ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΣΧΕΔΙΑΣΗΣ ΠΡΟΙΟΝΤΩΝ & ΣΥΣΤΗΜΑΤΩΝ

ΣΧΕΔΙΑΣΗ ΚΑΜΠΥΛΩΝ ΠΑΝΩ ΣΕ 3D ΝΕΦΗ ΣΗΜΕΙΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
ΝΙΚΟΣ ΚΟΛΑΤΣΗΣ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ
ΦΙΛΙΠΠΟΣ ΑΖΑΡΙΑΔΗΣ

Αφιερώνεται σε όσους
με στήριξαν

ΠΕΡΙΕΧΟΜΕΝΑ

1.Σκοπός της διπλωματικής εργασίας

Η παρούσα διπλωματική εργασία εκπονήθηκε στο τμήμα Μηχανικών Σχεδίασης Προϊόντων και Συστημάτων του Πανεπιστήμιου Αιγαίου που εδρεύει στη Σύρο.

Αντικείμενο της εργασίας αυτής είναι η παρουσίαση μιας νέας μεθόδου στη σχεδίαση προϊόντων, η όποια βασίζεται πάνω σε 3D νέφη σημείων χρησιμοποιώντας ως πρωταρχικό γεωμετρικό μοντέλο το σημείο. Ο πυρήνας της προτεινομένης μεθοδολογίας αποτελείται από έναν αξιόπιστο και αποδοτικό αλγόριθμο για την κατευθυνόμενη προβολή σημείου πάνω σε ένα 3D νέφος σημείων. Ο συγκεκριμένος αλγόριθμος συνδυάζεται με τεχνικές εξομάλυνσης για την κατασκευή ψηφιακών καμπυλών ελεύθερης μορφής πάνω σε μια επιφάνεια νέφους

Η προτεινόμενη μεθοδολογία επιλύει σημαντικά προβλήματα στον προκαταρτικό σχεδιασμό διαφόρων προϊόντων και επομένως ο χρήστης και κατά επέκταση το προϊόν ενός τέτοιου συστήματος γίνεται περισσότερο ανταγωνιστικό και μπορεί να προσφέρει σημαντικά οφέλη στις αντίστοιχες βιομηχανίες σε σχέση με τους παραδοσιακούς τρόπους σχεδίασης που ήδη γνωρίζουμε. Με αυτό τον τρόπο, ο σχεδιαστής κερδίζει περισσότερο χρόνο για την «μελέτη του προϊόντος» παρά για την σχεδίασή του, από τα συνολικά χρονικά περιθώρια που έχει στην διάθεσή του.

2.Βασικες Γεωμετρικές Έννοιες

2.1 Σημεία και Διανύσματα

Σημείο στο χώρο ονομάζεται μια οντότητα που έχει θέση αλλά δεν έχει διαστάσεις (μήκος, πλάτος ή ύψος). Στην Καρτεσιανή Γεωμετρία το σημείο ταυτίζεται με τις συντεταγμένες του. Έτσι σε έναν Ευκλείδειο χώρο τριών διαστάσεων το σημείο ορίζεται ως η διατεταγμένη τριάδα (x,y,z), όπου τα x,y,z είναι πραγματικοί αριθμοί. Έτσι, το σημείο αποδίδει με τις συντεταγμένες του την έννοια της θέσης χωρίς να παρέχει άλλες πληροφορίες.

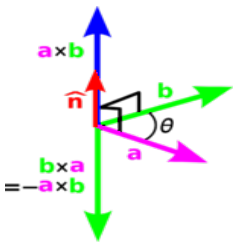
Διάνυσμα στα μαθηματικά είναι ένα ευθύγραμμο τμήμα με δεδομένο προσανατολισμό στο χώρο , στο οποίο έχουμε ορίσει το ένα άκρο του ως αρχή και το άλλο ως τέλος [41]. Στη γεωμετρία το διάνυσμα ορίζεται ως ένα προσανατολισμένο ευθύγραμμο τμήμα δηλαδή ως ένα ευθύγραμμο τμήμα του οποίου τα άκρα θεωρούνται διατεταγμένα και περιέχει τις έννοιες της διεύθυνσης και της φοράς. Ένα διάνυσμα με αρχή το Α και πέρας το Β συμβολίζεται με $\overrightarrow{AB}$ παριστάνεται με ένα βέλος που ξεκινά από το Α και καταλήγει στο Β.

2.2 Διεύθυνση Φορά και Μετρό διανύσματος

Διεύθυνση ενός διανύσματος ορίζεται η ευθεία πάνω στην όποια βρίσκεται το διάνυσμα και κάθε άλλη ευθεία προς αυτήν. Φορά ενός διανύσματος ορίζεται μια από τις δυο κατευθύνσεις που ορίζει μια ευθεία παράλληλη στο διάνυσμα.

Μέτρο ενός διανύσματος είναι ένας αριθμός που χαρακτηρίζει το διάνυσμα ή ένα βαθμωτό μέγεθος. Η διεύθυνση και η φορά (ονομάζονται κατεύθυνση) και όλα αυτά μαζί με το μέτρο χαρακτηρίζουν ένα διάνυσμα. Το μέτρο ή μήκος ενός διανύσματος $\overrightarrow{AB}$ συμβολίζεται με $|\overrightarrow{AB}|$ και ισούται με την τετραγωνική ρίζα του αθροίσματος των συνιστωσών του [41]. Έστω $\overrightarrow{AB} = a_1e_1 + a_2e_2 + \cdots + a_ne_n$, όπου $a_1, a_2, \ldots, a_n$ οι συνιστώσες του διανύσματος και $e_1, e_{2,} \ldots, e_n$ τα μοναδιαία διανύσματα της βάσης, τότε το μέτρο του διανύσματος θα ισούται με

$|\overrightarrow{AB}| = \sqrt{a_1^2 + a_2^2 + \cdots + a_n^2}$.



Σχήμα(1)

2.3 Εσωτερικό και Εξωτερικό γινόμενο

Το εσωτερικό γινόμενο δύο διανυσμάτων $\vec{a}$ και $\vec{b}$ συμβολίζεται με $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = ||\vec{\boldsymbol{a}}||\ ||\vec{\boldsymbol{b}}||\ \boldsymbol{\cos\theta}$ όπου $||\vec{\boldsymbol{a}}||$ και $||\vec{\boldsymbol{b}}||$ συμβολίζουμε τα μέτρα των διανυσμάτων $\vec{a}$ και $\vec{b}$ αντίστοιχα, με θ τη γωνία που σχηματίζεται ανάμεσα στα δύο διανύσματα και cosθ το συνημίτονο της γωνίας. Το εσωτερικό γινόμενο είναι ουσιαστικά το γινόμενο του πρώτου διανύσματος με τη προβολή του δευτέρου πάνω στο πρώτο [54]. Είναι επίσης φανερό πως το εσωτερικό γινόμενο δύο διανυσμάτων θα είναι πάντα ένας αριθμός και όχι ένα νέο διάνυσμα όπως στην πρόσθεση και την αφαίρεση διανυσμάτων. Όταν τα διανύσματα είναι κάθετα μεταξύ τους, το εσωτερικό γινόμενο είναι ίσο με το 0 (μηδενικό διάνυσμα), ενώ όταν είναι παράλληλα το εσωτερικό γινόμενο ισούται με το θετικό (ή αρνητικό αντίστοιχα) γινόμενο των μέτρων τους. Αυτό είναι φανερό γιατί $cos90^o = 0$ και $cos0^o = 1$ (επίσης $cos180^o = -1$).

Αν $\vec{a}$ και $\vec{b}$ είναι τα δύο διανύσματα, το εξωτερικό γινόμενο συμβολίζεται ως $\vec{a}\ x\ \vec{b}$ και ορίζεται ως $\vec{\boldsymbol{a}}\ \boldsymbol{x}\ \vec{\boldsymbol{b}} = ||\vec{\boldsymbol{a}}||\ ||\vec{\boldsymbol{b}}||\ \boldsymbol{\sin(\theta)n}$ που $||\vec{a}||\ και\ ||\vec{b}||$ είναι τα μέτρα των διανυσμάτων $\vec{a}\ και\ \vec{b}$, και θ είναι η γωνία μεταξύ των δύο διανυσμάτων και $\vec{n}$ είναι το μοναδιαίο διάνυσμα κάθετο στο επίπεδο στο οποίο βρίσκονται τα $\vec{a}\ και\ \vec{b}$. Γραφικά, το εξωτερικό γινόμενο αναπαρίσταται από το Σχήμα(1) πάνω δεξιά. Για παράλληλα διανύσματα το εξωτερικό γινόμενο δίνει το μηδενικό διάνυσμα, εφόσον $sin0^o = 0$ και $sin180^o = 0$ [41].

2.4 Πίνακες και Είδη πινάκων

Ονομάζουμε πίνακα Π τύπου ν x μ μία ορθογώνια διάταξη α_{ij} (i = 1...ν, j = 1...μ) με στοιχεία από ν γραμμές και μ στήλες, ώστε το στοιχείο α_{ij} να βρίσκεται ταυτόχρονα στην i-οστή γραμμή και στη j-οστή στήλη. Για παράδειγμα: Ας είναι $A=(\alpha_{ij})$ και $B=(\beta_{ij})$ δύο πίνακες ν x μ. Θα είναι ίσοι (δηλαδή A=B) αν και μόνο αν ισχύει ότι: Για κάθε είναι i =1…ν, j =1…μ, $\boldsymbol{\alpha_{ij} = \beta_{ij}}$. Ένας πίνακας 1×μ λέγεται πίνακας γραμμή ενώ ένας πίνακας ν×1 λέγεται πίνακας στήλη. Αν ένας πίνακας έχει ίδιο αριθμό γραμμών και στηλών, ονομάζεται τετραγωνικός ν-διάστασης. Ένας πίνακας που έχει όλα τα μη μηδενικά στοιχεία του στη διαγώνιό του ονομάζεται διαγώνιος. Επίσης, ένας (τετραγωνικός) πίνακας του

$$\Pi = \begin{matrix} \alpha_{11} & \alpha_{12} & \ldots & \alpha_{1\mu} \\ \alpha_{21} & \alpha_{22} & \ldots & \alpha_{2\mu} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{\nu 1} & \alpha_{\nu 2} & \ldots & \alpha_{\nu\mu} \end{matrix}$$

Σχήμα(2)

οποίου τα στοιχεία που βρίσκονται κάτω από τη διαγώνιό του είναι μηδενικά ονομάζεται άνω τριγωνικός [62]. Αντίστοιχα, αν έχει μηδενικά τα στοιχεία που βρίσκονται πάνω στην διαγώνιό του, τότε λέγεται κάτω τριγωνικός. Ένας διαγώνιος πίνακας $Π_{vxv}$ (τετραγωνικός) ο οποίος έχει όλα τα στοιχεία της διαγωνίου του ίσα με τη μονάδα και όλα τα άλλα στοιχεία του μηδενικά λέγεται μοναδιαίος και συμβολίζεται: I_v , ενώ ο πίνακας που έχει όλα του τα στοιχεία μηδενικά λέγεται μηδενικός και συμβολίζεται με Ο [41].

2.5 Παραμετρικές εξισώσεις

Μια καμπύλη είναι ουσιαστικά μια συλλογή σημείων του επίπεδου ή του τρισδιάστατου χώρου. Για να περιγράψουμε μαθηματικά μια καμπύλη πρέπει να γνωρίζουμε τις συντεταγμένες χ,y (επιπλέον και z για μη επίπεδες καμπύλες) όλων των σημείων της. Οι συντεταγμένες αυτές δίνονται από κάποιες κατάλληλες εξισώσεις οι οποίες περιγράφουν επακριβώς την καμπύλη.
Μια συνηθισμένη μορφή παράστασης καμπυλών είναι με αλγεβρικές εξισώσεις με την πιο απλή μορφή y=f(x).

Μια δεύτερη μορφή παράστασης καμπυλών είναι η παραμετρική. Στη μορφή αυτή οι συντεταγμένες κάθε σημείου της καμπύλης δίνονται ξεχωριστά με τη βοήθεια ανεξάρτητης παραμέτρου t. $\begin{cases} x = x(t) \\ y = y(t) \\ z = z(t) \end{cases}$ η παραμετρος t μπορεί να παίρνει όλες τις πραγματικές τιμές tε(-∞,+∞), στην περίπτωση άπειρων καμπυλών όπως οι ευθείες, ή να είναι ορισμένη μέσα σε ένα παραμετρικό διάστημα της μορφής t(a,b), όποτε η καμπύλη έχει αρχή και τέλος.
Για κάθε συγκεκριμένη τιμή του t μέσα στο πεδίο ορισμού της παραμέτρου η παραπάνω εξίσωση δίνει ένα σημείο της καμπύλης, το P(t)=$\begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix}$. Στην παράμετρο t μπορούμε να δώσουμε την εξής φυσική σημασία: αν κινούμαστε πάνω στην καμπύλη με σταθερή ταχύτητα, τότε την χρονική στιγμή t θα βρισκόμαστε ακριβώς στο σημείο P(t) της καμπύλης.

Αν για παράδειγμα μια ευθεία διέρχεται από δυο σημεία P1($x1, y1$) και P2($x2, y2$) και παρατηρούμε ακόμη ότι για τα σημεία που ορίζουν την ευθεία ισχύει P(0)=P1 και P(1)=P2, τότε η εξίσωση της ευθείας θα έχει P(t)=(1- t) P1 + t P2 t ϵ [0,1]. Αυτή είναι η παραμετρική εξίσωση του προσανατολισμένου ευθύγραμμου τμήματος με άκρα τα σημεία P1 και P2. Για t<0 κινούμαστε στην ευθεία πριν το P1, ενώ για t>1 κινούμαστε περά από το P2.

2.6 Νέφος σημείων (Point Cloud)

Η χρήση της τρισδιάστατης σάρωσης αποτελεί μια εναλλακτική μεθοδολογία και ένα σημαντικό εργαλείο στις αποτυπώσεις των αντικειμένων. Η σύγχρονη αυτή μέθοδος δίνει την δυνατότητα παραγωγής τρισδιάστατων ψηφιακών μοντέλων, παρέχοντας υψηλή ακρίβεια, ταχύτητα, ευκολία στη μοντελοποίηση και μείωση του κόστους παραγωγής συγκριτικά με τις κλασσικές μεθόδους αποτύπωσης. Το σύστημα σάρωσης προσδιορίζει την τρισδιάστατη γεωμετρία των φυσικών επιφανειών η οποία απεικονίζεται σε μορφή πυκνών σημείων (point cloud). Κατά τη διάρκεια της σάρωσης χιλιάδες μεμονωμένες τρισδιάστατες μετρήσεις εμφανίζονται δημιουργώντας σε

πραγματικό χρόνο μια τρισδιάστατη απεικόνιση της λήψης. Από τις πολικές συντεταγμένες υπολογίζονται στη συνέχεια αυτόματα οι καρτεσιανές συντεταγμένες (x, y, z) των σημείων του προς αποτύπωση αντικειμένου στο τρισορθογώνιο σύστημα του σαρωτή. Έτσι, σε κάθε σημείο αντιστοιχεί η πληροφορία (x, y, z,), με συνέπεια να ορίζεται ακριβώς η θέση του σημείου. Η διαδικασία αυτή επιτελείται από 3D scanner το οποίο είναι μια συσκευή που αναλύει ένα πραγματικό αντικείμενο για να συλλέξει από αυτό στοιχεία που αφορούν τη μορφή του αντικειμένου. Τα στοιχεία αυτά που συλλέγονται δηλαδή το νέφος σημείων (point cloud), μπορούν έπειτα να χρησιμοποιηθούν για την κατασκευή των τρισδιάστατων πρωτοτύπων τα οποία είναι χρήσιμα για μια ευρεία ποικιλία εφαρμογών.

Υπάρχουν πολλές και διαφορετικές τεχνολογίες που χρησιμοποιούνται για την αντιγραφή των προϊόντων, όμως θα πρέπει να σημειωθεί ότι η κάθε μια τεχνολογία έχει κάποιους περιορισμούς. Ένας από αυτούς τους περιορισμούς μπορεί να είναι η λάμψη ή η διαφάνεια ενός αντικειμένου.

Σε αυτό το σημείο θα ήταν σωστό να διευκρινίσουμε ότι υπάρχουν δύο τύποι τρισδιάστατων ανιχνευτών, αυτοί που βασίζονται στην επαφή και αυτοί που λειτουργούν από απόσταση (μη-επαφή). Οι τρισδιάστατοι ανιχνευτές 3D scanner μη-επαφής μπορούν να διαιρεθούν περαιτέρω σε δύο κύριες κατηγορίες, σε ενεργούς ανιχνευτές και ενεργητικούς ανιχνευτές. Υπάρχουν ποικίλες τεχνολογίες που εμπίπτουν σε κάθε μια από αυτές τις κατηγορίες.

Η απόδοση της τρισδιάστατης γεωμετρίας με νέφος σημείων (point cloud) και όχι με μεμονωμένα σημεία αποτύπωσης, αποδίδει όλες τις λεπτομέρειες της επιφάνειας που σαρώνεται. Με τη χρήση της τρισδιάστατης σάρωσης μπορεί κανείς να αποτυπώσει περισσότερα των 40 εκατομμυρίων σημείων μέσα σε μια ημέρα, κάτι αδιανόητο για τις κλασσικές μεθόδους μηχανικής. Το νέφος σημείων ενός αντικειμένου που έχει σαρωθεί αποτελεί δεδομένα έτοιμα για μετρήσεις και επεξεργασία. Μερικές από τις εφαρμογές που μπορεί να έχει ένα 3Δ νέφος σημείων (3D point cloud) είναι.

- Στις κατασκευές και στην βιομηχανία **(**Construction industry and industrial engineering)
- Στην εικονική ψηφιακή βιομηχανία (Entertainment industry)
- Στην Αντίστροφη Μηχανική (Reverse engineering)
- Στην Πολιτισμική κληρονομιά (Cultural Heritage)
- Στο Οδοντικό CAD/CAM (Dental CAD/CAM)
- Εξασφάλιση ποιότητας/βιομηχανική μετρολογία (Quality Assurance / Industrial Metrology)

Σχήμα(3) 3D Scanner FARO μη επαφής

Σχήμα(4) 3D Scanner FARO επαφής

3.Βασικά Λογισμικά Εργαλεία

3.1 Τι είναι Αλγόριθμος

Ως αλγόριθμος ορίζεται μια πεπερασμένη σειρά ενεργειών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος. Οι αλγόριθμοι είναι σημαντικοί γιατί σχετίζονται άμεσα με τον τρόπο τον οποίο οι υπολογιστές επεξεργάζονται πληροφορίες. Ένα πρόγραμμα αποτελείται από αλγόριθμο που καθορίζει ποια συγκεκριμένα βήματα πρέπει να εκτελεστούν και (σε ποια συγκεκριμένη σειρά) προκειμένου να επιτευχθεί ένας συγκεκριμένος στόχος. Κατά συνέπεια, ένας αλγόριθμος μπορεί να θεωρηθεί οποιαδήποτε ακολουθία εντολών που μπορεί να εκτελεσθεί από ένα πλήρες σύστημα. Για οποιαδήποτε τέτοια υπολογιστική διαδικασία, ο αλγόριθμος πρέπει να οριστεί αυστηρά: να είναι ορισμένος για όλες τις πιθανές περιστάσεις που θα μπορούσαν να προκύψουν. Δηλαδή οποιαδήποτε υπό όρους βήματα πρέπει να εξεταστούν συστηματικά, και σε κάθε περίπτωση τα κριτήρια πρέπει να είναι σαφή (και υπολογίσιμα).

Επειδή ένας αλγόριθμος είναι ένας ακριβής κατάλογος βημάτων ακριβείας, η σειρά του υπολογισμού θα είναι σχεδόν πάντα κρίσιμη για τη λειτουργία του αλγόριθμου. Οι εντολές συνήθως απαριθμούνται ρητά, και περιγράφονται σαν να ξεκινούν "από την κορυφή" και πηγαίνοντας "προς στο κατώτατο σημείο", μια ιδέα που περιγράφεται τυπικά με τον όρο της "ροής ελέγχου".

