Οικονόμου Αλέξανδρος AEM: 9260 email: alexanco@ece.auth.gr

Αναφορά στο 2ο παραδοτέο Γραφική με υπολογιστές

Θέμα εργασίας: Μετασχηματισμοί και Προβολές

Τα ζητούμενα της εργασίας ήταν

1. Μια κλάση για την υλοποίηση ενός πίνακα μετασχηματισμού
2. Μια συνάρτηση για τη διαδικασία σημειακού μετασχηματισμού affine
3. Μια συνάρτηση για τη διαδικασία υπολογισμού συντεταγμένων ενός σημείου ως προς ένα μετασχηματισμένο σύστημα συντεταγμένων
4. Μια συνάρτηση για τη διαδικασία παραγωγής προοπτικών προβολών και υπολογισμού βάθους των σημείων
5. Μια συνάρτηση όπως την προηγούμενη αλλά με διαφορετικά ορίσματα (ck, cu)
6. Μια συνάρτηση απεικόνισης
7. Μια συνάρτηση φωτογράφισης
8. Ένα script με όνομα demo.m

1. Κλάση για την υλοποίηση ενός πίνακα μετασχηματισμού

Η κλάση **transformation_matrix** έχει έναν πίνακα μετασχηματισμού T(4x4), έναν constructor και δυο μεθόδους.

Ο **constructor** αρχικοποιεί τον πίνακα στον μοναδιαίο (4x4).

Η μέθοδος **rotate** δέχεται ως είσοδο

- Μια γωνία περιστροφής θ,
- Ένα δίανυσμα u και
- Ένα αντικείμενο τύπου transformation_matrix

και έχει ως έξοδο το αντικείμενο με πίνακα μετασχηματισμού.

Περιγραφή διαδικασίας:

Δημιουργούμε τον πίνακα περιστροφής R (3x3) μέσα στον πίνακα T, με τον τύπο του Rodriguez.

Η μέθοδος **translate** δέχεται ως είσοδο

- Ένα δίανυσμα μετατόπισης t και
- Ένα αντικείμενο τύπου transformation_matrix

και έχει ως έξοδο το αντικείμενο με πίνακα μετασχηματισμού.

Περιγραφή διαδικασίας:

Προσθέτουμε στην 4η στήλη του πίνακα T, το διάνυσμα μετατόπισης t.

2. Συνάρτηση για τη διαδικασία σημειακού μετασχηματισμού affine

Η συνάρτηση **affine_transform** δέχεται ως είσοδο

- Τις συντεταγμένες ενός σημείου p (3x1) ως προς ένα σύστημα συντεταγμένων και
- Ένα αντικείμενο τύπου **transformation_matrix** που περιέχει έναν ενημερωμένο πίνακα μετασχηματισμού τύπου affine

και έχει ως έξοδο τις νέες συντεταγμένες του σημείου p.

Περιγραφή διαδικασίας μετασχηματισμού affine:

Αρχικά, προσθέτουμε μια γραμμή στον πίνακα με τις συντεταγμένες, ώστε να γίνει (4x1) για να είναι δυνατός ο πολλαπλασιασμός των πινάκων cp και T. Στη συνέχεια, πολλαπλασιάζονται οι πίνακες και το τελικό αποτέλεσμα, δηλαδή οι νέες συντεταγμένες του σημείου p αποθηκεύονται στον πίνακα cq.

3. Συνάρτηση για τη διαδικασία υπολογισμού συντεταγμένων ενός σημείου ως προς ένα μετασχηματισμένο σύστημα συντεταγμένων

Η συνάρτηση **system_transform** δέχεται ως είσοδο

- Ένα αντικείμενο τύπου **transformation_matrix** που περιέχει έναν ενημερωμένο πίνακα μετασχηματισμού μόνο για περιστροφή ως προς το διάνυσμα v0 και
- Τις συντεταγμένες ενός σημείου p (3x1) ως προς το ίδιο σύστημα συντεταγμένων με το v0

και έχει ως έξοδο τις νέες συντεταγμένες του σημείου p ως προς ένα νέο μετασχηματισμένο σύστημα συντεταγμένων.

Περιγραφή διαδικασίας:

Αρχικά, προσθέτουμε μια γραμμή στον πίνακα με τις συντεταγμένες, ώστε να γίνει (4x1) για να είναι δυνατός ο πολλαπλασιασμός των πινάκων cp και T. Εδώ, όμως, θα χρησιμοποιηθεί ο αντίστροφος του T. Πολλαπλασιάζονται οι πίνακες και το τελικό αποτέλεσμα, δηλαδή οι νέες συντεταγμένες του σημείου p αποθηκεύονται στον πίνακα dp.

