir. Posisi titik diskrit sepanjang garis lurus data diperhitungkan dari persamaan garis tersebut.



Untuk menentukan nilai suatu titik, dapat digunakan prosedur dasar dimana x sebagai nilai kolom pixel dan y sebagai nilai scan line sebagai berikut :

setPixel(x,y)

bila nilai x dan y sudah tersimpan pada frame buffer untuk dapat menampilkannya pada layer menggunakan fungsi dasar

getPixel(x,y)

3.2.1 Algoritma DDA

Algoritma Digital Differential Analyzer (DDA) adalah algoritma pembentukan garis berdasarkan perhitungan dx maupun dy dengan menggunakan rumus dy = m.dx. Garis dibuat dengan menentukan dua endpoint yaitu titik awal dan titik akhir. Setiap koordinat titik yang membentuk garis diperoleh dari perhitungan kemudian dikonversikan menjadi nilai integer.

Keuntungan dari algoritma ini adalah tidak perlu menghitung koordinat berdasarkan persamaan yang lengkap (menggunakan metode offset). Sedangkan kerugiannya adalah adanya akumulasi Round-off errors, sehingga garis akan melenceng dari garis lurus, selain itu operasi round-off juga menghabiskan waktu.

Algoritma pembentukan garis DDA adalah sebagai berikut :

```
void lineDDA (int x0, int y0, int xEnd, int yEnd)
  {
    int dx = xEnd - x0,  dy = yEnd - y0,  steps,  k;
    float xIncrement, yIncrement, x = x0, y = y0;
    if (fabs (dx) > fabs (dy))
      steps = fabs (dx);
    else
      steps = fabs (dy);
    xIncrement = float (dx) / float (steps);
    yIncrement = float (dy) / float (steps);
    setPixel (round (x), round (y));
    for (k = 0; k < steps; k++) {
      x += xIncrement;
      y += yIncrement;
      setPixel (round (x), round (y));
    }
  }
```

Contoh penerapan algoritma DDA :

Contoh 1 : garis dengan endpoint (1,3,8,5)

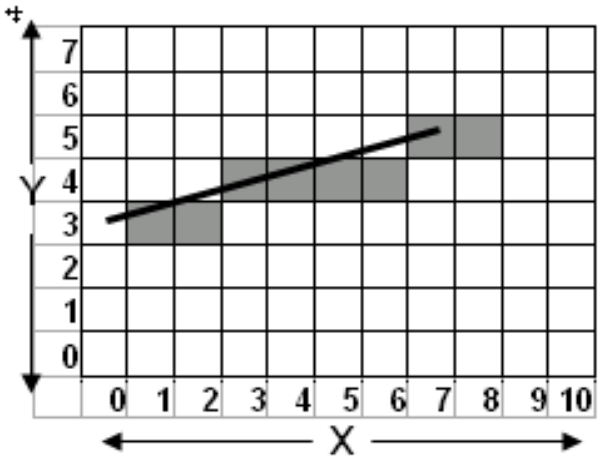
$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{5-3}{8-1} = \frac{2}{7}$$

$$c = y_1 - m.x_1 = 3 - \frac{2}{7}.1 = \frac{19}{7}$$

$$y = \frac{2}{7}.x \square \frac{19}{7}$$

Xi	Yi	Yi (dibulatkan)
1	3.00	3
2	3.29	3
3	3.57	4
4	3.86	4
5	4.14	4
6	4.43	4
7	4.71	5
8	5.00	5

Penggambaran garis :



3.2.2 Algoritma Bresenham

Algoritma Bresenham merupakan algoritma penggambaran garis yang efisien dengan menggunakan perhitungan incremental integer.

Prinsip dari algoritma Bresenham adalah :

1. Sumbu vertikal memperlihatkan posisi scan line.
2. Sumbu horizontal memperlihatkan kolom pixel
3. Pada tiap langkah, penentuan pixel selanjutnya didasari oleh parameter integer yang nilainya proporsional dengan pengurangan antara vertical separations dari dua posisi piksel dari nilai actual.