3.2 C++

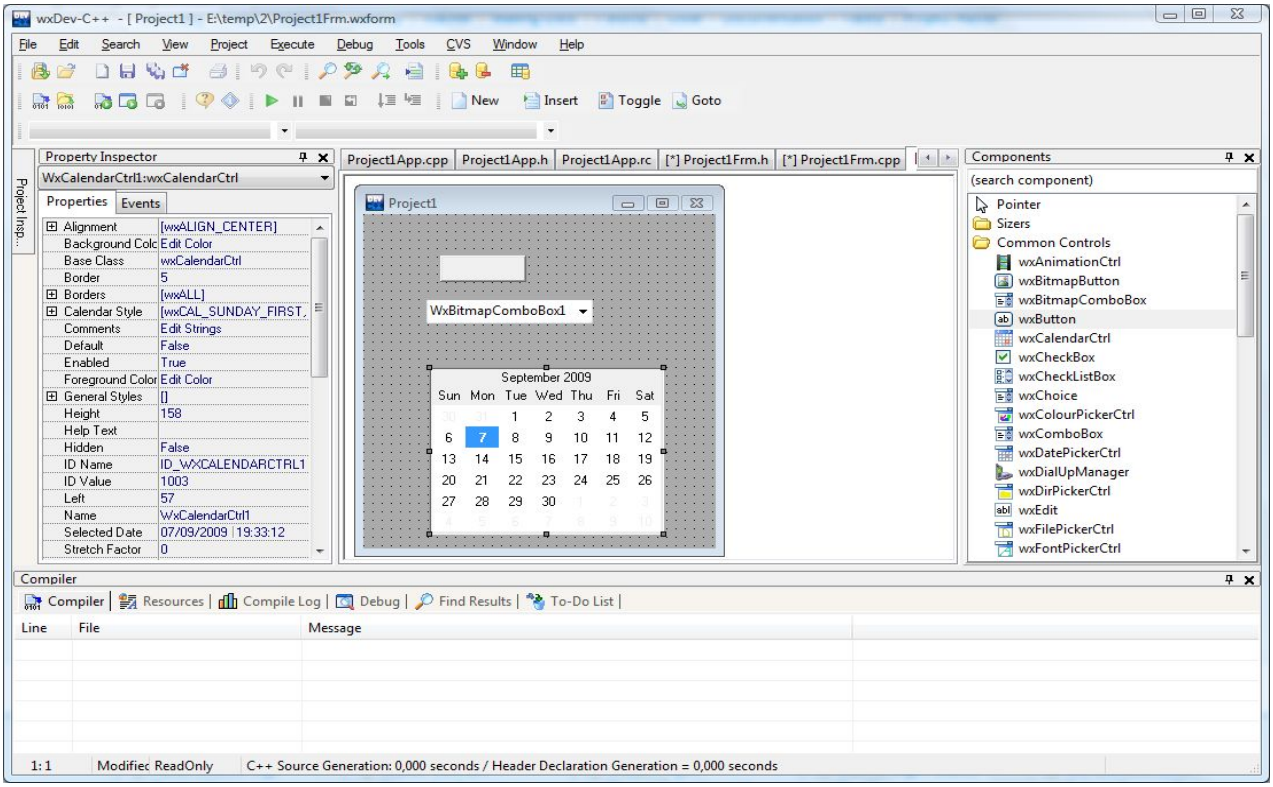
Σήμερα οι υπολογιστές είναι σε θέση να εκτελέσουν πολλές διαφορετικές λειτουργίες, από τις απλές μαθηματικές πράξεις, έως τις πιο περίπλοκες προσομοιώσεις. Αλλά ο υπολογιστής δεν μπορεί από μονός του να ενεργεί, θα πρέπει να εκτελεί μια σειρά από προκαθορισμένες ενέργειες που φτιάχνουν τον κώδικα και στο σύνολο του, την γλωσσά προγραμματισμού. Έτσι μια γλώσσα προγραμματισμού είναι ένα σύνολο οδηγιών και μια σειρά λεξικών συμβάσεων σχεδιασμένη ώστε να δίνει εντολή στον υπολογιστή τι πρέπει να κάνει. Στη συγκεκριμένη διπλωματική θα δουλέψουμε με C++ για τους παρακάτω λογούς.

- Αντικειμενοστραφής προγραμματισμός: Η δυνατότητα να προσανατολιστεί ο προγραμματισμός στα αντικείμενα επιτρέπει στον προγραμματιστή να σχεδιάσει εφαρμογές περισσότερο σαν μια επικοινωνία μεταξύ των αντικειμένων παρά ως μια δομημένη ακολουθία κώδικα. Επιπλέον, επιτρέπει μια καλύτερη επαναχρησιμοποίηση του κώδικα με έναν λογικότερο και παραγωγικότερο τρόπο.
- Ορατότητα: Μπορούμε να τρέξουμε τον ίδιο C++ κώδικα σχεδόν σε οποιοδήποτε τύπο υπολογιστή και λειτουργικού συστήματος χωρίς παραγωγή οποιασδήποτε αλλαγής, γιατί απλά η C++ είναι η πιο χρησιμοποιημένη γλώσσα προγραμματισμού στον κόσμο.
- Συντομία: Ο κώδικας που γράφεται σε C++ είναι πολύ σύντομος σε σύγκριση με άλλες γλώσσες, δεδομένου ότι η χρήση των ειδικών χαρακτήρων προτιμάται από λέξεις κλειδιά.
- Μορφωτικός προγραμματισμός: Μια εφαρμογή C++ μπορεί να αποτελείται από διάφορα κομμάτια κώδικα που συντάσσονται χωριστά και συνδέονται μετέπειτα. Θέλοντας να ελαττώσουμε τον χρόνο δεδομένου ότι δεν είναι απαραίτητο η ανασύνταξη (recompile) σε όλη την εφαρμογή αλλά μόνο στο αρχείο που περιέχει την αλλαγή. Αυτό το χαρακτηριστικό επιτρέπει τον συνδυασμό της C++ με κώδικα που παράγεται σε άλλες γλώσσες, όπως η Assembler ή C.
- Συμβατότητα της C: Η C++ είναι συμβατή με την C. Οτιδήποτε γράφεται σε C μπορεί εύκολα να περιληφθεί σε C++ χωρίς να παρατηρηθεί οποιαδήποτε αλλαγή.
- Ταχύτητα: Ο προκύπτων κώδικας από την C++ είναι πολύ αποδοτικός, λόγω της διπλής ιδιότητας που έχει ως υψηλού επιπέδου (high-level) και χαμηλού επιπέδου (low-level) γλώσσας καθώς και στο μειωμένο μέγεθος της ίδιας της γλώσσας [43].

3.3 Ανάπτυξη σε WxDEV

Στην παρούσα διπλωματική θα αναπτυχτεί μια νέα πειραματική εφαρμογή, η οποία θα συμβάλλει σε ένα νέο low level σχεδιασμό προϊόντων χρησιμοποιώντας ως βάση νέφη σημείων.

Η όλη εφαρμογή θα αναπτυχτεί στο WxDEV ένα free integrated development environment (IDE) [62] το όποιο είναι ένα πρόγραμμα που όπως αναφέρει υποστηρίζει την C++ και την OpenGL και ενδείκνυται για software application και software development. Ένα IDE συνήθως περιλαμβάνει, source code editor, compiler και interpreter, build automation εργαλεία και debugger. Όλα αυτά συμβάλουν στην απλοποίηση της σύνθεσης του GUI (Graphical User Interface). Πολλά μοντέρνα IDE όπως και το WxDEV [69] έχουν class browser, object inspector όπως και class hierarchy diagram για να συντελέσουν στην γρήγορη και σωστή ανάπτυξη του software.



Σχήμα(5) Περιβάλλον Wxdev

3.4 Τι είναι OpenGL

Η OpenGL είναι μια βιβλιοθήκη γραφικών η οποία αποτελείται από περίπου 150 διακριτές εντολές του υλικού γραφικών (graphics hardware) μέσω των οποίων καθορίζονται τα αντικείμενα και οι πράξεις που χρειάζονται για την δημιουργία τρισδιάστατων διαδραστικών γραφικών εφαρμογών. Η OpenGL σχεδιάστηκε ως ένα ανεξάρτητο υλικό μέσω αλληλεπίδρασης με υλοποιήσεις σε διαφορετικές πλατφόρμες υλικού. Για την επίτευξη αυτού του στόχου αποφεύχθηκε η ενσωμάτωση εντολών που αφορούν παραθυρικές εργασίες ή εντολές για την δημιουργία πυλών εξόδου γραφικών (graphics display ports) ή εντολές που αφορούν την είσοδο δεδομένων από τον χρήστη. Αντί αυτού ο κάθε χρήστης της OpenGL θα πρέπει να δουλέψει με τα εργαλεία που του παρέχει το λειτουργικό σύστημα για τον έλεγχο των παραθυρικών λειτουργιών. Η OpenGL δεν περιλαμβάνει υψηλού επιπέδου εντολές για τη σχεδίαση αντικειμένων αλλά δίνει τη δυνατότητα στο χρήστη να χτίσει το μοντέλο του χρησιμοποιώντας στοιχειώδη γεωμετρικά πρότυπα όπως σημεία, γραμμές, πολύγωνα, κοκ. Η συνολική υπολογιστική διαδικασία ξεκινά από τον ορισμό της στοιχειώδους μαθηματικής περιγραφής μιας σκηνής μέχρι την εμφάνισή της στην οθόνη. Τα μοντέλα ή αντικείμενα που αποτελούν μια σκηνή συντίθενται από στοιχειώδη πρότυπα σημεία, γραμμές και πολύγωνα τα οποία καθορίζονται από τις κορυφές τους (vertices). Η τελική εικόνα (rendered image) αποτελείται από pixels τα οποία σχεδιάζονται στην οθόνη. Το pixel είναι η μικρότερη ορατή διακριτή ποσότητα που το υλικό γραφικών μπορεί να τοποθετήσει στην οθόνη [44].

Επιπλέον, η OpenGL διατάσει τα αντικείμενα στον τρισδιάστατο χώρο και επιλέγει το επιθυμητό σημείο για την δημιουργία απόψεων σύνθετων σκηνών. Υπολογίζει το χρώμα κάθε αντικειμένου. Το χρώμα μπορεί να έχει ανατεθεί ρητά από την εφαρμογή, να έχει καθοριστεί από συγκεκριμένες συνθήκες φωτισμού, να έχει αποκτηθεί από την επικόλληση υφής σε ένα αντικείμενο

ή να έχει προέλθει από ένα σύνθετο συνδυασμό πολλών διαφορετικών παραμέτρων. Η OpenGL μετατρέπει την μαθηματική περιγραφή και τη συνδεόμενη χρωματική πληροφορία των αντικειμένων σε pixels στην οθόνη. Αυτή η διαδικασία ονομάζεται ψηφιοποίηση (rasterization)[42].

4. Graphical User Interface (GUI)

4.1 Target group

Το κοινό στο οποίο θα απευθύνεται η εφαρμογή που θα υλοποιήσουμε θα είναι κυρίως εταιρίες και άτομα με επαρκείς γνώσεις Η/Υ και κυρίως με γνώσεις σε 3D σχεδιαστικά προγράμματα ώστε να έχουν τη δυνατότητα να κατανοήσουν αλλά και να εφαρμόζουν παράλληλα αυτά που πρόκειται να υλοποιήσουμε με τη σχεδίαση σε νέφη σημείων. Ταυτόχρονα όμως δεν παύει να είναι μια εφαρμογή που στην τελική της μορφή θα είναι απλή προσιτή και εύχρηστη σε χρηστές με λιγότερη εμπειρία σε παρόμοια σχεδιαστικά περιβάλλοντα. Αυτό έχει σα συνέπεια το εύρος χρήσης της εφαρμογής να κυμαίνεται σένα μικρό και ειδικευμένο target group κυρίως άτομα που ασχολούνται με το 3D modeling και φυσικά εταιρίες που εδρεύουν και ειδικεύονται στην τεχνολογία αυτή. Θα μπορούσε όμως να χρησιμοποιηθεί και από οποιοδήποτε άλλο άτομο και επιχείρηση. Η εφαρμογή θα υλοποιηθεί σε αγγλική γλώσσα παραμένοντας έτσι στα πλαίσια των περισσότερων σχεδιαστικών προγραμμάτων.

4.2 Παράγοντες Διεπαφής

Ένας παράγοντας που έχει ιδιαίτερη σημασία για την επιτυχία μιας εφαρμογής είναι η σχεδίαση της διεπαφής χρήστη. Όσο ενδιαφέρον και αν είναι αυτό που σκοπεύουμε να υλοποιήσουμε εάν ο τρόπος που θα παρουσιαστεί στον χρήστη δεν είναι ελκυστικός και λειτουργικός τότε η εφαρμογή δεν πρόκειται να αφήσει καλές εντυπώσεις στον χρήστη με αποτέλεσμα να τον δυσκολέψει στην λειτουργιά. Το στήσιμο της διεπαφής αναλύεται σε δυο βασικά θέματα. Α) την εμφάνιση της εφαρμογής. Β) την λειτουργικότητα που θα πρέπει να παρέχει στον χρήστη [46][65]. Η προσπάθεια για όσο τον δυνατόν καλύτερη σχεδίαση τόσο της εμφάνισης όσο και της λειτουργιάς πρέπει να κινηθούν σε δυο βασικούς άξονες. Την αισθητική και την εργονομία.

4.3 Βασικές αρχές σχεδίασης διεπαφής.

Η διεπαφή θα πρέπει να είναι σχεδιασμένη κατά τέτοιο τρόπο ώστε να ελκύει το ενδιαφέρον του χρήστη, γεγονός που αποτελεί και την ουσία της εφαρμογής. Θα μπορούσαμε να ισχυριστούμε ότι η διεπαφή αποτελεί το περιτύλιγμα που θα κάνει την εφαρμογή πιο ελκυστική στον χρήστη.
Ωστόσο, δεν πρέπει να αποσπά την προσοχή του χρήστη από το περιεχόμενο της. Ο τρόπος σχεδίασης της διεπαφής θα πρέπει να είναι όσο γίνεται πιο φυσικός για να εξοικειωθεί με το περιβάλλον ο χρήστης, χωρίς να απαιτείται να μάθει κάποιες επιπλέον λειτουργιές

Αυτό είναι ιδιαίτερα δύσκολο, όταν πρέπει να συνδυαστεί με στοιχειά πρωτοτυπίας που θα αναδείξουν κάτι τελείως διαφορετικό στα υπάρχοντα CAD συστήματα. Πιο συγκεκριμένα, σε συνδυασμό με τις άλλες εφαρμογές με τα όποια ο χρήστης είναι ήδη εξοικειωμένος συνειδητοποιούμε ποσό δύσκολο είναι να δημιουργήσει κάνεις την τέλεια διεπαφή.
Τι θα πρέπει να περιμένει ο χρήστης από μια νέα CAD εφαρμογή.

- να είναι εύκολη στη χρήση
- να μην αλλάζει τον τρόπο που έχει συνηθίσει να χρησιμοποιεί τον υπολογιστή
- να του παρέχει ικανό βαθμό λειτουργικότητας
- να παρουσιάζει αυτό το όποιο αναμένει με το σωστό τρόπο και στο σωστό σημείο
- να είναι αποδοτική
- να διατηρεί το ενδιαφέρον του
- να έχει όσο γίνεται κοινή φιλοσοφία με αλλά CAD συστήματα
- να του επιτρέπει να παρεμβαίνει ο ίδιος και να επιλεγεί πως θα δει την πληροφορία

Δεν είναι πάντοτε εύκολο να ικανοποιηθούν όλες οι παραπάνω προσδοκίες του χρήστη. Ορισμένες από αυτές μπορεί να είναι αλληλοσυγκρουόμενες, ενώ άλλες μπορεί να είναι εκ φύσεως δύσκολο να εκπληρωθούν. Υπάρχουν όμως κάποιοι γενικά παραδεκτοί κανόνες που βοηθούν στη σχεδίαση μιας λειτουργικά και αισθητικά ορθής διεπαφής.
Αυτά είναι, η διάταξη οθόνης που είναι μια αισθητικά και εργονομικά επιτυχημένη σύνθεση οθόνης και πρέπει να χαρακτηρίζεται από τους ακόλουθους παράγοντες:
Ισορροπία, Αναλογία, Απλότητα, Παράταξη των αντικειμένων, Συνοχή Ενότητα

- Ισορροπία*:* Αναφέρεται στην οπτική κατανομή των στοιχείων που συνιστούν μια εικόνα πάνω στην οθόνη και ανάλογα με τη θέση κάθε αντικειμένου στην οθόνη και το οπτικό του βάρος μπορεί να χαρακτηρίζεται από κάποιο βαθμό ισορροπίας. Το οπτικό βάρος αναφέρεται στην εντύπωση που προκαλεί κάθε στοιχείο. Κάποια στοιχεία τραβούν το βλέμμα του χρήστη περισσότερο από κάποια άλλα και αυτό έχει να κάνει με μέγεθος, σχήμα, χρώμα, φωτεινότητα, αντίθεση με το φόντο.
- Αναλογία: Η έννοια της αναλογίας σχετίζεται με τη θέση ενός αντικειμένου σε σχέση με τα άκρα της οθόνης και με τις αποστάσεις των αντικειμένων μεταξύ τους.
- Απλότητα: Όσο πιο απλό τόσο πιο σωστό. Η απλότητα περιλαμβάνει έννοιες όπως το χρώμα, το μέγεθος και η απόσταση. Ιδιαίτερα στο χρώμα θα πρέπει να σταθούμε – παρατηρώντας ότι το χρώμα παίζει σημαντικό ρόλο σε μια εφαρμογή καθώς επηρεάζει την αισθητική διάσταση και εργονομία, τη λειτουργικότητα, των τονισμό των εννοιών και μηνυμάτων για τη μετάδοση πληροφοριών. Είναι κοινώς αποδεκτό πως συγκεκριμένα χρώματα ή συνδυασμοί χρησιμοποιούνται στην καθημερινή ζωή για μετάδοση συγκεκριμένων μηνυμάτων
- Παράταξη αντικειμένων: Τα οπτικά στοιχεία μπορούν να παραταχθούν στην οθόνη έτσι ώστε να καθοδηγήσουν το μάτι του χρήστη σε καθορισμένη πορεία ώστε να δώσουν έμφαση σε σημαντική πληροφορία.
- Συνοχή-Ενότητα: Τα οπτικά στοιχεία πρέπει να δίνουν την αίσθηση ότι αποτελούν μέρος ενός συνόλου, το οποίο εξασφαλίζει συνέπεια, δίνει εικόνα προσεγμένης σχεδίασης και απαιτεί υιοθέτηση γενικών αρχών (π.χ. κοινό φόντο, σταθερή γραμματοσειρά, κοινό ύφος)

4.4 Αρχές Παρουσίασης Περιεχομένου

Κατά την ανάπτυξη μιας εφαρμογής, υπάρχει συνήθως διαθέσιμος μεγάλος όγκος υλικού. Ο σχεδιαστής μπορεί να μπει στον πειρασμό να χρησιμοποιήσει όλο το υλικό και να προσπαθήσει να ενσωματώσει όσο το δυνατόν περισσότερα στοιχεία. Η χρήση του περιεχομένου πρέπει να ακολουθεί κάποιους κανόνες, ώστε να εξυπηρετηθούν τα μηνύματα που θέλει να μεταδώσει η εφαρμογή [66].

4.4.1 Κανόνες αποδοτικής χρήσης περιεχομένου

Κατά την υλοποίηση της εφαρμογής θα πρέπει να ληφθούν υπόψη κάποιοι κανόνες περιεχομένου, που συναντάμε και εφαρμόζουμε πολύ συχνά όταν υπάρχει αλληλεπίδραση ανθρώπου μηχανής.

- Μερικά από αυτά είναι να μην προβάλλονται υπερβολικές πληροφορίες στην ίδια οθόνη. αλλά να παρουσιάσουμε έννοιες στο χρήστη που παραπέμπουν σε καταστάσεις με τις οποίες είναι ήδη εξοικειωμένος από την καθημερινή ζωή. Επιπλέον, επιλέγουμε ενιαίο ύφος για τη διεπαφή το οποίο θα πρέπει να σχετίζεται με το θέμα και διατηρούμε το ύφος της αυτό σταθερό και στη συνέχεια παρουσιάζουμε τις έννοιες με σαφήνεια και συντομία.
- Στην εφαρμογή αυτή έχει δοθεί ιδιαίτερη σημασία σε θέματα εργονομίας, ώστε να γίνει πιο απλός ο τρόπος χρήσης της εφαρμογής. Οπότε ο χρήστης δεν αναλώνεται σε εκμάθηση λειτουργιών χειριστηρίων, ούτε χρειάζεται να απομνημονεύει διαδικασίες. Το μόνο που χρειάζεται είναι να επικεντρώνει την προσοχή του στις έννοιες τις εφαρμογής. Για αυτό όπως θα δούμε παρακάτω στην συνολική εικόνα της εφαρμογής που θα παρουσιαστεί, η κάθε λειτουργία και χειρισμός της έχει τοποθετηθεί με βάση την λειτουργικότητα του σε σχέση με άλλες παρόμοιες ενέργειες. Η εξασφάλιση ικανοποιητικού βαθμού εργονομίας απαιτεί την τήρηση ορισμένων στοιχειωδών αρχών.
- Συνέπεια στη μορφή των χειριστηρίων: Είναι σημαντικό να μη μεταβάλλεται η μορφή χειριστηρίων μεταξύ τμημάτων της εφαρμογής γιατί δημιουργεί σύγχυση στο χρήστη. Επίσης χρειάζεται η διατήρηση ενιαίου ύφους στη θέση των χειριστηρίων ενώ είναι σημαντικό να διατηρείται σταθερή η θέση τους στην εφαρμογή, διαφορετικά, αναγκάζεται ο χρήστης να τα αναζητήσει. Ακόμα και αν ένα χειριστήριο δεν χρειάζεται απαραίτητα σε μια οθόνη, δεν συνιστάται η αντικατάστασή του από ένα άλλο. Είναι προτιμότερη η απενεργοποίησή του και η ενημέρωση του χρήστη ότι η συγκεκριμένη λειτουργία δεν είναι διαθέσιμη στον τρόπο λειτουργίας.
- Ανάδραση: Η ανάδραση είναι επίσης ένα σημαντικό στοιχειό της εφαρμογής αυτής, η απόκριση δηλαδή εκ μέρους της εφαρμογής στις ενέργειες του χρήστη, π.χ. κίνηση ποντικιού, πάτημα πλήκτρου. Είναι σημαντικό να υπάρχει ανάδραση γιατί επιβεβαιώνει ότι οι εντολές του χρήστη έχουν ληφθεί από το σύστημα το όποιο θα ενημερώνει για το αποτέλεσμά τους. Θα πρέπει φυσικά να υπάρχει κάποιος περιορισμός ως προς την ανάδραση, γιατί μπορεί πολλές φορές να λειτουργήσει και αρνητικά.
- Πληροφόρηση: Η πληροφόρηση έχει να κάνει με τη λειτουργία του κάθε στοιχείου της εφαρμογής. Μπορεί να είναι σταθερή με τη μορφή λεζάντας, όμως καταναλώνεται αρκετός από το διαθέσιμο χώρο άρα δεν ενδείκνυται για μια σχεδιαστική εφαρμογή. Μια άλλη λύση είναι να εμφανίζεται λεζάντα μόλις ο δείκτης του ποντικιού περάσει πάνω από το εικονίδιο. Αυτή η λειτουργιά θα δούμε ότι χρησιμοποιείται στην εφαρμογή (rollover effect) και δίνει πληροφορία σχετικά με ότι πρόκειται να τρέξει. Η ετικέτα αυτή θα εμφανίζεται δίπλα στο εικονίδιο αν ο δείκτης μείνει στην περιοχή για κάποιο χρονικό διάστημα (tooltip)
- Οργάνωση και Ομαδοποίηση: Η οργάνωση των χειριστηρίων πρέπει να γίνεται με όσο το δυνατόν πιο φυσικό και λογικό τρόπο. Εξαρτάται άμεσα και από τις λειτουργίες που αυτά

καλούνται να υποστηρίξουν για αυτό έχει σχέση η τοποθέτηση των χειριστηρίων στην οθόνη και στην ομαδοποίησή τους, πράγμα που όπως θα δούμε να υλοποιείται μέσα από τα ToolBars Η ομαδοποίηση αυτή όπως θα δούμε έχει επιτευχθεί μέσα από την συγκέντρωση των Buttons σε συγκεκριμένο σημείο της οθόνης καθώς και με την χρήση των διαχωριστικών γραμμών (WxSeparators) για να χωρίζονται οι ομάδες αυτές και με την ευθυγράμμιση ομοίων αντικειμένων ως προς τον ίδιο άξονα.