4. Συνάρτηση για τη διαδικασία παραγωγής προοπτικών προβολών και υπολογισμού βάθους των σημείων

Η συνάρτηση **project_cam** δέχεται ως είσοδο

- Τα cv, cx, cy, cz, δηλαδή, τις συντεταγμένες των μοναδιαίων διανυσμάτων και του κέντρου μιας προοπτικής κάμερας ως προς το WCS
- Μια μεταβλητή w, για την απόσταση του πετάσματος από το κέντρο
- Έναν πίνακα p με τις συντεταγμένες σημείων ως προς το WCS

και έχει ως έξοδο έναν πίνακα P που περιέχει τις προοπτικές προβολές των τρισδιάστατων σημείων και έναν πίνακα D που περιέχει το βάθος κάθε σημείου πριν την προβολή του στις δυο διαστάσεις.

Περιγραφή διαδικασίας:

Αρχικά, δημιουργούμε ένα αντικείμενο τύπου **transformation_matrix**. Θέτουμε στον πίνακα μετασχηματισμού τις συντεταγμένες που μας δίνονται. Γίνεται αλλαγή του συστήματος συντεταγμένων από WCS σε CCS με τη χρήση της **system_transform**.

Υπολογίζονται οι προβολές για όλα τα σημεία με τον εξής τρόπο:

```
xq = (w * xp) / zp
yq = (w * yp) / zp
```

και τα βάθη:

```
depth = zp
```

5. Συνάρτηση όπως την προηγούμενη αλλά με διαφορετικά ορίσματα (ck, cu)

Η συνάρτηση **project_cam_ku** δέχεται ως είσοδο

- Μια μεταβλητή w, για την απόσταση του πετάσματος από το κέντρο
- Το cv, δηλαδή τις συντεταγμένες του κέντρου μιας προοπτικής κάμερας ως προς το WCS
- Το clookat, δηλαδή τις συντεταγμένες του σημείου στόχου K
- Το cup, δηλαδή τις συντεταγμένες του μοναδιαίου up vector και
- Έναν πίνακα p με τις συντεταγμένες σημείων ως προς το WCS

και έχει ως έξοδο έναν πίνακα P που περιέχει τις προοπτικές προβολές των τρισδιάστατων σημείων και έναν πίνακα D που περιέχει το βάθος κάθε σημείου πριν την προβολή του στις δυο διαστάσεις.

Περιγραφή διαδικασίας:

Αρχικά, υπολογίζεται το διάνυσμα με τις συντεταγμένες του άξονα z με τον εξής τρόπο:

```
Zc = CK / norm(CK)
```

Όπου CK, η διαφορά από το σημείο K και του κέντρου της κάμερας

Στη συνέχεια, υπολογίζεται το διάνυσμα Yc.

```
t = u - dot(u, Zc) * Zc
```

```
Yc = t / norm(t)
```

Έπειτα, το διάνυσμα Xc προκύπτει από το εξωτερικό γινόμενο των Zc και Yc.

```
Xc = cross(Yc, Zc)
```

Τέλος, καλείται η project_cam με τις παραπάνω συντεταγμένες για να υπολογιστούν οι προβολές και τα βάθη.

6. Συνάρτηση απεικόνισης

Η συνάρτηση **rasterize** δέχεται ως είσοδο

- Δυο μεταβλητές, H και W, δηλαδή τις διαστάσεις του πετάσματος μιας κάμερας σε ίντσες
- Δυο μεταβλητές, M και N, δηλαδή τις διαστάσεις της εικόνας σε pixels και
- Έναν πίνακα P με τις συντεταγμένες των σημείων ως προς τις ίντσες

και έχει ως έξοδο έναν πίνακα Prast που περιέχει τις συντεταγμένες των σημείων ως προς τα pixels.

Περιγραφή διαδικασίας απεικόνισης:

Για να γίνει αντιστοίχιση των σημείων στα κατάλληλα pixels θέλουμε από [-W/2, W/2] x [-H/2, H/2] να φτάσουμε σε [1, N] x [1, M]. Υπολογίζουμε την απόσταση dx, dy από pixel σε pixel. Προσθέτουμε σε κάθε σημείο W/2 και H/2 στα x και y αντίστοιχα, ώστε να έρθουν όλα τα σημεία στο μηδέν και πάνω. Έπειτα διαιρούμε τα σημεία με την απόσταση dx και dy και στρογγυλοποιούμε στο κοντινότερο pixel. Ο πίνακας Prast που προκύπτει έχει τα σημεία του ως προς τα pixels.

