Proses Clipping Menggunakan Algoritma Cohen-Sutherland pada Ruang Dimensi Tiga

Veronica Lusiana
Program Studi Teknik Informatika, Universitas Stikubank Semarang
Email: verolusiana@yahoo.com

Abstrak

Menentukan potongan garis yang perlu atau tidak perlu digambar di daerah jendela dikenal dengan istilah clipping. Salah satu algoritma clipping diusulkan oleh Danny Cohen and Ivan Sutherland pada tahun 1967 yaitu algoritma Cohen-Sutherland. Algoritma ini digunakan untuk menentukan apakah terdapat potongan garis yang digambar di dalam jendela dan sebaliknya akan menghilangkan potongan garis yang berada di luar jendela. Penggambaran obyek di jendela dapat menggunakan pendekatan dimensi dua atau dimensi tiga.

Analisa dilakukan terhadap proses pemotongan garis (clipping) pada ruang dimensi tiga menggunakan algoritma Cohen-Sutherland. Disediakan contoh enam buah garis dengan letak titik ujung bervariasi seperti pada kode pembagian area Cohen-Sutherland. Ruang dimensi tiga, ruas garis, dan hasil pemotongan garis dalam ruang dimensi tiga digambarkan menggunakan perangkat lunak Processing. Pada pendekatan dimensi tiga, untuk pengujian pada titik awal dan titik akhir garis maka algoritma Cohen-Sutherland membagi sebanyak 27 macam wilayah. Masing-masing wilayah diberi kode yang unik.

Kata kunci: clipping, algoritma Cohen-Sutherland, kode area Cohen-Sutherland.

PENDAHULUAN

Salah satu pokok bahasan yang cukup menarik pada grafika komputer (*computer graphics*) adalah bagaimana cara menggambarkan atau menampilkan obyek yang ada di dunia nyata atau dimensi tiga ke dalam media dimensi dua seperti layar monitor komputer. Ruang atau kedalaman yang dimiliki oleh obyek di dunia nyata masih dapat dilihat meskipun digambarkan di media dimensi dua. Obyek di dunia nyata yang akan digambarkan pada layar monitor komputer biasanya mempunyai ukuran yang lebih besar dibandingkan dengan ukuran media penampilnya, sehingga untuk menggambarkannya perlu dilakukan pemilihan dari bagian-bagian obyek.

Kita bisa menggunakan seluruh atau sebagian luas layar monitor komputer untuk menampilkan sebagian atau seluruh obyek secara utuh. Di dalam prakteknya, daerah layar monitor tidak harus secara penuh (*full screen*) untuk menampilkan obyek. Daerah yang dipilih untuk menampilkan obyek disebut dengan jendela atau *viewport* berbentuk persegi panjang. Penempatan dan luas jendela pada layar monitor bisa bervariasi.

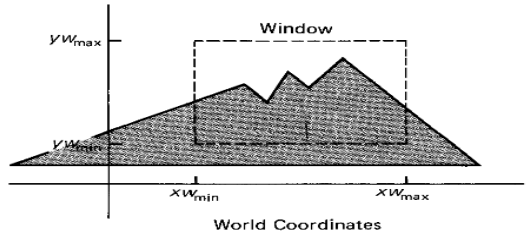
Penelitian ini akan difokuskan pada bagaimana menentukan obyek garis akan digambarkan atau tidak pada sebuah jendela. Disini garis mewakili suatu obyek yang ada di dunia nyata. Metode untuk menentukan potongan garis yang perlu atau tidak perlu digambar di daerah jendela dikenal dengan istilah *clipping*. Salah satu algoritma clipping diusulkan oleh Danny Cohen and Ivan Sutherland pada tahun 1967 yaitu algoritma Cohen-Sutherland. Algoritma ini cukup popular digunakan untuk menentukan apakah terdapat potongan garis yang digambar di dalam jendela, dan sebaliknya menghilangkan potongan garis yang berada di luar jendela. Penggambaran obyek di jendela dapat menggunakan

pendekatan dimensi tiga. Disediakan enam buah garis dengan posisi titik ujung bervariasi seperti pada kode pembagian area Cohen-Sutherland. Hasil pemotongan garis dalam dimensi tiga digambarkan menggunakan perangkat lunak Processing.