Algoritma Bresenham diuraikan sebagai berikut untuk $0<m<1$:

1. Tentukan dua endpoint garis dalam bentuk koordinat pixel. Endpoint sebelah kiri merupakan posisi (x_0,y_0) sedangkan endpoint kanan adalah (x_m,y_m), dimana n adalah jumlah pixel yang akan digambar setelah titik awal.
2. hitung konstanta $2\Delta x$, $2\Delta y - 2\Delta x$ dimana $\Delta y = y_n - y_0$ dan $\Delta x = x_n - x_0$
3. Plot (x_0,y_0) sebagai titik awal

4. Hitung parameter keputusan p_k (dimulai dari k = 0) dan gunakan parameter tersebut untuk mencari y selanjutnya dengan ketentuan sebagai berikut :

   Jika $p_k < 0$,

   titik selanjutnya adalah (x_k+1, y_k) dan $P_{k+1} = P_k + 2\Delta y$

   Jika $p_k >= 0$,

   titik selanjutnya adalah $((x_k+1, y_k+1)$ dan $P_{k+1} = p_k + 2\Delta y – 2\Delta x$

5. Ulangi langkah 4 sampai x_n tercapai

Contoh-contoh penerapan algoritma Bresenham :

Contoh 1 : Garis dengan titik (20,10,30,18)

$\Delta X = 10, \Delta Y = 8$

$Po = 2\Delta Y - \Delta X = 6$

$2.\Delta Y = 16$

$2\Delta Y - 2\Delta X = -4$

K	Pk	(Xk+1,Yk+1)
0	6	(21,11)
1	2	(22,12)
2	-2	(23,12)
3	14	(24,13)
4	10	(25,14)
5	6	(26,15)
6	2	(27,16)
7	-2	(28,16)
8	14	(29,17)
9	10	(30,18)

3.3 Algoritma Pembentukan Lingkaran

Lingkaran merupakan objek grafik yang paling sering digunakan pada grafik sederhana.

Lingkaran dapat didefinisikan sebagai kumpulan titik yang memiliki jarak r dari posisi pusat (x_c,y_c). Persamaan lingkaran dengan titik pusat (x_c,y_c) dan radius r dapat dispesifikasikan menggunakan koordinat rectangular berikut :

$$(x - x_c)^2 + (y-y_c)^2 = r^2$$

Lingkaran juga dapat didefinisikan menggunakan koordinat polar. Lingkaran yang sama dapat didefinisikan sebagai berikut :

$$x = r \cos \theta + x_c$$

$$y = r \sin \theta + y_c$$

dimana $0 \leq \theta \leq 2\pi$

Kita dapat menggambarkan lingkaran dengan menggunakan persamaan koordinat rectangular diatas, akan tetapi pendekatan ini menimbulkan dua masalah yaitu :

1. Persamaan tersebut mengandung perhitungan akar yang operasinya memakan waktu.
2. Timbul gap yang cukup signifikan pada lingkaran ketika digambarkan.

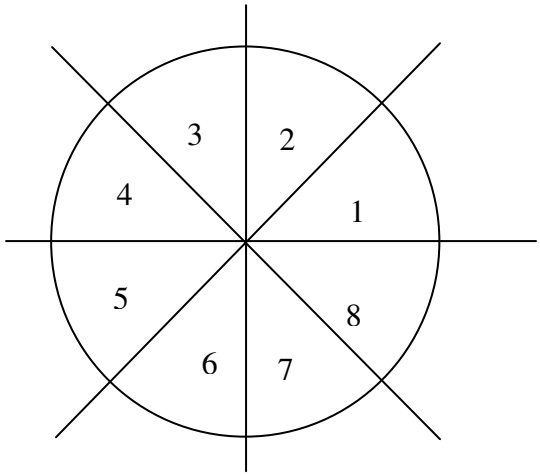
Lingkaran dapat juga digambarkan dengan menggunakan persamaan koordinat polar, tetapi fungsi trigonometri juga membutuhkan cost yang tidak sedikit sehingga algoritma yang disusun tidak akan efisien.

