BAB II

TINJAUAN PUSTAKA

2.1. *Lighting*

Menurut Oxford Advanced Learner's Dictionary (2010), *Lighting* adalah Penyusunan atau jenis cahaya yang ada di di suatu tempat, dalam Collins English Dictionary disebutkan bahwa *Lighting* adalah Pendistribusian cahaya pada sebuah objek atau rupa, seperti pada Foto atau lukisan. Dari beberapa pengertian ini, tentunya kita harus memulai pembahasan kita dari cahaya.

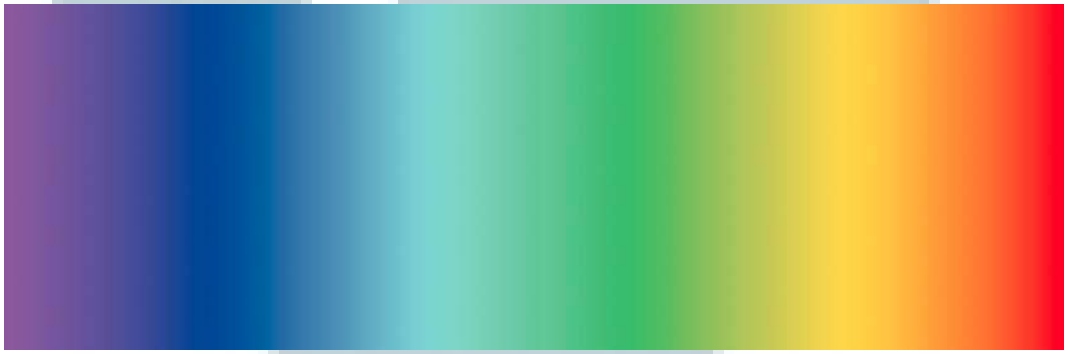
2.1.1. Definisi Cahaya

Dalam buku *3D Lighting History, Concepts, and Techniques*. Arnold Gallardo (2000) mengatakan bahwa cahaya ada dimana-mana dan mungkin adalah hal pertama yang kita alami saat lahir ke dunia ini. Keberadaan cahaya mengontrol cara kita melakukan banyak hal, kita bangun saat cahaya mulai muncul, kita bekerja di area yang berisikan cahaya. Cahaya juga diketahui sebagai bagian terlihat dari radiasi elektromagnetik yang memiliki kecepatan konstan.

Hal ini dapat lebih diperjelas dengan melihat dari Penjelasan mengenai Spektrum terlihat oleh Darren Brooker (2008) dalam buku *Essential CG Lighting Techniques with 3ds Max,*ada berbagai variasi berbeda dari gelombang yang ada di sekitar kita dalam kehidupan sehari - hari, dari *X-rays* sampai gelombang radio. perbedaan mendasar dari jenis-jenis gelombang ini adalah panjang dari gelombang tersebut. Ini semua adalah bentuk bagian dari spektrum

elektromagnetik, yang dimulai dengan panjang gelombang pendek *x-ray* di salah satu ujung sampai ke gelombang radio pada ujung lainnya, yang memiliki panjang gelombang sangat panjang. antara dua jarak ekstrim ini terletak sebuah pita yang sangat sempit yang terlihat oleh kita, dan ini adalah spektrum terlihat.

Cahaya yang terlihat oleh mata kita memiliki panjang gelombang yang lebih dekat kepada ujung dari gelombang *x-ray*, karena Panjang gelombangnya sangat kecil - dari sekitar 400 nanometer pada jumlah terkecil dan lebih kecil dari 800 nanometer pada jumlah terbanyaknya (1 nanometer adalah 1 miliar meter).

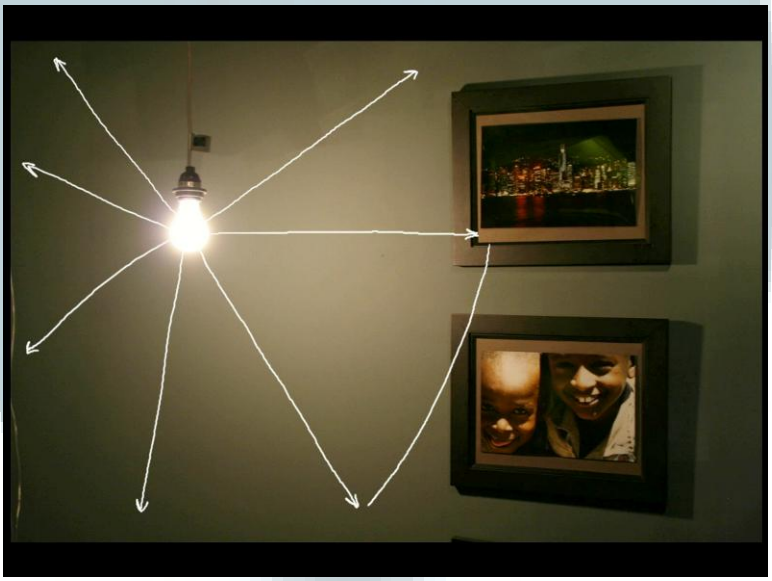


Gambar 2. 1 Distribusi cahaya pada spektrum yang dapat kita lihat
(Essential CG Lighting Techniques with 3Ds Max , Darren Brooker)

2.1.2. Perilaku Cahaya

Brooker (2008) menjelaskan bahwa Cahaya memiliki Perilaku dimana perilaku ini bertumpu pada banyak aturan, satu aturan yang sangat relevan dalam pembahasan penulis mengenai *lighting* pada *CG* adalah Hukum kuadrat terbalik, dimana hukum ini menjelaskan bagaimana cahaya semakin memudar mengikuti jarak, hal ini bisa dijelaskan lebih mudah dengan melihat perilaku suhu atau panas, jika kita berjalan perlahan menuju ke arah sumber panas misalnya api, perlahan-lahan kita

akan merasakan panas secara bertahap, ini adalah hukum kuadrat terbalik.*Luminosity* cahaya (Emisi energi cahaya perdetik) tidak berubah, yang berubah adalah terangnya cahaya yang dapat dirasakan atau dilihat oleh pelihat dari cahaya tersebut. Semakin cahaya berjalan semakin jauh dari sumbernya, cahaya mencakupi area lebih banyak dan inilah yang membuat cahaya semakin kehilangan intensitasnya.



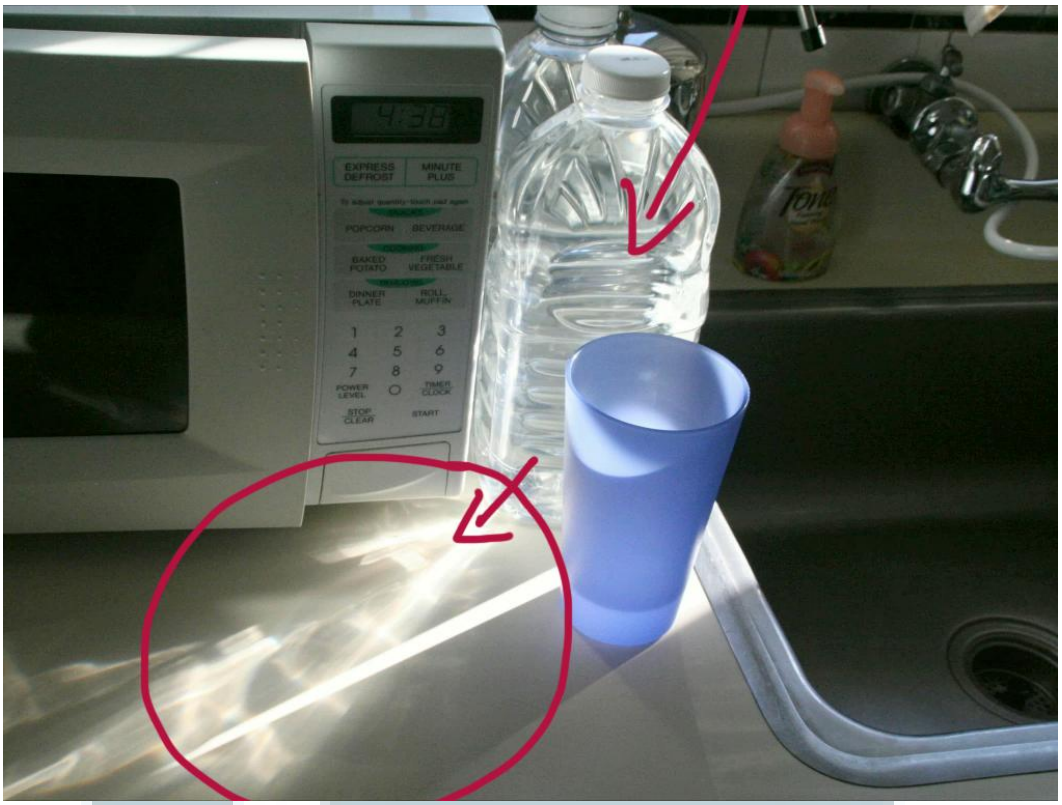
Gambar 2. 2 semakin memudar intensitasnya berdasarkan jaraknya

(The Gnomon Workshop - Practical Light and Color with Jeremy Vickery)

Cahaya juga mematuhi hukum refleksi. Hukum ini menjelaskan mengenai bagaimana cahaya dipantulkan pada permukaan. Dalam buku ini dijelaskan bahwa sudut dari refleksi cahaya bernilai sama dengan sudut dari insiden, yang diukurrelatif terhadap permukaan normal pada titik insiden.

