Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Πολυτεχνική Σχολή
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Ηλεκτρονικής και Υπολογιστών

Εργασία 2 στην Γραφική με Υπολογιστές

Εργασία του
Φώτη Αλεξανδρίδη, ΑΕΜ: 9953
faalexandr@ece.auth.gr

23 Μαΐου 2023

Περιεχόμενα

Κεφάλαιο 1

Overview

1.1 Στόχος

Ο στόχος της εργασίας είναι να αναπαρασταθούν τρισδιάστατα σημεία στον χώρο σε δισδιάστατη έγχρωμη εικόνα, με χρήση προβολής μοντέλου προοπτικής pin hole κάμερας. Τα δεδομένα μας δίνονται και μας ζητείται τόσο η αναπαράστασή τους, όσο και η εφαρμογή μετασχηματισμών μετατόπισης και περιστροφής σε αυτά.

Τα σημεία μας δίνονται από αρχείο δεδομένων που δίνεται με την εκφώνηση της άσκησης [1], με το τελικό σχήμα να είναι το αποτέλεσμα της άσκησης 1. Για τον χρώματισμό της τελικής εικόνας, χρησιμοποιείται η μέθοδος χρωματισμού Gouraud που είχε αναπτυχθεί στα πλαίσια της πρώτης εργασίας. Μιας και η πρώτη εργασία υλοποιήθηκε χωρίς κάποια απώλεια, η εισαγωγή του τμήμας της υλοποίησής της σε αυτή την εργασία δεν προκάλεσε ουδεμία καθυστέρηση στην ανάπτυξή της.

1.2 Υλοποίηση

Έχουν αναπτυχθεί οι 7 ζητούμενες από την εκφώνηση συναρτήσεις, χωρισμένες σε 3 αρχεία πηγαίου κώδικα python. Επιπρόσθετα έχει αναπτυχθεί ένα τέταρτο αρχείο-πρόγραμμα για την επίδειξη της λειτουργικότητας του συστήματος.

Χρησιμοποιούνται επίσης δύο αρχεία από την εργασία 1, το αρχείο `helpers.py`, και το αρχείο `fill_triangles.py`, με το δεύτερο να περιέχει μόνο την συνάρτηση `gourauds`, καθώς στα πλαίσια της εργασίας η συνάρτηση `flats` δε χρησιμοποιείται. Μιας και η λειτουργία τους έχει αναλυθεί εκτενώς στην αναφορά της πρώτης εργασίας, δεν θα γίνει η επεξήγησή τους ξανά στο κομμάτι επεξήγησης του κώδικα.

Κεφάλαιο 2

Code

2.1 Γενική οργάνωση κώδικα

Όπως αναφέρθηκε, οι συναρτήσεις που μας ζητούνται έχουν υλοποιηθεί σε 3 αρχεία, τα `transformation_functions.py`, `camera_functions.py` και `render_functions.py`. Για την επίδειξη της λειτουργικότητας, ένα τέταρτο αρχείο, το `demo.py` διαβάζει δεδομένα από το παρεχόμενο `h2.npy` και χρησιμοποιεί τον κώδικα που αναπτύχθηκε για να καταγράψει μια έγχρωμη RGB εικόνα από την προοπτική προβολή των σημείων του χώρου από ένα συγκεκριμένο σημείο προς ένα άλλο.

Ακόμη, από την πρώτη εργασία χρησιμοποιείται η συνάρτηση χρωματισμού τριγώνου `gourauds` για την δημιουργία της έγχρωμης εικόνας από τα προβαλλόμενα σημεία στην εικόνα και τα χρώματά τους.

2.2 Το αρχείο `transformation_functions.py`

Το αρχείο αυτό περιέχει τις τρεις πρώτες ζητούμενες συναρτήσεις, που είναι συναρτήσεις οι οποίες ασχολούνται με την αλλαγή των συντεταγμένων τρισδιάστατων σημείων στον χώρο μέσω της εφαρμογής μετασχηματισμών (μετατόπισης και περιστροφής).

