Algoritma Kliping Garis yang Efisien untuk Ruang 3D

Artikel · Mei 2012

KUTIPAN
7

BACA
795

3 penulis, termasuk:

Saluka Ranasinghe Kodituwakku
Universitas Peradeniya
78 PUBLIKASI **498** KUTIPAN

LIHAT PROFIL

PETA Chamikara
Universitas RMIT
58 PUBLIKASI **243** KUTIPAN

LIHAT PROFIL

Volume 2, Edisi 5, Mei 2012 ISSN: 2277 128X

Jurnal Internasional Penelitian Lanjutan dalam Ilmu Komputer dan Rekayasa Perangkat Lunak

Makalah Penelitian

Tersedia secara online di: www.ijarcsse.com

Algoritma Kliping Garis yang Efisien untuk Ruang 3D

R. Kodituwakku
dept. Statistik & Ilmu Komputer
Fakultas Sains
Universitas Peradeniya

KR Wijeweera
dept. Statistik & Ilmu Komputer
Fakultas Sains
Universitas Peradeniya

PETA Chamikara
Institut Sains Pascasarjana
Fakultas Sains
Universitas Peradeniya
pathumchamikara@gmail.com

Abstrak— **Makalah ini mengusulkan algoritma kliping baris baru untuk ruang 3D terhadap balok yang tidak dihasilkan berdasarkan algoritma kliping garis CohenSutherland atau Liang-Barsky. Algoritma yang diusulkan didasarkan pada teori sederhana yang baru diusulkan yang dikembangkan menggunakan konsep matematika dasar. Semua hampir semua algoritma kliping garis 3D melibatkan tiga langkah untuk memeriksa apakah segmen garis terletak sepenuhnya di dalam volume kliping atau terletak sepenuhnya di luar volume kliping atau perhitungan persimpangan ketika tidak sepenuhnya di dalam atau di luar. Algoritma yang diusulkan tidak mengikuti langkah-langkah ini. Algoritme diuji untuk sejumlah besar segmen garis acak dan hasilnya menunjukkan bahwa algoritme pemotongan garis ruang 3D yang baru berkinerja lebih baik daripada algoritme pemotongan garis 3D CohenSutherland dalam hal waktu dan ruang.**

Kata kunci— **Grafik Komputer, Kliping Garis, geometri 2D, geometri 3D.**

SAYA. **sayaPENDAHULUAN**

Dalam grafik komputer, sangat penting untuk memotong area atau volume yang akan ditampilkan pada monitor komputer. Daerah yang diinginkan ini biasanya persegi panjang atau poligon umum dalam dua dimensi dan dikenal sebagai jendela kliping [2]. Ketika datang ke kliping tiga dimensi, volume digunakan untuk mengekstrak bagian dari adegan tiga dimensi. Umumnya garis terpotong oleh volume kliping ini dan merupakan polihedron. Kubus banyak digunakan sebagai volume kliping. Kliping tiga dimensi adalah salah satu proses paling penting dalam aplikasi medis, video game, desain berbantuan komputer, dan banyak aplikasi lainnya. Kadang-kadang perlu untuk membuang data yang tidak terdapat di wilayah yang terlihat untuk menghindari meluapnya register internal perangkat tampilan [10]. Selanjutnya, konsumsi memori yang lebih rendah dapat dicapai dari memuat hanya bagian tertentu dari suatu adegan ke memori dengan memotong bagian yang tidak perlu [9]. Oleh karena itu, meningkatkan efisiensi algoritma kliping memiliki dampak besar pada efisiensi sistem grafis secara keseluruhan.

Algoritma kliping garis Cohen-Sutherland [1], algoritma kliping garis LiangBarsky [2], algoritma kliping garis Cyrus-Beck [3] dan algoritma kliping garis Nicholl-Lee-Nicholl [4] adalah beberapa dari algoritma kliping garis tradisional. Algoritma Cohen-Sutherland dan Liang-Barsky dapat diperluas ke kliping tiga dimensi. Algoritma Nicholl-Lee-Nicholl melakukan perbandingan dan pembagian yang lebih sedikit sehingga lebih cepat dari yang lain [1]. Namun, sulit untuk memperluas kliping tiga dimensi.

