**BAB 2**

**TINJAUAN PUSTAKA**

**2.1 Pusat Massa**

Pusat massa merupakan posisi rata-rata dari sebuah atau sekumpulan objek berdasarkan massanya. Pusat massa sering disamakan dengan istilah pusat gravitasi. Dalam ilmu fisika, pusat massa umumnya digunakan untuk menyederhanakan persamaan gerak seperti momentum sudut dan momen inersia.

Pusat massa dapat terletak di dalam maupun di luar objek. Posisi ini merupakan titik dimana benda terpengaruhi apabila benda tersebut diberikan gaya. Jika kita mendorong sebuah objek tepat di titik pusat massanya, maka objek tersebut hanya berpindah dan tidak akan berputar dalam sumbu manapun.

Kalkulasi letak pusat massa suatu sistem dalam suatu sumbu dapat ditentukan dengan melakukan pembagian antara jumlah hasil perkalian titik sumbu dan massa tiap benda dengan total massa :

Dimana merupakan massa *segment*, merupakan pusat massa *segment* di sumbu X, merupakan massa total benda, dan merupakan pusat massa sistem di sumbu X.



Gambar 2.1 Sistem objek dalam suatu diagram kartesian (sumber : <https://cdn.kastatic.org/ka-perseus-images/4cf588bbf9e241a4c6c8c13ab9c6eb582eca6f37.svg>)

**2.2 Pusat Massa Segment Tubuh Manusia**

Pada penelitian *R. A. Clark, A. L. Bryant, Y. Pua, P. McCrory, K. Bennell, and M. Hunt. “Validity and reliability of the Nintendo Wii balance board for assessment of standing balance”*, bahwa pusat massa *segment* umumnya dihitung dengan mencari persentase dari panjang *proximal end*. Sedangkan massa *segment* umumnya dihitung dengan mencari persentase dari total massa manusia.

Tabel 2.1 Persentase Massa Segment terhadap Massa Total

|  |  |  |
| --- | --- | --- |
| Segment Mass Percents: | | |
| Segment | Males | Females |
| Head & Neck | 6.94 | 6.68 |
| Trunk | 43.46 | 42.58 |
| Upper Arm | 2.71 | 2.55 |
| Fore Arm | 1.62 | 1.38 |
| Hand | 0.61 | 0.56 |
| Thigh | 14.16 | 14.78 |
| Shank | 4.33 | 4.81 |
| Foot | 1.37 | 1.29 |

Tabel 2.2 Persentase Panjang Segment terhadap Proximal End

|  |  |  |  |
| --- | --- | --- | --- |
| Segment Length Percents: | | | |
| Segment | Males | Females | End Points |
| Head & Neck | 50.02 | 48.41 | Top Of Head –C7 |
| Trunk | 43.10 | 37.82 | MidS - MidH |
| Upper Arm | 57.72 | 57.54 | SJC - EJC |
| Fore Arm | 45.74 | 45.59 | EJC - WJC |
| Hand | 79.00 | 74.74 | WJC - MCPIII |
| Thigh | 40.95 | 36.12 | HJC - KJC |
| Shank | 43.95 | 43.52 | KJC - AJC |
| Foot | 44.15 | 40.14 | Heel - Toe |

**2.3 BioVision Hierarchy Data**

*Format file* BVH merupakan ekstensi file yang berisi *hierachical strcuture* dan *motion data*. Sebuah *file* BVH dibagi kedalam dua bagian, bagian pertama diawali dengan *header* “HIERARCHY”, sedangkan bagian kedua ditandai dengan *header* “MOTION”.

Bagian HIERARCHY mendefinisikan satu atau lebih *skeleton* secara rekursif. Setiap *skeleton* diawali dengan satu *ROOT* yang merupakan akar dari *skeleton*. Kemudian diikuti dengan satu atau lebih *JOINT* yang merupakan tungkai. Diakhiri dengan *End Site* yang merupakan representasi panjang dari *JOINT* terakhir. Setiap bagian dibatasi dengan kurung kurawal. Apabila terdapat *JOINT* didalamnya, maka *JOINT* tersebut merupakan *child*.

