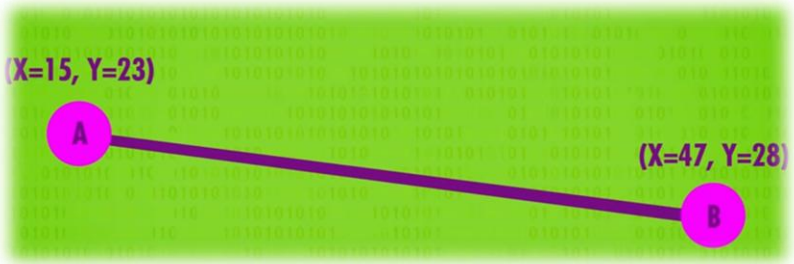
Γραφικά Υπολογιστών

Ιόνιο Πανεπιστήμιο
Τμήμα Πληροφορικής

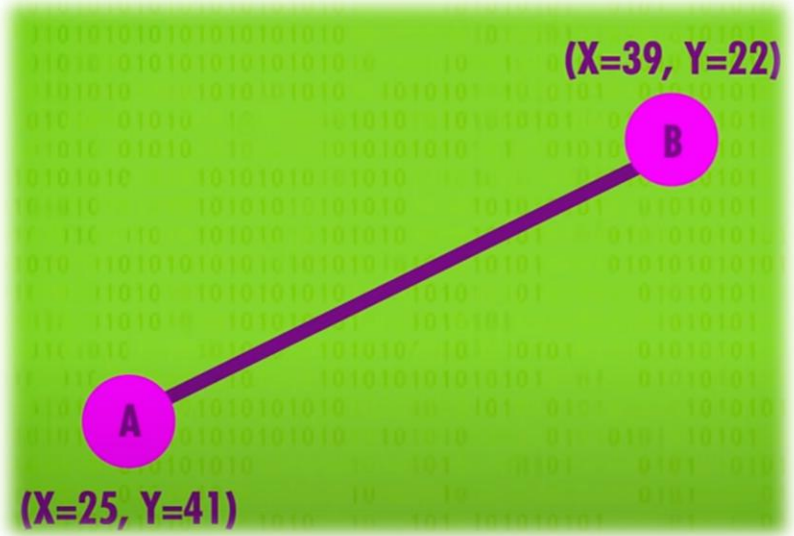
Στέργιος Παλαμάς, Επίκουρος Καθηγητής

Μάθημα 8:

Βασικές Αρχές 3D Γραφικών

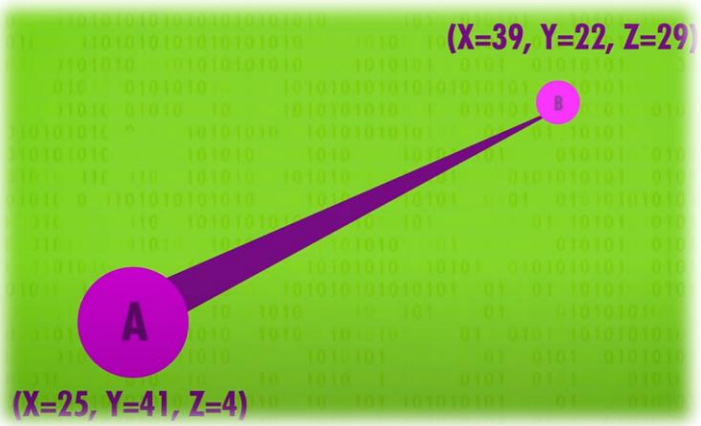


Στις δύο διαστάσεις (2D) μπορούμε να σχεδιάζουμε γραμμές μεταξύ δύο σημείων Α και Β, με την κατάλληλη συνάρτηση και τις συντεταγμένες των δύο σημείων.



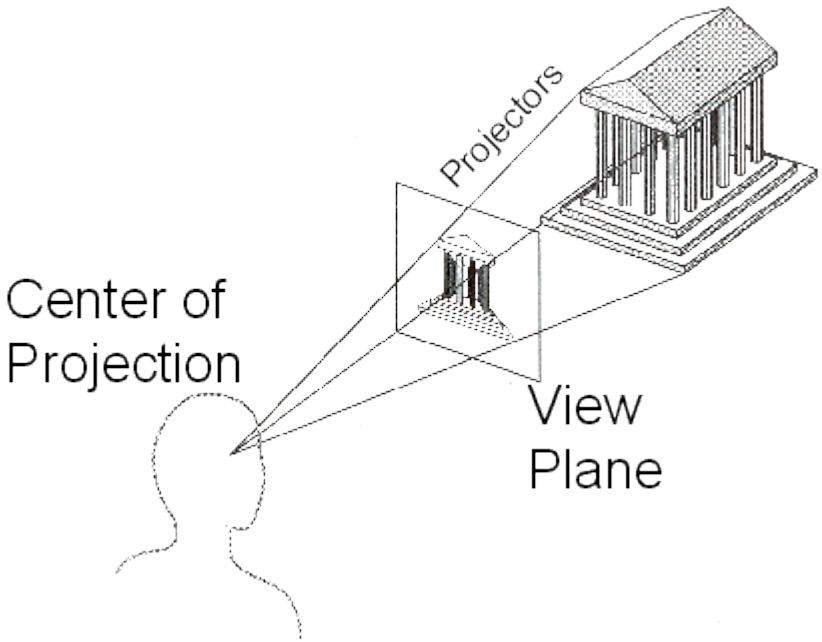
Αλλάζοντας τις συντεταγμένες Χ και Υ των δύο σημείων μπορούμε να «χειριστούμε τη γραμμή»

1. 3D Projection



Στις τρεις διαστάσεις (3D) προστίθεται και Τρίτη συντεταγμένη, το βάθος (Z).

Στην εικόνα το σημείο B (Z=29) είναι πιο μακριά από τον παρατηρητή (Z=4)

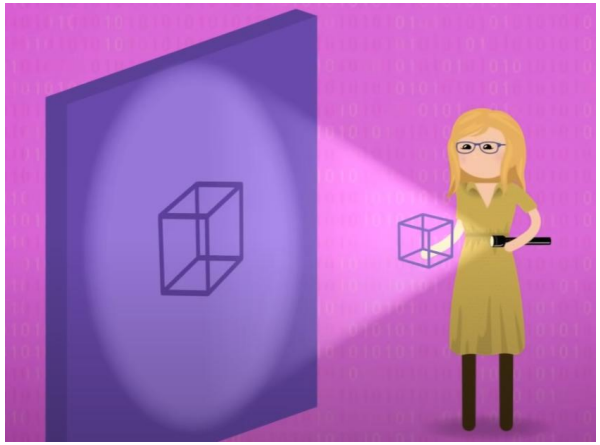


Καθώς όμως δε διαθέτουμε (ακόμα) τρισδιάστατες οθόνες, πρέπει τα 3D αντικείμενα να «προβληθούν» στην 2D επιφάνεια της οθόνης. Οι συντεταγμένες 3D (X,Y,Z) πρέπει να αντιστοιχηθούν σε 2D συντεταγμένες (X,Y)

Η διαδικασία αυτή λέγεται «3D Projection» και γίνεται με κατάλληλους αλγόριθμους.

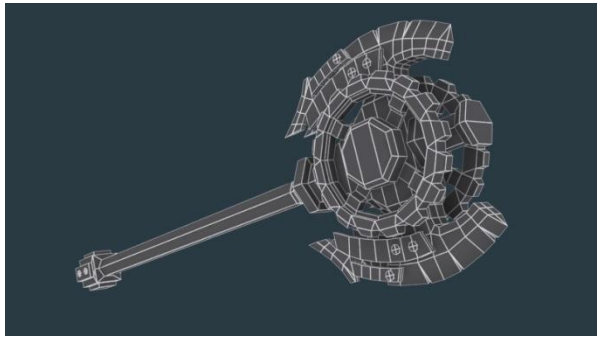
Όταν όλα τα 3D σημεία μετατραπούν σε 2D μπορούμε πλέον να χρησιμοποιήσουμε τους συνηθισμένες 2D συναρτήσεις για να ενώσουμε τα σημεία μεταξύ τους.

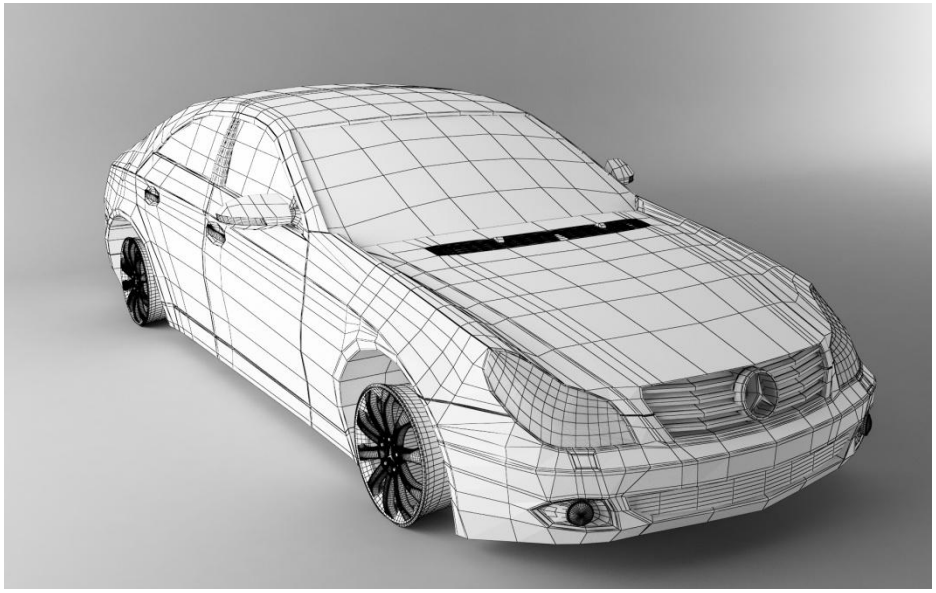
1. 3D Projection



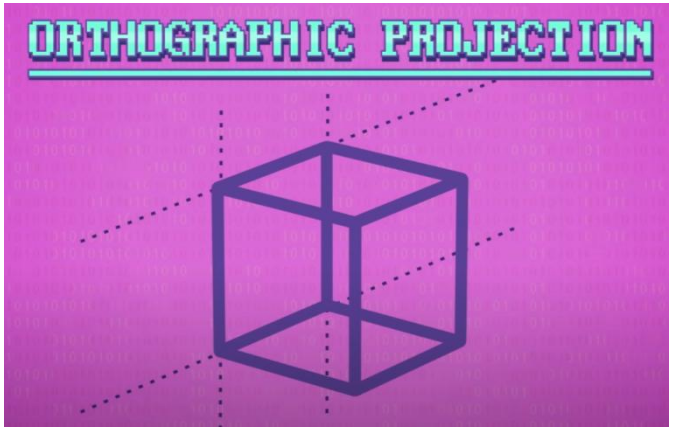
Όταν (μέσω του 3D projection) όλα τα 3D σημεία μετατραπούν σε 2D μπορούμε πλέον να χρησιμοποιήσουμε τους συνηθισμένες 2D συναρτήσεις για να ενώσουμε τα σημεία μεταξύ τους.