Algoritma Cohen-Sutherland adalah salah satu algoritma kliping yang paling sederhana dan paling banyak digunakan dalam grafik komputer. Algoritma ini bekerja sangat cepat dalam situasi seperti segmen garis sepenuhnya di dalam atau di luar jendela kliping. Ketika segmen garis tidak sepenuhnya di dalam atau di luar, algoritma menjadi tidak efisien karena perhitungan berulang [1]. Algoritma ini dapat dengan mudah diperluas ke kliping tiga dimensi, tetapi ruang diperlukan untuk membagi menjadi 27 volume yang saling eksklusif, ketika volume kliping adalah kubus atau balok. Juga setiap volume diberi kode wilayah [11]. Sepanjang proses kliping kode wilayah tersebut harus disimpan dalam memori. Jadi dalam hal kompleksitas ruang dan waktu algoritma Cohen – Sutherland tidak efisien untuk masalah yang kompleks. Menurut Hearn dan Baker [1] hampir semua algoritma kliping garis 3D melibatkan tiga langkah:

1. Untuk segmen garis tertentu, periksa apakah segmen tersebut terletak sepenuhnya di dalam volume kliping.
2. Jika tidak, periksa apakah terletak sepenuhnya di luar volume kliping.
3. Jika tidak, lakukan perhitungan persimpangan dengan satu atau lebih bidang kliping.

Ketiga langkah ini mengarahkan algoritma untuk melakukan banyak perhitungan. Dalam algoritma kliping garis ruang 3D yang diusulkan, prosedur tiga langkah tradisional belum digunakan. Algoritma kliping garis ruang 2D baru yang efisien diperkenalkan pada tahap awal penelitian ini [7]. Kode Pseudo dari algoritma kliping garis ruang 2D adalah sebagai berikut.

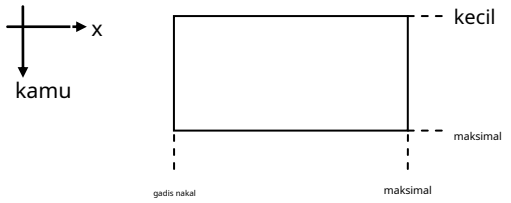
Mulai
//Menghitungm dan c
m = (y[1]-y[0])/(x[1]-x[0]); //Gradien segmen garis
c= (x[0]*y[1]-x[1]*y[0])/(x[0]-x[1]);//Y-intercept segmen garis
Untuk saya=0 **untuk** i=1 //Untuk setiap titik akhir ruas garis

Jika x[i] < minx //Titik akhir berada di sisi -ve dari garis x=minx
//Menghitung titik potong dengan garis x=minx
x[i] = minx;
y[i] = m*minx + c;
LainJika x[i] > maxx //Titik akhir berada di sisi +ve garis x=maxx
//Menghitung titik potong dengan garis x=maxx
x[i] = maxx;
y[i] = m*maxx + c;
Berakhir jika

Jika y[i] < miny //Titik akhir ada di sisi -ve dari y=garis miny
//Menghitung titik potong dengan y=garis kecil
x[i] = (miny-c)/m;
y[i] = kecil;
LainJika y[i] > maxy //Titik akhir berada di sisi +ve garis y=maxy
//Menghitung titik potong dengan y=garis maksi
x[i] = (maks-c)/m;
y[i] = maksimal;
Berakhir jika
AkhirUntuk

// Baris awal benar-benar di luar
Jika (x[0]-x[1]<1) **DAN** (x[1]-x[0]<1) **Kemudian //**koordinat x sama
//Tidak melakukan apapun
Lain //koordinat x tidak sama
//Simpan garis dengan titik akhir (x[0],y[0]),(x[1],y[1])
Berakhir jika
Akhir

Dimana, titik akhir segmen garis adalah A=(x[0], y[0]) dan B=(x[1], y[1]), dan konvensi digambarkan pada Gambar 1 telah digunakan untuk memberi label kliping persegi panjang jendela.

