Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Team project ©2017
Dony Pratidana S. Hum | Bima Agus Setyawan S. IIP

BAB II

TINJAUAN PUSTAKA

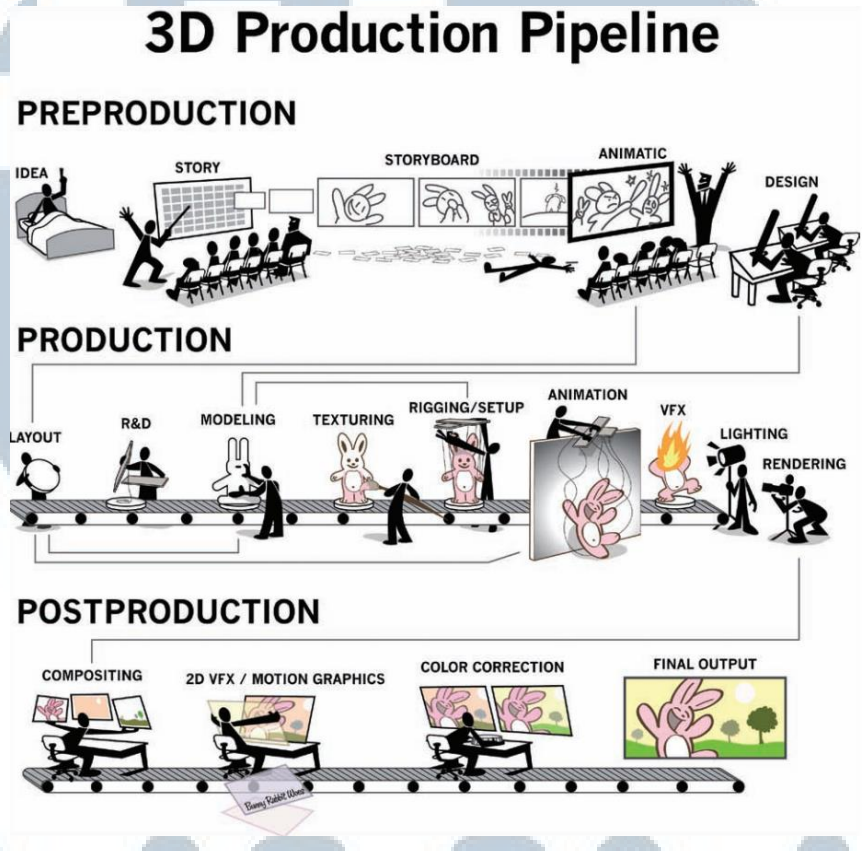
2.1. Tahapan Produksi

Proses produksi animasi terbagi dalam tiga tahap, yaitu praproduksi, produksi dan pascaproduksi. Tahap praproduksi adalah tahap yang cukup besar dan menentukan keseluruhan produksi. Apakah ide yang ingin disampaikan, bagaimana menerapkan ide tersebut dalam cerita dan bagaimana cerita tersebut divisualisasikan dengan baik merupakan perencanaan yang dilakukan dalam praproduksi. Produksi dapat dimulai setelah semua perancangan awal sudah selesai dilakukan. Produksi adalah tahap penciptaan elemen visual sesuai dengan apa yang sudah direncanakan sebelumnya. Pada akhirnya, semua elemen yang sudah diciptakan disatukan dan disempurnakan dalam tahap pascaproduksi (Beane, 2012).

Beane juga menambahkan bahwa tahap produksi terbagi dalam sembilan tahap, yaitu *layout, research and Development, modelling, texturing, rigging/setup, animation, vfx, lighting,* dan *rendering*. Tahap *rigging* adalah tahapan yang dilakukan setelah objek tiga dimensi diciptakan dalam tahap *modelling*. *Rigging* adalah tahap yang membantu *animator* dalam menggerakkan suatu benda atau tokoh tiga dimensi. Benda atau tokoh akan sulit untuk bergerak sesuai keinginan *animator* tanpa adanya bantuan dari proses *rigging*, yaitu proses pembentukan sistem kerangka beserta kontrol sesuai dengan kebutuhan *animator*.

Perancangan Algoritma Animasi..., Yosua, FSD UMN, 2018

Rigging juga mendukung *animator* dengan menciptakan sistem kontrol yang mudah bagi *animator* untuk bekerja secara cepat serta efisien (Beane, 2012).



Gambar 2.1. 3D Production Pipeline
(*3D Animation Essentials*, 2012)

2.2. Animasi

Animasi adalah tahap pembuatan pergerakan objek atau tokoh sehingga menciptakan kesan atau ilusi nyata dan hidup. Animasi sendiri terbagi dalam tiga jenis, yaitu *hand-keyframed animation, motion capture*, dan animasi prosedural (Beane, 2012). Dalam menghasilkan pergerakan langkah kaki, Karim et al. (2012) menegaskan tiga jenis animasi yaitu *data driven techniques, physic-based techniques* dan *procedural techniques*.

2.3. Animasi Prosedural

Animasi Prosedural adalah animasi yang dibuat oleh *programmer* dengan cara membuat aturan terlebih dahulu. Aturan yang telah dibuat kemudian diterapkan pada tokoh sehingga mereka mengikuti aturan-aturan tersebut (Beane, 2012). Karim et.al. (2012) menambahkan bahwa animasi prosedural membentuk pergerakan langkah kaki hanya dengan menggunakan formula matematis dan berbagai algoritma, tanpa ada data pergerakan yang diperlukan.

