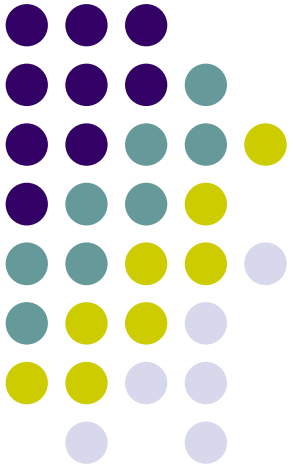
Ray Tracing

S1 Teknik Informatika



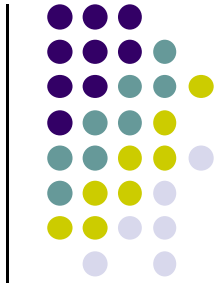
Definisi

- *Ray tracing* adalah salah satu dari banyak teknik yang ada untuk membuat gambar dengan komputer. Ide dibalik *ray tracing* adalah bahwa gambar yang benar secara fisik tersusun oleh cahaya dan cahaya biasanya akan berasal dari sumber cahaya dan pantulan sekitar sebagai sinar cahaya (mengikuti jalur garis terputus) dalam adegan sebelum tertangkap mata kita atau kamera.
- Kemampuan mereproduksi dalam simulasi komputer jalan yang diikuti dari sumber cahaya ke mata kita, kita kemudian akan dapat menentukan apa yang mata kita lihat.

Definisi

- Gagasan kedua adalah bahwa gambar yang kita hasilkan biasanya berupa grid pixel dengan resolusi terbatas.
- Ray Tracing adalah metode untuk menghitung jalan gelombang atau partikel melalui suatu sistem.
- Ray Tracing atau yang dikenal dengan *Ray Casting, menjelaskan hal yang terlihat dari* permukaan dengan mengikuti gambaran cahaya dari sinar yang berasal dari penglihatan mata kita terhadap objek di layar.

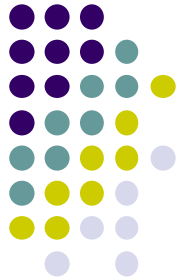
Definisi



- Ray Tracing adalah teknik *rendering grafik tiga dimensi dengan* interaksi sinar yang kompleks.
- Ray tracing dilakukan dalam dua bentuk yang berbeda :

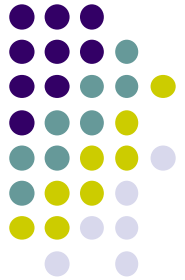
1. Ray Tracing (physics), yang digunakan untuk menganalisis sistem optik.
2. Ray Tracing (graphics), yang digunakan untuk generasi gambar 3D.

Ray Tracing (physics)



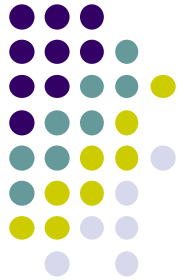
- Dalam fisika, *ray tracing adalah metode untuk menghitung jalan gelombang atau partikel* melalui suatu sistem dengan berbagai propagasi daerah kecepatan, penyerapan karakteristik, dan mencerminkan permukaan. Dalam keadaan ini, permukaan gelombang dapat menekuk, mengubah arah, atau mencerminkan permukaan, dengan analisis yang rumit.

Ray Tracing (physics)



- Ray tracing memecahkan masalah dengan mempercepat idealisasi berkas sempit secara berulang-ulang yang disebut dengan *ray yang melalui suatu medium dengan* sejumlah diskrit.
- Masalah sederhana dapat dianalisis dengan menyebarkan beberapa sinar dengan menggunakan matematika sederhana. Analisis yang lebih detailnya dapat dilakukan dengan menggunakan komputer untuk menyebarkan banyak sinar.

Ray Tracing (physics)



- Ray tracing bekerja dengan mengasumsikan bahwa partikel atau gelombang dapat dimodelkan sebagai sejumlah besar berkas sinar yang sangat sempit, dan bahwa ada beberapa sinar yang melewati batas jarak seperti sinar yang bertempat datar.
- Sinar pelacak akan mempercepat sinar yang melewati jarak ini, dan kemudian menggunakan daerah turunan dari medium untuk menghitung arah sinar baru.