7. Συνάρτηση απεικόνισης

Η συνάρτηση **render_object** δέχεται ως είσοδο

- Έναν πίνακα p που περιέχει τις τρισδιάστατες συντεταγμένες των σημείων του αντικειμένου
- Έναν πίνακα F που περιέχει τις κορυφές των K τριγώνων. Κάθε γραμμή έχει τις τρεις κορυφές του αντίστοιχου τριγώνου
- Έναν πίνακα C που περιέχει τα χρώματα των κορυφών
- Δυο μεταβλητές, H και W, δηλαδή τις διαστάσεις του πετάσματος μιας κάμερας σε ίντσες
- Δυο μεταβλητές, M και N, δηλαδή τις διαστάσεις της εικόνας σε pixels
- Μια μεταβλητή w, δηλαδή η απόσταση του πετάσματος από το κέντρο
- Το cv, δηλαδή τις συντεταγμένες του κέντρου μιας προοπτικής κάμερας ως προς το WCS
- Το clookat, δηλαδή τις συντεταγμένες του σημείου στόχου K και
- Το cup, δηλαδή τις συντεταγμένες του μοναδιαίου up vector

και έχει ως έξοδο μια έγχρωμη εικόνα I που περιέχει K χρωματισμένα τρίγωνα.

Περιγραφή διαδικασίας φωτογράφισης:

Αρχικά, μέσω της **project_cam_ku** υπολογίζονται οι προβολές και τα βάθη των σημείων της εικόνας. Έπειτα μέσω της **rasterize** υπολογίζονται οι συντεταγμένες των σημείων στα αντίστοιχα pixels και τέλος μέσω της **render** γίνεται ο χρωματισμός της εικόνας.

Για την γρηγορότερη υλοποίηση χρησιμοποίησα την gouraud από έναν συμφοιτητή μου, τον Κωνσταντίνο Χατζή, AEM: 9256. Συγκεκριμένα είναι οι συναρτήσεις **vector_interp, update_scan_line, previous, paint_triangle_gouraud, next** και **multi_vector_interp.**

8. Script με όνομα demo

Το script αυτό:

- διαβάζει το αντικείμενο από το αρχείο hw2.mat
- εκτελεί ένα προκαθορισμένο σύνολο μετασχηματισμών στις κορυφές των τριγώνων
    - Τις μετατοπίζει κατά t1
    - Τις περιστρέφει κατά γωνία φ rad περί άξονα που διέρχεται από το σημείο O και έχει κατεύθυνση παράλληλη προς διάνυσμα g
    - Τις μετατοπίζει κατά t2
- φωτογραφίζει το αντικείμενο, καλώντας τη συνάρτηση **render_object**
- συνολικά παράγονται 4 φωτογραφίες και αποθηκεύονται με τη συνάρτηση imwrite του Matlab.

Συγκεκριμένα, γίνεται ένα **render_object** και αποθηκεύεται η αρχική θέση του ρακούν στην εικόνα με όνομα 0.jpg.

Έπειτα, για την μετατόπιση κατά t1, δημιουργούμε ένα αντικείμενο τύπου **transformation_matrix**. Με τη χρήση της translate τροποποιούμε τον πίνακα μετασχηματισμού T του αντικειμένου. Εφαρμόζουμε την **affine_transform** για να υπολογιστούν οι νέες συντεταγμένες των σημείων και τρέχουμε την **render_object** και αποθηκεύουμε την μετατοπισμένη θέση του ρακούν στην εικόνα με όνομα 1.jpg.

Στη συνέχεια, για την περιστροφή, αρχικοποιούμε εκ νέου το αντικείμενο, ώστε να αποκτήσουμε με τη χρήση της rotate τον νέο πίνακα μετασχηματισμού T. Εκτελείται η **affine_transform** για να υπολογιστούν οι νέες συντεταγμένες των σημείων και τρέχουμε την **render_object** και αποθηκεύουμε τη θέση του ρακούν μετά την περιστροφή στην εικόνα με όνομα 2.jpg.

Τέλος, για την μετατόπιση κατά t2, αρχικοποιούμε εκ νέου το αντικείμενο, ώστε να αποκτήσουμε με τη χρήση της translate τον νέο πίνακα μετασχηματισμού T. Εκτελείται η **affine_transform** για να υπολογιστούν οι νέες συντεταγμένες των σημείων και τρέχουμε την **render_object** και αποθηκεύουμε την μετατοπισμένη θέση του ρακούν στην εικόνα με όνομα 3.jpg.

Οικονόμου Αλέξανδρος AEM: 9260 email: alexanco@ece.auth.gr

Αποτελέσματα του demo:

Εικόνα 0.jpg: Αρχική θέση

Εικόνα 1.jpg: Μετατοπισμένη κατά t1

Εικόνα 2.jpg: Περιστροφή

Εικόνα 3.jpg: Μετατόπιση κατά t2