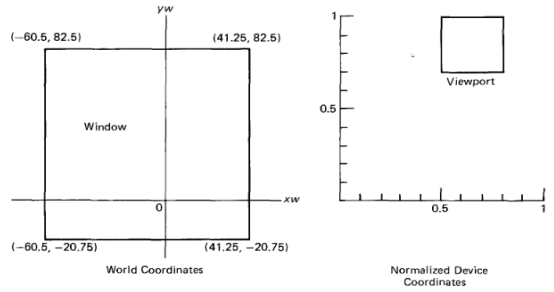
TINJAUAN PUSTAKA

Jendela (viewport)

Jendela (*viewport*) merupakan wilayah di layar monitor yang digunakan untuk menampilkan atau menggambar obyek. Jendela (viewport) bisa diubah luas dan posisinya disesuaikan dengan bagian gambar yang akan dilihat dan bagian dari layar yang akan dipilih untuk menggambarkan bagian obyek tersebut. Jika ukuran jendela diperbesar maka kita melihat obyek dengan lebih luas, demikian pula sebaliknya. Contoh sebuah jendela dapat dilihat pada Gambar 1, dengan pengalamatan atau koordinat jendela dapat dilihat pada Gambar 2.



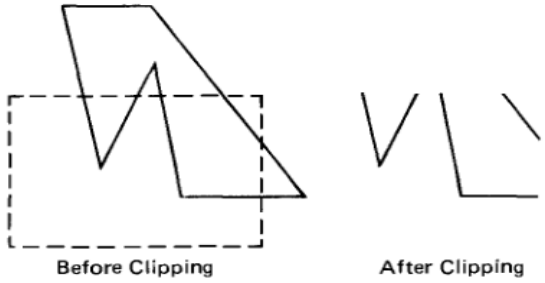
Gambar 1 Jendela yang menampilkan sebagian obyek (Hearn dan Baker, 1986)



Gambar 2. Jendela dan sistem koordinat (Hearn dan Baker, 1986)

Pemotongan garis (*clipping*)

Pemotongan (*clipping*) perlu dilakukan terhadap sebagian dari obyek yang akan digambarkan pada jendela, yaitu apabila obyek yang ditampilkan memiliki luas yang lebih besar dibandingkan dengan luas jendela. Pada prinsipnya pemotongan ini akan memisahkan bagian obyek yang tampak dan bagian yang tidak tampak pada jendela. Pada Gambar 3 dapat dilihat contoh pemotongan terhadap obyek rangka dimensi dua, setelah pemotongan maka obyek digambar mengikuti luas jendela. Berikut ini adalah penjelasan bagaimana proses pemotongan terhadap titik dan garis. Seperti telah kita ketahui bahwa penyusun terkecil dari gambar adalah titik dan garis yang ditampilkan dengan intensitas warna dan pada lokasi tertentu (Santosa, 1996; Nugroho, 2005; Agoston, 2005).



Gambar 3. Pemotongan pada obyek rangka dimensi dua (Hearn dan Baker, 1986)

Pemotongan titik

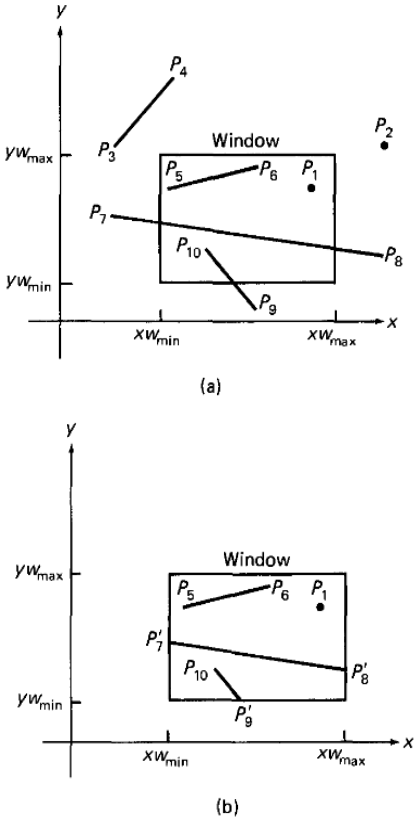
Pemotongan titik dilakukan untuk mengetahui apakah suatu titik terdapat di dalam atau di luar jendela. Apabila titik terletak di dalam jendela maka titik tersebut akan ditampilkan, apabila sebaliknya maka titik tersebut tidak ditampilkan. Dengan menggunakan jendela berbentuk persegi panjang atau bujur sangkar yang memiliki koordinat batas kiri bawah (X*min*, Y*min*) dan batas kanan atas (X*max*, Y*max*), maka titik T (X_t, Y_t) akan ditampilkan di dalam jendela jika kedua pertidaksamaan berikut ini terpenuhi:

$$Xmin <= X_t <= Xmax \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

$$Ymin <= Y_t <= Ymax \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Dengan kata lain jika satu atau kedua pertidaksamaan di atas tidak terpenuhi maka

titik T tidak akan ditampilkan. Pada Gambar 4 dapat dilihat contoh titik P1 berada di dalam jendela dan titik P2 di luar jendela.