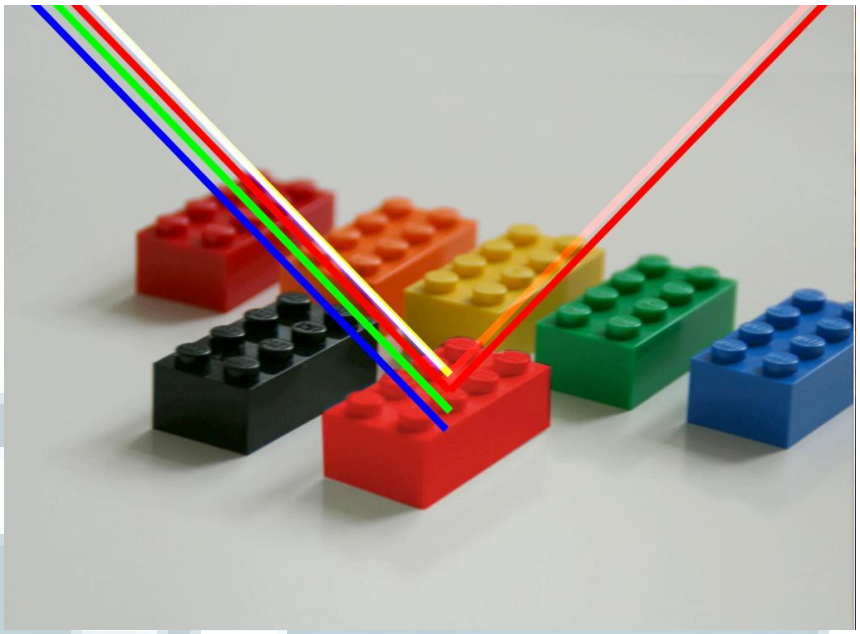
Gambar 2. 3 Cahaya yang masuk dari lubang, memantul ke tanah dan berpengaruh pada sisi dinding
(The Gnomon Workshop - Practical Light and Color with Jeremy Vickery)

Selain itu cahaya juga mengikuti hukum refraksi , refraksi dapat dideskripsikan sebagai bagaimana cara pancaran cahaya menekuk, dengan mempertimbangkan objek transparan atau semitransparan.Padadasarnya ini menentukan tingkatrefraksi ketika cahaya melewati bahan atau material yang berbeda. cahaya yang menekuk ini menyebabkan distorsi yang dapat kita lihat dengan melihatlensa.



Gambar 2. 4 Cahaya menembus botol air mengakibatkan pembengkokkan cahaya
(The Gnomon Workshop - Practical Light and Color with Jeremy Vickery)

Selain hal-hal yang disebutkan oleh Brooker pada buku, Jeremy Vickery dalam Video pembelajaran Keluaran Gnomon Workshop : *practical Light and Color*, menyebutkan bahwa terdapat satu lagi perilaku cahaya yaitu *Absorbtion* atau penyerapan cahaya, hal ini bisa kita rasakan apabila sebuah objek terkena tembakan cahaya berwarna putih (mengingat putih berisikan semua spektrum cahaya yang terlihat), dan yang terlihat oleh mata kita adalah sebuah warna tertentu misalnya merah, ini berarti spektrum cahaya selain warna merah terserap oleh objek tersebut, mengakibatkan kita hanya melihat warna merah, apabila objek tersebut nampak dimata kita berwarna hitam, berarti semua cahaya itu terserap oleh objek tersebut.



Gambar 2. 5 Spektrum cahaya selain warna merah diserap oleh objek menghasilkan warna objek merah pada penglihatan kita

(The Gnomon Workshop - Practical Light and Color with Jeremy Vickery)

2.1.3. Tujuan *Lighting*

Dalam buku *Digital Lighting and Rendering,* Jeremy Birn (2006) menjabarkan beberapa tujuan visual dari penataan *Lighting* pada produksi *CG* yaitu:

a. *"**Making Things Read**"* yang artinya membuat sesuatu menjadi lebih terbaca, seperti halnya dalam fotografi, sinematografi, dan Melukis, proses pencahayaan di *software* 3D adalah proses menghasilkan sebuah gambar dua dimensi yang menggambarkan keadaan tiga dimensi.

b. "***Making things Believable***" yang artinya membuat sesuatu menjadi nampak nyata, meskipun komputer dapat menghasilkan berbagai macam hasil akhir dengan berbagai gaya visual yang berbeda, seperti

hasil yang realistik, atau nampak seperti kartun, namun hasil dari proses *rendering* gambar haruslah berdasarkan pada dunia nyata.

c. "***Mantaining Continuity***" yang artinya menjaga kontinuitas dari hasil gambar. Bekerjapada proyek-proyek berjangka waktu lama membuat banyak kondisi dimana kita terlibat dalam pencahayaan gambar yang berbeda, menjaga kontinuitas merupakan hal terpenting. Semua hasil harus konsisten untuk mempertahankan pengalaman terbaik bagi penonton.

d. -"***Directing the Viewer's Eye***" yang artinya mengarahkan mata penonton kepada fokus utama dalam sebuah adegan animasi kita berdasarkan dengan penataan pencahayaan, dengan penataan yang baik mata penonton akan tertarik untuk selalu menuju kepada hal hal yang ingin kita sampaikan dalam adegan tersebut.

e. -"***Emotional Impact***" yang artinya Dampak emosional, setelah mata penonton tertarik kepada area yang penting dalam sebuah adegan cerita, sebagian besar penonton tidak akan pernah sadar untuk melihat penataan pencahayaan, melainkan akan merasakannya. Membantu menciptakan suasana atau nada yang meningkatkan pengalaman emosional menonton sebuah adegan adalah tujuan visual yang paling penting dari penataan pencahayaan atau *lighting*.

2.2. Teknik *Lighting*

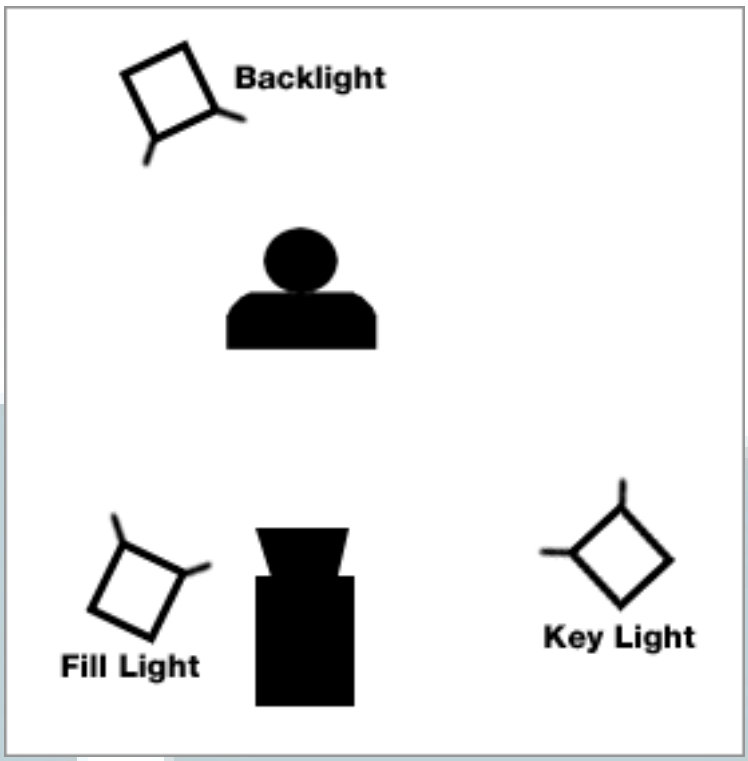
Andy beane (2012) dalam buku *3D Animation Essentials* menjabarkan beberapa teknik dasar *Lighting* yang dipakai dalam berbagai kegunaan seperti dalam fotografi, film, televisi, teater, dan lukisan yang dapat memberikan *mood* khusus sekaligus *lighting* yang baik.

1. ***Three-Point Lighting***, atau pencahayaan dengan 3 titik adalah teknik yang paling umum digunakan, teknik ini menggunakan 3 Cahaya

- ***Key light*** adalah cahaya yang memiliki intensitas tertinggi dan merupakan sumber cahaya utama. Cahaya ini dengan peraturan standar ditempatkan di 1 sisi subjek dan setidaknya sedikit lebih tinggi dari objek.

- ***Fill light*** adalah cahaya yang sedikit lebih rendah intensitasnya dari *key light*dan diletakkan pada arah yang berlawanan dari *key light* untuk mengisi bagian bayangan yang dihasilkan dari *key light*, tujuannya adalah untuk mengisi di area bayangan, bukan untuk menghilangkan bayangan.

