ΑΣΚΗΣΗ 1-Β OPENGL

ΣΤΟΙΧΕΙΑ ΟΜΑΔΑΣ: -Τσαπικούνη Γεωργία

-Κοτοφώλη Χριστίνα

ΗΜΕΡΟΜΗΝΙΑ: 18/11/2022

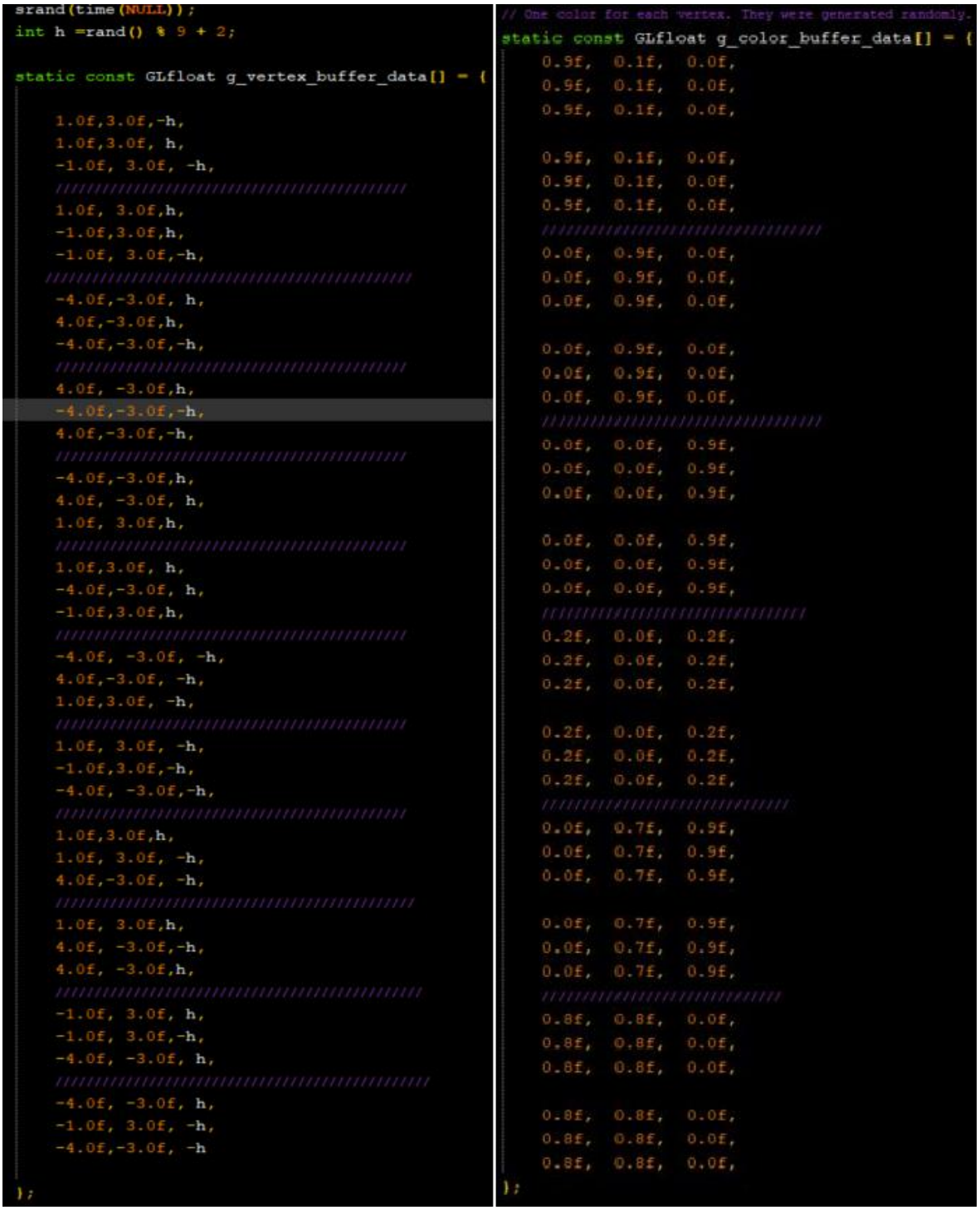
ΠΕΡΙΓΡΑΦΗ ΕΡΓΑΣΙΑΣ

- Ερώτημα i

Αρχικά στο πρώτο ερώτημα μας ζητείτε να ανοίγει ένα βασικό παράθυρο 1000x1000 και να εμφανίζεται ως τίτλος «Εργασία 1Β- τραπεζοειδές πρίσμα». Αυτό υλοποιείται με την εντολή **''window = glfwCreateWindow(1000, 1000, "Ergasia 1B-trapezoeides prisma", NULL, NULL);''**.Το background το κάνουμε σκούρο μπλε **'' glClearColor(0.0f, 0.0f, 0.3f, 0.0f);''**.Τέλος , η εφαρμογή τερματίζει πατώντας το πλήκτρο space **''(glfwGetKey(window, GLFW_KEY_SPACE) != GLFW_PRESS && glfwWindowShouldClose(window) == 0);''**.Η ανάλυση αυτού του ερωτήματος πραγματοποιήθηκε στην προηγούμενη άσκηση.