Pendefinisian *ROOT* dan *JOINT* diikuti dengan nama dan berisi data tentang *OFFSET, CHANNELS,* dan *children*. *End Site* hanya memiliki *OFFSET* karena tidak akan dilakukan transformasi. *OFFSET* merupakan jarak relatif suatu *JOINT* terhadap *parent*-nya. *CHANNELS* merupakan urutan data berupa translasi kartesian dan rotasi *euler* yang akan didapatkan dari bagian *MOTION*.

Bagian *MOTION* berisi *Frames* (jumlah *frame* total), *Frame Time* (selisih waktu setiap *motion* data dalam satuan detik), diikuti dengan baris-baris *motion* *data*. Satu baris *motion data* merupakan isi dari *CHANNELS*. Setiap *motion data* dari kiri ke kanan di *mapping* kedalam *channels* dari atas ke bawah.



Gambar 2.2 Contoh Hierarki Skeleton (sumber : http://mocappys.com/wp-content/uploads/2014/03/BaseJointNames.jpg)

**2.4 Transformasi Matrix**

Dalam aljabar linier, untuk menentukan transformasi pada 3 dimensi diperlukan *matrix 4x4 homogenous coordinate*. Matriks ini mampu merepresentasikan translasi, rotasi, dilatasi dan proyeksi. Matriks T merupakan matriks transformasi translasi. Matriks R dibagi menjadi 3 (Rx, Ry, dan Rz) yang masing-masing merupakan matriks rotasi pada sumbu tersendiri. Dan Matriks S merupakan matriks *scaling* / dilatasi.

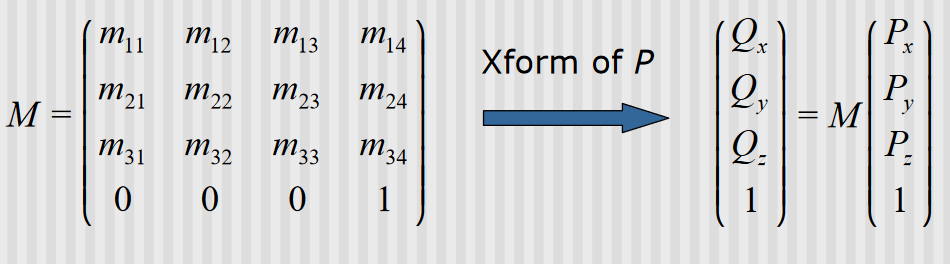
Dalam penelitian ini, digunakan *Column Major Matrices* dan *Right Handed Coordinate System. Column Major Matrices* menganggap kolom pertama adalah transformasi pada sumbu X, kolum kedua adalah transformasi pada sumbu Y, kolom ketiga adalah transformasi pada sumbu Z, dan kolom keempat merupakan vektor translasi. Perkalian matriks pada *Column Major Matrices* dilakukan dengan cara menempatkan matriks pertama pada posisi paling kanan sampai. Kemudian diikuti dengan matriks-matriks selanjutkan ke arah kiri. Contohnya ketika ingin melakukan translasi diikuti dengan rotasi maka notasi matriksnya adalah M = RT.

*Right Handed Coordinate System* merupakan cara menafsirkan kemana arah sumbu Z. Apabila kita menggunakan jempol kanan sebagai sumbu X, jari telunjuk kanan sebagai sumbu Y, maka jari tengah akan menunjukan arah positif dari sumbu Z. Perbedaan Left Handed Coordinate System dan Right Handed Coordinate System terletak pada arah positif sumbu Z yang masing-masing saling membelakangi.