Gambar 4. Pemotongan pada obyek titik dan garis dimensi dua (Hearn dan Baker, 1986)

Pemotongan garis

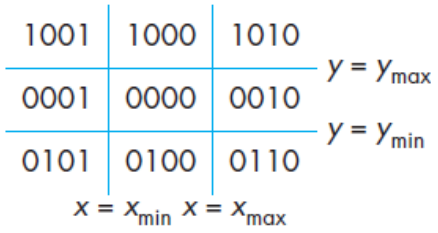
Pemotongan garis dilakukan jika terdapat bagian garis yang terletak di luar jendela. Garis disusun oleh dua buah titik atau lebih. Untuk menguji apakah sebuah garis akan ditampilkan di dalam jendela maka kita dapat menguji seluruh titik yang menyusun garis tersebut. Kelemahan dari cara ini adalah metode ini kurang efisien untuk diterapkan jika garis tersebut cukup panjang dan banyak jumlahnya. Pada Gambar 4(a) dapat dilihat beberapa buah garis dalam tiga kondisi yang berbeda yaitu:

a. Garis yang ditampilkan seluruhnya di jendela (garis P_5-P_6)
b. Garis yang tidak ditampilkan di jendela (garis P_3-P_4)
c. Garis yang ditampilkan sebagian di jendela (garis P_2-P_8, P_9-P_{10})

Setelah dilakukan proses pemotongan maka hasilnya dapat dilihat pada Gambar 4(b), terdapat tiga buah garis dan sebuah titik yang berada di dalam jendela.

Algoritma Cohen Sutherland mengusulkan cara pengujian pada titik awal dan titik akhir garis untuk menentukan apakah sebuah garis berada di dalam atau di luar jendela, tanpa harus menguji seluruh titik penyusun garis tersebut. Algoritma ini membagi sebanyak 9 macam wilayah yang digunakan untuk memetakan titik awal dan titik akhir garis yang akan digambar menggunakan pendekatan dimensi dua (Pandey dan Jain, 2013). Pada pendekatan dimensi tiga, untuk keperluan yang sama maka algoritma ini membagi sebanyak 27 macam wilayah, dimana setiap wilayah diberi kode yang unik.

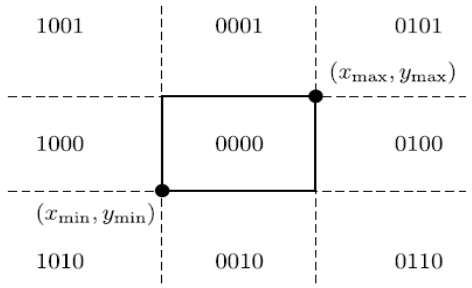
Pada Gambar 5 merupakan contoh penamaan kode 4 bit untuk setiap wilayah pada dimensi dua. Setiap bit dapat bernilai 0 atau 1 digunakan untuk menyimpan informasi letak titik. Empat bit ini jika dibaca berurutan menyimpan informasi [atas/top (bit ke 3), bawah/bottom (bit ke 2), kanan/right (bit ke 1), kiri/left (bit ke 0)].



Gambar 5. Kode 4 bit algoritma Cohen-Sutherland (Angel dan Shreiner, 2012)

Deskripsi wilayah tengah diberi kode 0000 berarti: tidak di atas y_{max}, tidak di bawah y_{min}, tidak di sebelah kanan x_{max}, dan tidak di sebelah kiri x_{min}. Wilayah ini merupakan jendela tempat garis digambarkan, apabila terdapat garis atau potongan garis di luar wilayah ini maka tidak akan digambar. Deskripsi untuk wilayah paling kiri bawah diberi kode 0101 berarti: tidak di atas y_{max}, ya di bawah y_{min}, tidak di sebelah kanan x_{max}, dan ya di sebelah kiri x_{min}. Sedangkan untuk wilayah paling kanan atas

diberi kode 1010 berarti: ya di atas y_{max}, tidak di bawah y_{min}, ya di sebelah kanan x_{max}, dan tidak di sebelah kiri x_{min}. Demikian seterusnya untuk wilayah yang lain. Alternatif penamaan kode yang lain dapat dilihat pada Gambar 6.