Η πρώτη συνάρτηση που ζητείται από την εκφώνηση είναι η `rotmat`:

```
def rotmat(theta, u):
```

Η συνάρτηση αυτή δημιουργεί τον $3x3$ πίνακα περιστροφής γύρω από τον άξονα που αναπαριστά το μοναδιαίο διάνυσμα u κατά γωνία *theta*. Η υλοποίησή της λαμβάνεται από τις σημειώσεις του μαθήματος [2], σελίδες 60-63.

Η δεύτερη ζητούμενη συνάρτηση είναι η `rotate_translate`:

```
def rotate_translate(cp, theta, u, A, t):
```

Η συνάρτηση αυτή πρώτα μετασχηματίζει το (ή τα) σημείο (ή σημεία) που δίνονται στην μεταβλητή *cp* περιστρέφοντάς τα γύρω από άξονα που διέρχεται από το σημείο A και είναι παράλληλος προς το διάνυσμα u κατά γωνία *theta*, και μετά τα μετατοπίζει κατά t. Η υλοποίηση ξανά είναι μια κλασσική υλοποίηση ενός πίνακα ομογενούς μετασχηματισμού στις τρεις διαστάσεις από τις σημειώσεις [2]. Όπως μας ζητείται,

έχουμε προνοήσει ώστε η συνάρτηση να λειτουργεί τόσο για ένα σημείο όσο και για πίνακα σημείων.

Η τελευταία συνάρτηση του αρχείου είναι η `change_coordinate_system`:

```
def change_coordinate_system(cp, r, c0):
```

Η οποία αλλάζει το σύστημα συντεταγμένων του (ή των) σημείων που δίνονται στην μεταβλητή *cp* με βάση τον πίνακα περιστροφής *r* και το αρχικό σημείο *c*0 του νέου συστήματος συντεταγμένων. Η υλοποίηση ανάγεται σε παρόμοια υλοποίηση με την συνάρτηση `rotate_translate` και ο μετασχηματισμός ξανά βρίσκεται από τις σημειώσεις [2].

2.3 Το αρχείο `camera_functions.py`

Το αρχείο αυτό περιέχει συναρτήσεις οι οποίες μετατρέπουν τα σημεία από τον τρισδιάστατο χώρο στον δισδιάστατο χώρο της εικόνας. Συγκεκριμένα, περιέχει τις επόμενες τρεις ζητούμενες συναρτήσεις.

Αρχικά υλοποιείται η συνάρτηση `pin_hole`:

```
def pin_hole(f, cv, cx, cy, cz, p3d):
```

η οποία επιστρέφει τις δισδιάστατες συντεταγμένες προβολής, καθώς και το βάθος, των σημείων αφού υποστούν προοπτική προβολή από μια pin hole camera, με f την απόσταση f, *cv* τις συντεταγμένες του σημείου της κάμερας, *cx, cy, cz* τα μοναδιαία διανύσματα προσανατολισμού της κάμερας, και *p3d* το (ή τα) σημεία του τρισδιάστατου χώρου προς μετατροπή. Για να το κάνει αυτό, αρχικά υπολογίζει τον πίνακα στροφής της κάμερας με βάση τα μοναδιαία διανύσματα προσανατολισμού (τύπος από την θεωρία [2]), και με βάση τον πίνακα αυτό και τις συντεταγμένες του σημείου της κάμερας, μετασχηματίζει τις συντεταγμένες των σημείων από το WCS στο σύστημα συντεταγμένων της κάμερας. Έπειτα, για κάθε σημείο, υπολογίζεται η προοπτική προβολή, πολλαπλασιάζοντας την τετμημένη και την τεταγμένη του με τον όρο $\frac{f}{z_c}$, όπου z_c το βάθος του σημείου στο σύστημα συντεταγμένων της κάμερας. Τέλος, επιστρέφει τόσο την προοπτική προβολή των σημείων αυτών όσο και το βάθος τους πριν την προοπτική προβολή τους.

