Γραφική με Υπολογιστές

Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης - Τμήμα Ηλεκτρολόγων Μηχανικών Μηχανικών Υπολογιστών

2η Εργασία Εξαμήνου: Μετασχηματισμοί και Προβολές

Αντωνίου Αντώνιος - aantonii@ece.auth.gr - 9482

Σκοπός της εργασίας

Στο δεύτερο σκέλος των εργασιών του μαθήματος, αναλαμβάνουμε επί της ουσίας να δημιουργήσουμε τα arguments που μας είχαν δοθεί για την υλοποίηση της συνάρτησης `render()` του πρώτου παραδοτέου. Αυτό σημαίνει πως μας δίνονται ως δεδομένα:

- Οι συντεταγμένες στον τρισδιάστατο κόσμο **(θα το συναντήσουμε πολλές φορές ως WCS - World Coordinate System)** των κορυφών των τριγώνων που απαρτίζουν ένα αντικείμενο $(verts_{3D})$,
- Οι συντεταγμένες της κάμερας (c_{org}),
- Οι συντεταγμένες του σημείου που θεωρείται *στόχος της κάμερας*, δηλαδή το κέντρο της φωτογραφίας που θα παραχθεί (c_{lookat}),
- Το μοναδιαίο διάνυσμα **up** (c_{up}): ποια διεύθυνση θεωρείται κατακόρυφη, και
- Την εστιακή απόσταση f.

Ωστόσο, για να φτάσουμε στο σημείο να καλέσουμε τη `render_object()`, που υπολογίζει τις προβολές των σημείων και καλεί τη `render()`, πρέπει πρώτα να υλοποιήσουμε μία σειρά από συναρτήσεις, οι οποίες αναλύονται παρακάτω.

$affine\_transform(c_p, \theta, u, t)$

Της δίνεται ένα σημείο (ή πίνακας από σημεία) c_p και επιστρέφει τις συντεταγμένες του σημείου/των σημείων μετά από έναν Affine μετασχηματισμό, ο οποίος αποτελείται από μία περιστροφή γύρω από άξονα u ή μια μετατοπίση κατά t ή έναν συνδυασμό αυτών. Για την υλοποίηση της περιστροφής, σχηματίζουμε τον **πίνακα R**, βασιζόμενοι στον ***τύπο του Rodrigues***:

$$(1 - cos\theta) \cdot \begin{bmatrix} u_x^2 & u_x u_y & u_x u_y \\ u_y u_x & u_y^2 & u_y u_z \\ u_z u_x & u_z u_y & u_z^2 \end{bmatrix} + cos\theta \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + sin\theta \cdot \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix}$$

Ο R χρησιμοποιείται ως:

$$c_q = R \cdot c_p$$

Ύστερα, αν ορίζεται από τα ορίσματα, πραγματοποείται και μία μετατόπιση κατά t, που υλοποιείται με απλή πρόσθεση των συντεταγμένων του διανύσματος στις συντεταγμένες του διανύσματος θέσης κάθε σημείου:

$$c_q' = c_q + t$$

$system\_transform(c_p, R, c_o)$

R είναι ο πίνακας περιστροφής που εξάγεται για την αλλαγή των συντεταγμένων ενός σημείου σε ένα νέο σύστημα συντεταγμένων, με νέα αρχή το c_o. Σύμφωνα με τη θεωρία επιστρέφουμε το διάνυσμα:

$$R^T \cdot (c_p - c_o)$$

$project\_cam(f, c_v, c_x, c_y, c_z, p)$

Επιστρέφει τις προβολές των σημείων που βρίσκονται στον πίνακα *p*, σε μια φωτογραφία στον "κόσμο της κάμερας" **(CCS - Camera Coordinate System)** του οποίου έχουμε βάση c_x, c_y, c_z και αρχή c_v, με εστιακή απόσταση *f*. Για την επίτευξη αυτού του σκοπού καλούμε τη `system_transform()` με πίνακα περιστροφής $R = [x_c, y_c, z_c]$. Παράγουμε τις καινούριες συντεταγμένες (έστω x, y, z) στο CCS και έπειτα, ξανά από τη θεωρία:

- $$x' = f \cdot \frac{x}{z}$$
- $$y' = f \cdot \frac{y}{z}$$
- $$depth = z$$

Τα βάθη *depth* είναι πληροφορία που χρειάζεται η `render()` για να καθορίσει με ποια σειρά θα χρωματιστούν τα τρίγωνα με τα οποία τροφοδοτείται.