Ray Tracing (physics)

- Dari lokasi ini, sinar yang baru akan dikirim keluar dan proses akan diulang sampai jalan yang lengkap dihasilkan. Jika simulasinya mencakup benda padat, sinar dapat diuji pada persimpangan dengan setiap langkahnya, melakukan penyesuaian pada arah sinar jika ditemukan adanya suatu tabrakan.

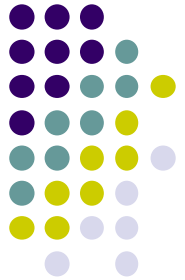
Ray Tracing (physics)

- Properti lain dari sinar dapat diubah sebagai pencepatan simulasi juga., seperti intensitas, panjang gelombang, atau polarisasi.
- Contoh kegunaan Ray Tracing (physics) ada pada sinyal radio, samudra akustik, dan desain optis.

Ray Tracing (graphics)

- Dalam grafik komputer, *ray tracing adalah teknik untuk menghasilkan sebuah gambar* dengan menelusuri jalan cahaya melalui pixel dalam gambar. Teknik ini mampu menghasilkan tingkat ketajaman gambar yang sangat tinggi – biasanya lebih tinggi dari pada metode tipe *scanline rendering, tetapi biaya komputasi lebih besar.*

Ray Tracing (graphics)



- Hal ini membuat *ray tracing paling cocok untuk aplikasi di mana gambar dapat di-render perlahan* terlebih dahulu, seperti pada gambar diam dan film dan *special effects televisi, dan kurang* lebih cocok untuk real-time aplikasi seperti *game komputer, dimana kecepatan sangat* penting.
- Ray tracing mampu mensimulasikan berbagai efek optis, seperti refleksi dan pembiasan penyebaran, dan aberasi kromatik.

Ray Tracing (graphics)

Ray Tracing (graphics)

- Ray tracing telah digunakan dalam lingkungan produksi untuk off-line rendering selama beberapa dekade sekarang – yaitu rendering yang tidak perlu menyelesaikan seluruh adegan dalam waktu kurang dari beberapa milidetik. Tentu saja kita tidak boleh men-generalisasi dan membiarkan pengguna mengetahui bahwa beberapa implementasi raytracer telah mampu menekan tanda “interaktif”.

Ray Tracing (graphics)

- Terdapat 2 metode pada Ray Tracing yaitu:

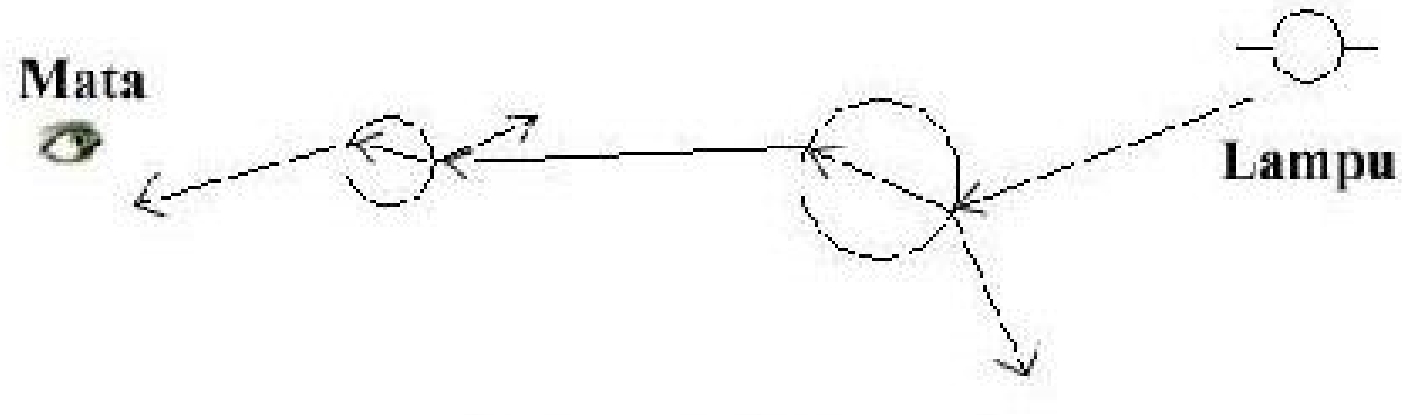
1. Forward Ray Tracing. Metode ini memperhitungkan semua sinar yang dipancarkan oleh sumber cahaya, baik yang mengenai mata ataupun tidak.
2. Backward Ray Tracing. Cara kerja dari metode ini adalah dengan menelusuri sinar yang mengenai mata ditelusuri kembali ke sumber cahaya.

