

Όνοματεπώνυμο	Νικόλαος Γέροντας
Αριθμός Μητρώου	1092813

## Περιεχόμενα

Lab - 04.....	1
HOMEWORK 1.....	1
HOMEWORK 2.....	2
HOMEWORK 3.....	3

## Lab - 04

### HOMEWORK 1

Χρωματίστε τις κορυφές της καρδιάς σύμφωνα με την απόστασή τους από το επίπεδο.

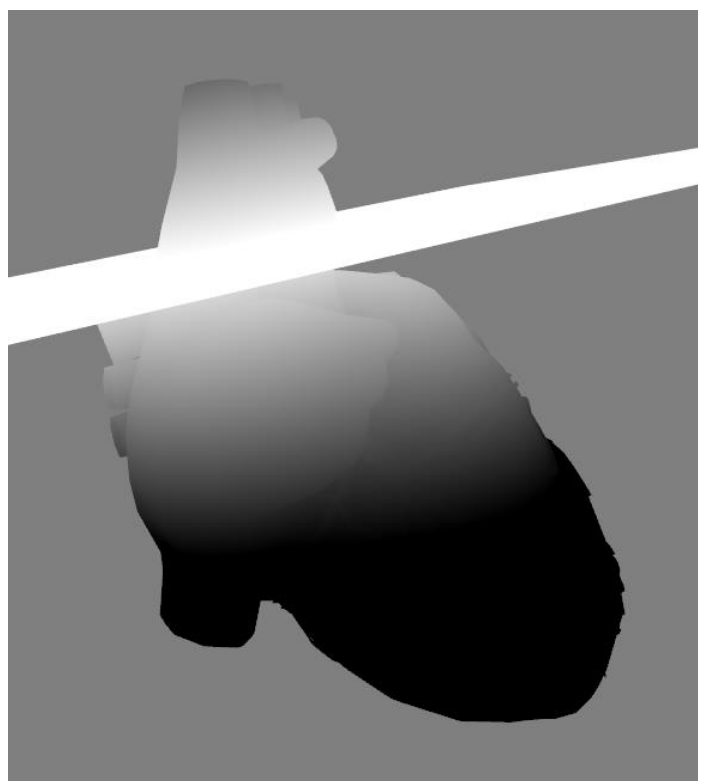
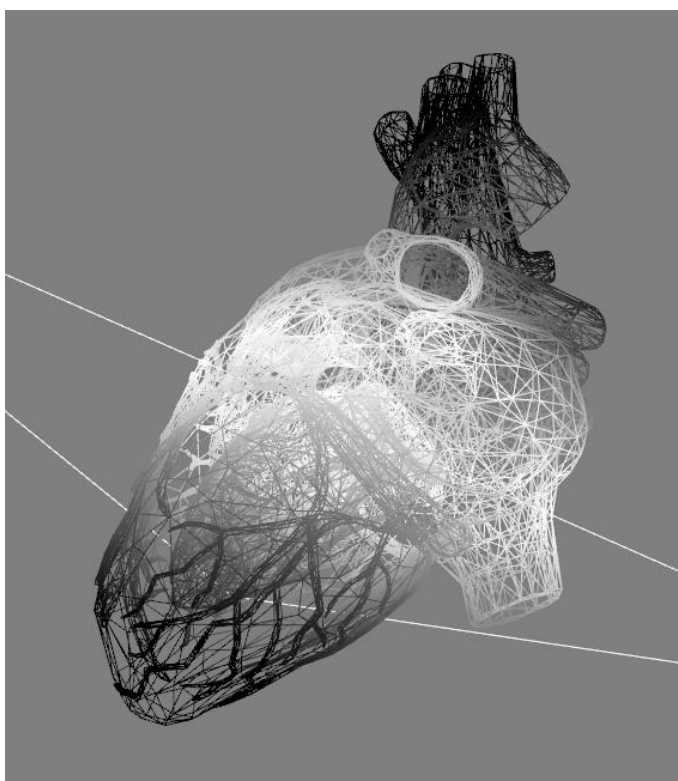
#### Απάντηση:

Η απόσταση ενός σημείου  $P = (x_1, y_1, z_1)$  από το επίπεδο  $Ax + By + Cz + D = 0$  δίνεται από τον τύπο:

$$d = \frac{|Ax_1 + By_1 + Cz_1 + D|}{\sqrt{A^2 + B^2 + C^2}}$$

Στον fragment shader, υπολογίζουμε αυτή την απόσταση και την θέτουμε στη μεταβλητή  $t$ . Χρησιμοποιούμε την τιμή αυτή για να καθορίσουμε το χρώμα του σημείου: όσο μεγαλύτερη είναι η απόσταση (ενός σημείου από το επίπεδο), τόσο πιο σκούρα/μαύρη γίνεται η απόχρωσή του. Αντίθετα, τα σημεία που βρίσκονται πολύ κοντά στο επίπεδο εμφανίζονται λευκά.