2.4. Scripting

Scripting adalah sebuah perangkat untuk menciptakan alat bantu yang disediakan oleh aplikasi 3D seperti *Autodesk Maya. Scripting* menggunakan salah satu dari berbagai bahasa pemrograman, contohnya adalah C++, Python, Maya Embedded Language (MEL), dan Javascript (Beane, 2012). Bahasa pemrograman yang akan dibahas lebih dalam adalah MEL.

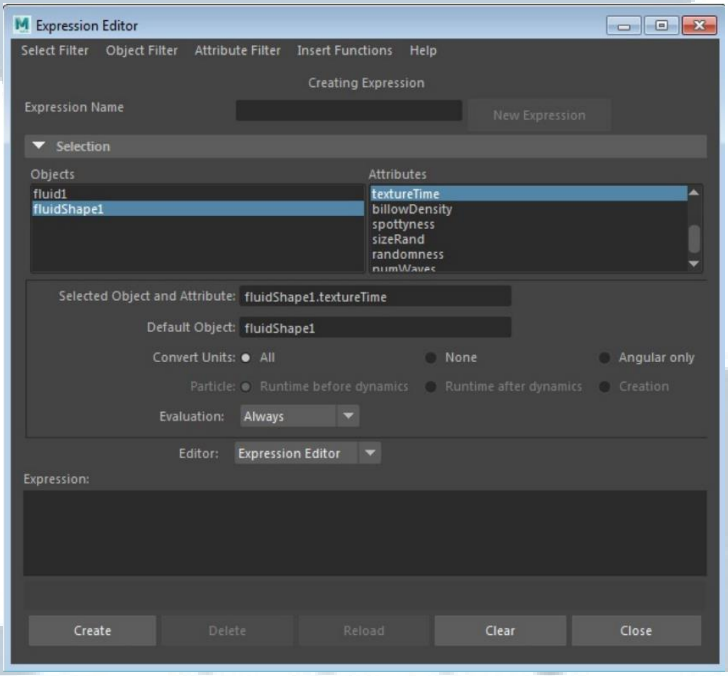
2.4.1. MEL

MEL adalah salah satu bahasa yang digunakan untuk melakukan *scripting* penggunaan program *Autodesk Maya*. *MEL* adalah bahasa pemrograman yang lebih mudah untuk mengendalikan Maya serta dibuat hanya untuk penggunaan *Maya* (Autodesk Knowledge Network, 2014, *"MEL and Expressions")*. *MEL* dapat dijalankan di *menu script editor* ataupun di dalam *menu expression*.

Perancangan Algoritma Animasi..., Yosua, FSD UMN, 2018

2.4.2. *Expressions*

Expressions adalah cara lain untuk menentukan nilai sebuah atribut dengan menggunakan MEL. *Expressions* dapat menggunakan beragam variabel lain untuk mengganti nilai. *Expressions* dapat menjadi salah satu metode untuk menambah kontrol dalam penciptaan *rig*. (Beane, 2012). Dalam membuat pergerakan atau animasi atribut yang bersifat tambahan, acak, ataupun ritmis, *expression* adalah teknik yang ideal untuk melakukan hal tersebut (Autodesk Knowledge Network, 2014*, "Animation Expressions"*).



Gambar 2.2. *Menu Expression Editor*

(Autodesk Maya 2017: A Comprehensive Guide, 2016)

2.4.3. Variabel

Variabel adalah nilai-nilai yang dinyatakan dengan nama yang simbolis, dan digunakan dalam sistem penulisan script MEL. Penamaan variabel dimulai

dengan simbol *dollar* ($), kemudian dilanjutkan dengan nama variabel tersebut. Penamaan variabel hanya boleh menggunakan huruf, angka dan garis bawah (_). Variabel sensitif terhadap kapitalisasi huruf serta tidak boleh ada angka yang mengawali nama sebuah variabel (Autodesk Knowledge Network, 2014, *"MEL and Expressions")*. Contoh: $x, $floaty5000, $longDescriptiveName, $name_with_underscores, $_line.

Menurut Autodesk Knowledge Network (2014) variabel memiliki berbagai jenis nilai salah satunya yaitu *integer*, *strings*, *floating point number* atau *floats*, *array*, *matrix* serta *vector*. Dalam penulisan ini penulis hanya akan menekankan pada dua variabel, yaitu *floats* dan *vector*. *Floats* adalah nilai bilangan desimal, sedangkan *vector* adalah nilai besaran vektor. Variabel dapat dibuat dengan cara menuliskan jenis variabel terlebih dahulu sesuai fungsinya, kemudian memberi nama variabel Contoh: float $param; int $counter; string $name; vector $position.

2.4.4. Fungsi *Expression*

Autodesk Knowledge Network (2014) menyatakan bahwa *expression* memiliki beragam fungsi, namun fungsi *expression* yang akan digunakan dalam penulisan ini hanyalah sebagian kecil, yaitu:

1. *Getattr*

Getattr berfungsi untuk mendapatkan nilai dari kondisi sebuah atribut. Fungsi tambahan yaitu –*time*(–t) berfungsi untuk mendapatkan nilai sebuah atribut pada waktu yang telah ditentukan, bukan dari waktu pada saat ini.

Perancangan Algoritma Animasi..., Yosua, FSD UMN, 2018

2. *Mag*

Mag berfungsi untuk menghitung panjang dari jarak *vector*. Nilai hasil dari fungsi mag berupa *float*. Contoh:

```
mag <<7,8,9>>;
```

Maka hasilnya adalah 13.9284

3. *If … else if …*

if… else if… ini berfungsi sebagai fungsi logika. *If…else if…* ini akan membuat keputusan sesuai dengan kondisi yang ditetapkan oleh pembuat. Contoh:

```
if($x > 10){print ("It's greater than 10!");}
```

Maka hasilnya adalah "It's greater than 10!" jika $x mencapai nilai lebih dari 10. Tidak akan ada yang terjadi bila nilai $x adalah 10 atau lebih kecil.