$project\_cam\_lookat(f, c_{org}, c_{lookat}, c_{up}, verts_{3d})$

Της δίνονται όλες οι κορυφές $verts_{3d}$ των τριγώνων, που δίνονται μία-μία στην `project_cam()` μαζί με τη **βάση του CCS**, αφού έχει εξαχθεί βάσει της θεωρίας:

- $$\overline{CA} = c_{lookat} - c_{org}$$
- $$\hat{z_c} = \frac{\overline{CA}}{||CA||}$$
- $$\bar{t} = c_{up} - < c_{up}, \hat{z_c} > \cdot \hat{z_c}$$
- $$\hat{y_c} = \frac{\bar{t}}{||t||}$$
- $$\hat{x_c} = \hat{y_c} \times \hat{z_c}$$

Επιστρέφει τα ίδια δεδομένα με τη συνάρτηση που καλεί.

$rasterize(verts_{2d}, img_h, img_w, cam_h, cam_w)$

Αφού οι κορυφές προβληθούν σε ένα επίπεδο 2 διαστάσεων, πρέπει να αντιστοιχίσουμε τιε θέσεις τους με τα indices της εικόνας `img`, η οποία είναι ένας πίνακας $img_w \times img_h \times 3$, δηλαδή ένας χάρτης με τις RGB τιμές κάθε pixel της φωτογραφίας. Για να γίνει αυτή η αντιστοίχιση, επιλέγουμε να μεγεθύνουμε τον πίνακα αυτό αρκετά, ώστε να χωράει όλο το αντικείμενο που φωτογραφίσαμε, ασχέτως αν θα βρίσκεται όλο στην τελική φωτογραφία ή όχι. Οπότε κάνουμε μετατόπιση τέτοια, ώστε όλα τα indices που θα επιστρέψει η `rasterize()` να είναι θετικά (η ελάχιστη τιμή της μετατόπισης είναι $\frac{img_w}{2}$ κατά x και $\frac{img_h}{2}$ κατά y). Η συνάρτηση επιστρέφει την ποσότητα κατά την οποία έκανε offset όλους τους δείκτες, καθώς τα δεδομένα αυτά χρειάζονται στο τελευταίο βήμα του αλγορίθμου.

Χρησιμοποιώντας τα παραπάνω στη render_object()

Η `render_object()` είναι η συνάρτηση που ομαδοποιεί τις υλοποιημένες λειτουργικότητες και παράγει την τελική φωτογραφία με τη βοήθεια της `render()`. Στο πρώτο βήμα, καλείται η `project_cam_lookat()`, ώστε να εξαχθούν οι συντεταγμένες των κορυφών των τριγώνων στις 2 διαστάσεις, μαζί με τα αντίστοιχα *depths*. Έπειτα, η `rasterize()` αναλαμβάνει να αποτυπώσει αυτές τις κορυφές σε έναν πίνακα-φωτογραφία.

Ο πίνακας αυτός δεν έχει διαστάσεις $img_w \times img_h$, (αυτό είναι το ελάχιστο μέγεθος). Το πραγματικό μέγεθος της φωτογραφίας αρχικά καθορίζεται από τη μέγιστη τετμημένη και τεταγμένη των στοιχείων της φωτογραφίας, αφού όλα τα indices γίνουν θετικά, σύμφωνα με τις αρχές που αναφέρθηκαν στη `rasterize()`.