Untuk mengatasi masalah yang timbul dari penerapan koordinat polar maupun rectangular, Bresenham menyusun suatu algoritma pembentukan lingkaran yang hanya menggunakan aritmetika integer. Secara prinsip algoritma ini sejenis denga algoritma penggambaran garis yang disusun oleh orang yang sama.

Lingkaran merupakan objek yang simetris sehingga karakteristik ini dapat dimanfaatkan untuk mengurangi pekerjaan pada saat menggambar lingkaran. Lingkaran dibagi menjadi 8 oktan (lihat gambar 3.x), misalkan kita menyusun algoritma untuk menggambarkan lingkaran di oktan pertama, maka koordinat untuk 7 oktan selanjutnya dapat ditentukan pada table 3.1 berikut :

Tabel 3.1 Koordinat simetri 8 oktan

Oktan	x	y
1	x	y
2	-x	y
3	x	-y
4	-x	-y
5	y	x
6	-y	x
7	y	-x
8	-y	-x



Gambar 3.x Lingkaran dengan 8 oktan

Tahapan penggambaran lingkaran dengan menggunakan algoritma yang dikenal dengan nama algoritma midpoint ini adalah sebagai berikut :

1. Input jari-jari r dan koordinat pusat lingkaran (x_C, y_C), kemudian tntukan koordinat untuk titik awal yaitu **$(x_0, y_0) = (0, r)$.**
2. Hitung nilai awal untuk parameter keputusan **$p_0 = 1 - r$**
3. Untuk setiap x_k, mulai dari k=0, lakukan langkah berikut :

   jika $p_k<0$, maka titik selanjutnya pada lingkaran dengan pusat (0,0) adalah **$(x_k + 1, y_k)$ dan $p_{k+1} = p_k + 2\,x_{k+1} + 1$,**

   jika $p_k \geq 0$, titik berikutnya adalah **$(x_k + 1, y_k - 1)$**

   dan **$p_{k+1} = p_k + 2\,x_{k+1} + 1 - 2\,y_{k+1}$**

   dimana **$2\,x_{k+1} = 2\,x_k + 2$**, dan **$2\,y_{k+1} = 2\,y_k - 2$**
4. Tentukan titik simetri untuk 7 oktan lainnya dengan menggunakan table 3.x
5. Untuk lingkaran dengan pusat bukan di (0,0). Pindahkan setiap posisi pixel hasil perhitungan (x, y) dengan rumus **$x = x + x_c$, $y = y + y_c$**
6. Ulangi langkah 3 sampai 5, hentikan ketika **x >= y**

Contoh Penerapan algoritma midpoint untuk mengggambarkan lingkaran.

Contoh 1 : lingkaran dengan persamaan $X^2 + Y^2 = 100$

k	(X,Y)	2X	2Y	Pk
-	(0,10)	0	20	-9
0	(1,10)	2	20	-6
1	(2,10)	4	20	-1
2	(3,10)	6	20	6
3	(4,9)	8	18	-3
4	(5,9)	10	18	8
5	(6,8)	12	16	5
6	(7,7)	14	14	6

3.4 Algoritma Pembentukan Ellips

Ellips merupakan salah satu objek grafis dengan persamaan koordinat rectangular sebagai berikut :

$$\frac{(x-x_c)^2}{r_x}+\frac{(y-y_c)^2}{r_y}=1$$

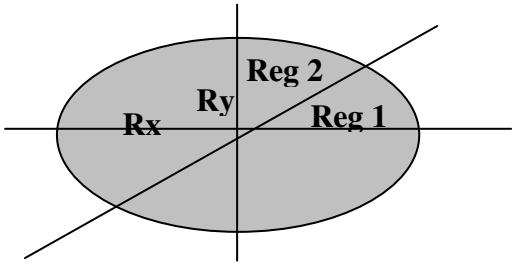
dan persamaan polar :

$$x = x_c + r_x.\cos\theta$$
$$y = y_c + r_y.\sin\theta$$

Teknik yang digunakan untuk menggambarkan garis dan lingkaran yang telah dibicarakan sebelumnya dapat diimplementasikan untuk menggambarkan ellips.