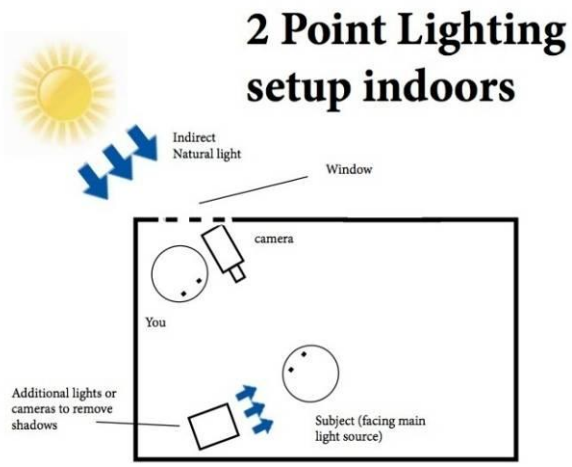
- ***Rim light*** atau ***kicker***adalah cahaya yang ditempatkan di belakang objek untuk menambahkan area *highlight* sisi pinggir objek, untuk memisahkannya dari *background*.



Gambar 2. 6 Penyusunan cahaya lampu pada *three-point lighting*

(http://2.bp.blogspot.com/-hkICw7wK4jo/TcfVfMjsLtI/AAAAAAAAARE/7fCNvUYquqc/s1600/3+point+lighting+p.gif)

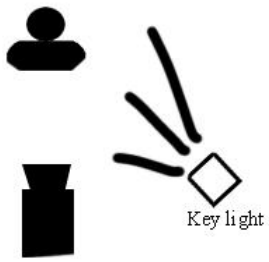
2. ***Two-Point Lighting,*** atau pencahayaan dengan 2 titik memiliiki kesamaan dengan jenis *lighting* yang kita lihat setiap hari pada saat cahaya matahari tiba dan cahaya *ambient*dari langit meliputi kita sebagai cahaya ke 2. Penyusunan *lighting* ini mirip dengan ***Three-Point Lighting,*** namun tidak memerlukan *Rim light*



Gambar 2. 7 Penyusunan cahaya pada *two-point lighting*

(http://www.howto.gov/sites/default/files/b3-two-point-lighting.jpg)

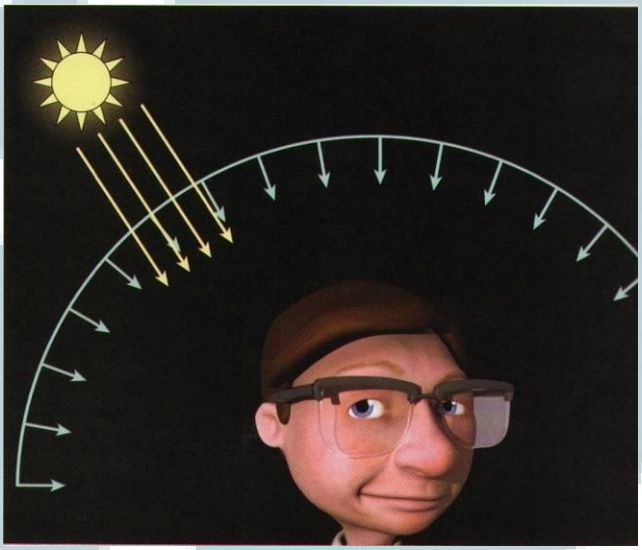
3. *One-Point Lighting*, atau pencahayaan dengan 1 titik adalah teknik pencahayaan ekstrim untuk memberikan efek dramatis. Penataan *lighting* ini hanya memiliki sebuah cahaya utama atau *key light* tanpa *fill light*. Penataan *lighting* ini menghasilkan transisi antara cahaya dan bayangan.



Gambar 2. 8 Penyusunan cahaya pada *one-point lighting*

(data pribadi)

4. *Natural Lighting* atau *lighting* alami adalah *lighting* dari lingkungan alami yang tidak bisa kita atur atau susun. Sebagai contoh, pada siang hari di luar, matahari adalah sumber cahaya langsung dengan yang menghasilkan bayangan, namun langit akan memberikan sejumlah cahaya rata yang dihasilkan oleh hampir setiap arah untuk menghasilkan *fill light* yang baik dengan warna biru.



Gambar 2. 9 Susunan cahaya pada *natural lighting*
(3D Animation essentials, Andy beane)

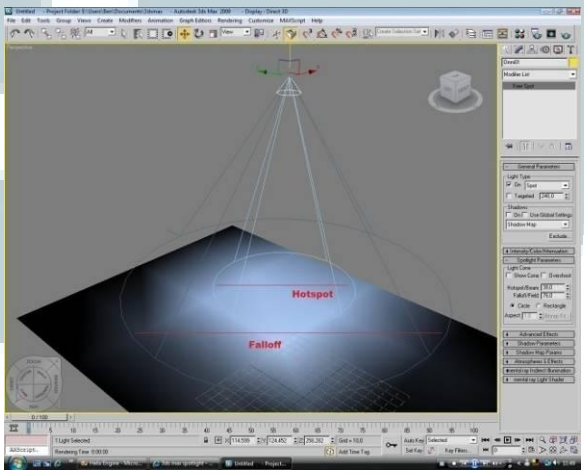
2.3. *Lighting* pada 3D animasi

Beane mengatakan bahwa pada proses *Lighting* pada 3D animasi,*lighting Artist* menyusun cahaya atau lampu pada *3D environment* selain untuk menunjukkan objek utama, juga untuk membangun suasana. Pekerjaan ini sangat serupa dengan *lighting artist* pada produksi film dan pelukis artistik. *Lighting* artist harus bisa menciptakan atmosfer dan suasana dalam setiap adegan. Mereka juga harus dapat menghasilkan produk *lighting* yang baik untuk menunjukkan setiap detail objek tanpa membuatnya nampak datar dan membosankan.

2.3.1. Tipe-tipe cahaya pada *software* 3D

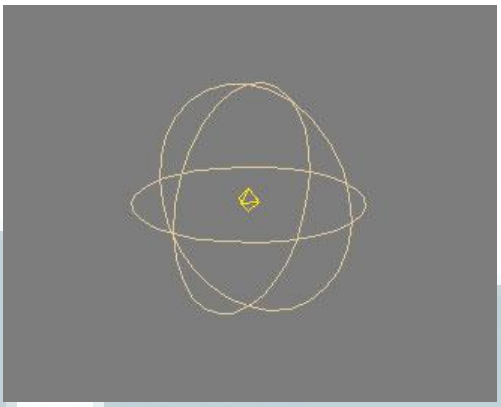
Beane mengatakan bahwa tipe-tipe cahaya yang digunakan dalam pekerjaan sebagai seorang *lighting* artist berbeda-beda. Namun, hampir semua *software* 3D memiliki jenis cahaya berikut ini :

1. ***Spotlight*** cahaya jenis ini adalah salah satu jenis cahaya yang paling sering digunakan pada *software* 3D. cahaya ini memancar dari sebuah titik, pada 1 arah, dan kita dapat mengatur sudut *cone* untuk menghasilkan hasil yang berbeda.



Gambar 2. 10 *Spotlight* dalam *software* 3ds Max
(http://www.gamedev.net/topic/492742-calculating-spotlight-exponent-from-falloff-and-hotspot/)

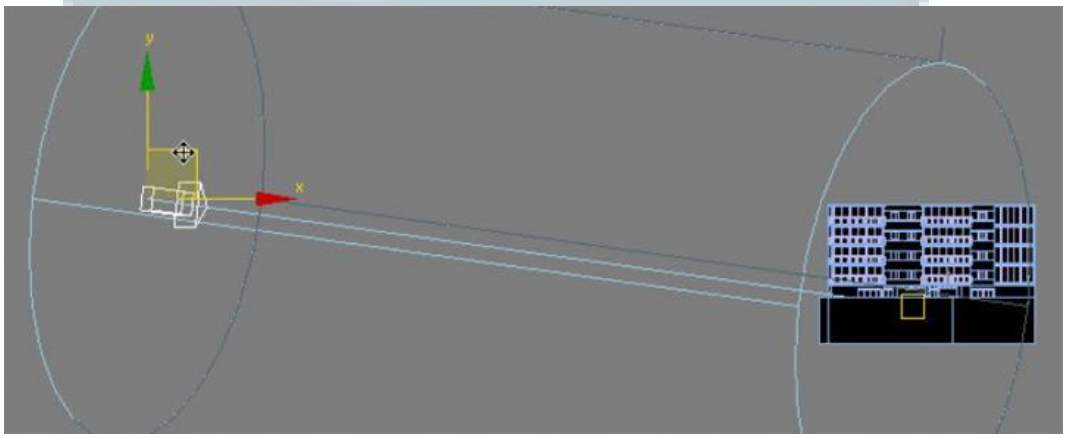
2. ***Point light / Omni*** cahaya ini memancar dari 1 titik ke semua arah, mirip seperti lampu pijar pada dunia nyata.