Forward Ray Tracing

- Metode ini memperhitungkan keakuratan penghitungan warna, namun menjadi tidak efektif karena jumlah sinar yang dipancarkan oleh suatu sumber cahaya sangat banyak (bisa mencapai jutaan sinar), dan jika sinar tidak mengenai mata maka sinar tersebut akan tidak diperhitungkan meski telah dihitung sebelumnya. Hal ini akan menimbulkan banyak penghitungan sia-sia karena banyaknya sinar yang tidak diperhitungkan kemudian.

Forward Ray Tracing

- Kelebihan dari metode ini adalah dapat memperoleh jumlah sinar yang lebih banyak daripada metode backward ray tracing.

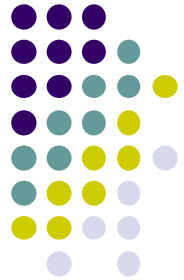


Gambar 2. Forward Ray Tracing

Forward Ray Tracing

- Pada gambar di atas tampak bahwa penelusuran sinar dilakukan mulai dari sumber cahaya (dalam hal di atas adalah lampu) menuju ke mata, sehingga semua sinar yang berasal dari lampu harus diperhitungkan. Metode penelusuran dari sumber cahaya menuju ke mata inilah yang kemudian dinamakan metode forward ray tracing. (Dari sumber diteruskan menuju ke tujuan).

Backward Ray Tracing



- Sinar yang mengenai mata tersebut akan ditelusuri menuju ke layar penggambaran dengan memperhitungkan nilai dari objek-objek yang ada pada penggambaran sehingga didapatkan apakah sinar tersebut mengenai objek yang ada. Proses penelusuran ini dilakukan untuk setiap *pixel* dari ukuran layar penggambaran. (Hal ini menyebabkan semakin besar ukuran layar penggambaran maka semakin lama proses perhitungan yang dilakukan, dan demikian pula sebaliknya)

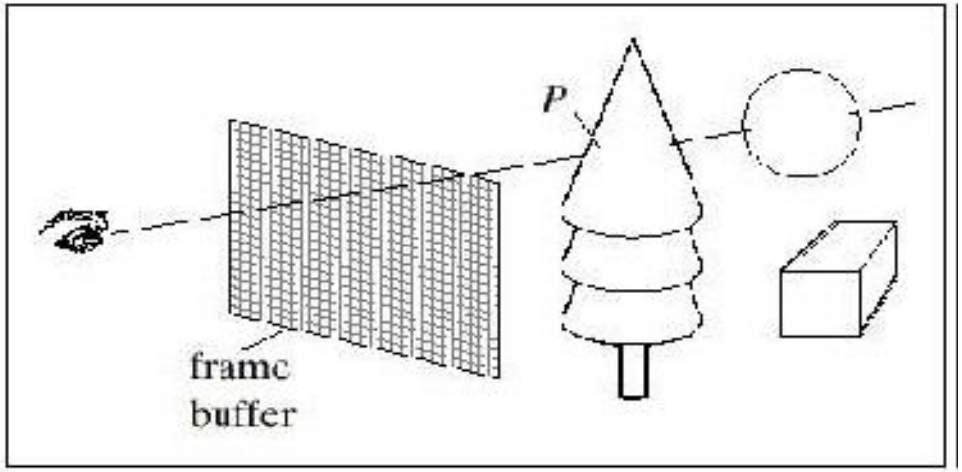
Backward Ray Tracing

- Jika sinar mengenai salah satu benda maka akan diperhitungkan warna pixel tersebut dengan memperhitungkan warna benda dan juga nilai pencahayaan yang mengenai benda tersebut.
- Jika sinar tidak mengenai benda maka nilai pixel akan diset menjadi warna background (default warna background adalah warna hitam).