Έπειτα, υπάρχει η συνάρτηση `camera_looking_at`:

```
def camera_looking_at(f, cv, cK, cup, p3d):
```

η οποία επίσης επιστρέφει τις δισδιάστατες συντεταγμένες προοπτικής προβολής και το βάθος των σημείων πριν την προβολή τους, αλλά αυτή την φορά αντί να δέχεται τα *cx, cy, cz*, δέχεται τις συντεταγμένες του σημείου στόχου *cK* και τις συντεταγμένες του διανύσματος up της κάμερας *cup*.

Η συνάρτηση αρχικά υπολογίζει τα *cx, cy, cz* από τα *cK, cv, cup* (η μεθοδολογία εξηγείται ενδελεχώς στις σημειώσεις [2]) και έπειτα καλεί την συνάρτηση `pin_hole` για να λάβει τα απαραίτητα αποτελέσματα, τα οποία και επιστρέφει.

Τέλος, υλοποιούμε την συνάρτηση `rasterize`:

```
def rasterize(p2d, rows, columns, H, W):
```

η οποία λαμβάνει σαν είσοδο τις προοπτικές προβολές του (ή των) σημείων *p2d*, τις διαστάσεις του πετάσματος της κάμερας *H, W* και τις διαστάσεις του καμβά της εικόνας σε εικονοστοιχεία *rows, columns*. Αυτό που κάνει είναι να μετατρέψει τα σημεία από τον χώρο συντεταγμένων του πετάσματος στον χώρο συντεταγμένων της εικόνας. Για να γίνει αυτό, ξέρουμε πως οι συντεταγμένες αρχικά έχουν το σημείο $(0, 0)$ τους στο κέντρο του πετάσματος. προσθέτουμε σε αυτές το μισό του πλάτους και του ύψους του πετάσματος ανάλογα αν μιλάμε για την τετμημένη ή την τεταγμένη, έπειται διαιρούμε με το αντίστοιχο μέγεθος (ύψος/πλάτος) για να κάνουμε scale τις συντεταγμένες μας σε κανονικοποιημένο χώρο, κι έπειτα πολλαπλασιάζουμε το αποτέλεσμα με τις γραμμές ή τις στήλες της είκόνας, για να σιγουρέψουμε ότι οι συντεταγμένες που βρήκαμε αντιστοιχούν σε όλη την εικόνα. Τέλος, επειδή το τελικό αποτέλεσμα είναι συντεταγμένες εικονοστοιχείων, κάνουμε round τις τιμές που βρήκαμε ώστε να είναι συμβατές με την εικόνα μας.

Να σημειωθεί ότι σε αυτή την απλοϊκή υλοποίηση της συνάρτησης, δεν ελέγχουμε αν βρούμε συντεταγμένες εκτός του δεκτού χώρου (π.χ. να ξεκινήσουμε με συντεταγμένες εκτός του πετάσματος της κάμερας). Στην συγκεκριμένη εργασία δεν μας απασχολεί (και απλά αργότερα υπάρχει ένας έλεγχος ασφαλείας που θα εξηγηθεί στην επόμενη συνάρτηση), αλλά σε μια πιο πλήρη υλοποίηση θα έπρεπε να σκεφτούμε αν και πως να προβάλλουμε τα σημεία αυτά στην εικόνα μας.

2.4 Το αρχείο `render_functions.py`

Το αρχείο αυτό περιέχει μια μοναδική συνάρτηση, την τελευταία που ζητείται, την `render_object`, η οποία λαμβάνει σημεία στον τρισδιάστατο χώρο και επιστρέφει μια έγχρωμη εικονα όπου περιέχει το αντικείμενο που δημιουργείται από τα προδιαγεγραμμένα τρίγωνα:

```
def render_object(p3d, faces, vcolors, H, W,
rows, columns, f, cv, cK, cup):
```

Η συνάρτηση αυτή λειτουργεί σε αντιστοιχία με την παρόμοια συνάρτηση της εργασίας 1. Σκοπός της είναι να λάβει ένα σύνολο από σημεία στον τρισδιάστατο χώρο *p3d*, τριάδες που συμβολίζουν τα τρίγωνα με βάση τις κορυφές τους *faces*, το χρώμα κάθε κορυφής *vcolors*, τις διαστάσεις του πετάσματος της κάμερας *H, W*, τις διαστάσεις της παραγόμενης εικόνας *rows, columns*, και τις παραμέτρους κάμερας *f, cv, cK, cup* που έχουν περιγραφεί στην συνάρτηση `camera_looking_at`, να προβάλει τα σημεία αυτά σε μια εικόνα και μετά να χρωματίσει τα τρίγωνα που σχηματίζουν με βάση το ποια τρίγωνα δίνονται.