- Τήρηση συμβάσεων: Συνήθως για πρωτοτυπία μια εφαρμογής υποκρύπτουμε την λειτουργία του συστήματος, δηλαδή. παράκαμψη των χειριστηρίων του. Αυτό εν μέρη έχει αποδειχτεί επιτυχημένη πρακτική θα πρέπει όμως να δώσουμε έμφαση στο να μην αλλάξει ο τρόπος λειτουργιάς του συστήματος. Άρα το ποιο σωστό είναι να αλλάζει μόνο η εμφάνιση και όχι ο τρόπος ενέργεια των χειριστηρίων, πράγμα υλοποιείτε εν μέρη στην συγκεκριμένη εφαρμογή.
- Εύκολη πρόσβαση: θα πρέπει σε κάθε εφαρμογή να υπάρχει εύκολη και γρήγορη πρόσβαση στις λειτουργίες της. Ο χρήστης δε θα πρέπει να είναι αναγκασμένος να εκτελέσει πολύπλοκες διαδικασίες για να φτάσει στο αποτέλεσμα. Έτσι για σημαντικά κομβικά σημεία, η πρόσβαση πρέπει να γίνεται με μια μόνο ενέργεια, π.χ. μόνο με ένα πάτημα του πλήκτρου του ποντικιού. Κάθε άλλη λειτουργία θα πρέπει να είναι προσβάσιμη με τρεις ενέργειες το μέγιστο. Για παράδειγμα το πλήκτρο «Projected» θα πρέπει να είναι άμεσα προσβάσιμο, ενώ ο συντελεστής καμπυλότητας «γ», που δεν χρησιμοποιείται τόσο συχνά θα μπορούσε να είναι προσβάσιμο με παραπάνω από μία ενέργειες.
- Παροχή βοήθειας: Θα πρέπει να τονιστεί σε αυτό το σημείο ότι είναι απαραίτητη σε κάθε εφαρμογή να υπάρχει βοήθεια στο χρήστη. Η παροχή βοήθειας μπορεί να γίνει με διάφορους τρόπους, όπως με αρχεία βοήθειας,-κάτι το οποίο θα ισχύσει και στην εφαρμογή που θα υλοποιηθεί στην παρούσα διπλωματική-, αλλά πολλές φορές γίνεται και με χρήση βοηθητικών ετικετών (tool tips), που εμφανίζονται μόλις ο δείκτης του ποντικιού βρεθεί πάνω από κάποιο χειριστήριο. Επιπλέον, τα επεξηγηματικά κείμενα ή σχόλια ή και ηχητικά αποσπάσματα όπως επίσης διάφορα αποσπάσματα βίντεο, που προσομοιώνουν τις ενέργειες που πρέπει να ακολουθήσει ο χρήστης θα μπορούσαν να αποτελούν βοηθητικές λειτουργίες [46].

5. Σχεδίαση της Διεπαφής με τον Χρήστη

Η πειραματική αυτή εφαρμογή που θα υλοποιηθεί στην παρούσα διπλωματική όπως έχουμε αναφέρει προσφέρει ένα low level σχεδιασμό πάνω σε νέφη σημείων που όμως πάνω σε αυτούς τους άξονες μπορεί να αναπτυχθεί μια τελείως διαφορετική προσέγγιση στον όρο σχεδιασμός αντικειμένων με τη χρήση 3Δ βασικών επιφανειών, καθώς επίσης και τα εργαλεία που θα αναπτύξουμε για την υλοποίηση της εφαρμογής σχετίζονται και αυτά με τη 3Δ σχεδίαση. Φυσικά για να στηθεί μια νέα εφαρμογή σωστά θα πρέπει να τηρηθούν μια σειρά από κανόνες και συμβάσεις που αφορούν την εργονομία, την λειτουργία και την αλληλεπίδραση μεταξύ χρηστή και υπολογιστή. Όλοι αυτοί οι παράγοντες που έχουμε αναφέρει μέχρι τώρα καθώς και το software που θα τα υποστηρίξει πρέπει να συμπεριληφθούν, να συμψηφιστούν και να συσχετιστούν για να δώσουν μια άρτια και αποτελεσματική σχεδιαστική εφαρμογή.

5.1 Αναλυτική κατηγοριοποίηση εργαλείων και περιεχομένου

Η αναλυτική κατηγοριοποίηση των εργαλείων και του περιεχομένου προσδιορίζει τις βασικές κατηγόριες πληροφορίας που πρέπει να συμπεριλάβει η εφαρμογή. Η οργάνωση των εργαλείων σε κατηγόριες εξαρτάται από διάφορους παράγοντες όπως είναι οι στόχοι της εφαρμογής, ο λειτουργικός προσανατολισμός και αλλά. Υπάρχουν διάφορες προσεγγίσεις για την κατηγοριοποίηση. Σε αύτη την εφαρμογή χρησιμοποιούμε την αναγωγική μέθοδο, με την οποία ξεκινάει κανείς από ασαφείς γενικές έννοιες όπως εμείς από ένα νέφος σημείων και εξειδικεύει σταδιακά μέχρι το τελικό αποτέλεσμα και την χρήση κάποιου εργαλείου.

Έτσι ενώ αρχικά ξεκινάμε την κατηγοριοποίηση των σημείων με την μορφή x, y, z συντεταγμένων στη συνεχεία αυτή εξειδικεύεται περισσότερο και περνάει σε ένα άλλο επίπεδο που θα είναι η σχεδίαση χρησιμοποιώντας αυτά τα σημεία και στη συνεχεία σε ένα ανώτερο επίπεδο που θα είναι η δημιουργία σχεδιαστικών εργαλείων και διεπαφής με τον χρήστη-σχεδιαστή. Τα ενδεικτικά αυτά σχεδιαστικά εργαλεία που θα δημιουργηθούν είναι αρκετά. Επιγραμματικά αναφέρονται κάποια όπως το Move, Zoom, Rotate, εύρεση τόμων και Trim. Πολλά βέβαια από αυτά τα εργαλεία είναι ήδη γνωστά αλλά πάντα πρέπει να τηρηθούν αρχές σχεδίασης. Όλα αυτά θα συντελέσουν στην άνετη και σωστή σχεδίαση. Βέβαια σε αυτό το σημείο θα πρέπει να σημειωθεί ότι δεν θα ήταν σωστό να υπάρχουν θέματα και στοιχειά που επαναλαμβάνονται μέσα σε πολλές κατηγόριες μιας και τότε δεν θα έχουμε πετύχει τους στόχους και ο χρηστής-σχεδιαστής θα χαθεί μέσα στα μη σωστά ταξινομημένα εργαλεία-εντολές πληροφορίες, όπως επίσης αν υπάρχουν επικαλυπτόμενες πληροφορίες και κρυμμένες κατηγοριοποιήσεις για τις εντολές

5.2 Περιγραφή του τρόπου διεπαφής εργαλείων με το χρήστη

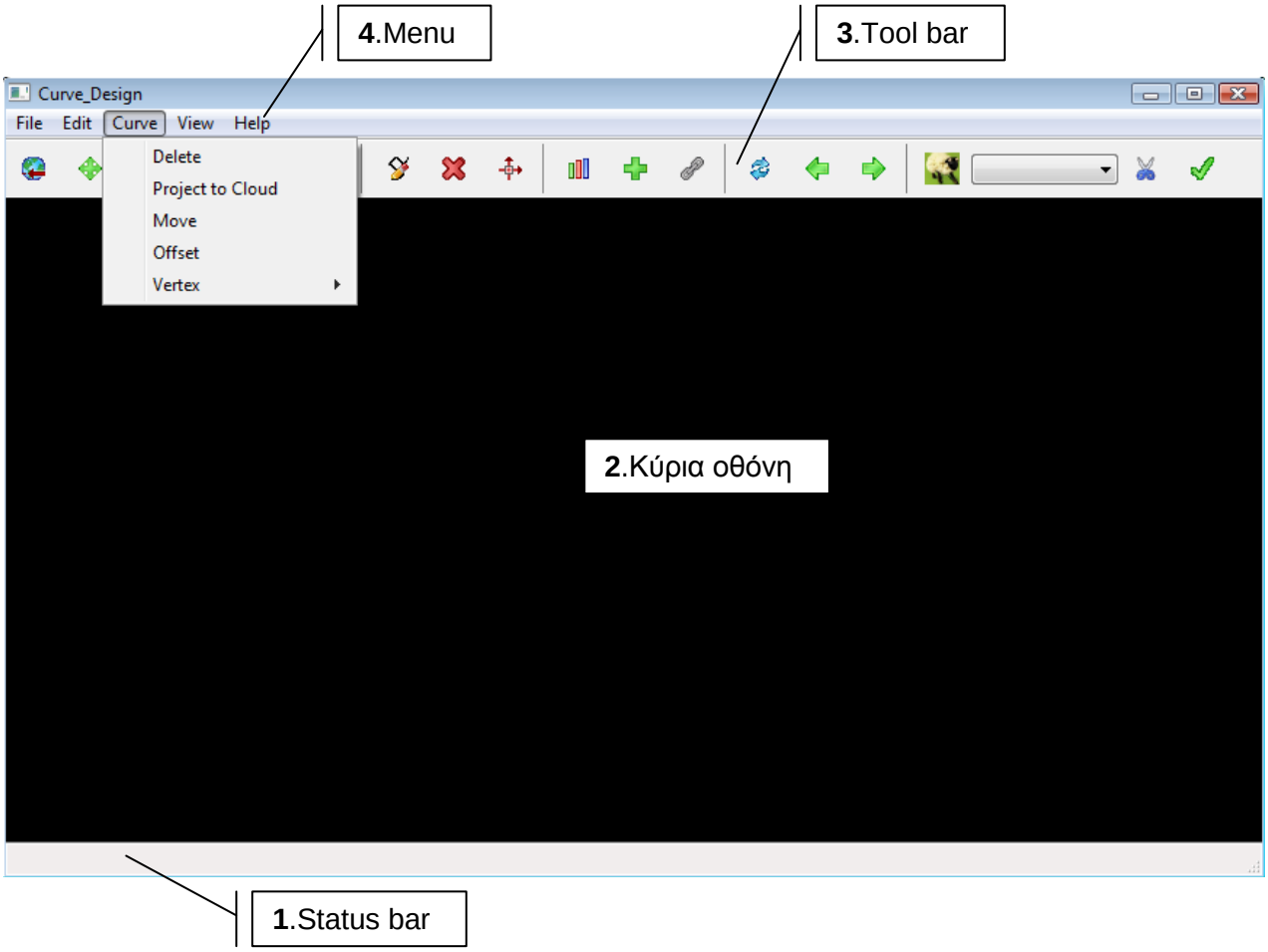
Σ’αυτό το σημείο θα μπορούσε να γίνει μια περιγραφή του τρόπου λειτουργίας των εργαλείων και της διεπαφής με το χρήστη. Θα πρέπει να σημειώσουμε ότι η εφαρμογή αυτή έχει στηθεί πάνω σε πολύ βασικές αρχές. Οι λόγοι που μας οδήγησαν εκεί είναι απλοί, δε σκοπεύουμε να αντικαταστήσουμε κάποιο σχεδιαστικό πρόγραμμα ούτε να κερδίσουμε μερίδια αγοράς απλά να προτείνουμε ένα νέο τρόπο σχεδίασης που θα δώσει λύσεις και ιδέες και σε άλλους ενδιαφερόμενους να προχωρήσουν σε ένα διαφορετικό επίπεδο.

Αρχικά θα πρέπει να ξεκαθαρίσουμε ότι όσο αφορά το GUI της διεπαφής αυτής όπως έχουμε προαναφέρει αυτό στηρίχτηκε και στήθηκε στο wxDev. Επίσης θα πρέπει να αναφερθεί ότι το interface χωρίζεται σε 4 κομμάτια ή περιοχές. Αυτά είναι το Status bar**(1)** η .Κύρια οθόνη **(2)** το Tool bar **(3)** και το Menu **(4)**

1.Θα πρέπει να αναφέρουμε ότι το Status bar βρίσκεται στο κάτω μέρος της οθόνης. Ο ρόλος του σε αυτή την εφαρμογή είναι να επεξηγεί το κάθε τι που επιθυμούμε να προβάλουμε, είτε στην οθόνη είτε στο Tool bar είτε δουλεύοντας τα διάφορα εργαλεία. Ουσιαστικά το Status bar θα ενημερώνει τον χρήστη για το τι είναι σωστό και πως πρέπει να λειτουργεί το κάθε τι στην εφαρμογή. Θα λέγαμε ότι είναι συμπληρωματικό του Help αλλά με πιο ευρεία έννοια και πιο γενικό ρολό που αφορά όμως το κάθε κουμπί και εντολή ξεχωριστά για το πώς ενδείκνυται να λειτούργει.
Έτσι κοιτώντας πολύ απλά ο χρήστης-σχεδιαστής την Status bar στην οθόνη του να γνωρίζει πως θα δουλέψει και τι μπορεί να κάνει. Είναι ευκόλως κατανοητό πως και αυτό θα ακολουθεί το χρώμα και στυλ της όλης εφαρμογής, ενώ ταυτόχρονα θα ξεχωρίζει για τα μηνύματα που θα εμφανίζει.

2. Η κυρία οθόνη είναι αυτή στην οποία θα είναι ορατές όλες οι κύριες λειτουργιές. Όλα θα έχουν ως σημείο αναφοράς αυτήν και εκεί θα είναι ορατό και το αποτέλεσμα των ενεργειών από αυτά που έχουμε κάνει. Σχετικά με το χρώμα της εφαρμογής αυτό θα είναι αυστηρά μαύρο σε όλη την επιφάνεια του φόντου για να μην αποσπάτε η προσοχή από διάφορα αλλά χρώματα και στοιχεία

και να δίνει μια συνοχή πληροφοριών σε όλη την έκταση ώστε να αποδίδονται και οι διαφορές εντολές με σαφήνεια και ευκρίνεια

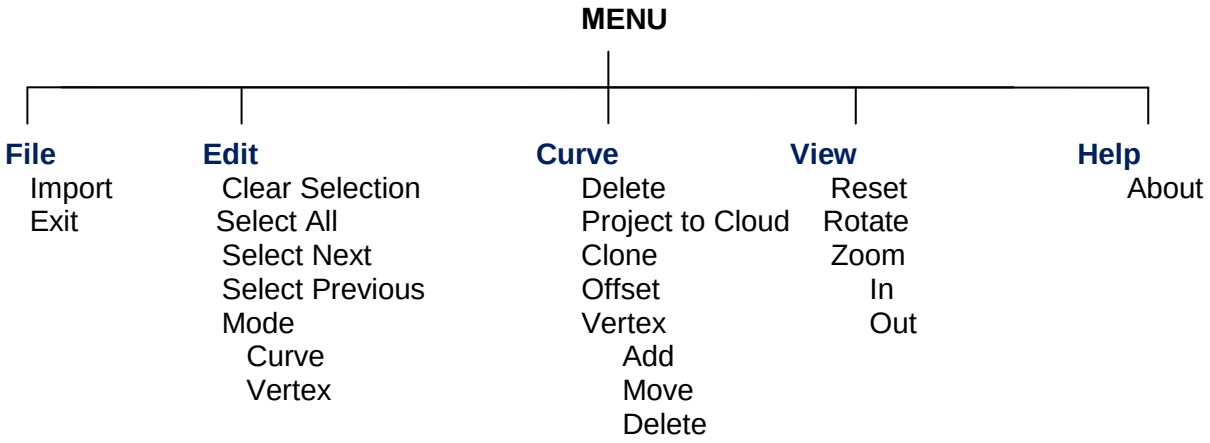


Σχήμα(6) Κύρια οθόνη εφαρμογής

3. Τα Tool bar είναι η γραμμή των πιο συχνών εντολών της εφαρμογής. Αποτελούν ουσιαστικά shortcuts των εντολών που υπάρχουν στο Menu αλλά για λογούς ευκολίας και ταχύτητας έχουν τοποθετηθεί σε μια μπάρα πάνω από την κυρία οθόνη. Συνεχίζει να επικρατεί το μοτίβο του γκρι φόντου για τα κουμπιά και διατηρώντας τις βασικές αρχές τις εργονομίας των εφαρμογών πετυχαίνουμε αυτό που αποκαλέσαμε πριν απλότητα και λειτουργικότητα, δυο βασικά στοιχειά που πρέπει να συνδυάζονται σωστά για να έχουμε και σωστά αποτελέσματα.

Τα κουμπιά όπως προαναφέραμε διακρίνονται για την ομοιογένεια τους μέσα από τη στοίχιση που έχει δημιουργηθεί. Ακόμη, υπάρχει αναλογία στην εφαρμογή πάντα με βάση το στήσιμο ολόκληρης της οθόνης. Το βασικότερο όλων όμως είναι τα μηνύματα που περνάμε μέσα από όλα αυτά. Πιο συγκεκριμένα θα πρέπει τα κουμπιά από το Tool bar να είναι έτσι δομημένα ώστε να αφήνουν μηνύματα στον χρηστή-σχεδιαστή, να είναι κατανοητά για να αποτελούν ευχάριστη ενασχόληση για το χρήστη Όποτε σημαντικό ρολό σε αυτά το σημείο έχει και η επιλογή των σωστών εικόνων και επεξεργασία αυτών ώστε να απεικονίζουν όσο το δυνατόν πιο απλά τις λειτουργιές και εντολές που πρόκειται να τρέξουν.

4. Το τελευταίο κομμάτι της εφαρμογής είναι το Menu bar. Στο σημείο αυτό θα πρέπει να αναφέρουμε ότι ο βασικός μας στόχος είναι να δημιουργήσουμε ένα περιβάλλον για το χρήστη όσο πιο οικείο και παρεμφερές με τις εφαρμογές που ήδη έχει συνηθίσει να λειτούργει. Το γεγονός αυτό μας οδήγησε να ομαδοποιήσουμε της εντολές σε πέντε κλασσικές κατηγορίες.



Σχήμα(7) Κατηγοριοποίηση εντολών και περιεχομένου

Αυτές είναι αρχικά το File, το όποιο είναι και στα περισσότερα προγράμματα που χρησιμοποιούμε και οι υπό εντολές που η συγκεκριμένη εντολή περιλαμβάνει είναι το Import, Exit. Επιγραμματικά παρουσιάζεται η λειτουργιά του κάθε ενός από αυτά αν και υποδηλώνονται από το ίδιο το όνομα Έτσι έχουμε το Import που είναι αυτό που σχετίζεται με την εισαγωγή νεφών σημείων στην συγκεκριμένη εφαρμογή. Το Exit μας οδηγεί στο να εγκαταλείψουμε το πρόγραμμα.

