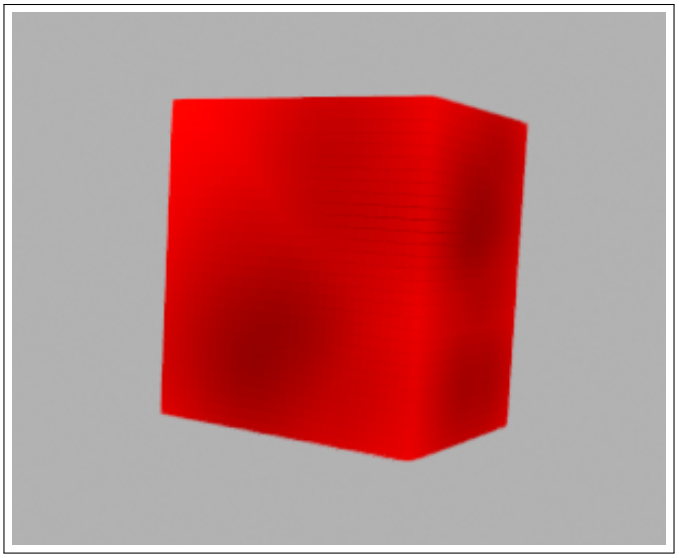
1. Part B

1.1 Δημιουργια Signed Distance Function

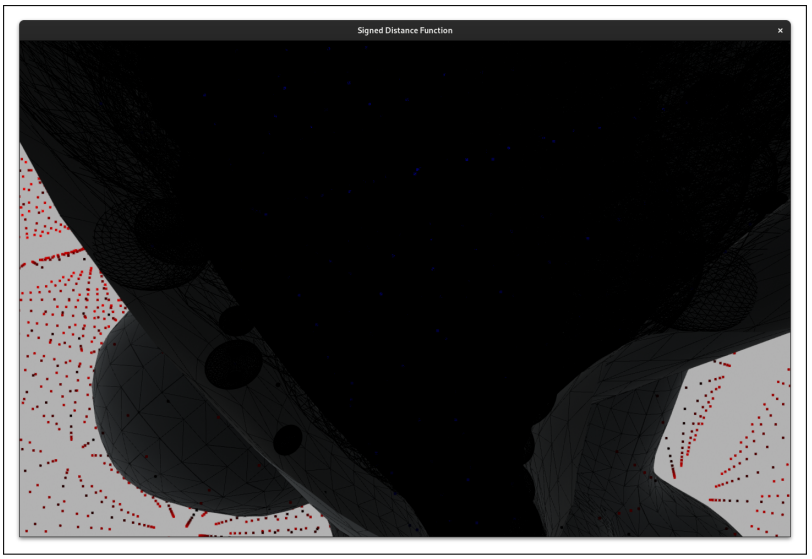
Έχοντας μεθόδους που υπολογίζουν την απόσταση ενός σημείου από το mesh, και αν βρίσκεται εντός ή εκτός αυτού, δειγματοληπτείται το AABB box του μοντέλου ώστε να δημιουργηθεί ένα ομοιόμορφο grid με σημεία που είναι γνωστές οι αποστάσεις τους από το μοντέλο. Επιλέχθηκαν 30 σημεία ανά άξονα, $30^3 = 27k$ σημεία σύνολο, που προσφέρουν αρκετά καλή ακρίβεια σε επόμενα ερωτήματα.

Για επαλήθευση ότι οι αποστάσεις των σημείων ευπολογίστηκαν σωστά, χρωματίζονται με βάση την απόστασή τους. Από μακρία διακρίνεται αχνά, το περίγραμμα της γάτας.



Σχήμα 1.1.1: Φαίνεται το περίγραμμα τις γάτας στα πιο σκούρα σημεία, τα οποία βρίσκονται πιο κοντά στο mesh

Για τα εσωτερικά σημεία, δημιουργούνται σφαίρες με κέντρο το σημείο και ακτίνα την απόσταση του από το mesh. Αν οι αποστάσεις έχουν υπολογιστεί σωστά, οι σφαίρες δεν θα φάινονται από έξω αλλά θα εφάπτονται στο εσωτερικό του mesh. Όντως επαληθεύται αυτό όπως φαίνεται στην εικόνα 1.1.2.