Αρχικά, υπολογίζεται η προοπτική προβολή και το βάθος των σημείων με την χρήση της συνάρτησης camera_looking_at. Οι προοπτικές προβολές μετατρέπονται σε συντεταγμένες εικονοστοιχείων με την χρήση της rasterize.

Έπειτα, ακολουθείται η ακριβώς ίδια διαδικασία με την `render` της εργασίας 1. Επιγραμματικά, δημιγουργείται μια λευκή εικόνα του επιθυμητού μεγέθους, τα τρίγωνα διατάσσονται με βάση το βάθος του κέντρου βάρους τους (θέλουμε πρώτα να χρωματίσουμε τα πιο απομακρυσμένα τρίγωνα), και έπειτα χρωματίζονται μόνο τα τρίγωνα τα οπία έχουν και τις τρεις κορυφές τους σε συντεταγμένες εντός εικόνας. Ουσιαστικά εδώ υλοποιούμε την ¨αποκοπή¨ στο πρόγραμμά μας. Τα τρίγωνα χρωματίζονται

με βάση την μέθοδο Gouraud, η οποία αναπτύχθηκε στην εργασία 1 στην συνάρτηση `gourauds`. Η τελική εικόνα επιστρέφεται στον χρήστη.

2.5 Το αρχείο `demo.py`

Το πρόγραμμα αυτό έχει ως σκοπό την επίδειξη των λειτουργιών που αναπτύχθηκαν στις υπόλοιπες συναρτήσεις με μια συγκεκριμένη σειρά διαδικασιών που μας ζητείται από την εκφώνηση.

Αρχικά, ορίζονται κάποιες σταθερές (το μέγεθος του πετάσματος και του καμβά), διαβάζονται από το παρεχόμενο αρχείο `h2.npy` δεδομένα όπως οι παράμετροι της κάμερας, το σύνολο των σημείων στον τρισδιάστατο κόσμο, και ο ορισμός/χρωματισμός των τριγώνων/ακμών τους. Έπειτα, χρησιμοποιείται τέσσερις φορές η `render_object` για να παραχθούν τέσσερις έγχρωμες εικόνες.

Η πρώτη εικόνα αποτελείται από τα σημεία αυτούσια όπως διαβάζονται από το αρχεί ο-δεδομένων. Η δεύτερη αποτελείται από τα σημεία μετακινημένα κατά t_1, διάνυσμα που λαμβάνεται από το αρχείο δεδομένων. Για την μετατόπιση των σημείων χρησιμοποιείται η `rotate_translate`, με έναν dummy άξονα u και μηδενική γωνία περιστροφής. Στην τρίτη εικόνα, τα σημεία μετά την μετακίνησή τους έχουν υποστεί περιστροφή κατά γωνία ϕ γύρω από άξονα u που διαβάζονται από το αρχείο δεδομένων. Πάλι χρησιμοποιείται η `rotate_translate` με αυτή την φορά να χρησιμοποιείται μηδενικό διάστημα μετατόπισης. Η τέταρτη και τελευταία εικόνα περιέχει τα σημεία μετά την περιστροφή τους μετατοπισμένα κατά t_2, το οποίο επίσης διαβάζεται από το αρχείο δεδομένων. Η μετατόπιση πραγματοποιείται ακριβώς όπως και η πρώτη μετατόπιση των σημείων κατά t_1.