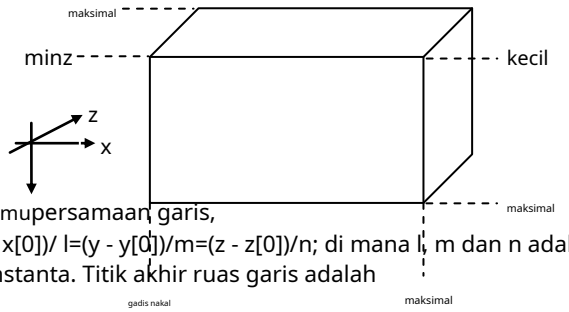


Gambar.1. 2DCliping jendela

Dalam makalah ini, perpanjangan tiga dimensi dari algoritma tersebut diusulkan. Ini adalah perpanjangan dari algoritma kliping 2D di atas.

II. **sayaETODOLOGI**

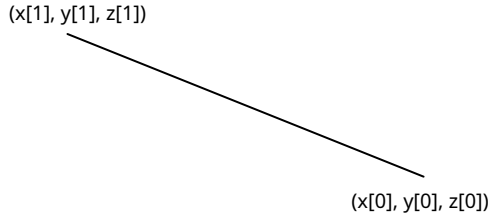
Bagian ini menyajikan algoritma kliping baris yang diusulkan dan menganalisis kinerjanya. Untuk kliping garis, volume kliping berbentuk kubus dipertimbangkan. Gambar 2 menggambarkan konvensi yang telah digunakan untuk memberi label volume.



(x - x[0])/ l=(y - y[0])/m=(z - z[0])/n; di mana l, m dan n adalah konstanta. Titik akhir ruas garis adalah A=(x[0], y[0], z[0]) dan B=(x[1], y[1], z[1]).

Gbr.2. Volume kliping

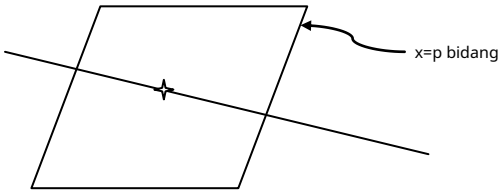
1) Latar belakang matematika dari algoritma yang diusulkan



Gbr.3. Garis lurus

Persamaan garis yang ditunjukkan pada Gambar 3 adalah
(x - x[0])/ l=(y - y[0])/m=(z - z[0])/n; di mana l, m dan n adalah konstanta.
Dengan mensubstitusi, (x[1], y[1], z[1]), persamaan menjadi (x[1] - x[0])/ l=(y[1] - y[0])/m =(z[1] - z[0])/n; Dari (x[1] - x[0])/l = (y[1] - y[0])/m
=> (x[1] - x[0])/(y[1] - y[0]) = l/m = a;
Dari (y[1] - y[0])/m = (z[1] - z[0])/n =>
(y[1] - y[0])/(z[1] - z[0]) = m/n = b; Oleh karena itu, ab = (l/m)(m/n) = l/n;

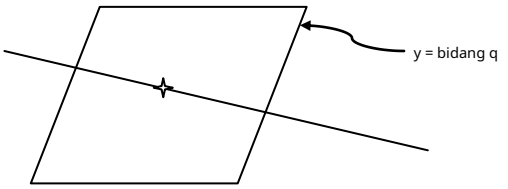
Sekarang perhatikan perpotongan garis dan bidang x = p yang digambarkan pada Gambar 4.



Gbr.4. Garis lurus berpotongan dengan bidang x=p

Pertimbangkan, (x - x[0])/l = (y - y[0])/m = (z - z[0])/n;
Dari (y - y[0])/m = (x - x[0])/l => (y - y[0])/m = (p - x[0])/l => y = (p - x[0])(m/l) + y[0];
Dari (x - x[0])/ l = (z - z[0])/n => (z - z[0])/n = (p - x[0])/l => z = (p - x[0])(n/l) + z[0];
Jadi, titik potongnya adalah
{p, (p - x[0])/a + y[0], (p - x[0])/(ab) + z[0]}

Demikian pula pertimbangkan perpotongan garis dan bidang y = q yang ditunjukkan pada Gambar 5.