Το Edit είναι αυτό που επιμελείται κάποια στοιχειά για την επεξεργασία της εφαρμογής. Αυτά είναι τα Clear Selection, Select All, Select Next, Select Previous, Mode. Το Clear Selection, είναι αυτό που φέρει την εντολή για την διαγραφή όλων των εντολών που έχουν γίνει μέχρι εκείνη την στιγμή που την επικαλούμαστε. Βέβαια από default είναι σε θέση να διαγράφει μόνο σημεία όμως ανάλογα με το Mode που έχουμε διαλέξει ανάμεσα σε σημεία ή καμπύλες τότε λειτούργει με βάση αυτή την διάκριση. Η εντολή Select All έχει να κάνει την επιλογή όλων των σημείων ή καμπυλών που έχουμε κάνει και επιθυμούμε να επεξεργαστούμε μέχρι εκείνη τη χρονική στιγμή. Για το τι έπεται είναι στην επιλογή του χρήστη. Η Select Next, είναι μια εντολή η οποία μας επιτρέπει την επεξεργασία κάποιων σημείων ή καμπυλών. Η ιδία φιλοσοφία επικρατεί και στην εντολή Select Previous. Όπως καταλαβαίνουμε και οι δυο αυτές εντολές είναι πολύ σημαντικές για την τοπική επεξεργασία στην εφαρμογή. Η εντολή Mode δίνει τη δυνατότητα στο χρήστη-σχεδιαστή να επιλέξει τι ακριβώς έχει σκοπό να κάνει. Επεξεργασία σε κάποιο Curve ή επεξεργασία σε κάποια Vertex που θα αποτελέσουν την καμπύλη στην συνέχεια

Η εντολή Curve που αποτελείται από τις εντολές Delete, Project to Cloud, Clone, Offset, Vertex. Όπως καταλαβαίνουμε και από το όνομα της ομάδας είναι ένα σύνολο εντολών που έχουν να κάνουν με τις καμπύλες και πως αυτές μπορούμε να τις χειριστούμε με τον πιο σωστό σχεδιαστικό λειτουργικό αλλά και εργονομικό τρόπο. Αρχικά το Delete που σχετίζεται με την ακύρωση, δηλαδή τη διαγραφή μιας καμπύλης ή σημείου από αυτά που έχουμε κάνει στο νέφος σημείων. Στην συνεχεία έχουμε το Project to Cloud, η συγκεκριμένη εντολή σχετίζεται με τη λειτουργία και την προβολή της καμπύλης πάνω στο νέφος. Στην συνεχεία έχουμε την εντολή Move. Η εντολή αυτή έχει να κάνει με τον τρόπο που θα μπορούσαμε να κινήσουμε καμπύλες που είναι αποτυπωμένες στο νέφος σημείων σε κάποιο άλλο μέρος του νέφους χωρίς όμως αυτό να αλλάξει την δομή της καμπύλης. Το εγχείρημα αυτό στην ουσία είναι κάτι μεταξύ Copy και Paste αλλά επειδή το Copy στη συγκεκριμένη εφαρμογή ίσως να μην είχε πολύ νόημα αφού η κάθε καμπύλη χαρακτηρίζεται από την μοναδικότητα της στο 3D χώρο. Παρόλα αυτά το Move δεν παύει να είναι ένα πολύ χρήσιμο εργαλείο για να συμπληρώσει το σύνολο των εντολών και εργαλείων της εφαρμογής. Το επόμενο εργαλείο που θα αναπτύξουμε είναι το Offset. Πρόκειται για μία εντολή η

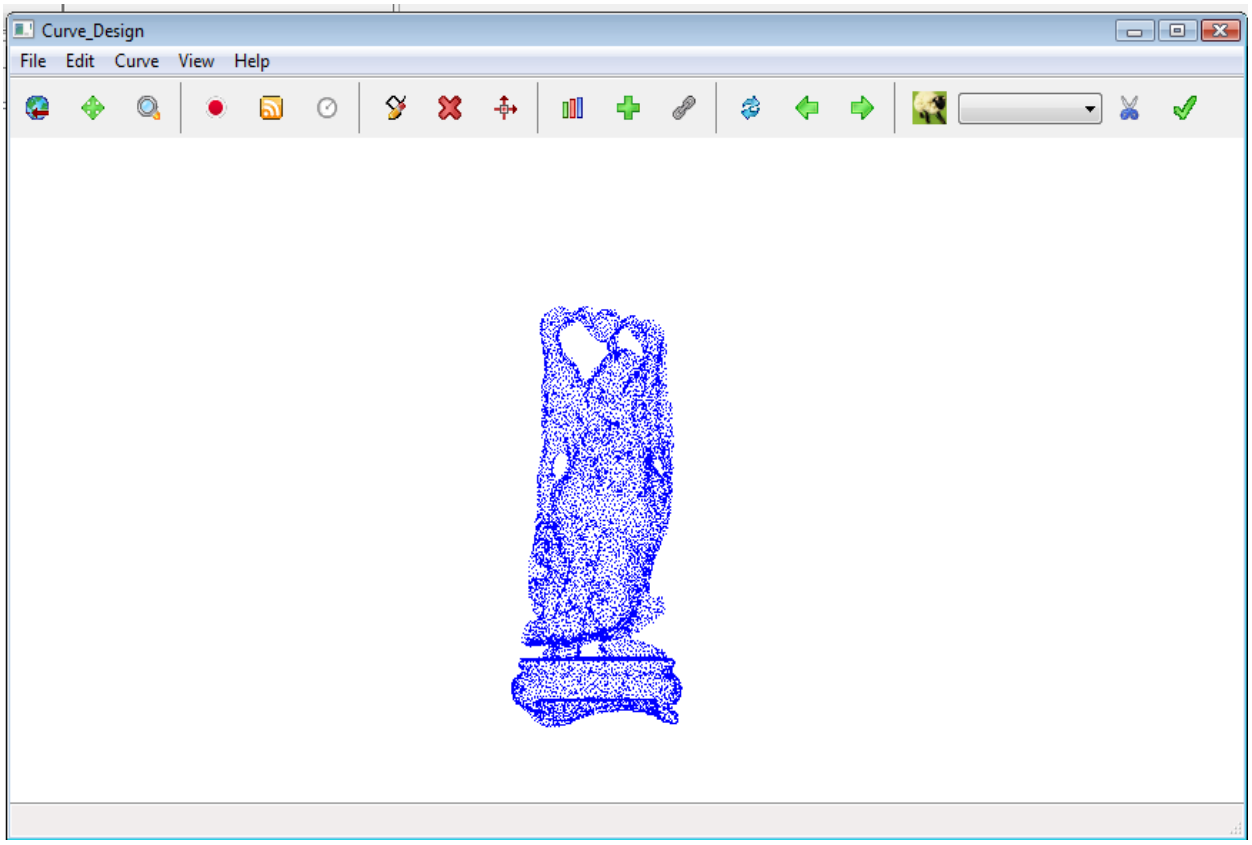
οποία μπορεί να είναι πολύ χρήσιμη για την ανάπτυξη επαναλαμβανομένων στοιχείων. Με αυτό τον τρόπο καταφέρνουμε να έχουμε την ιδία καμπύλη σε μια σταθερή απόσταση από εκεί που έχουμε ορίσει για να σχηματιστεί κάτι καινούριο στην βάση του πρώτου. Όπως και η Vertex που μας επιτρέπει να χειριστούμε με εύκολο τρόπο την επεξεργασία των σημείων που έχουν προβληθεί.

Η τέταρτη ομάδα εντολών είναι η View που σχετίζεται με αυτά που εμείς σχεδιάζουμε και θέλουμε να έχουμε οπτική απεικόνιση. Η View αποτελείται από υπό εντολές όπως είναι Reset, Rotate, Zoom (In, Out). Έτσι η Reset είναι η εντολή που μας επιστρέφει το νέφος στην αρχική κατάσταση που αυτό εμφανίστηκε στην οθόνη μας, διευκολύνοντας έτσι αρκετές φορές από συνεχόμενα ίσως Zoom In ή Out . Η εντολή Rotate είναι αυτή που έχουμε ήδη αναφέρει στην περιγραφή που έχουμε κάνει. Τελευταίο το Zoom (In, Out), λίγο πολύ είναι γνωστό σε όλους τους χρηστές ο τρόπος λειτουργιάς του.

Η τελευταία εντολή που θα αναπτυχθεί στην διεπαφή είναι η Help. Όπως καταλαβαίνουμε θα είναι μια εντολή που θα βοηθήσει το χρήστη στην επίλυση προβλημάτων αλλά σε πολύ χαμηλό επίπεδο. Ο χρήστης μπορεί έτσι να έχει ανά πάσα στιγμή γενικότερες πληροφορίες για τον τρόπο λειτουργίας του. [65].

5.3 Ενδεικτικές οθόνες της εφαρμογής

Η τελευταία διαδικασία πριν την εισαγωγή του κώδικα στην εφαρμογή είναι η δημιουργία του πρωτότυπου. Αυτό συμβάλει ώστε ο σχεδιαστής-χρήστης να αποκτήσει μια πιο απτή εικόνα για τη μορφή που θα έχει η εφαρμογή. Το πρωτότυπο ενδέχεται να μην είναι απόλυτα λειτουργικό και να χρειάζεται κάποια περεταίρω επεξεργασία και δημιουργία στον τομέα της διεπαφής (mock up demo) δείχνει όμως μια γενική εικόνα της μορφής που θα έχει κατά την λειτουργία της η εφαρμογή.
Στόχος του πρωτοτύπου είναι περισσότερο να παρουσιάσει πως θα είναι το τελικό αποτελέσματα έτσι ώστε ο σχεδιαστής να έχει μια γενική εικόνα της εφαρμογής. Τα γραφικά, το περιεχόμενο, η διάταξη της οθόνης όπως και οι λειτουργίες περιγράφονται σε αφαιρετικό επίπεδο.



Σχήμα(8) Ενδεικτική οθόνη εφαρμογής με άσπρο φόντο

6. Η Έννοια του Digital-CAD

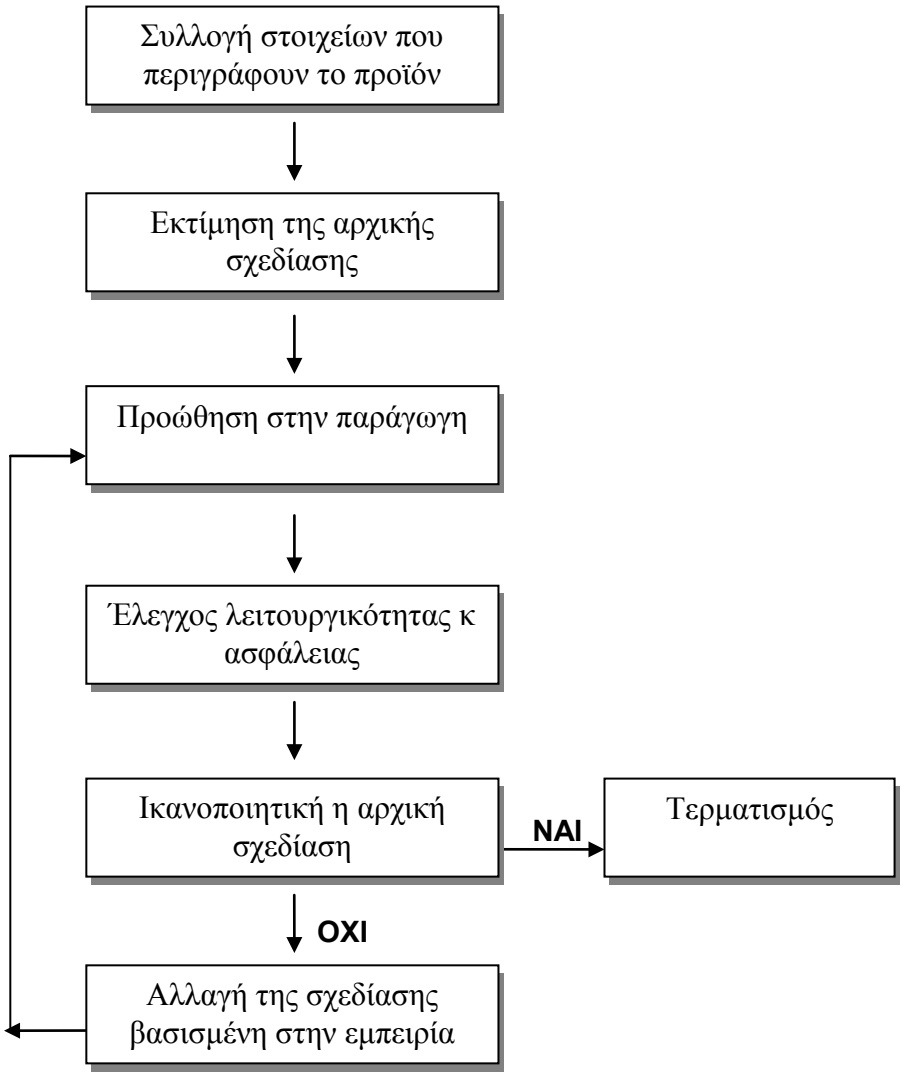
Η διαδικασία σχεδίασης ενός προϊόντος ή μίας κατασκευής γίνεται αισθητά πιο εύκολη, καθώς η τεχνολογία και η επιστήμη εξελίσσονται συνεχώς παρέχοντας νέα γνώση. Οι σχεδιαστές αντιμετωπίζουν την πρόκληση της συνεχούς ενημέρωσης και εξοικείωσης με τα νέα τεχνολογικά εργαλεία ώστε το προϊόν της εργασίας τους να είναι βιώσιμο και ανταγωνιστικό. Επιπλέον, οι σχεδιαστές πρέπει να επεκτείνονται και σε άλλους τομείς εξειδίκευσης για το προϊόν που πρόκειται να σχεδιάσουν έτσι ώστε να μπορεί το αρχικό μοντέλο να καλύπτει όλες τις απαιτούμενες ανάγκες και τους περιορισμούς και συνεπώς να ελαττώνεται ο χρόνος ανάπτυξης και παραγωγής του τελικού προϊόντος.

Κατά το παρελθόν, οι βιομηχανικοί σχεδιαστές ετοίμαζαν το προϊόν, το οποίο περνούσε στην παραγωγή χωρίς πλήρη έλεγχο της λειτουργικότητας και της ασφάλειας του και συχνά χωρίς να ικανοποιούνται όλες οι απαιτήσεις και οι περιορισμοί που είχαν τεθεί για το συγκεκριμένο προϊόν. Η συνολική διαδικασία ήταν αρκετά αργή, δοκιμαστική και τα λάθη της σχεδίασης γινόντουσαν αντιληπτά με ιδιαίτερη καθυστέρηση. Η δυνατότητα παραγωγής του προϊόντος δεν ήταν εφικτή με την πρώτη δοκιμή και χρειάζονταν να γίνουν αρκετές επαναλήψεις μέχρι την τελική λύση.

Με την ανάπτυξη όμως των κατάλληλων λογισμικών εργαλείων η σχεδίαση ενός προϊόντος μπορεί να είναι γρήγορη, ασφαλής και αποδοτική, καθώς ελέγχεται η ικανοποίηση των κριτηρίων λειτουργίας και των λοιπών συνθηκών πριν το προϊόν μπει στη διαδικασία παραγωγής.

6.1 Μεθοδολογία Βέλτιστης Σχεδίασης

Η παραδοσιακή προσέγγιση της σχεδίασης βασίζεται κυρίως στην ατομική γνώση, την εμπειρία και σε γενικούς κανονισμούς. Στο παρακάτω διάγραμμα παρουσιάζεται η κλασσική διαδικασία σχεδίασης. Αρχικά, προσδιορίζεται το πρόβλημα και η λύση προσεγγίζεται βάση της προηγούμενης εμπειρίας και επίκτητης γνώσης. Το προϊόν περνά στην παραγωγή, εάν ικανοποιεί πλήρως τις απαιτούμενες συνθήκες, αλλιώς τροποποιείται για να γίνει ο έλεγχος εκ νέου. Η επαναληπτική αυτή διαδικασία σχεδίασης εκτελείται μέχρις ότου επιτευχθεί μια ικανοποιητική λύση.



Σχήμα(9) Διάγραμμα παραδοσιακής μεθόδου προσέγγισης της σχεδίασης

Ακολουθώντας τη συμβατική μέθοδο προσέγγισης του προϊόντος, απαιτούνταν αρκετοί μήνες, ενώ παράλληλα τα κριτήρια που έπρεπε να ικανοποιούνται ήταν πιο ελαστικά με στόχο να αποφευχθούν οι πολυάριθμες επαναλήψεις και να προχωρήσει η διαδικασία παραγωγής. Αντίθετα στις μέρες μας, χάριν των υπολογιστικών εργαλείων, η σχεδίαση είναι πιο απαιτητική και αυστηρή ενώ οι σχεδιαστές πρέπει να παρέχουν βέλτιστες και επαρκείς λύσεις για το νέο προϊόν μέσα σε μερικές εβδομάδες. Στο ακόλουθο διάγραμμα φαίνεται η σχεδιαστική διαδικασία που ακολουθείται για την προσέγγιση βέλτιστου προϊόντος. Η βέλτιστη σχεδίαση επιτυγχάνεται με μεθόδους που παρέχονται από τα

σύγχρονα λογισμικά πακέτα και τα οποία βασίζονται στη μέθοδο των πεπερασμένων στοιχείων. Έτσι εξασφαλίζονται πιο ακριβείς υπολογισμοί που επιτρέπουν την άμεση αξιολόγηση και βελτίωση των προϊόντων, καθώς τα νέα υπολογιστικά εργαλεία προσφέρουν τη δυνατότητα στον σχεδιαστή ερευνητή να αναζητήσει εύκολα και γρήγορα βέλτιστες λύσεις για μία κατασκευή θέτοντας παραμέτρους, περιορισμούς και στόχους, κάτι που είναι εξαιρετικά χρονοβόρο και επίπονο να επιτευχθεί με τη συμβατική μέθοδο ή με αναλυτικό τρόπο.

Σχήμα(10) Διάγραμμα βέλτιστης σχεδίασης προϊόντος

Ο καταναλωτισμός και η εκθετική αύξηση των απαιτήσεων οδήγησε στην δημιουργία ποιοτικών προϊόντων σε μεγάλες ποσότητες. Ταυτόχρονα όμως ώθησε την βιομηχανία στην διαμόρφωση νέων μεθόδων παραγωγής και στην αξιοποίηση νέων τεχνολογιών. Οι πιέσεις αυτές που δημιουργήθηκαν είχαν ως αποτέλεσμα την συντόμευση της διάρκειας ζωής του κύκλου των προϊόντων και στην αύξηση της ανάγκης για εξατομίκευση αυτών.
Εστιάζοντας την προσοχή μας στη σχεδίαση προϊόντων αναφερόμαστε σε συστήματα CAD, των οποίων η εξέλιξη ακολούθησε τα βήματα της "Ηλεκτρονικής Επανάστασης" και συνετέλεσαν στην εξοικονόμηση χρόνου, στη μείωση του κόστους παραγωγής και στην αύξηση της ποιότητας των παραγόμενων προϊόντων.

Χαρακτηριστικό των προγραμμάτων CAD είναι η δυνατότητα σχεδίασης με ακρίβεια και ταχύτητα στην οθόνη του υπολογιστή των διαφόρων αντικειμένων ή κατασκευών που θέλουμε να δημιουργήσουμε, όσο πολύπλοκες και εάν είναι αυτές, η δυνατότητα άμεσης επέμβασης στην μορφή τους, και η δυνατότητα εξέτασης πολλαπλών παραλλαγών τους, χωρίς να κατασκευάζεται το

τελικό προϊόν στην πραγματικότητα. Οι εταιρείες που μπορούν να ανταποκριθούν σε αυτές τις απαιτήσεις των καταναλωτών θα είναι πιο επιτυχημένες από άλλες. Τα πιθανά οφέλη για τις εταιρείες που εμπλέκονται στην τεχνολογία CAD είναι πολλαπλά και εγγυημένα και όλα αυτά αποσκοπούν στο περισσότερο και μεγαλύτερο κέρδος για την εταιρεία. Ορισμένα από αυτά τα οφέλη του CAD είναι τα εξής.

1. Ταυτόχρονη μηχανική Concurrent Engineering (CE) – Μηχανικών και διαδικασιών παραγωγής ενεργοποιημένα ταυτόχρονα από κοινού σε CAD δεδομένα.
2. Υψηλή ποιότητα – Χάρη στην αυξημένη αποτελεσματικότητα που προκύπτει από την ικανότητά να αναλύουμε σε μεγαλύτερο βαθμό το σχεδιασμό του αντικειμένου κατά τη διάρκεια της ανάπτυξης του προϊόντος.
3. Χαμηλότερη μονάδα κόστους – Λόγω της μειώσεις του κόστους κατά την διάρκεια της ανάπτυξης και του πρωτότυπο.
4. Ταχεία προτυποποίηση Rapid prototyping (RP) – Τα CAD μοντέλα μπορούν να χρησιμοποιηθούν για την παραγωγή πρωτοτύπων από Stereolithography και από άλλες τεχνολογίες RP
5. Εξέλιξη προσωπικού – Τα CAD παρέχουν ένα άκρος προκλητικό περιβάλλον για τους εργαζόμενους. Μια ποικιλία θέσεων που αφορούν τη διαχείριση και την εποπτεία των CAD συστημάτων είναι διαθέσιμα για την προώθηση της σταδιοδρομίας του κάθε εργαζόμενου.
6. Αύξηση του όγκου εργασίας – Η αποτελεσματική χρήση CAD επιτρέπει την αύξηση της παραγωγικής εργασίας, διατηρώντας παράλληλα υψηλό το προσωπικό επίπεδο.
7. Αμφίδρομος έλεγχος της παραγωγικής διαδικασίας – Τα CAD και τα NC εργαλεία επιτρέπουν την δημιουργία διαύλου επικοινωνίας και ενημέρωσης με τις μηχανές τα όποια επαληθεύονται αυτόματα με ελάχιστη ανθρώπινη παρέμβαση.
8. Βελτίωση της συνολικής επικοινωνίας – Το CAD επιτρέπει και συντελεί στη μετάβαση από την παραδοσιακή σχεδίαση σε χαρτί, στο σχεδιασμό και την κατασκευή συστήματος σε ηλεκτρονική μορφή.
9. Αυξημένη ακρίβεια των δεδομένων MRP – Τα CAD αρχεία και δεδομένα μπορούν εύκολα να συνδέονται και να διαχειρίζεται από λογισμικό MRP.
10. Αυξημένη ευελιξία σχεδιασμού – Το CAD προσφέρει ένα πιο εύρωστο σύνολο εργαλείων και μεθόδων για την τροποποίηση των σχεδίων.
11. Αύξηση της ακεραιότητας των δεδομένων σχεδιασμού – Με ένα και μόνο μοντέλο CAD έχουμε υποστήριξη όλων των μεταγενέστερων διαδικασιών, οι αλλαγές που μπορεί να προκύψουν ανταποκρίνονται γρήγορα και με ακρίβεια.

7.Γεωμετρικά Εργαλεία Ψηφιακής Σχεδίασης Προϊόντων

7.1 Εξέλιξη στο Χώρο τον Γραφικών

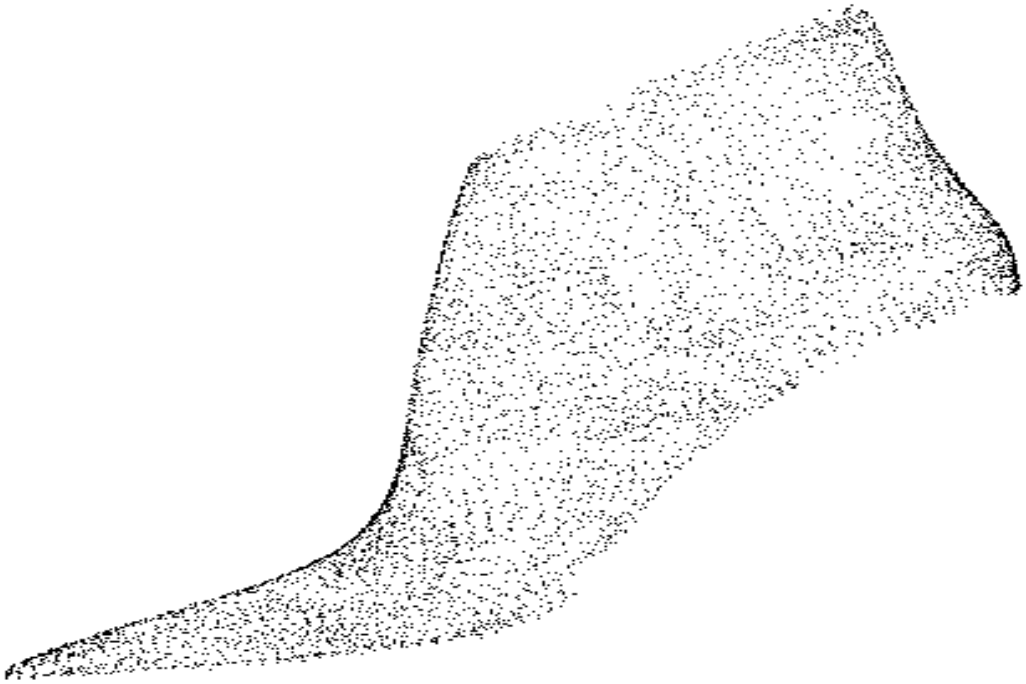
Στο χώρο των Γραφικών με Η/Υ έχει αναπτυχθεί τα τελευταία χρόνια μια έντονη τάση για την αξιοποίηση επιφανειακών μοντέλων που βασίζονται σε διακριτά στοιχεία όπως είναι τα πολύγωνα, τα τρίγωνα και πολύ πρόσφατα τα σημεία σύμφωνα και με Azariadis & Sapidis [1, σ.109-121].