Gambar 2. 11 *Omni light*dalam *software* 3ds Max

(http://www.expertrating.com/courseware/3DCourse/3D-Lights-1_clip_image003.jpg)

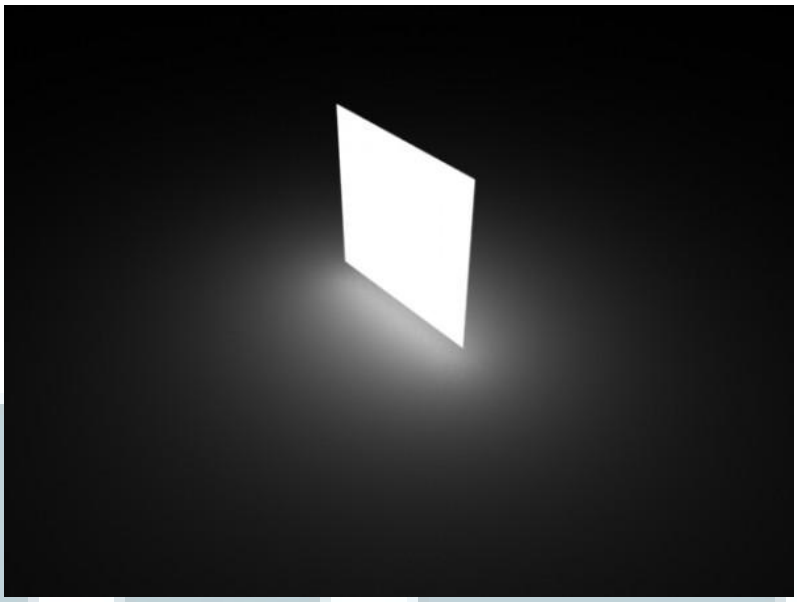
3. ***Infinite / Directional Light*** cahaya ini memancar secara paralel, seperti cahaya matahari.



Gambar 2. 12 *Directional light*dalam *software* 3ds Max

(http://www.cgarena.com/freestuff/tutorials/max/hdri-vray/directLight.jpg)

4. ***Area Light*** cahaya jenis ini adalah cahaya yang paling reallistik dan rumit. Area light memancarkan cahaya dari sebuah area atau permukaan seperti sebuah jendela, layar kaca komputer, atau kap lampu. Cahaya ini menciptakan bayangan yang lembut dan realistik.



Gambar 2. 13 *Area light*dalam *software* Maya
(http://www.jozvex.com/wp-content/uploads/defaultAreaWom-600x450.jpg)

2.3.2. Atribut cahaya pada *software* 3D

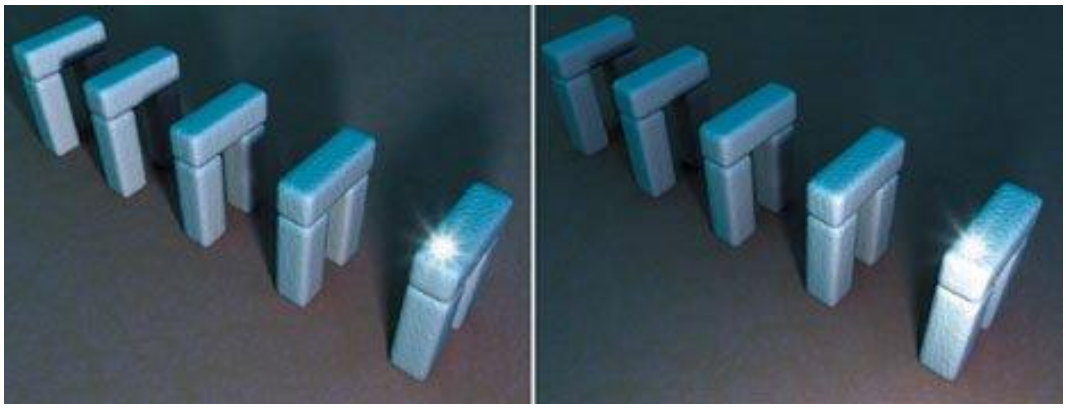
Beane menjelaskan mengenai atribut dari cahaya pada proses penataan *lighting*, tugas utama sebagai seorang *lighting* artist adalah memanipulasi atribut-atribut dari tipe-tipe sumber cahaya untuk menghasilkan hasil yang diinginkan.

1. **Intensitas** atribut ini mengindikasikan seberapa kuat cahaya tersebut. Dalam buku 3D Art essentials, Ami Chopine (2011) menjelaskan bahwa intensitas adalah seberapa terangnya cahaya, dan dapat diukur dengan satuan hitung dalam dunia nyata, seperti *lumen, candela,* dan *lux*, atau dengan satuan yang disediakan oleh *software* 3D. Untuk lebih menjelaskan hal ini, Brooker menjelaskan atribut ini lebih spesifik pada *software* 3Ds Max, dimana atribut intensitas disebut sebagai *multiplier*, nilai pada atribut ini mewakili kekuatan pada cahaya (dengan nilai positif atau negatif), sehingga apabila kita memberi nilai 2.0 , intensitas cahaya akan

digandakan, menggunakan nilai ini untuk meningkatkan intensitas cahaya dapat menyebabkan cahaya tampak terbakar dan menghasilkan warna yang tidak bisa ditampilkan oleh video. Oleh karena itu menggunakan nilai lebih dari 2 pada *multiplier* sangat tidak dianjurkan.

2. **Warna** atribut ini membuat artist bisa memberikan warna kepada sumber cahaya. Dalam dunia nyata, cahaya / lampu berwarna dapat dibuat denganmemberikan semacam jel atau filter berwarna pada sumber cahaya.

3. ***Decay / Attenuation*** atribut ini mengatur mengenai bagaimana cahaya yang dihasilkan sumber cahaya perlahan lahan menurun intensitasnya seilir dengan jarak, sebagaimana perilaku cahaya pada dunia nyata. *Software* 3D memberikan pilihan untuk pengguna untuk memilih apakah ingin menggunakan atribut decay ini atau tidak. Pilihan khas dari decay ini disajikan dengan *presets,* seperti linear falloff dan quadratic / falloff kuadrat terbalik. 2 jenis decay ini mengizinkan cahaya untuk jatuh dengan berbagai pola yang berbeda.



Gambar 2. 14 gambar kiri adalah hasil *scene* yang tidak menggunakan sistem decay (linear falloff), gambar di kanan adalah hasil penggunaan decay pada *scene*

(Digital Lighting & Rendering, second edition, Jeremy Birn)

4. **Bayangan** atribut ini sangatlah penting untuk memberikan kesan 3 dimensi pada sebuah bentuk. Tanpa bayangan, semua akan tampak datar dan membosankan. Tetapi bayangan tidak selalu diinginkan. *Lighting artist* dunia nyata seperti pada produksi film dan fotografi tidak memiliki pilihan untuk mengatur cahaya agar dapat menghasilkan bayangan atau tidak, tetapi 3D *lighting artist* bisa mengatur hal tersebut. Pilihan ini dapat membantu 3D *artist* dalam menghasilkan hasil yang mereka inginkan tanpa masalah akan adanya bayangan yang tidak diinginkan. Terdapat 2 jenis utama dari bayangan : bayangan raytraced dan bayangan *depthmap*. Masing-masing bayangan ini memiliki keuntungan dan kerugian.

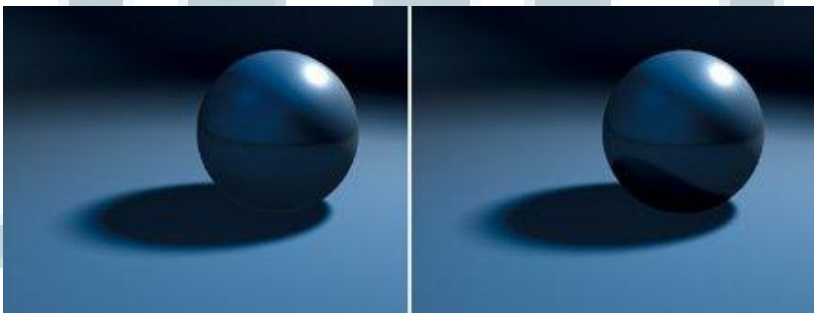
   a. **Bayangan *raytraced***

      adalah bayangan yang memiliki hasil paling akurat. Jenis ini menghasilkan bayangan yang tajam dengan *outline* yang sempurna. Bayangan raytraced juga dapat menembus objek transparan. Dari kedua jenis bayangan yang ada, Bayangan *Raytraced* sangat realistis dan halus, namun proses *rendering* membutuhkan waktu lama. Bayangan ini dihasilkan dari permukaan geometri, dimana memancarkan sinar dari setiap titik permukaan yang terlihat oleh kamera.

      Brooker menjelaskan bahwa bayangan *raytraced* dihasilkan dengan melakukan *tracing* pada jalan yang dilalui sinar, inilah mengapa jenis bayangan ini disebut *raytraced*, dengan mengikuti arah sinar *software* mampu mengkalkulasi sampai

tingkat derajat akurasi yang sangat tinggi dimana ini dihasilkan dari objek yang terdapat pada jalan dari sinar cahaya, pemilihan jenis bayangan ini dapat mempengaruhi lamanya waktu penghitungan *render* dengan sangat signifikan.