Αυτή η διαδικασία λέγεται «wireframe rendering”



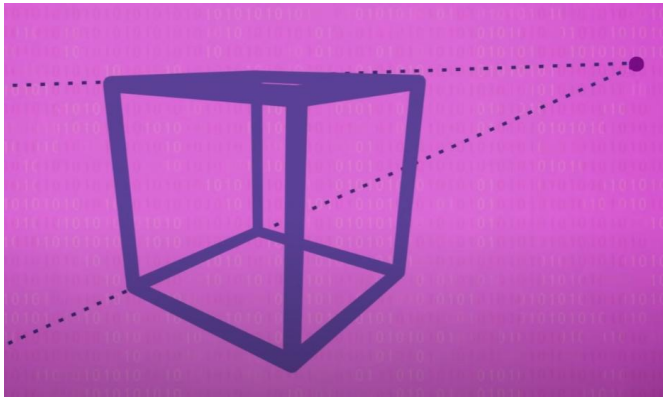


1. 3D Projection



Η προβολή του 3D αντικειμένου στις δύο διαστάσεις (3D Projection) μπορεί να γίνει με περισσότερους από έναν τρόπους.

Στην **Ορθογραφική Προβολή (Orthographic Projection)** , παράλληλες γραμμές του 3D αντικειμένου παραμένουν παράλληλες και στην 2D προβολή του



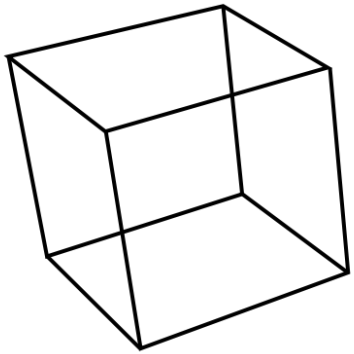
Στον πραγματικό 3D κόσμο όμως οι παράλληλες γραμμές φαίνεται να συγκλίνουν σε κάποια απόσταση από τον παρατηρητή (προοπτική).

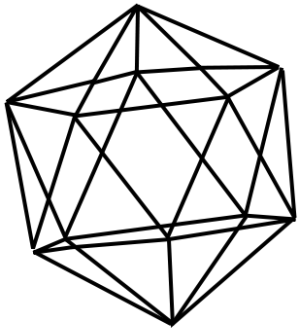
Αυτού του είδους η 3D προβολή (Projection) Ονομάζεται «Perspective Projection» (Προβολή Προοπτικής).

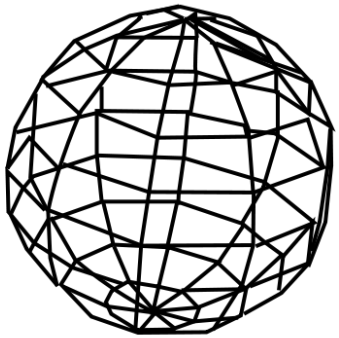
Ανάλογα με το σκοπό, επιλέγεται σε κάθε εφαρμογή ο κατάλληλος τύπος Projection.

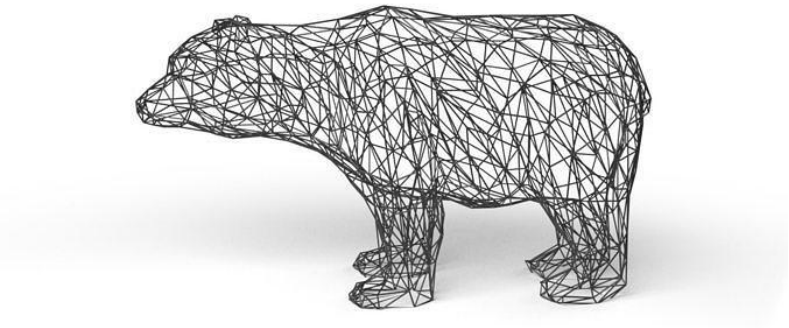
1. 3D Projection

2. Polygons





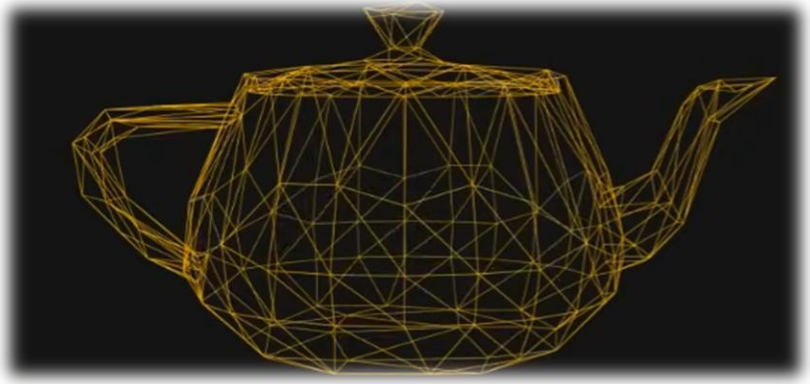


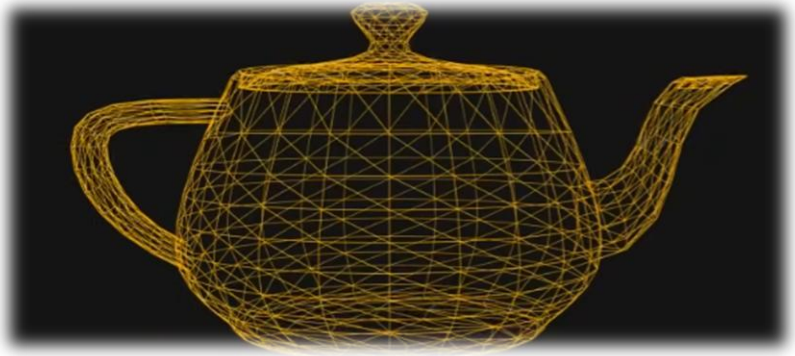


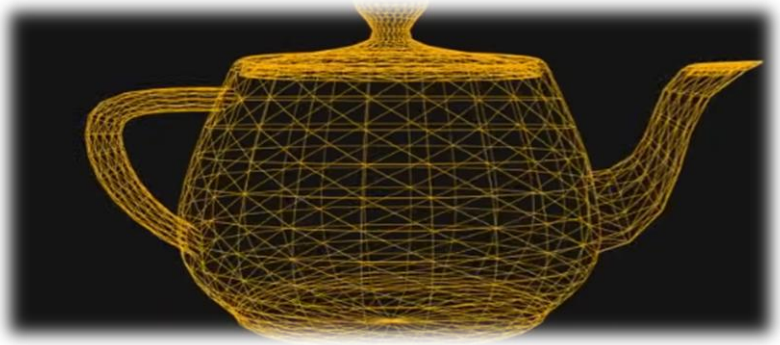
Ενώ η σχεδίαση απλών αντικειμένων όπως ο κύβος μπορεί να γίνει με απλές γραμμές, για πιο πολύπλοκα 3D αντικείμενα απαιτούνται πολύγωνα (Polygons).

Μια συλλογή από πολύγωνα που αναπαριστούν ένα αντικείμενο , ονομάζεται «mesh» (πλέγμα)

2. Polygons





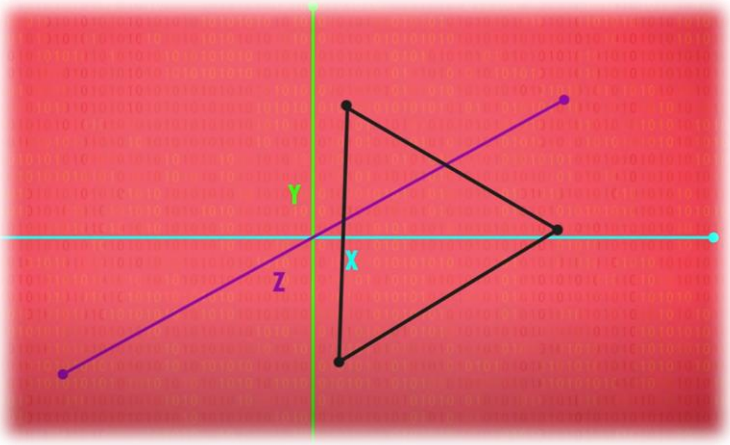


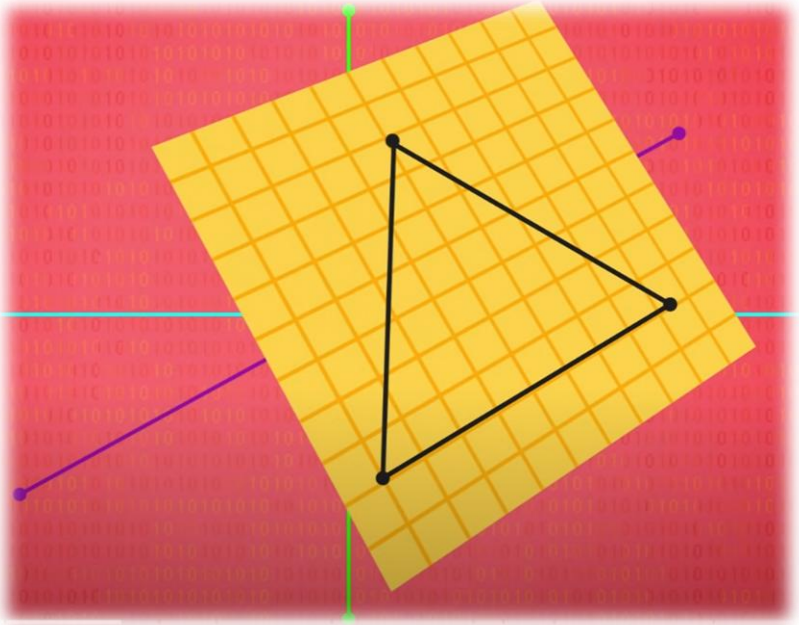
Όσο πιο πυκνό είναι το πλέγμα (mesh), τόσο καλύτερη και λεπτομερέστερη είναι η απεικόνιση του αντικειμένου. Αυξάνεται όμως σημαντικά ο αριθμός των πολυγώνων που χρησιμοποιούνται (polygon count) και επομένως και των υπολογισμών που πρέπει να γίνονται για το χειρισμό τους και την απεικόνισή τους.

Στις εφαρμογές 3D γραφικών – ιδίως στις πραγματικού χρόνου όπως τα παιχνίδια και οι προσομοιώσεις – πρέπει να επιτευχθεί μια ισορροπία ανάμεσα στην ακρίβεια απεικόνισης (polygon count) και τους απαιτούμενος υπολογιστικούς πόρους.