Το πολύγωνα και τα τρίγωνα θεωρούνται εδώ και πολύ καιρό τα βασικά στοιχειά των γραφικών ενώ πρόσφατα έχει παρατηρηθεί αύξηση στη σημασία και έννοια του σημείου στους χώρους CAD/CAE/CAM. Είναι γεγονός πως οι μοντέρνες τεχνολογίες σχεδιάσεις προϊόντων όπως είναι η Αντίστροφη Μηχανική [50], η Εικονική Μηχανική [15] και η Ταχεία Προτυποποίηση [37] χρησιμοποιούν σχεδόν αποκλειστικά πολυγωνικά μοντέλα [28]. Στις μέρες μας η τάση για «διακριτά μοντέλα» έχει πλέον επικρατήσει σε ολόκληρο τον επιστημονικό κόσμο φέρνοντας στην επιφάνεια το πιο απλό γεωμετρικό στοιχειό που είναι το «σημείο». Οι λόγοι είναι πολύ απλοί όσο και πειστικοί:

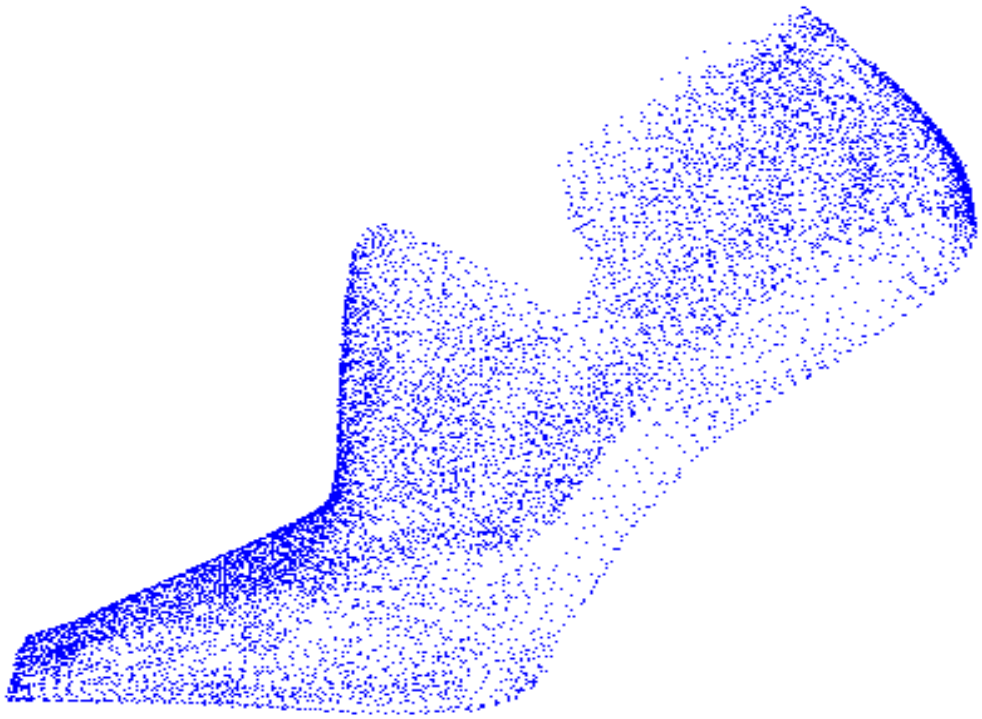
στα σύγχρονα Γραφικά «το μέγεθος των τριγώνων είναι εφάμιλλο του μεγέθους των pixel» [3] έτσι τα μη συνδεδεμένα σημεία είναι η προφανής επιλογή για το μοντέλο των επιφανειών. Για το CAD εφόσον η σχεδίαση ενός προϊόντος αρχίζει από σημεία (είτε από τον χρηστή είτε μέσω αντιστροφής μηχανικής) και τελειώνει σε σημεία (Εικονική Μηχανική/ανάλυση μοντέλων ή δεδομένα NC) γεννιέται το ερώτημα γιατί να χρησιμοποιηθεί κάτι άλλο ως πρωταρχικό μοντέλο;

Η σχεδίαση όμως σε 3D νέφη σημείων είναι ένα πολύ δύσκολο εγχείρημα, που ξεπερνά τα όρια της συνηθισμένης αντίληψης των 2D διαστάσεων. Αν λάβουμε υπόψη και το γεγονός ότι οι δημοσιεύσεις στο συγκεκριμένο θέμα είναι ελάχιστες και ασχολούνται με τις 2D εκδοχές του προβλήματος τότε καταλαβαίνουμε το κενό που υπάρχει στην συγκεκριμένη προσέγγιση. Η ιδέα είναι να αναπτύξουμε σχεδιαστικά εργαλεία τα όποια προσεγγίζουν τα νέφη σημείων και στη πορεία θα επικεντρωθούμε στην σχεδίαση καμπυλών. Στόχος μας θα είναι να προσεγγίσουμε όσο το δυνατόν πιο κοντά και πιο ομαλά τα σημεία του νέφους με διάφορες τεχνικές εξομαλύνσεις πάνω στις B-Spline καμπύλες. Πρόσφατα οι Benko και Varady [7] ασχολήθηκαν με την τριγωνοποίηση των σημείων σε επιφάνειες αλλά η καμπύλη προσέγγισης δεν μπορούσε να εφαρμοστεί και δημιούργησε επιπλέον προβλήματα. Επίσης η πολυγωνική παρεμβολή σε 3D νέφη που εμφανίζεται στο [21] είναι μια ευριστική προσέγγιση στο θέμα όπου λύνει κατά κάποιο τρόπο το 2D πρόβλημα πράγμα το όποιο δίνει μια προσέγγιση στην ανάπτυξη του θέματος.

Ο Lee [19] ο όποιος επιζητεί και αυτός την λύση στο πρόβλημα των καμπυλών παρεμβολής σε νέφη σημείων έχει όμως να αντιμετωπίσει δυσκολίες που σχετίζονται με τα διάφορα παχύ των νεφών. Αν και έχει καταλήξει σε μια μέθοδο moving least-squares των σημείων, παρόλα αυτά η μέθοδος του αυτή δεν ενδείκνυται για 3D περιπτώσεις μιας και δεν έχει καθορίσει. α) την επιλογή των αρχικών σημείων από που θα αρχίσει ο αλγόριθμος και β) περιλαμβάνει 4 παραμέτρους που πρέπει ο χρήστης να ορίσει. γ) επίσης δεν έχει αποδεικτικές μεθόδους μετρήσεις για το χρόνο και τα ποσοστά επιτυχίας της μεθόδου αυτής.



Σχήμα(11) Ενδεικτικό Point Cloud



Σχήμα(12) Ενδεικτικό Point Cloud

7.2 Θεωρητικό υπόβαθρο του αλγορίθμου προβολής

Η πρόταση που θα αναπτυχτεί ως τεχνική για τη σχεδίαση καμπυλών παρεμβολής μεταξύ σημείων θα πρέπει να σημειωθεί ότι δε χρησιμοποίει καμία μέθοδο επιφανειών παρεμβολής (Όλα τα παρακάτω αποτελούν αποσπάσματα από papers των Azariadi & Sapidi.)

1. Η σχεδίαση καμπυλών σε νέφη σημείων παρουσιάζεται ως ένας “low level” σχεδιασμός που όμως περιλαμβάνει πολλές “high level” λειτουργίες [32].
2. Με την μέθοδο της απευθείας προβολής στο νέφος σημείων αποφεύγουμε να έχουμε απώλειες πληροφοριών για τα σημεία (πράγμα που συμβαίνει με την επανασχεδίαση επιφανειών)
3. Τα 3D Scanner που υπάρχουν στο εμπόριο παράγουν μεγάλης ακριβείας νέφη δημιουργώντας έτσι τα παρακάτω συμπεράσματα.

α. Παρόλο που τα νέφη σημείων όπως προαναφέραμε έχουν μεγάλη πυκνότητα, συχνά παρατηρούμε διάφορες περιοχές μειωμένης δειγματοληψίας (undersample area) δηλαδή περιοχές με μεγάλα κενά. Αυτό έχει μεγάλη σημασία για τα βιομηχανικά συστήματα τα όποια υιοθετούν ένα αλγόριθμο που δεν περιορίζεται σε αυστηρές συνθήκες πυκνότητας.

β. Πρέπει να διευκρινίσουμε ότι είναι ανούσιο να παρεμβάλεις επιφάνεια όταν έχεις την δυνατότητα να κάνεις παρεμβολή καμπυλών που προσφέρονται για επιφανειακή σχεδίαση.

γ. Είναι σωστό από την αρχή να γνωρίζουμε ότι στα πυκνά νέφη τα σημεία που τα αποτελούν δεν είναι εφοδιασμένα με κανονικά διανύσματα ούτε προσανατολισμένα.

δ. Θα πρέπει να λάβουμε υπόψη ότι το underlying surface σε ένα δοσμένο νέφος είναι είτε smooth είτε έχει γωνίες είτε θα είναι κλειστό.

Όλες αυτές οι παρατηρήσεις, αναφορές και δημοσιεύσεις έχουν να κάνουν με μεθόδους σχεδίασης επιφανειών από το πεδίο των Computational Geometry Computer Graphics [4,Section 2] και δεν

ενδείκνυνται για βιομηχανική χρήση. Άρα πλέον μπορούμε να πούμε πως είναι συνειδητή η επιλογή μας να αποφύγουμε να ασχοληθούμε με την επανασχεδίαση επιφανειών
Όλα αυτά σε συνδυασμό με τον προτεινόμενο αλγόριθμο που θα χρησιμοποιήσουμε στην πορεία μας οδηγούν στο συμπέρασμα ότι η μέθοδος που πρόκειται να ακολουθήσουμε είναι μια πολύ σταθερή μέθοδος η όποια ενδείκνυται για διαφορές πυκνότητες νεφών ανεξάρτητα από το μέγεθος και το πάχος (δηλαδή την πυκνότητα) που μπορεί να έχουν. Αν και όπως θα δούμε στην συνεχεία η πυκνότητα διαδραματίζει πρωταρχικό ρολό αρχικά στη σχεδίαση των καμπυλών και στον αλγόριθμο που θα χρησιμοποιήσουμε.

7.3 Προβολή σημείου σε νέφος

Στο κομμάτι αυτό θα αναπτύξουμε την κύριας θεωρία της προβολής του σημείου πάνω στο νέφος σημείων. Σε αυτό το σημείο θα πρέπει να σημειωθεί ότι το νέφος σημείων έχει προκύψει από 3D Scanners τα όποια όπως γνωρίζουμε είναι όργανα-συσκευές που συλλέγουν δεδομένα από το περιβάλλον ή τα αντικείμενα και τα μετατρέπουν σε ψηφιακή 3D μορφή νέφους [13]. Το νέφος σημείων είναι αυτό που θα μας απασχολήσει στην συγκεκριμένη εργασία, όπως απασχολεί πολλούς άλλους ερευνητές μιας και όπως έχουμε προαναφέρει πλέον τα σημεία αυτά βρίσκουν εφαρμογή σε παρά πολλούς τομείς βιομηχανία ψυχαγωγίας (video games, ταινίες), τεχνολογία (industrial design, orthotics and prosthetics), reverse engineering και rapid prototyping, quality control. Το νέφος αυτό όμως που προκύπτει από το πραγματικό αντικείμενο-προϊόν δεν έχει απολύτως κάποια τυπολογική πληροφορία[32][38].

Το να προβάλουμε όμως ένα σημείο πάνω σε μια επιφάνεια νέφους είναι μια δύσκολη διαδικασία η όποια θα πρέπει να βασίζεται πάνω σε μια σωστή μεθοδολογία. Αρχικά ορίζουμε μια συνάρτηση που θα μετράει την απόσταση μεταξύ του σημείου που θα προβληθεί και του νέφους. Η συνάρτηση αυτή ορίζεται ως error function. Στην συνέχεια το σημείο που μας ενδιαφέρει προβάλλεται πάνω στο νέφος σημείων ελαττώνοντας την error function στο μέγιστο βαθμό της. Κατά αυτόν τον τρόπο το επιλεγμένο σημείο είναι όσο γίνεται πιο κοντά πάνω στην επιφάνεια νέφους. Θα πρέπει να τονιστεί ότι στην παρούσα εργασία η μέθοδος που παρουσιάζουμε δεν προσφέρει απόλυτα ικανοποιητική λύση στο πρόβλημα του καθορισμού σημείου στο ‘point cloud. Όπως επίσης και η μέθοδος που παρουσιάζεται στην [13] δεν μπορεί να εφαρμοστεί απόλυτα στην παρούσα εργασία αφού βασίζεται στην ‘polygonal representation’ του μοντέλου επιφάνειας. Για την ορθότερη επίλυση όμως χρησιμοποιούμε ένα παραγόμενο ‘point cloud’, το όποιο και θα χρησιμοποιηθεί ως βοηθητικό νέφος για την επίλυση της προβολής του σημείου πάνω σε αυτό[39].

7.3.1 Το πρόβλημα της προβολής

Στο σημείο αυτό θα αναπτύξουμε τον τρόπο με τον όποιο επιτυγχάνεται η προβολή του σημείου πάνω στο νέφος. Η συγκεκριμένη διαδικασία αποτελεί και κατά κάποιο τρόπο τον πυρήνα της εργασίας αυτής. Για να μπορέσουμε να μπούμε όμως στην ουσία του προβλήματος της προβολής του σημείου θα πρέπει αρχικά να ορίσουμε το
$\boldsymbol{C_N = \{\rho_\mu=(x_\mu, y_\mu, z_\mu) \mid \mu = 0,\dots N\text{-}1\}}$ οπού θα είναι το πλήθος των δοσμένων σημείων νέφους για την εφαρμογή και $\boldsymbol{\rho} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ ένα τυχαίο 3D σημείο στο χώρο με $\boldsymbol{n = (n_\mu, n_\mu, n_\mu)}$ ένα αντίστοιχο

διάνυσμα προβολής του τυχαίου p σημείου στο χώρο. Ας θεωρήσουμε επίσης ότι το ζεύγος αυτό των p,n θα το συμβολίζουμε από εδώ και περά ως εξής $\hat{\rho}$=(ρ,n).
Ορισμός 1.[1]
Το προσβληθέν σημείο **ρ*** από το **ρ** σημείο του χώρου στην κατεύθυνση του n πάνω στο νέφος σημείων $\boldsymbol{C_N}$ ορίζεται ως εξής. Κάθε σημείο $p_\mu \in \boldsymbol{C_N}$ σχετίζεται μένα θετικό βάρος $\boldsymbol{\alpha_\mu}$. Από το σημείο ρ όπως και από το $\boldsymbol{\alpha_\mu}$ καθώς και από το προσβληθέν σημείο **ρ*** προκύπτει η λύση της εξίσωσης του προβλήματος προβολής. Έτσι βρίσκουμε το **ρ*** που ελαχιστοποιεί την

$$\mathbf{E(\rho^*)} = \sum_{k=0}^{N-1} \boldsymbol{\alpha_\mu} \, \| \boldsymbol{\rho^* - \rho_\mu} \|^2 \quad \mathbf{(1)}$$

Το ρ* ονομάζεται και προσβληθέν σημείο του $\hat{\rho}$=(ρ,n).στο $\boldsymbol{C_N}$. Για δοσμένα βάρη $\{\alpha_\mu\}$ το ρ* μπορεί να γραφτεί ως ρ*=(x*,y*,z*) και

$$\boldsymbol{\rho^* = \rho^*(t) = \rho + tn} \quad ..t \in R \quad \mathbf{(2)}$$

Μετά από όλα αυτά προκύπτει ότι λύση στο πρόβλημα προβολής **(1)** είναι η [1].
Να συμπληρώσουμε ότι το t προκύπτει από το

$$\mathbf{t} = \frac{\boldsymbol{\lambda - \rho \times n}}{\|\mathbf{n}\|^2} \quad \mathbf{(3)} \text{ οπου } \boldsymbol{\lambda} = \frac{c_1 n_x + c_2 n_y + c_3 n_z}{c_0},$$

$$c_0 = \sum_{\mu=0}^{N-1} \alpha_\mu \,, \quad c_1 = \sum_{\mu=0}^{N-1} \alpha_\mu x_\mu \,, \quad c_2 = \sum_{\mu=0}^{N-1} \alpha_\mu y_\mu \,, \quad c_3 = \sum_{\mu=0}^{N-1} \alpha_\mu z_\mu \ \mathbf{(4)}$$

Πλέον μπορούμε να ισχυριστούμε ότι τον κύριο ρολό στην μέθοδο της παρεμβολής σε δοσμένο νέφος σημείων από άλλο σημείο καθορίζεται από τη συνάρτηση (2) και (3), λαμβάνοντας ωστόσο πάντα υπόψη το σημείο $\hat{p}$

7.3.2 Ερευνώντας για το αν ισχύει ο Ορισμός 1.

Παρά τα όσα προαναφέραμε για τη μέθοδο προβολής σημείου και την εξάρτηση από τις εξισώσεις (2) και (3) ένα είναι το επιχείρημα που μας προσφέρει την θεωρητική δικαίωση για αυτά που ισχυριστήκαμε. Μια πιθανή προσέγγιση θα μπορούσε να θεωρηθεί ένα σύνολο σημείων C_N που βρίσκεται πάνω στη παραμετρική επιφάνεια S=S(u,v). Εφόσον το Sρ* είναι το προβαλλόμενο σημείο του $\hat{p}$ πάνω στο S οι μέθοδοι από τις εξισώσεις (2) και (3) πρέπει να μας δώσουν ένα σημείο ρ* αρκετά κοντά στο Sρ*. Μολονότι κανένα θεωρητικό αποτέλεσμα δεν έχει προσεγγίσει το θέμα από αυτή την κατεύθυνση, όποτε και τα σχετικά αριθμητικά δεδομένα μπορεί να τα θεωρήσουμε σωστά και ισχύοντα.

7.3.3 Ανάλυση σφάλματος για τη μέθοδο προβολής του Ορισμού 1

Η παρούσα μέθοδος που χρησιμοποιούμε για την προβολή σχετίζει την error απόσταση που έχουμε ορίσει με το αποτέλεσμα-λύση που προκύπτει από την απευθείας προβολή. Θεωρούμε από την αρχή ότι τα παραγόμενα σημεία από το νέφος έχουν τοποθετηθεί με την (maximum) μεγαλύτερη ευκλείδεια απόσταση από την κανονική τους τοποθεσία.

$C_N'=\{p_\mu'=(x_\mu',y_\mu',z_\mu') \; |||p_\mu'-p_\mu||\leq\varepsilon, \quad \mu=0,\ldots,N-1\}$.
Χρησιμοποιώντας το C_N' η λύση για το πρόβλημα της προβολής δίνεται από

$$\mathbf{t'} = \frac{\boldsymbol{\lambda'} - \boldsymbol{\rho}\times\mathbf{n}}{||\mathbf{n}||^2} \quad \textbf{(5)}$$ όπου

$$\boldsymbol{\lambda'} = \frac{\mathbf{c'_1 n_x + c'_2 n_y + c'_3 n_z}}{\mathbf{c'_0}}$$ και

$$\boldsymbol{C_0'= C_0}, \qquad \mathbf{c'_1} = \sum_{\mu=0}^{N-1} \boldsymbol{\alpha_\mu x'_\mu} \qquad \mathbf{c'_2} = \sum_{\mu=0}^{N-1} \boldsymbol{\alpha_\mu y'_\mu} \qquad \mathbf{c'_3} = \sum_{\mu=0}^{N-1} \boldsymbol{\alpha_\mu z'_\mu}$$

όποτε μετά έχουμε
$$\mathbf{|t'-t|=}\frac{|\boldsymbol{\lambda'} - \boldsymbol{\lambda}|}{||\mathbf{n}||^2} \quad \textbf{(6)}$$ και

$$|\boldsymbol{\lambda'} - \boldsymbol{\lambda}| = \frac{|(c'_1-\mathbf{c_1})\mathbf{n_x}+(c'_2-\mathbf{c_2})\mathbf{n_y}+(c'_3-\mathbf{c_3})\mathbf{n_z}|}{\mathbf{c_0}} \leq \frac{1}{\mathbf{C_0}}[|(\mathbf{c'_1} - \mathbf{c_1})||\mathbf{n_z}| + |(\mathbf{c'_2} - \mathbf{c_2})||\mathbf{n_z}| + |(\mathbf{c'_3} - \mathbf{c_3})||\mathbf{n_z}|] \leq \frac{1}{\mathbf{C_0}}\left[\boldsymbol{\varepsilon}\sum_{\mu=0}^{N-1}\boldsymbol{\alpha_\mu}(|\mathbf{n_x}|,|\mathbf{n_y}|,|\mathbf{n_z}|)\right]$$

Όπου και η πρώτη παράγωγος της παράστασης θα μου δώσει το :

$$\mathbf{|t'-t|} \leq \boldsymbol{\varepsilon}\frac{\sum_{\mu=0}^{N-1}\boldsymbol{\alpha_\mu}(|\mathbf{n_x}|,|\mathbf{n_y}|,|\mathbf{n_z}|)}{\mathbf{C_0}||\mathbf{n}||^2} \leq \boldsymbol{\varepsilon}\frac{3||\mathbf{n}||^2}{\mathbf{C_0}||\mathbf{n}||^2} = \boldsymbol{\varepsilon}\frac{3}{||\mathbf{n}||} \quad \textbf{(7)}$$

Άρα η εξίσωση (7) μας δείχνει ότι υπάρχει ένας σταθερός όρος κ ανεξάρτητος από το νέφος σημείων, το μέγεθος και την πυκνότητα του και θα ισχύει ότι $|t'-t|\leq \kappa\cdot\varepsilon$. Με αλλά λόγια το error που προκύπτει από την επίλυση της εξίσωσης (3) εξαρτάται γραμμικά από το error που προκύπτει στη μέτρηση του νέφους σημείων.