[https://mathinsight.org/distance\\_point\\_plane](https://mathinsight.org/distance_point_plane)



## HOMEWORK 2

Δημιουργήστε ένα νέο επίπεδο, το οποίο:

- i. Θα χρωματίσετε με κόκκινο χρώμα για να το ξεχωρίσετε από το αρχικό,
- ii. Θα πρέπει να χειρίζεστε όπως το πρώτο επίπεδο (translation/rotation).
- iii. Μη ξεχάσετε να προσθέσετε τους shaders του νέου επιπέδου (υπάρχουν ήδη στο φάκελο του lab04) στο αρχείο "CMakeLists.txt" και να κάνετε rebuild το project, ώστε να προστεθούν στο solution της άσκησης.

### Απάντηση:

i. Για να εμφανίζεται το νέο επίπεδο αποκλειστικά με κόκκινο χρώμα, δημιουργώ 2<sup>ο</sup> shader program (plane2ShaderProgram) με τα shaders plane.vertexshader και plane.fragmentshader. Το fragment shader του νέου προγράμματος επιστρέφει σταθερά vec4(1.0, 0.0, 0.0, 1.0), άρα ό,τι σχεδιάζεται με αυτό είναι πάντα κόκκινο. Στο mainLoop() κάνω **glUseProgram(plane2ShaderProgram)** **μόνο πριν το plane2->draw()**, ώστε το 2<sup>ο</sup> επίπεδο να ζωγραφίζεται με το κόκκινο shader, και στη συνέχεια επανέρχομαι στο αρχικό shaderProgram (με glUseProgram(shaderProgram)) για να σχεδιάσω το 1<sup>ο</sup> επίπεδο και το heart.obj με το κανονικό shader του εργαστηρίου.

ii. Όπως και για το 1<sup>ο</sup> επίπεδο, ορίζονται κατάλληλες global μεταβλητές για το 2<sup>ο</sup> επίπεδο, ώστε να ελέγχονται η κατακόρυφη μετατόπιση, η γωνία περιστροφής και ο **συντελεστής διαχωρισμού** (plane2Y, plane2Angle, **detachCoeff2 {HOMEWORK 3}**). Οι μεταβλητές αυτές ενημερώνονται στο pollKeyboard() με τα πλήκτρα του NumPad (8/5 για μετακίνηση στο  $\pm y$ , 4/6 για περιστροφή γύρω από τον άξονα z **και 7/9 για τον διαχωρισμό {HOMEWORK 3}**) και στη συνέχεια χρησιμοποιούνται στο mainLoop() για τον υπολογισμό του model matrix και των αντίστοιχων συντελεστών του κόκκινου επιπέδου που στέλνονται στους shaders.

iii. Πραγματοποιήθηκε κατά την διάρκεια του εργαστηρίου:

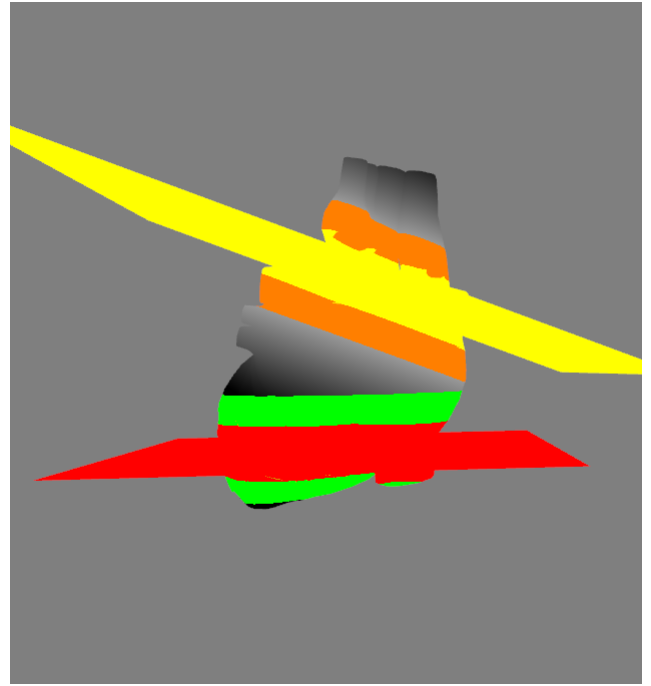
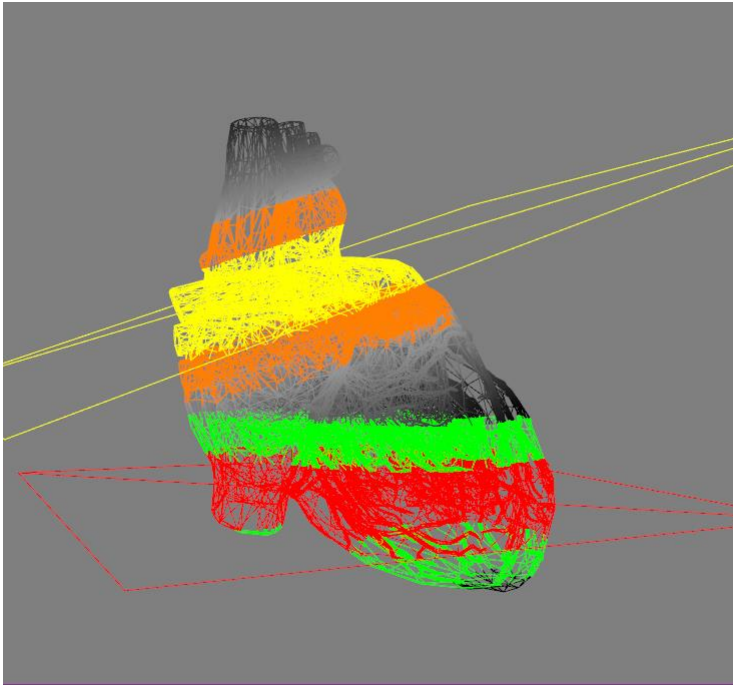
```
#####
# lab04
add_executable(lab04
    lab04/lab.cpp

    common/util.cpp
    common/util.h
    common/shader.cpp
    common/shader.h
    common/camera.cpp
    common/camera.h
    common/model.cpp
    common/model.h
    common/texture.cpp
    common/texture.h

    lab04/Shader.fragmentshader
    lab04/Shader.vertexshader

    lab04/plane.fragmentshader
    lab04/plane.vertexshader
)

[...]
```



### HOMEWORK 3

Από τις τομές του μοντέλου με τα δύο επίπεδα προκύπτουν κάποια κομμάτια. Τα κομμάτια πρέπει να μπορούν να κινούνται ανεξάρτητα το ένα από το άλλο, όπως στο Task 4.

#### Απάντηση:

Για να κινούνται ανεξάρτητα τα κομμάτια που προκύπτουν από τις τομές με τα 2 επίπεδα, ακολούθησα τη λογική του Task 4, επεκτείνοντάς την όμως για 2 επίπεδα. Για κάθε επίπεδο ορίζεται ξεχωριστό διάνυσμα διαχωρισμού στον shader (`detachementDisplacement`, `detachementDisplacement2`), το οποίο υπολογίζεται στο `lab.cpp`. Στον shader υπολογίζω την απόσταση κάθε σημείου του μοντέλου από τα 2 επίπεδα και, ανάλογα με το πρόσημο, το κάθε τμήμα του μοντέλου μετατοπίζεται κατά  $\pm \text{detachementDisplacement}$  ή  $\pm \text{detachementDisplacement2}$  πάνω στην αντίστοιχη κάθετη του επιπέδου. Οι συντελεστές διαχωρισμού τροποποιούνται από το πληκτρολόγιο (ξεχωριστά ή ταυτόχρονα με NumPad 0/1), με αποτέλεσμα τα κομμάτια που βρίσκονται σε διαφορετικές πλευρές των επιπέδων να κινούνται ανεξάρτητα. Τέλος, με τη μεταβλητή `isPlane` εξασφαλίζω ότι ο διαχωρισμός εφαρμόζεται μόνο στο μοντέλο της καρδιάς και όχι στα ίδια τα επίπεδα!

