

Dokumentasi Final Project Grafika Komputer Aplikasi Tata Kota

5112100004 Andrean Hutama Koosasi

5112100013 Fransiskus G. N. Dwika S.

5112100067 Rahardian Dewa Bimantara

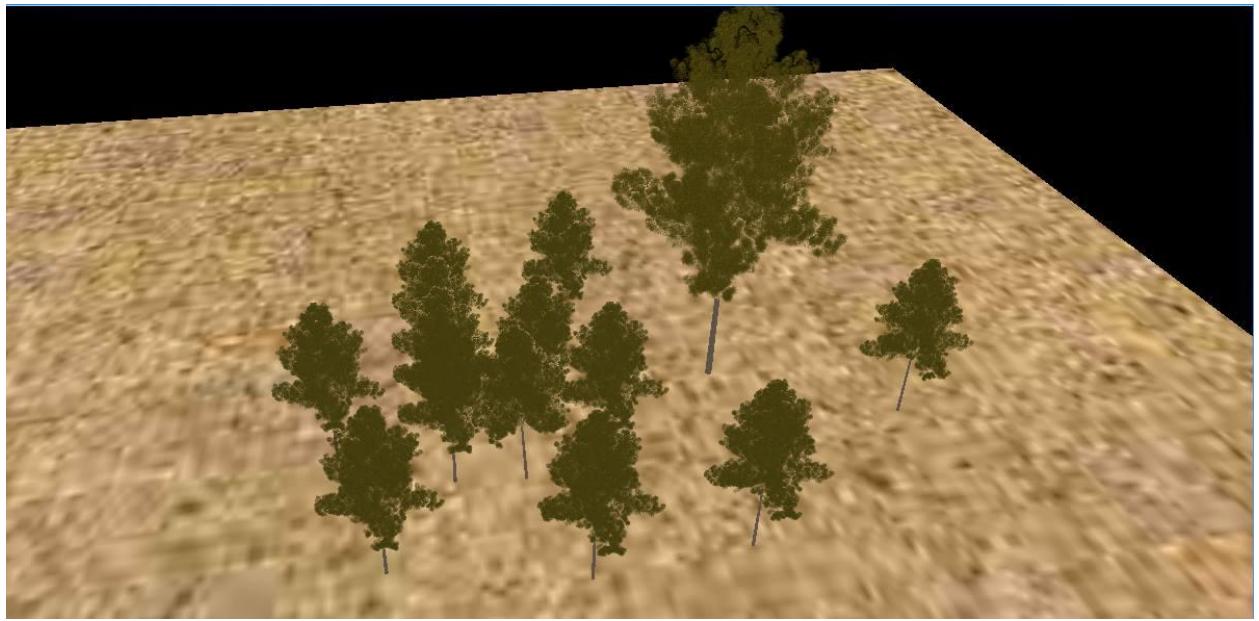
5112100182 Dicky Irwanto

5112100207 Andrys Daniel Silalahi

1. Pendahuluan

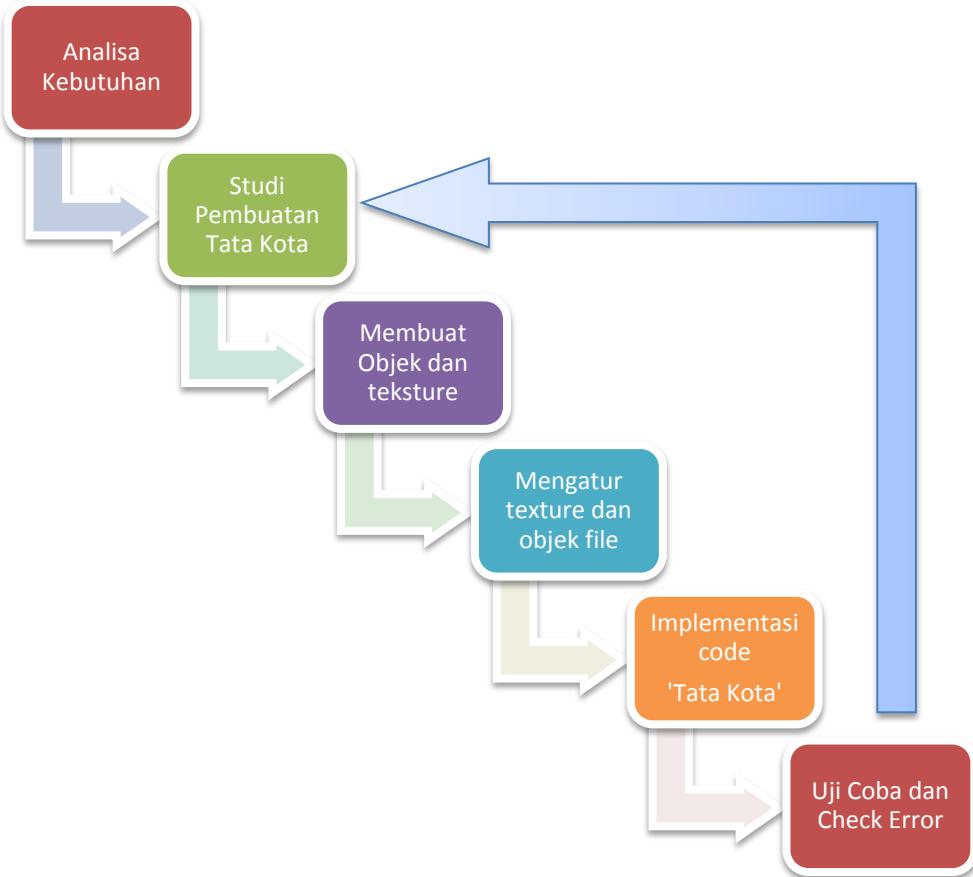
Final project grafika komputer kelompok kami memiliki tema mengenai tata kota dimana pengguna dapat membuat susunan komposisi kota sesuai dengan keinginan dan kreativitas pengguna. Dalam final project tata kota ini, terdapat objek pohon, rumah, pabrik, tanah, jalan kota, dan tanah. Masing-masing dari objek tersebut dapat dilakukan translasi, rotasi, scaling, dan perubahan tekstur. Pengoperasian dari final project ini menggunakan media mouse dan keyboard untuk melakukan pembuatan tata kota dan pengaturan objek-objek yang ada.

Implementasi final project tata kota menggunakan library-library pendukung yakni glm, freeglut, dan devil. GLM library dan devil library digunakan untuk memuat dan menampilkan objek dan teksture yang digunakan dalam final project tata kota, dimana tampilan dari objek yang bertipe .obj dan .mtl file dan teksture yang dimuat bertipe .jpg dan .png yang di import dari blender. Freeglut library digunakan untuk mengatur tampilan window, implementasi transformasi masing-masing objek yang dapat dilakukan (translasi, rotasi, scaling), penggunaan tekstur, serta pengaturan lighting yang membuat objek terlihat nyata.



Gambar 1. Hutan ‘Tata Kota’

2. Tahap Pembuatan ‘Tata Kota’



Gambar 2. Diagram Alur Pembuatan 'Tata Kota'

Analisa Kebutuhan

Pada bagian ini pengembang melakukan identifikasi kebutuhan untuk keperluan pembuatan aplikasi 'tata kota'. Dan memikirkan proses jalan cerita aplikasi yang akan dibangun, pada tahap ini menghasilkan prototype dengan jumlah objek sebagai berikut:

1. 2 macam pabrik.
2. 2 macam rumah.
3. 2 macam jumlah pohon.
4. 3 macam jalan raya.
5. 1 macam latar tanah.

Kemampuan objek yang dibagun ialah:

1. Dapat diubah letak sesuai dengan kebutuhan.
2. Dapat melakukan scaling(resize), rotate dan penggantian texture.
3. Menerapkan metode Collision sehingga setiap bangunan tidak dapat bertumpukan.
4. Objek diletakkan pada bidang sesuai dengan keadaan tanah.

Kemampuan kamera:

1. Melakukan gerakan bebas yaitu geser, zoom in, zoom out, putar.

Pengoperasian tersebut dengan menggunakan mouse dan keyboard.

Studi Pembuatan ‘Tata Kota’

Pada tahap ini pengembang melakukan pembelajaran yang berkaitan dengan pembuatan ‘Tata Kota’. Pembelajaran dilakukan dengan cara pembagian tugas belajar. Nantinya bagian anggota tugas belajar akan menjelaskan ke anggota ke yang lain tentang materi yang telah dipelajari.

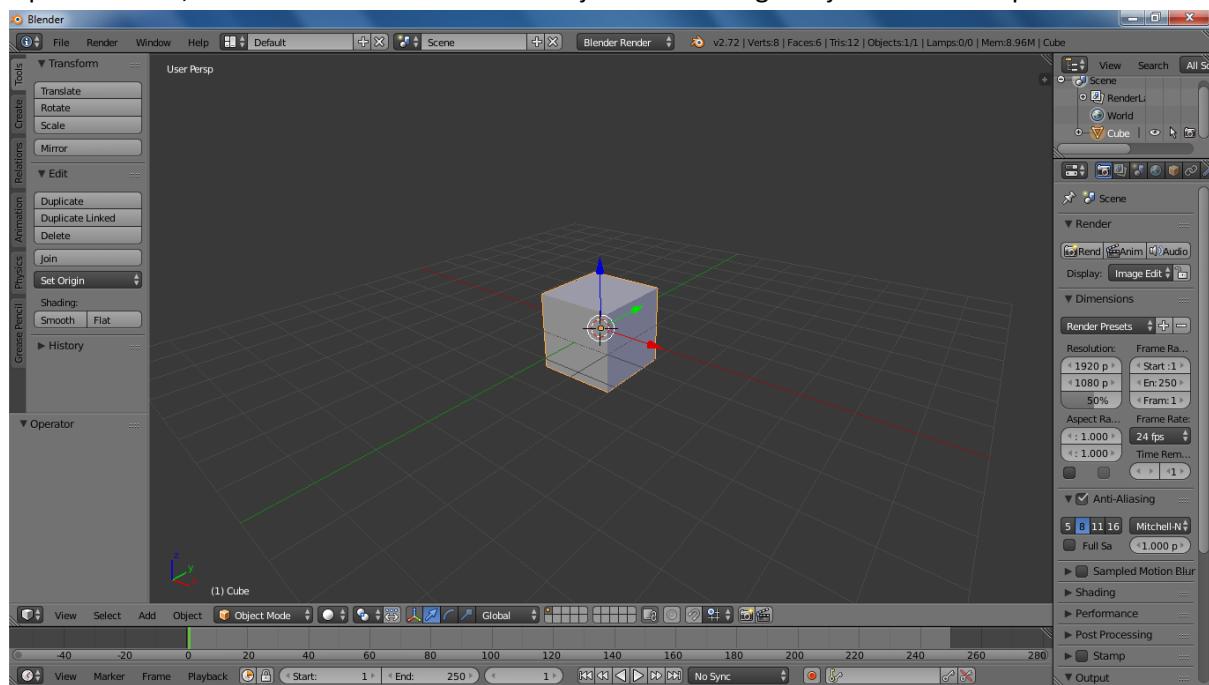
Membuat Objek dan Texture

Pada bagian ini, akan didemokan sebuah pembuatan objek rumah sederhana, beserta texturing rumah untuk memberikan kesan nyata pada objek yang telah dibuat. Diasumsikan bahwa dalam tutorial ini komputer telah terpasang program Blender.

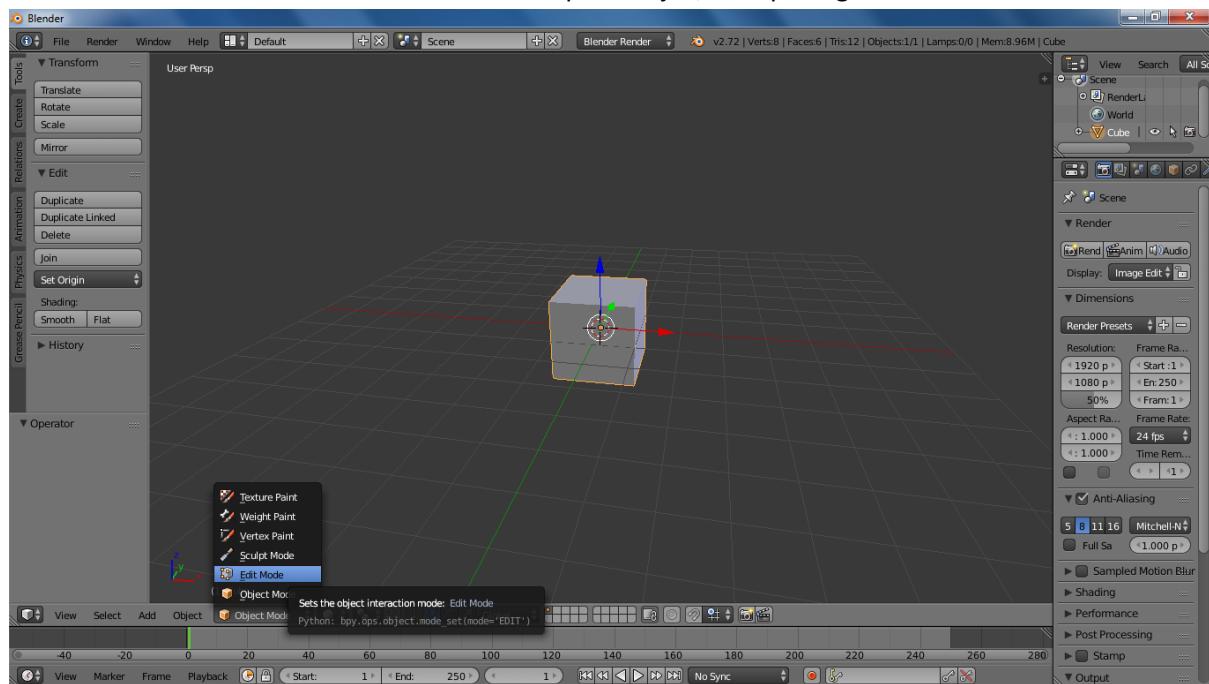
1. Buka Program Blender.