Birn menambahkan sebuah hal penting pada penggunaan bayangan r*aytraced*, yaitu perhitungan *trace depth*, proses *raytracing* tanpa batasan akan berpotensial untuk menghasilkan pengulangan perhitungan yang membuat komputer dapat me*render* refleksi dari refleksi dari refleksi berulang ulang apabila tidak dibatasi dengan angka yang terbatas pada proses kalkulasinya. Batasan ini dapat menimbulkan masalah kepada hilangnya bayangan *raytraced*. Apabila bayangan *raytraced* tidak muncul pada sebuah *refleksi*, atau tidak muncul ketika kita melihat objek dengan material gelas yang menghasilkan refraksi, kemungkinannya adalah, kita menjalankan penghitungan *depth* yang terbatas.

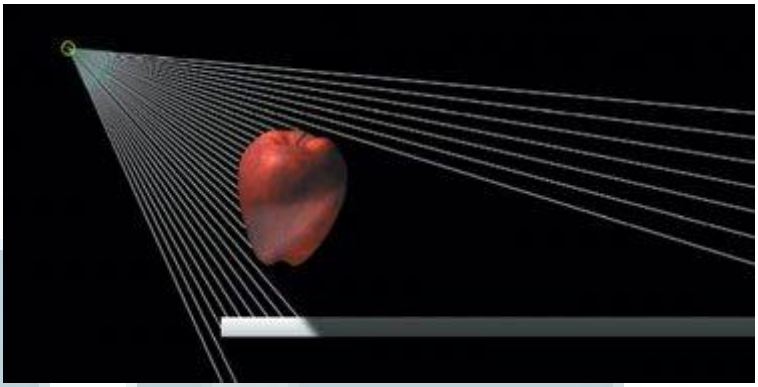


Gambar 2. 15 nilai 1 pada *trace depth*, pada refleksi dari bola (gambar kiri), dengan nilai 2, kita bisa melihat adanya refleksi bayangan *raytraced* pada bola (gambar kanan)
(Digital Lighting &*Rendering*, second edition, Jeremy Birn)

b. **Bayangan *Depth Map***

adalah bayangan yang sangat cepat dihasilkan saat proses *rendering*, dan dapat menghasilkan hasil yang bagus apabila dipakai dengan benar. Bayangan jenis ini dapat membuat bayangan yang halus atau tajam namun tidak dapat menghasilkan bayangan transparan. Untuk menghasilkan bayangan ini, setiap lampu atau cahaya memancarkan *resolution based map* dalam *scene* 3D, dan *render engine* menyimpan informasi depth dari sumber cahaya. Hanya geometri yang terdekat dengan cahaya, seperti yang telah dikalkulasikan oleh *depth map*, yang akan menghasilkan bayangan.

Birn menjelaskan bahwa bayangan *depth map* atau yang biasa disebut juga dengan *shadow map* adalah susunan angka yang merepresentasikan jarak. Sebelum r*enderer* melakukan proses *render* pada *scene* yang terlihat pada kamera, ia menghitung *depth map* dari titik penglihatan setiap cahaya yang akan menghasilkan bayangan *depthmap*. Untuk setiap arah yang disinari cahaya, *depth map* menyimpan jarak dari cahaya ke objek yang menghasilkan bayangan.



Gambar 2. 16 Bayangan *depth map* dihasilkan berdasarkan susunan dari penghitungan cahaya ke geometri terdekat, di gambar ini ditunjukkan dengan garis putih

(Digital Lighting & Rendering, second edition, Jeremy Birn)

Selama proses *rendering*, cahaya akan dipotong pada jarak tertentu oleh *depth map*, untuk melihat dimana bagian yang menghasilkan bayangan dan yang tidak. Proses ini menghemat lamanya waktu *rendering* secara efektif, karena *renderer* tidak lagi perlu untuk menghitung berulang kali untuk memverivikasi dimana objek berada diantara tanah dan cahaya dalam *scene*.

Birn juga menjelaskan mengenai resolusi dan penggunaan memori komputer, sebuah penghitungan jarak di dalam *depthmap* disimpan sebagai nilai *floating point*. Nilai *floating point* dapat menyimpan semua angka, dari pecahan kecil sampai jarak yang besar, namun menggunakan 4 *bytes* untuk menyimpan setiap nilai. Resolusi bayangan *depthmap* dipakai baik untuk dimensi vertikan maupun dimensi horizontal dari *map*, yang dapat diartikan bahwa jumlah *bytes* yang dipakai adalah 4 * (resolusi^2)

, berikut adalah tabel yang menunjukkan memori yang digunakan oleh resolusi *shadow map*, dalam *megabytes*.

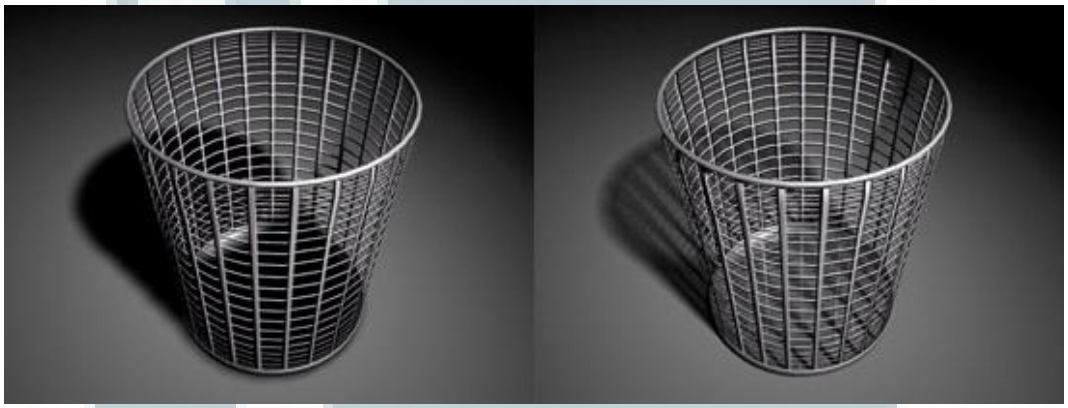
***DepthMap* Resolution**	**Memory Used**
128	0.06 MB
256	0.25 MB
512	1 MB
1024	4 MB
2048	16 MB
4096	64 MB

Tabel 2. 1 Penggunaan Memori pada *Shadow Map*
(Digital Lighting & Rendering, second edition, Jeremy Birn)

Birn juga menjelaskan mengenai bias, terkadang hasil *render* memiliki artefak seperti pola garis atau kotak-kotak ini dikarenakan pengaturan parameter bernama bias terlalu rendah. Untuk *scene* yang dibuat dengan skala yang besar, nilai bias perlu ditingkatkan agar munculnya artefak dapat dihindari.

Gambar 2. 17 Artefak yang muncul dikarenakan nilai bias yang rendah, gambar kiri dengan nilai 0.005, tengah 0.02, dan kanan 0.25
(Digital Lighting &Rendering, second edition, Jeremy Birn)

Bias adalah angka yang diberikan untuk setiap penghitungan jarak dari *shadow map*, mendorong jarak mulainya bayangan menjauh dari sumber cahaya. Meningkatkan nilai bias, agar bayangan tidak mulai terlalu cepat dan pada akhirnya menghasilkan artefak.



Gambar 2. 18 Sebelah kiri bayangan *depth map* tidak dapat menembus objek transparan sedangkan di kanan bayangan *raytrace* dapat menembus objek transparan
(http://www.thegnomonworkshop.com/img/tutorials/transparency_shadows/trashcan_dmap_vs_raytrace.jpg)

2.3.3. *Environment Lighting*

Sesuai dengan tugas akhir penulis yang membahas mengenai proses *lighting* pada produksi 3D animasi, sehingga pembahasan mengenai *environment* ini sangat penting untuk dibahas.

Menurut Birn, proses *lighting* sebuah *environment* membutuhkan kepekaan akan bagaimana *lighting* bekerja pada dunia nyata. Secara alamiah kondisi *lighting* pada siang hari dan malam hari pastinya berbeda,begitu juga kondisi cuaca, juga sudut pancaran matahari.

2.3.4. *Scene* Siang Hari

Lighting artist dapat membuat pencahayaan luar ruangan sederhana dengan menambahkan tiga elemen pada *scene* Anda: Pertama, adegan siang hari sering didominasi oleh sinar matahari, pencahayaan datang langsung dari matahari. Kedua, cahaya dari langit perlu ditambahkan.

Dalam kehidupan nyata cahaya langit sebenarnya adalah cahaya dari matahariyang telah menembus melalui atmosfer, tetapi dalam dunia 3D kita menganggap cahaya langit sebagai sumber penerangan terpisah. Sehingga, *indirect light* harus ditambahkan. *Indirect light* adalah cahaya yang dipantulkan dari permukaan lain dalam *scene*, ia tidak datang langsung dari matahari atau langit.

2.3.4.1. Cahaya Matahari

Cahaya matahari tidak memerlukan *decay* atau *attenuation*, karena cahaya matahari telah memancar dari miliaran kilometer.

Untuk sebagian besar hari , dapat menggunakan warna kuning pada cahaya matahari , lalu berubah menjadi oranye atau bahkan merah saat matahari terbit atau tenggelam.

