Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Γραφική με Υπολογιστές

Εργασία #1: Πλήρωση τριγώνων

Κούτση Χριστίνα

ΑΕΜ: 9871

cvkoutsi@ece.auth.gr

Απρίλιος, 2023

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

Η παρούσα εργασία υλοποιεί αλγορίθμους πλήρωσης τριγώνων με σκοπό τον χρωματισμό εικόνων. Συγκεκριμένα, υλοποιούνται δύο μέθοδοι πλήρωσης τριγώνων, η μέθοδος flats shading και η μέθοδος gourauds shading.

Η εργασία αυτή υλοποιήθηκε σε Python και ο κώδικας χωρίζεται σε 4 αρχεία python:

1. helpers.py
2. render.py
3. demo_flats.py
4. demo_gourauds.py

Κεφάλαιο 2

Βοηθητικές Συναρτήσεις

Για την υλοποίηση των μεθόδων flats shading και gourauds shading θα πρέπει πρώτα να δημιουργηθούν κάποιες βοηθητικές συναρτήσεις.

Η υλοποίηση της ενότητας αυτής σε κώδικα βρίσκεται στο αρχείο helpers.py

2.1 Γραμμική Παρεμβολή

Η συνάρτηση της γραμμικής παρεμβολής υπολογίζει την διανυσματική τιμή ενός διανύσματος V με συντεταγμένες [x,y] από παρεμβολή των διανυσμάτων V_1, V_2 με συντε-

ταγμένες $[x_1,y_1]$,$[x_2,y_2]$. Η τιμή του V μπορεί να υπολογιστεί με παρεμβολή στον άξονα x ή y ανάλογα με την τιμή του ορίσματος της συνάρτησης.

Συγκεκριμένα, η τιμή του V μπορεί να υπολογιστεί ως εξής:

- Με παρεμβολή στον άξονα x:

$$\left.\begin{aligned} \text{V1\_coefficient} &= \frac{|x_2 - x|}{|x_2 - x_1|} \\ \text{V2\_coefficient} &= \frac{|x_1 - x|}{|x_2 - x_1|} \end{aligned}\right\} \Rightarrow V = V_1 * \text{V1\_coefficient} + V_2 * \text{V2\_coefficient}$$

Η συνάρτηση που υλοποιεί τα παραπάνω σε κώδικα βρίσκεται στο αρχείο helpers.py με όνομα interpolate_vectors

2.2 Bresenham

Η συνάρτηση αυτή υλοποιεί τον αλγόριθμο του Bresenham για την χάραξη ευθειών σε διακριτό καμβά. Συγκεκριμένα, η συνάρτηση αυτή δέχεται δύο σημεία A και B με συντεταγμένες $[x_1,y_1]$ και $[x_2,y_2]$ και επιστρέφει ε τον Bresenham. Επιπλέον, η συνάρτηση δέχεται και μια μεταβλητή που καθορίζει τον άξονα κατά τον οποίο θα υλοποιηθεί ο αλγόριθμος. Παρακάτω δίνεται ο ψευδοκώδικας για την υλοποίηση του αλγορίθμου Bresenham στον άξονα x και y

Algorithm 1 Bresenham in y axis

```
deltaY ← 2 × (y2 − y1)
deltaX ← 2 × (x2 − x1)
f ← −deltaX + deltaY/2
x ← x1
for y ← y1 to y2 do
    if f < 0 then
        if x1 < x2 then
            x ← x + 1
        else
            x ← x - 1
        end if
        f ← f + deltaY
    end if
    f ← f - deltaX
line.insert(x,y)
end for
```

Algorithm 2 Bresenham in x axis

```
deltaY ← 2 × (y2 − y1)
deltaX ← 2 × (x2 − x1)
f ← −deltaY + deltaX/2
x ← x1
for x ← x1 to x2 do
    if f < 0 then
        if y1 < y2 then
            y ← y + 1
        else
            y ← y - 1
        end if
        f ← f + deltaX
    end if
    f ← f - deltaY
line.insert(x,y)
end for
```

Παρατήρηση: Στον παραπάνω κώδικα όπως και σε όλη την εργασία υιοθετείται η σύμβαση ότι το x παίρνει τιμές στην κάθετη διεύθυνση και αυξάνει από πάνω προς τα κάτω, ενώ το y παίρνει τιμές στην οριζόντια διεύθυνση και αυξάνει από τα αριστερά προς τα δεξιά. Επομένως άξονας x θεωρείται ο κατακόρυφος άξονας, ενώ άξονας y θεωρείται ο οριζόντιος άξονας