2. Open New File, maka akan terbentuk sebuah objek kubus sebagai objek awal start-up.

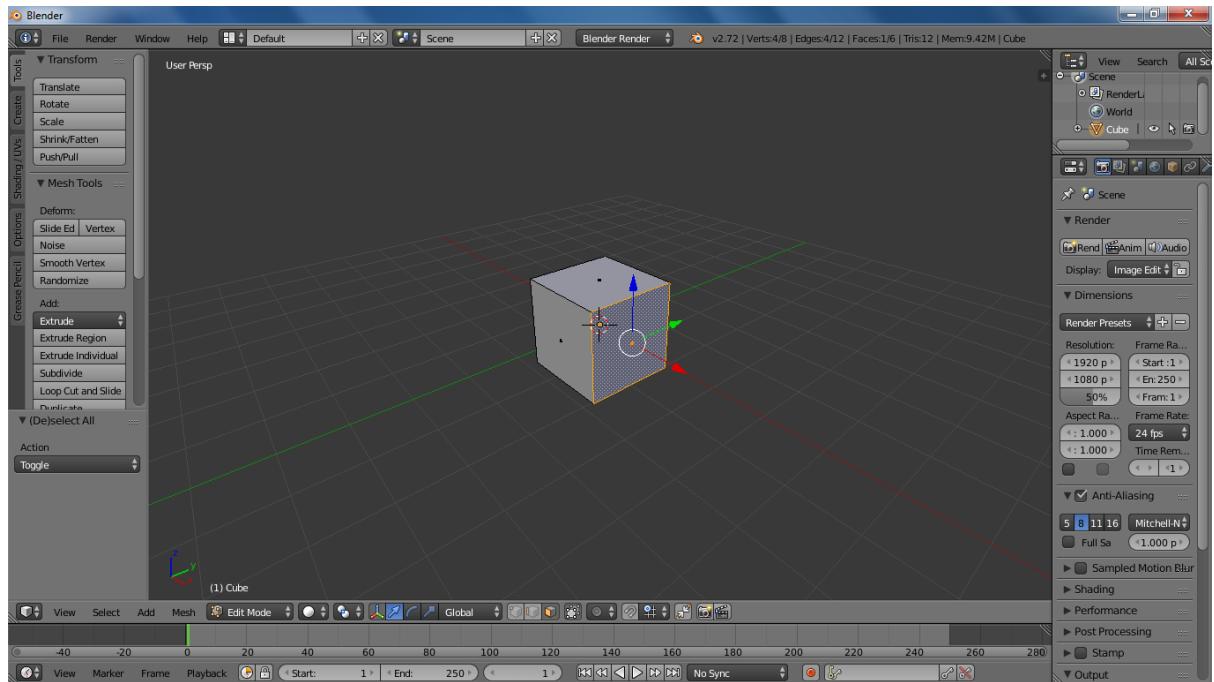


3. Pilih Edit mode untuk melakukan transformasi pada objek, lihat pada gambar.

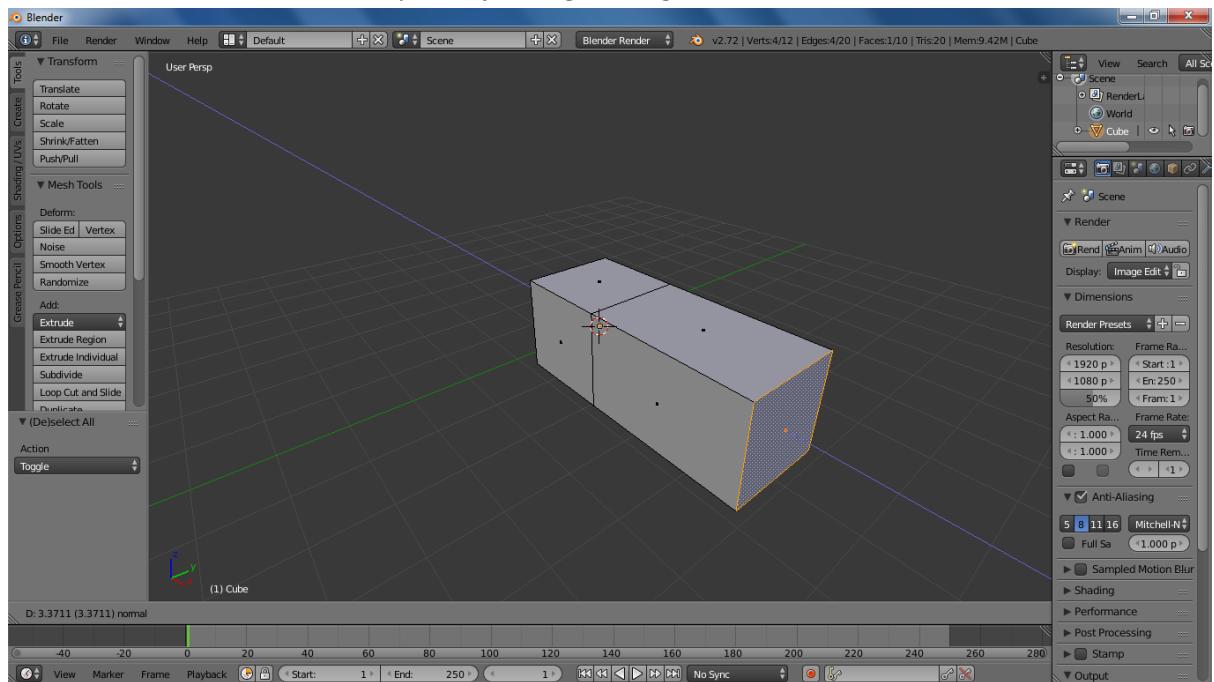


4. Lakukan select pada objek, untuk melakukan transformasi pada bagian terpilih tersebut. Sebagai contoh, dilakukan select pada face objek kubus, namun bisa dilakukan select yang lebih

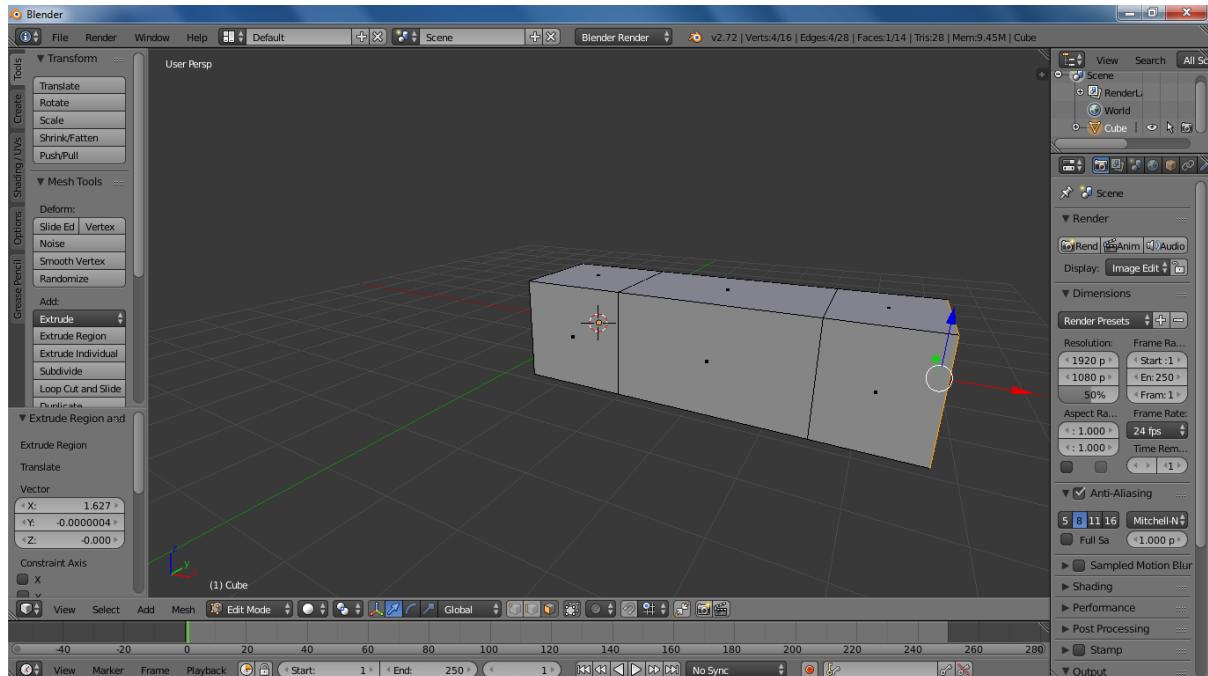
beragam, seperti select pada salah satu titik (vertex) objek, select pada salah satu garis (edge) objek, dan select pada salah satu permukaan (face) objek.



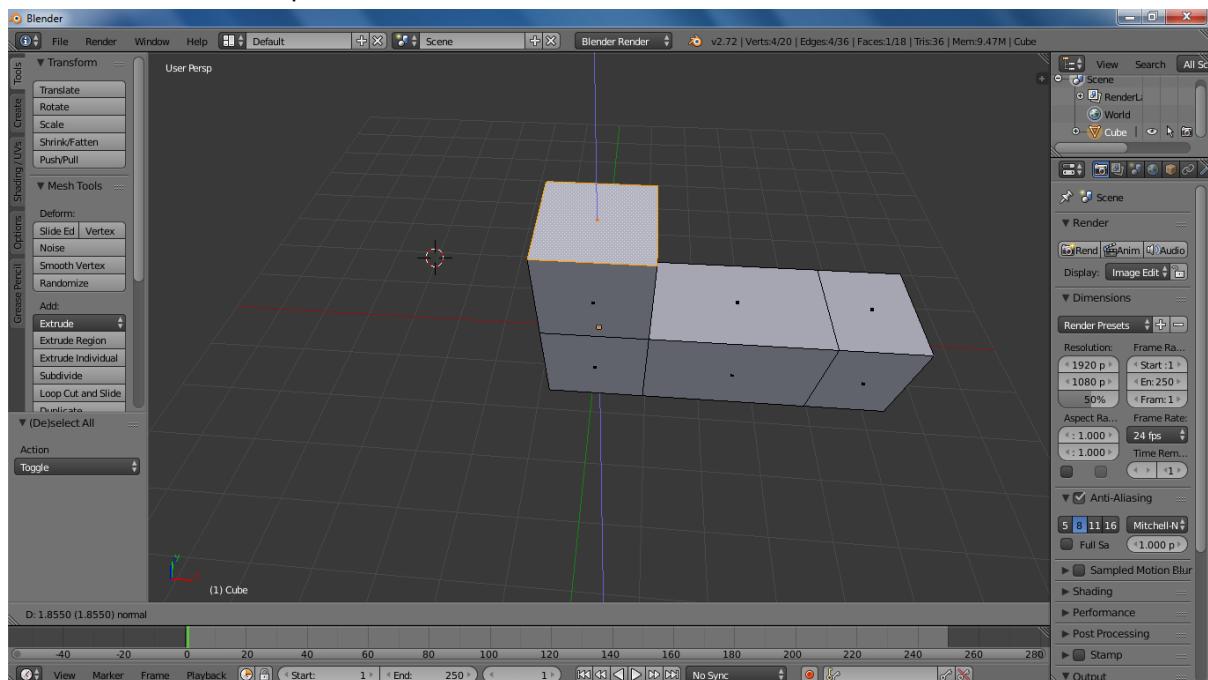
5. Tekan tombol 'E' pada keyboard sebagai shortcut untuk melakukan extrude (penggandaan objek yang telah dipilih sebelumnya) dan lakukan pengubahan bentuk serta ukuran objek hasil extrude sesuai keinginan. Pada contoh, akan dilakukan extrude membentuk balok baru dari salah satu face kubus awal. Balok ini nantinya menjadi bagian tengah rumah sederhana.



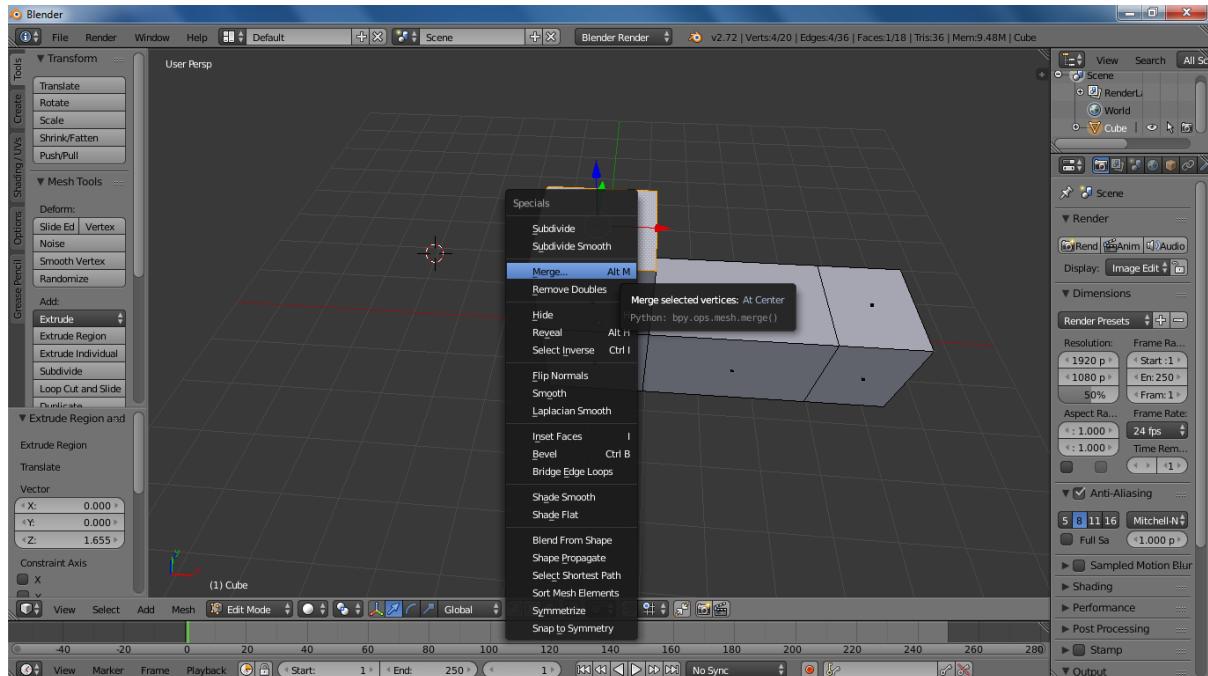
6. Melakukan extrude pada face balok baru yang telah dibentuk pada face sebelumnya, untuk membentuk bagian samping rumah.