- Ερώτημα ii

Σε αυτό το ερώτημα προσδιορίσαμε τις συντεταγμένες των κορυφών του ισοσκελούς τραπεζοειδούς πρίσματος με την βοήθεια του υπολογιστικού μέσου Geogebra 3D. Το ισοσκελές τραπέζιο της πάνω και κάτω βάσης αντίστοιχα έχει μήκος α=2 , b=8 και c=6, όπως ζητήθηκε στο ερώτημα. Επίσης το ύψος του πρίσματος h έχει προσδιοριστεί με γεννήτρια τυχαίων αριθμών στο [2,10] **''int h =rand() % 9 + 2;''**. Γνωρίζουμε ότι η κάθε πλευράς του πρίσματος αποτελείται από δύο τρίγωνα, συνεπώς για να δοθεί το ίδιο χρώμα σε κάθε πλευρά έχουμε δώσει ίδιο RGB ανά δύο τρίγωνα. Όλη αυτή η διαδικασία που περιγράψαμε φαίνεται αναλυτικά στο παρακάτω screenshot.



```
srand(time(NULL));
int h =rand() % 9 + 2;

static const GLfloat g_vertex_buffer_data[] = {

    1.0f,3.0f,-h,
    1.0f,3.0f, h,
    -1.0f, 3.0f, -h,
    ////////////////////////////////////////
    1.0f, 3.0f,h,
    -1.0f,3.0f,h,
    -1.0f, 3.0f,-h,
   ////////////////////////////////////////
    -4.0f,-3.0f, h,
    4.0f,-3.0f,h,
    -4.0f,-3.0f,-h,
    ////////////////////////////////////////
    4.0f, -3.0f,h,
    -4.0f,-3.0f,-h,
    4.0f,-3.0f,-h,
    ////////////////////////////////////////
    -4.0f,-3.0f,h,
    4.0f, -3.0f, h,
    1.0f, 3.0f,h,
    ////////////////////////////////////////
    1.0f,3.0f, h,
    -4.0f,-3.0f, h,
    -1.0f,3.0f,h,
    ////////////////////////////////////////
    -4.0f, -3.0f, -h,
    4.0f,-3.0f, -h,
    1.0f,3.0f, -h,
    ////////////////////////////////////////
    1.0f, 3.0f, -h,
    -1.0f,3.0f,-h,
    -4.0f, -3.0f,-h,
    ////////////////////////////////////////
    1.0f,3.0f,h,
    1.0f, 3.0f, -h,
    4.0f,-3.0f, -h,
    ////////////////////////////////////////
    1.0f, 3.0f,h,
    4.0f, -3.0f,-h,
    4.0f, -3.0f,h,
    ////////////////////////////////////////
    -1.0f, 3.0f, h,
    -1.0f, 3.0f,-h,
    -4.0f, -3.0f, h,
    ////////////////////////////////////////
    -4.0f, -3.0f, h,
    -1.0f, 3.0f, -h,
    -4.0f,-3.0f, -h

};
```

```
// One color for each vertex. They were generated randomly.
static const GLfloat g_color_buffer_data[] = {
    0.9f,  0.1f,  0.0f,
    0.9f,  0.1f,  0.0f,
    0.9f,  0.1f,  0.0f,

    0.9f,  0.1f,  0.0f,
    0.9f,  0.1f,  0.0f,
    0.9f,  0.1f,  0.0f,
    ////////////////////////////////
    0.0f,  0.9f,  0.0f,
    0.0f,  0.9f,  0.0f,
    0.0f,  0.9f,  0.0f,

    0.0f,  0.9f,  0.0f,
    0.0f,  0.9f,  0.0f,
    0.0f,  0.9f,  0.0f,
    ////////////////////////////////
    0.0f,  0.0f,  0.9f,
    0.0f,  0.0f,  0.9f,
    0.0f,  0.0f,  0.9f,

    0.0f,  0.0f,  0.9f,
    0.0f,  0.0f,  0.9f,
    0.0f,  0.0f,  0.9f,
    ////////////////////////////////
    0.2f,  0.0f,  0.2f,
    0.2f,  0.0f,  0.2f,
    0.2f,  0.0f,  0.2f,

    0.2f,  0.0f,  0.2f,
    0.2f,  0.0f,  0.2f,
    0.2f,  0.0f,  0.2f,
    ////////////////////////////////
    0.0f,  0.7f,  0.9f,
    0.0f,  0.7f,  0.9f,
    0.0f,  0.7f,  0.9f,

    0.0f,  0.7f,  0.9f,
    0.0f,  0.7f,  0.9f,
    0.0f,  0.7f,  0.9f,
    ////////////////////////////////
    0.8f,  0.8f,  0.0f,
    0.8f,  0.8f,  0.0f,
    0.8f,  0.8f,  0.0f,

    0.8f,  0.8f,  0.0f,
    0.8f,  0.8f,  0.0f,
    0.8f,  0.8f,  0.0f,
};
```