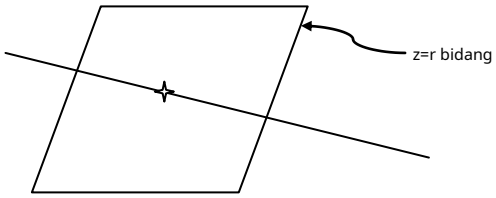


Gbr.5. Garis lurus berpotongan dengan bidang y=q

Pertimbangkan, (xx[0])/l=(yy[0])/m=(zz[0])/n;
Dari (x - x[0])/l = (y - y[0])/m => (x - x[0])/l = (q - y[0])/m => x = (q - y[0])(l/m) + x[0];

Dari (z - z[0])/n = (y - y[0])/m => (z - z[0])/n = (q - y[0])/m => z = (q - y[0])(n/m) + z[0];
Oleh karena itu, titik potongnya adalah
{a(q - y[0]) + x[0], q, (q - y[0])/b + z[0]}

Akhirnya, pertimbangkan perpotongan garis dan bidang z = r yang ditunjukkan pada Gambar 6.



Gbr.6. Garis lurus berpotongan dengan bidang x=p

Pertimbangkan (x - x[0])/ l = (y - y[0])/m = (z - z[0])/n;
Dari (x - x[0])/l = (z - z[0])/n => (x - x[0])/ l = (r - z[0])/n => x = (r - z[0])(l/n) + x[0];
Dari (y - y[0])/m = (z - z[0])/n => (y - y[0])/m = (r - z[0])/n => y = (r - z[0])(m/n) + y[0];
Oleh karena itu, titik potongnya adalah
{ab(r - z[0]) + x[0], b(r - z[0]) + y[0], r}

2) Kode semu dari algoritma yang diusulkan

Semua simbol yang digunakan dalam kode semu berikut ditunjukkan pada Gambar 2 atau disediakan di bagian 2.1.

Mulai
//Menghitung a dan b
a = (x[1] - x[0])/(y[1] - y[0]); b = (y[1] - y[0])/(z[1] - z[0]);
Untuk saya=0 **untuk** i=1 //Untuk setiap titik akhir ruas garis
Jika x[i] < minx**Kemudian //**Titik akhir berada di sisi -ve bidang x=minx
//Menghitung titik potong dengan bidang x=minx
y[i] = (minx - x[0])/a + y[0];
z[i] = (minx - x[0])/(a*b) + z[0];
x[i] = minx;
LainJika x[i] > maksx **Kemudian//**Titik akhir berada di sisi +ve bidang x=maxx
//Menghitung titik potong dengan bidang x=maxx
y[i] = (maxx - x[0])/a + y[0];
z[i] = (maks - x[0])/(a*b) + z[0];
x[i] = maks;
Berakhir jika
Jika y[i] <miny **Kemudian //**Titik akhir berada di sisi -ve dari y=bidang kecil
//Menghitung titik potong dengan y=bidang kecil
x[i] = a*(miny - y[0]) + x[0];
z[i] = (miny - y[0])/b + z[0];
y[i] = kecil;
LainJika y[i] > maxy **Kemudian //**Titik akhir berada di sisi +ve y=bidang maksi
//Menghitung titik potong dengan y=maxy plane
x[i] = a*(maxy - y[0]) + x[0];
z[i] = (maks - y[0])/b + z[0]; y[i] = maksimal;
Berakhir jika
Jika z[i] < minz **Kemudian //** Titik akhir berada di sisi -ve bidang z=minz
//Menghitung titik potong dengan bidang z=minz
x[i] = a*b*(minz - z[0]) + x[0];
y[i] = b*(minz - z[0]) + y[0];
z[i] = minz;
LainJika z[i] > maksz **Kemudian//**Titik akhir berada di sisi +ve bidang z=maxz
//Menghitung titik potong dengan bidang z=maxz
x[i] = a*b*(maxz - z[0]) + x[0];
y[i] = b*(maksz - z[0]) + y[0];
z[i] = maks;
Berakhir jika
AkhirUntuk