Gambar 6 Kode 4 bit algoritma Cohen-Sutherland (klawonn, 2012)

Berdasarkan pembagian wilayah pada algoritma Cohen-Sutherland maka hasil uji sebuah garis dapat dibagi menjadi tiga kategori, yaitu:

a. Garis terlihat seluruhnya (*fully visible*)

Garis terlihat seluruhnya apabila titik awal dan akhir garis berada di dalam jendela. Garis yang memiliki titik ujung T1 (X_{t1}, Y_{t1}) dan T2 (X_{t2}, Y_{t2}) akan terletak di dalam *viewport* jika memenuhi kedua pertidaksamaan berikut ini:

$X_{min} <= X_{t1}, X_{t2} <= X_{max}$…….(3)

$Y_{min} <= Y_{t1}, Y_{t2} <= Y_{max}$……..(4)

Melihat pada kode pembagian wilayah pada Gambar 5 maka garis terlihat jika kedua titik ujungnya mempunyai kode 0000.

b. Garis tidak terlihat sama sekali (*fully invisible*)

Garis tidak terlihat sama sekali atau seluruh garis terletak di luar jendela. Garis yang memiliki titik ujung T1 (X_{t1}, Y_{t1}) dan T2 (X_{t2}, Y_{t2}) akan terletak di luar jendela jika salah satu dari empat pertidaksamaan berikut ini dipenuhi:

$X_{t1}, X_{t2} > X_{max}$……..(5)

$X_{t1}, X_{t2} < X_{min}$……..(6)

$Y_{t1}, Y_{t2} > Y_{max}$…… (7)

$Y_{t1}, Y_{t2} < Y_{min}$……..(8)

Melihat pada kode pembagian wilayah pada Gambar 5 maka garis tidak terlihat jika terdapat nilai 1 pada urutan bit yang sama dari kedua titik ujung garis sehingga hasil operasi AND dari kedua ujung garis tidak sama dengan 0000.

c. Garis terlihat sebagian (*partially visible* atau *clipping candidate*)

Garis terlihat sebagian atau ada sebagian garis yang tidak tampak di dalam jendela, disini garis menjadi kandidat untuk dipotong. Melihat pada kode pembagian wilayah pada Gambar 5 maka kondisi ini terjadi jika syarat a dan b tidak terpenuhi. Pada kondisi ini, setelah diuji dapat juga terjadi ternyata garis tidak tampak seluruhnya.

Algoritma Cohen-Sutherland pada Ruang Dimensi Tiga

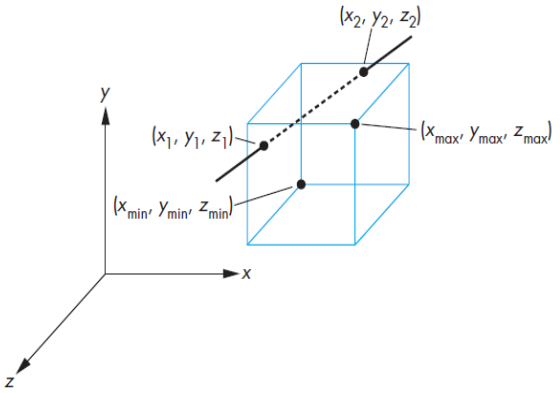
Pada pendekatan dimensi tiga, untuk pengujian pada titik awal dan titik akhir garis maka algoritma Cohen-Sutherland membagi sebanyak 27 macam wilayah. Ini adalah perluasan dari pendekatan dimensi dua. Masing-masing wilayah diberi kode yang unik (Kodituwakku, dkk., 2012). Pada Gambar 7 dapat dilihat ruas garis pada dimensi tiga. Pada Gambar 8 digunakan untuk menyusun deskripsi dari 27 kode algoritma Cohen-Sutherland pada dimensi tiga.

Pada Gambar 9 merupakan contoh penamaan kode 6 bit untuk setiap wilayah pada dimensi tiga. Setiap bit dapat bernilai 0 atau 1 digunakan untuk menyimpan informasi letak titik. Enam bit ini jika dibaca berurutan menyimpan informasi [depan/front (bit ke 5), belakang/back (bit ke 4), atas/top (bit ke 3), bawah/bottom (bit ke 2), kanan/right (bit ke 1), kiri/left (bit ke 0)].