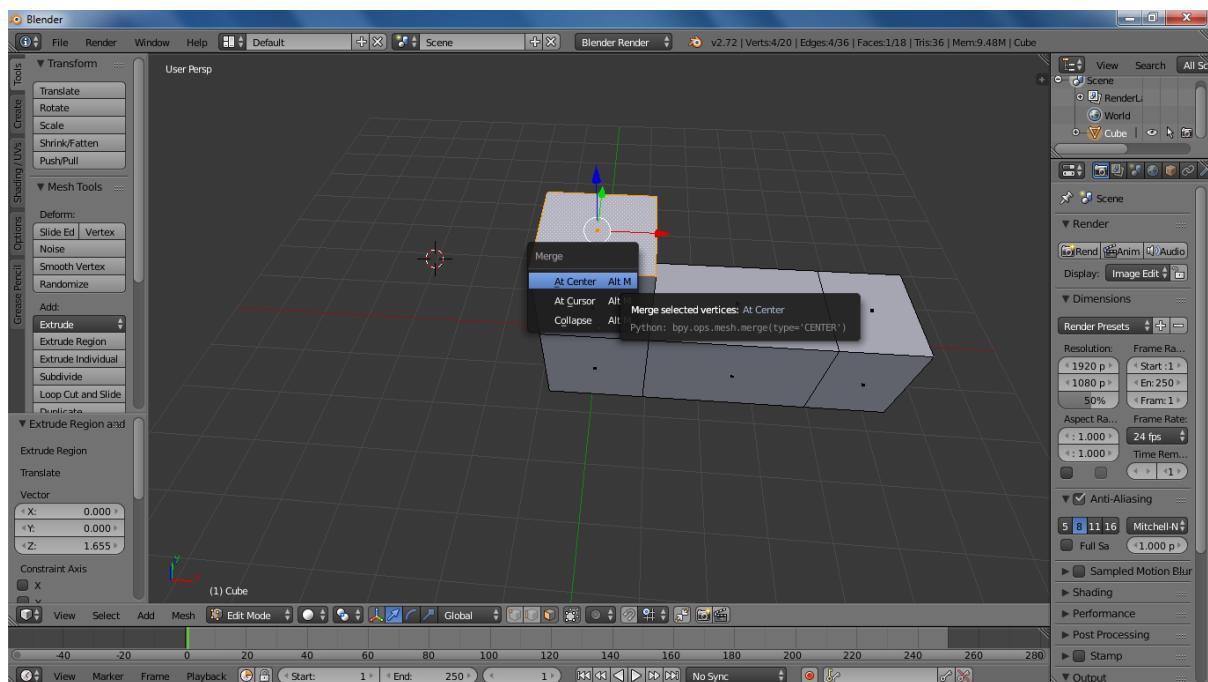
7. Melakukan extrude pada face bagian atas kubus pertama. Hasil extrude ini nantinya akan membentuk sebuah atap rumah.



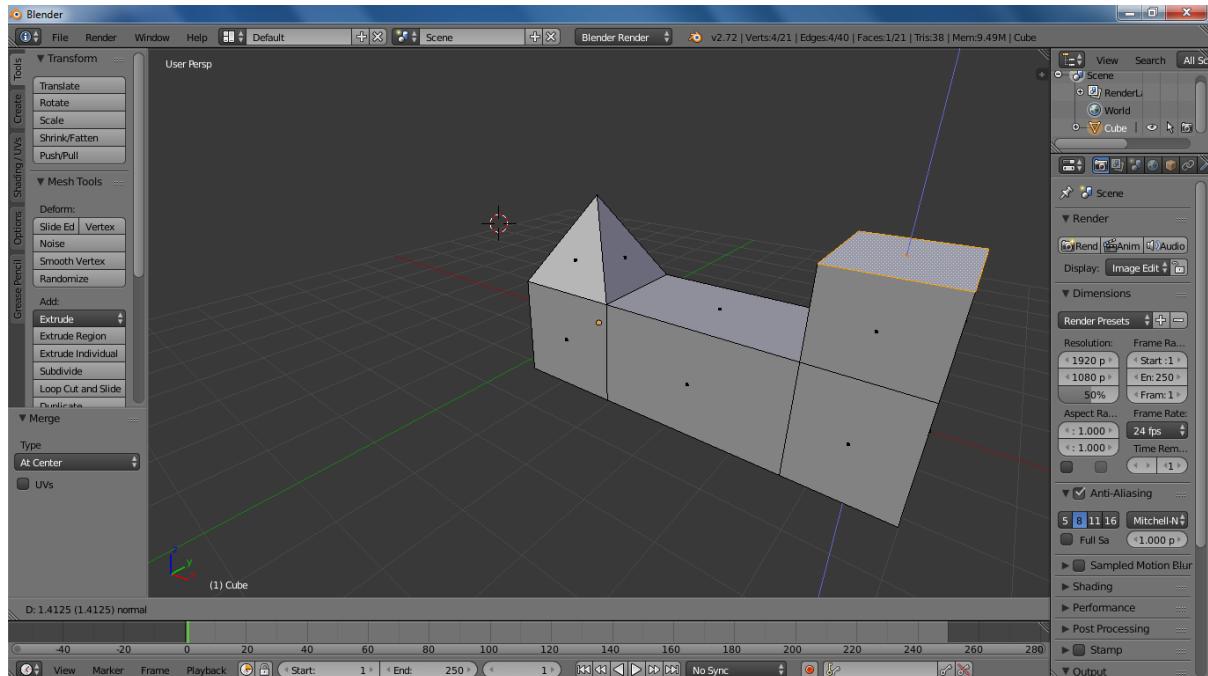
8. Lakukan seleksi pada face atas kubus hasil extrude sebelumnya, lalu tekan tombol 'W' pada keyboard, sebagai shortcut untuk memunculkan menu spesial (Special Actions). Pada contoh ini, karena akan dibentuk sebuah atap rumah, maka akan dipilih submenu merge.



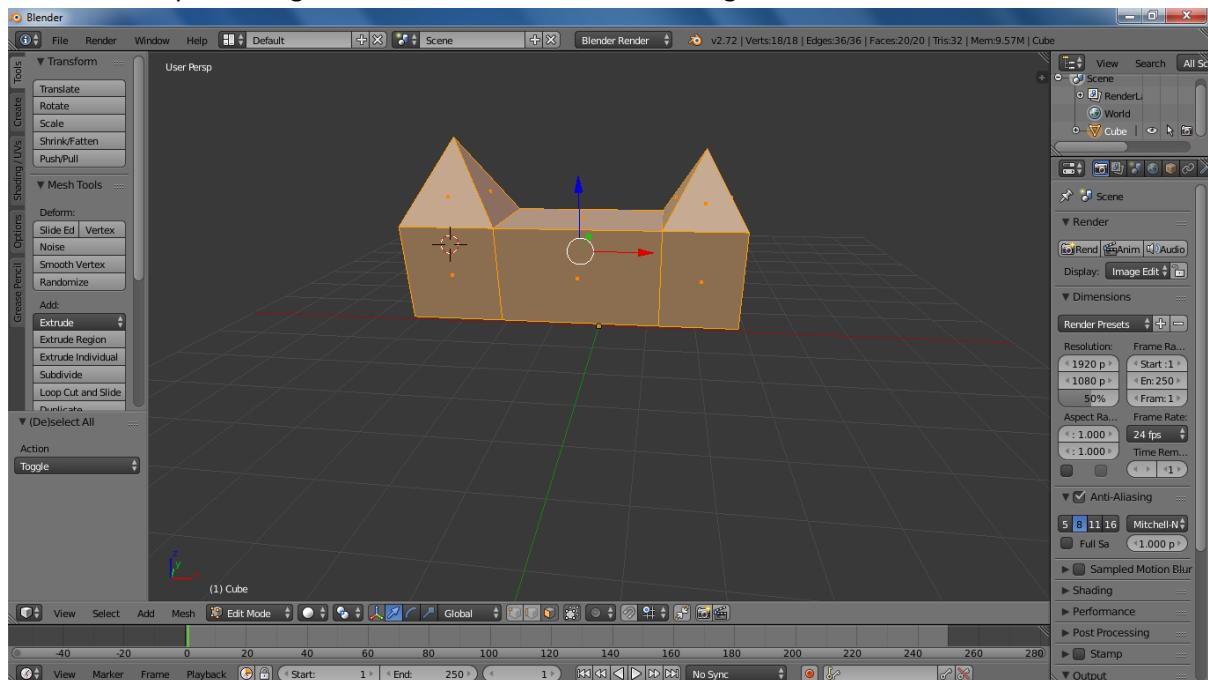
9. Setelah submenu merge dipilih, maka akan muncul beberapa pilihan merge. Untuk membentuk sebuah atap yang runcing (berpusat pada satu titik/vertex) maka dipilih pilihan Merge > At Center.



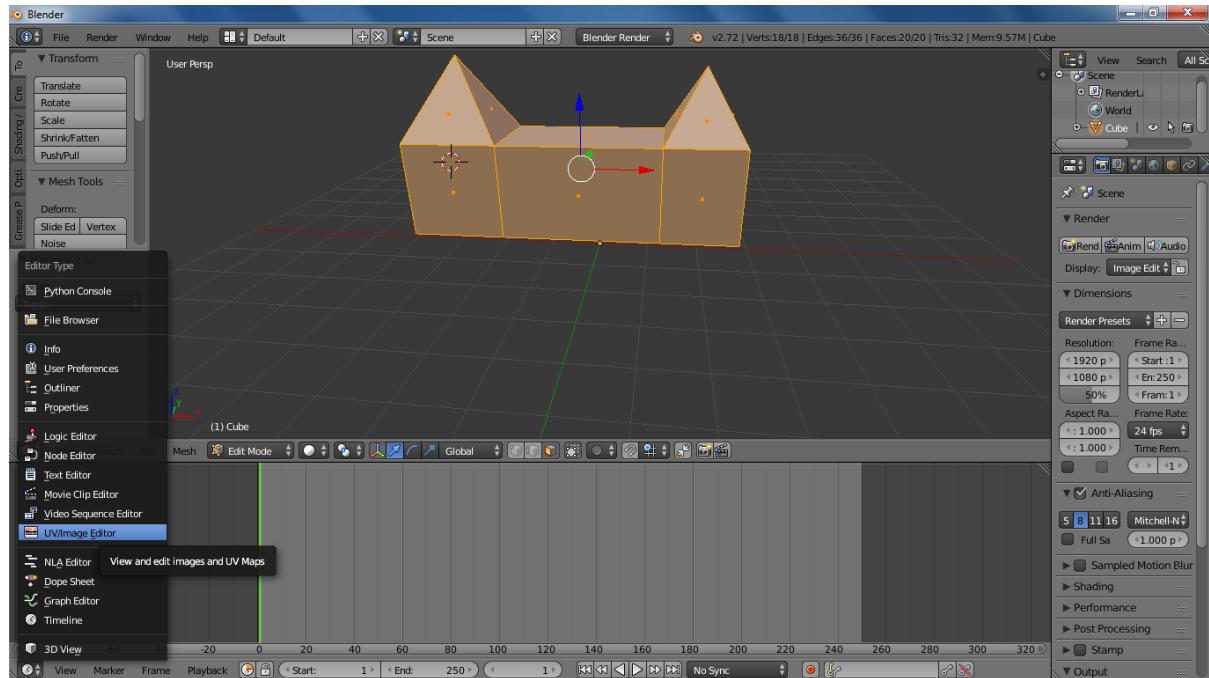
10. Setelah melakukan Merge Center, ulangi langkah 6 sampai langkah 9 namun pada face atas kubus kedua. Pada gambar terlihat bahwa sisi kubus pertama telah membentuk sebuah atap rumah yang berpusat pada satu vertex.