// Baris awal benar-benar di luar
Jika (x[0] - x[1]) < 1 **DAN (**x[1] - x[0]) < 1 **Kemudian //**koordinat x sama
// Tidak melakukan apapun
Lain //x-koordinat tidak sama
// Simpan garis dengan titik akhir (x[0], y[0], z[0]), (x[1], y[1], z[1])
Berakhir jika
Akhir

3) Implementasi algoritma yang diusulkan

Algoritma yang diusulkan telah dikembangkan menggunakan bahasa pemrograman C++. Untuk meningkatkan kemampuan pemahaman algoritma, kode sumber disajikan di bawah ini.

```
void clipMY3D(double x[],double y[],double z[],double minx,double
miny,double minz,double maxx,double maxy,double maxz) {

di aku;
ganda a,b; // Dua konstanta tergantung pada garis

if((x[0]!=x[1]) && (y[0]!=y[1]) && (z[0]!=z[1])) // Garis tidak sejajar
dengan xy, bidang yz dan zx
{
a=(x[1]-x[0])/(y[1]-y[0]);
b=(y[1]-y[0])/(z[1]-z[0]);

untuk(i=0; i<2; i++)
{
        jika(x[i] < minx)
        {
                y[i] = (minx - x[0])/a + y[0]; z[i] =
                (minx - x[0])/(a*b) + z[0]; x[i] =
                minx;
        }
        else if(x[i] > maxx)
        {
                y[i] = (maks - x[0])/a + y[0]; z[i] =
                (maks - x[0])/(a*b) + z[0]; x[i] =
                maks;
        }
        jika(y[i] < miny)
        {
                x[i] = a*(miny - y[0]) + x[0];
                z[i] = (miny - y[0])/b + z[0];
                y[i] = kecil;
        }
        else if(y[i] > maxy) {

                x[i] = a*(maks - y[0]) + x[0]; z[i]
                = (maks - y[0])/b + z[0]; y[i] =
                maksimal;
        }
        jika(z[i] < menit)
        {
                x[i] = a*b*(minz - z[0]) + x[0];
```

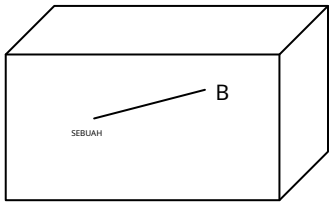
```
                y[i] = b*(minz - z[0]) + y[0];
                z[i] = minz;
        }
        else if(z[i] > maxz)
        {
                x[i] = a*b*(maksz - z[0]) + x[0];
                y[i] = b*(maksz - z[0]) + y[0]; z[i]
                = maks;
        }
}

if((x[0] - x[1] < 1) && (x[1] - x[0] < 1)) // Baris awal sudah lengkap
di luar
{
        // Tidak melakukan apapun
}
lain
{
        cout<< x[0] << "," << y[0] << "," << z[0] << " ; "
        << x[1] << "," << y[1 ] << "," << z[1] << endl;
}

}
else if(z[0] == z[1]) // Garis sejajar dengan bidang xy
{
        clipMY1(x, y, z, minx, miny, maxx, maxy);
}
else if(x[0] != x[1]) // Garis sejajar dengan bidang xz
{
        clipMY2(x, y, z, minz, minx, maksz, maksx);
}
else if(y[0] != y[1]) //Garis sejajar dengan bidang yz
{
        clipMY3(x, y, z, miny, minz, maxy, maxz);
}
else // Garis sejajar dengan sumbu z
{
        // baris awal benar-benar di luar
        if((x[0] <= minx) || (x[0] >= maxx) || (y[0] <= miny) ||
(y[0] >= maxy))
        {
                // tidak melakukan apapun
        }
        lain
        {
                untuk(i=0; i<2; i++)
                {
                        jika(z[i] < menit)
                        {
                                z[i] = minz;
                        }
                        else if(z[i] > maxz)
                        {
                                z[i] = maks;
                        }
                }
                // baris awal benar-benar di luar
                if((z[0] - z[1] < 1) && (z[1] - z[0] < 1)) {

                        // tidak melakukan apapun
                }
                // gambar garis terpotong
                lain
                {
                        Cout << x[0] << "," << y[0] << "," << z[0] << " ; " <<
                        x[1] << "," << y[1 ] << "," << z[1] << endl;
                                }
                        }
}

}
```