Elipps merupakan objek yang memiliki empat bagian yang simetris seperti digambarkan pada gambar 3.x. dari karakteristik ini, dapat disusun suatu algoritma yang memplot pixel di kuadran pertama dan menentukan titik di tiga kuadran lainnya.

Kuadran pertama dibagi menjadi 2 (dua) region dan dengan menggunakan algoritma midpoint ellipse, plot titik untuk region pertama, kemudian koordinat akhir pada region I menjadi koordinat awal untuk region II.



Region 1 dan 2 dapat digunakan dengan berbagai macam cara. Pertama dimulai dari posisi (0,r) dan melangkah searah jarum jam sepanjang jalur ellips pada kuadran pertama. Pergeseran dengan unit step dalam x pada saat slope lebih besar dari 1.

Alternative lain dimulai pada rx,0) dan seleksi titik dalam arah berlawanan dengan jarum jam pergeseran unit step y ke unit step x pada saat kemiringan lebih besar daripada -1.

Algoritma untuk menggambarkan ellips yang dikenal dengan sebutan Midpoint ellipse algorithm adalah sebagai berikut :

1. Input r_x, r_y dan pusat Ellips (x_c, y_c), tentukan titik pertama pada pusat ellips sebagai : $(x_0,y_0) = (0,R_y)$
2. Hitung nilai awal parameter keputusan di region 1 :

   $\mathbf{P1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r^2_x}$

3. Untuk semua x_k di region 1, dimulai dari k=0 lakukan tes berikut :

   jika $p1_k < 0$ titik selanjutnya dari ellips yang berpusat di (0,0) adalah

   (x_{k+1},y_k) dan $p1_{k+1} = p1_k + 2r^2_y x_{k+1} + r_y^2$

   jika p1k >=0 maka titik selanjutnya adalah :

   (x_{k+1},y_{k-1}) dan $p1_{k+1} = p1_k + 2r_x^2 y_{k+1} + r_y^2$

   dengan **$2r_y^2 x_{k+1} = 2r^2_y x_k + 2r_y^2$** dan **$2r_x^2 y_{k+1} = 2r^2_x y_k + 2r_x^2$**

4. Hitung nilai awal dari parameter keputusan di region 2 menggunakan titik akhir dari region 1 sebagai (x_0,y_0) dengan rumus :

   $P2_0 = r_y^2(x0 + ½)2 + r_x^2 (y0-1)2 - r_x^2 r_y^2$

5. Untuk setiap y_k di region 2 dimulai dari k=0 lakukan uji berikut :

   jika $p2_k < 0$ titik selanjutnya dari ellips yang berpusat di (0,0) adalah

   (x,y_k-1) dan $p2_{k+1} = p2_k - 2r_x^2 y_{k+1} + r_x^2$

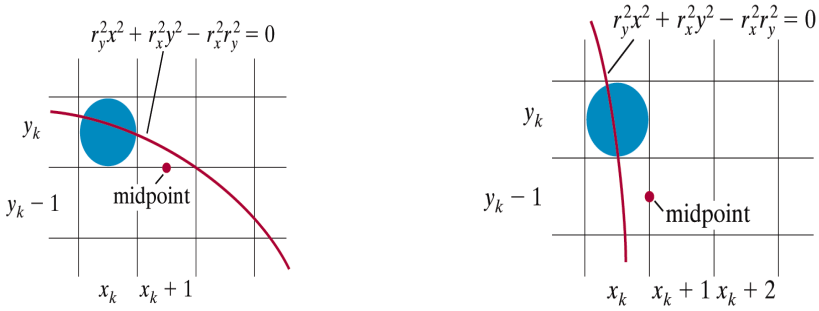
   jika p2k >=0 maka titik selanjutnya adalah :