Gambar 2.2 Left Handed Coordinates dan Right Handed Coordinates (sumber : <https://upload.wikimedia.org/wikipedia/commons/thumb/b/b2/3D_Cartesian_Coodinate_Handedness.jpg/220px-3D_Cartesian_Coodinate_Handedness.jpg>)

Pada *affine transformation*, *subscripts* m11, m12, m13, m21, m22, m23, m31, m32 dan m33 mempengaruhi transformasi rotasi dan dilatasi, sedangkan *subscripts* m14, m24, dan m34 mempengaruhi translasi. Dikarenakan perkalian pada matriks tidak bersifat komutatif (TRS ≠ SRT ) urutan transformasi dilakukan dari dilatasi, kemudian diikuti oleh rotasi, dan kemudian di akhiri dengan translasi. Secara notasi rumusan dapat ditulis M = TRS.



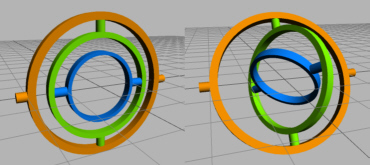
Gambar 2.3 Pengaplikasian transformasi matrix 4x4 dalam 3D

**2.5 Euler Rotate Order**

Pada transformasi 3D menggunakan matrix, terdapat order rotasi yang biasanya menjadi konvensi transformasi dalam suatu sistem. Konvensi ini disebut *Euler Rotate Order*. Pada *format file BVH* sebagaimana yang digunakan dalam penulisan ini, *rotate order* yang digunakan adalah *ZXY*. *ZXY rotate order* berarti transformasi rotasi dilakukan 3 kali, diawali dengan rotasi di sumbu Z, kemudian rotasi di sumbu X, dan diakhiri rotasi di sumbu Y.

Pada umumnya, *rotate order* akan selalu konsisten untuk mencegah *gimbal lock*. *Gimbal lock* merupakan keadaan dimana salah satu sumbu putar sama dengan sumbu putar lainnya. Keadaan ini mengakibatkan dua sumbu putar menghasilan hasil perputaran yang sama sehingga transformasi rotasi tiga dimensi tidak dimungkinkan.

Untuk menghindari *gimbal lock*, dapat dilakukan dengan menggunakan notasi *quaternions* dalam melakukan transformasi rotasi. Akan tetapi pada penelitian ini, semua rotasi telah didefinisikan dengan *rotate order* Rz, Rx, Ry maka tidak perlu menggunakan *quaternions* untuk mendapatkan rotasi yang konsisten.



Gambar 2.4 Gimbal Lock Sumber (sumber : https://i.stack.imgur.com/SRwlZ.jpg)

**2.6 Transformation Inheritance**

*Transformation Inheritance* merupakan keadaan dimana terdapat relasi *parent-child* antara dua atau lebih objek. Transformasi yang dilakukan kepada *parent* akan berdampak pada *children*-nya. Sedangkan transformasi yang dilakukan kepada *child* tidak akan mempengaruhi *parent*-nya. *Hierarchial Skeleton* merupakan salah satu contoh pemanfaatan *transformation inheritance*.

Setiap objek yang dihubungkan dengan relasi *parent-child* masing-masing memiliki *local transform* dan *global transform*. *Local* *transform* merupakan transformasi relatif terhadap parent, sedangkan *global* *transform* merupakan transformasi relatif terhadap titik *origin*.

Dibutuhkan global transform dari semua *segments* untuk menggambarkan *Hierarchial Skeleton* secara keseluruhan. Penghitungan *global transform* dapat dilakukan dengan *pre-multiplication* daripada joint yang ingin dicari dengan *parent-parent*-nya sampai menuju *root*:

Mleftfootglobal = Mhips.Mleftupleg.Mleftlowleg.Mleftfootlocal

**2.7 Model World Camera Screen Space**

*Model World Camera Screen Space* merupakan istilah yang umum digunakan dalam membuat aplikasi tiga dimensi dalam komputer grafik. *Model View Projection Matrices* diperlukan karena keterbatasan monitor dalam merepresentasikan sebuah dunia digital tiga dimensi. Penggambaran monitor hanya mampu mencakup luasan daerah bujur sangkar dua dimensi.