Beberapa hal yang perlu diperhatikan saat membuat cahaya matahari adalah lebih baik untuk menggunakan *direct light* daripada spot light karena *direct light* menghasilkan cahaya yang linear sesuai dengan sifat asli dari matahari, selain itu pemilihan bayangan *raytrace* lebih disarankan.

Gambar 2. 19 Hasil tes *render scene* 3D yang telah diberikan cahaya matahari
(Digital Lighting & Rendering, Jeremy Birn)

2.3.4.2. Cahaya langit

Menambahkan cahaya langit realistik dapat mengurangi kelebihan kontras, mengisi bagian bayangan dengan cahaya halus berwarna biru, dan membuat keseimbangan warna pada *scene* menjadi lebih seimbang.

Salah satu cara untuk membuat cahaya langit adalah dengan menggandakan cahaya matahari, dan memutar hasil penggandaan menjauh 90 derajat dari cahaya matahari. turunkan intensitas dari cahaya menjadi 50 persen lebih kecil dari intensitas matahari, lalu ubah warna lampu menjadi biru. untuk menghasilkan bayangan yang lebih lembut, bisa dengan cara menurunkan resolusi *shadowmap* atau menaikkan *blurring* atau *filtering*. Metode ini biasa dinamakan dengan *array*.

Gambar 2. 20 Hasil tes *render scene* 3D sebelumnya yang telah ditambahkan dengan cahaya langit
(Digital Lighting & Rendering, Jeremy Birn)

2.3.4.3. Cahaya tidak langsung

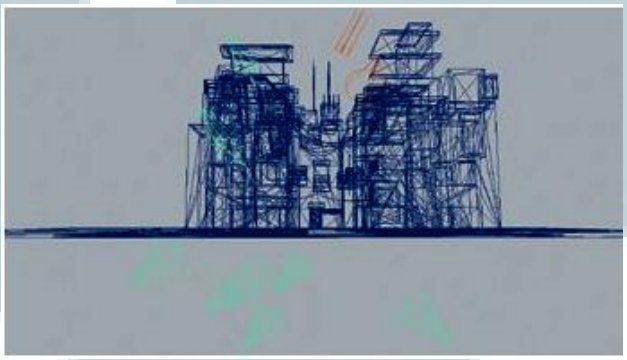
Sebagian besar cahaya dalam *scene lighting* siang hari didominasi oleh cahaya matahari dan langit, namun untuk menghasilkan hasil yang realistis juga memerlukan cahaya tidak langsung atau *bounce light. Bounce light* mensimulasikan pantulan cahaya dari matahari dan langit pada tanah dan permukaan lainnya dan secara tidak langsung menerangi benda benda lainnya.

Bounce light mirip dengan cahaya langit, mereka tidak banyak mempengaruhi spekularitas objek, karena kita tidak akan memberikan *bounce light* pada permukaan reflektif.

Pemilihan warna pada *bounce light* harus didasarkan kepada permukaan yang memantulkan cahaya. Biasanya kita akan membutuhkan *bounce light* yang cukup untuk mencakupi luasnya wilayah *scene* 3D kita.

Posisikan lampu dibawah tanah, yang mengarah kepada bangunan, karakter, atau objek lain yang membutuhkan cahaya tidak langsung ini

Terkadang *bounce light* perlu memberikan bayangan yang lembut, terutama jika mereka menerangi karakter, sehingga mereka tidak membuat interior mulut karakter menjadi terlalu terang. Jika lampu kita ada dibawah tanah dan perlu untuk menyinari bagian tersebut, Kita harus melakukan *unlink* kepada tanah agar tanah tidak menghasilkan bayangan yang dapat menutupi *bounce light.*



Gambar 2. 21 Skema penyusunan lampu pada *scene* 3D, *bounce light* diwakili oleh objek berwarna hijau

(Digital Lighting & Rendering, Jeremy Birn)

Gambar 2. 22 Hasil akhir *scene* dengan semua cahaya

(Digital Lighting & Rendering, Jeremy Birn)

2.3.5. *Scene* Malam Hari

Kita dapat mengatur *lighting* pada malam hari dengan prinsip yang sama dengan siang hari, dengan beberapa modifikasi. Sinar bulan dan cahaya langit malam dapat dibuat dengan cara yang sama dengan sinar matahari dan cahaya langit pada siang hari, namun cahaya bulan dibuat dengan lebih redup.

Pada malam hari, cahaya dari langit haruslah berwarna biru lembut. Cahaya bulan dapat menggunakan baik warna kuning maupun warna biru. Pada penggunaannya, lebih sering ditemukan cahaya bulan sebagai warna kuning pada saat cahaya hanya datang dari bulan dan langit, namun apabila pada *scene* kita terdapat juga lampu-lampu jalan, cahaya bulan akan nampak lebih biru.

Kunci dari menyusun *lighting* pada *scene* malam hari adalah tidak melakukan under-expose seluruh *scene*, melainkan harus menggunakan banyak kontras. *Scene* mungkin didominasi oleh bayangan, namun kegelapan harus dapat dipecah dengan memberikan *highlight* terang dan penambahan *rim light* dan kilauan cahaya.

Gambar 2. 23 *Scene* malam hari memberikan kontras yang dihasilkan oleh kilauan
(Digital Lighting & Rendering, Jeremy Birn)

2.4. *Rendering*

Proses penataan *lighting* saat erat hubungannya dengan proses *rendering*, Beane mengatakan bahwa *rendering*, adalah tahap akhir dari sebuah pipeline produksi, mengambil semua unsur unsur yang ada dalam *scene* 3D kita menjadi sebuah hasil video 2D atau gambar 2D. Hasil dari proses *rendering* ini akan diberikan kepada tim post produksi untuk persiapan akhir dan hasil jadi akhir.

Beberapa *render engine* banyak terdapat di pasaran saat ini. Ini semua termasuk *render engine* bawaan dari *software* 3D, terdapat juga *plug-in* tambahan yang dapat bekerja di dalam *software* 3D animasi, atau sebagai *software rendering* yang berdiri sendiri.

2.4.1. Metode Dasar *Rendering*

Setiap *render engine* yang berbeda pastinya juga memiliki cara *render* yang berbeda. Semuanya menawarkan metode dasar *rendering* yang berbeda : *scanline* dan *raytracing*.

2.4.1.1. Scanline

Pada *rendering* dengan metode ini, algoritma berjalan dengan sangat cepat. Kerugian dari metode ini adalah tidak dapat menghitung refleksi, refraksi, atau *global illumination* yang kompleks seperti yang ditawarkan oleh *lighting* dan *rendering engine* yang beredar sekarang ini seperti mental ray, V-Ray, dan RenderMan. *Scanline* adalah metode *rendering* yang sangat menguntungkan saat digunakan untuk melakukan informasi pre-visualisai dan percobaan cepat untuk melihat bagaimana sebuah *scene* mengalami kemajuan. Metode *scanline* juga bekerja sangat baik pada *rendering* untuk hasil seperti kartun, *flat*, atau *cell shaded*.

Rendering scanline bekerja dengan basis baris-perbaris dalam menyelesaikan seluruh gambar, pertama dengan mengukur permukaan poligon apa yang ada di dalam sebuah *scene* lalu setelah itu ia memutuskan poligon mana yang akan terlihat atau yang tidak terlihat di kamera, setelah itu baru melakukan proses *render*. Kegiatan pemilihan objek poligon ini mengizinkan *scanline rendering* untuk tidak menghitung apa yang tidak terdapat pada kamera, dimana hal ini akan sangat menghemat waktu.

2.4.1.2. Raytracing

Metode *raytracing* adalah sebuah metode yang lebih kuat dan sempurna dibandingkan dengan *scanline*. namun memiliki kerugian. Metode ini dapat mengkalkulasikan refleksi, refraksi, dan berbagai pilihan *rendering* yang lebih rumit *pada render engine*.

Raytracing memancarkan sinar kepada setiap *pixel* dalam layar dan mengambil sampel shape dan shader dari objek. Pada titik sampel, apabila *shader* tersebut reflektif, *renderer* akan membuat pancaran cahaya baru dan mengambil sampel untuk menentukan apa yang di refleksikan. Cahaya - cahaya yang dipancarkan ini akan terus memancar sampai mengenai permukaan yang tidak reflektif . Pemancaran cahaya ini mengizinkan metode *raytracing render* untuk menghasilkan tingkatan realisme yang lebih besar daripada yang bisa dihasilkan oleh *scanline rendering*, namun tentunya dengan membutuhkan waktu yang lebih lama untuk komputer dapat memproses hasil akhir gambar.

2.4.2. Global Illumination

Global illumination adalah istilah umum yang dipakai untuk mendeskripsikan sekelompok algoritme dan metode kompleks untuk menghasilkan hasil *render* yang lebih realistik. Algoritma-algoritma ini mirip dan berdasar kepada algoritma *Raytracing* namun ditambahkan fungsi lebih kepada algoritma *rendering* untuk menghasilkan hasil yang lebih realistik. Berikut akan dijelaskan mengenai metode *global illumination*.