CATATAN: clipMY1, clipMY2, clipMY3 adalah fungsi yang telah dirancang untuk memotong garis dalam ruang 2D7. Setelah segmen garis sejajar dengan salah satu bidang utama, ini dapat dianggap sebagai masalah kliping 2D.

4) Analisis algoritma

Bagian ini menganalisis algoritma untuk beberapa kemungkinan situasi.

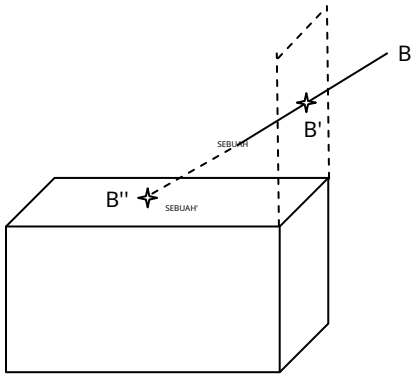
Kasus 1: Garis benar-benar di dalam sebagai ditunjukkan pada Gambar 7



Gbr.7.Line benar-benar di dalam

x[A] != x[B] benar
y[A] != y[B] benar
z[A] != z[B] benar
Perhatikan titik A,
x[A] < minx salah
x[A] > maksx salah
y[A] < miny salah
y[A] > maks salah
z[A] < minz salah
z[A] > maksz Salah
Oleh karena itu, posisi awal A tidak berubah.
Perhatikan titik B,
x[B] < minx salah
x[B] > maksx salah
y[B] < miny salah
y[B] > maks salah
z[B] < minz salah
z[B] > maksz Salah
Oleh karena itu, posisi awal B tidak berubah. (x[A] - x[B] < 1) && (x[B] - x[A]) Salah
Oleh karena itu, garis dengan titik akhir A dan B ditarik.

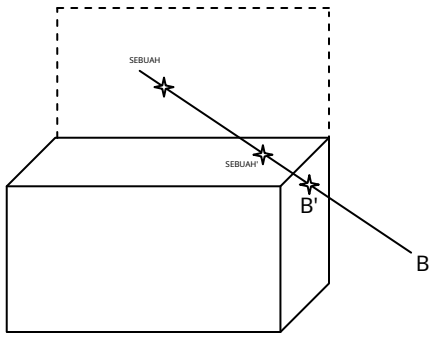
Kasus 2: Garis benar-benar di luar seperti yang digambarkan pada Gambar 8



Gbr.8. Jalur benar-benar di luar

x[A] != x[B] benar
y[A] != y[B] benar
z[A] != z[B] benar
Perhatikan titik A,
x[A] < minx salah
x[A] > maksx salah
y[A] < miny benar
Jadi, A SEBUAH'
z[A'] < minz false z[A']
> maksz salah
Pertimbangkan titik B,
x[B] < minx salah
x[B] > maksx benar
Jadi, B B'
y[B'] < miny benar
Jadi, B' B''
z[B''] < minz false
z[B''] > maksz Salah
(x[A'] - x[B''] <1) && (x[B''] - x[A']) benar
Oleh karena itu, garis diabaikan.

