**Steganografi 3D**

**Pemahaman Dasar**

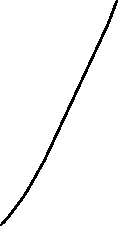
Setiap object 3D memiliki beberapa atribut utama, seperti yang kita tahu sebuah kubus memiliki sudut, rusuk dan sisi atau biasa disebut dengan vertex, edge dan face.

Setiap atribut utama tersebut memiliki satu titik koordinat yang terdiri dari sumbu x, y dan z. misalnya sebuah kubus memiliki sudut sudut betikut :

* A (-1, -1, -1)
* B (1, -1, -1)

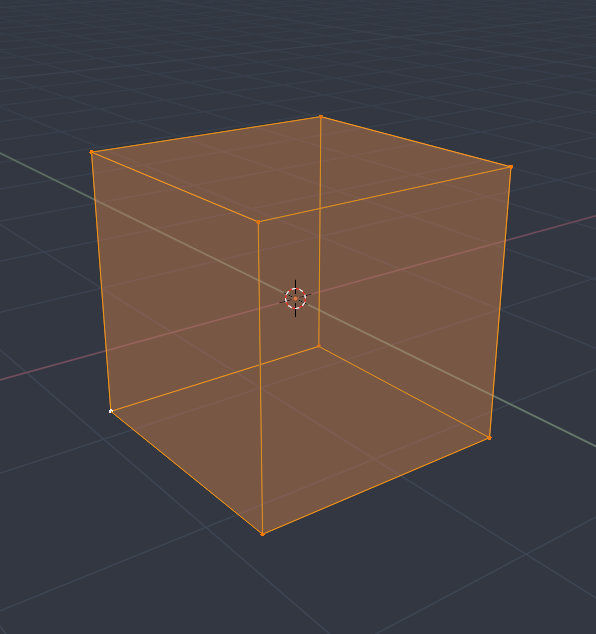
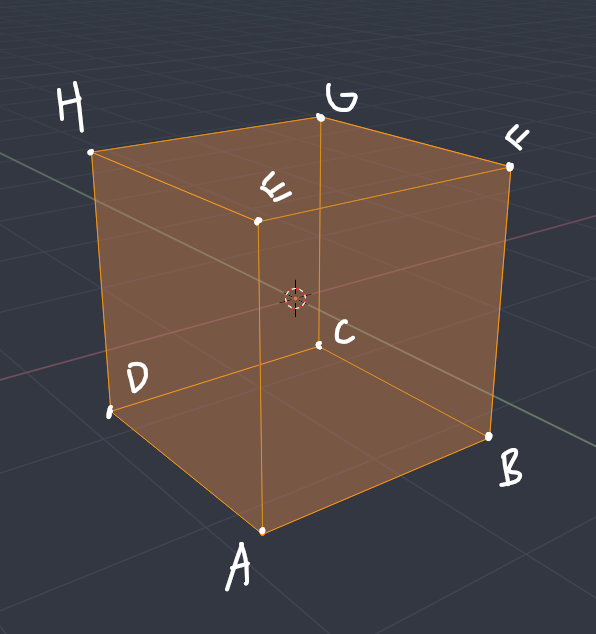


* C (1, 1, -1)



* D (-1, 1, -1)
* E (-1, -1, 1)
* F (1, -1, 1)
* G (1, 1, 1)
* H (-1, 1, 1).

Dengan adanya data vertex tersebut kita dapat menentukan bentuk dari sebuah object dalam kasus ini adalah sebuah kubus

Setiap Data Tersebut akan dapat digunakan untuk menampung data tambahan yang nantinay akan menjadi sebuah metode Steganografi dengan media objek 3D.

**Kasus Paling Sederhana**

Saya ingin menyisipkan sebuah pesan tersembunyi kedalam objek 3D tersebut, maka akan dilakukan dengan merubah pesan tersebut menjadi biner.

Misalnya **‘A’**, maka akan diubah dulu menjadi biner **‘01000001’**

dengan biner tersebut kita dapat membaginya menjadi blok blok bit, dalam kasus ini akan dibagi menjadi 3bit/blok dengan tambahan padding jika diperlukan.

**‘01000001’** akan dirubah menjadi **‘010’, ‘000’, ‘001’**. (Merah = Hasil Penambahan Bit)

Blok blok bit tersebut bisa dikonversikan kedalam bilangan basis desimal, menjadi **‘2, 0, 1’**

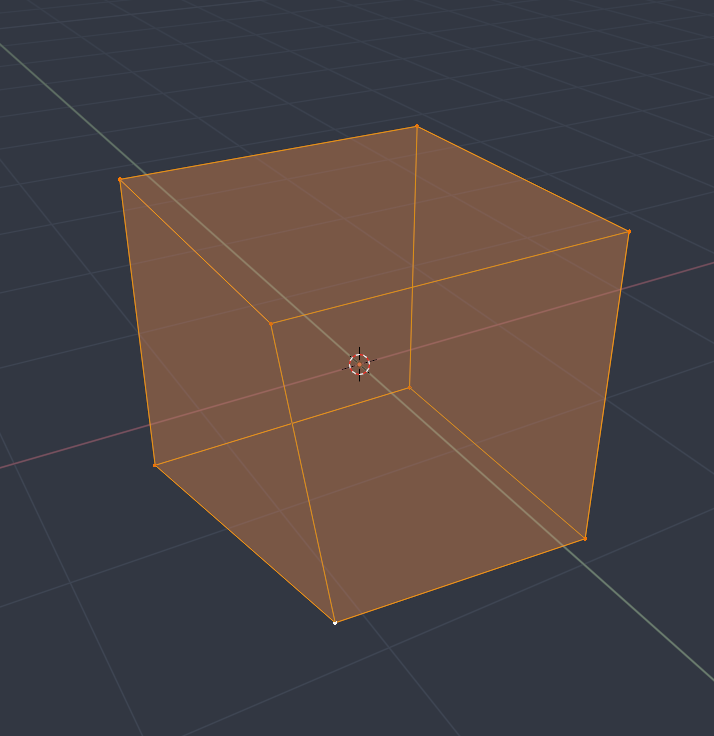
Karena nilai maksimum dari 3 bit adalah 7 maka hasil dari bilangan biner yang sudah dikonversi ke desimal akan dibagi 7 sehingga akan menjadi **‘0.285, 0, 0.142’** dan bisa disebut dengan data ternormalisasi