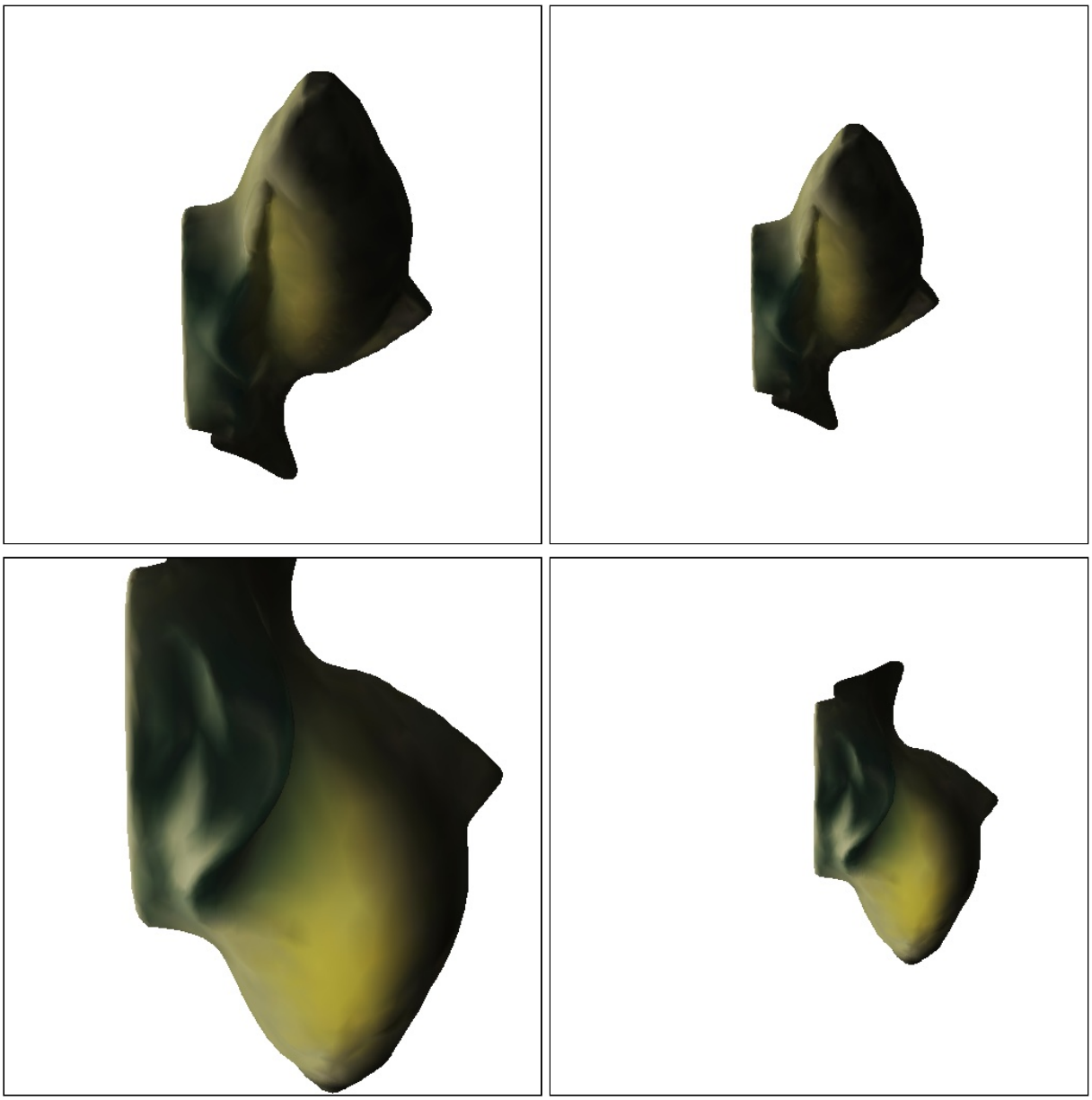
Η `render()`, λοιπόν, καλείται να παράξει αυτή τη φωτογραφία. Το τελικό αποτέλεσμα είναι το crop που θα γίνει, αγνοώντας τα πρώτα $crop[0]$ στοιχεία οριζόντια και τα πρώτα $crop[1]$ στοιχεία κατακόρυφα, και συνεχίζοντας μέχρι να δημιουργηθεί ένας πίνακας $img_w \times img_h$. Για τον πίνακα `crop` ισχύει:

- $$crop[0] = M - \frac{img_w}{2}$$
- $$crop[1] = N - \frac{img_h}{2}$$

(Θυμηθείτε τα M και N από τη `rasterize()`)

Αποτελέσματα

Τραβήχτηκε μία φωτογραφία από κάθε βήμα:
0. Αρχική κατάσταση
1. Μετατόπιση κατά $t_1 = [0, 0, -5000]$
2. Περιστροφή κατά γωνία $\phi = -\pi$, γύρω από άξονα $u = [0, 1, 0]$
3. Μετατόπιση κατά $t_2 = [0, 500, -10000]$



Acknowledgment

Ευχαριστώ πολύ το συνάδελφο Ανέστη Καϊμακαμίδη (ΑΕΜ 9627) για την παραχώρηση του κώδικα της 1ης εργασίας, μιας και η δική μου σωστή υλοποίηση ακόμα αγνοείται. Το αρχείο `functions.py` περιέχει τις υλοιποιήσεις του, τις οποίες οικειοποιήθηκα και επεξεργάστηκα, ώστε να φέρω τον κώδικα στα μέτρα μου.

Σας ευχαριστώ για το χρόνο σας

Αντωνίου Αντώνιος