Η συνάρτηση που υλοποιεί τα πραραπάνω σε κώδικα βρίσκεται στο αρχείο helpers.py με όνομα bresenham

2.3 Χάραξη γραμμής

Η παρούσα συνάρτηση για δεδομένα σημεία A και B με συντεταγμένες $[x_1,y_1]$ και $[x_2,y_2]$ επιστρέφει έναν πίνακα με τα σημεία που ανήκουν στην ευθεία που τα ενώνει. Πραγματοποιεί έλεγχο για 3 περιπτώσεις:

1. Αν $x_1 = x_2$, επιστρέφει σημεία σε οριζόντια ευθεία
2. Αν $y_1 = y_2$, επιστρέφει σημεία σε κατακόρυφη ευθεία
3. Αν $x_1 \neq x_2$ και $y_1 \neq y_2$
   (α') Υπολογίζει την κλίση της ευθείας m
   (β') Αν $m \geq 1$, η κλίση της ευθείας είναι μεγαλύτερη από 45° και επιστρέφει σημεία σε ευθεία που υπολογίζεται με τον αλγόριθμο Bresenham στον άξονα y
   (γ') Αν $m \leq 1$, η κλίση της ευθείας είναι μικρότερη από 45° και επιστρέφει σημεία σε ευθεία που υπολογίζεται με τον αλγόριθμο Bresenham στον άξονα x

Η συνάρτηση που υλοποιεί τα πραραπάνω σε κώδικα βρίσκεται στο αρχείο helpers.py με

Κεφάλαιο 3

Πλήρωση τριγώνων

Η πλήρωση τριγώνων υλοποιεί δύο διαδικασίες

1. Εύρεση πλευρών τριγώνου και εσωτερικά σημεία
2. Ανάθεση χρώματος σε κάθε pixel

Δοκιμάζονται δύο μέθοδοι πλήρωσης τριγώνων, η μέθοδος flats shading και η μέθοδος gourauds shading.

3.1 Flats shading

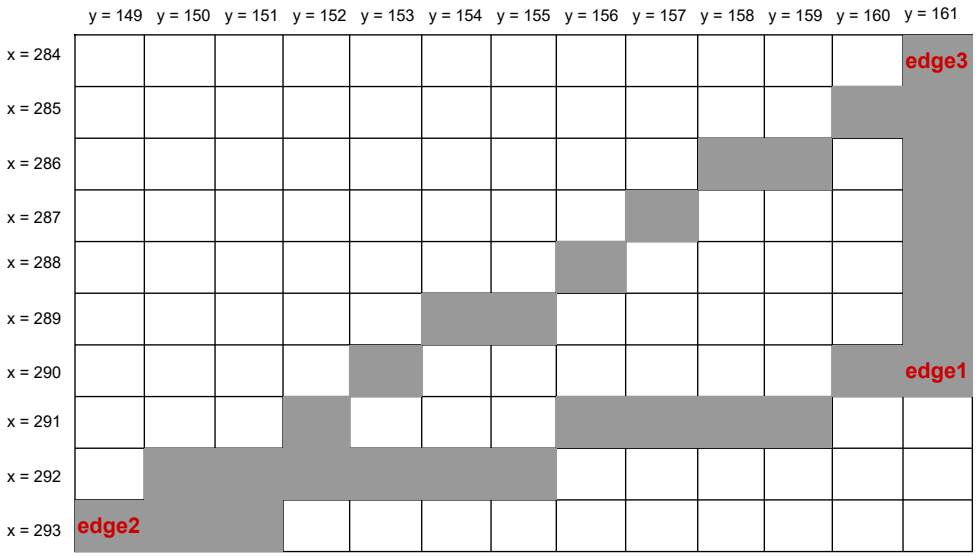
Η συνάρτηση που υλοποιεί το Flats shading λαμβάνει ως ορίσματα τις τρεις κορυφές των τριγώνων και τις τριάδες [r,g,b] στις κορυφές. Επιστρέφει έναν πίνακα με τις συντεταγμένες [x,y] των σημείων του τριγώνου και μία τριάδα [r,g,b] που ανατίθεται σε όλα τα pixels.

