

computer graphix

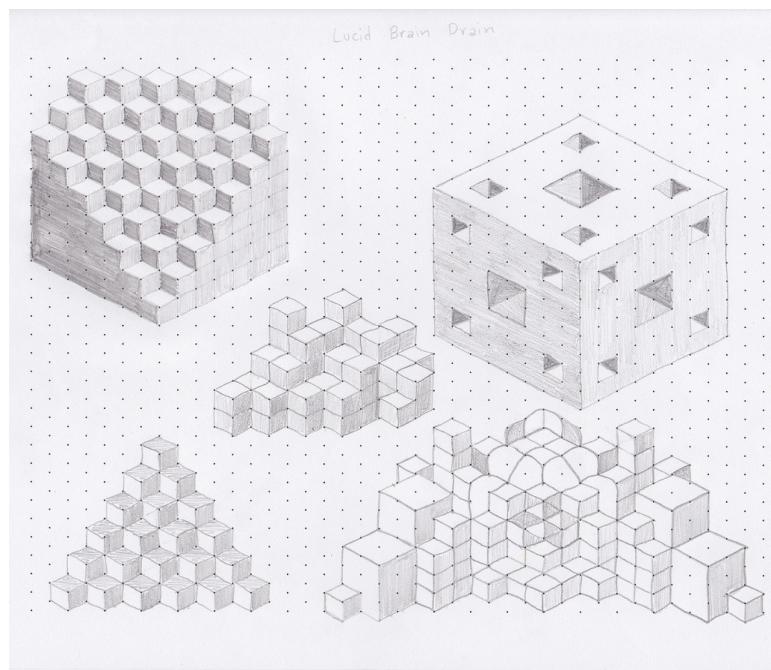
final project

MapMaker

```
2166 //ngSwitchScope, element, attr, ngSwitchController) {  
2167     var scope = element[0].$scope; //ngSwitchController.scope;  
2168     var value = attr.ngSwitch || attr.on;  
2169     var selectedTranscludes = [],  
2170     previousElements = [],  
2171     selectedElements = [],  
2172     selectedScopes = [];  
2173  
2174     scope.$watch(watchExpr, function ngSwitchWatchAction(value) {  
2175         var i, ii;  
2176         for (ii = 0, ii = previousElements.length; i < ii; ++i) {  
2177             previousElements[i].remove();  
2178         }  
2179         previousElements.length = 0;  
2180  
2181         for (i = 0, ii = selectedScopes.length; i < ii; ++i) {  
2182             var selected = selectedElements[i];  
2183             selectedScopes[i].$destroy();  
2184             previousElements[i] = selected;  
2185             $animate.leave(selected, function() {  
2186                 previousElements.splice(i, 1);  
2187             });  
2188         }  
2189  
2190         selectedElements.length = 0;  
2191         selectedScopes.length = 0;  
2192  
2193         if ((selectedTranscludes = ngSwitchController.cases['!' + value] || ngSwitchController.cases[value])) {  
2194             scope.$eval(attr.change);  
2195             forEach(selectedTranscludes, function(selectedTransclude) {  
2196                 var selectedScope = scope.$new();  
2197                 selectedScopes.push(selectedScope);  
2198             });  
2199         }  
2200     });  
2201 }
```

we started out with a basic idea of wanting to combine what we collectively learned in this course into one thing. although it seemed like not much it all added up into this simple but awesome project. the starting idea was bigger than the one we have right now. we thought of making a small game where the player needs to move a ball over isometric shapes and reach a goal to complete the level. although we met many other hurdles there was one hurdle in particular that led to making this project

and it's creating the levels that players play on we wanted a simple, easy and fast process to create those levels and it doesn't have to require code every time and maybe the player



can create the level himself. introducing mapmaker. it's basically all the description above.