7.3.4 Επιλογή του σωστού βάρους

Το ειδικό βάρος α_μ που έχει το κάθε σημείο στο νέφος ανάλογα με τη θέση του σε αυτό, θα δούμε ότι παίζει καθοριστικό ρολό στον υπολογισμό του κατάλληλου σημείου ρ* και για τον λόγο αυτό θα πρέπει να επιλεχτεί με πολύ μεγάλη προσοχή. Γενικά αναφέρουμε ότι το βάρος $\alpha_\mu \geq 0$ στο δοσμένο νέφος $\rho_\mu \in C_N$ θα πρέπει να παίρνει μεγάλη τιμή όταν ρ_μ είναι κοντά στο σημείο ρ, το όποιο θα προβληθεί και να ελαττώνεται όσο η απόσταση από το ρ_μ στο ρ αυξάνεται.
Η συνάρτηση βάρους που ορίζεται συμφώνα και με το [1] είναι

$$\boldsymbol{\alpha_\mu}=\frac{1}{||\boldsymbol{\rho}-\boldsymbol{\rho_\mu}||^4} \quad \boldsymbol{\alpha_\mu}\in[0,\infty] \quad \textbf{(8)}$$

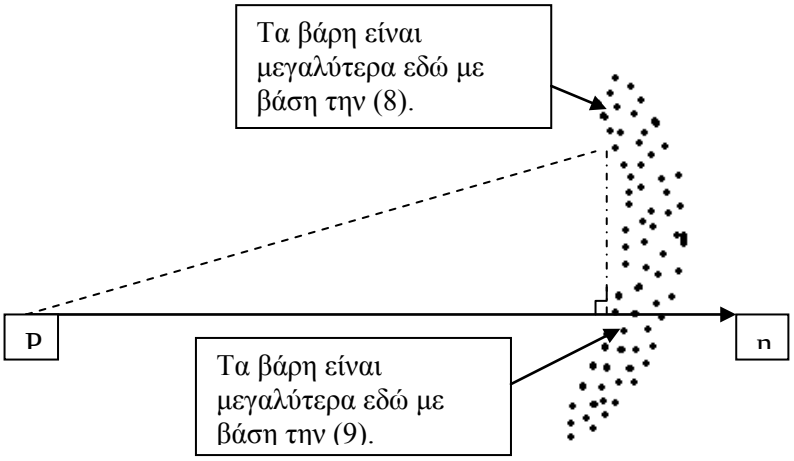
Η όποια όμως είναι ανεπαρκής για να περιλαμβάνει έννοιες όπως point-parameterization, όμως η παραπάνω συνάρτηση (8), λαμβάνει υπόψη της για τον υπολογισμό μόνο την απόσταση μεταξύ ρ_μ και ρ. Οι Erikson και Manocha [13] αξιοποίησαν την συνάρτηση βάρους βασισμένοι στην επιφάνεια και στα γειτονικά ρ_μ σημεία. Η μέθοδος αυτή όμως είναι ακατάλληλη για την εργασία αυτή. Αντιμετωπίζοντας όμως το πρόβλημα αυτό βαθύτερα και κάτω από τον Ορισμό 1 το όποιο μας ξεκαθαρίζει ότι στα νέα βάρη τα όποια υπολογίζονται από την συνάρτηση πρέπει να ληφθεί υπόψη και η κατεύθυνση n σε σχέση με το σημείο ρ.

Μετά από μελέτες, έρευνες και πειράματα που έγιναν καταλήξαμε στο ότι θα πρέπει στη συνάρτηση της απόστασης βάρους να λάβουμε υπόψη και τα δυο μεγέθη όπως την απόσταση των

μεταξύ τους ρ και το ρ_μ δηλ. $||\rho_\mu-\rho||$ όπως και στην ποσότητα $||(\rho_\mu-\rho) * n||$ που ορίζεται μεταξύ των ρ_μ και του άξονα ορισμένο από το ρ και n. Αυτό που προτείνουμε είναι η συνάρτηση

$$\alpha_\mu=\frac{1}{1+||\rho\mu-\rho||2\ ||(\rho\mu-\rho)*n||2}.\quad \alpha_\mu\in[0,1] \quad \textbf{(9)}$$

Θεωρούμε ότι $\alpha_\mu\in[0,1]$ θέλοντας να έχουμε μια αριθμητική σταθερά θεωρούμε ότι $\alpha_\mu=1$ όταν $\rho \in \boldsymbol{C_N}$ το ρ στην περίπτωση αυτή θεωρείτε μέρος του νέφους σημείων η ότι βρίσκεται στην επιφάνεια του.



Σχήμα(13) Διαφορά τιμής του α από την (8) και την (9) εξίσωση.

7.3.5 Ορίζοντας το προβαλλόμενο διάνυσμα

Θα πρέπει εδώ να σημειωθεί ότι το geometric pipeline αξιοποιώντας τους αλγορίθμους προβολής και το viewport μετατρέπει τις συντεταγμένες του σημείου στο παράθυρο του υπολογιστή σε συντεταγμένες οθόνης, θα πρέπει να υπάρχει αυτή η αντιστροφή για να υπάρχει αλληλεπίδραση με το περιβάλλον. Άρα υπάρχει ένας αλγόριθμος που μετατρέπει τις παγκόσμιες συντεταγμένες σε συντεταγμένες οθόνης για να μπορούμε να δούμε όλο το αντικείμενο και πως αυτό θα πρέπει να το μετασχηματίσουμε για να πετύχουμε αυτό που θέλουμε.

7.4 Η πρόταση για αλγόριθμο προβολής σημείου.

Στο σημείο αυτό θα αναφερθούμε στον αλγόριθμο προβολής Azariadis & Sapidis [1] σημείου πάνω στο νέφος σημείων, η διαδικασία αυτή ονομάζεται «σημείο προβολής» και όπως έχουμε προαναφέρει θα πρέπει να ορίσουμε μια ελάχιστη απόσταση που να θεωρεί την προβολή που εκτελείται σωστή ή όχι.

Procedure **Point Projection** ($\hat{p}$, C_N, p*, α, t);
Input:
$\hat{p}$=(p,n) // το σημείο που θα προβληθεί
$C_N \epsilon R^{Nx3}$ // το δοσμένο νέφος σημείων
Output:
p* // το προβαλλόμενο σημείο
$t\epsilon$ R // η σχετική λύση που προκύπτει από την εξ.(3)
Local variables:
K // ο αριθμός των επαναλήψεων
$c_n \subseteq C_N$ // βοηθητικό νέφος
$n \epsilon N^+$ // αριθμός των σημείων του c_n
$c_{temp} \subseteq C_N$ // ένα σετ σημείων προσωρινό
$\alpha \epsilon R$ // το διάνυσμα για τα βάρη
$\alpha_{mean} \epsilon R$ // το mean σύμβολο για τα βάρη
$\alpha_{max} \epsilon R$ // το max σύμβολο για τα βάρη
$\alpha_{limit} \epsilon R$ // το min σύμβολο για τα βάρη

Begin:
K:=0;
$c_n := C_N$;
while (K++<MAX_ITERS)
  Begin
    OptimProjectToCloud ($\hat{p}$, c_n, p*,α, t);
     If (||p-p*||<ε) then return;
    Compute local variables α_{mean} , α_{max} , α_{limit} ;
    If (α_{max} ==1) then return ;
    C_{temp}= Ø ;
    for (i=0;i<n;i++) do
        if ($\alpha_\mu \geq \alpha_{limit}$) then $c_{temp} := c_{temp} + c_n(i)$;
    p:=p* ;
    $c_n := c_{temp}$;
  End;
End

Ο αλγόριθμος παίρνει ως input ένα σημείο $\hat{p}$ και ένα νέφος σημείων C_N και υπολογίζει την προβολή ρ* του $\hat{p}$ στο νέφος σημείων C_N. Αυτό επιτυγχάνεται μέσω μιας επαναληπτικής διαδικασίας και με τη βοήθεια ενός συμπληρωματικού τοπικού νέφους c_n στο C_N.
Αρχικά $c_n = C_N$ τα δυο νέφη δηλ. τα θεωρούμε όμοια. Στην πορεία όμως της προβολής ο βασικός κορμός δηλ. ο αριθμός των c_n σημείων ελαττώνεται σταδιακά καταργώντας ουσιαστικά από το νέφος C_N τα ασήμαντα σημεία που δεν χρειάζονται στο υπολογισμό της προβολής. με αυτόν τον τρόπο μας δίνεται η δυνατότητα να επιτύχουμε δυο στόχους.
Α) Να ελαττώσουμε τον χρόνο επεξεργασίας των στοιχειωδών σημείων για την προβολή.
Β) Να αυξήσουμε ακόμα πιο πολύ την ακρίβεια με την όποια θα γίνει η επιλογή του κατάλληλου σημείου.
Έτσι ο ρόλος και η σημασία του κάθε σημείου στο νέφος c_n εξαρτάται από το αντίστοιχο βάρος α_μ που υπολογίζεται από την συνάρτηση (9) όπως έχουμε προαναφέρει. Έπειτα όλα τα βάρη αποθηκεύονται σένα πινάκα = vector με $\alpha \in R^n$
Θέλοντας να αποφασίσουμε ποια είναι τα βαρύ αυτά που θα αγνοηθούν στην K επανάληψη, είμαστε αναγκασμένοι να ορίσουμε άλλη μια μεταβλητή που συμβολίζεται με α_{limit} και ορίζεται

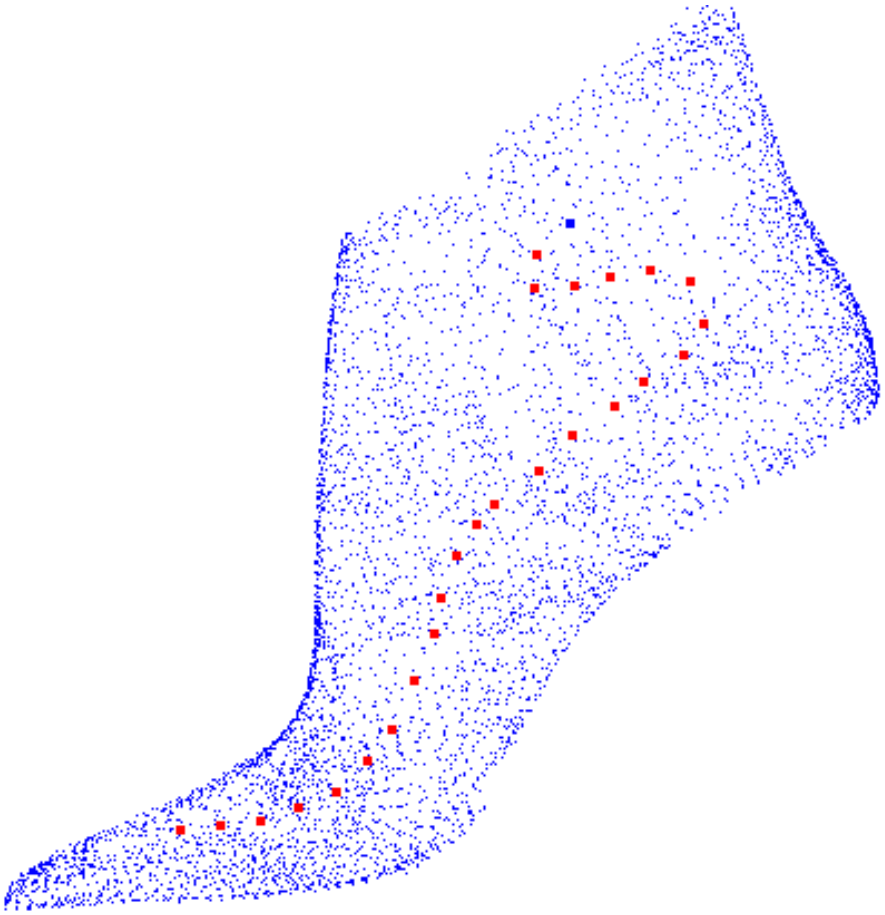
$$\alpha_{limit} = \begin{cases} \alpha_{mean} + \frac{a_{max} - a_{mean}}{10-K}, & K < 9 \\ a_{mean} + \frac{a_{max} - a_{mean}}{2}, & otherwise \end{cases} \quad \textbf{(10)}$$

Θα πρέπει σ'αυτό το σημείο να προσθέσουμε ότι οι παράμετροι α_{man} και α_{min} αντιπροσωπεύουν το μέγιστο και το ελάχιστο αντίστοιχα από τα υπολογισμένα βαρύ (α_{μ}). Όποτε η συνάρτηση (10) ορίζει τον τρόπο με τον όποιον θα γίνεται η επιλογή των σημείων που θα χρειαστούν στον υπολογισμό και αυτόματα απορρίπτονται τα σημεία με την μικροί επιρροή στον αλγόριθμο από την πρώτη κι ολας επανάληψη. Η μέθοδος αυτή όπως μπορούμε να καταλάβουμε είναι πολύ χρήσιμη και εξυπηρετεί σε παρά πολύ μεγάλο βαθμό περιπτώσεις όπως αυτή που έχουμε να υπολογίσουμε χιλιάδες σημεία για να καταλήξουμε σε κάποια εξ αυτών.
Αν ρίξουμε και μια μάτια τώρα στον αλγόριθμο της Point Projection. Θα δούμε ότι ο προτεινόμενος αλγόριθμος αρχίζει θεωρώντας ως ίσον και επεξεργάσιμο νέφος σημείων το $c_N = \mathcal{C}_N$. Έπειτα από αυτό ένας επαναληπτικός βρόχος αρχίζει και το δεδομένο σημείο $\hat{p}$ μαζί με το τρέχον νέφος σημείων c_n να περνάνε στην OptimProjectToCloud για επεξεργασία δίνοντας μας το ρ* το οποίο θα αποτελεί και το ποιο σωστό σημείο για το προβαλλόμενο $\hat{p}$ στο νέφος c_n.

Procedure **OptimProjectToCloud** ($\hat{p}$,c_n,p*,α,t);
Input:
$\hat{p}$=(p,n) // το σημείο που θα προβληθεί
$c_n \subseteq \mathcal{C}_N$ // βοηθητικό νέφος
$n \in N^+$ // αριθμός σημείων στην c_n
Output:
p* // το βέλτιστο σημείο για την $\hat{p}$
$t \in R$ // αντίστοιχη στην εξ.(3)
$\alpha \in R^n$ // βάρος σημείων
Begin
Υπολογίζει τα βάρη α μέσο της εξ.(9)
Υπολογίζει την p* χρησιμοποιώντας εξ.(3) και εξ.(2)
End

Επίσης OptimProjectToCloud επιστρέφει το βάρος α συμφώνα με το input σημείο $\hat{p}$. Σαυτό το σημείο θα πρέπει να συμφωνήσουμε ότι εάν η Ευκλείδεια απόσταση μεταξύ του επιλεγμένου σημείου ρ* και του σημείου $\hat{p}$ είναι μικρότερη από κάποιο error (ε) που έχουμε θέση, τότε ο αλγόριθμος τερματίζει. Σ'αυτήν την περίπτωση το σημείο $\hat{p}$ μετακινείται στο επόμενο επιλεγμένο σημείο από τον αλγόριθμο ρ* όποτε ($\hat{p}$=ρ*) και ένας νέος επαναληπτικός κύκλος αρχίζει επαναπροσδιορίζοντας τα βάρη που είναι μικρότερα από το α_{limit} τα όποια μετακινούνται από το c_n για τους λογούς που έχουμε προαναφέρει.

Τέλος εάν $\alpha_{max}=1=\alpha_k$, ο αλγόριθμος τερματίζει και επιστρέφει ως σημείο προβολής το σημείο νέφους δηλ. ρ_k. Στην μέθοδο αυτή καταλήξαμε μετά από αρκετά πειράματα τα όποια αποδείκνυαν την ακρίβεια και αυτονομία του συγκεκριμένου αλγορίθμου που προτείνουμε.



Σχήμα(14) Ενδεικτικό νέφη σημείων (Point Cloud) με προβαλλόμενα σημεία στην επιφάνεια

7.5 Τεστάροντας διάφορα νέφη

7.5.1 Τεστάροντας την επίδραση σε λεπτό και παχύ νέφος

Πειράματα που έγιναν από Azariadis & Sapidis [1] για την ακρίβεια με την όποια γίνεται η προβολή σένα λεπτό η παχύ νέφος σημείων μας δείχνουν ότι όντως η πυκνότητα επηρεάζει διάφορους παράγοντες που έχουν άμεση σχέση με τον αλγόριθμο προβολής.

Στόχος είναι να βρεθεί ποτέ το προβαλλόμενο σημείο θα μπορέσει να ακουμπήσει στην επιφάνεια του δοσμένου σε συγκεκριμένο πάχος και αριθμό σημείων. Επίσης θα πρέπει η αυτονομία του αλγορίθμου να μας δώσει αμετάβλητα αποτελέσματα με βάση το πάχος του νέφους αυτού. Τα αποτελέσματα που έχουμε για ένα λεπτό νέφος σημείων είναι τα εξής:

Α) Η ακρίβεια της προβολής (με την $R_{i,N}$) είναι πολύ ικανοποιητική. Επίσης το εύρος της τιμής $R_{i,N}$ είναι 0,05mm ακόμα και όταν το N είναι σχετικά μικρό.

Β) Παρόλο που το δεδομένο λάθος είναι πολύ μικρό error (ε=10-6) η ανταπόκριση με τον αλγόριθμο είναι γρήγορη, στην χειρότερη χρειάζονται 7 επαναλήψεις.

Γ) Ο χρόνος εκτέλεσης του αλγορίθμου είναι σχετικά γρήγορος παρά την έλλειψη «τεχνικών επιτάχυνσης»

Δ) Γενικά όταν ο αριθμός των σημείων αυξάνεται το αντίστοιχο error προβολής ελαττώνεται. Παράλου που μερικές ελάχιστες αποκλίσεις έχουν παρατηρηθεί (για i=2,3), αυτά ευθύνονται για δυο λογούς. Για τον υπολογισμό των $\rho^{S}_{i,N}$ και $\rho^{*}_{i,N}$ όπως επίσης για την επιλογή του τελικού βοηθητικού νέφους.

Έτσι γνωρίζουμε ότι αλλαγές στο βοηθητικό νέφος υπονοούν αλλαγές στην εξ.(4) το όποιο με την σειρά του μπορεί να γυρίσει το αποτέλεσμα σε διαφορετικές λύσεις.

Τα αποτελέσματα που έχουμε από την άλλη για ένα παχύ νέφος σημείων είναι τα εξής:

Α) Παρόλο που το περιθώριο σφάλματος ε=10-6μμ είναι πολύ μικρό το convergence του αλγορίθμου είναι γρήγορο σένα εύρος μικρότερο των 14 επαναλήψεων που χρειάζονται.

Β) Το υπολογισμένο λάθος $R_{i,p}$ είναι μικρότερο από ρ, υπονοώντας ότι ο προτεινόμενος αλγόριθμος θεωρείτε πιο ακριβείς για επιφάνειες διαφορετικού πάχους.

Γ) Θα πρέπει να δώσουμε προσοχή στο γεγονός ότι και στις δυο περιπτώσεις το πάχος του κάθε νέφους δεν είναι σταθερό αλλά μεταβάλλεται μεταξύ [-ρ,ρ]. αυτό εξηγεί γιατί στην περιοχή του σημείου $p^{S}_{i,p}$ το νέφους έχει πάχος μικρότερο από ρ ως αποτέλεσμα σένα $R_{i,p}$ και αντίστροφα.

8.Μετατροπη Καμπυλών σε Smooth Curve

Στο σημείο αυτό θα επικεντρωθούμε στην κατασκευή ασυνεχών (digital καμπυλών σε δοσμένη digital επιφάνεια). Το παράδειγμα που θα αναλύσουμε βασίζεται στην επιφάνεια νέφους καθώς και στην λεία καμπύλη που θα προσπαθήσουμε να προσεγγίσουμε μέσα από ένα αριθμό σημείων. Δίνοντας ως input κάποια σημεία $\hat{p} = (p_i, n_i)$ το output είναι είτε γραμμικό είτε είναι μια καμπύλη κάποιου βαθμού που θα παρεμβάλει η θα προσεγγίζει τα σημεία ρ_i* από $\hat{p}_i$ στο C_N. Ο αλγόριθμος προβολής είναι αυτός που παραμένει ο πυρήνας της προβολής σημείου στο νέφος , τώρα όμως υπολογίζεται από μεθόδους καθορισμού της διευθύνσεις προβολής όπως περιγράφεται στις εργασίες [2,3].

Αξιοποιώντας πλέον τον αλγόριθμο προβολής μπορούμε να δώσουμε στον χρηστή την δυνατότητα να μπορεί να σχεδιάζει επιθυμητές καμπύλες σε οποιοδήποτε νέφος σημείων. Καθώς όμως και τα νέφη είναι καμπυλοειδή επιφάνειες συχνά οι γραμμές και οι καμπύλες μπορεί να μην εφάπτονται στην επιφάνεια και αυτό είναι ένα πρόβλημα που πρέπει να λυθεί.