3.1.1 Έυρεση πλευρών τριγώνου

Για την εύρεση των πλευρών του τριγώνου:

1. Για κάθε συνδιασμό κορυφών καλείται η συνάρτηση *fill_line*. Έτσι δημιουργείται η πλευρά 1 από τις κορυφές 1 και 2, η πλευρά 2 από τις πλευρές 1 και 3 και η πλευρά 3 από τις κορυφές 2 και 3
2. Τα σημεία των πλευρών στοιβάζονται σε έναν πίνακα
3. Αφαιρούνται οι διπλοεγγραφές

Παρακάτω φαίνεται μια απεικόνιση των σημείων που προστέθηκαν με την παραπάνω διαδικασία για το τρίγωνο με id = 2444:



3.1.2 Εύρεση εσωτερικών σημείων τριγώνου

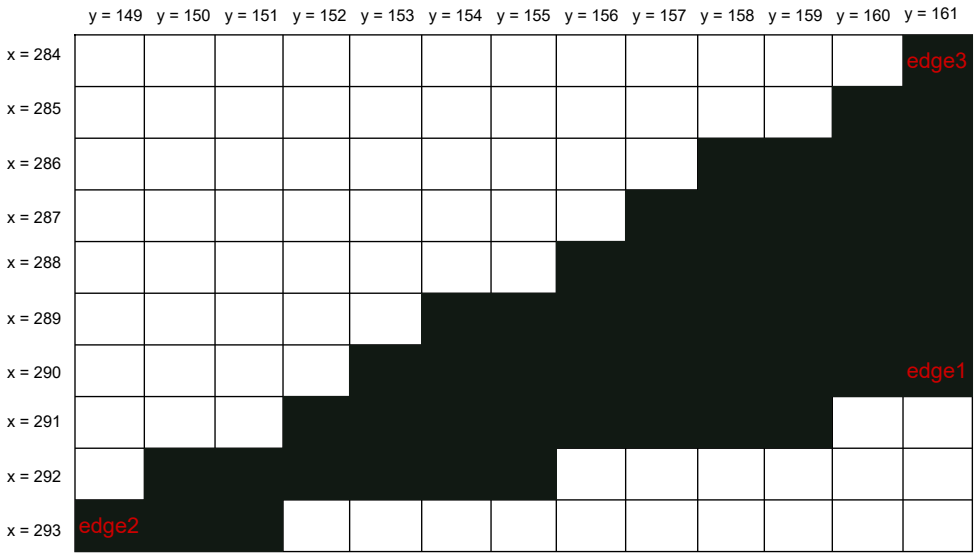
Για την εύρεση των εσωτερκών σημείων του τριγώνου χρησιμοποιούμε γραμμές σάρωσης. Συγκεκριμένα:

1. Βρίσκουμε την μεγαλύτερη και μικρότερη τιμή του x στο τρίγωνο (υψηλότερη και χαμηλότερη πλευρά)
   Για το παραπάνω τρίγωνο $x_{max} = 293$ και $x_{min} = 284$
2. Επαναλαμβάνουμε την παρακάτω διαδικασία για $x \in [x_{min}, x_{max}]$:
   (α') Ορίζουμε γραμμή σάρωσης x_{scan} = x
   (β') Αν η γραμμή δεν περιέχει καθόλου σημεία που να ανήκουν σε πλευρές, συνεχίζουμε στην επόμενη γραμμή
   (γ') Αν η γραμμή περιέχει παραπάνω από δύο σημεία που να ανήκουν σε πλευρες, δηλαδή συνεχόμενες γραμμές, κρατάμε μόνο τα μη διαδοχικά σημεία.
   Για την γραμμή $x = 291$ του παραπάνω τριγώνου κρατάμε μόνο τα σημεία [291,152] και [291,156]
   (δ') Για τα σημεία των πλευρών που προέκυψαν, που θα είναι 2, αναθέτουμε στο τρίγωνο όλα τα ενδιάμεσα
   Για την γραμμή $x = 291$ του παραπάνω τριγώνου προσθέτουμε τα σημεία [291,153], [291,154], [291,155], ενώ για την γραμμή $x = 285$ δεν προσθέτουμε κανένα σημείο

3.1.3 Εύρεση χρώματος

Για την ανάθεση χρώματος σε τρίγωνο με flat shading βρίσκουμε την μέση τιμή χρώματος των κορυφών τριγώνουν και την αναθέτουμε σε όλα τα σημεία του τριγώνου.