How to use IT

W , S foreword and back

A , D right and left

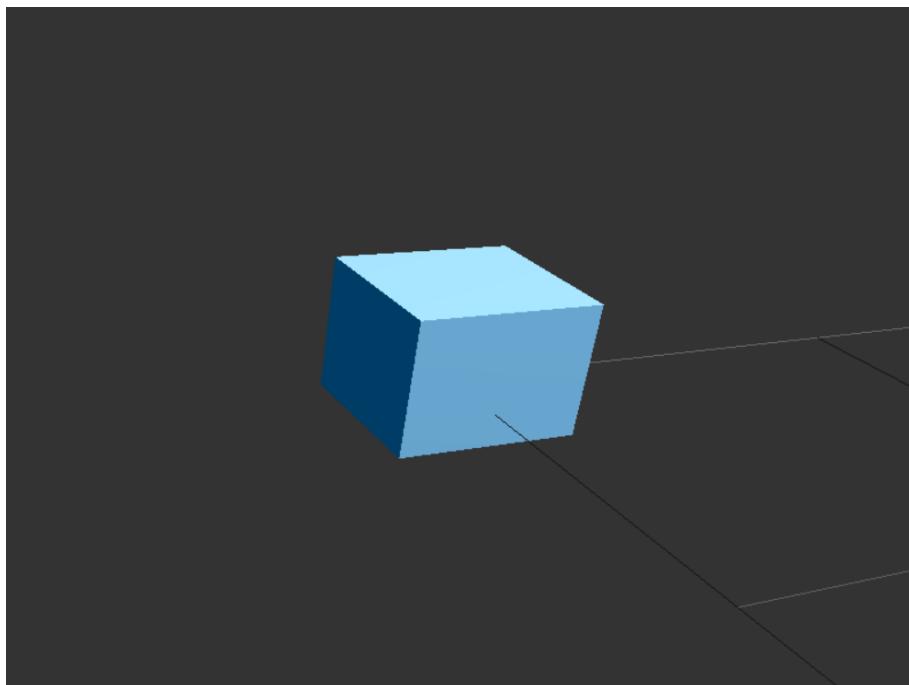
Q , Z up and down spacebar register a vertex

```
void thecube()
```

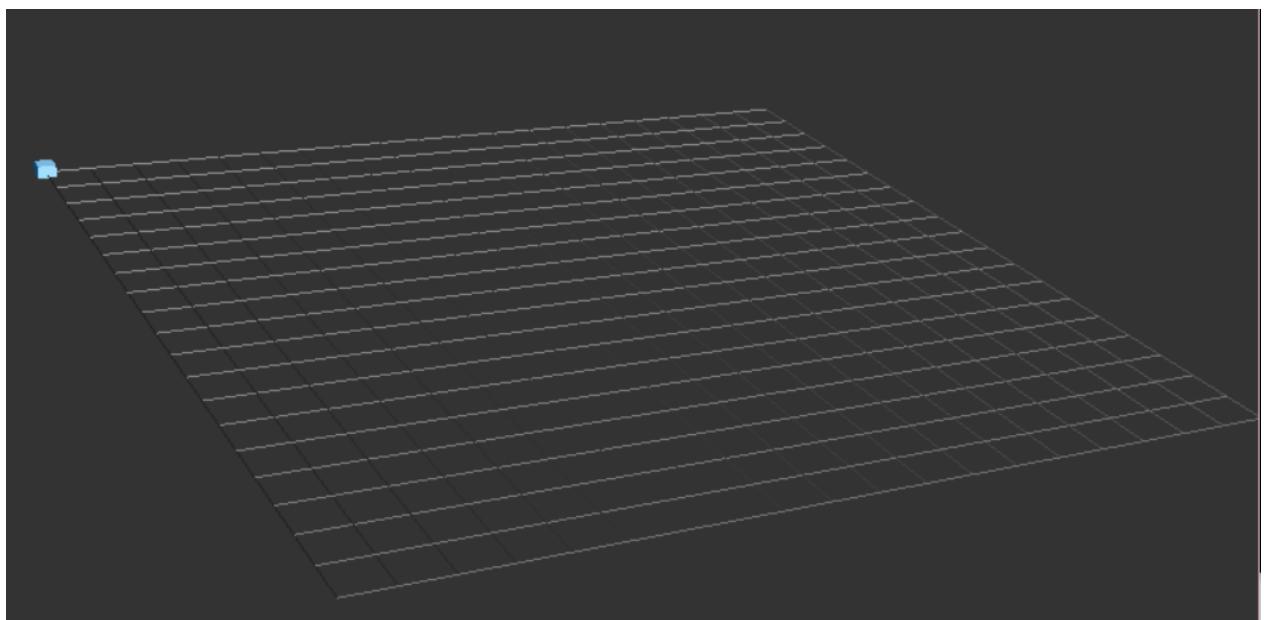
we start by making a cube that is gonna be the cursor of the user interface. this cube can be translated by user input . and we rotated it to look more like a cube

```
void drawGrid()
```