4. Abs

Abs berfungsi untuk memberikan nilai absolut atau nilai mutlak kepada suatu variabel. fungsi abs akan menghilangkan tanda negatif yang terdapat pada variabel *int*, *float* atau *vector*. Contoh:

```
abs (-1);
```

Karena nilai -1 adalah nilai negatif maka hasil fungsi abs akan menjadi positif (1).

```
abs(<<-1,-2.43,555>>);
```

setiap variabel yang terdapat pada variabel vector akan menjadi positif, yaitu menjadi <<1, 2.43, 555>>.

5. *Clamp*

Clamp berfungsi untuk memberi batas nilai dari sebuah variabel. Jika nilai melebihi batas maksimum, maka nilai akhir adalah batas maksimum, jika lebih

rendah dari batas minimum maka yang diambil adalah nilai minimum. Fungsi *clamp* ditulis sebagai berikut:

```
Clamp(float minnumber, float maxnumber, float
parameter)
```

Hasil nilai variabel *clamp* akan berupa *float*.

Contoh fungsi *clamp*:

```
Clamp(4,6,22);
```

Karena nilai dari 22 lebih besar dari batas maksimum yaitu 6, maka hasil fungsi *clamp* adalah 6.

```
Clamp(4,6,2);
```

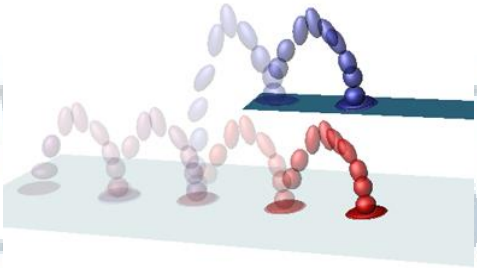
Karena nilai 2 lebih kecil dari batas minimum yaitu 4, maka hasil fungsi *clamp* adalah 4.

```
Clamp(4,6,5);
```

Karena nilai 5 berada dalam jarak aman, maka hasil fungsi *clamp* adalah 5.

2.5. Animation Layer

Animation layer adalah fitur pada Autodesk Maya untuk membuat animasi bertingkat. *Animation layer* berfungsi untuk membuat animasi *keyframe* diatas animasi *keyframe* yang sudah ada tanpa merusak kurva animasi aslinya.



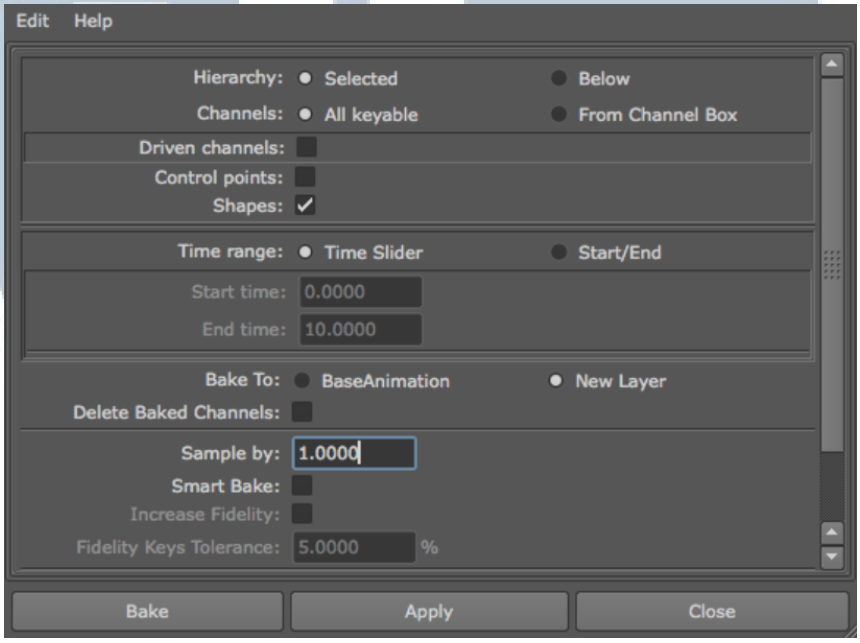
Gambar 2.3. Ilustrasi *Animation Layer*

(Autodesk Knowledge Network, 2015)

Perancangan Algoritma Animasi..., Yosua, FSD UMN, 2018

2.6. *Bake Simulation*

Bake simulation adalah proses mematangkan atau mengunci pergerakan yang sudah dibuat. Pergerakan yang dinamis dapat dikunci pergerakannya dalam bentuk *keyframe* sehingga lebih mudah untuk diubah. (Derakhshani, 2015).



Gambar 2.4. *Menu Bake Simulation*
(Autodesk Knowledge Network, 2015)

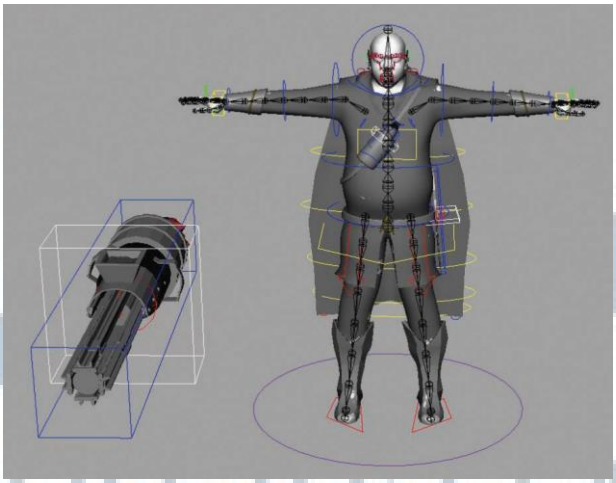
Keterangan:

Time Range : Jarak waktu untuk memproses *bake simulation*

Bake to : *Animation Layer* tujuan *bake simulation*. *Menu new layer* akan menghasilkan animasi *bake simulation* yang tidak merusak kurva animasi aslinya serta tidak merusak fungsi *expression* yang terdapat pada atribut tersebut.