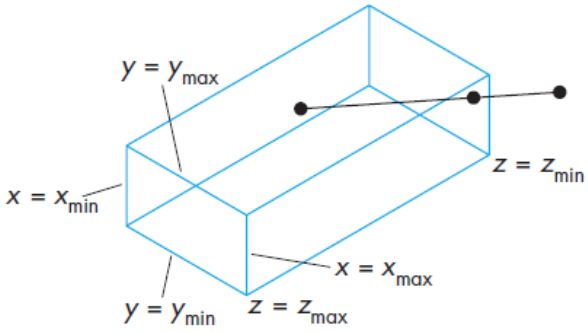
Deskripsi wilayah tengah diberi kode 000000 berarti: tidak di depan z_{max}, tidak di belakang z_{min}, tidak di atas y_{max}, tidak di bawah y_{min}, tidak di sebelah kanan x_{max}, dan tidak di sebelah kiri x_{min}. Wilayah ini merupakan jendela tempat garis digambarkan, apabila terdapat garis atau potongan garis di luar wilayah ini maka

tidak akan digambarkan.

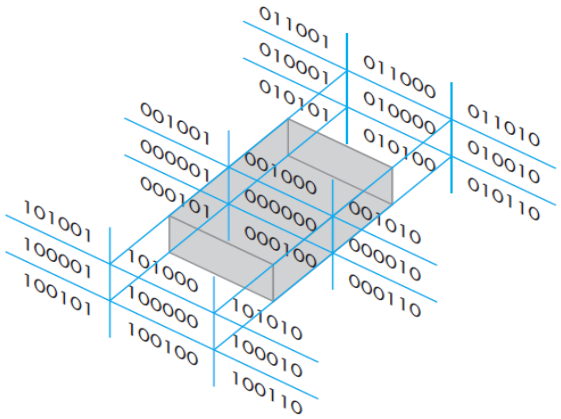
Deskripsi untuk wilayah paling belakang kiri bawah diberi kode 010101 berarti: tidak di depan z_{max}, ya dibelakang z_{min}, tidak di atas y_{max}, ya di bawah y_{min}, tidak di sebelah kanan x_{max}, dan ya di sebelah kiri x_{min}. Sedangkan untuk wilayah paling depan kanan atas diberi kode 101010 berarti: ya di depan z_{max}, tidak di belakang z_{min}, ya di atas y_{max}, tidak di bawah y_{min}, ya di sebelah kanan x_{max}, dan tidak di sebelah kiri x_{min}. Demikian seterusnya untuk wilayah yang lain.



Gambar 7. Ruas garis pada ruang dimensi tiga (Angel dan Shreiner, 2012)



Gambar 8. Deskripsi kode algoritma Cohen-Sutherland pada ruang dimensi tiga (Angel dan Shreiner, 2012)

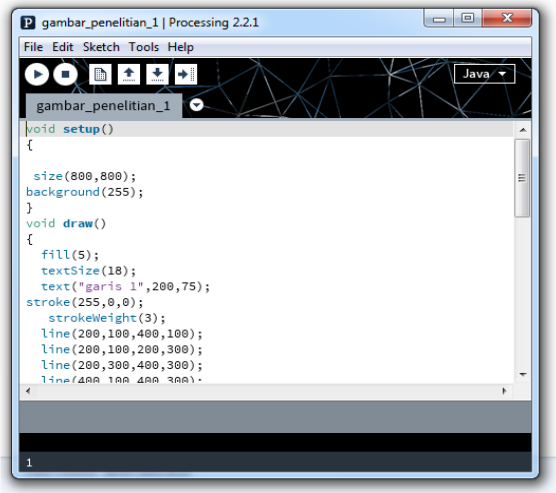


Gambar 9. Kode 6 bit algoritma Cohen-Sutherland (Angel dan Shreiner, 2012)

Pengantar Processing

Processing adalah sebuah bahasa pemrograman yang bersifat terbuka (*open source*) dan menyediakan lingkungan pemrograman yang terintegrasi (*integrated development environment*). Lingkungan pemrograman Processing terdiri dari: teks editor untuk menuliskan baris perintah atau dengan cara membuka file program, proses kompilasi perintah, dan jendela untuk menampilkan hasil keluaran (*output*) program beserta proses interaksi dengan pemakai. Processing digunakan untuk menampilan hasil proses pemotongan garis menggunakan algoritma Cohen-Sutherland.