this is a function that we found necessary to help the user distinguish where is the cube's current position.



we translated the cube and the grid to look better for the viewr



```
void keyboard(unsigned char key, int x, int y).  
a function to move the cube in X,Y,Z directions  
and to add vertices to the quad by using spacebar
```

struct Quads

is a structure where we store all the coordinates of the vertices, state is to know which vertex are the user in now,

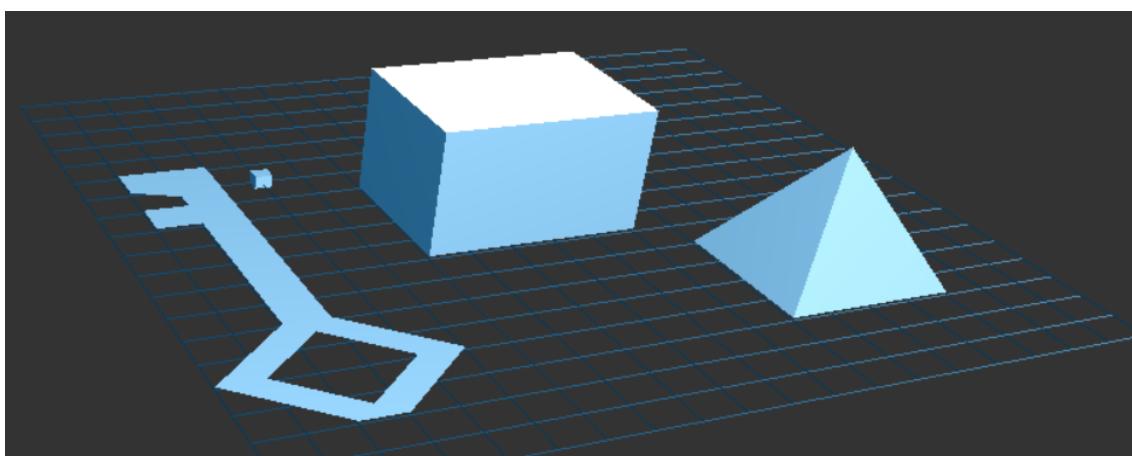
```
struct Quads  
{  
    int x1, y1, z1, x2, y2, z2, x3, y3, z3, x4, y4, z4;  
    //float r, g, b;  
    int state;  
    int total;  
}; Quads Q[100];
```

void addQuad()

is a function to make and store the quad and by pressing spacebar the user register the vertex for that quad and increment the state to make the other vertices

void drawQuads()

is a loop function that loops on the number of quads stored in the total variable and takes the input and draws the quad



`void display()`

it contains the `drawGrid()` and `drawQuad()` functions as well as the rotation and translation of the whole cube and grid to have a better view.

`void myinit()`

it has all the lighting information that the program uses like ambient, diffusion and specular lighting

`void init()`

the function that initialize the perspective view

`void getnormal(int x1, int y1, int z1, int x2, int y2, int z2, int x3, int y3, int z3)`

is a function to get the unit vector of every quad so that the light can interact with it.