2.7. Tahapan *Rigging*

Proses *rigging* adalah proses yang dilakukan tanpa adanya aturan baku yang mengikatnya; Beragam cara dapat dilakukan untuk mendapatkan hasil perancangan yang serupa. Dalam perancangan *rig*, tidak semua proses perlu dilakukan, hanya tahap yang penting dan efisien yang digunakan. Dunlop (2014) menyatakan bahwa proses *rigging* terbagi dalam tiga tahap yaitu penciptaan sistem kerangka, pembuatan "*puppet rig"*, pengaturan hubungan antara rangka dengan objek. Penciptaan sistem kerangka sangat beragam tergantung dari kebutuhan kontrol dari tokoh atau objek yang akan digerakkan, namun keragaman *rig* tersebut didasari oleh tiga komponen, yaitu *bones*, *kinematics*, dan *constraints.* Tickoo (2016) melengkapi ketiga tersebut dengan komponen keempat, yaitu *Parenting.*

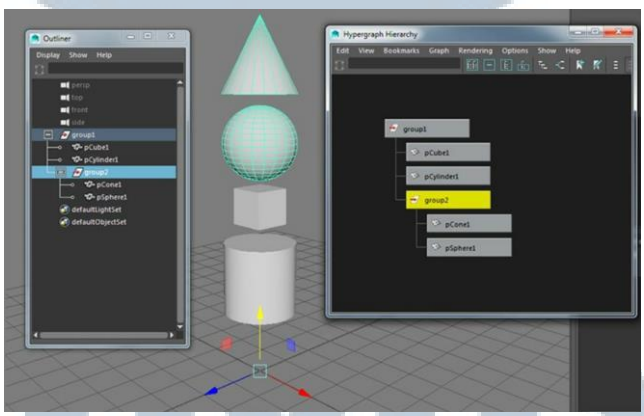


Gambar 2.5. Contoh *Rig* senjata (kiri) dan *rig* manusia (kanan) dalam *Maya* (*3D Animation Essentials*, 2012)

UNIVERSITAS MULTIMEDIA NUSANTARA

2.7.1. *Scene Hierarchy / Parenting*

Scene hierarchy atau parenting adalah hubungan bertingkat antarobjek. Transformasi dua objek atau lebih dapat dihubungkan dengan sistem *parenting*, dimana objek yang pertama dipilih akan menjadi *child* dan objek kedua menjadi *parent*. Dalam *parenting,* objek *parent* akan mengaplikasikan beberapa perubahan atribut yang terjadi padanya kepada objek *child*, seperti atribut *position, scale, rotation* dan *visibility* (Palamar, 2015, hlm. 15). Assaf (2016) menambahkan bahwa setelah melakukan tahap *parenting,* objek *child* tetap memiliki kebebasan untuk melakukan berbagai trasformasi lainnya dan tidak terikat penuh pada objek *parent*(hal. 82).



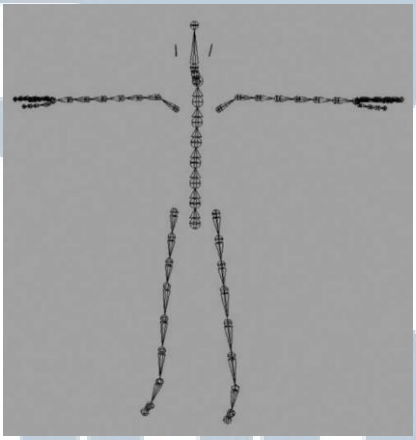
Gambar 2.6. Tampilan *Menu Outliner*(kiri) dan *Menu Hypergraph*(kanan)
(Introducing Autodesk Maya 2016, 2015)

Menurut Tickoo (2016), ada ketentuan dalam melakukan hubungan *parenting*, yaitu *child* tidak dapat memiliki lebih dari satu *parent* namun satu *parent* dapat memiliki lebih dari satu objek *child.* Derakshani (2015) menambahkan bahwa *hierarchy* dapat ditampilkan dengan berbagai *menu* Maya

seperti *outliner*, *hypergraph* dan sebagainya. Hubungan *hierarchy* terlihat dari tampilan *outliner* atau *hypergraph* dimana *child* akan terlihat sedikit indentasi dibandingkan dengan *parent*.

2.7.2. Sistem Kerangka

Menurut Tickoo (2016), Tubuh makhluk hidup umumnya memiliki kerangka yang terstruktur untuk dapat bergerak, demikian pula dengan objek atau tokoh dalam animasi 3D. Objek atau tokoh 3D yang memiliki struktur tubuh menyerupai makhluk hidup dirancang dengan sistem kerangka yang serupa (chap.9).