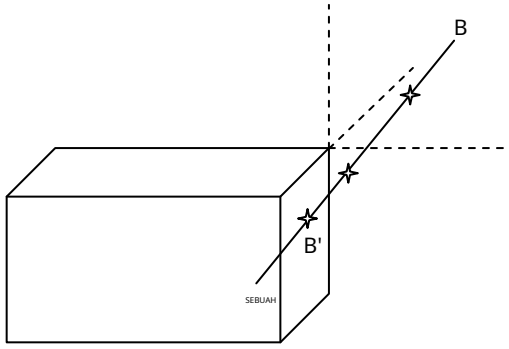
Kasus 3: Garis memotong kliping jendela seperti yang ditunjukkan pada Gambar 9



Gbr.9. Garis memotong batas

x[A] != x[B] benar
y[A] != y[B] benar
z[A] != z[B] benar
Perhatikan titik A,
x[A] < minx salah
x[A] > maksx salah
y[A] < miny benar
Jadi, A SEBUAH'
z[A'] < minz false z[A']
> maksz salah
Pertimbangkan titik B,
x[B] < minx salah
x[B] > maksx benar
Jadi, B B'
y[B'] < miny salah
y[B'] > maks salah
z[B'] < minz false
z[B'] > maksz Salah
(x[A'] - x[B'] < 1) && (x[B'] - x[A']) false Oleh karena itu, garis dengan titik akhir A' dan B' ditarik.

Kasus 4: Garis sebagian di dalam jendela kliping seperti yang digambarkan pada Gambar 10



Gambar 10. Garis sebagian berada di dalam jendela kliping

x[A] != x[B] benar
y[A] != y[B] benar
z[A] != z[B] benar
Perhatikan titik A,
x[A] < minx salah
x[A] > maksx salah
y[A] < miny salah
y[A] > maks salah
y[A] < minz salah
y[A] < maksz Salah
Oleh karena itu, posisi awal A tidak berubah.
Perhatikan titik B,
x[B] < minx salah
x[B] > maksx benar
Oleh karena itu, B B'
y[B'] < miny salah
y[B'] < maxy salah
z[B'] < minz salah
z[B'] < maksz Salah
(x[A] - x[B'] < 1) && (x[B'] - x[A]) Salah
Oleh karena itu, ditarik garis dengan titik akhir A dan B'.

AKU AKU AKU. **RHASIL DAN DPEMBAHASAN**

Algoritma yang diusulkan diuji untuk semua kemungkinan kasus uji segmen garis. Hasil pengujian menunjukkan bahwa ia berkinerja baik dalam semua situasi yang memungkinkan. Untuk memvalidasi algoritma, itu dibandingkan dengan algoritma kliping garis ruang 3D Cohen-Sutherland. Perangkat keras dan perangkat lunak berikut digunakan untuk pengujian.

Komputer: Intel(R) Pentium(R) Dual CPU; E2180 @ 2,00 GHz; 2,00 GHz, 0,98 GB RAM IDE Detail: Turbo C++; Versi 3.0; Hak Cipta (c) 1990, 1992 oleh Borland International, Inc. Metode [2]:

Volume kliping (kubus) dengan nilai minx = miny = minz = 100 dan maxx = maxy = maxz = 300 digunakan untuk kliping. Titik acak dihasilkan dalam rentang 0 - 399 dengan menggunakan fungsi randomize(). Titik acak ini dianggap sebagai titik akhir untuk menghasilkan garis acak. Jumlah siklus clock yang diambil oleh setiap algoritma untuk memotong 1000000 garis acak dihitung menggunakan fungsi clock(). Hasilnya ditunjukkan pada Tabel 1.

TABEL I
tidakUMBEROF CMENGUNCI CYCLES COMPARISON

Langkah	Cohen-Sutherland algoritma	yang diusulkan algoritma
1	3596	3403
2	3497	3271
3	3946	3431
4	3867	3980
5	3489	3097
6	4014	3790
7	3906	3767
8	3638	3737
9	4018	3914
10	3839	3684

Hasilnya membuktikan bahwa 3D baru kliping garis ruang algoritma lebih cepat dalam 8 langkah dari 10 langkah daripada algoritma CohenSutherland. Performa dalam algoritme kami buruk ketika segmen garis benar-benar keluar dari ide karena sejumlah perhitungan persimpangan diperlukan untuk menghapus garis tersebut yang sepenuhnya berada di luar jendela kliping. Oleh karena itu, dalam situasi di mana banyak garis yang dihasilkan secara acak sepenuhnya berada di luar jendela kliping, kinerja algoritma yang baru diusulkan sedikit lebih rendah daripada algoritma Cohen-Sutherland.