Αν η ταχύτητα σχεδιασμού των frames πέσει κάτω από κάποιο αριθμό/sec η κίνηση δεν εκλαμβάνεται ως ομαλή

2. Polygons



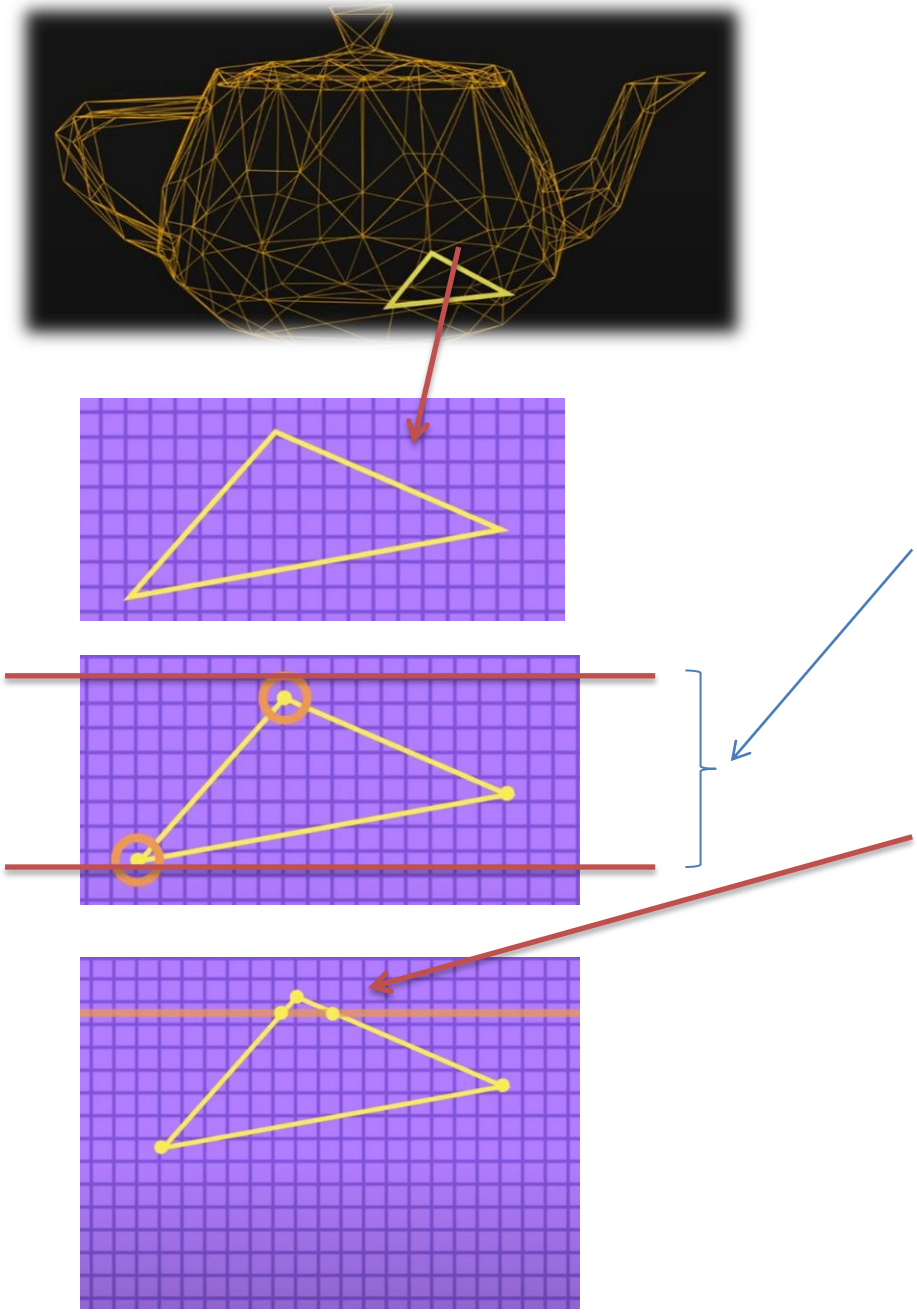


Στους υπολογιστές προτιμούμε να χρησιμοποιούμε **τρίγωνα αντί για πολύγωνα.**

Ο λόγος είναι η μεγαλύτερη απλότητα καθώς και το γεγονός ότι τρία σημεία στο χώρο ΟΡΙΖΟΥΝ ΜΟΝΟΣΗΜΑΝΤΑ ένα επίπεδο που διέρχεται από αυτά. Κάτι που δε συμβαίνει με 4 και περισσότερα σημεία.

Επομένως με τη χρήση Τριγώνων μπορούμε να σχηματίσουμε 3D αντικείμενα αποτελούμενα από μονοσήμαντεςς μικρές τριγωνικές επιφάνειες.

3. Scanline Rendering



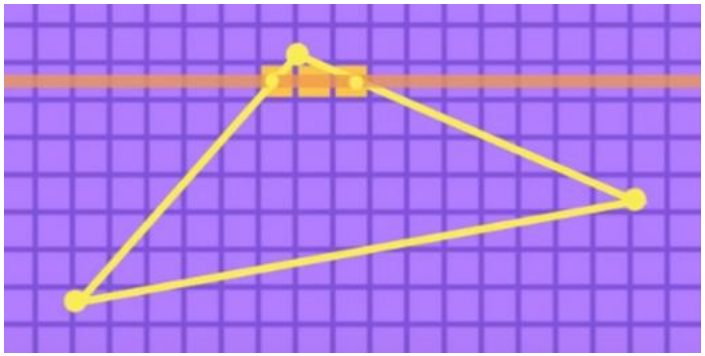
Ένα άλλο θέμα στην απεικόνιση 3D αντικειμένων σε μια 2D οθόνη , είναι το «γέμισμα» του εσωτερικού των τριγώνων που απαρτίζουν το αντικείμενο. Πρέπει λοιπόν να δούμε ποια pixels θα χρειαστεί να χρωματιστούν.

Ένας απλός και παλιός αλγόριθμος για αυτή τη διαδικασία είναι το “scanline rendering” (Πανεπιστήμιο Utah, 1967!).

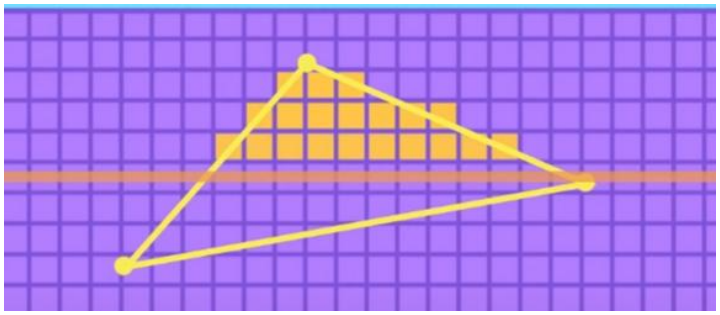
1 Ο Αλγόριθμος πρώτα εξετάζει τις τρείς κορυφές του τριγώνου (μετά το 3D projection) Και βρίσκει τη χαμηλότερη και την υψηλότερη από αυτές. Θα λάβει στη συνέχεια υπόψη μόνο τις γραμμές Pixels ανάμεσα σε αυτές

2. Μετά ξεκινά να σαρώνει μια προς μία τις γραμμές από πάνω προς τα κάτω. Σε κάθε γραμμή εξετάζει σε ποια σημεία η ευθεία που διέρχεται από το κέντρο των pixels της γραμμής τέμνει το τρίγωνο.

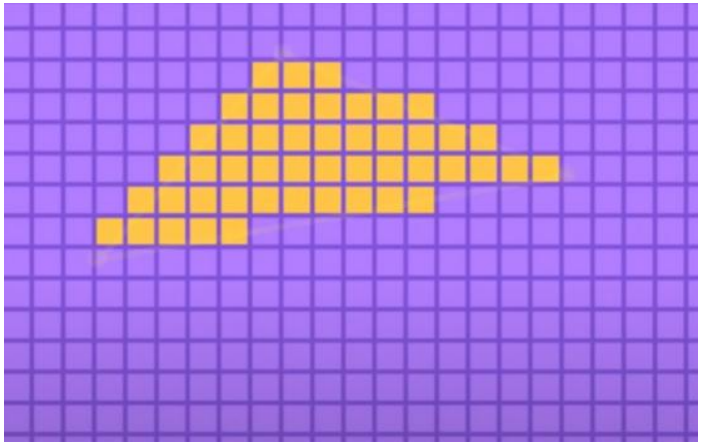
3. Scanline Rendering



3. Στη συνέχεια χρωματίζονται όλα τα Pixels ανάμεσα στα δύο σημεία τομής.



4. Ο αλγόριθμος συνεχίζει με τον ίδιο τρόπο γραμμή προς γραμμή μέχρι να ολοκληρωθεί η σάρωση και ο χρωματισμός όλων των γραμμών που περιλαμβάνει το τρίγωνο.



Ο ρυθμός με τον οποίο γεμίζονται τα τρίγωνα /πολύγωνα ονομάζεται «fill rate»

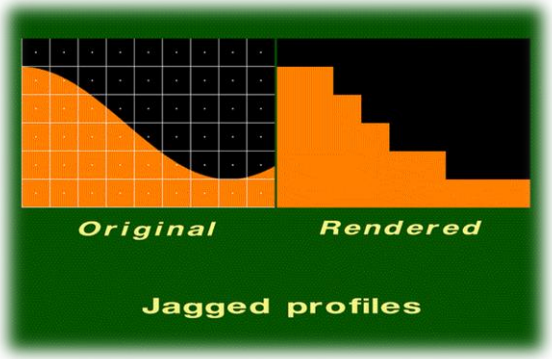
4. Anti-Aliasing

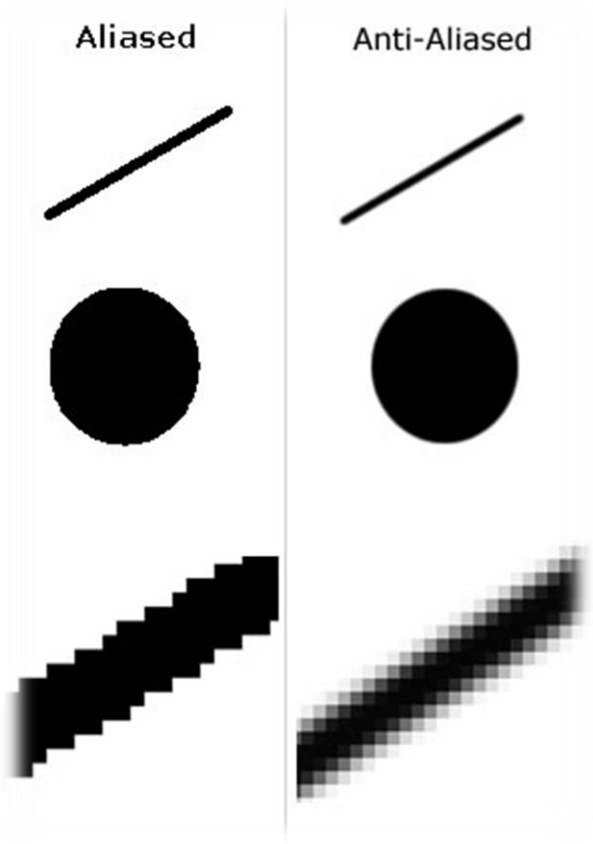
Το φαινόμενο σχηματισμού «σκαλοπατιών» κατά το γέμισμα των πολυγώνων που οφείλεται στο χωρισμό της οθόνης σε pixels ονομάζεται Aliasing