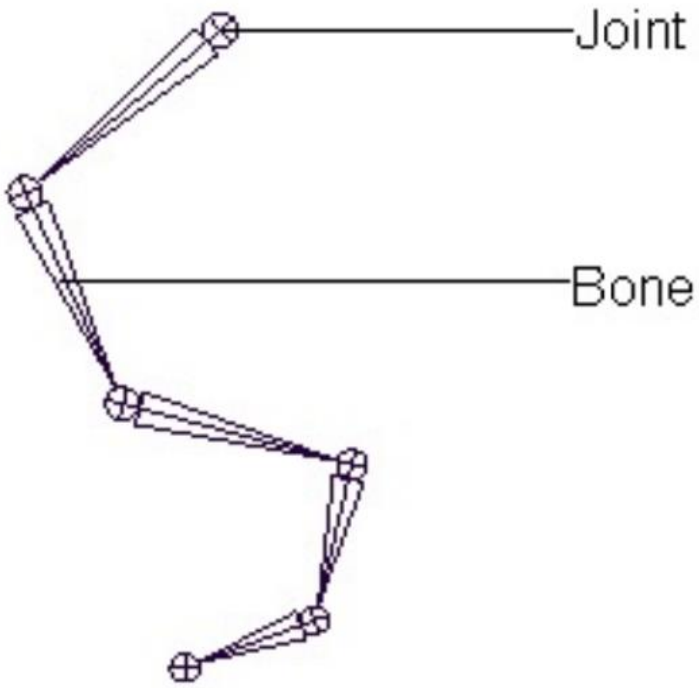
Gambar 2.7. Contoh dari Sebuah Sistem Kerangka
(*3D Animation Essentials*, 2012)

Walaupun teknik kerangka lebih efisien untuk perancangan tokoh, teknik kerangka juga berfungsi sebagai alat bantu gerak benda mati (Derakhshani, 2015). Kerangka makhluk hidup terdiri dari tulang dan sendi, demikian pula dengan kerangka *rig* juga terdiri dari dua elemen yang serupa yaitu *bones* dan *joints* yang dapat dihubungkan dengan *kinematics*. Assaf (2016) menambahkan bahwa kerangka yang baik disusun oleh *joint* dengan orientasi yang searah.

Perancangan Algoritma Animasi..., Yosua, FSD UMN, 2018

2.7.3. *Bone* dan *Joint*

Bones adalah struktur kaku yang menghubungkan antara *joints* dengan *joints* lainnya. *Joints* adalah titik artikulasi yang memampukan tulang untuk bergerak. Menurut dokumentasi *online* dari *Autodesk Knowledge Network* (2014), *joints* memiliki tiga macam pergerakan sesuai dengan jumlah kemampuan rotasi. Jenis *joint* yang pertama adalah *ball joint* dimana *joint* dapat perputar dalam tiga sumbu menyerupai sendi peluru pada pangkal paha manusia. Jenis yang kedua adalah *universal joint* dimana *joint* dapat berputar dalam dua sumbu seperti sendi pergelangan tangan. Jenis yang ketiga adalah *hinge joint* dimana *joint* hanya dapat berputar pada satu sumbu seperti sendi pada lutut.



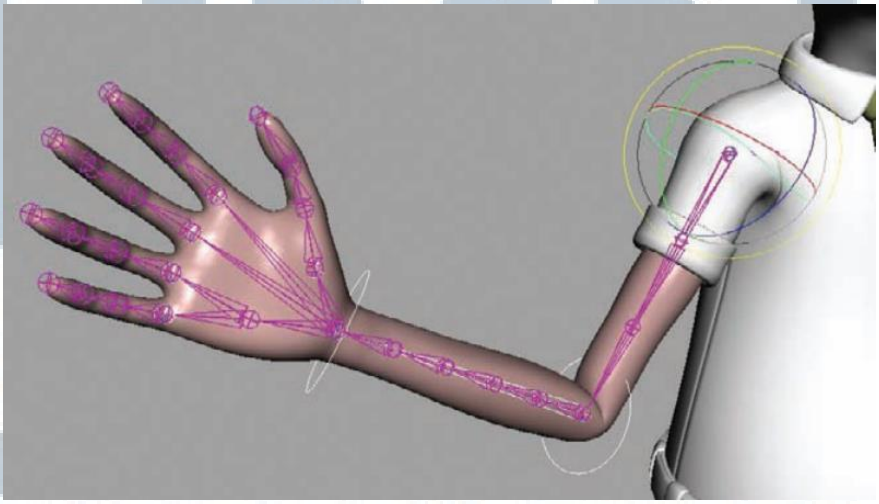
Gambar 2.8. Contoh dari *Bone* dan *Joint*
(Autodesk Maya 2017: A Comprehensive Guide, 2016)

2.7.4. *Kinematics*

Menurut Derakhshani (2015), *kinematics* adalah jenis pergerakan yang digunakan dalam sistem kerangka. *Kinematics* memiliki dua metode, yaitu metode *scene hierarchy* bernama *Forward Kinematics* (FK), atau dengan metode *Inverse Kinematics* (IK). Beane (2012) menambahkan bahwa pengguna dapat mengganti jenis tersebut dengan IK saat dibutuhkan, karena kedua metode bekerja untuk menghasilkan kebutuhan animasi yang berbeda.

a. FK

Beane (2012) menyatakan bahwa FK adalah jenis pergerakan dengan melakukan rotasi pada setiap *joint* sesuai dengan urutan *scene hierarchy*. Contoh yang dapat diambil dari jenis pergerakan FK adalah pergerakan yang dilakukan oleh lengan. Dalam FK, *animator* harus melakukan rotasi di setiap sendi untuk menghasilkan gerakan yang diinginkan, seperti bahu, siku dan pergeralangan tangan (hal. 185).