Jendela utama Processing dapat dilihat pada Gambar 10. Perangkat lunak ini banyak digunakan untuk membuat program yang mengimplementasikan antara lain teori grafika komputer, pengolahan citra, animasi dan interaksi gambar, serta pemodelan grafis (Shiffman, 2008; Reas dan Fry, 2007). Pengembangan Processing diinisiasi oleh Ben Fry dan Casey Reas. Berkembang dari ide-ide yang dieksplorasi di Aesthetics and Computation Group (ACG) di MIT Media Lab pada tahun 2001 sampai dengan 2004. Kini tersedia untuk sistem operasi GNU/Linux 32/64 bit, Mac OS X, dan Windows 32/64 bit, dengan versi stabil terbaru adalah 2.2.1 yang dirilis pada 19 Mei 2014.



Gambar 10. Jendela utama Processing

PROSES PENELITIAN

Proses penelitian dilakukan dengan cara menggambar enam buah ruas garis secara terpisah pada ruang dimensi tiga. Terdapat ruas garis yang berada di dalam maupun di luar jendela. Berikut ini adalah baris perintah (*source code*) untuk dua buah contoh garis pada ruang dimensi tiga yaitu Garis 1 dan Garis 2.

Baris perintah Garis 1

```
void setup()
{
  size(800,800); background(255);
}
void draw()
{
  fill(5); textSize(18);
  text("garis 1",200,75);
  stroke(255,0,0);
  strokeWeight(3);
  line(200,100,400,100);
  line(200,100,200,300);
  line(200,300,400,300);
  line(400,100,400,300);
  line(200,100,100,250);
  line(100,250,100,450);
  line(100,450,200,300);
  line(100,250,300,250);
  line(300,250,300,450);
  line(300,450,100,450);
  line(300,250,400,100);
  line(300,450,400,300);
  stroke(0,255,255);
  line(400,50,25,350);
  stroke(0,0,255);
  strokeWeight(10);
  point(335,100);
  point(125,268);
}
```

Baris perintah Garis 2

```
void setup()
{
  size(800,800); background(255);
}
void draw()
{
  fill(5); textSize(18);
  text("garis 2",200,500);
  stroke(255,0,0);
  strokeWeight(3);
  line(200,100,400,100);
  line(200,100,200,300);
  line(200,300,400,300);
  line(400,100,400,300);
  line(200,100,100,250);
  line(100,250,100,450);
  line(100,450,200,300);
  line(100,250,300,250);
  line(300,250,300,450);
  line(300,450,100,450);
  line(300,250,400,100);
  line(300,450,400,300);
  stroke(0,255,255);
  line(25,400,500,400);
  stroke(0,0,255);
  strokeWeight(10);
  point(300,400);
  point(100,400);
}
```

HASIL DAN PEMBAHASAN

Penyusunan kode Cohen-Sutherland dapat dilihat pada Gambar 8 deskripsi kode algoritma Cohen-Sutherland pada ruang dimensi tiga, dan pada Gambar 9 kode enam bit algoritma Cohen-Sutherland. Disini, setiap bit dapat bernilai 0 atau 1 yang digunakan untuk menyimpan informasi letak titik awal dan akhir sebuah garis. Enam bit tersebut digunakan untuk menyimpan informasi, yaitu:

a. bit ke 5 [depan/front; 0 = tidak di depan z_{max}; 1 = di depan z_{max}]

b. bit ke 4 [belakang/back; 0 = tidak di belakang z_{min}; 1 = di belakang z_{min}]

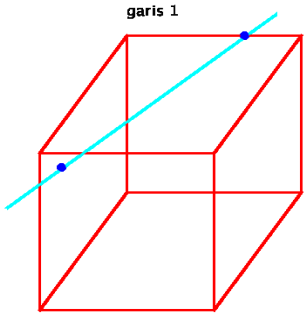
c. bit ke 3 [atas/top; 0 = tidak di atas y_{max}; 1 = di atas y_{max}]

d. bit ke 2 [bawah/bottom; 0 = tidak di bawah y_{min}; 1 = di bawah y_{min}]