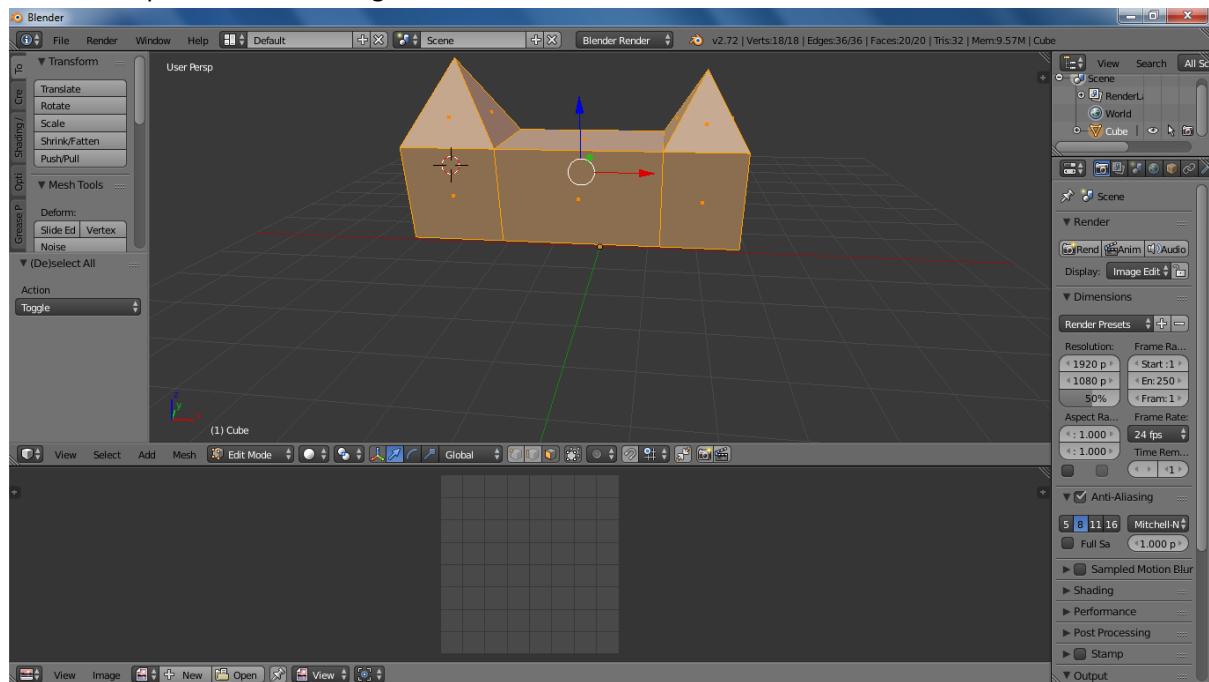
11. Setelah terbentuk dua buah atap dalam wujud bangun limas segiempat, lakukan seleksi pada keseluruhan faces objek yang ada dengan menekan tombol "A" pada keyboard. Seleksi seluruh faces ini merupakan langkah awal untuk melakukan texturing.



12. Pada workspace Blender bagian bawah, lakukan pengubahan workspace menjadi tempat untuk melakukan texturing. Caranya, temukan ikon kecil bergambar jam yang terletak pada posisi sebelah kiri bawah. Ikon jam menandakan workspace tersebut sedang digunakan untuk timeline (tidak dibahas pada dokumentasi ini). Klik dan pilih workspace untuk U/V Image Editor seperti pada gambar. Bergantung pada pengguna, workspace bagian bawah mampu diubah ukurannya sesuai kebutuhan.

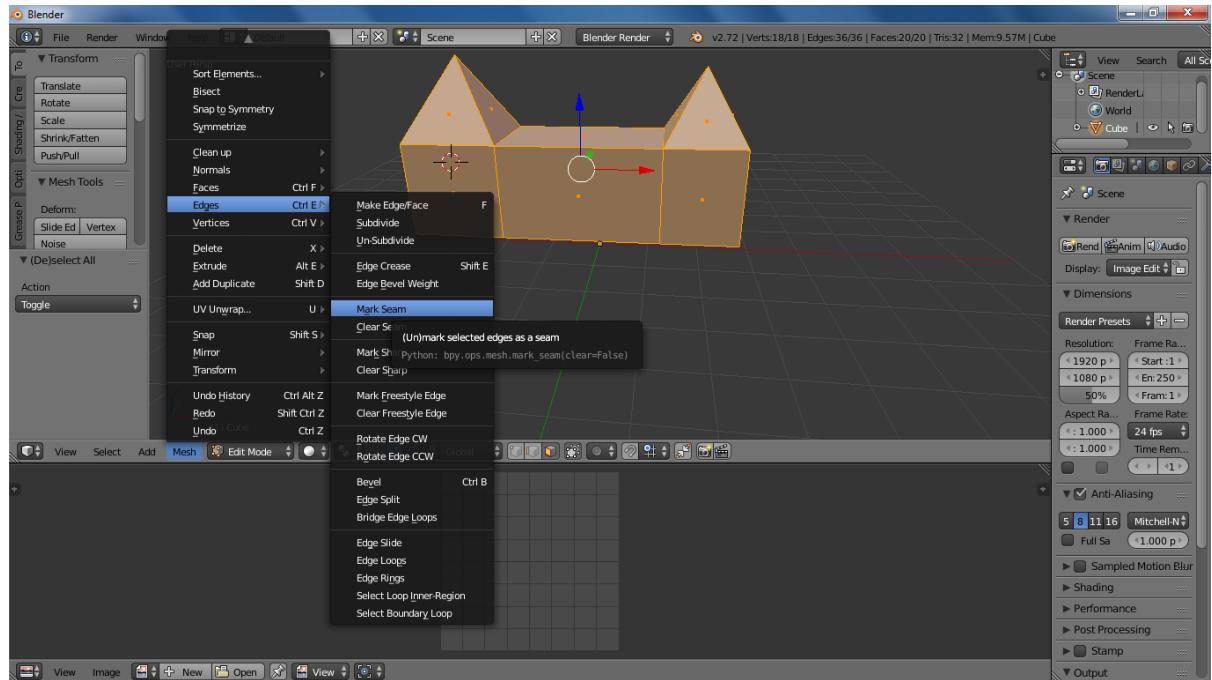


13. Contoh hasil perubahan workspace menjadi workspace U/V Image Editor. Apabila telah melihat bidang persegi pada workspace bagian bawah, maka workspace tersebut telah berganti mode untuk mempermudah texturing.

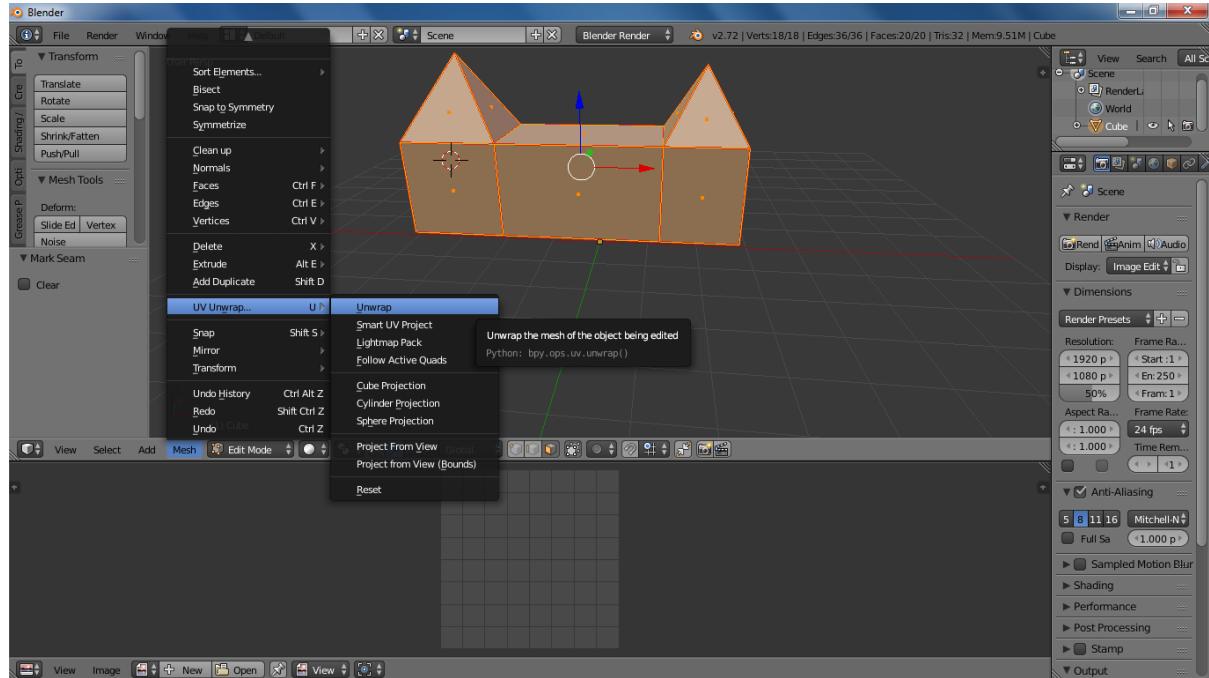


14. Arahkan mouse pada workspace bagian atas (tempat dimana objek rumah sederhana dibuat), lalu pilih menu Mesh > Edges > Mark Seam. (setelah sebelumnya telah dilakukan seleksi)

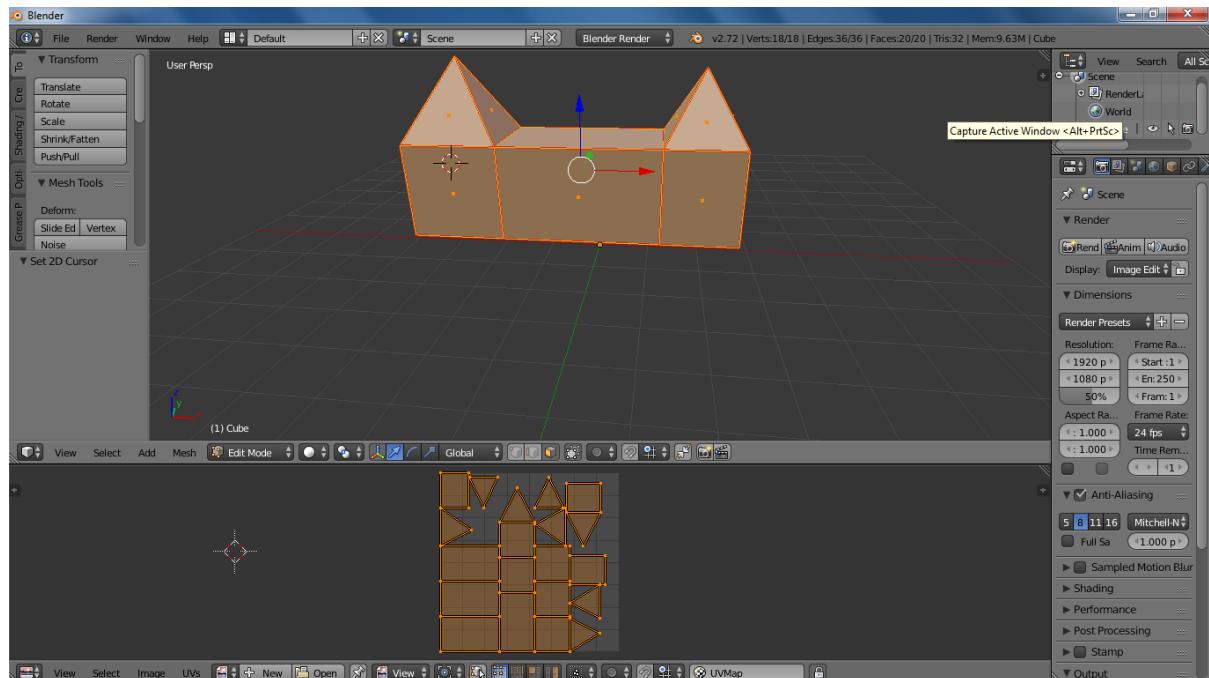
keseluruhan faces objek



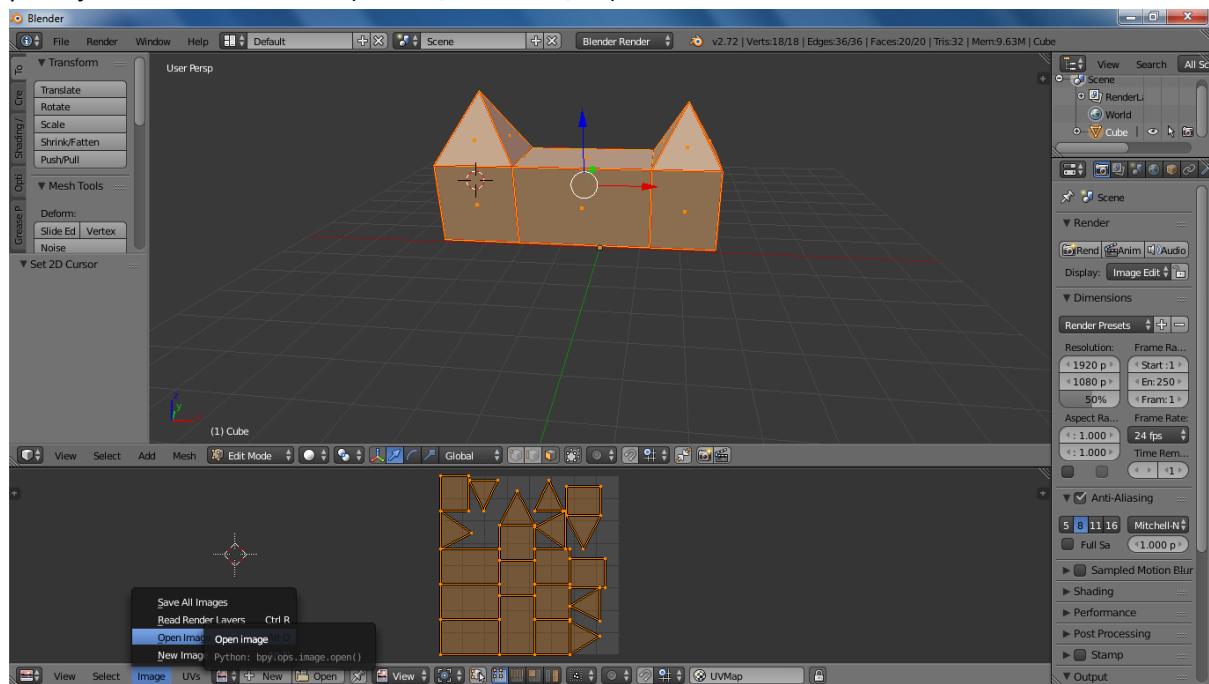
15. Lanjutkan dengan unwrapping (U/V Unwrap > Unwrap) seluruh faces. Unwrapping memudahkan kita dalam memilih bagian dari tekstur yang diinginkan untuk mengisi face tertentu. Langkah ini bisa saja dilewati, namun akan menyulitkan pengguna dalam texturing, karena harus menyediakan gambar yang tepat dengan lokasi faces objek yang telah dipetakan oleh Blender.