Εικόνα 1

- Ερώτημα iii

Αρχικά τοποθετούμε την κάμερα στο σημείο (10,50,0) , κοιτάει προς το σημείο που βρίσκεται το πρίσμα (0,0,0) με up vector (0,0,1).

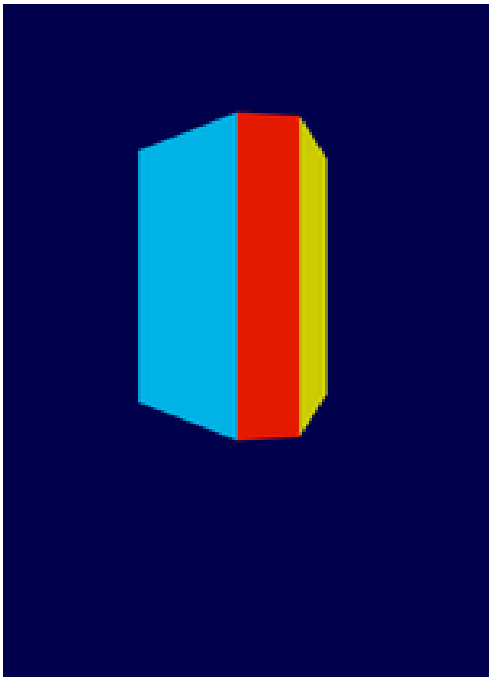
```
glm::mat4 Projection = glm::perspective(glm::radians(50.0f), 4.0f / 3.0f, 0.1f, 100.0f);

glm::mat4 View = glm::lookAt(
    glm::vec3(10,50,0),
    glm::vec3(0,0,0),
    glm::vec3(0,0,1)
);

glm::mat4 Model = glm::mat4(1.0f);
glm::mat4 MVP = Projection * View * Model;
```

Εικόνα 2

Αποτέλεσμα της τοποθέτησης της κάμερας σε αυτό το σημείο είναι η εικόνα 3 στην οποία φαίνονται και τα χρώματα ανά πλευρά που αναφέραμε στο ερώτημα ii.



Εικόνα 3

- Ερώτημα iv

Σε αυτό το ερώτημα δημιουργούμε ένα καινούριο matrix, τον οποίο όταν τον καλούμε μεγαλώνει ή μικραίνει το ύψος h του πρίσματος. Πιο αναλυτικά πατώντας το πλήκτρο <p> κάνουμε scale up, ενώ πατώντας το πλήκτρο <u> κάνουμε scale down.

```
glm::mat4 ProjectionMatrix = getProjectionMatrix();
glm::mat4 ViewMatrix = getViewMatrix();
glm::mat4 ModelMatrix = glm::mat4(1.0);
if (glfwGetKey(window, GLFW_KEY_P) == GLFW_PRESS) {

    ModelMatrix = glm::scale(ModelMatrix, glm::vec3(1.0f, 1.0f, 2.0f));

}
else if (glfwGetKey(window, GLFW_KEY_U) == GLFW_PRESS) {

    ModelMatrix = glm::scale(ModelMatrix, glm::vec3(1.0f, 1.0f, 0.5f));

}
glm::mat4 MVP = ProjectionMatrix * ViewMatrix * ModelMatrix;
```

Εικόνα 4

Στην παραπάνω εικόνα πραγματοποιείται scale up διπλασιάζοντας το ύψος h του πρίσματος όπως μας ζητείται από την άσκηση με την εντολή **''ModelMatrix = glm::scale(ModelMatrix, glm::vec3(1.0f, 1.0f, 2.0f));''** και scale down με την εντολή **''ModelMatrix = glm::scale(ModelMatrix, glm::vec3(1.0f, 1.0f, 0.5f));''** υποδιπλασιάζοντας έτσι το ύψος h.