2.4.2.1. Photon Mapping

Photon mapping adalah algortima *render globa lillumination* yang mengikuti cara dari *raytracing*, tapi digunakan secara terbalik. Melainkan dengan cara memproyeksikan cahaya dari kamera untuk mengkalkulasikan *scene*, *photons* dipancarkan dari sumber cahaya dan memantul-mantul dalam *scene*, meninggalkan bekas cahaya yang didapat dari terkena pantulan *photon*. Lalu bekas cahaya tersebut mengambil sampel dari bekas cahaya lain di sekitarnya untuk menentukan intensitas akhir *lighting* untuk bekas cahaya . Tipe *lighting* ini sangat membutuhkan waktu *render* dan sumber daya komputer yang banyak untuk menghasilkan hasil akhirnya, namun hasil yang didapat bisa menjadi sangat realistis.

Joep van der Steen (2007) menjelaskan dalam bukunya, *Rendering with Mental ray and 3Ds Max* bahwa sumber cahaya memancarkan sinar cahaya, yang dapat dipertimbangkan dalam keadaan tertentu sebagai partikel kecil, inilah yang disebut dengan *photon*

Photon tercermin oleh cermin, dibiaskan melalui kaca, atau tersebar oleh permukaan bersifat *diffuse*. Keuntungan yang besar dari penghitungan *photon* adalah bahwa mereka meniru apa yang terjadi di dunia nyata. Hal ini dapat mencakup refleksi pada logam dan pembiasan melalui kaca. Setiap cahaya membawa sejumlah energi yang terbagi atas jumlah *photon* cahaya yang diatur untuk dipancarkan. Ketika *photon* mulai bergerak di dalam *scene*, ia memiliki energi dan warna yang sama dengan warna cahaya.

Warna dan perubahan energi terjadi ketika *photon* memantul dari permukaan berwarna atau berinteraksi dengan cara lain dengan benda di dalam *scene*. Ketika proses ini berhenti, *photon* yang tersisa dengan sejumlah energi dan warna, dapat benar-benar berbeda dari ketika memulai rute.

2.4.2.2. Image-Based Lighting

Image-based lighting adalah metode *global illumintaion* yang memperbolehkan pengguna untuk membuat sebuah kubah atau bola yang mengelilingi *scene* untuk mencakup *scene* yang sedang dikerjakan dan menetapkan gambar yang akan menerangi *scene*, alih-alih memakai lampu atau cahaya (*light*). Gambar yang digunakan untuk image-based *lighting* ini biasanya adalah gambar *high dynamic range (HDRI).*

Brooker menjelaskan bahwa *Dynamic range* pada *HDRI* bisa diartikan sebagai rasio kontras antara bagian tergelap dan bagian paling terang pada gambar, sebuah foto dengan sumber cahaya yang terang dan bayangan yang gelap akan menjadi sebuah contoh yang tepat untuk *HDRI*.

Gambar dengan 8-bit yang biasa kita pakai dalam bekerja terbatas hanya menyimpan informasi sebanyak 8-bit pada setiap channel warna. Ini berarti warna merah, hijau, dan biru yang menghasilkan setiap pixel pada gambar, direpresentasikan melalui sebuah bilangan bulat antara 0 - 255. Hasil demikian merupakan standar pada alat alat seperti kamera digital dan lainnya. Tingkat pixel warna putih yang sama pada R=255, G=255, B=255

dipakai pada gambar dengan basis 8-bit di daerah seperti sebagian langit, tembok juga matahari, kita tahu bahwa di dunia nyata setiap elemen ini memiliki tingkat luminance yang berbeda, namun gambar 8-bit tetap menampilkan gambar dengan tingkat putih yang sama.

Bebeda dengan itu, *HDRI* menyimpan informasi gambar pada format yang tidak dibatasi, meskipun mintor kita hanya dapet menampilkan gambar dalam bentuk 8-bit, sehingga warna putih tetap akan ditampilkan sebagai warna yang sama pada R=255, G=255, B=255; di dalam semua hal disimpan dalam format *floating point*, mampu menghasilkan tingkatan warna dari bagian paling gelap sampai bagian paling terang pada sebuah gambar. tingkat pixel putih dengan nilai 300, berada pada tingkatan lebih tinggi sedikit dari jarak standar 0 - 255 RGB dan dapat merepresentasikan permukaan putih yang dicat, namun tingkatan *pixel* putih dengan nilai 10.000 misalnya akan dipakai untuk merepresentasikan sebuah area yang sangat terang, bisa dari langit, ataupun matahari. Karena nilai pixel pada *HDRI* proporsional secara langsung kepada tingkat luminance dari objek, *renderengine* dapat membedakan antara tingkat luminance yang berbeda ini dan menghasilkan sebuah hasil gambar realistik yang mengejutkan.

Untuk menghasilkan gambar *HDRI* dapat dengan 2 cara, bisa dengan me*render* menggunakan algoritma *global illumination* dan menyimpan informasi pada format *HDR*, atau dengan mengambil foto yang memiliki tingkat *exposure* yang berbeda dari tingkat paling terang,

terang, normal, dan gelap. Setelah mendapat semua hasil foto, bisa dimasukkan ke program yang memang dapat menciptakan foto *HDR* .

Gambar 2. 24 3 foto yang diambil dengan tingkat *exposure* yang berbeda
(Essential CG Lighting Techniques , Darren Brooker)

Biasanya foto untuk menciptakan *HDRI* diambil pada sebuah bola khrom yang bersifat reflektif sehingga dapat dijadikan *map* berbentuk bola yang menyimpan semua informasi cahaya.

Gambar 2. 25 yang diambil menggunakan bola khrom reflektif
(http://3.bp.blogspot.com/_uQ4W_w5mS6I/TJ9CQaLMzUI/AAAAAAAAARE/SizJdowfNcs/s1600/HdriWIP.jpg)

2.4.3. Final Gather

Steen menjelaskan bahwa *Final gather* adalah sebuah teknik yang yang dapat digunakan untuk menciptakan inderect *illumination*. *Final gather* menembakkan sinar dalam *scene*, banyaknya berdasarkan bagaiman sinar di representasikan oleh *photon* yang ditembakkan oleh sumber cahaya atau kamera. namun ada 1 perbedaan yang fundamental. Sinar yang dipakai oleh *final gather* tidak muncul dari sumber cahaya atau kamera, mereka muncul dari dalam geometri sendiri. Teknik ini menembakkan sinar kepada *environment* disekeliling *scene* untuk mengumpulkan informasi dari *environment* ini, lalu ia mengambil semua informasi dari sinar-sinar tersebut lalu menghitung berapa banyak cahaya sampai ke titik dimana *final gather* dimulai, juga mengambil akun informasi dari titik yang bersebalahan dan menggunakan proses yang sama dan merata-ratakan semuanya. *Final gather* dapat digunakan tanpa *globa lillumination* untuk menciptakan efek indirect *illumination* pada seluruh *scene*, meskipun hasil dari *final gather* dan *photon mapping* akan menunjukkan hasil yang berbeda, keduanya 100 persen tepat secara fisik. Perbedaanya biasa disebabkan karena shader yang digunakan didalam *scene*, dan bukan karena algoritma yang dijalankan dibawahnya.

Final gather diatur dari awalnya untuk tidak menghitung pantulan pantulan cahaya, sehingga hasil akhirnya biasanya nampak lebih gelap dari gambar yang dihasilkan oleh teknik *photon mapping*, yang memang menghitung pantulan-pantulan dari awal dan memegang distribusi energi yang benar dalam

keseluruhan *scene*. Penting untuk mencatat bahwa teknik *final gather* dapat diatur untuk mengkalulasi pantulan pantulan ini.

Untungnya teknik *final gather* dan *photon mapping* dapat digunakan bersamaan. Secara ideal, ini dapat meningkatkan kualitas keseluruhan dari gambar, karena *final gather* dapat menghasilkan lebih baik untuk gambar dengan detail kecil dan halus, juga dapat menghaluskan warna di dalam gambar. Sangat digunakan berkombinasi, teknik *final gather*melakukan pencarian kepada cahaya tidak langsung dari *photon* untuk menghasilkan hasil *render* yang memiliki kedalaman cahaya yang baik pada bayangan dan variasi warna yang lembut pada area yang tersinari.