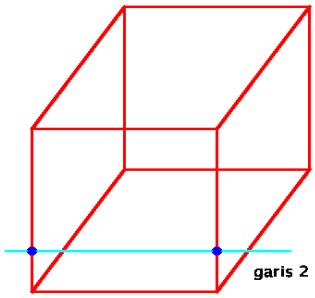
e. bit ke 1 [kanan/right; 0 = tidak di sebelah kanan x_{max}; 1 = di sebelah kanan x_{max}]

f. bit ke 0 [kiri/left; 0 = tidak di sebelah kiri x_{min}; 1 = tidak di sebelah kiri x_{min}]

Pada Gambar 11, seluruh potongan Garis 1 disusun dari ujung garis yang berada di kiri bawah (u_i) sampai dengan ujung garis yang berada kanan atas (u_{ii}). Kode untuk (u_i) adalah 100001 sedangkan kode untuk (u_{ii}) adalah 011000. Ruas garis yang berada di dalam jendela dimensi tiga memiliki batas {$z=z_{max}$} di titik awal dan {$y=y_{max}$; $z=z_{min}$} di titik akhir pada batas jendela tersebut. Garis yang ada diluar batas jendela tersebut akan dipotong atau tidak ditampilkan.

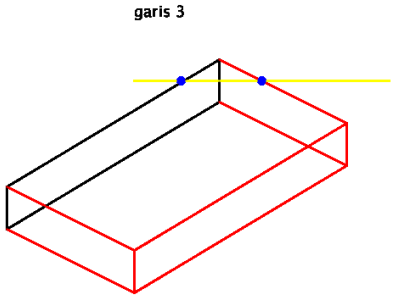


Gambar 11. Garis 1 pada ruang dimensi tiga



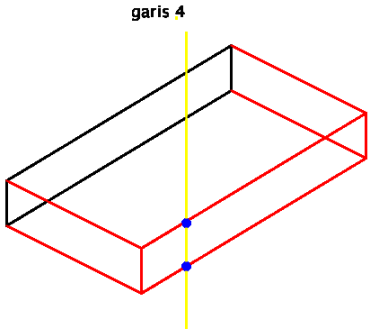
Gambar 12 Garis 2 pada ruang dimensi tiga

Pada Gambar 12, seluruh potongan Garis 2 disusun dari ujung garis yang berada di kiri (u_i) sampai dengan ujung garis yang berada di kanan (u_{ii}). Kode untuk (u_i) adalah 000001 sedangkan kode untuk (u_{ii}) adalah 000010. Ruas garis yang berada di dalam jendela dimensi tiga memiliki batas {$x=x_{min}$; $z=z_{max}$} di titik awal dan {$x=x_{max}$; $z=z_{max}$} di titik akhir pada batas jendela tersebut. Garis yang ada diluar batas jendela tersebut akan dipotong atau tidak ditampilkan.



Gambar 13. Garis 3 pada ruang dimensi tiga

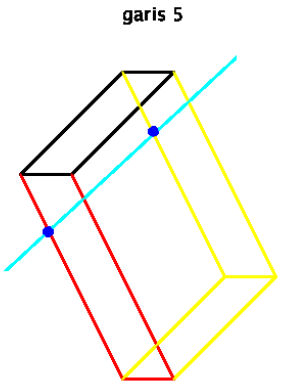
Pada Gambar 13, seluruh potongan Garis 3 disusun dari ujung garis yang berada di kiri (u_i) sampai dengan ujung garis yang berada di kanan (u_{ii}). Kode untuk (u_i) adalah 000001 sedangkan kode untuk (u_{ii}) adalah 010110. Ruas garis yang berada di dalam jendela dimensi tiga memiliki batas {$x=x_{min}$; $z=z_{min}$} di titik awal dan {$y=y_{max}$; $z=z_{min}$} di titik akhir pada batas jendela tersebut. Garis yang ada diluar batas jendela tersebut akan dipotong atau tidak ditampilkan.



Gambar 14 Garis 4 pada ruang dimensi tiga

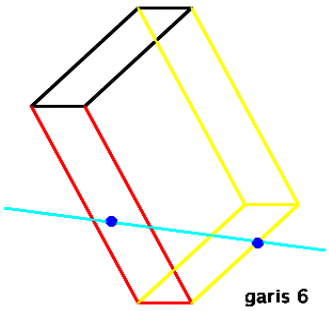
Pada Gambar 14, seluruh potongan Garis 4 disusun dari ujung garis yang berada di atas (u_i) sampai dengan ujung garis yang berada di bawah (u_{ii}). Kode untuk (u_i) adalah 001000 sedangkan kode untuk (u_{ii}) adalah 000100. Ruas garis yang berada di dalam jendela dimensi tiga memiliki batas {$x=x_{max}$; $y=y_{max}$} di titik awal dan {$x=x_{max}$; $y=y_{min}$} di titik akhir pada batas

jendela tersebut. Garis yang ada diluar batas jendela tersebut akan dipotong atau tidak ditampilkan.