Backward Ray Tracing

- Hal yang perlu diperhitungkan adalah bila sinar mengenai benda dan terdapat benda lain di belakang benda yang ditabrak maka sinar yang mengenai benda tersebut hanya diperhitungkan untuk tabrakan dengan benda pertama (benda terdepan) karena benda yang terletak di belakang benda yang lain pasti tidak akan terlihat.

Backward Ray Tracing



Gambar 3. Backward ray tracing

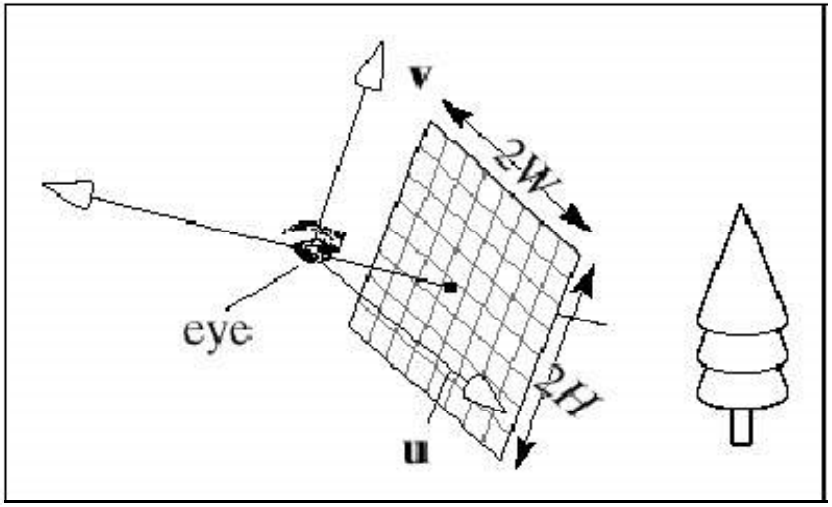
- Pada gambar di atas tampak bahwa sinar yang berasal dari sumber cahaya terus ke mata dan kemudian dari titik mata, sinar tersebut ditelusuri kembali. Dalam contoh kasus di atas, sinar yang ditelusuri kembali ternyata menabrak benda pada posisi u,v pada *frame buffer* / layar penggambaran.

Backward Ray Tracing

- Pada saat menabrak inilah maka nilai *pixel* pada *frame buffer* akan dihitung dengan memperhitungkan semua nilai *ambient / diffuse / specular* dari semua cahaya yang ada.
- Hal pertama yang harus dilakukan adalah melakukan setting / digunakan untuk penghitungan objek-objek 3 dimensi.

Backward Ray Tracing

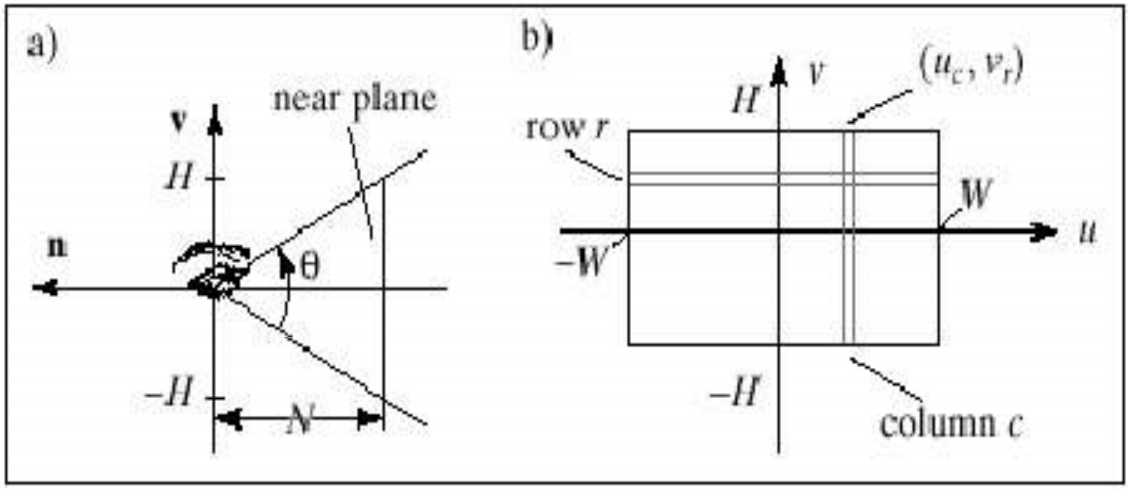
- Hal tersebut dilakukan dengan mengasumsikan bahwa layar penggambaran memiliki 2 variabel sumbu yaitu u dan v. Sumbu u adalah sumbu ke kanan dan range dari sumbu u adalah -W sampai dengan W. Sumbu v adalah sumbu ke atas dan range dari sumbu v adalah –H sampai dengan H.