- Ερώτημα v

Σε αυτό το ερώτημα υλοποιούμε την κίνηση της κάμερας γύρω από τον άξονα x και z καθώς και zoom in, zoom out με την χρήση συγκεκριμένων πλήκτρων.

```
glm::vec3 positionCam= glm::vec3(10.0f, 50.0f, 0.0f);
glm::vec3 frontDirectionCam = glm::vec3(1.0f,0.0f,0.0f);
glm::vec3 up = glm::vec3(0.0f, 0.0f, 1.0f);

const float speedCam = 0.3f;

void camera_function()
{
        glm::vec3 startCam = glm::vec3(0.0f, 0.0f, 0.0f);
        glm::vec3 direction = glm::normalize(positionCam - startCam);

        ProjectionMatrix = glm::perspective(glm::radians(100.0f), 1.0f / 1.0f, 0.1f, 100.0f);
        ViewMatrix = glm::lookAt(positionCam, direction, up);

        if (glfwGetKey(window, GLFW_KEY_X) == GLFW_PRESS)
            positionCam += glm::normalize(glm::cross(up, positionCam)) * speedCam;

        if (glfwGetKey(window, GLFW_KEY_W) == GLFW_PRESS)
            positionCam -= glm::normalize(glm::cross(up, positionCam)) * speedCam;

        if (glfwGetKey(window, GLFW_KEY_Z) == GLFW_PRESS)
            positionCam += glm::normalize(glm::cross(direction, frontDirectionCam)) * speedCam;
        if (glfwGetKey(window, GLFW_KEY_Q) == GLFW_PRESS)
            positionCam -= glm::normalize(glm::cross(direction, frontDirectionCam)) * speedCam;


        if (glfwGetKey(window, GLFW_KEY_KP_ADD) == GLFW_PRESS)
            positionCam -= speedCam * direction;
        if (glfwGetKey(window, GLFW_KEY_KP_SUBTRACT) == GLFW_PRESS)
            positionCam += speedCam * direction;

}
```

Εικόνα 5

Κάθε φορά που πατάμε ένα από τα πλήκτρα X,W,Z,Q,+,-, η θέση της κάμερας ενημερώνεται ανάλογα. Αν θέλουμε να κάνουμε zoom in ή zoom out προσθέτουμε ή αφαιρούμε το διάνυσμα κατεύθυνσης από το διάνυσμα θέσης κλιμακούμενο κατά κάποια τιμή ταχύτητας ‘’**positionCam += speedCam * direction;’’**. Αν θέλουμε να περιστραφούμε γύρω από τον άξονα x κάνουμε

εξωτερικό γινόμενο του up vector με την θέση της κάμερας κλιμακούμενο κατά κάποια τιμή ταχύτητας **''positionCam += glm::normalize(glm::cross(up, positionCam)) * speedCam;''** για να δημιουργήσουμε ένα σωστό διάνυσμα και κινούμαστε γύρω από τον άξονα x . Αντίστοιχα, για να περιστραφούμε γύρω από τον άξονα z κάνουμε εξωτερικό γινόμενο του διανύσματος κατεύθυνσης της κάμερας με το διάνυσμα που κοιτάει πάντα η κάμερα κλιμακούμενο κατά κάποια τιμή ταχύτητας **''positionCam += glm::normalize(glm::cross(direction, frontDirectionCam)) * speedCam;''.**

- Περιγραφή δυσκολιών

Τα σκέλη που δεν καταφέραμε να υλοποιήσουμε είναι στο ερώτημα α τους ελληνικούς χαρακτήρες στον τίτλο του παραθύρου και στο ερώτημα iv την διατήρηση της κλιμάκωσης ή της σμίκρυνσης του ύψους του πρίσματος.

ΠΛΗΡΟΦΟΡΙΕΣ ΣΧΕΤΙΚΑ ΜΕ ΤΗΝ ΥΛΟΠΟΙΗΣΗ

Το λειτουργικό σύστημα είναι windows και το περιβάλλον που υλοποιήσαμε την εργασία είναι Visual Studio x64.

ΑΞΙΟΛΟΓΗΣΗ ΛΕΙΤΟΥΡΓΙΑΣ ΤΗΣ ΟΜΑΔΑΣ

Η λειτουργία της ομάδας ήταν αποτελεσματική καθώς οργανώσαμε τα μέρη της άσκησης, έψαξε και δούλεψε ο καθένας ξεχωριστά και στην συνέχεια μαζί λύσαμε τις απορίες μας κα ολοκληρώσαμε την άσκηση.

ΑΝΑΦΟΡΕΣ

- Οι πηγές που αξιοποιήθηκαν κατά την εκπόνηση της άσκησης ήταν τα βιντεάκια από το εργαστήριο της Κυρίας Σταμάτη.
- https://learnopengl.com/Getting-started/Camera?fbclid=IwAR0BKLgs4QhDLS1CXE-zdD1IMUYd3AfggECxdHnluZesIqBlDSM4UXO4qx4
- http://www.opengl-tutorial.org/
- https://www.geogebra.org/3d?lang=el