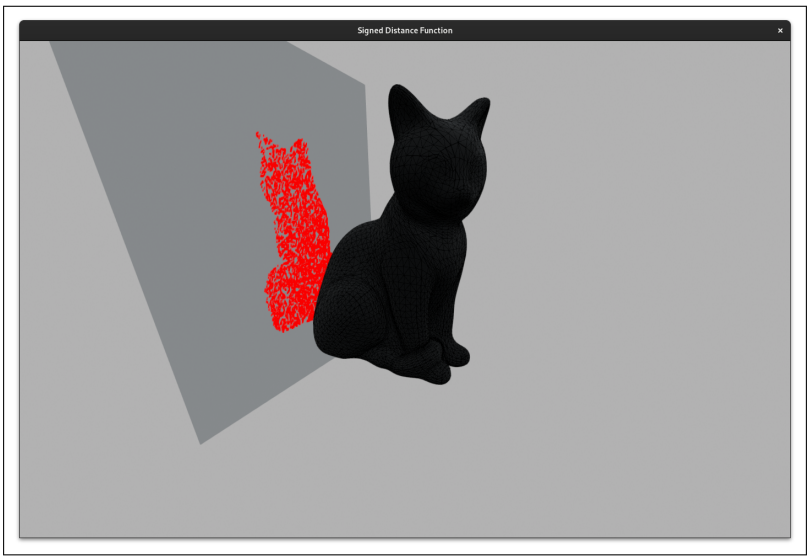


Σχήμα 1.1.2: Οι σφαίρες εφάπτονται στο εσωτερικό του mesh, όπως αναμένεται

Οι αποστάσεις που υπολογίστηκαν για αυτό το grid χρησιμοποιούνται σε έναν trilinear interpolator της scipy [1] για να δημιουργηθεί το signed distance function.

1.2 Ray Marching Algorithm

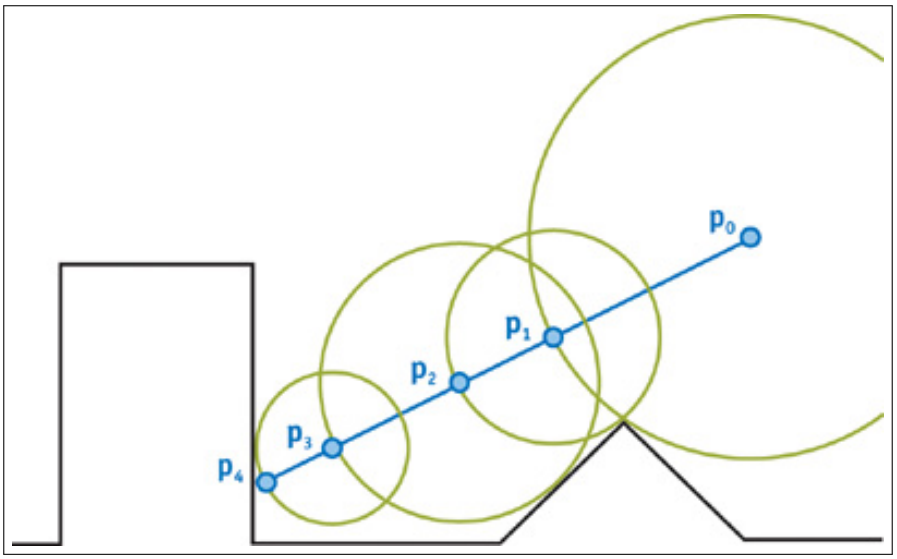
Με το signed distance fucntion που δημιουργήθηκε, μπορεί να εφαρμοστεί ο αλγόριθμος ray marching για να υπολογιστεί η προβολή του mesh στο επίπεδο. Ο αλγόρθμος επιστέφει αν κάποιο ακτίνα που εκπέμπεται από κάποιο σημείο τέμνεται με το mesh. Έτσι, θεωρόντας ακτίνες που εκπέμπεται από τα τυχαία επιλεγμένα σημεία του επιπέδου, με κατεύθυνση κάθετη προς αυτό, δηλαδή την διεύθυνση του normal του, χρωματίζονται κόκκινα τα σημεία που οι ακτίνες τους τέμνουν το mesh και μαύρα τα υπόλοιπα.