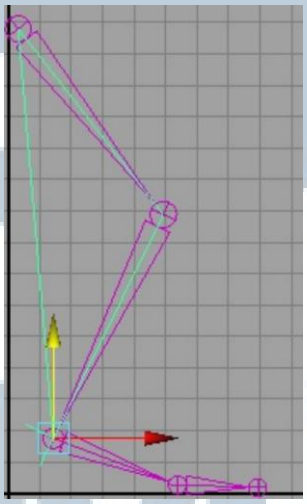
Gambar 2.9. Contoh dari FK

(*3D Animation Essentials*, 2012)

Perancangan Algoritma Animasi..., Yosua, FSD UMN, 2018

b. IK

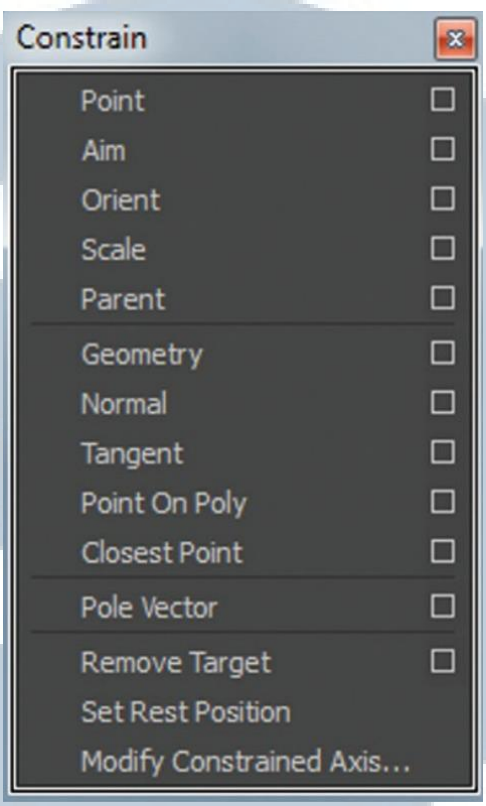
Menurut Beane (2012), IK menentukan pergerakan dengan cara yang berbeda. IK menentukan pergerakan dan rotasi *joints* berdasarkan *scene hierarchy* yang terbalik. IK menentukan titik akhir dari sebuah sistem kerangka, kemudian mengatur pergerakan kerangka sesuai jarak yang telah ditentukan. Derakhshani (2015) berpendapat bahwa metode IK baik digunakan untuk pergerakan kaki karena IK memudahkan kontrol telapak kaki agar dapat berpijak dan tidak bergeser.



Gambar 2.10. Contoh dari Sebuah IK
(Autodesk Maya 2017: A Comprehensive Guide, 2016)

2.7.5. *Constraint*

Constraint adalah metode yang menghubungkan antara gerakan suatu objek dengan objek sumber. Hubungan antara kedua objek dapat berupa atribut *translate, rotate,* atau *scale* (Derakhshani, 2015). Tickoo (2016) menambahkan bahwa dalam Autodesk Maya terdapat 11 jenis *constraint*, yaitu:



Gambar 2.11. *Menu Constraint*

(“*Rigging for Games*”, 2016)

a. *Parent Constraint* :*Parent constraint* adalah *constraint* dengan fungsi yang menyerupai teknik *parenting* tanpa adanya hubungan langsung *parenting* dari objek tersebut. *Parent constraint* membatasi atribut translasi dan rotasi dari objek ke objek lain.

b. *Point Constraint* :*Point constraint* adalah *constraint* dengan fungsi mengikat atribut translasi sesuai dengan objek.

UNIVERSITAS MULTIMEDIA NUSANTARA

c. *Aim Constraint* :*Aim constraint* membatasi orientasi objek seakan melihat ke arah objek yang dituju.

d. *Orient Constraint* :*Orient constraint* meluruskan rotasi objek sejajar dengan orientasi objek sumber.

e. *Scale Constraint* :*Scale constraint* mengikat skala objek dan menyesuaikan dengan skala objek sumber.

f. *Geometry Constraint* :*Geometry constraint* melekatkan pivot point objek pada permukaan objek sumber.

g. *Normal Constraint* :*Normal constraint* mengikat rotasi objek dengan membuat rotasi tegak lurus dengan permukaan dari objek sumber.

h. *Pole Vector Constraint* :*Pole vector constraint* m engikat arah *pole vector* yang dimiliki objek mengikuti arah objek lainnya.

i. *Point on Poly Constraint* :*Point on poly constraint* mengikat objek dengan satu titik *vertex* objek lainnya.

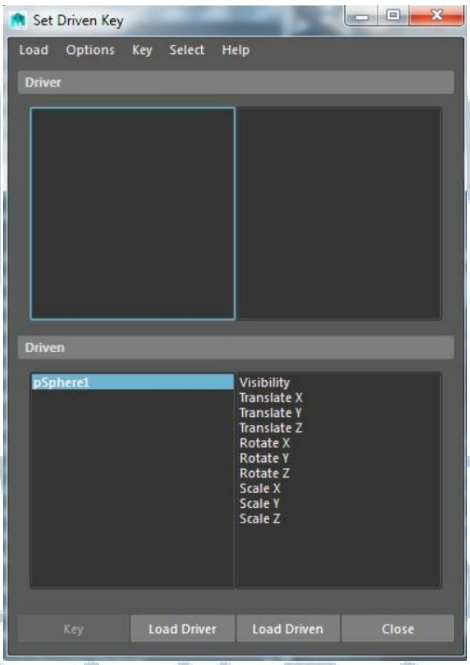
j. *Closest Point Constraint* :*Closest point constraint* adalah sebuah perhitungan untuk mencari titik terdekat dari permukaan geometri.

k. *Tangent Constraint* :*Tangent constraint* mengikat orientasi objek dan mengarahkan orientasi tersebut kearah *curve.*