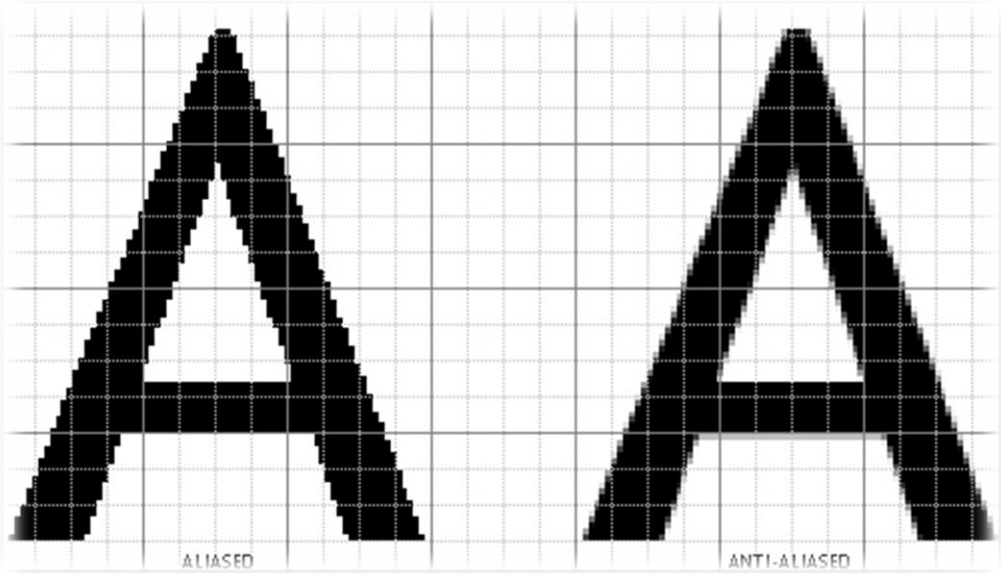
4. Anti-Aliasing

Το Aliasing γίνεται λιγότερο ορατό όταν αυξάνεται ο αριθμός των Pixels, για δεδομένο μέγεθος οθόνης, καθώς μικραίνει το μέγεθός τους κάνοντας λιγότερο ορατά τα «σκαλοπάτια» στις ακμές.

4. Anti-Aliasing



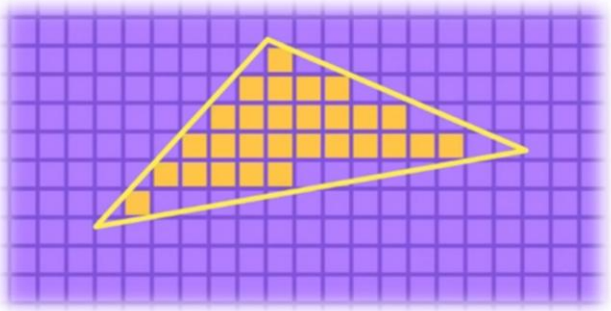




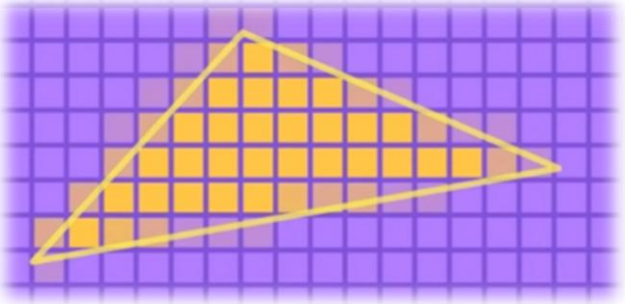
Το φαινόμενο του Aliasing ΔΕΝ αφορά μόνο τα 3D γραφικά στους υπολογιστές αλλά και τη σχεδίαση 2D, όπως σε γεωμετρικά σχήματα, χαρακτήρες στην οθόνη, εικονίδια κλπ.

Οι τεχνικές άμβλυνσης του φαινομένου ονομάζονται anti-aliasing και ουσιαστικά βασίζονται στο γέμισμα γειτονικών pixels με αποχρώσεις του χρώματος της ακμής (πχ αποχρώσεις του γκρι) ώστε να δημιουργείται η οπτική αίσθηση της εξομάλυνσής της.

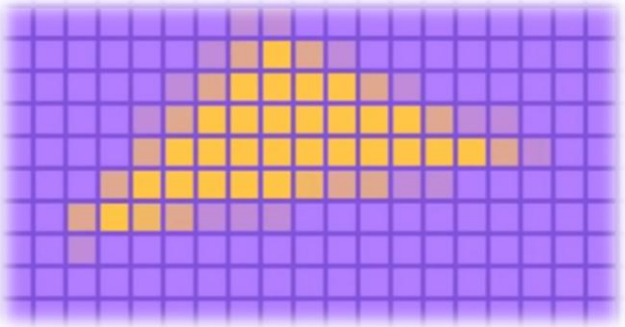
4. Anti-Aliasing



Κατά το γέμισμα του πολυγώνου τα Pixles που περιέχονται ολόκληρα στο εσωτερικό του χρωματίζονται με το βασικό χρώμα.



Τα pixels που περιέχονται εν μέρει στο πολύγωνο χρωματίζονται με πιο ανοικτές αποχρώσεις του βασικού χρώματος, δημιουργώντας στο μάτι ένα πιο ομαλό «σβήσιμο» της γραμμής.

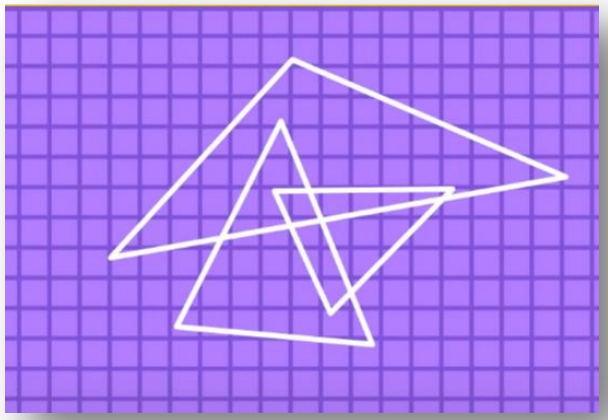


Το τελικό αποτέλεσμα με εφαρμογή του anti-aliasing είναι πολύ πιο ικανοποιητικό .

4. Anti-Aliasing

Το anti-aliasing χρησιμοποιείται ευρέως και σε εφαρμογές 2D για την εξομάλυνση χαρακτήρων, εικονιδίων κλπ σε γραφικά περιβάλλοντα εργασίας

5. Occlusion



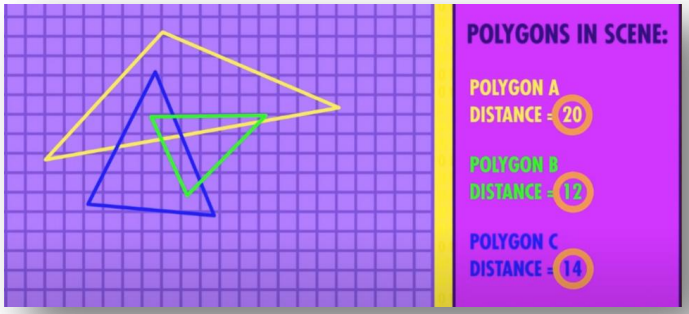
Σε μια 3D σκηνή που σχεδιάζεται από κάποια γωνία θέασης, αντικείμενα θα επικαλύπτουν μερικώς ή εντελώς άλλα αντικείμενα. Το ίδιο ισχύει και για τα πολύγωνα/τρίγωνα.

Ο όρος που χρησιμοποιείται είναι το “Occlusion”.

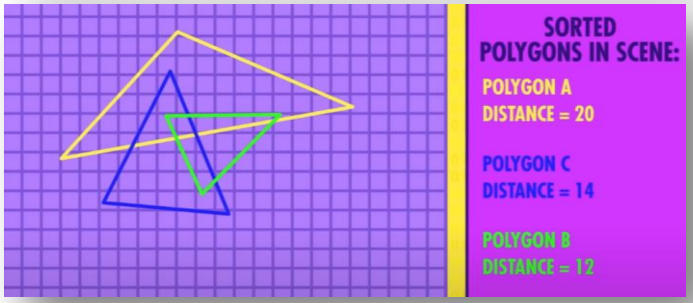
Ο πιο απλός τρόπος/αλγόριθμος για να λυθεί το πρόβλημα είναι να ταξινομηθούν όλα τα πολύγωνα από το πιο μακρυνό προς το πιο κόντινό και να σχεδιαστούν με αυτή τη σειρά. Έτσι στο τέλος θα ζωγραφιστούν τα πιο κοντινά στον παρατηρητή πολύγωνα , κρύβοντας άλλα που είναι σε μεγαλύτερο βάθος/απόσταση στη σκηνή.

Ο αλγόριθμος λέγεται «painter’s algorithm» καθώς το ίδιο κάνουν και οι ζωγράφοι ξεκινώντας από το background και προσθέτοντας αντικείμενα στον καμβά.

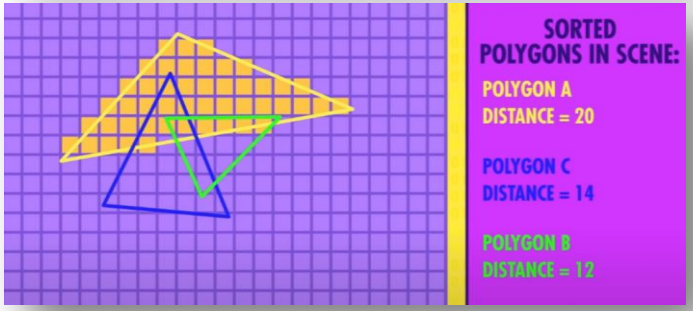
5. Occlusion – painter's algorithm



Ας θεωρήσουμε τρία επικαλυπτόμενα πολύγωνα στη σκηνή , διαφορετικά χρωματισμένα ανάλογα με το βάθος/απόστασή τους .

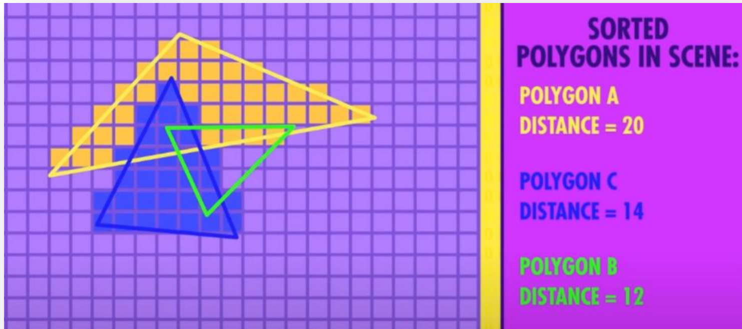


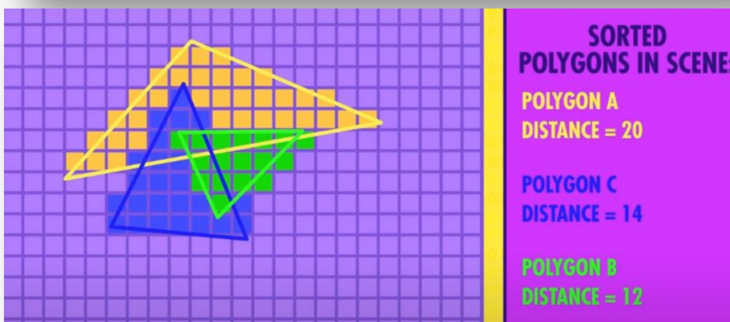
1. Το πρώτο που θα κάνει ο «painter's algorithm» είναι να ταξινομήσει τα πολύγωνα με το βάθος τους από το πιο μακρινό προς το πιο κοντινό.