8.1 Αρχικές Καμπύλες Παρεμβολής

Η προβολή ενός ευθύγραμμου τμήματος q_i πάνω σένα νέφος σημείων μετατρέπει το q_i σε μια πολυγραμμή (polyline) q_i*. Η μετατροπή αυτή συμβαίνει λαμβάνοντας υπόψη την ελαχιστοποίηση του (1), πολλοί και διάφοροι μεταβλητές μπορεί να εμφανιστούν στα q_i* το οποίο θα χρησιμοποιηθεί για την ελάττωση των πτυχώσεων κατά την προβολή [1] .

Ένας τρόπος για την κατασκευή Smooth προβόλων q_i* πάνω σε νέφη σημείων C_N είναι να ενσωματώσουμε στην διαδικασία προβολής τον περιορισμό για το ελάχιστο μήκους q_i* που προκύπτει από την προβολή. Με αυτόν τον τρόπο μπορούμε να ελαττώσουμε τις πτυχές με “stretching” στο παραγόμενο p-polyline. Αντίστοιχες μέθοδοι έχουν χρησιμοποιηθεί και από άλλους ερευνητές για επίλυση σχετικού προβλήματος. Όπως ο Felderman [14], ο όποιος θεώρησε μια 2Δ πολυγραμμή (polyline) και παρέθεσε μια διαδικασία διευθετήσεις των σημείων μένα δοσμένο εύρος ώστε το συνολικό της πολυγραμμής (polyline) να ελαχιστοποιείται. Επιπλέον οι ασυνεχείς αυτές

αντιλήψεις χρησιμοποιηθήκαν από πολλούς άλλους ερευνητές για να προσεγγίσουν με σημεία 2D ή 3D χώρους [12,22]. Παρόλα αυτά κανένας δεν κατάφερε να επίλυση το πρόβλημα του ομαλού πολύγραμμου (smoothing polyline) πάνω σένα νέφος σημείων, όπως θα το αντιμετωπίσουμε σ'αυτήν την διπλωματική.

Με βάση όλα όσα έχουμε προαναφέρει υπάρχουν δυο τρόποι για να επιτευχθεί η ομαλοποίηση που θέλουμε μέσω ενός αλγορίθμου που ομαλοποιεί τις καμπύλες.

Smoothing methodology A. Ενσωματώνοντας μια smoothing μέθοδο αν και αυτή η προσέγγιση δεν είναι απόλυτα ενδεδειγμένη (όπως έχουν αποδείξει και διαφορές έρευνες) καθώς υπολογίζει τα προβαλλόμενα σημεία διαδοχικά που κάτι τέτοιο μπορεί να υιοθετηθεί μόνο για τοπικό "Smoothing Technique" βελτιστοποιώντας κάποια σημεία σε σχέση με αυτά που έχουν ήδη υπολογίσει.

Smoothing methodology B. Έχοντας ως δοσμένο το ευθύγραμμο τμήμα q_i το οποίο καθορίζεται από το σετ $\hat{p}_i = \{\hat{p}_{i,k} = (\hat{p}_{i,k}, n_{i,k}) \mid k=0,\ldots k(i)-1\}$ των k(i) για τα σαφή σημεία nodes $p_{i,k}$ η ομαλή προβολή του q_i πάνω στο C_N είναι ένα p-polyline $q_i^* = (p^*_{i,k})$ που ελαχιστοποιεί την τιμή της συναρτήσεις,

$$\mathbf{E_i = (1-\gamma)P_i + \gamma L_i} \quad \gamma \in [0,1] \quad \textbf{(11)} \text{ οπου}$$
$$\mathbf{P_i} = \sum_{\mathbf{k=i}}^{\mathbf{k(i)-1}} \mathbf{E}\,(\mathbf{p^*_{i,k}}) \quad \textbf{(12)} \text{ και}$$
$$\mathbf{L_i} = \sum_{\mathbf{k=0}}^{\mathbf{k(i)-2}} \| \mathbf{p^*_{i,k}} - \mathbf{p^*_{i,k+1}} \|^2 \quad \textbf{(13)}$$

Το οποίο εκφράζει και το μήκος του ρ-segment q_i^*.

Κάθε $p^*_{i,k}$ ορίζεται από το $\mathbf{p^*_{i,k} = p_{i,k} + t_k n_{i,k}}$ (σε σύγκριση με την (2) όλο το ρ polyline q_i^* ορίζεται από το διάνυσμα $\mathbf{t = t_k}$ **(κ=1,....κ(i)-1)** διατηρώντας την άγνωστη παράμετρο t_k. Η συνάρτηση (11) είναι ένας συνδυασμός από 2 συναρτήσεις όπου ελαχιστοποιούνται όταν η παράμετρος **t** είναι λύση του συστήματος

$$\mathbf{[(1-\gamma)\, I + \gamma A]\, t = (1-\gamma)\, b + \gamma c} \quad \textbf{(14)} \quad \text{τότε έχω}$$
$$\mathbf{t = t(\gamma) = [(1-\gamma)\, I + \gamma A]^{-1}\, [(1-\gamma)\, b + \gamma c]} \quad \textbf{(15)}$$

το όποιο είναι μια δευτέρου βαθμού εξίσωση με άγνωστο το γ. Α είναι (κ(i)-2) χ (κ(i)-2) ένας τρεις διαγώνιος συμμετρικός πινάκας. Τα τρία μη μηδενικά στοιχειά της κ_{th} γραμμής του Α είναι

$$\mathbf{A_{\kappa,\kappa-1} = -n_{i,k-1}\, n_{i,k}} \quad \mathbf{A_{\kappa,\kappa} = 2} \quad \text{για } \kappa=2,\ldots,\kappa(\text{ι})-3, \quad \mathbf{A_{\kappa,\kappa+1} = -n_{i,k}\, n_{i,k+1}} \quad \textbf{(16)} \text{ με}$$
$$\mathbf{A_{1.1} = 2} \quad \mathbf{A_{1,2} = -n_{i,1}\, n_{i,2}} \quad \mathbf{A_{\kappa(\text{ι})-2,\kappa(\text{ι})-3} = -n_{i,k-2}\, n_{i,k-3}} \quad \mathbf{A_{\kappa(\text{ι})-2,\kappa(\text{ι})-2} = 2} \quad \textbf{(17)}$$

το κth στοιχείο του διανύσματος c δίνεται από

$$\mathbf{c_k = (p_{i,k-1} - p_{i,k})\, n_{i,k} + (p_{i,k} + p_{i,k+1})\, n_{i,k}} \text{ για } \kappa=1,\ldots\kappa(\text{ι})-1$$

Καθώς το κth στοιχειό του διανύσματος b είναι

$$\mathbf{b} = \frac{\boldsymbol{\lambda}_\kappa - \mathbf{p_{i,k} n_{i,k}}}{\|\mathbf{n_{i,k}}\|^2} \text{ για } \kappa=1,\ \kappa(\text{ι})-1 \text{ όπου } \boldsymbol{\lambda}_\kappa = \frac{\mathbf{c^1_{i,k} n^x_{i,k} + c^2_{i,k} n^y_{i,k} + c^3_{i,k} n^z_{i,k}}}{\mathbf{c'_0}} \quad \textbf{(18)}$$

και τα $c^0_{i,k}$, $c^1_{i,k}$, $c^2_{i,k}$, $c^3_{i,k}$ ορίζονται από την εξίσωση (4) για κάθε $\hat{p}_{i,k}$ και I είναι (κ(i)-2) x (κ(i)-2) μοναδιαίος πινάκας. Στην περίπτωση μας ένα ομαλό παραγόμενο polyline με μεγάλη απόκλιση από το νέφος σημείων θα σχεδιαστεί με μια μεγάλη τιμή για το γ σε σχέση με αυτό που θα σχεδιαστεί από μια μικρή τιμή στην μεταβλητή γ.

Για γ=0 επιτυγχάνουμε μεγάλη ακρίβεια στην προβολή αλλά πιθανότατα θα έχουμε και ρυτίδες (κυματισμούς) στην απεικόνιση που θα έχουμε, ενώ για γ=1 επιτυγχάνουμε ευθείες γραμμές που συνδέονται με τα σημεία $\rho_{i,0}$ και $\rho_{i,k(i)-1}$.

Στην Εικόνα(6) έχουμε ένα ψευδοκώδικα των παραπάνω προβάλουν την smooth προβολή. Με αυτόν τον τρόπο υπολογίζουμε τους σχετικούς πινάκες και διανύσματα από εξισώσεις (16,17,18) και με την βοήθεια του LU Decomposition και με αναφορές σε άλλες προηγούμενες εξισώσεις επιτυγχάνουμε την επίλυση της συναρτήσεις (15). Αυτό με την σειρά του μας δίνει την κρίσιμη θέση των κόμβων (nodes) $\rho^*_{i,k}$ για την p- polyline q_i^*.

Μετά από επαναλαμβανόμενες δόκιμες, βλέπουμε ότι η πολυγραμμή καμπύλη αγκάλιαζε και ακούμπησε πολύ απαλά το νέφος του παπουτσιού μας (χωρίς κυματισμούς και ρυτίδες). Για τα περισσότερα πειράματα θεωρήσαμε το γ=0,5 το όποιο πιστεύουμε ότι είναι ένα καλό μέγεθος, ανάμεσα στον κυματισμό και την ακρίβεια.

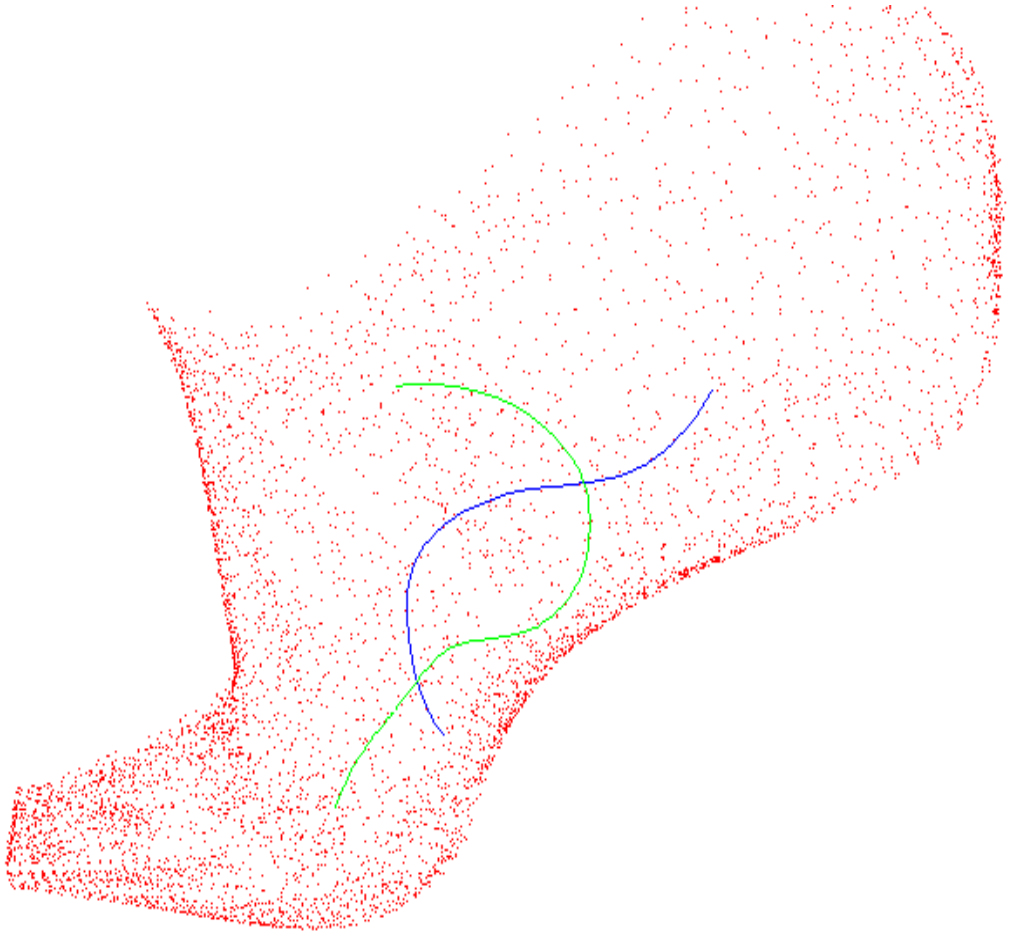
Procedure **SmoothSegProjection** ($\hat{p}_I$, C_N, γ, $p^*_{i,k}$)
Input:
$\hat{p}_I$ είναι το d-segment που θα προβληθεί από το q_i
$C_N \in R^{Nx3}$ το δοσμένο νέφος σημείων
$\boldsymbol{\gamma} \in [0,1]$ συντελεστής ομαλότητας
Output:
$p^*_{i,k}$ τα σημεία του p-segment q^*_i
Τοπικοί μεταβλητές
$c \in R^{k(i)-2}$ το διάνυσμα που υπολογίζεται από την εξ.(20)
$b \in R^{k(i)-2}$ το διάνυσμα που υπολογίζεται από την εξ.(21)
$t \in R^{k(i)-2}$ το αποτέλεσμα του υπολογίζεται από την εξ.(17)
Begin
//compute solution, από τον τύπο της εξ.(16) και υπολόγιζε την λύση του t από την εξ.(17)
//compute projection $p^*_{i,k}$
for (κ=1;κ<κ(i)-1;κ++)
$p^*_{i,k} = p_{i,k} + t * n_{i,k}$;
End

Σχεδιάζοντας “digital Spline” πάνω σε νέφη σημείων.

Καθώς οι πολύγραμμες καμπύλες polylines είναι αρκετά εύκαμπτα και μπορούν να κατασκευαστούν σχετικά εύκολα οι designers χρειάζονται και προτιμούν smooth καμπύλες τα όποια στην εφαρμογή μας παρουσιάζονται σαν καμπύλες Spline χαμηλού βαθμού. Στα υπάρχοντα CAD συστήματα η χάραξη καμπυλών σε πολυωνυμικές επιφάνειες δεν είναι μια ευθεία διαδικασία όπως πιστεύουν ή θεωρούν οι περισσότεροι χρηστές CAD συστημάτων κ καμπυλών σε δοσμένες επιφάνειες[12]. Μπορούμε να δούμε πλέων ότι η σχεδίαση καμπυλών σε νέφη σημείων δεν είναι τόσο πολύπλοκη διαδικασία όσο φαίνεται σε διάφορα συστήματα. Η μέθοδος που εφαρμόζουμε για την σχεδίαση πολυωνύμικων καμπυλών Spline πάνω σε νέφη σημείων χρησιμοποιώντας παρεμβολή όπως αναφέρεται στο [27,9.3.4] για την κατασκευή ομαλής Smooth Spline ξ-βαθμού παρεμβάλλοντας τα $\rho^{\xi}(u)$ σημεία που ορίζει ο χρήστης από το d-polyline.

Χρησιμοποιώντας ίδιους μεθόδους είναι εφικτό να υπολογίσουμε την αντίστοιχη διεύθυνση για $n^{\xi}(u)$. Ένα σετ $\hat{p}^{\xi}$ από κόμβους που παράγεται από το $\rho^{\xi}(u)$ το οποίο τελικά προβάλλεται στην επιφάνεια του νέφους. Το τελικό αποτέλεσμα είναι ένα ομαλό polyline που ακουμπά επάνω στο νέφος. Η διακριτοποίηση για το $\rho^{\xi}(u)$ μπορεί να επετεύχθη χρησιμοποιώντας την μέθοδο που αναφέρεται στην [35], το οποίο ενισχύει το γεγονός ότι η τοπολογία του αποτελέσματος της γραμμικής παρεμβολής είναι σταθερό και ίδιο με την αρχική καμπύλη.

Προβολή ρ^{ξ} πάνω σένα νέφος σημείων είναι μια straightforward εφαρμογή δυο αλγορίθμων του Point Projection και του Smooth Projection καθορίζοντας ότι τουλάχιστον δυο boundary σημεία από τα ρ^{ξ} θα πρέπει να είναι σταθερά.



Σχήμα(15) Ενδεικτική Smooth καμπύλη πάνω στο Point Cloud

8.2 Τοπική Παρεμβολή Καμπυλών

Στο σημείο αυτό θα αναλύσουμε τον τρόπο που επιτυγχάνεται η τοπική παρεμβολή καμπυλών από σημεία και εφαπτόμενες. Θα παρουσιαστούν κάποιες βασικές έννοιες που δείχνουν μέσα από μια σειρά εξισώσεων πως φτάνουμε στο επιθυμητό αποτέλεσμα της παρεμβολής.

Τοπική παρεμβολή καμπυλών

Ας θεωρήσουμε δοσμένα σημεία $\{Q_k\}$, k=0,…,n. Με την local curve interpolation εννοούμε μια μέθοδο η όποια κατασκευάζει n polynomial curve segments, $C_i(u)$, i=0,…,n-1, με τα Q_i και Q_{i+1} να είναι τα ακριανά σημεία της C_i.Τα τμήματα που συνορεύουν περιγράφονται με κάποιο βαθμό συνεχείας και η μέθοδο κατασκευής τους το ίδιο. Στο γενικότερο πλαίσιο για την κατασκευή των NURBS, ορίζουμε αρχικά segments χρησιμοποιώντας τα polynomial Bezier curve μετά πετυχαίνουμε τα NURBS επιλέγοντας το κατάλληλο knot διάνυσμα[27,9][35].

Ας υποθέσουμε ότι η u_i ορίζει ως παραμετρική αρχή το $C_i(u)$ και ως τέλος $C_{i-1}(u)$. $C_i(u)$ και $C_{i-1}(u)$ συναντιούνται στο u_i με G^1 συνεχεία αν και μόνο αν οι εφαπτόμενες τους θεωρήσουμε ότι συμπίπτουν εκεί. Παρόλο που η G^1 συνεχεία θεωρείτε ως μια ομαλή συνεχεία μπορεί να έχουμε προβλήματα στην παραμετροποίηση. Για περισσότερα στοιχειά για την συνεχεία σε σημείο είναι πιο σωστό να μελετηθεί ο Barsky DeRose υπάρχουν βεβαία αλγόριθμοι για να παρέχουν μεγαλύτερο του 3 βαθμού συνεχείας όμως σε αυτήν την εφαρμογή θα ασχοληθούμε με συνεχεία μέχρι τρίτου βαθμού.

Παρατηρώντας τα Bezier segments, $C_i(u)$, απαιτείτε να υπολογίσουμε αρχικά τα Bezier control points, ένα σημείο έλεγχου για την τετραγωνική Bezier και δυο σημεία για την κυβική. Αυτά τα σημεία έλεγχου βρίσκονται πάνω σε τμήματα που σχηματίζονται από την εφαπτομένη στο Q_k, γι'αυτό τον λόγο χρειαζόμαστε το εφαπτομενικό διάνυσμα T_k σε κάθε Q_k μερικές φορές μπορεί να υπολογιστή εύκολα από την Q_k. Όμως το σωστό είναι να αποτελεί μέρος του αλγορίθμου παρεμβολής. Υπάρχουν μια σειρά μέθοδων, ο Boehm δίνει μια ιδέα για την μέθοδο του. Ας υποθέσουμε ότι $\Delta u_k = u_k - u_{k-1}$ $q = Q_k - Q_{k-1}$ $d = \frac{q_k}{\Delta u_k}$

όλες οι μέθοδοι έχουν μια από τις δυο αυτές φόρμες.

$D_k = (1- \alpha_k)d_k + \alpha_k d_{k+1}$ (19) ή

$T_k = \frac{V_k}{|V_k|}$ $\quad$ **$V_k = (1- \alpha_k)q_k + \alpha_k q_{k+1}$ (20)**

Παρατηρούμε ότι στην (19) θεωρούμε ότι η τιμή για το u_k έχει υπολογιστεί. Το D_k μπορεί να εκτιμηθεί από την παράγωγο. Η συνάρτηση όμως (20) δεν χρησιμοποίει την παράμετρο u_k , και το αποτέλεσμα του διανύσματος πρέπει να ληφθεί ως εφαπτόμενη διανύσματος κατευθύνσεις μόνο. Η απόδοση του βαθμού και των παραμέτρων πρέπει να γίνεται σε συνδυασμό το ένα με το άλλο και όχι ως κάτι ανεξάρτητο. Σημειώνεται ότι το T χρησιμοποιείται μόνο ως μέγεθος για το μοναδιαίο διάνυσμα των εφαπτόμενων. Οι αλλαγές που υπάρχουν οφείλονται στον τρόπο που υπολογίζεται τα α_κ . κανονικά εξαρτάται από τα τρία ή πέντε γειτονικά σημεία. Για παράδειγμα Bessel χρησιμοποιεί μια μέθοδο τριών σημείων σύμφωνα και με τον τύπο.

$\alpha_\kappa = \frac{\Delta u_k}{\Delta u_k + \Delta u_{k+1}}$ k=1,…,n-1 (21) μαζί με την (19) έχουμε Q_i

$\alpha_\kappa = \frac{|q_{k-1} \times q_k|}{|q_{k-1} \times q_k| + |q_{k+1} \times q_{k+2}|}$ k=2,…,n-2 **(22)**

μαζι με την εξίσωση (20) μας δίνουν μια μέθοδο για πέντε σημεία υπολογισμού του $T\alpha_\kappa$ [Akim70;Renn82;Pieg87d]. Έχει το πλεονέκτημα ότι τρία διαδοχικά σημεία Q_{k-1} Q_k Q_{k+1} αποδίδουν το T_k το οποίο είναι παράλληλο στο line segment. Ο παρανομαστής της (22) εξαφανίζεται εάν Q_{k-2} Q_{k-1} Q_k είναι collinear και το Q_k Q_{k+1} Q_{k+2} είναι collinear επίσης. Αυτό υπονοεί είτε ότι είτε θα έχουμε γωνία στο Q_k είτε μια ευθεία γραμμή από το Q_{k-2} στο Q_{k+2} . Σ'αυτήν την περίπτωση το α_κ μπορεί να ορίσει σένα σύνολο από wavs: εμείς επιλέγουμε $\alpha_\kappa = 1$ το οποίο σημαίνει $V_\kappa = q_{\kappa+1}$ αυτό μας δίνει γωνία στο $\alpha_\kappa = ½$, το οποίο σημαίνει ότι $V_\kappa = ½ (q_\kappa + q_{\kappa+1})$ αυτό δίνει μια πιο ομαλή γωνία. Βασισμένοι στα παραπάνω, μια τοπική γραμμική παρεμβολή ρουτίνας μπορεί να δεχτεί μια flag υποδεικνύοντας που ή όχι υπάρχουν γωνίες. Όλοι οι μέθοδοι χρησιμοποιούν ειδικούς τύπους για το τα άκρα.