Perancangan Algoritma Animasi..., Yosua, FSD UMN, 2018

2.7.6. *Set Driven Key*

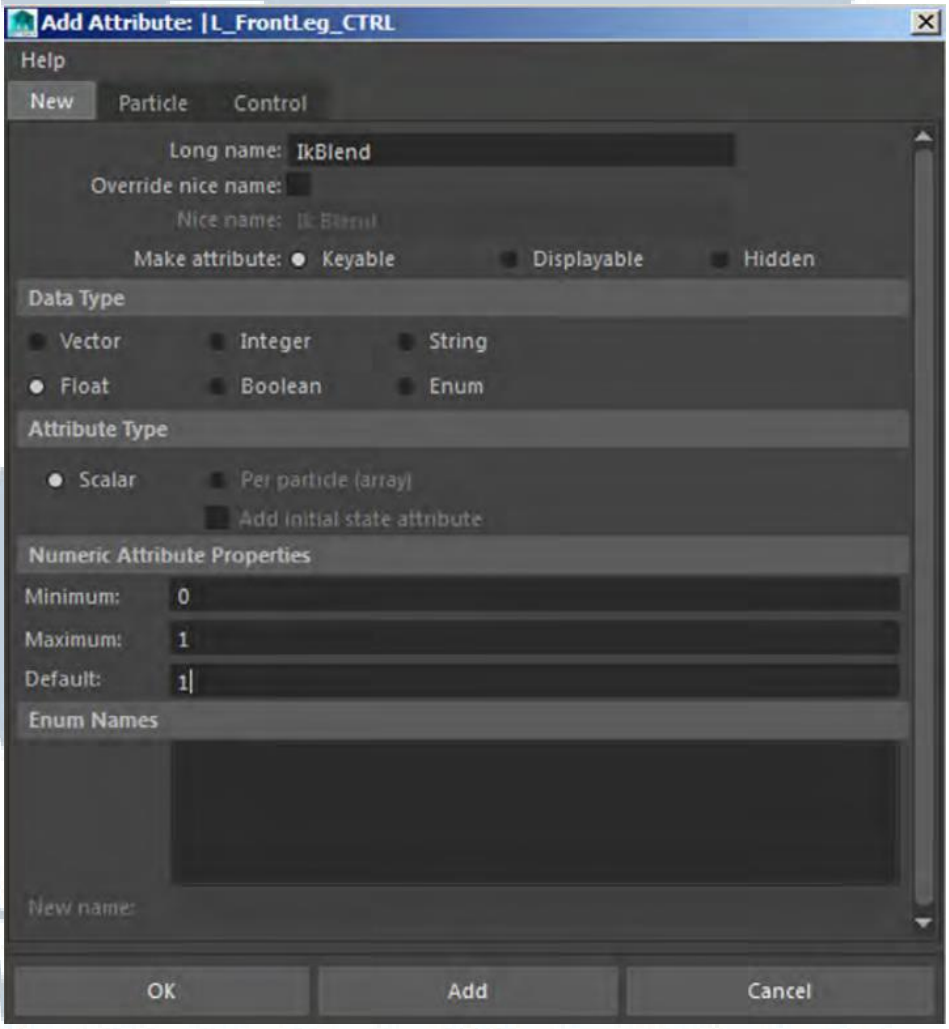
Menurut Tickoo (2016), *set driven key* adalah cara untuk mengontrol suatu atribut objek melalui atribut objek lainnya, dimana objek penggerak disebut *driver* dan objek yang digerakkan disebut sebagai *driven*. Sebuah objek dapat menjadi *driver* dan *driven* sekaligus, dimana terdapat atribut yang saling mempengaruhi. *Set driven key* bekerja dengan menyesuaikan nilai yang terdapat dalam masing-masing atribut. Perubahan nilai atribut *driven* akan terjadi berdasarkan nilai atribut *driver* yang sudah ditentukan. Derakhshani (2015) menambahkan bahwa *Set driven key* dapat dilakukan dengan atribut berjumlah lebih dari dua, sebuah atribut dapat mengontrol lebih dari satu atribut lainnya. Atribut yang terhubung dalam *set driven key* tidak hanya sebatas atribut sejenis, namun dapat mengatur atribut yang berjenis lain.



Gambar 2.12. *Menu Set Driven Key*
(Autodesk Maya 2017: A Comprehensive Guide, 2016)

2.7.7. *Custom Attributes*

Custom attributes adalah atribut yang dibuat untuk mempermudah pengaturan berbagai atribut lain. *Custom attributes* digunakan bersamaan dengan *set driven key* untuk mempermudah menggerakkan *rig* serta memberikan fleksibilitas bagi *animator* (Palamar, 2015, hlm. 225). *Custom Attributes* juga dapat membantu perancangan algoritma *expression* bila digunakan sebagai variabel dalam algoritma tersebut.



Gambar 2.13. *Menu Add Attributes*

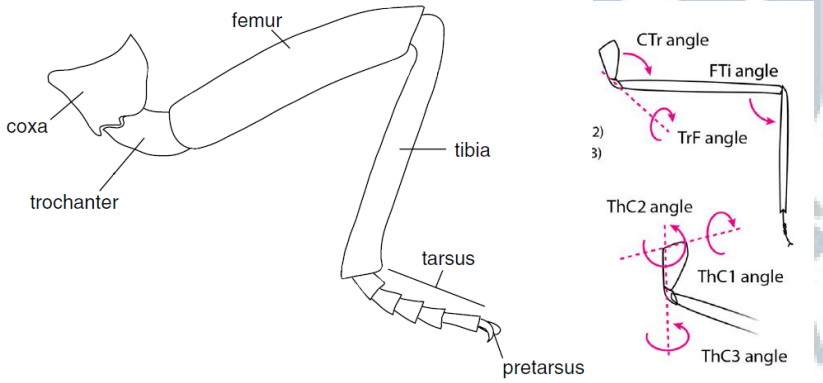
(Mastering Autodesk Maya 2016: Autodesk Official Press, 2015)

2.8. Belalang Sembah

Capinera (2008) mendefinisikan Belalang sembah merupakan serangga dengan ciri khusus dimana dua kaki depan memiliki bentuk dan fungsi yang unik, bentuk tubuh yang memanjang serta gerakan kepala yang fleksibel. Bila kedua kaki depan dilipat, maka penampilannya terkesan seperti sedang berdoa. Kedua kaki depan tersebut digunakan untuk menangkap mangsa dengan cepat. Kebanyakan dari belalang sembah memiliki sayap, namun ada jenis belalang sembah yang tidak memiliki sayap (hal. 3034).