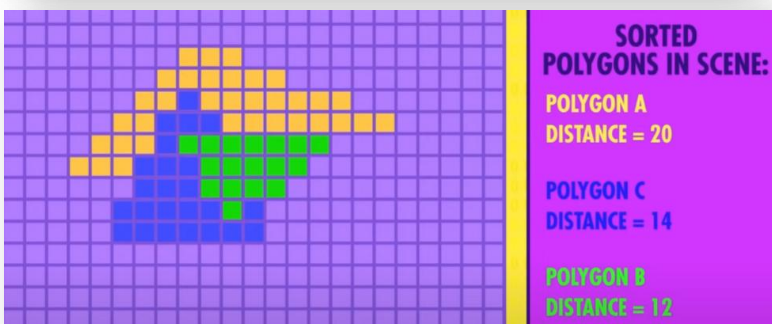


Στη συνέχεια ο «scanline algorithm» θα «γεμίσει» το πιο μακρινό πολύγωνο όπως έχουμε δει.

5. Occlusion – painter's algorithm



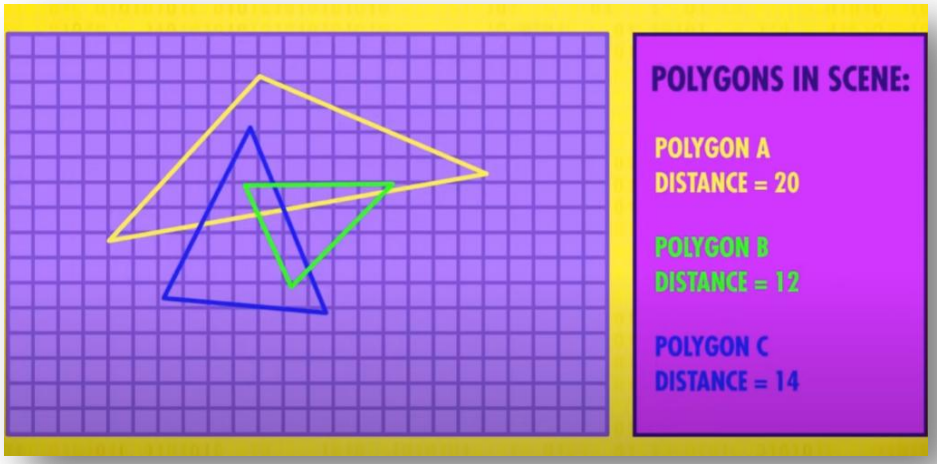


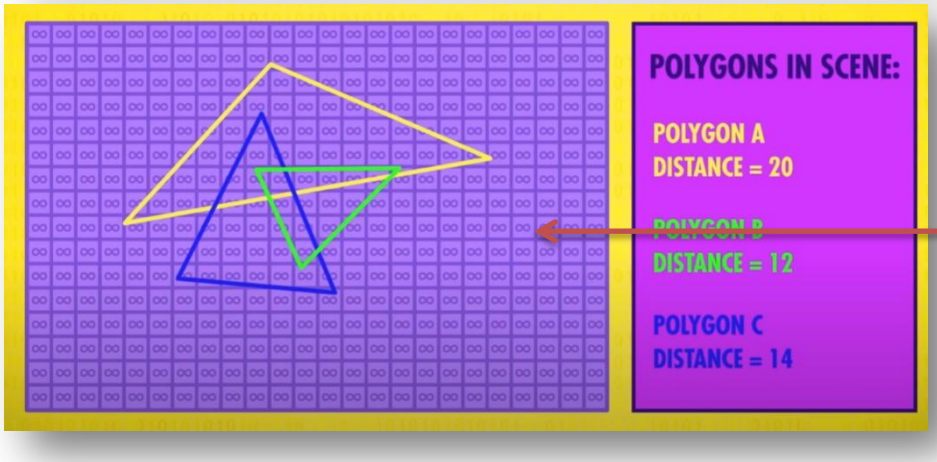


Το scanline rendering επαναλαμβάνεται στη συνέχεια για κάθε πολύγωνο ακολουθώντας την ταξινομημένη ως προς το βάθος σειρά.

Με αυτό τον τρόπο καταλήγουμε σε ένα σωστό οπτικά αποτέλεσμα όπου κάθε πολύγωνο επικαλύπτει αυτά που βρίσκονται πίσω του – σε πιο μακρινή απόσταση

5. Occlusion – Z-Buffering





Μια εναλλακτική τεχνική υλοποίησης του occlusion culling είναι η Z-Buffering.

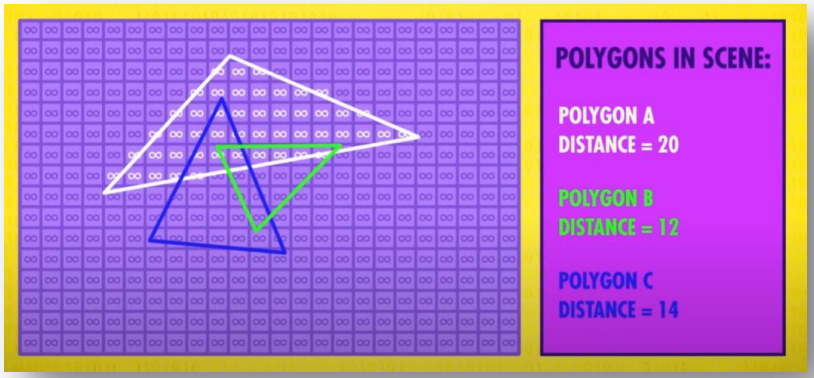
Είναι μια πιο γρήγορη τεχνική καθώς δεν απαιτεί την ταξινόμηση των πολυγώνων ως προς την απόστασή τους.

Το Z-buffer είναι μια περιοχή της μνήμης που αποθηκεύει έναν πίνακα (matrix) που αντιστοιχεί στα Pixels της οθόνης.

Σε κάθε θέση του πίνακα (του Z-buffer) αποθηκεύεται η απόσταση του αντίστοιχου pixel από τη θέση σχεδίασης/παρατήρησης της σκηνής.

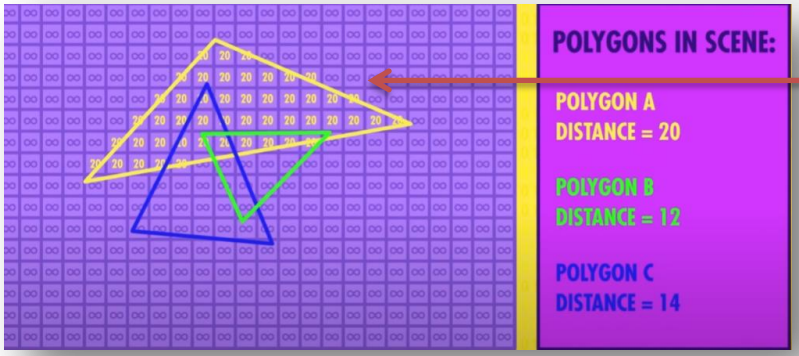
1. Αρχικά όλα τα pixels αρχικοποιούνται με μια πολύ μεγάλη απόσταση που αντιστοιχεί στο άπειρο.

5. Occlusion – Z-Buffering

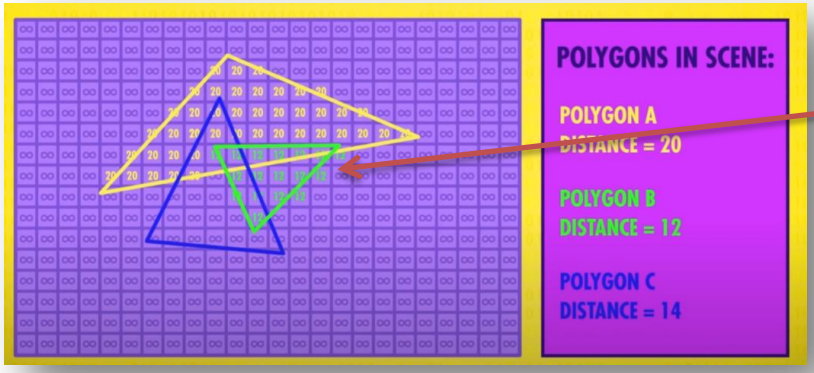


2. Στη συνέχεια εξετάζεται το πρώτο πολύγωνο στη (μη ταξινομημένη) λίστα – το Α.

Ακολουθείται μια παρόμοια τεχνική με το svanline rendering αλλά αντί να «γεμίζουμε» το πολύγωνο, συγκρίνουμε το βάθος κάθε σημείου του πολυγώνου με την τιμή που είναι αποθηκευμένη για το αντίστοιχο pixel στο z-buffer και κρατάμε την μικρότερη.

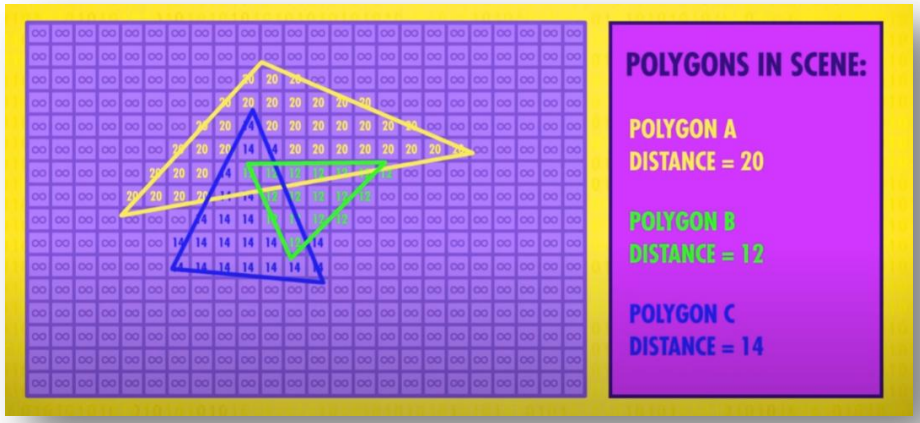


Έτσι μετά την εξέταση του πολυγώνου Α οι αντίστοιχες θέσεις στο Z-buffer θα έχουν τιμή 20 αντί για άπειρο που ήταν πριν.