Τέλος, εμφανίζεται και αποθηκεύεται ένα συνολικό διάγραμμα που περιέχει και τις τέσσερις εικόνες μαζί, μαζί με τις παραμέτρους που έχουν διαβαστεί από τα δεδομένα, και αποθηκεύονται οι μεμονωμένες εικόνες. Για την αποθήκευση των εικόνων χρησιμοποιείται η βιβλιοθήκη opencv, και ο τρόπος αποθήκευσης καθώς και οι συμβάσεις που χρησιμοποιεί έχουν αναλυθεί και πάρθηκαν από την πρώτη εργασία.

Κεφάλαιο 3

Αποτελέσματα

Οι παράμετροι-μεγέθη που μας δίνονται από το αρχείο της εργασίας είναι οι ακόλουθοι:

$$t_1 = \begin{bmatrix} 0 & 0 & -15000 \end{bmatrix}^T \tag{3.1}$$

$$\phi = -0.08726656259971647 \tag{3.2}$$

$$u = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T \tag{3.3}$$

$$t_2 = \begin{bmatrix} 0 & 500 & -10000 \end{bmatrix}^T \tag{3.4}$$

$$c_v = \begin{bmatrix} 0 & 0 & 12000 \end{bmatrix}^T \tag{3.5}$$

$$c_k = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{3.6}$$

$$c_u = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T \tag{3.7}$$

$$f = 70 \tag{3.8}$$

με:

- t_1: το διάνυσμα μετατόπισης του πρώτου βήματος
- ϕ: η γωνία περιστροφής του δεύτερου βήματος
- u: το διάνυσμα του άξονα περιστροφής του δεύτερου βήματος
- t_2: το διάνυσμα μετατόπισης του τρίτου βήματος
- c_v: το σημείο συντεταγμένων της κάμερας
- c_k: το σημείο στόχου της κάμερας
- c_u: το διάνυσμα up vector της κάμερας
- f: η απόσταση f

Τα αποτελέσματα φαίνονται στις παρακάτω εικόνες:

Σχήμα 3.1: Αρχική εικόνα πριν από τους μετασχηματισμούς

Σχήμα 3.2: Εικόνα μετά την μετακίνηση των σημείων κατά t_1

Σχήμα 3.3: Εικόνα μετά την μετακίνηση των σημείων κατά t_1 και περιστροφή κατά φ γύρω από τον άξονα u

Σχήμα 3.4: Εικόνα μετά την μετακίνηση των σημείων κατά t_1 και περιστροφή κατά φ γύρω από τον άξονα u και μετακίνηση των σημείων κατά t_2

3.1 Σχολιασμός

Αρχικά, μπορούμε από τα διανύσματα/σημεία θέσης και σχόχου της κάμερας να δούμε εύκολα ότι η κάμερα βρίσκεται πάνω στον άξονα z και κοιτάει προς τα κάτω. Έτσι, μετά την μετακίνηση κατά t_1, δηλαδή την μετακίνηση των σημείων προς τα κάτω, πιο μακριά δηλαδή, βλέπουμε το αντικείμενο να μικραίνει. Αντίστοιχα, ο άξονας περιστροφής είναι παράλληλος προς το διάνυσμα up της κάμερας, και τα σημεία απέχουν αρκετά από την αρχή των αξόνων, επομένως φαίνεται σαν να μετακινούνται αριστερά (αντιωρολογιακή φορά πίσω από τον άξονα είναι προς τα αριστερά). Τέλος στο βήμα 3 το αντικείμενο φαίνεται τόσο να απομακρύνεται από εμάς (μετατόπιση πιο κάτω στον άξονα z) όσο και να κινείται προς τα πάνω (μετακίνηση προς τον άξονα y που είναι ο up), σκέψεις που συνάδουν με τις τιμές του διανύσματος. Για την εμφάνιση των εικόνων, έχουν αλλαχτεί οι άξονες x και y μεταξύ τους (σύστημα συντεταγμένων των καμερών).

Bibliography

[1] A. Delopoulos, *Computer Graphics assignment 2: transformations and projections*, 2023.

[2] ——, *Computer Graphics Notes*, 2009.