1. Part A

1.1 Porjection of model onto a plane

1.1.1 Υπολογισμός της προβολής

Ο έλεγχος αν ένα σημείο ανήκει στην προβολή του μοντέλου δεν είναι τόσο απλός. Στην πιο απλή μέθοδο, υπολογίζονται οι προβολές των τριγώνων του μοντέλου στο επίπεδο και ελέγχεται αν το σημείο βρίσκεται εντός κάποιου τριγώνου. Αν βρίσκεται, τότε ανήκει στην προβολή του μοντέλου. Οι υπολογισμοί γίνονται πάλι στις συντεταγμένες του επιπέδου για ευκολία. Οι κορυφές των τριγώνων μεταφέρονται στο σύστημα συντεταγμένων του επιπέδου και έπειτα ξεσκαρτάρεται η z συνταταγμένη για να προκύψει η προβολή.

Για τον έλεγχο αν ένα σημείο p βρίσκεται εντός ενός τριγώνου χρησιμοποιήθηκαν οι βαρυκεντρικές συντεταγμένες οι οποίες περιγράφουν το σημείο ως το γραμμικό συνδιαδμό των κορυφών του τριγώνου.

$$p = u \cdot p_1 + v \cdot p_2 + w \cdot p_3 \tag{1.1.1}$$

όπου p_1, p_2, p_3 οι κορυφές του τριγώνου και u, v, w οι βαρυκεντρικές συντεταγμένες.
Για να βρίσκεται το σημείο μέσα στο τρίγωνο αρκεί να ισχύει

$$u, v, w \geq 0 \text{ και } u + v + w = 1. \tag{1.1.2}$$

Για τον υπολογισμώ των u, v, w χρησιμοποιήθηκαν οι εξισώσες: [?]

$$\begin{aligned}
u &= \frac{\begin{vmatrix} p_2.x - p_1.x & p.x - p_1.x \\ p_2.y - p_1.y & p.y - p_1.y \end{vmatrix}}{\begin{vmatrix} p_2.x - p_1.x & p_3.x - p_1.x \\ p_2.y - p_1.y & p_3.y - p_1.y \end{vmatrix}} \\
v &= \frac{\begin{vmatrix} p_3.x - p_1.x & p.x - p_1.x \\ p_3.y - p_1.y & p.y - p_1.y \end{vmatrix}}{\begin{vmatrix} p_2.x - p_1.x & p_3.x - p_1.x \\ p_2.y - p_1.y & p_3.y - p_1.y \end{vmatrix}} \\
w &= 1 - u - v
\end{aligned} \tag{1.1.3}$$

1.1.2 Επιτάχυνση των υπολογισμών

Οι υπολογισμοί αυτοί είναι αρκετά αργοί για μεγάλα μοντέλα με πολλά τρίγωνα και πολλά σημεία που χρειάζεται να ελεγχθούν. Η βιβλιοθήκη numpy βοηθάει στην παραλληλοποίηση πολλών από αυτών των υπολογσμών αξιοποιώντας πολλαπλασιασμούς πινάκων ώστε αντί να γίνονται οι υπολογισμοί σε κάποιο βρόγχο για κάθε σημείο, γίνονται για όλα τα σημεία ταυτόχρονα.
Ορίζεται μία νέα συνάρτηση η οποία παίρνει σαν όρισμα ένα numpy array από σημεία μεγάθους $[n, 3]$ και επιστρέφει ένα boolean numpy array $[n, m]$ όπου m το πλήθος των τριγώνων. Για κάθε σημείο, που αντιστοιχεί σε μία γραμμή στον πίνακα που επιστρέφεται, αν κάποια τιμή είναι True, σημαίνει ότι πέφτει πάνω στην προβολή κάποιοι τριγώνου του μοντέλου. Αν όλες οι τιμές στην γραμμή είναι False, σημαίνει ότι βρίσκεται εκτός της προβολής.

Επιπλέον, στην εξίσωση υπολογισμού των βαρυκεντρικών συντεταγμένων (??) κάποιες τιμές είναι ανεξάρτητες του σημείου p που ελέγχεται και μπορούν να προϋπολογιστούν.

Ακόμα χρειάζεται ο έλεγχος με όλα τα τρίγωνα του μοντέλου που καθυστερεί σημαντικά τους υπολογισμούς. Για να μειωθούν οι πράξεις που χρειάζονται, υλοποιήθηκε μία δομή KDTree η οποία χωρίζει τα τρίγωνα σε μικρότερες ομάδες ανάλογα με την θέση τους στον χώρο. Έτσι, ένα σημείο δεν ελέγχεται με τρίγωνα που βρίσκονται πολύ μακριά του.
Για την δημιουργία του KDTree, δίνονται σαν όρισματα οι προβολές των vertices του μοντέλου στο επίπεδο και ένα indexed list με τα τρίγωνα. Στον αναδρομικό αλγόριθμο κατασκευής, επιλέγεται ένας άξονας και βρίσκεται το μέσο σημείο των κορυφών επί του άξονα. Τα σημεία τότε χωρίζονται σε δύο ομάδες ανάλογα με την θέση τους προς τον άξονα. Για τα τρίγωνα θα ισχύει ότι οι κορυφές τους βρίσκονται αποκλιστηκά στην μία ή στην άλλη ομάδα, ή και στις δύο. Για τα τρίγωνα που όλες τους οι κορυφές βρίσκονται στην ίδια ομάδα, συνεχίζουν σαν όρισμα στην αναδρομική κλήση. Για αυτά που έχουν κορυφές σε διαφορετικές ομάδες, που τα τέμνει ο άξονας που επιλέχθηκε, αποθηκεύονται στον κόμβο του δέντρου.

Για τον έλεγχο των σημείων, ξεκινώντας από την ρίζα, ελέγχονται για συγκρούσης με τα τρίγωνα αποθηκευμένα στον κόμβο. Τα σημεία που βρίκσονται μέσα σε κάποιο από αυτά τα τρίγωνα δεν συνεχίζουν ενώ τα υπόλοιπα χωρίζονται σε δύο ομάδες ανάλογα με την θέση τους και συνεχίζουν στο αντίστοιχο παιδί. Με αυτόν τον τρόπο, γλυτώνονται πολλές συγκρύσεις τριγώνων-σημείων που δεν χρειάζονται.

Αυτή η βελτίωση είναι αισθητή για μεγάλα πλήθη σημείων και τριγώνων. Για τα 12.0k τρίγωνα έγιναν οι ακόλουθες μετρήσεις.

Πλήθος σημείων	Χρόνος χωρίς KDTree (sec)	Χρόνος με KDTree (sec)
150	0.125	0.032
500	0.392	0.063
1000	0.765	0.097
5000	5.02	0.403
10000	11.66	0.728
15000	N/A	1.216

Για 15k σημεία, ο αλγόριθμος χωρίς KDTree δεν ολοκληρώθηκε λόγω της μεγάλης πολυπλοκότητας και το πρόγραμμα τερματίστηκε με error. Αντίθετα, ο αλγόριθμος με KDTree ανταπεξήλθε στο απαιτητικό πλήθος σημείων και είχε παντού σημαντικά μικρότερο χρόνο εκτέλεσης.

Bibliography

[1] Philippe B. Laval. Mathematics for computer graphics - barycentric coordinates. https://users.csc.calpoly.edu/~zwood/teaching/csc471/2017F/barycentric.pdf, November 2003. Section 1.1.