   (x_k+1,y_k-1) dan $p2_{k+1} = p2_k + 2r_x^2 y_{k+1} + r_x^2$

6. Tentukan titik simetris pada tiga kuadran lainnya
7. Pindahkan posisi (x,y) ke titik pusat ellips (x_c, y_c) dengan rumus

   $x = x + x_c$ dan $y = y + y_c$

8. Ulangi langkah untuk region 1 sampai **$2r_y^2 x \geq 2r_x^2 y$**



Contoh penerapan algoritma penggambaran ellips

Contoh 1 :

Penggambaran Ellips dengan pusat (0,0), Rx = 8 dan Ry = 5

Region I

k	x	y	Px	Py	Pk
0	0	5	0	640	-279
1	1	5	50	640	-204
2	2	5	100	640	-79
3	3	5	150	640	96

k	x	y	Px	Py	Pk
4	4	4	200	512	-191
5	5	4	250	512	84
6	6	3	300	384	25
7	7	2	350	256	144

Region II

k	x	y	Px	Py	Pk
-	7	2	350	256	-129.75
0	8	1	400	128	206.25
1	8	0	400	0	270.25

Contoh 2

Penggambaran Ellips dengan pusat (0,0), Rx = 6 dan Ry = 2

Region I

k	x	y	Px	Py	Pk
0	0	2	0	144	-59
1	1	2	8	144	-47
2	2	2	16	144	-27
3	3	2	24	144	1
4	4	1	32	72	-35
5	5	1	40	72	9
6	6	0	48	0	61

Region II

Tidak ada karena y sudah = 0

3.5 Rangkuman

1. Objek grafik standar yang dibahas pada bab ini adalah titik, garis, lingkaran, dan ellips
2. Algoritma untuk menggambarkan garis adalah Bresenham dan DDA
3. Algoritma DDA adalah suatu algoritma pengkonversian suatu himpunan pixel menjadi suatu garis

4. Algoritma Bresenham merupakan algoritma penggambaran garis yang efisien dengan menggunakan perhitungan incremental integer.
5. Algoritma penggambaran lingkaran Bresenham membagi lingkaran menjadi 8 bagian yang simetris sehingga pixel yang perlu dihitung hanya pada bagian 1 saja.
6. Algoritma penggambaran ellips membagi ellips menjadi 2 region

3.6 Latihan Soal

1. Jelaskan perbedaan penggambaran garis menggunakan algoritma bresenham dan algoritma DDA
2. Modifikasi algoritma Bresenham untuk menggambarkan garis dengan gradien (m) lebih dari 1
3. Gunakan algoritma Bresenham dan DDA untuk menggambarkan garis (0,4,3,12)
4. Gambarkan lingkaran dengan persamaan berikut :
   a. $x^2 + y^2 = 64$
   b. $(x-3)^2 + (y-2)^2 = 25$
   c. $(x+1)^2 + (y+2)^2 = 36$
5. Gambarkan ellips dengan persamaan berikut :
   a. $\frac{x^2}{15} + \frac{y^2}{2} = 1$
   b. $\frac{(x-2)^2}{5} + \frac{(y-1)^2}{2} = 1$
   c. $\frac{(x+2)^2}{5} + \frac{(y+3)^2}{2} = 1$
6. Implementasikan algoritma pembentukan lingkaran menggunakan Java sehingga membentuk lingkaran penuh

3.7 Referensi

[1] Hearn, Donald, M. Pauline Baker, *Computer Graphics*, Prentice Hall.

[2] Rowe, Glenn W, *Computer Graphics with Java*, Palgrave, 2001

[3] Sutopo, Ariesto Hadi, Pengantar Grafika Komputer, Gava Media, 2002