Gambar 15. Garis 5 pada ruang dimensi tiga

Pada Gambar 15, seluruh potongan Garis 5 disusun dari ujung garis yang berada di kiri bawah (u_i) sampai dengan ujung garis yang berada di kanan atas (u_{ii}). Kode untuk (u_i) adalah 000001 sedangkan kode untuk (u_{ii}) adalah 000010. Ruas garis yang berada di dalam jendela dimensi tiga memiliki batas {$x=x_{min}$; $z=z_{min}$} di titik awal dan {$x=x_{max}$; $z=z_{min}$} di titik akhir pada batas jendela tersebut. Garis yang ada diluar batas jendela tersebut akan dipotong atau tidak ditampilkan.



Gambar 16 Garis 6 pada ruang dimensi tiga

Pada Gambar 16, seluruh potongan Garis 6 disusun dari ujung garis yang berada di kiri (u_i) sampai dengan ujung garis yang berada di kanan (u_{ii}). Kode untuk (u_i) adalah 000001 sedangkan kode untuk (u_{ii}) adalah 101000. Ruas garis yang berada di dalam jendela dimensi tiga memiliki batas {$x=x_{min}$} di titik awal dan {$z=z_{max}$} di titik akhir pada batas jendela tersebut. Garis yang ada diluar batas jendela tersebut akan dipotong atau tidak ditampilkan.

KESIMPULAN DAN SARAN

Kesimpulan dan saran penelitian ini adalah:

1. Melalui penelitian ini maka dapat dipahami langkah-langkah algortima Cohen-Sutherland untuk menentukan dan memahami arti kode titik awal dan titik akhir garis pada ruang dimensi tiga, serta dapat menampilkan secara visual garis yang berada di dalam dan di luar jendela penampil.
2. Pengembangan dapat dilakukan dengan membuat perangkat lunak visualisasi jendela penampil (*windowing*, *viewport*) dan pemotongan garis (*clipping*) yang memiliki antar muka pengguna (*user interface*) lebih menarik dan interaktif.

DAFTAR PUSTAKA

Agoston, M.K., (2005). *Computer Graphics and Geometric Modeling: Implementation and Algorithms*. Springer-Verlag.

Angel, E., Shreiner, D. (2012). *Interactive Computer Graphics a Top-Down Approach with Shader-Based OpenGL 6th ed*. Addison-Wesley.

Hearn, D., Baker, M.P. (1986). *Computer Graphics*. Prentice-Hall International, USA.

Klawonn, F. (2012). *Introduction to Computer Graphics Using Java 2D and 3D 2nd ed.* Springer-Verlag London.

Kodituwakku, R., Wijeweera, K.R., Chamikara, M.A.P. (2012). An Efficient Line Clipping Algorithm for 3D Space. *International Journal of Advanced Research in Computer Science and Software Engineering*. Volume 2, Issue 5, May 2012.

Nugroho, E. (2005). *Teori dan Praktek Grafika Komputer Menggunakan Delphi dan OpenGL*. Yogyakarta: Penerbit Graha Ilmu.

Pandey, A., Jain, S. (2013). Comparison of Various Line Clipping Algorithm for Improvement. *International Journal of Modern Engineering Research* (IJMER). pp-69-74, Vol.3, Issue.1, Jan-Feb 2013.

Reas, C., Fry, B. (2007). *Processing: a programming handbook for visual designers and artists.* The MIT Press, Massachusetts.

Santosa, P.I. (1996). *Grafika komputer dan antarmuka grafis: teknik penyusunan program aplikasi berbasis grafis yang profesional*. Yogyakarta: Andi Offset.

Shiffman, D. (2008). *Learning Processing A Beginner's Guide to Programming Images*, Animation, and Interaction. Morgan Kaufmann Publishers.

http://processingjs.org

http://blogs.itb.ac.id/wnugroho/processing/

http://en.wikipedia.org/wiki/Cohen–Sutherland_algorithm

http://en.wikipedia.org/wiki/Line_clipping#Fast_clipping