3. Στη συνέχεια εξετάζεται το επόμενο πολύγωνο, το Β.
Στις θέσεις του Z-buffer που αντιστοιχούν στα pixels εντός του πολυγώνου Β , θα μπει η τιμή 12, αφού αυτή είναι μικρότερη και από το άπειρο αλλά και από το 20 που έχουν pixels που ανήκουν στο πολύγωνο Α και εξετάστηκαν στο προηγούμενο βήμα

5. Occlusion – Z-Buffering



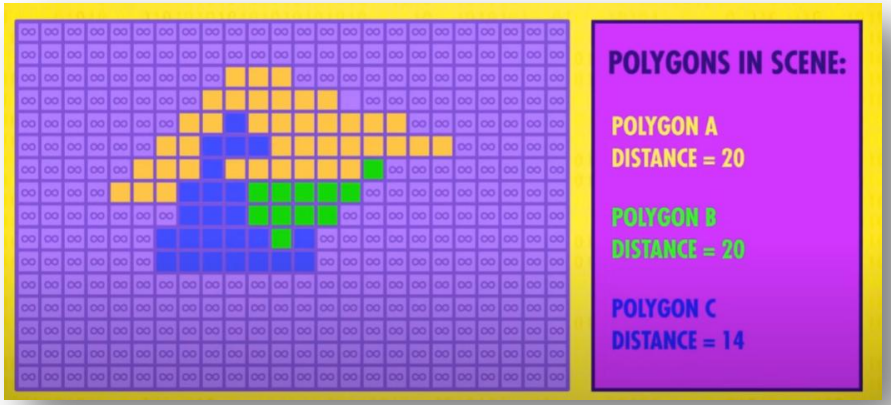
4. Τέλος θα εξεταστεί και το πολύγωνο C. Στο Z-buffer θα μπει η τιμή 14 για τα pixels που επικαλύπτουν το άπειρο του φόντου ή το 20 του πολυγώνου Α.

Δε θα επικαλύψουν όμως τα pixels που στο z-buffer έχουν τιμή 12 από το πολύγωνο Β καθώς το 12 είναι μικρότερο σαν τιμή από το 14.

Στο τέλος στο Z-buffer υπάρχει για κάθε pixel μια τιμή βάθους που όσο πιο μικρή είναι τόσο πιο κοντά είναι στη θέση παρατήρησης .

Το rendering Ολοκληρώνεται με μια παραλλαγή του scanline rendering όπου για το γέμισμα κάθε πολυγώνου δεν εξετάζεται μόνο αν τα pixels είναι ανάμεσα στα δύο σημεία τομής ώστε να γεμίσουν, αλλά εξετάζεται (με χρήση του z-buffer) και αν το pixel του συγκεκριμένου πολυγώνου θα φαίνεται τελικά ή θα επικαλυφθεί από Πιο κοντινό Pixel άλλου πολυγώνου

5. Occlusion – Z-fighting



Ένα πρόβλημα που ανακύπτει είναι τι γίνεται στην περίπτωση που δύο πολύγωνα έχουν το ίδιο βάθος;

Ποιο θα σχεδιαστεί τελικά;

Το αποτέλεσμα είναι συχνά απρόβλεπτο και οδηγεί σε ένα «τρεμόπαιγμα» στις προβληματικές επιφάνειες που εμφανίζεται στο σχεδιασμό της σκηνής ως τεχνούργημα (gltch) καθώς αντικρουόμενα ως προς το βάθος πολύγωνα εναλλάσσονται στην προτεραιότητα σχεδιασμού **(φαινόμενο Z-fighting)**.

https://www.youtube.com/watch?v=i3kH91NzwEk

Άλλο ένα θέμα είναι το λεγόμενο "backface culling". Όλα τα πολύγωνα έχουν δύο πλευρές. Συνήθως όμως ορατή είναι μόνο η μία – η εξωτερική.

Για λόγους βελτιστοποίησης , σχεδιάζεται ΜΟΝΟ η ορατή πλευρά του πολυγώνου και όχι η πίσω πλευρά (εξού και το backface) .

Σε κάποια πχ παιχνίδια λόγω bug μπορεί να βρεθούμε σε σημείο μη αναμενόμενο – πχ κάτω από το έδαφος – οπότε εμφανίζεται το πρόβλημα καθώς η κάτω πλευρά δε σχεδιάζεται αφού δεν αναμένονταν να είναι ορατή.

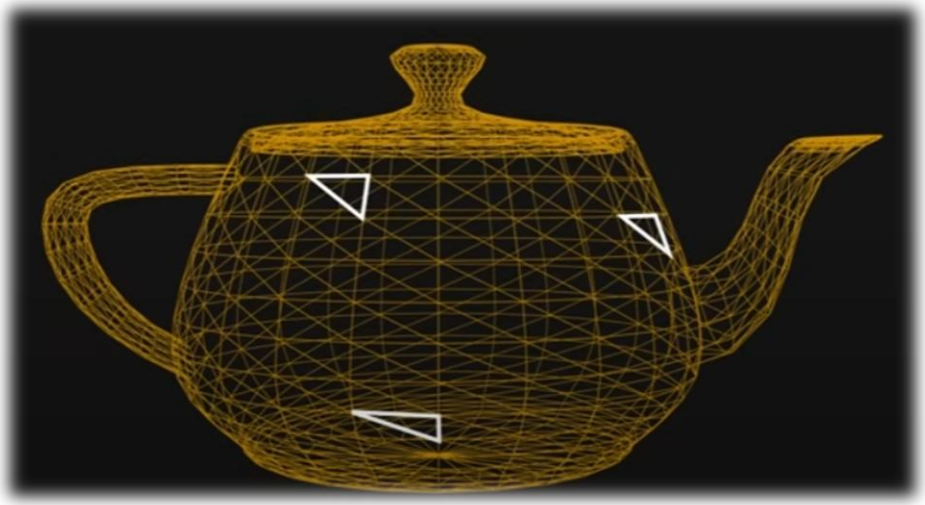
backface culling

6. Lighting/Shading

Το αποτέλεσμα του scanlight rendering είναι όπως φαίνεται δίπλα. Τα πολύγωνα γέμισαν αλλά το αντικείμενο δε φαίνεται ρεαλιστικό καθώς δεν έχει «όγκο».

Χρειαζόμαστε φωτισμό και σκίαση για να αποδοθεί σωστά η αίσθηση του τρισδιάστατου όπως στον πραγματικό κόσμο.

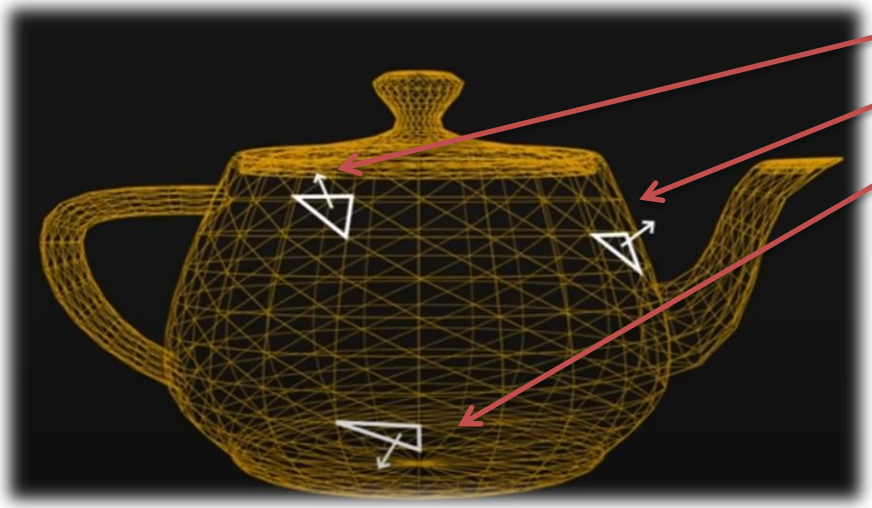
6. Lighting/Shading



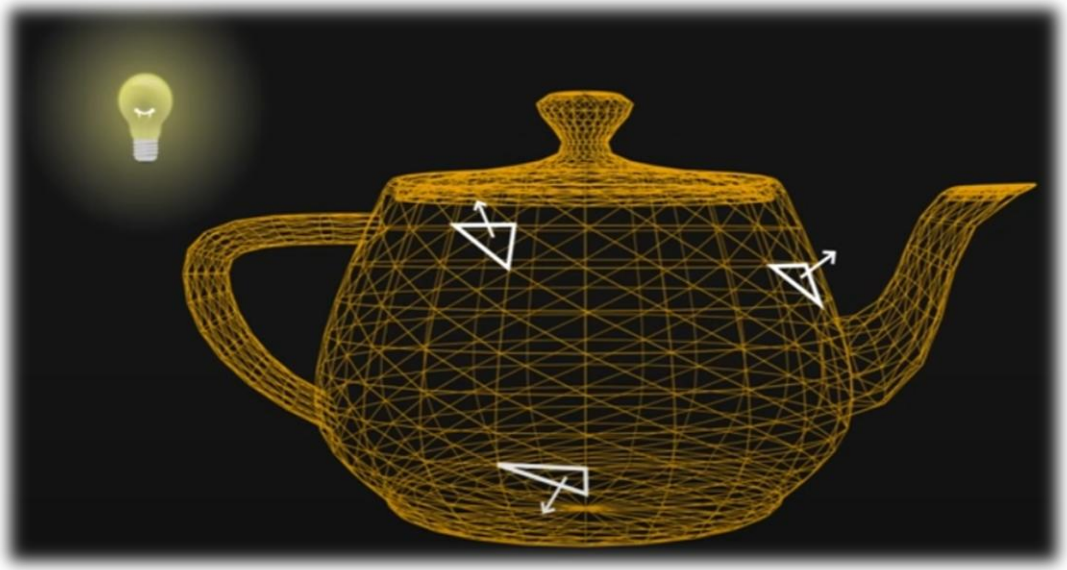
Τα πολύγωνα δεν είναι παράλληλα με την οθόνη αλλά είναι στραμμένα σε διάφορες κατευθύνσεις.

Μπορούμε να αναπαραστήσουμε την κατεύθυνση προς την οποία «βλέπει» ένα πολύγωνο με ένα διάνυσμα που ονομάζεται «surface normal».