IV. CKESIMPULAN

Menurut hasil pengujian, algoritma kliping garis ruang 3D yang diusulkan lebih cepat daripada algoritma 3D Cohen–Sutherland. Oleh karena itu, algoritma ini dapat berhasil digunakan dalam aplikasi 3D di mana kliping garis terlibat. Algoritma ini selanjutnya dapat diperluas ke garis klip dalam volume polihedron.

REFERENSI

[1] D. Hearn dan MP Baker, Computer Graphics,|| C Version, 2nd Edition, Prentice Hall, Inc., Upper Saddle River, 1998, hlm. 224-248.

[2] Wenjun Huang, Algoritma Kliping Garis Berdasarkan Transformasi Affine, Manajemen Informasi Cerdas, 2010, 2.380-385, Diterbitkan Online Juni 2010 (http:// www.SciRP.org/journal/iim)

[3] M. Cyrus dan J. Beck, Umumnya Kliping Dua dan Tiga Dimensi,|| Komputer dan Grafik, Vol. 3, No. 1, 1978, hlm. 23-28.

[4] TM Nicholl, DT Lee dan RA Nicholl, Algoritma Baru yang Efisien untuk Kliping Garis 2D: Pengembangan dan Analisisnya,|| Komputer dan Grafik, Vol. 21, No. 4, 1987, hlm. 253-262.

[5] CB Chen and F. Lu, Computer Graphics Basis,|| Publishing House of Electronics Industry, Beijing, 2006, hlm.167-168.

[6] V. Skala, O (lg N) Algoritma Line clipping in E , Komputer dan Grafik, Vol. 18, No. 4, 1994, hlm. 517-527.

[7] SR Kodituwakku, KR Wijeweera, PETA Chamikara. Algoritme yang efisien untuk kliping garis di komputer pemrograman grafis; akan muncul di Ceylon Journal of Science (Ilmu Fisika), 2012.

[8] You-Dong Liang, Brian A. Barsky, Mel Slater. Beberapa Perbaikan pada Algoritma Kliping Garis Parametrik. hal.2

[9] Patrick-Gilles Maillot. Model Clipping Triangle St rips dan Quad Meshes, Sun Microsystems, Inc.

[10] Fuhua Cheng, Yue-Kwo Yen. Algoritma Kliping Garis Paralel dan Implementasinya.

[11] JD Foley dan A. van Dam, Fundamentals of Interactive Computer Graphics, Addison Wesley, Reading, Mass, 1982. hal. 227.

SR Kodituwakku adalah profesor di Departemen Statistik dan Ilmu Komputer, Universitas Peradeniya, Sri Lanka. Minat penelitiannya meliputi sistem basis data, komputasi terdistribusi, sistem Kontrol Akses berbasis Peran, dan perangkat lunak teknik.

KR Wijeweera adalah seorang sarjana, mengikuti gelar khusus ilmu komputer di Universitas Peradeniya, Sri Lanka. Minat penelitiannya meliputi grafik komputer, pemrosesan gambar, visi komputer, dan karya seni intelijen.

MA Pathum Chamikara bekerja sebagai asisten peneliti di Post Graduate Institute of Science (PGIS), University of Peradeniya, Sri Lanka. Ia menerima gelar BSc (Khusus) di bidang Ilmu Komputer, University of Peradeniya, Sri Lanka (2010). Minat penelitiannya meliputi analisis Kejahatan, GIS (Sistem Informasi Geografis), pemrosesan gambar, visi komputer, dan kecerdasan buatan.