Gambar 4. Penentuan sumbu u dan v

Backward Ray Tracing

- Langkah selanjutnya yang dilakukan adalah penentuan nilai dari W dan H yang sebelumnya diasumsikan sebagai range dari sumbu u dan v tersebut. Penghitungan nila W dan H tampak seperti gambar di bawah ini:



Gambar 5. Penentuan nilai W dan H dan transformasinya

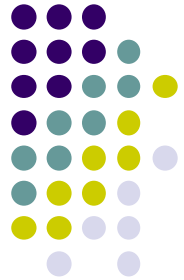
Backward Ray Tracing

- Pada gambar di atas (gambar (a)) tampak bahwa mata memiliki sudut pandang yang kita namakan sebagai θ . Sehingga untuk mendapatkan nilai tinggi dari near plane / W maka bisa didapatkan dari rumus matematika yaitu (Hill 1990);

$$H = N.\mathrm{Tan}(\theta/2) \quad \text{(Rumus 1.)}$$

- Variabel N adalah jarak antara mata dengan bidang-bidang u dan v. Sedangkan untuk penentuan nilai W, didapatkan dengan mengalikan nilai H dengan *aspect ratio* layar penggambaran / bidang u-v. (W = H. *aspect ratio*). Setelah nilai H dan W ditentukan, maka nilai posisi U_c dan V_r, yang bila diturunkan adalah sebagai berikut (Hill 1990):

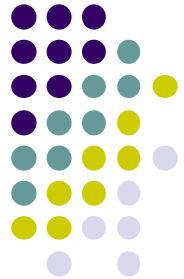
Backward Ray Tracing



$$v_r = -H + H\frac{2r}{nRows}, u_c = -W + W\frac{2c}{nCols}$$ (Rumus 2.)

- Rumus di atas digunakan untuk menentukan nilai (U_c,V_r) dalam hubungannya dengan W dan H. Hal berikutnya yang dilakukan adalah penentuan persamaan sinar ditelusuri dari mata ke pixel tujuan yang dilakukan dengan menggunakan rumus (Hill 1990):

Backward Ray Tracing



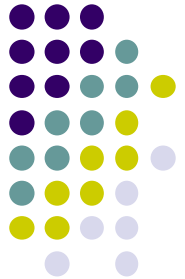
$$r(t)=eye(1-t) + (eye - N\mathbf{n} + U_c\mathbf{u} + V_r\mathbf{v})t \qquad \text{(Rumus 3.)}$$

- *Eye* adalah titik mata (dalam x,y,z), N adalah jarak antara mata dengan bilangan u-v, U_c dan V_r adalah posisi pixel pada bidang u-v dan t adalah titik tabrak sinar dengan benda (akan diperhitungkan kemudian). Rumus di atas kemudian disederhanakan menjadi (Hill 1990):

$$r(t)=eye+\mathbf{dir}_{rc}\,.t,\ \mathbf{dir}_{rc}=-N\,\mathbf{n} + U_c\mathbf{u} + V_r \qquad \text{(Rumus 4.)}$$

- Secara umum, *ray tracing* dapat dibentuk dari algoritma berikut ini:

Backward Ray Tracing



define the objects and light sources in the scene
set up the camera

```
for(int r = 0; r < nRows; r++)
  for(int c = 0; c < nCols; c++)
  {
```

1. *Build the rc-th ray*
2. *Find all intersections of the rc-th ray with objects in the scene*
3. *Identify the intersection that lies closest to, and in front of, the eye*
4. *Compute the "hit point" where the ray hits this object, and the normal vector at that point*
5. *Find the color of the light returning to the eye along the ray from the point of intersection*
6. *Place the color in the rc-th pixel.*

```
  }
```