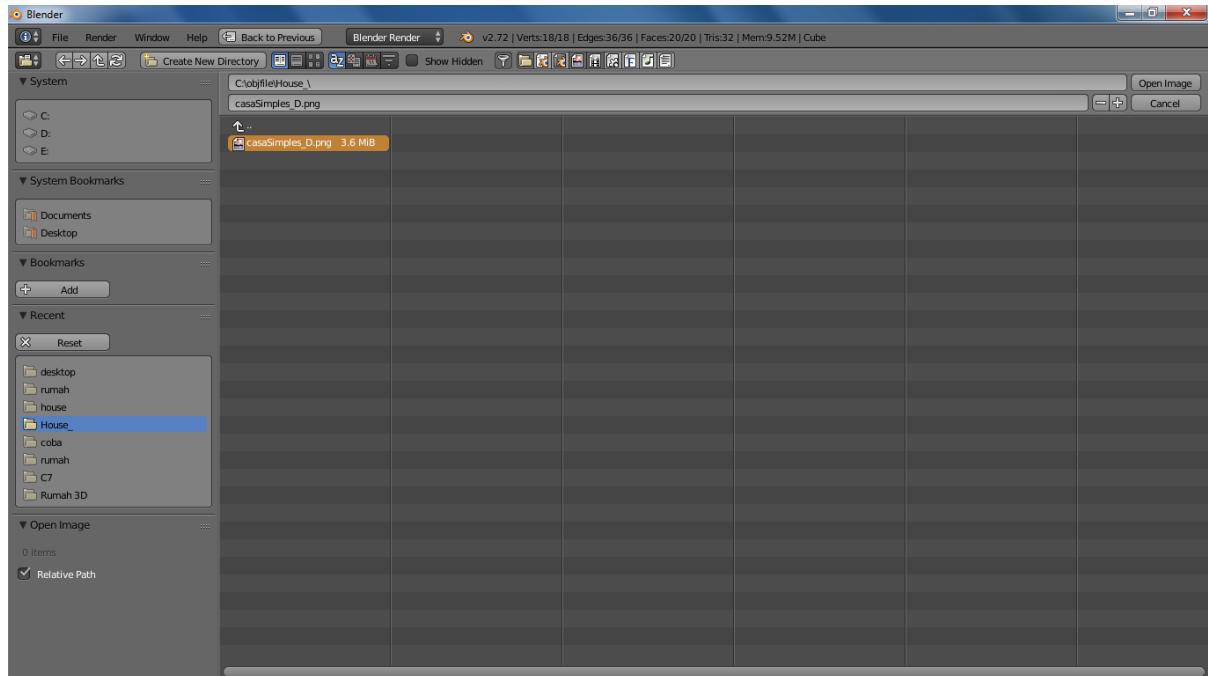
16. Setelah dilakukan Unwrap pada seluruh faces, maka pada workspace bagian bawah akan tampak seluruh bagian rumah sederhana yang terpecah – pecah, siap dipetakan sesuai keinginan pengguna.



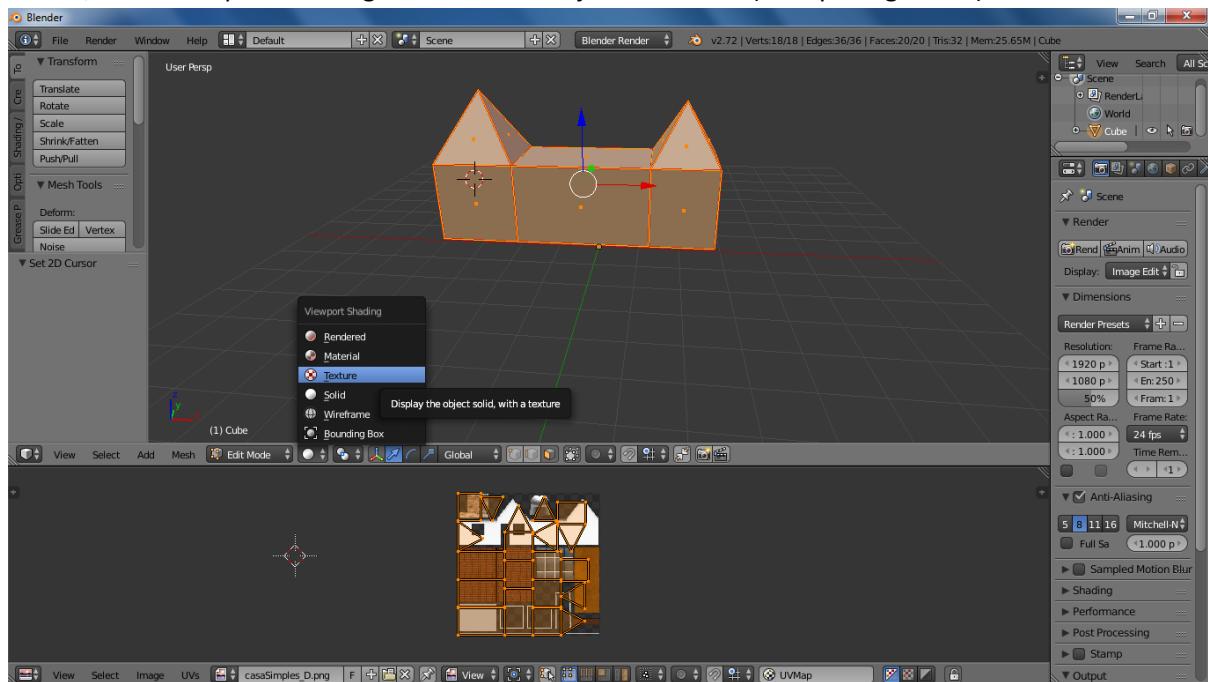
17. Setelah seluruh faces objek dipecah, saatnya melakukan load image untuk texturing rumah sederhana yang telah dibuat. Pada workspace bagian bawah, pilih menu Image > Open Image, dan pilih gambar yang diinginkan untuk menjadi tekstur rumah sederhana. Gambar yang bisa dimasukkan menjadi tekstur setidaknya harus berukuran persegi, supaya tidak terjadi kesalahan penerjemahan oleh Blender (stretch, extended, dll).



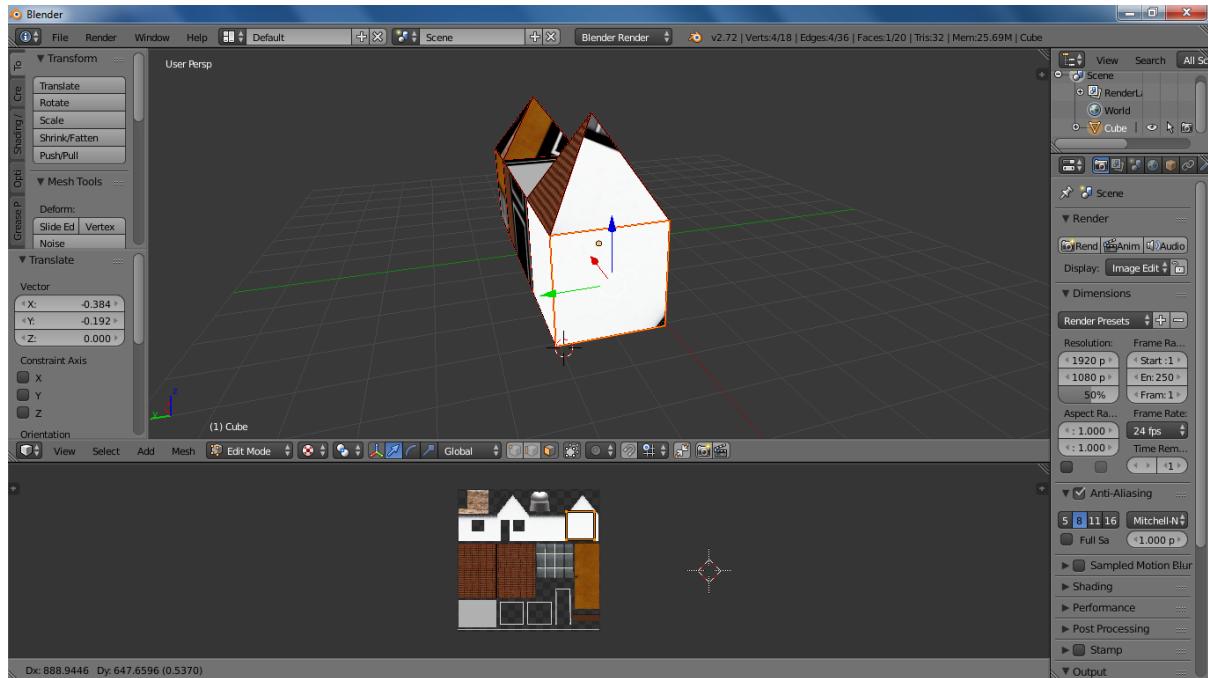
18. Contoh gambar yang dipilih untuk texturing berformat Portable Network Graphics (PNG). Gambar bisa berformat JPEG (.jpg), PNG (.png), maupun Bitmap (.bmp). Sebagai catatan, apabila menginginkan tekstur yang berbeda antar faces, maka lakukan seleksi pada face yang dikehendaki memiliki tekstur berbeda (pada workspace bagian atas/workspace pembuatan objek), lalu lakukan Open Image seperti biasa.



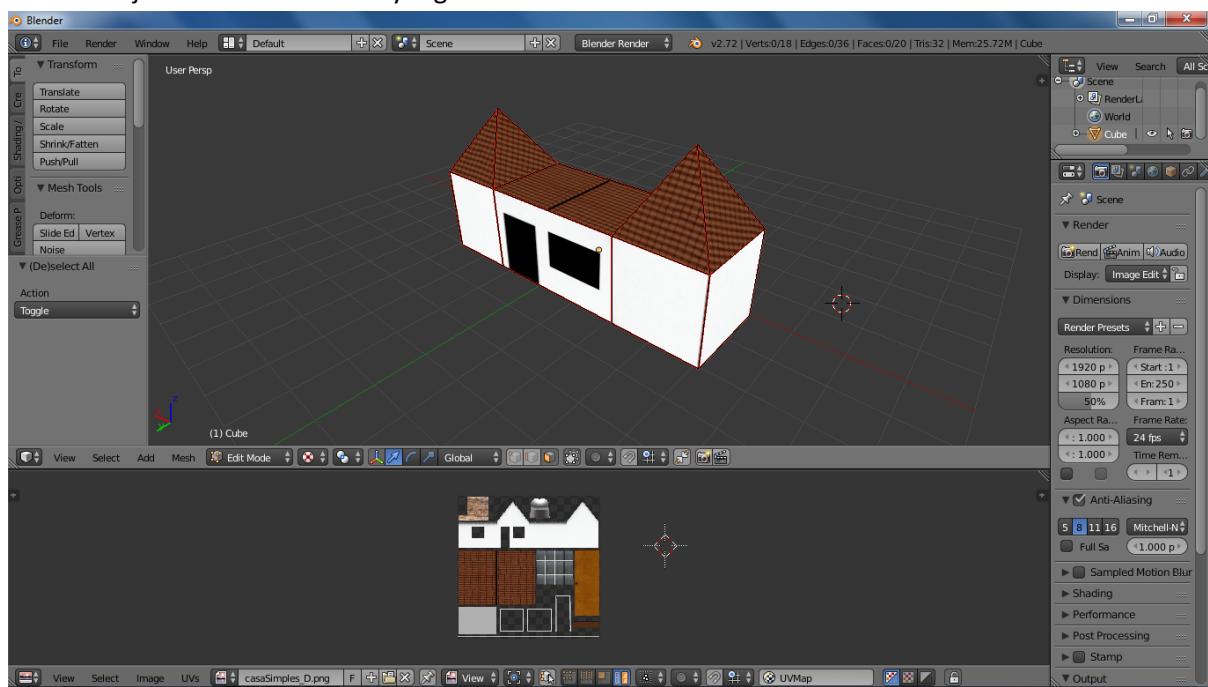
19. Setelah memilih gambar yang diinginkan, maka workspace bagian bawah akan terisi oleh gambar tersebut. Untuk dapat langsung melihat hasilnya (texturing) pada objek yang telah dibuat, ubah Viewport Shading dari "Solid" menjadi "Texture" (lihat pada gambar).