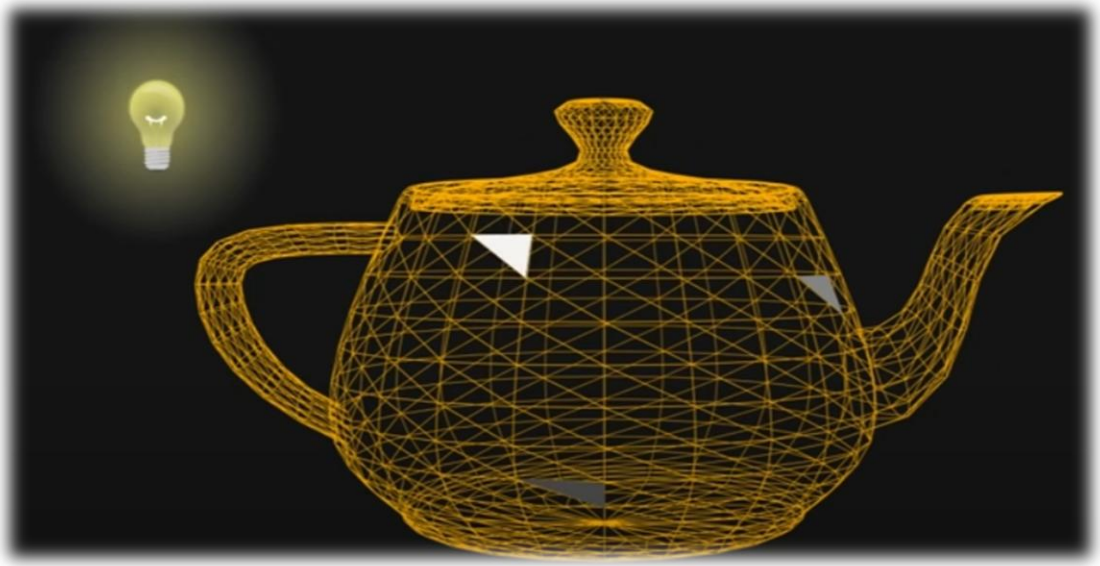
Τα surface normals είναι ΚΑΘΕΤΑ στην επιφάνεια κάθε πολυγώνου .



6. Lighting/Shading



Αν προσθέσουμε μια πηγή φωτισμού στη σκηνή τότε κάθε πολύγωνο θα φωτιστεί διαφορετικά ανάλογα με το πού είναι στραμμένο (άρα ανάλογα με το surface normal του).



Αυτά που το surface normal δείχνει προς τη πηγή φωτισμού θα φωτιστούν περισσότερο ενώ όσα δείχνουν στην αντίθετη κατεύθυνση (οπότε η αντίστοιχη επιφάνεια του πολυγώνου έχει «πλάτη» στο φώς) θα λάβουν λίγο ή και καθόλου φωτισμό.

6. Lighting/Shading

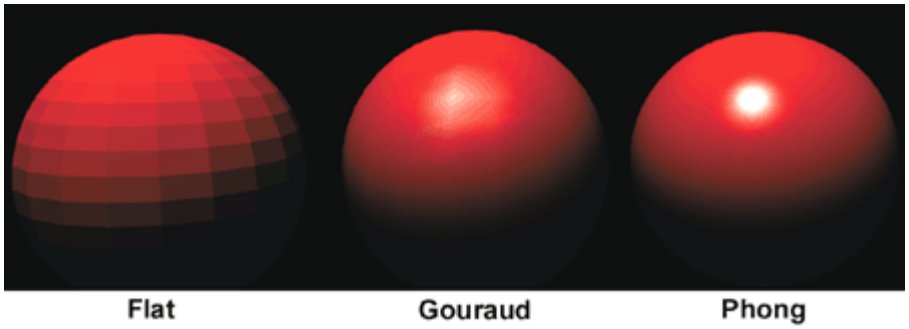
Το τελικό αποτέλεσμα είναι πολύ πιο ρεαλιστικό από πριν και το αντικείμενο απέκτησε «όγκο» και δείχνει όντως τρισδιάστατο. **Ωστόσο υπάρχει μια σοβαρή ατέλεια καθώς κάθε πολύγωνο χρωματίζεται με μια μόνο απόχρωση καθιστώντας ορατά τα πολύγωνα που απαρτίζουν το 3D αντικείμενο στον υπολογιστή.**

Αυτή η προσέγγιση λέγεται «FLAT SHADING» και είναι η πιο βασική – απλή μέθοδος φωτισμού/σκίασης (οπότε και η λιγότερο απαιτητική σε υπολογιστικούς πόρους).

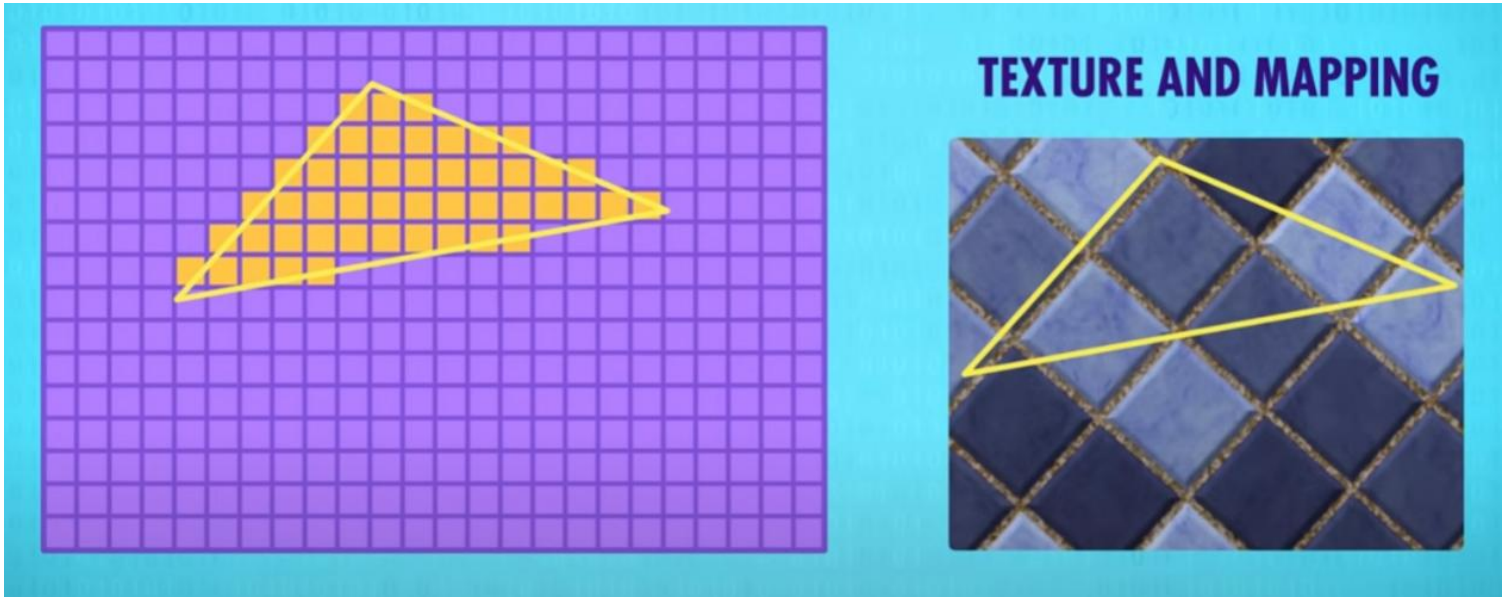
6. Lighting/Shading

Οι ατέλειες του Flat Shading οδήγησε στην ανάπτυξη πιο εξελιγμένων αλγόριθμων φωτισμού/σκίασης όπως **το GOURAUD Shading (**Henri Gouraud) **και το PHONG Shading (**Phong Bui Tuong).

Αυτοί αντί να εφαρμόζουν μια ενιαία απόχρωση σε όλη την επιφάνεια του πολυγώνου, το χρωματίζουν με μεταβλητές αποχρώσεις (πχ λαμβάνοντας υπόψη και τα γειτονικά πολύγωνα και τα surface normal τους, ώστε η μετάβαση μεταξύ τους να είναι ομαλή με διαβάθμιση).



7. Texture Mapping

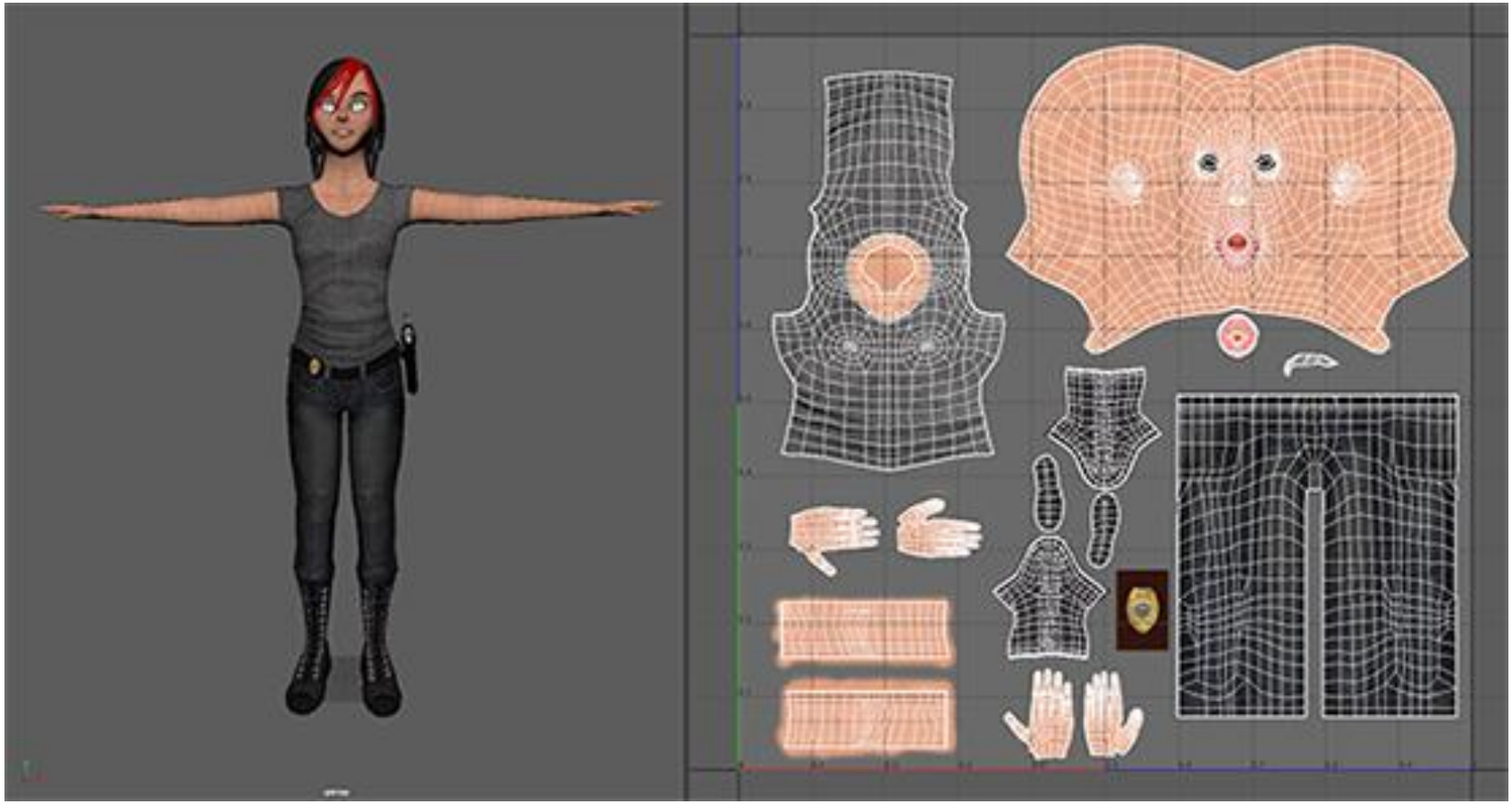


Ένας τρόπος να αυξήσουμε σημαντικά την ρεαλιστικότητα των 3D αντικειμένων χωρίς να αυξήσουμε παράλληλα τον αριθμό πολυγώνων, είναι εφαρμογή στην επιφάνειά τους εικόνων που μπορεί να προέρχονται από φωτογραφίες πραγματικών αντικειμένων ή να έχουν σχεδιαστεί ψηφιακά στον υπολογιστή.