POSSIBLE and UPCOMING FEATURES

1-save and load

as per doctor recommendation adding a 3 dimensional matrix to save the vertices could help us in solving the save and load issue

2-`void sphere() / circle()`

adding and drawing a sphere inside the program

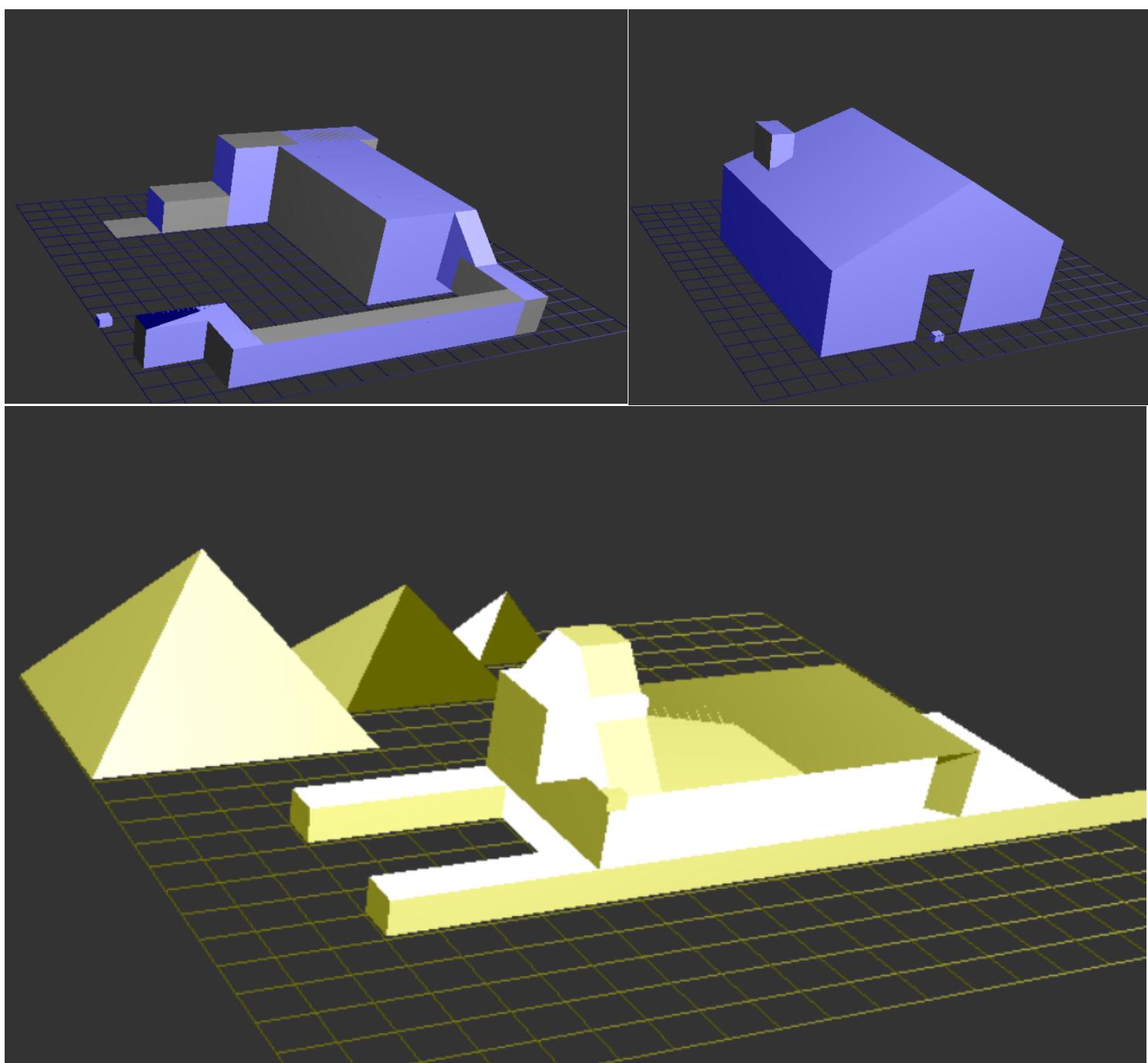
3-RGB coloring of the quads

although we have RGB code already but it's not playing well with the lighting. might fix that later

4- a unit vector for each vertex
instead of having a vector for each whole quad. it will
give a nice gradient lighting to skewed quads.

5-maybe make the original game after all
although there is a ton of hurdles and kinematics to
understand this was the original plan.

what you could do with mapmaker



BONUS

STAGES OF DEVELOPMENT OF LIGHT