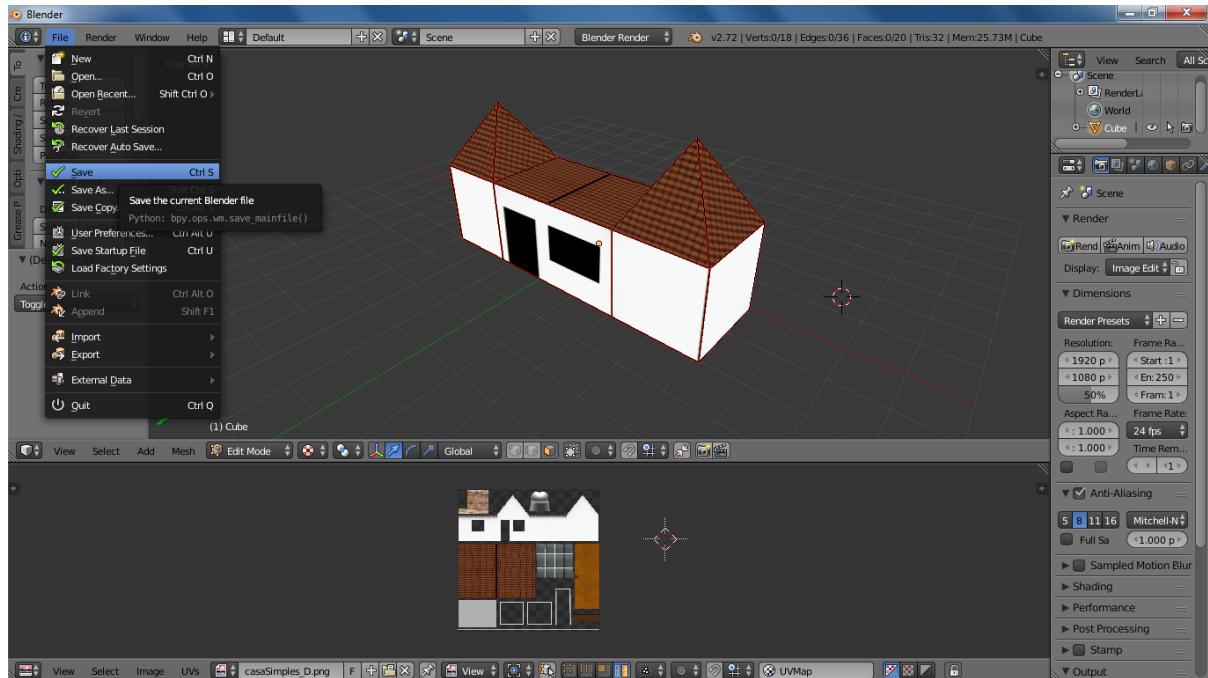
20. Lakukan texturing dengan cara melakukan seleksi (workspace bagian atas) pada face yang diinginkan, lalu drag (klik mouse kanan) dan drop (klik mouse kiri) face pada workspace ke bagian gambar yang diinginkan untuk muncul sebagai tekstur pada objek rumah sederhana.



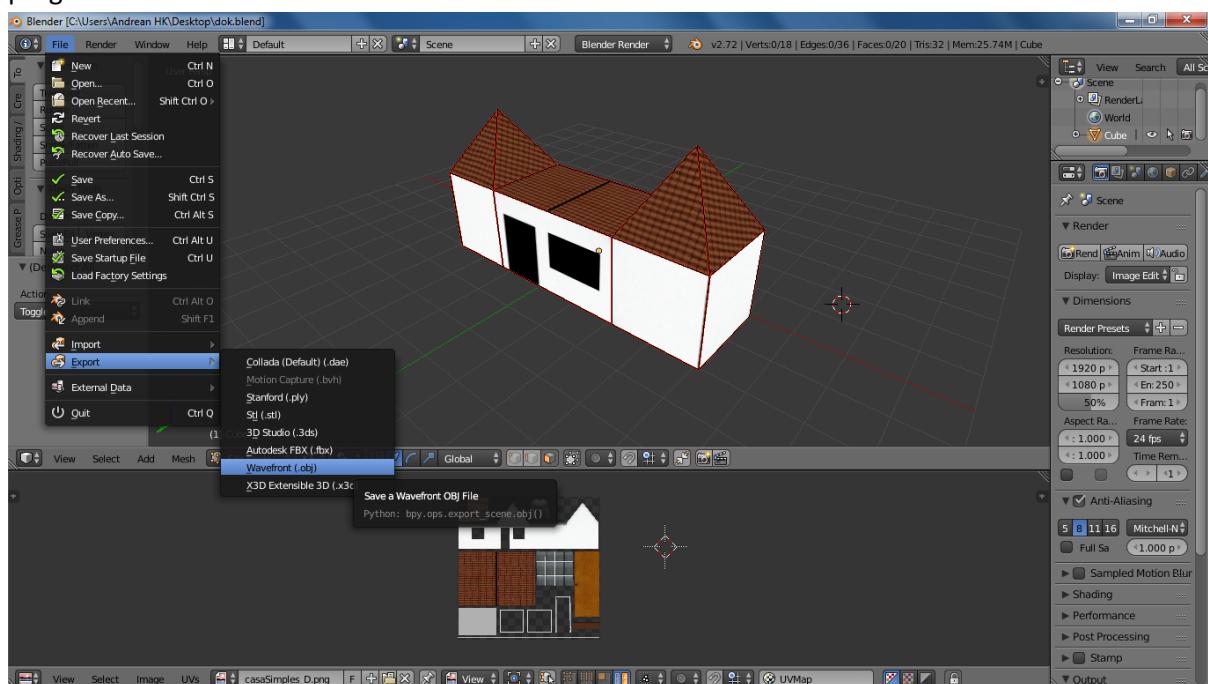
21. Contoh objek rumah sederhana yang telah selesai diberi tekstur.



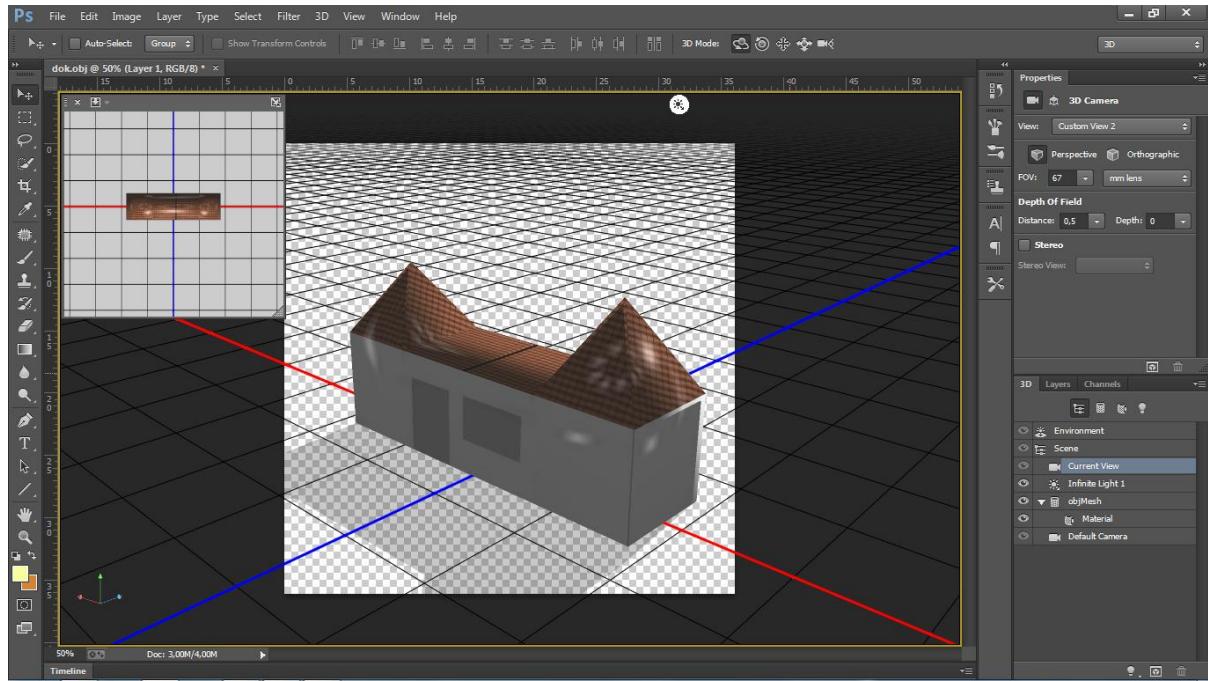
22. Simpan objek rumah sederhana apabila telah selesai melakukan modifikasi terhadap objek dengan memilih menu (pada workspace bagian atas) File > Save. Akan ditampilkan direktori tempat menyimpan objek. Ubah sesuai keinginan.



23. Blender juga mampu melakukan penyimpanan dengan format lain. Pada dokumentasi ini, akan dilakukan penyimpanan dengan format Wavefront (.obj) untuk kemudian diload kedalam program.



24. Contoh objek berformat Wavefront (.obj) yang dibuka dengan Adobe Photoshop untuk melihat hasilnya.



Mengatur objek dan Texture

Pada bagian ini akan menjelaskan tentang bagaimana mengganti teksture dan pada objek yang telah dibuat.

Bagian dari file .obj yang harus diketahui

Jenis	Penjelasan
v/vt/vn	Vertex/Vertex texture/ Vertex Normal (Merupakan Titik Dalam Objek)
f	Face (Merupakan bidang datar dalam objek)
l	Line (Merupakan garis dalam objek)
p	Point (Merupakan titik dalam objek)
Mtlb	Merupakan pemanggilan file .mtl

Bagian dari file .mtl yang harus diketahui

Jenis	Penjelasan
newmtl	Definisi awal untuk objek baru/bagian objek
Ka	Untuk definisi bagian teksture yang memiliki ambient color.
Kd	Untuk definisi bagian teksture yang memiliki difuse color.
Ks	Untuk definisi bagian teksture yang memiliki spectacular color.
d	Untuk implementasi transparen.
Tr	Tingkat transparen yang diinginkan
Ns	Range untuk spectacular pada Ks.
map_Ks/map_Ns/ map_d/map_bump	Merupakan pemanggilan teksture pada .mtl file. Map_Ks untuk teksture spectacular, map_Ns untuk komponen highlight, map_d adalah sebuah alpha teksture map, dan bump bila ketiganya tidak

ada maka teksture map_bum akan dipanggil.

1. Buka File .obj menggunakan aplikasi text editor. Contoh aplikasi text editor yaitu notepad, notepad ++, sublime text.

```
# Blender v2.72 (sub 0) OBJ File: ''
# www.blender.org
#mtllib pohon1.mtl
mtllib pohon1_2.mtl
o leaves_leaves.002
v 0.140424 6.545578 -0.097314
v -0.016506 6.601730 -0.130777
v -0.037497 6.386647 -0.393263
v 0.119434 6.330494 -0.359801
v 0.273818 6.547428 -0.280166
v 0.225988 6.709633 -0.262794
v -0.099891 6.612866 -0.256514
v -0.052061 6.450661 -0.273885
v 0.154640 6.474280 -0.458149
v 0.242756 6.611789 -0.505339
v 0.117133 6.783337 -0.240027
v 0.029017 6.645828 -0.192838
v -0.061902 6.534091 -0.438503
v -0.011605 6.560382 -0.590750
v 0.257756 6.734327 -0.477667
v 0.207458 6.708037 -0.317420
v -0.069677 6.728718 -0.315212
v -0.180526 6.731639 -0.444068
v 0.066283 6.638350 -0.658504
v 0.177132 6.635429 -0.529648
v 0.113838 6.851040 -0.366119
v -0.027483 6.945006 -0.356168
v -0.185804 6.729089 -0.565726
v -0.044483 6.635124 -0.575677
v 0.174646 6.806018 -0.579443
v 0.168453 6.975551 -0.598400
v -0.149585 6.971699 -0.470244
v -0.143392 6.802165 -0.459289
v -0.011040 6.758009 -0.699642
v 0.080372 6.866620 -0.793170
v 0.095606 7.080804 -0.529554
v 0.004194 6.972193 -0.436027
v -0.184919 6.876003 -0.615526
v -0.194622 6.904865 -0.782777
v 0.130270 7.005079 -0.784330
v 0.139973 6.976217 -0.617079
v -0.106991 7.071668 -0.532613
```

2. Kemudian perhatikan baris dengan awalan 'mtllib', baris tersebut merupakan baris pemanggilan file .mtl. File .mtl merupakan file yang berisikan koordinat untuk peletakan material/teksture pada objek yang dibuat. Pada data export yang dilakukan blender sebelumnya akan menghasilkan data .obj dan .mtl

```
# Blender v2.72 (sub 0) OBJ File: ''
# www.blender.org
#mtllib pohon1.mtl
mtllib pohon1_2.mtl
o leaves_leaves.002
v 0.140424 6.545578 -0.097314
v -0.016506 6.601730 -0.130777
v -0.037497 6.386647 -0.393263
v 0.119434 6.330494 -0.359801
v 0.273818 6.547428 -0.280166
v 0.225988 6.709633 -0.262794
v -0.099891 6.612866 -0.256514
v -0.052061 6.450661 -0.273885
v 0.154640 6.474280 -0.458149
v 0.242756 6.611789 -0.505339
v 0.117133 6.783337 -0.240027
v 0.029017 6.645828 -0.192838
v -0.061902 6.534091 -0.438503
v -0.011605 6.560382 -0.590750
v 0.257756 6.734327 -0.477667
v 0.207458 6.708037 -0.317420
v -0.069677 6.728718 -0.315212
v -0.180526 6.731639 -0.444068
v 0.066283 6.638350 -0.658504
v 0.177132 6.635429 -0.529648
v 0.113838 6.851040 -0.366119
v -0.027483 6.945006 -0.356168
v -0.185804 6.729089 -0.565726
v -0.044483 6.635124 -0.575677
v 0.174646 6.806018 -0.579443
v 0.168453 6.975551 -0.598400
v -0.149585 6.971699 -0.470244
v -0.143392 6.802165 -0.459289
v -0.011040 6.758009 -0.699642
v 0.080372 6.866620 -0.793170
v 0.095606 7.080804 -0.529554
v 0.004194 6.972193 -0.436027
v -0.184919 6.876003 -0.615526
v -0.194622 6.904865 -0.782777
v 0.130270 7.005079 -0.784330
v 0.139973 6.976217 -0.617079
v -0.106991 7.071668 -0.532613
```