Gambar 2. 26 Hasil *render* kombinasi antara teknik *final gather* dan *global illumination*

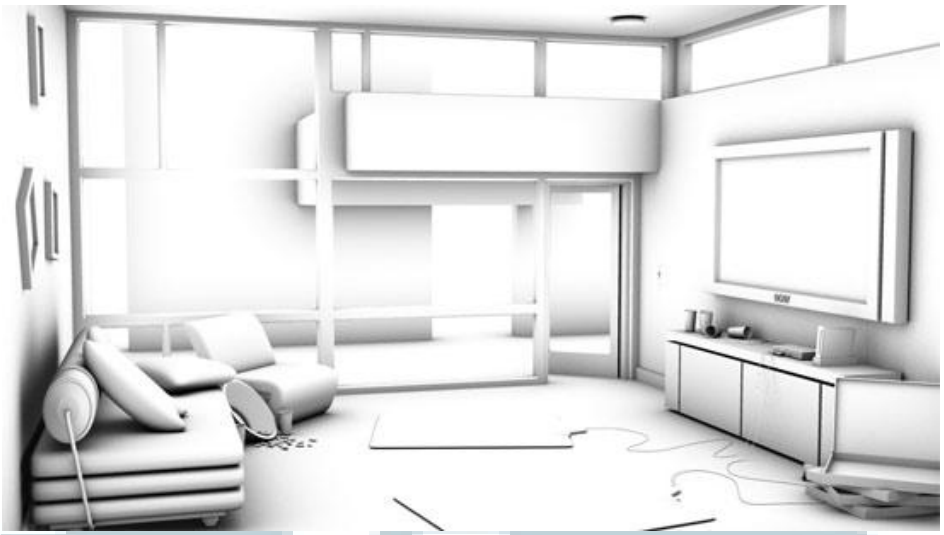
(*Render*ing with Mental ray and 3Ds Max , Joep van der Steen)

2.4.4. Ambient Oclusion

Brooker menambahkan penjelasan mengenai *Ambient* oclussion, sebuah teknik yang ditemukan oleh perusahaan industrial Light & Magic dan dipublikasikan pada tahun 2002 di SIGGRAPH. Teknologi ini digunakan oleh studio studio besar untuk memberikan tingkatan realisme yang tinggi tanpa harus menggunakan sistem *render*ing *Global Illumination.*

Penamaan *Ambient occlusion* diambil dari definisi akurat terhadap apa yang dijelaskan, yaitu berapa banyaknya cahaya *ambient* yang didapat oleh permukaan *scene* melalui bentuk objeknya. Namun, *ambientocclusion* masih sangat menguras waktu *rendering,* sehingga akan menjadi pilihan tepat untuk me*render ambientocclusion* pass secara terpisah dan baru nanti pada proses compositing, hasil *renderambientocclusion* dapat ditambahkan, atau dapat mem*bake* hasil *ambientocclusion* sebagai material.

Steen menambahkan, bahwa teknik *AmbientOcclusion* adalah teknik yang tidak memerlukan penataan *lighting* yang rumit, namun tetap dapat menghasilkan sebuah hasil gambar yang sangat realistik. *Ambientocclusion* menciptakan map hitam putih (*grayscale*) berdasarkan kepada berapa banyak area *environment* yang dapat terlihat dari setiap titik permukaan pada geometri. *Map grayscale* yang dihasilkan, menunjukkan berapa banyaknya *ambient* light yang bisa diterima oleh permukaan.

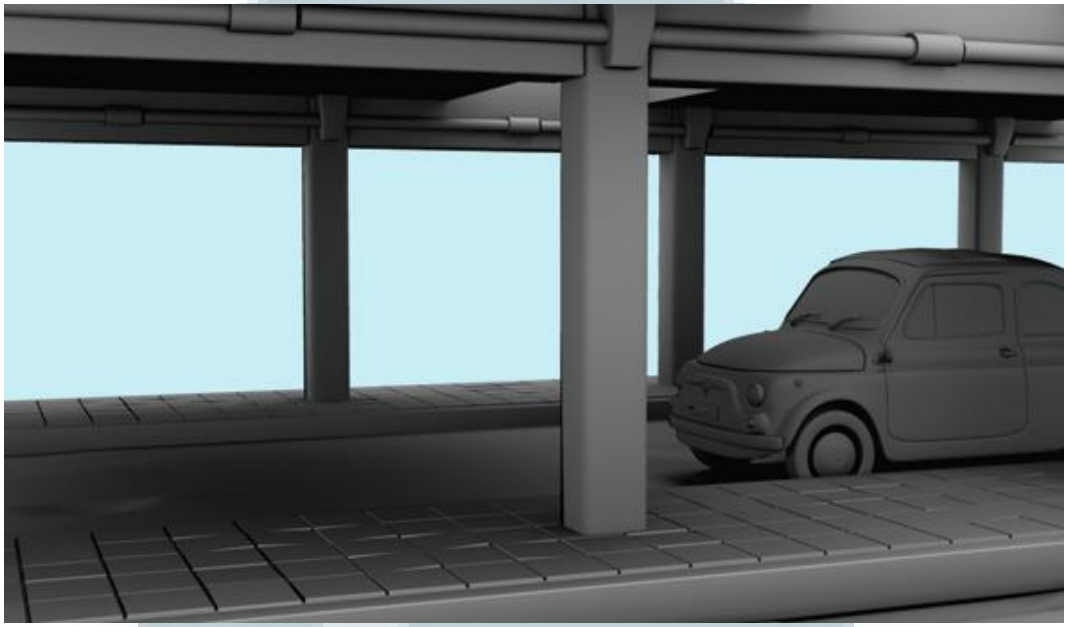


Gambar 2. 27 Hasil *render* menggunakan *mapambientocclusion*
(http://my.smithmicro.com/tutorials/1405.html)

Teori yang mendasari *ambient occlusion* ini adalah, selalu ada sebuah *ambientlight* abadi. Dengan demikian permukaan yang ada selalu mendapat sejumlah *ambient light*, namun tidak dengan jumlah yang konstan. Jumlahnya tergantung kepada berapa banyak permukaan yang teroklusi, atau terhalangi oleh geometri lain.Apa yang terjadi di dalam adalah area di atas titik yang akan terkena *shade* akan menjadi sampel untuk geometri yang menjadi penghalang. Apabila ada yang ditemukan, presentase dari halangan langsung diterjemahkan menjadi faktor *occlusion*.

Occlusion memiliki beberapa kegunaan. salah satunya adalah *ambient occlusion*, dimana *shader* yang dipakai untuk menghitung skala kontribusi cahaya *ambient*. Kegunaan lain *occlusion* adalah *occlusion* reflektif, dimana *shader* digunakan untuk menskala kontribusi dari *reflection map*. kegunaan lainnya adalah untuk menciptakan files untuk kompositing eksternal, dimana *shader*

occlusion di pakai di setiap material dalam *scene*. Hasilnya dapat digunakan untuk memodulasi hasil *render pass* yang lain untuk menghasilkan kompositing yang baik di postproduksi.



Gambar 2. 28 *render*dengan hanya *ambient occlusion* dan tanpa cahaya
(*Rendering* with Mental ray and 3Ds Max , Joep van der Steen)

2.5. Mental Ray *Rendering*

Steen menjelaskan bahwa Mental Ray adalah sebuah *renderengine* yang kita dapatkan dengan standar dari *software* 3Ds Max buatan Autodesk, mental ray adalah sebuah *engine* yang memiliki banyak potensi dan mungkin adalah yang terbaik pada saat ini.

Sebagai sebuah *render engine*, mental ray dapat menghasilkan dan melakukan berbagai efek atau pengaturan yang dapat menghasilkan hasil yang tidak dapat dicapai oleh *rendering scanline* standar yang menjadi default dari 3Ds

Max, keunggulannya adalah mental ray dapat melakukan eksekusi *Global Illumintaion, reflection/ refraction, ray-tracing, caustic, depth of field* dan berbagai hal lainnya.

karena mental ray telah sepenuhnya terintegrasi dengan *user intraface* utama dari 3Ds Max, kita tidak akan melihat perbedaan saat melakukan *rendering*. Sebuah *Scene* yang dibuat pada 3Ds Max akan melewati mental ray *rendering engine*, lalu ditransformasikan dan dibuat ulang di dalam mental ray, akhirnya *scene* tersebut di *render* dan hasilnya terlihat di layar komputer kita. Karena adanya integrasi di dalam *software* utama, kita tidak menyadari proses ini.

Cara terbaik untuk memulai menggunakan mental ray adalah dengan menggunakan berbagai pengaturan yang biasa dipakai pada *scanline render*, dengan begini kita dapat tetap menggunakan tipe material dan cahaya yang sama dengan tipe yang sudah terbiasa kita pakai. Setelah mulai terbiasa, kita dapat memulai menggunakan pilihan pilihan yang terdapat khusus pada mental ray.

2.6. *Multi Pass Rendering*

Chopine menjelaskan bahwa sebuah gambar dapat di*render* dengan beberapa lapisan juga disebut dengan *pass* atau *element* . Lapisan-lapisan ini dapat mengandung informasi yang berbeda seperti *lighting map*, bayangan, *dan Z-depth*. Ini dapat digunakan pada saat proses pasca produksi dan compositing. Ini memberikan kita kewenangan untuk mengedit gambar tanpa harus mengulang proses *render*ing. Sebagai contoh apabila kita me*rendermap lighting* untuk setiap cahaya dan mengaturnya untuk berwarna putih, kita dapat mengatur seberapa

besar terangnya cahaya juga warna dari cahaya tersebut pada pasca produksi. keuntungan lainnya lagi adalah mampunya kita untuk menyembunyikan objek dari proses *render*. Ini memungkinkan kita untuk menggunakan pentauran dengan kualitas terbaik untuk beberapa bagian saja pada *scene*. Menggunakan ini dan *mapZ-depth*, kita dapat melakukan kembali proses kompositing, dan dapat melakukan kedalaman palsu dengan menciptakan efek *blurring*, elemen seperti partikel dapat di *render* secara terpisah juga agar dapat memudahkan proses pasca produksi.

Gambar 2. 29 Beberapa hasil *multi pass rendering*

(http://www.cgsociety.org/static/images/feature/mo_crash_15_passes.jpg)