Μετά την διαδικασία εύρεσης εσωτερικών σημείων τριγώνου και ανάθεσης του μέσου χρώματος των πλευρών, το τρίγωνο με id = 2444 έχει την εξής μορφή:



Η συνάρτηση που υλοποιεί τα παραπάνω σε κώδικα βρίσκεται στο αρχείο render.py με όνομα flats

3.2 Gourauds shading

3.2.1 Εύρεση πλευρών τριγώνου και χρώματος στις πλευρές

Για να βρούμε τις πλευρές του τριγώνου και το χρώμα των πλευρών:

1. Αρχικοποιούμε τις πλευρές με την συνάρτηση *fill_line*, με την οποία δημιουργουμε:

   - Πίνακα για την πλευρά 1 από τις κορυφές 1 και 2
   - Πίνακα για την πλευρά 2 από τις κορυφές 1 και 3
   - Πίνακα για την πλευρά 3 από τις κορυφές 2 και 3

   Επομένως για κάθε πλευρά έχουμε έναν πίνακα με τα pixels που ανήκουν σε αυτή.

2. Αρχικοποιούμε τρείς πίνακες τα χρώματα των τριών πλευρών με τις τριάδες [r,g,b] που αντιστοιχούν στο πρώτο και τελευταίο στοιχείο του πίνακα της κάθε πλευράς

3. Κάνουμε έλεγχο για πλευρές που περιέχουν μόνο ένα pixel και τις αφαιρούμε. Αν και οι τρεις πλευρές του τριγώνου περιέχουν ένα pixel επιστρέφουμε το pixel και την τριάδα [r,g,b] της τρίτης πλευράς

4. Κάνουμε έλεγχο για ίδιες πλευρές. Αν κάποια πλευρά είναι ίδια με κάποια άλλη τότε διαγραφολυμε τον πίνακα της μίας από τις δύο και τον αντίστοιχο πίνακα με χρώματα

5. Για τις πλευρές που κρατήθηκαν κάνουμε έλεγχο για ίδιο χρώμα στις δύο κορυφές. Αν μια πλευρά έχει ίδιο χρώμα στις δύο κορυφές της τότε για αυτή τη πλευρά κρατάμε το χρώμα αυτό και δεν κάνουμε γραμμική παρεμβολή

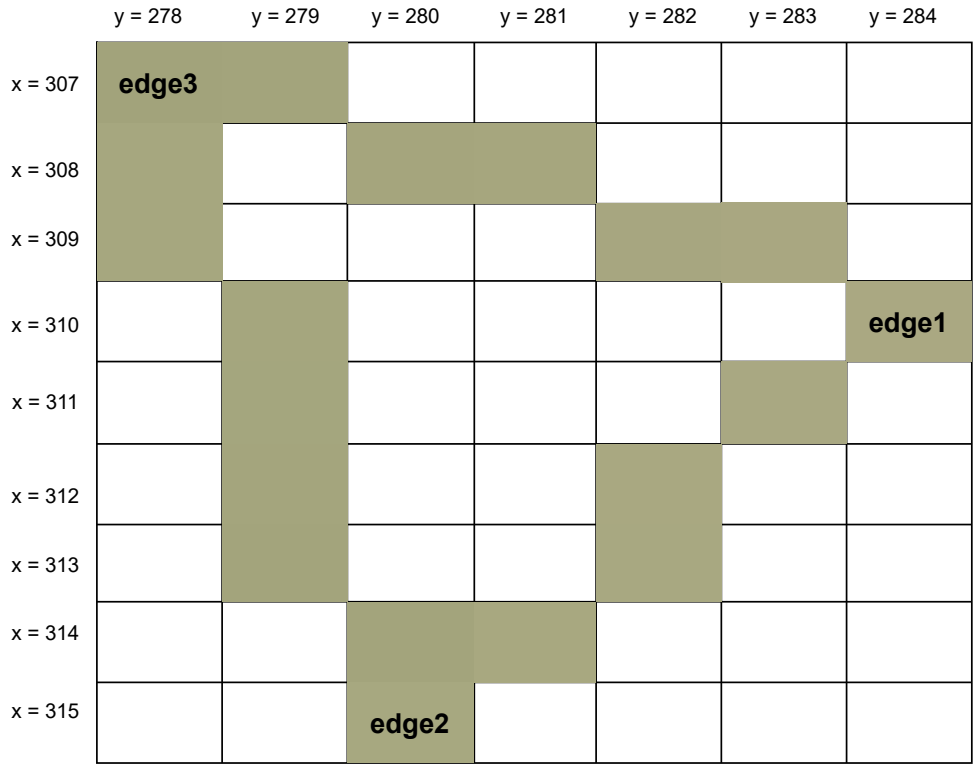
6. Για τις υπόλοιπες πλευρές βρίσκουμε το χρώμα στις θέσεις μεταξύ των κορυφών με γραμμική παρεμβολή:

   (α') Επιλέγουμε αν θα κάνουμε γραμμική παρεμβολή στον άξονα x ή στον άξονα y.

   - Υπολογίζουμε το $dx = x_{max} - x_{min}$ και το $dy = y_{max} - y_{min}$
   - Αν dx ≥ dy κάνουμε παρεμβολή στον άξονα x (κατακόρυφος άξονας)
   - Αν dx ≤ dy κάνουμε παρεμβολή στον άξονα y (οριζόντιος άξονας)

   (β') Για κάθε σημείο της πλευράς μεταξύ των κορυφών κάνουμε γραμμική παρεμβολή στον άξονα που προέκυψε από την παραπάνω διαδικασία

Παρακάτω φαίνεται μια απεικόνιση του τριγώνου με id = 3331 μετά την παραπάνω διαδικασία πλήρωσης των πλευρών του τριγώνου:



3.3 Εύρεση εσωτερικών σημείων τριγώνου και χρώματος στο εσωτερικό

Στη συνέχεια, αφού έχουμε βρει τα σημεία και τα χρώματα των πλερών βρίσκουμε και πάλι τα εσωτερικά σημεία με γραμμές σάρωσης και τα χρώματα των εσωτερικών σημείων με παρεμβολή στα σημεία των πλευρών:

1. Βρίσκουμε την μεγαλύτερη και μικρότερη τιμή του x στο τρίγωνο (υψηλότερη και χαμηλότερη πλευρά)
   Για το παραπάνω τρίγωνο $x_{max} = 307$ και $x_{min} = 315$

2. Επαναλαμβάνουμε την παρακάτω διαδικασία για x ∈ [x_{min},x_{max}]:

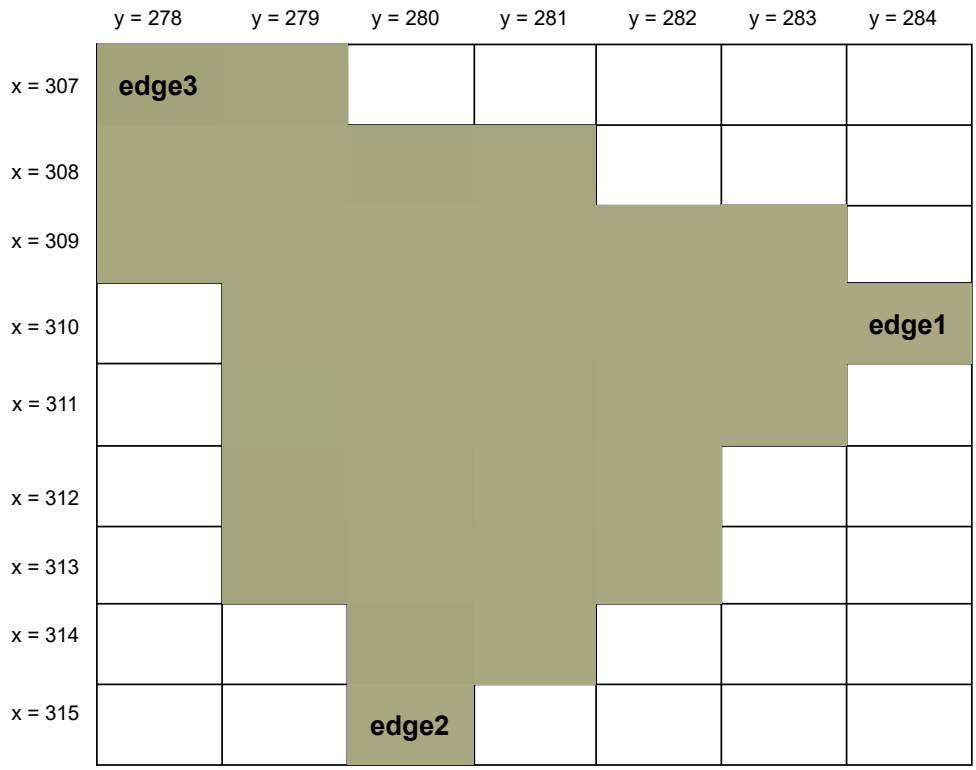
(α') Ορίζουμε γραμμή σάρωσης x_{scan} = x