3. Bila tidak ada path pada baris mtllib, maka file .mtl berada 1 folder dengan file .obj, namun bila file .obj dan .mtl tidak terletak pada 1 folder maka path harus didefinisikan path untuk .mtl

```
# Blender v2.72 (sub 0) OBJ File: ''
# www.blender.org
#mtllib pohon1.mtl
mtllib C:/x/pohon1_2.mtl
o leaves_leaves.002
v 0.140424 6.545578 -0.097314
v -0.816506 6.601730 -0.130777
v -0.037497 6.386647 -0.393263
v 0.119434 6.330494 -0.359881
v 0.273818 6.547422 -0.280166
v 0.225988 6.709633 -0.262794
v -0.699891 6.612866 -0.256514
v -0.852861 6.450661 -0.273885
v 0.154646 6.474280 -0.458149
v 0.242753 6.611783 -0.505339
v 0.117133 6.783337 -0.240027
v 0.029017 6.645828 -0.192838
v -0.061902 6.534091 -0.430503
v -0.811605 6.566382 -0.590750
v 0.255756 6.734327 -0.477667
v 0.207458 6.708837 -0.317420
v -0.069677 6.728718 -0.315212
v -0.180526 6.731639 -0.444068
v 0.066283 6.638356 -0.658584
v 0.177132 6.635429 -0.529648
v 0.113834 6.851046 -0.366119
v -0.027483 6.945006 -0.356168
v -0.185804 6.729089 -0.565726
v -0.044483 6.635124 -0.575677
v 0.174646 6.806018 -0.579443
v 0.168453 6.975551 -0.598400
v -0.149585 6.971699 -0.470244
v -0.143392 6.802165 -0.459289
v -0.011044 6.758009 -0.699642
v 0.080372 6.866620 -0.793170
v 0.095606 7.080804 -0.529554
v 0.004194 6.972193 -0.436027
v -0.184919 6.876003 -0.615526
v -0.194622 6.904865 -0.782777
v 0.130270 7.005079 -0.784330
v 0.139973 6.976217 -0.617079
v -0.106991 7.071668 -0.537613
```

4. Pada file .mtl berisikan koordinat peletakan texture dan pengambilan teksture yang akan digunakan. Newmtl menunjukkan daerah objek yang digunakan pada saat kita membuat objek pada blender. Dan map_Kd merupakan baris pemanggilan teksture.

```
# Blender MTL File: 'None'
# Material Count: 2

newmtl Leaves.002
Ns 92.156863
Ka 0.000000 0.000000 0.000000
Kd 0.512000 0.512000 0.512000
Ks 0.500000 0.500000 0.500000
Ni 1.000000
d 1.000000
illum 2
map_Kd BarkDeciduous0143_5_S.jpg

newmtl Material.001
Ns 92.156863
Ka 0.000000 0.000000 0.000000
Kd 0.512000 0.512000 0.512000
Ks 0.500000 0.500000 0.500000
Ni 1.000000
d 1.000000
illum 2
map_Kd BarkDeciduous0143_5_S.jpg
```

5. Penggantian texture pada .obj dapat dilakukan dengan pengeditan baris map_Kd pada file .mtl bila file texture dan mtl pada 1 folder maka hanya dituliskan nama file dan tipe. Bila tidak dalam satu folder maka path harus dituliskan.

```

# Blender MTL File: 'None'
# Material Count: 2

newmtl Leaves.001
Ns 92.156863
Ka 0.000000 0.000000 0.000000
Kd 0.512000 0.512000 0.512000
Ks 0.500000 0.500000 0.500000
Ni 1.000000
d 1.000000
illum 2
map_Kd C:/x/Leaves0142_4_S_.png

newmtl Material.001
Ns 92.156863
Ka 0.000000 0.000000 0.000000
Kd 0.512000 0.512000 0.512000
Ks 0.500000 0.500000 0.500000
Ni 1.000000
d 1.000000
illum 2
map_Kd x/wood.png

```

Implementasi Code 'Tata Kota'

- Definisi library yang digunakan untuk membuat 'Tata Kota'. Object.h dan object.cpp merupakan file tambahan yang digunakan untuk mendefinisikan keseluruhan fungsionalitas/kemampuan objek.

```

#include <Windows.h>
#include <gl\GL.h>
#include <gl\GLU.h>
#include <gl\glut.h>
#include "glm.h"
#include <glm\glm.hpp>
#include <glm\gtc\type_ptr.hpp>
#include <stdio.h>
#include <cmath>
#include "Object.h"
#include <vector>
#include <map>
#include <iostream>
#define MIN_CAMDIST 5.0
#define MAX_CAMDIST 100.0
#define MAX_DERAJATZ 80.0
#define MIN_DERAJATZ 10.0
#define GROUND_CAST_ERROR -100.0f
#define MDIV 0.05

```

- Membuat struct untuk menyimpan objek dan teksture objek yang telah dibuat, yang nantinya objek tersebut akan dipanggil melalui keyboard function.

```

}typedef struct {
    float minX, maxX, minZ, maxZ;
} boundingBox_t;

}typedef struct {
    GLMmodel* models[10];
    int jumlahModel;
    int indexModel;
} modelCollection_t;

void nextModel() {
    indexModel = (indexModel + 1) % jumlahModel;
}

void prevModel() {
    indexModel = indexModel == 0 ? jumlahModel-1 : indexModel - 1;
}

GLMmodel* getModel() {
    return models[indexModel];
}

void setCollection(int jumlah, string pathList[10]) {
    jumlahModel = jumlah;
    for (int i=0; i<jumlahModel; i++) {
        models[i] = glmReadOBJ(pathList[i].c_str());
    }
    indexModel = 0;
}
} modelCollection_t;

```

3. Fungsi untuk mendefinisikan penggunaan light pada ‘Tata Kota’. Light yang digunakan adlaah difuse, ambient dan postion.

```

void light(){
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_NORMALIZE);
    GLfloat lightColor0[] = {1.0f, 1.0f, 1.0f, 1.0f};
    GLfloat lightPos0[] = {600.0f, 3.0f, 400.0f, 1.0f};
    glLightfv(GL_LIGHT0, GL_DIFFUSE, lightColor0);
    glLightfv(GL_LIGHT0, GL_POSITION, lightPos0);
    glLightfv(GL_LIGHT0, GL_AMBIENT, lightPos0);
    //glDisable(GL_LIGHTING);
}

```

4. Fungsi untuk memilih objek 3d/click-on dengan menggunakan mouse function.

```

vec3 cameraRaycast(int x, int y) {
    //Referensi: http://www.antongerdelen.net/opengl/raycasting.html
    //Device Coordinate -> Normalized Device Coordinate
    float mx = 2.0f * (float)x / wwidth - 1.0f;
    float my = 1.0f - 2.0f * (float)y / wheight;
    vec3 ray_nds = vec3(mx, my, 1.0f);

    //Normalized Device Coordinate -> Homogenous Clip Coordinate
    vec4 ray_clip = vec4(ray_nds.x, ray_nds.y, -1.0f, 1.0f);

    //Homogenous Clip Coordinate -> Camera Coordinate
    GLfloat projectionMatrixv[16];
    glGetFloatv(GL_PROJECTION_MATRIX, projectionMatrixv);
    mat4 projectionMatrix;
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            projectionMatrix[i][j] = projectionMatrixv[i * 4 + j];
        }
    }
    vec4 ray_cam = inverse(projectionMatrix) * ray_clip;
    ray_cam.z = -1.0f;
    ray_cam.w = 0.0f;

    //Camera Coordinate -> World Coordinate
    GLfloat viewMatrixv[16];
    glGetFloatv(GL_MODELVIEW_MATRIX, viewMatrixv);
    mat4 viewMatrix;
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            viewMatrix[i][j] = viewMatrixv[i * 4 + j];
        }
    }
    vec4 temp = inverse(viewMatrix) * ray_cam;
    vec3 ray_world = vec3(temp.x, temp.y, temp.z);
    ray_world = normalize(ray_world);
    return ray_world;
}

vec3 groundRaycast(int x, int y) {
    vec3 camray = cameraRaycast(x, y);
    float t = (float)(-cameraY) / camray.y;
    if (t > 0) {
        vec3 result = vec3((float)cameraX + t*camray.x, 0.0f, (float)cameraZ + t*camray.z);
        return result;
    }
    else {
        vec3 result = vec3(0.0f, GROUND_CAST_ERROR, 0.0f);
        return result;
    }
}

```

5. Fungsi untuk pengecekan collision

```

bool collisionCheck(Object *o1, Object *o2) {
    bool left;
    bool right;
    bool up;
    bool down;
    left = (o1->maxX + o1->x) < (o2->minX + o2->x);
    right = (o1->minX + o1->x) > (o2->maxX + o2->x);
    down = (o1->maxZ + o1->z) < (o2->minZ + o2->z);
    up = (o1->minZ + o1->z) > (o2->maxZ + o2->z);
    return (!left && !right && !down && !up);
}

bool pointCollisionCheck(Object *o1, GLfloat x, GLfloat z) {
    bool left;
    bool right;
    bool up;
    bool down;
    left = (o1->maxX + o1->x) < x;
    right = (o1->minX + o1->x) > x;
    down = (o1->maxZ + o1->z) < z;
    up = (o1->minZ + o1->z) > z;
    return (!left && !right && !down && !up);
}

```

6. Fungsi pergerakan kamera

```

void cameraPanLeft(double dist) {
    double degY = derajatY / 180.0 * M_PI;
    lookZ += dist * cos(degY);
    lookX += dist * sin(degY);
}

void cameraPanRight(double dist) {
    double degY = derajatY / 180.0 * M_PI;
    lookZ -= dist * cos(degY);
    lookX -= dist * sin(degY);
}

void cameraMoveForward(double dist) {
    double degY = (derajatY - 90.0) / 180.0 * M_PI;
    lookZ += dist * cos(degY);
    lookX += dist * sin(degY);
}

void cameraMoveBackward(double dist) {
    double degY = (derajatY - 90.0) / 180.0 * M_PI;
    lookZ -= dist * cos(degY);
    lookX -= dist * sin(degY);
}

```

7. Fungsi peletakan objek bila tidak terjadi collision

```

void placeMode(Object *o) {
    ghostObject = o;
    isPlacingObject = true;
    glutPostRedisplay();
}

```

8. Fungsi motion function digunakan untuk pergeseran posisi kamera bila mouse function dilakukan.

```

    ]void motionFunc(int x, int y) {
    } if (isRotatingCamera) {
        derajatY -= (x - offsetX) * cameraRotationRate;
        derajatZ += (y - offsetY) * cameraRotationRate;
        if (derajatZ > MAX_DERAJATZ)
            derajatZ = MAX_DERAJATZ;
        else if (derajatZ < MIN_DERAJATZ)
            derajatZ = MIN_DERAJATZ;
        offsetX = x;
        offsetY = y;
        glutPostRedisplay();
    }
    if (isPanningCamera) {
        double dX = (double)x - (double)offsetX;
        double dY = (double)y - (double)offsetY;
        cameraPanRight(cameraSpeed * -dX * MDIV);
        cameraMoveBackward(cameraSpeed * -dY * MDIV);
        offsetX = x;
        offsetY = y;
        glutPostRedisplay();
    }
}

```

9. Fungsi void display digunakan untuk fungsi pengaturan objek dan tampilan objek pada kamera.

```

void display() {
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();

    double degI = derajatI / 180.0 * M_PI;
    double degY = derajatY / 180.0 * M_PI;

    cameraX = cameraDistance * cos(degI);
    cameraY = cameraDistance * sin(degI);
    cameraI = -cameraX * sin(degY);
    cameraX = cameraX * cos(degY);

    cameraX += lockX;
    cameraY += lockY;
    cameraI += lockI;

    gluLookAt(cameraX, cameraY, cameraI, lookX, lookY, 0, 1, 0);

    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);

    //XXXXXX XXXXX
    glPushMatrix();
    glScaled(10.0, 1.0, 1.0);
    glColor3ub(255, 255, 255);
    glDraw(modelTanah, GLM_SMOOTH | GLM_TEXTURE);
    glPopMatrix();

    int size = objectList.size();
    for (int i = 0; i < size; i++) {
        if (isPlacingObject) {
            glColor4d(0.6, 1.0, 0, 0.5);
            glBegin(GL_POLYGON);
            glVertex3f(objectList[i].minX + objectList[i].x, 0.5, objectList[i].minZ + objectList[i].z);
            glVertex3f(objectList[i].maxX + objectList[i].x, 0.5, objectList[i].minZ + objectList[i].z);
            glVertex3f(objectList[i].maxX + objectList[i].x, 0.5, objectList[i].maxZ + objectList[i].z);
            glVertex3f(objectList[i].minX + objectList[i].x, 0.5, objectList[i].maxZ + objectList[i].z);
            glEnd();
        }
        glColor3ub(255, 255, 255);
        objectList[i].draw(GLM_SMOOTH | GLM_TEXTURE);
    }

    if (isPlacingObject) {
        if (isPlaceable)
            glColor4f(0.3, 1.0, 0.0, 0.5);
        else
            glColor4d(1.0, 0.5, 0.0, 0.5);
        glBegin(GL_POLYGON);
        glVertex3f(ghostObject->minX + ghostObject->x, 0.5, ghostObject->minZ + ghostObject->z);
        glVertex3f(ghostObject->maxX + ghostObject->x, 0.5, ghostObject->minZ + ghostObject->z);
        glVertex3f(ghostObject->maxX + ghostObject->x, 0.5, ghostObject->maxZ + ghostObject->z);
        glVertex3f(ghostObject->minX + ghostObject->x, 0.5, ghostObject->maxZ + ghostObject->z);
        glEnd();
        ghostObject->draw(GLM_SMOOTH | GLM_TEXTURE);
    }

    glFlush();
    glutSwapBuffers();
}