Αυτές οι εικόνες ονομάζονται TEXTURES .

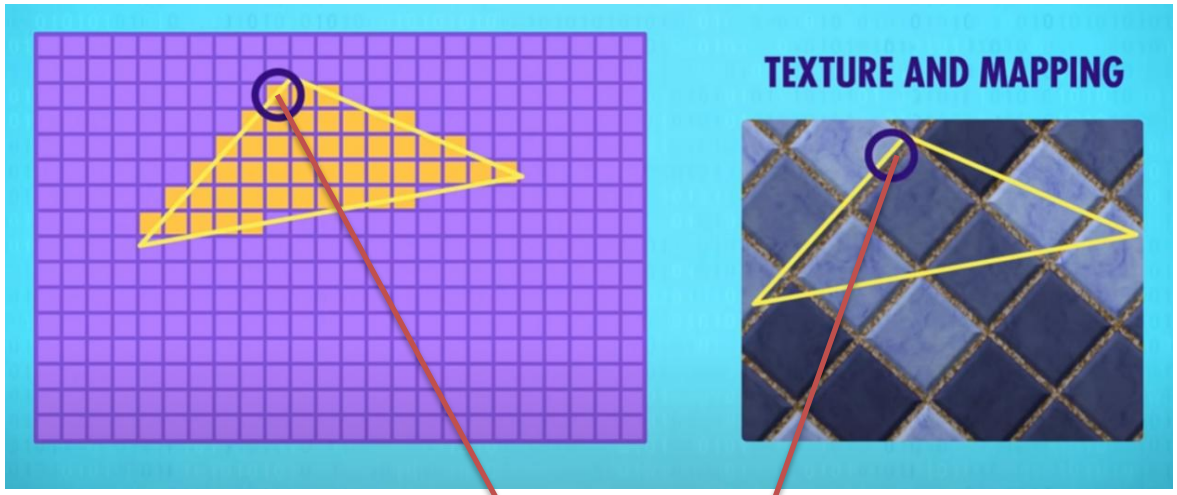
Τα textures είναι λοιπόν ψηφιακές εικόνες . Για να εφαρμοστούν σε ένα αντικείμενο πρέπει κατά το σχεδιασμό του αντικειμένου στο κατάλληλο λογισμικό (πχ 3DS Max, Blender) να γίνει το λεγόμενο TEXTURE MAPPING. Με τα κατάλληλα εργαλεία που δίνει το λογισμικό , αντιστοιχίζονται οι επιφάνειες του αντικειμένου σε κομμάτια του texture. Σα να «τυλίγουμε» κατά κάποιο τρόπο το texture γύρω από το αντικείμενο, να το «ντύνουμε με αυτό». Έτσι οι συντεταγμένες του πολυγώνου αντιστοιχίζονται με συντεταγμένες του Texture (UV Mapping).

7. Texture Mapping

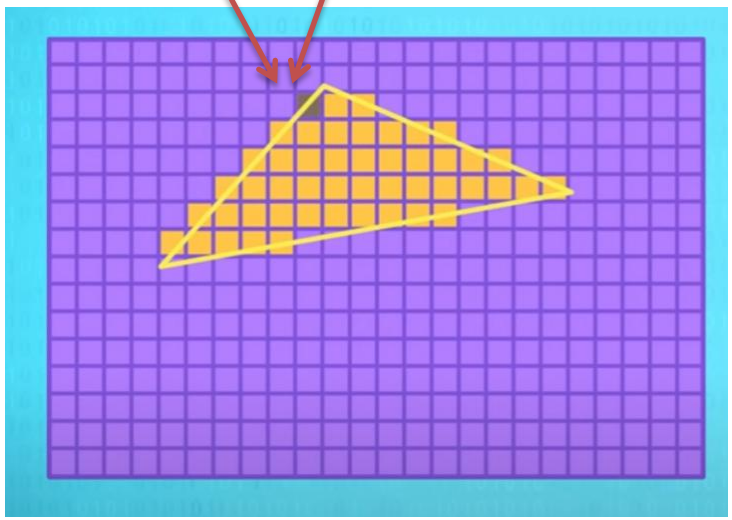


Εφαρμογή Texture σε 3D χαρακτήρα στο λογισμικό 3D Design “Maya”. Το λογισμικό «ξεδιπλώνει» τα πολύγωνα σε επιφάνεια και τους αντιστοιχίζουμε την εικόνα – texture.

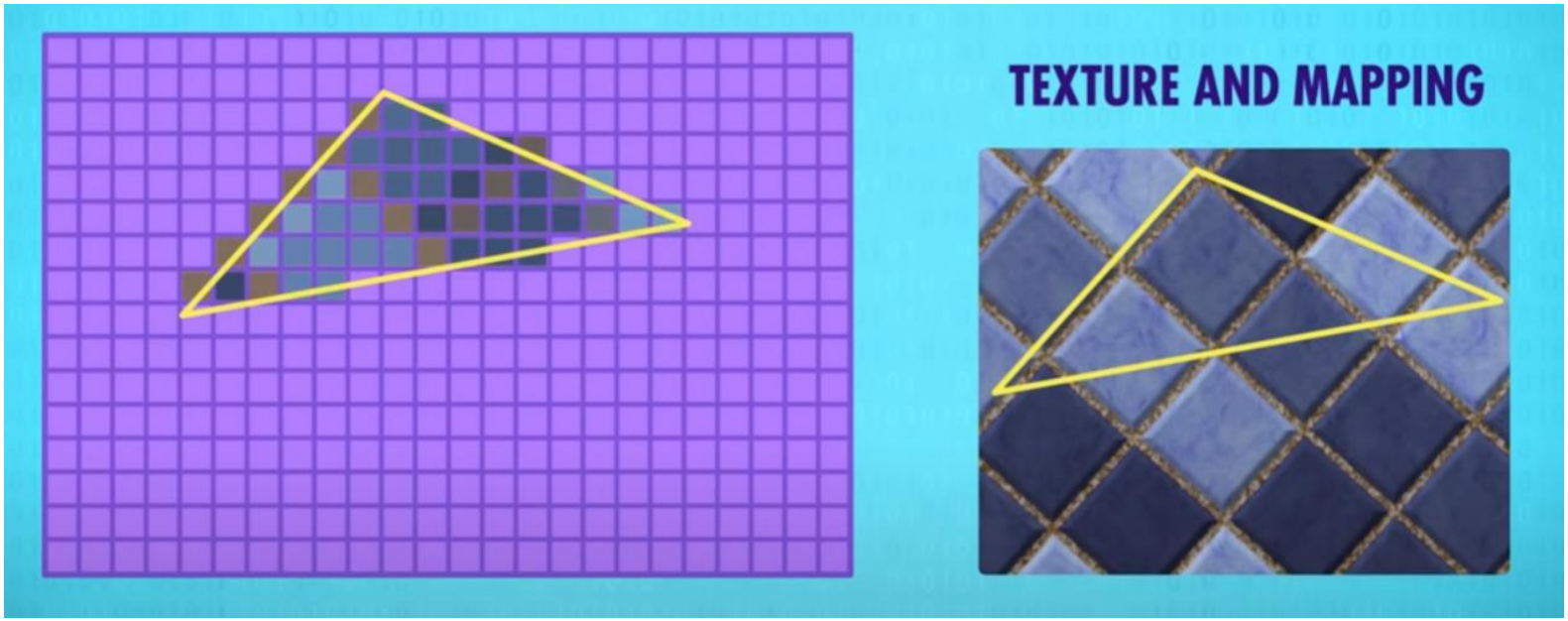
7. Texture Mapping

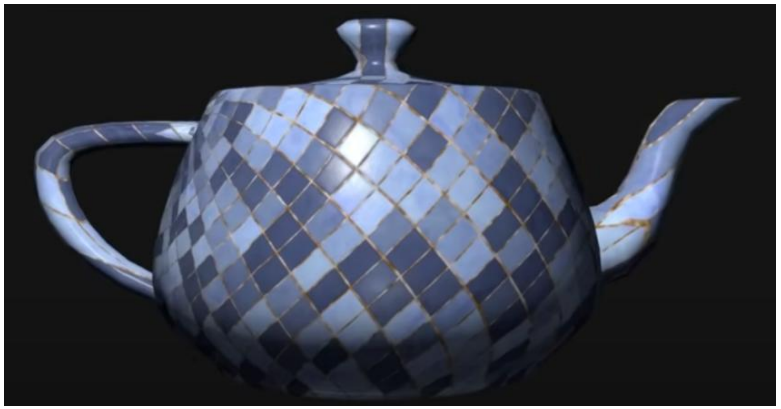


Κατά το γέμισμα του πολυγώνου, ο αλγόριθμος texturing θα βρει ποιο σημείο της εικόνας/texture αντιστοιχεί στο συγκεκριμένο Pixel του πολυγώνου, και θα το χρωματίσει με τη μέση τιμή των χρωμάτων του texture σε εκείνο το σημείο.



Η διαδικασία επαναλαμβάνεται για όλα τα pixels του πολυγώνου , τα οποία χρωματίζονται με βάση τις αντίστοιχες περιοχές του texture.





Συνδυάζοντας όλα τα προηγούμενα δημιουργούνται τα σύγγχρονα 3D γραφικά. Απαιτούνται πολλοί μαθηματικοί υπολογισμοί οι οποίοι όμως επαναλαμβάνονται ξανά και ξανά. Αυτό έχει οδηγήσει στο σχεδιασμό ηλεκτρονικών κυκλωμάτων που επιταχύνουν αυτούς τους υπολογισμούς , απαλλάσσοντας τη CPU από το φόρτο εργασίας.

Έτσι γεννήθηκαν οι GPU (Graphics Processing Units) που σήμερα είναι ενσωματωμένες σε όλες τις σύγχρονες κάρτες γραφικών.

Πλέον οι κάρτες γραφικών διαθέτουν δικιά τους μνήμη RAM για αποκλειστική χρήση (η οποία είναι μάλιστα πιο γρήγορη από τη RAM ττου υπολογιστή και όχι σπάνια πλέον μεγαλύτερη σε μέγεθος), ενώ ενσωματώνουν και κυκλώματα επιτάχυνσης άλλως συχνών λειτουργιών όπως τα Physics των 3D εφαρμογών.

Χαρακτηριστικό της ισχύος που έχουν αποκτήσει πλέον οι GPUs είναι ότι χρησιμοποιούνται για εφαρμογές τεχνητής νοημοσύνης, coin mining κλπ αντί για CPUs , εκτοξεύοντας για κάποιο χρονικό διάστημα τις τιμές του εξοπλισμού προς μεγάλη απογοήτευση πχ των gamers.