Setelah didapat nilai dari blok blok bit tersebut kita bisa melakukan substitusi atau proses penjumlahan terhadap vertex yang ada di objek semula.

Maka akan didapatkan hasil sebagai berikut :



Berikut adalah bentuk objek yang sudah diubah :



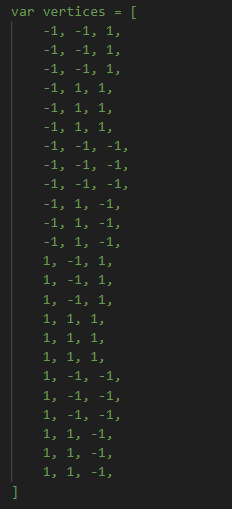
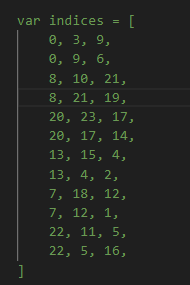
**Dekripsi**

Untuk dekripsi akan sulit jika model asli tidak dimiliki karena untuk menghitung offset atau perubahan yang terjadi pada vertex diperlukan data aslinya sehingga key merupakan object aslinya itu sendiri.

Untuk mengatasi masalah tersebut, pada saat enkripsi atau penyisipan pesan dapat dilakukan dengan juga menambahkan sebuah json pada file 3D dalam kasus ini format file yang digunakan adalah GLTF/GLB yang dimana didalamnya memiliki data json yang berisi scene, nama objek dan lain lain yang biasa disebut dengan node.

Dengan adanya node tambahan maka kita dapat memasukan data tambahan juga dalam hal ini adalah data vertex object aslinya dan data indexnya jika diperlukan, sehingga untuk merekonstruksi sebuah objek hasil penyisipan akan lebih mudah dilakukan.

Contoh data yang akan digunakan :

**Pembulatan**

Akan dilakukan pembualatan nilai untuk semua angka yang akan diproses, dari mulai vertex pada saat upload, lalu hasil dari normalisasi nilai desimal yang didapat dari biner, lalu pada saat hasil akhir akan dilakukan pembulatan.

Untuk pembulatan data vertex akan diambil hanya nilai utama dengan 3 angka di belakang koma sehingga akan lebih mudah untuk melakukan kalkulasi dan tidak rentan terhadap error yang disebabkan oleh ketidak konsistenan jumlah angka di belakang koma

**Ilustrasi Proses**

**Proses Enkripsi**

1. Misal data Plaintext yang diterima adalah **‘B’**.
2. **‘B’** memiliki nilai biner **[ 01000010 ]**.
3. Dilakukan proses segmenting sehingga didapat **[ 010, 000, 10 ]**.
4. Dilakukan Padding dan didapat nilai **[ 010, 000, 010 ]**.
5. Selanjutnya nilai tersebut diubah menjadi nilai desimal, **[ 2, 0, 2 ]**.
6. Lakukan normalisasi pada nilai tersebut **[ 2/7, 0, 2/7 ]** atau **[ 0.2857142857142857, 0, 0.2857142857142857 ]**.
7. Lakukan pembulatan menjadi [ **0.286, 0, 0.286 ]**.
8. Sandingkan dengan data koordinat vertex sesuai dengan indexnya, misalnya didapat **[ 1.25425, 1.557693, 1.312619346 ]**.
9. Lakukan pembulatan terhadap data koordinat vertex, sehingga didapatkan nilai **[ 1.254, 1.557, 1.312 ]**.
10. Jumlahkan data koordinat dengan data hasil pembulatan normalisasi, didapat **[ 1.540, 1.557, 1.598 ]**. ( ini menjadi vertex object yang memiliki pesan tersembunyi).
11. Memasukkan data vertex dan index awal ke objek tersebut sebagai atribut sehingga data dapat diekstrak kembali menjadi plaintext

**Proses Dekripsi**

* Membaca data vertex dari model yang didapat, misalnya **[ 1.540, 1.557, 1.598 ]**.
* Membaca data vertex yang ada di atribut objek tersebut, misalnya didapat **[ 1.254, 1.557, 1.312 ]**.
* Mengurangi data vertex model dengan data vertex dari atribut model, sehingga didapat [ **0.286, 0, 0.286 ]**.
* Nilai dikalikan 7 dan dibulatkan menjadi 1 angka sehingga didapat **[2, 0, 2]**.
* Mengubahnya kedalam bentuk biner **[010, 000, 010]**.
* Menghilangkan padding dan merge array setiap 3 data, didapat **[01000010]**.
* Konversi kedalam bentuk text sehingga didapat **‘B’**