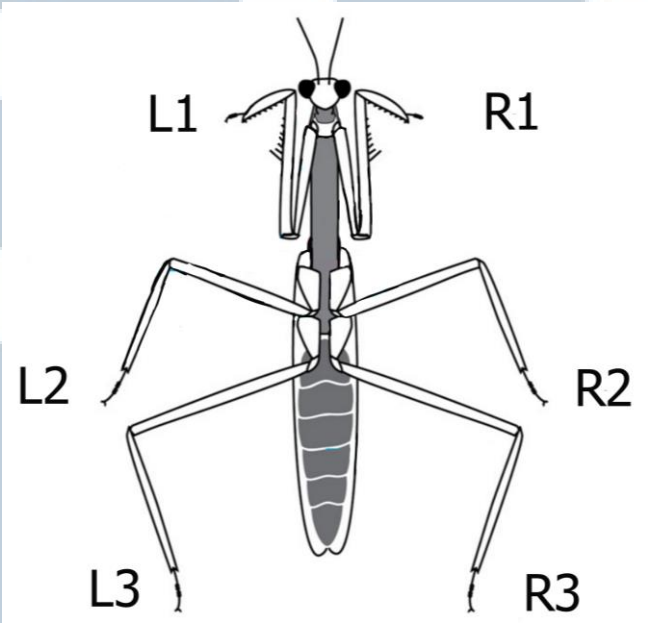
2.8.1. Anatomi Kaki dan Sendi Belalang Sembah

Anatomi kaki belalang sembah terdiri dari enam segmen yaitu *coxa*, *trochanter*, *femur*, *tibia*, *tarsus* dan *pretarsus* dan diantara segmen tersebut terletak berbagai macam sendi. Belalang sembah memiliki enam buah sendi engsel pada setiap kaki dan tiga buah sendi engsel pada bagian badannya (Szczecinski, 2015). Chapman (1998) juga menyatakan bahwa serangga secara umum memiliki anatomi kaki yang serupa.



Gambar 2.14. Segmen Kaki serta Sendi Belalang Sembah *Tenodera Sinensis.*
(Kiri: (Chapman, 1998) kanan: (Szczecinski, 2015))

2.8.2. Cara Berjalan Belalang Sembah



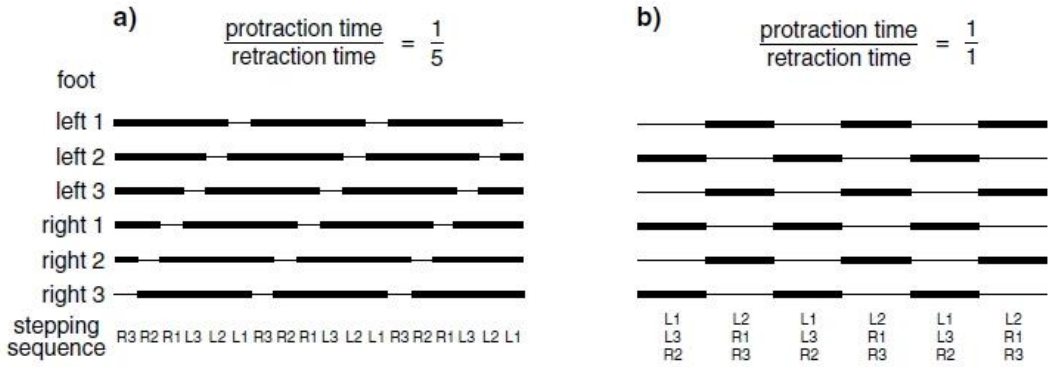
Gambar 2.15. Ilustrasi Belalang Sembah
(Szczecinski, 2015)

Wigglesworth dan Vincent (1982) berpendapat bahwa belalang sembah berjalan menggunakan keenam kakinya dengan cara berjalan secara menyilang. Wigglesworth dan Vincent menambahkan bila kedua kaki tengah diamputasi maka belalang sembah akan berjalan dengan keempat kaki belakangnya. Belalang sembah melakukan langkah awal dengan menggunakan kaki kiri depan dan kaki kanan belakang, kemudian melanjutkan langkah berikutnya dengan kaki kanan depan dan kaki kiri belakang (hal.193).

Chapman (1998) menambahkan bahwa serangga akan memiliki pola berjalan yang berbeda berdasarkan kecepatan berjalannya. Bila serangga berjalan dengan lambat maka serangga tersebut akan berjalan dengan pola R3 R2 R1 L3 L2 L1 R3 dan seterusnya seperti pada diagram sebelah kiri (a). Namun bila

serangga berjalan dengan cepat maka R3 L2 R1 akan mengangkat hampir bersamaan, lalu bergantian dengan L3 R2 L1 sesuai diagram sebelah kanan (b) (hal 164).

Keterangan: *protraction time* adalah waktu kaki mengangkat mengayun melangkah, sedangkan *retraction time* adalah waktu kaki berada di tanah dan bergerak kebelakang relatif dengan pergerakan badan.



Gambar 2.16. Diagram Cara Berjalan Belalang Sembah

(The Insect Structure and Function, 1998)