*Model space* merupakan posisi relatif setiap titik terhadap pusat *origin* *model*. Penggunaan *model space* umumnya untuk mendefinisikan bentuk-bentuk yang diinginkan sehingga saat menempatkannya kedalam *world space* objek tersebut memiliki kesan solid. Dalam penulisan ini, setiap *segment* hanya memiliki dua titik dikarenakan pada program yang dibuat hanya menggambarkan *segment* dalam bentuk garis.

*World space* merupakan posisi relatif setiap model terhadap pusat *origin* *world space*. *World space* umumnya digunakan untuk menempatkan beberapa *model* agar memberi kesan terbentuknya sebuah populasi. Dalam penulisan ini, hanya terdapat satu *instance* dari skeleton yang digunakan.

*Camera space* merupakan posisi relatif setiap model yang berada di *world space* terhadap *camera virtual*. Simulasi *camera virtual* dapat dilakukan dengan cara melakukan perkalian dengan *inverse* matriks yang diinginkan. Maksudnya adalah untuk menggerakkan *camera virtual* ke kiri sama saja dengan menggerakan setiap model yang ada di *world space* ke kanan.

*Screen space* merupakan posisi relatif setiap model yang berada di *world space* yang hanya disorot oleh *camera virtual* terhadap posisi *pixel* di layar *monitor*. Setiap *monitor* memiliki resolusi dan ukuran yang berbeda-beda sehingga dibuatlah sebuah istilah yaitu *Normalized Device Coordinate*s (NDC) dimana sisi paling kiri bernilai -1 dan sisi paling kanan bernilai 1. Begitupun juga dengan sisi paling atas bernilai 1 dan sisi paling bawah bernilai -1. NDC kemudian diperbaiki dengan mengalikan setiap posisi pada *screen space* dengan aspect ratio pada masing-masing *monitor*.

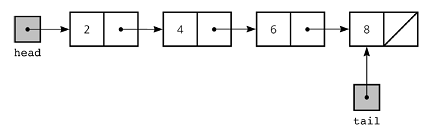
**2.9 Linked List**

*Linked list* merupakan sebuah struktur data *linear* dimana setiap elemen juga merupakan objek pemisah. Sebuah *linked list* umumnya memiliki satu *head* yang merupakan awal dan satu atau lebih *tail* yang merupakan akhir. Setiap objek dalam *linked list* berisi data dan sebuah *pointer* ke objek lainnya. Sebuah *linked* *list* berakhir ketika mencapai *tail* dimana *pointer* pada objek bernilai *null*.

Keuntungan menggunakan linked list meliputi:

1. Ukuran *linked* *list* dapat ditentukan saat *run-time*
2. Operasi penambahan, penyisipan, penghapusan pada *linked list* sangat mudah karena hanya dengan merubah *pointer*
3. Elemen dari sebuah *linked list* dapat disebar ke memori sehingga memungkinkan *program* berjalan pada spesifikasi rendah
4. Pengaksesan menggunakan *pointer* memungkinkan *data* selalu terupdate

Relasi *parent-child* dalam sebuah *hierachial skeleton* umumnya dibuat dalam bentuk *linked list*. Didalam sebuah *file BVH, ROOT* dapat merepresentasikan *head* dan *End Site* dapat merepresentasikan *tail*. Sehingga dalam penelitian ini, *linked list* merupakan struktur *data* yang digunakan dalam menyimpan, memproses, dan menggambarkan *skeleton*.



Gambar 2.5 Contoh sebuah Linked List (sumber : <https://s3-us-west-2.amazonaws.com/ib-assessment-tests/problem_images/singly-ll.png>)