(β') Αν η γραμμή δεν περιέχει καθόλου σημεία που να ανήκουν σε πλευρές, συνεχίζουμε στην επόμενη γραμμή

(γ') Αν η γραμμή περιέχει παραπάνω από δύο σημεία που να ανήκουν σε πλευρές, δηλαδή συνεχόμενες γραμμές, κρατάμε μόνο τα μη διαδοχικά σημεία. *Για την γραμμή x = 309 του παραπάνω τριγώνου κρατάμε μόνο τα σημεία [309,278] και [309,282]*

(δ') Για τα σημεία των πλευρών που προέκυψαν, που θα είναι 2, αναθέτουμε στο τρίγωνο όλα τα ενδιάμεσα
Για την γραμμή x = 309 του παραπάνω τριγώνου προσθέτουμε τα σημεία [309,279], [309,280], [309,281], ενώ για την γραμμή x = 314 δεν προσθέτουμε κανένα σημείο

Μετά την εφαρμογή της παραπάνω διαδικασίας στο τρίγωνο με id = 3331 παίρνουμε το παρακάτω αποτέλεσμα:



Η συνάρτηση που υλοποιεί τα παραπάνω σε κώδικα βρίσκεται στο αρχείο render.py με όνομα gourauds

3.4 Αποτελέσματα σε δεδομένα εισόδου

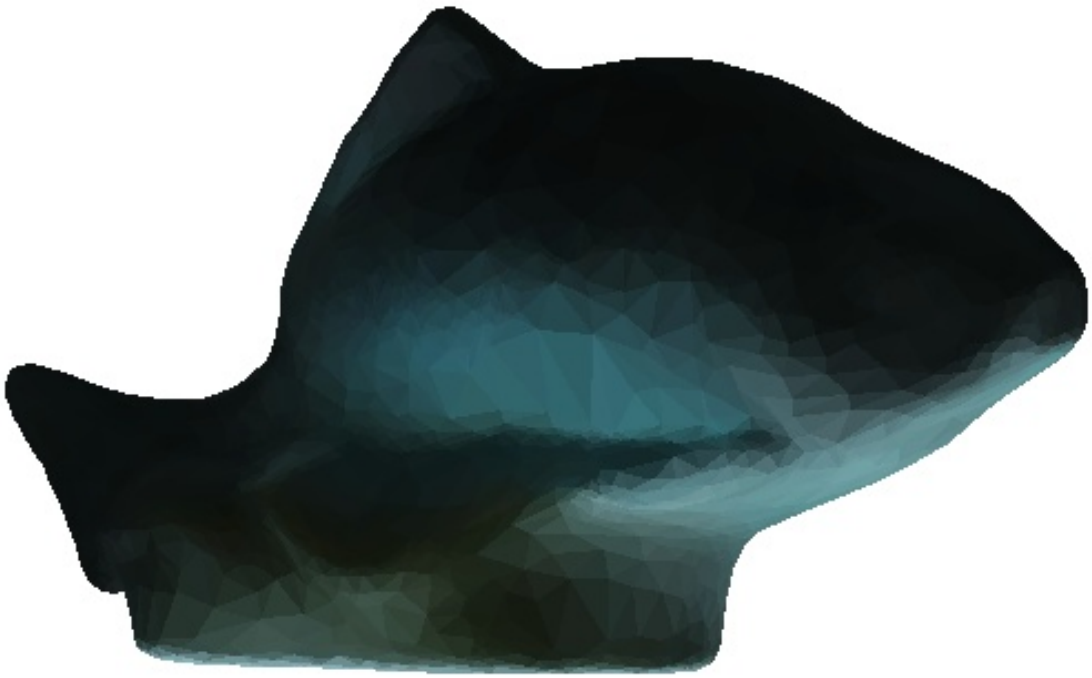
Αφού έχουμε υλοποιήσει τα παραπάνω, στην συνέχεια δοκιμάζουμε τις ρουτίνες για flat και Gouraud shading στα εξής δεδομένα εισόδου:

- Πίνακας vertices: πίνακας διάστασης (K,2) με τις συντεταγμένες [x,y] των κορυφών

- Πίνακας faces: πίνακας διάστασης (L,3) με δείκτες στον πίνακα vertices. Ορίζει τις κορυφές του κάθε τριγώνου
- Πίνακας vcolors: πίνακας διάστασης (K,3) με τις [r,g,b] τιμές της κάθε κορυφής
- Πίνακας depth: πίνακας διάστασης (K,1) με το βάθος την κάθε κορυφής

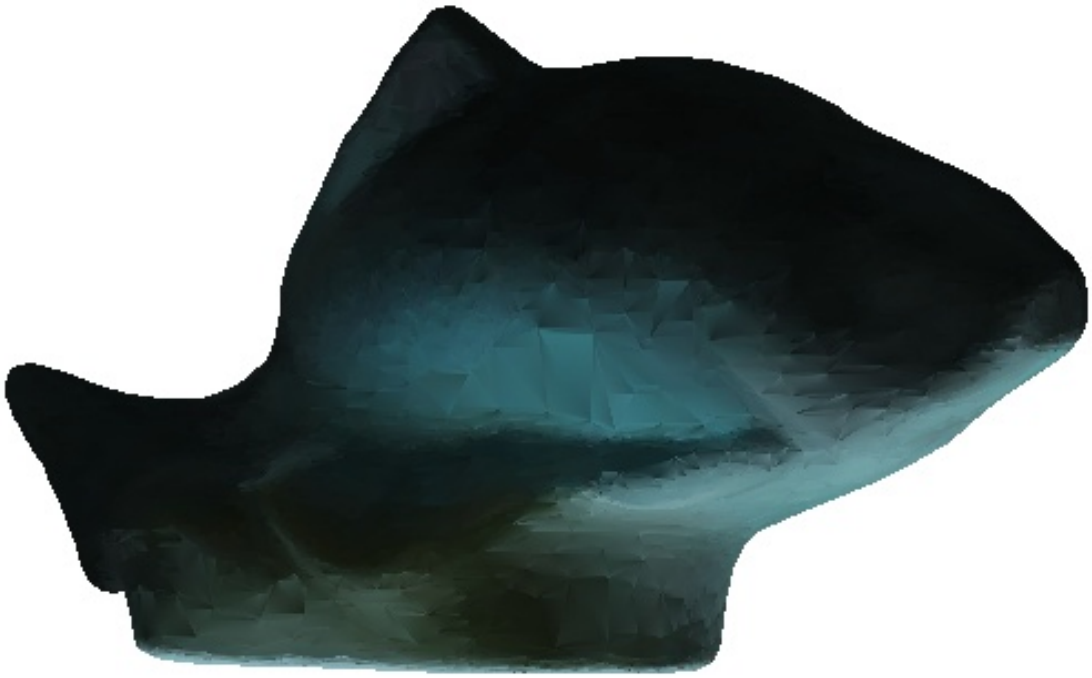
Όπου K είναι ο αριθμός των κορυφών και L είναι ο αριθμός τριγώνων

3.4.1 Flats shading



Το αρχείο που δίνει το παραπάνω αποτέλεσμα είναι το demo_flats.py

3.4.2 Gourauds shading



Το αρχείο που δίνει το παραπάνω αποτέλεσμα είναι το demo_gourauds.py

3.4.3 Σύγκριση μεθόδων

- Ως προς την ποιότητα του αποτελέσματος
  Και οι δύο μέθοδοι έχουν ως αποτέλεσμα σχετικά ομοιόμορφο χρωματισμό. Παρόλα αυτά η συνάρτηση Gourauds shading έχει καλύτερο αποτέλεσμα σε ορισμένα σημεία.
- Ως προς τον χρόνο εκτέλεσης
  Η συνάρτηση Flats shading ολοκληρώνει τον χρωματισμό σε 6.64 δευτερόλεπτα, ενώ η Gourauds shading τερματίζει σε 20.54 δευτερόλεπτα. Επομένως η Gourauds καθυστερεί περισσότερο.