Σχήμα 1.2.1: Προβολή του mesh στο επίπεδο με τον αλγόριθμο ray marching

Ο αλγόριθμος ray marching θεωρεί τις ακτίνες από τα σημεία ως μια συνράρτηση $f(t) = p + t \cdot d$, όπου p το σημείο αφαιτηρίας, d η κατεύθυνση της ακτίνας και t μια μεταβλητή. Αν $f(t_x)$ είναι εντός του mesh, τότε η ακτίνα τέμνει το mesh. Για να βρεθεί το t_x, αν υπάρχει, αυξητικά ελέγχονται διάφορα t. Η επιλογή των κατάλληλων t είναι κρίσιμη για την απόδοση

του αλγορίθμου και σε αυτό βοηθάει πολύ το signed distance function. Από το signed distance function, είναι γνωστή η ελάχιστη απόσταση κάθε σημείου από το mesh, έστω $sdf(t)$. Τότε είναι σίγουρο ότι το για $t < sdf(t)$ η ακτίνα δεν τέμνει το mesh. Έτσι επιλέγται βήμα λίγο μικτότερο του $sdf(t)$ για το επόμενο t. Αν το $sdf(t)$ είναι πολύ μικρό, τότε πιθανό να χρειαστούν πολλά βήματα μέχρι να φτάσει η ακτίνα στο mesh. Οπότε επιλέγεται κάποια απόσταση "κατώφλι" όπου αν $sdf(t) < \epsilon$, θεωτείται ότι η ακτίνα τέμνει το mesh. Σε αυτήν την υλοποίηση επιλέχθηκε $\epsilon = 10^{-3}$. [2]

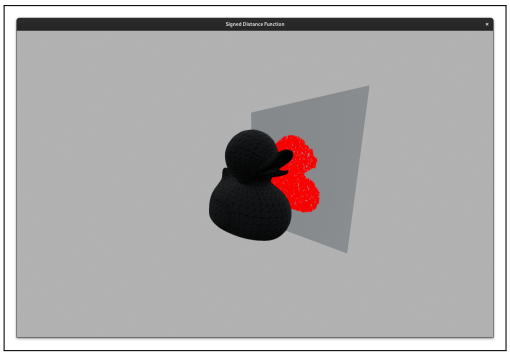


Σχήμα 1.2.2: Περιγραφή του αλγορίθμου ray marching [3]

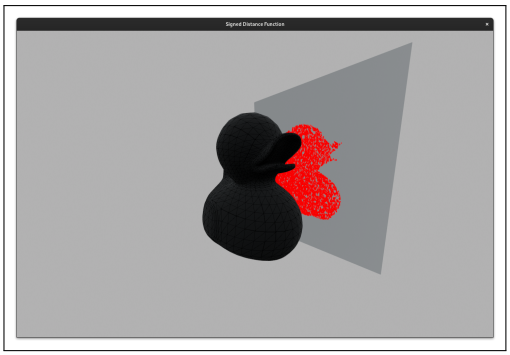
Ο αλγόριθμος τερματίζεται όταν $sdf(t) < \epsilon$ ή $t > t_{max}$, όπου t_{max} μια μέγιστη τιμή για το t, που σημαίνει ότι η ακτίνα έχει φτάσει πολύ μακρία από το mesh και δεν το τέμνει.

Σχήμα 1.2.3: Τελικές θέσεις των ακτίνων κατά την εκτέλεση του ray-marching αλγορίθμου

Ο αλγόριθμος ray-marching βασίζεται αρκετά στην ποιότητα του signed distance function. Αν το signed distance fucntion έχει δημιουργηθεί από ένα πολύ αραιό grid, τα αποτελέσματα θα είναι αλοιωμένα καθώς το $sdf(t)$ δεν θα επιστρέφει μια ικανοποιητική προσέγγιση της πραγματικής απόστασης του κάθε σημείου από το mesh.



Σχήμα 1.2.4: Προβολή της πάππιας με grid από 15 σημεία ανά άξονα



Σχήμα 1.2.5: Προβολή της πάππιας με gird από 30 σημεία ανά άξονα

Όπως φαίνεται από τις εικόνες 1.2, για μικρή δειγματοληψία grid χάνονται λεπτομέριες του mesh όπως το ράμφος της πάππιας. Με μεγαλύτερη δειγματοληψία βελτιώνεται το αποτέλεσμα και φαίνονται περισσότερες λεπτομέρειες αλλά όχι σε τέλειο βαθμό. Δείχνει ένα πρόβλημα του ray-marching αλγορίθμου και του signed distance function σε meshes με απότομες αλλαγές στην καμπυλότητα.

Bibliography

[1] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17:261–272, 2020.

[2] Ray marching and signed distance functions. https://jamie-wong.com/2016/07/15/ray-marching-signed-distance-functions/.

[3] William Donnelly. Gpu gems 2, chpater 8. per-pixel displacement mapping with distance functions. https://developer.nvidia.com/gpugems/gpugems2/part-i-geometric-complexity/chapter-8-pixel-displacement-mapping-distance-functions.