```

10. Fungsi ini untuk memanggil objek yang akan digunakan(pendefinisan objek)

```

void init() {
    glLineWidth(10.0f);
    glEnable(GL_TEXTURE_2D);
    glEnable(GL_POINT_SMOOTH);
    glHint(GL_POINT_SMOOTH_HINT, GL_DONT_CARE);
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LESS);
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    //glEnable(GL_COLOR_MATERIAL);
    //	glColorMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE);
    glClearColor(0, 0, 0, 0);

    //modelMap["gadung"] = glmReadOBJ();
    modelTanah = glmReadOBJ("C:/objfile/tanah.obj");

    modelCollection_t m;
    string pathList[10];
    pathList[0] = "C:/objfile/FP/balk/jalan/partigean11.obj";
    m.setCollection(1, pathList);
    modelList.push_back(m);

    pathList[0] = "C:/objfile/FP/balk/jalan/partigean21.obj";
    m.setCollection(1, pathList);
    modelList.push_back(m);

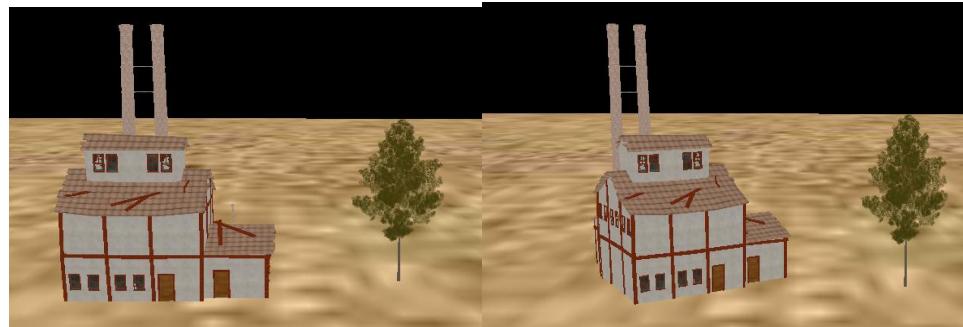
    pathList[0] = "C:/objfile/FP/balk/pohon/pohon1.obj";
    pathList[1] = "C:/objfile/FP/balk/pohon/pohon2.obj";
    m.setCollection(2, pathList);
    modelList.push_back(m);
    boundingBoxList.push_back({-1.0, 1.0, -1.0, 1.0});
    isPlacingObject = true;
    ghostObject = new Object(modelList[0].getModel(), 0, 0, 0, 1, 0, -1.0, 1.0, -1.0, 1.0);

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(45, 1200.0 / 800.0, 1.0, 1000.0);
}

```

Uji Coba dan Check Error

1. Uji coba Rotate



*) Perubahan arah pada rumah yang dirotate.

2. Uji Coba Collision



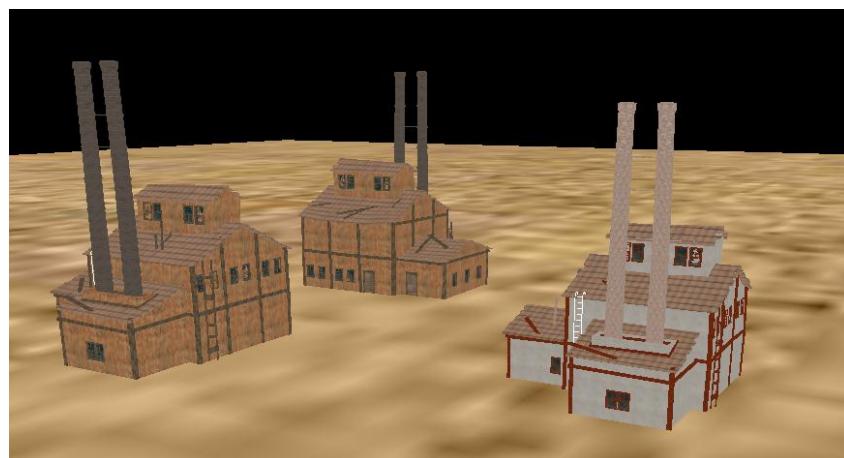
*) Warna merah menandakan objek tidak dapat diletakkan.

3. Uji Coba Scalling



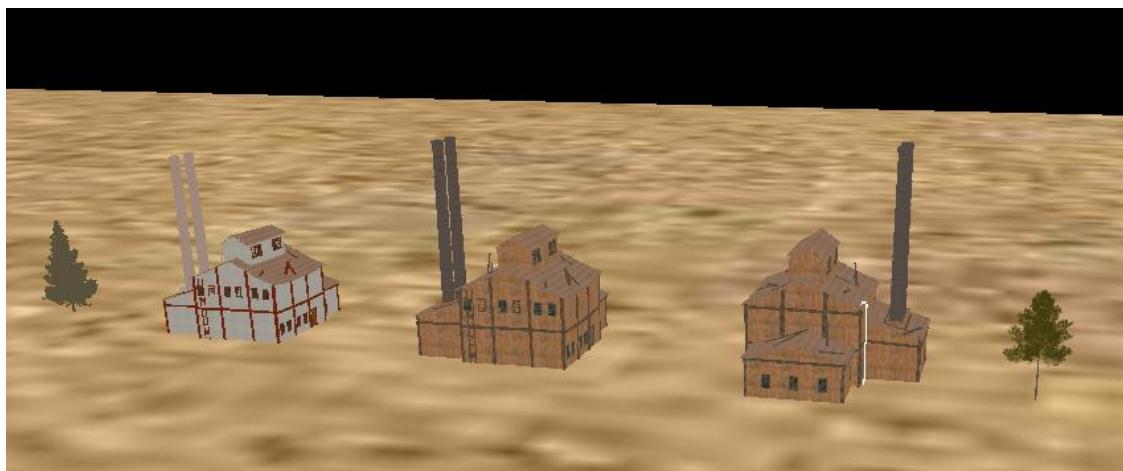
*) Dilakukan scaling pada objek pohon yang awalnya pohon berukuran lebih kecil dibandingkan dengan pabrik kemudian dilakukan scaling sehingga pohon berukuran sama dengan pabrik.

4. Uji Coba Perubahan texture



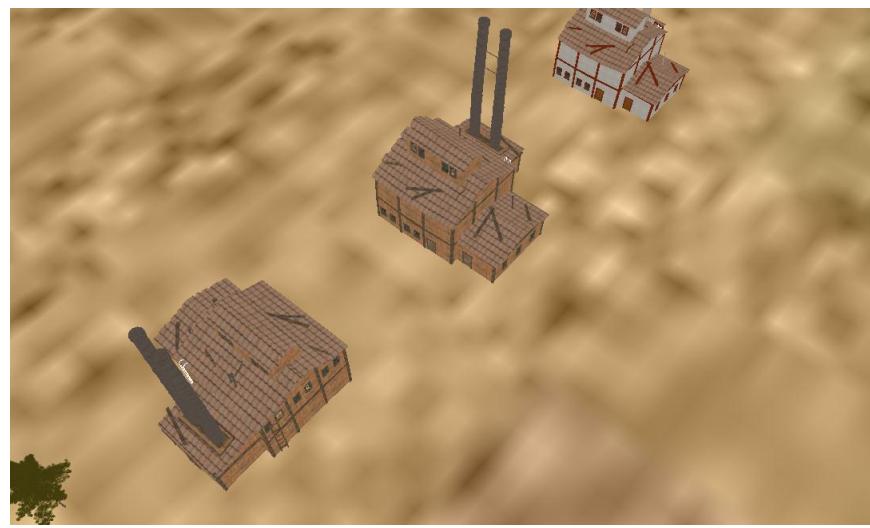
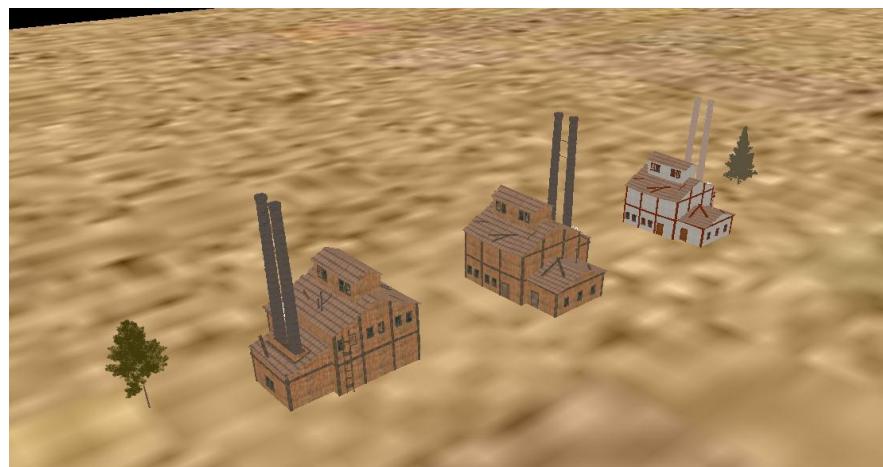
*) Pada objek factory dilakukan perubahan teksture objek. Perubahan tersebut dilakukan dengan menekan tombol +/-.

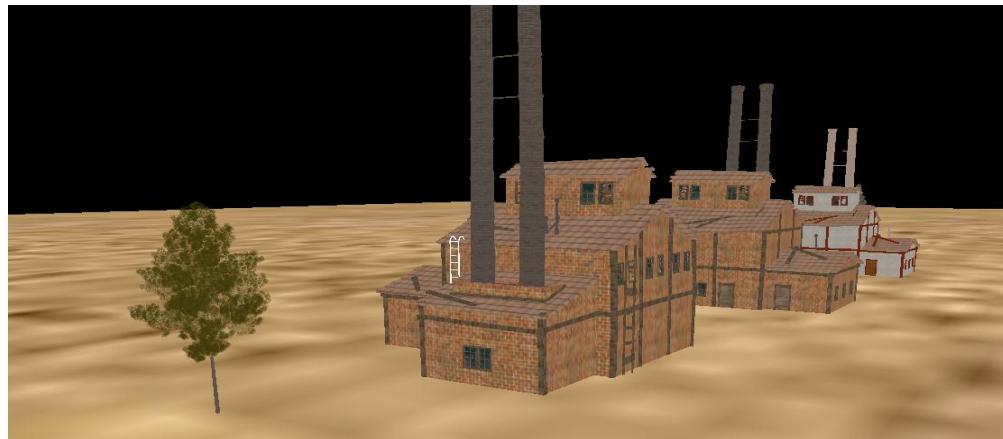
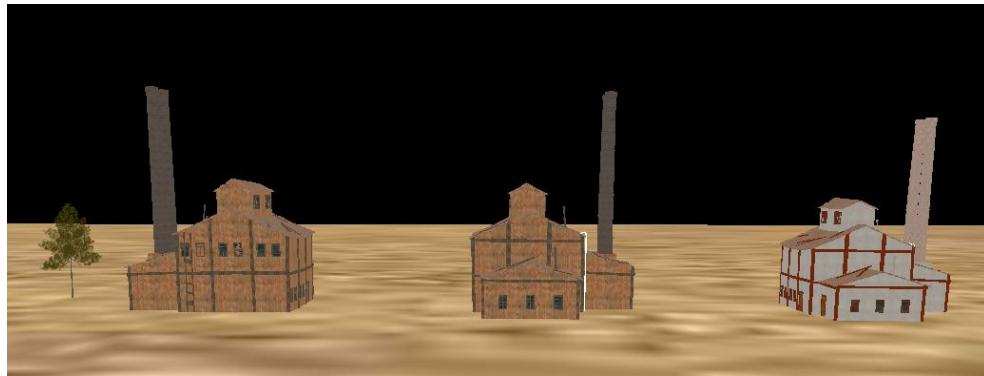
5. Uji Coba meload objek lebih dari 1.



*)Pada gambar diatas menggambarkan pabrik yang bersampingan dengan sebuah pohon.

6. Uji Coba gerak kamera





*)Pada gambar diatas menggambarkan pengambilan gambar pada posisi yang berbeda.

Cara Penggunaan Aplikasi Tata Kota

Keyboard

SPACE

tombol space digunakan untuk penambahan objek baru.

SHIFT

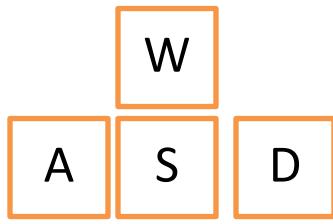
W

Tombol shift+w digunakan untuk Zoom Out pada aplikasi tata kota.

SHIFT

S

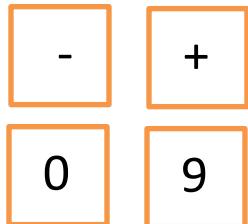
Tombol shift+s digunakan untuk Zoom In pada aplikasi tata kota.



Tombol W untuk kamera maju.
Tombol S untuk kamera mundur.
Tombol D untuk kamera geser kanan.
Tombol A untuk kamera geser kiri.

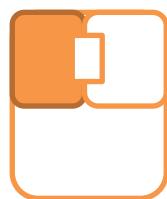


Tombol E digunakan untuk rotate
kanan.
Tombol Q digunakan untuk rotate kiri.
Tomol R digunakan untuk scale up.
Tombol F digunakan untuk scale
down.



Tombol – dan + digunakan untuk
mengganti tekstur.
Tombol 0-9 digunakan untuk
memilih objek yang akan ditaruh.

Mouse



Left Klik digunakan untuk memilih
objek dan menaruh objek.



Middle klik + hold untuk
menggunakan kamera secara bebas.



Right klik + hold untuk menggunakan kamera secara bebas.