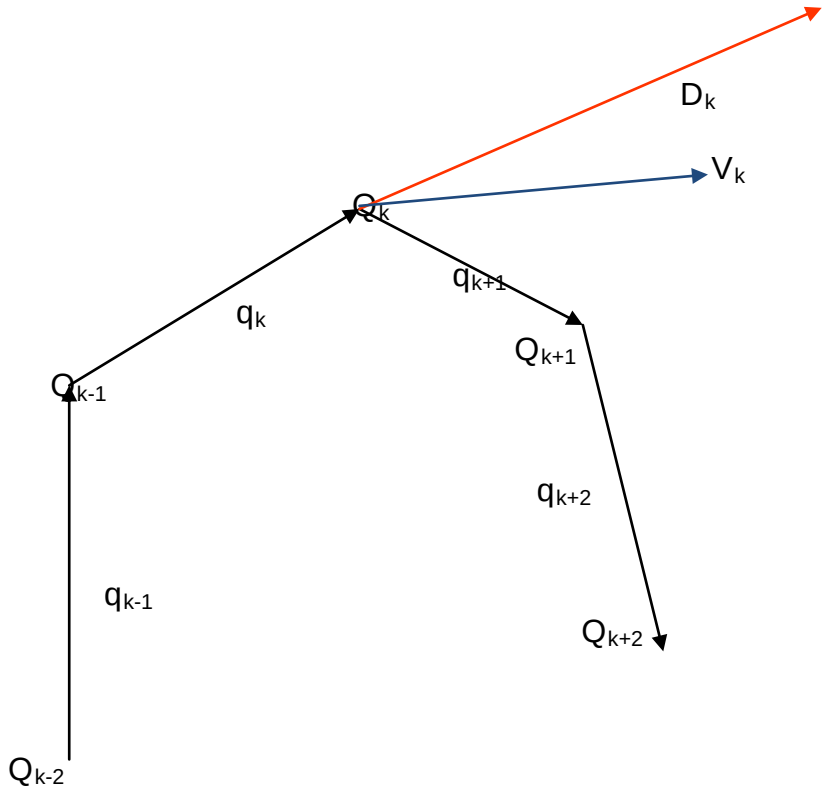
Για μια παρεμβολή μεταξύ τριών σημείων έχουμε:

$D_0 = 2d_1 - D_1$ $\quad$ **$D_n = 2d_n - D_{n-1}$ (23)**

Ενώ για μια παρεμβολή πέντε σημείων έχω:

$q_0 = 2q_1 - q_2$ $\quad$ **$q_{-1} = 2q_0 - q_1$** $\quad$ **$q_{n+1} = 2q_n - q_{n-1}$** $\quad$ **$q_{n+2} = 2q_{n+1} - q_n$ (24)**

με αντικατάσταση στα (22) και (20) προκύπτουν τα T_0 T_1 και T_{n-1} T_n.



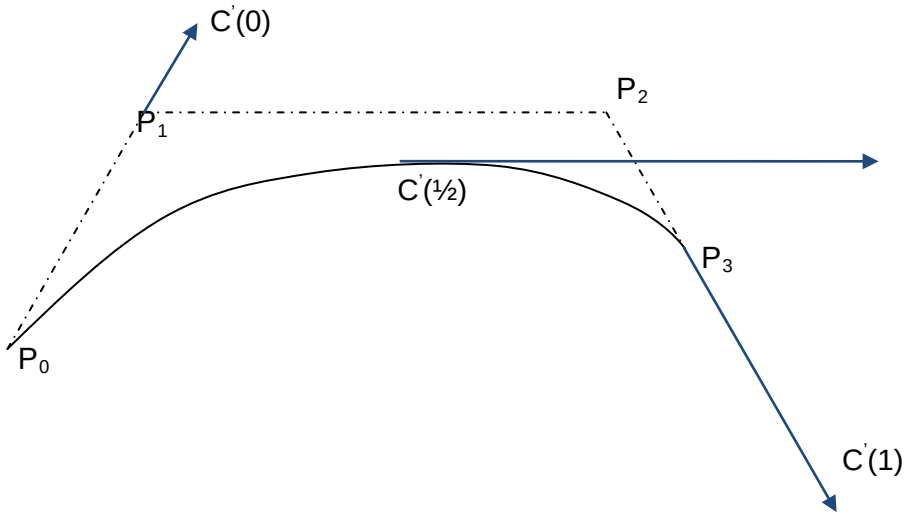
Σχήμα(16) Υπολογίζουμε την εφαπτομένη (V_k) και την παράγωγο (D_k) για την τοπική παρεμβολή.

8.3 Κυβικές Καμπύλες Παρεμβολής

Οι κυβικές καμπύλες Bezier μπορούν εύκολα να μετασχηματιστούν σε 3 διαστάσεις.
Ας υποθέσουμε ότι P_0 και P_3 τα δυο άκρα μιας καμπύλης και T_0 και T_3 οι αντίστοιχες μοναδιαίες εφαπτόμενες. Με αυτά τα δεδομένα είναι δυνατόν να κατασκευάσουμε μια κυβική καμπύλη Bezier C(u) u∈[0,1] με σημεία και εφαπτόμενες που έχουμε και να ισχύει ότι:

$\alpha = |C'(0)| = |C'(1/2)| = |C'(1)|$ (25) όπως επίσης για τα ενδιάμεσα σημεία ισχύει:
$P_1 = P_0 + 1/3\ \alpha T_0$ $P_2 = P_3 - 1/3\ \alpha T_3$ (26)

Εφαρμόζοντας τον αλγόριθμο του deCasteljau για u = ½ έχουμε P = C(1/2).



Σχήμα(17). Μια κυβική καμπύλη Bezier με εφαπτόμενες στις παραμέτρους (0, 1/2, 1).

Με αποτέλεσμα να έχουμε και

$\mathbf{P_1^2 - P_0^3 = 1/8\ (P_3 + P_2 - P_1 - P_0)}$ **(27)**

Εφαρμόζοντας την εξίσωση (1.10) έχουμε
$\mathbf{C'\ (1/2) = 6\ (P_1^2 - P_0^3)}$ **(28)**

Θεωρώντας το α ίσον και αντικαθιστώντας την (27) στην (28) και χρησιμοποιώντας την (26) παίρνουμε ότι,

$\mathbf{8/6\ \alpha = |P_3 + P_2 - P_1 - P_0| = |P_3 + (P_3 - 1/3\ \alpha T_3) - (P_0 + 1/3\ \alpha T_0) - P_0| \Rightarrow}$
$\mathbf{16\ \alpha^2 = \alpha^2|T_0 + T_3|^2 - 12\alpha\ (P_3 - P_0)\ (T_0 - T_3) + 36|P_3 - P_0|^2 \Rightarrow}$
$\mathbf{\alpha\alpha^2 + b\alpha + c = 0}$ **(29)** οπού
$\mathbf{\alpha = 16 - |T_0 + T_3|^2}$ $\mathbf{b = 12\ (P_3 - P_0)\ (T_0 - T_3)}$ $\mathbf{c = 36|P_3 - P_0|^2}$

Η εξίσωση (29) έχει δυο πραγματικές λύσεις για το α, μια θετική και μια αρνητική. Αντικαθιστώντας την θετική λύση στην συνάρτηση (26) προκύπτουν τα επιθυμητά P_1 και P_2.
Τώρα για την κατασκευή της καμπύλης ας υποθέσουμε ότι $\{Q_k\}$, k = 0,…,n ένα σετ από τρισδιάστατα data points.
Εάν το διάνυσμα τις εφαπτόμενης δεν δίνεται τότε το υπολογίζουμε από (20, 22, 24). Το να κατασκευάσουμε μια καμπύλη Bezier $C_k(u)$ μεταξύ των Q_k και Q_{k+1}. Τα σημεία έλεγχου που προκύπτουν από την Bezier θα είναι

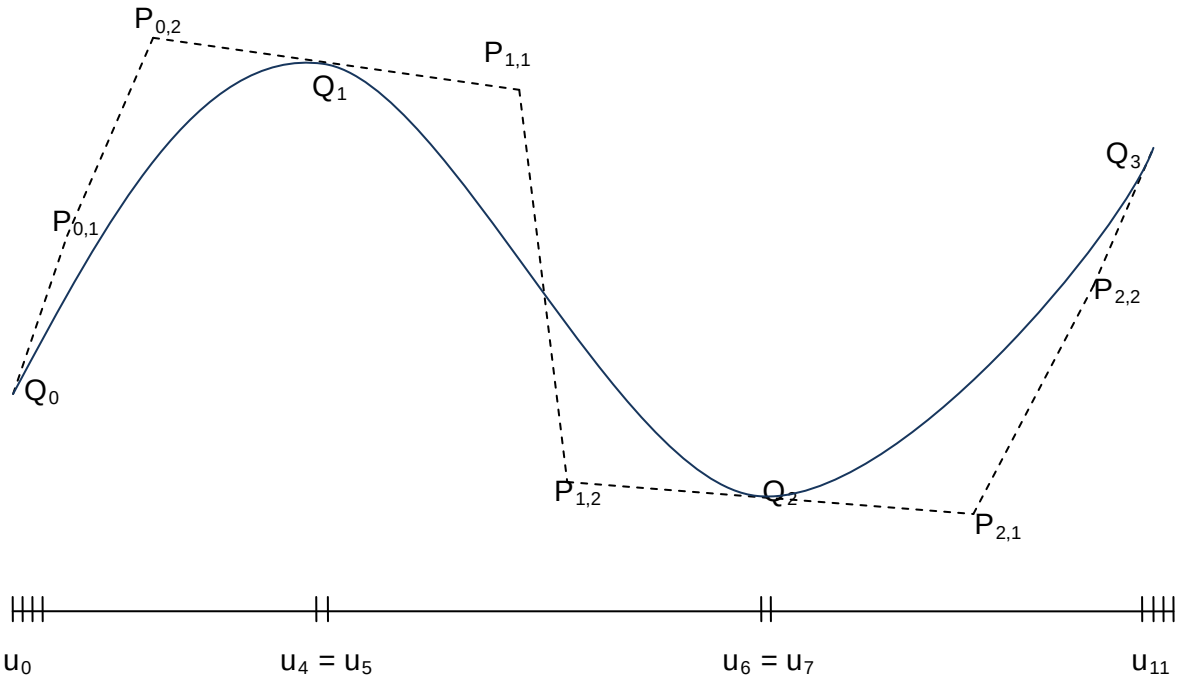
$\mathbf{P_{\kappa,0} = Q_k}$ $\mathbf{P_{\kappa,1}}$ $\mathbf{P_{\kappa,2}}$ $\mathbf{P_{\kappa,3} = Q_{k+1}}$ **(30)**

Πρέπει να ορίσουμε κατάλληλα τα $P_{\kappa,1}$ $P_{\kappa,2}$ για τα κατάλληλα T_κ και $T_{\kappa+1}$ αντίστοιχα.
Είναι πολύ πιθανόν έτσι να έχουμε μια ικανοποιητική συνεχεία C^1 για να πετύχουμε μια καλή προσέγγιση στην παραμετροποίηση που έχουμε. Η μέθοδος καθορίζεται από τον Renner [Renn82]. Ορίζοντας τώρα το $u_0 = 0$, και για κ=0,…,n-1, το $u_{\kappa+1}$ και τα δυο πρώτα σημεία έλεγχου του $C_k(u)$ υπολογίζονται όπως πιο κάτω:

1. Χρησιμοποιούμε την εξ.(29) για να υπολογίσουμε το α και την (26) για να υπολογίσουμε την $P_{\kappa,1}$ $P_{\kappa,2}$
2. Ορίζουμε την $\mathbf{u_{0+1} = u_\kappa + 3\ |P_{\kappa,1} - P_{\kappa,0}|}$ **(32)**

Αυτός ο αλγόριθμος παράγει n Bezier segments, το κάθε ένα από τα όποια είναι σταθερά σε κάποιους παραμέτρους. Γι'αυτό και C^1 συνεχεία στα κυβικά B-Spline καμπύλες παρεμβολής έχει εξ ορισμού την Q_k που ορίζεται από τα σημεία έλεγχου.

$Q_0, P_{0,1}, P_{0,2}, P_{1,1}, P_{1,2}, \ldots, P_{n-2,2}, P_{n-1,1}, P_{n-1,2}, Q_k$ **(33)** και οι κομβόι έχουν
$U = \{0, 0, 0, 0, \frac{u_1}{u_n}, \frac{u_1}{u_n}, \frac{u_2}{u_n}, \frac{u_2}{u_n}, \ldots, \frac{u_{n-1}}{u_n}, \frac{u_{n-1}}{u_n}, 1, 1, 1, 1\}$ **(34)**



Σχήμα(18) Κυβική καμπύλη παρεμβολής C^1.

Ανάλυση βασικών σχεδιαστικών εργαλείων.

Στην παράγραφο αυτή θα αναπτυχθούν κάποιες από τις βασικότερες σχεδιαστικές εντολές και ταυτόχρονα κάποια εργαλεία της εφαρμογής μαζί με τους ψευδοκώδικες που τις υλοποιούν. Πρόκειται για τις εντολή Point Projection και την OptimProjectToCloud που αποτελούν τον πυρήνα και τον κορμό της εφαρμογής αντίστοιχα, όπως έχει ήδη αναφερθεί. Στη συνέχεια ακολουθούν η Curve Estimation, η Project Curve, η Offset Curve, η Clone Curve, η Circle_και τέλος η Trim Curves. Οι εντολές αυτές θα αναλυθούν και πιο εκτενεστέρα με τους αλγορίθμους τους. Στη συνέχεια ακολουθεί η περιγραφή της γενικής λειτουργίας της κάθε εντολής.

9. Λογισμική υλοποίηση εργαλείων

9.1 PointProjection

Η εντολή Point Projection είναι η πιο βασική εντολή που έχουμε σχεδιάσει στη συγκεκριμένη εφαρμογή. Είναι κατά βάση ο πυρήνας της εφαρμογής αυτής. Τα πάντα στηρίζονται στην αρχή που η εντολή αυτή υλοποιεί. Η εντολή Point Projection είναι αυτή που λαμβάνει υπόψη όλες τις classes που έχουν υλοποιηθεί στην εφαρμογή. Στη συνέχεια ελέγχει το σφάλμα error της συνάρτησης και από εκεί αρχίζει η διαδικασία τις ρουτίνας του αλγορίθμου να εκτελείται.

Έτσι δημιουργούμε τις συνθήκες για την έναρξη της διαδικασίας που ενεργοποιεί τον αλγόριθμο ο οποίος δέχεται ως input όπως θα δούμε μια σειρά δεδομένων ,υπολογίζει κάποιες αποστάσεις και στην πορεία μετά από έλεγχο των κριτηρίων που έχουμε ορίσει, λαμβάνουμε το ορθότερο αποτέλεσμα από ένα σύνολο πράξεων που συντελούν στον αλγόριθμο προβολής.

PointProjection

Procedure **PointProjection**($\hat{p}$,C_N,p*,t);
Input:
$\hat{p}$=(p,n) // το σημείο που θα προβληθεί
$C_N \epsilon R^{Nx3}$ // το δοσμένο νέφος σημείων
Output:
p* // το προβεβλημένο σημείο
$t \epsilon$ R // η σχετική λύση που προκύπτει από την εξ.(3)
Local variables:
K // ο αριθμός των επαναλήψεων
$c_n \subseteq C_N$ // βοηθητικό νέφος
$n \epsilon N^+$ // αριθμός των σημείων του c_n
$c_{temp} \subseteq C_N$ // ένα σετ σημείων προσωρινό
$\alpha \epsilon$ R // το διάνυσμα για τα βάρη
$\alpha_{mean} \epsilon$ R // το mean σύμβολο για τα βάρη
$\alpha_{max} \epsilon$ R // το max σύμβολο για τα βάρη
$\alpha_{limit} \epsilon$ R // το min σύμβολο για τα βάρη

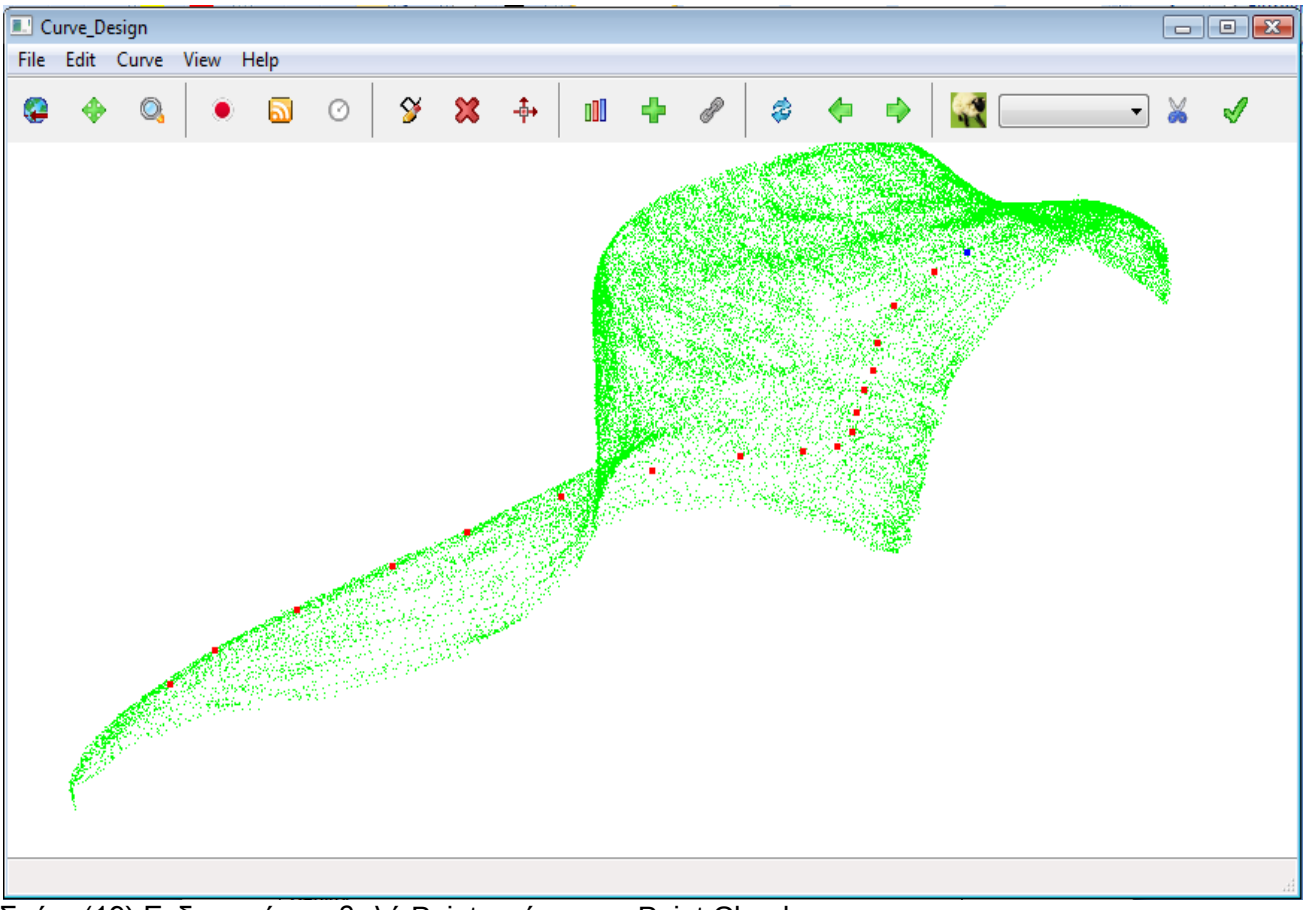
Begin:
K:=0;
$c_n := C_N$;
while (K++<MAX_ITERS)
Begin
OptimProjectToCloud ($\hat{p}$, c_n,p*,α,t);
If (||p-p*||<ε) then return;
Compute local variables α_{mean} , α_{max} , α_{limit} ;
If (α_{max} ==1) then return ;
C_{temp}= ∅ ;
for (i=0;i<n;i++) do
if ($\alpha_\mu \geq \alpha_{limit}$) then $c_{temp} := c_{temp} + c_n(i)$;
p:=p* ;
$c_n := c_{temp}$;
End;
End

9.2 OptimProjectToCloud

Η εντολή OptimProjectToCloud είναι επίσης βασική εντολή της εφαρμογής. Η εντολή αυτή είναι υπεύθυνη για τη σωστή προβολή του σημείου καθώς και την τελική θέση που ενδέχεται να πάρει το σημείο αυτό πάνω στο νέφος. Οπότε εμείς έχουμε την εντολή OptimProjectToCloud που λαμβάνει μια σειρά από μεταβλητές για να μας δώσει το βέλτιστο τελικό σημείο μαζί με τα σωστά βάρη που κάθε σημείο περιλαμβάνει. Με αυτό τον τρόπο έχουμε την καλύτερη δυνατή θέση για το σημείο σε σχέση με το νέφος.

OptimProjectToCloud

Procedure **OptimProjectToCloud** ($\hat{p}$,c_n,p*,α,t);
Input:
$\hat{p}$=(p,n) // το σημείο που θα προβληθεί
$c_n \subseteq C_N$ // βοηθητικό νέφος
$n \in N^+$ // αριθμός σημείων στην c_n
Output:
p* // το βέλτιστο σημείο για την $\hat{p}$
$t \in R$ // αντίστοιχη στην εξ.(3)
$\alpha \in R^n$ // βάρος σημείων
Begin
Υπολογίζει τα βάρη α μέσο της εξ.(9)
Υπολογίζει την p* χρησιμοποιώντας εξ.(3) και εξ.(2)
End



Σχήμα(19) Ενδεικτική προβολή Points πάνω στο Point Cloud

9.3 Curve Estimation

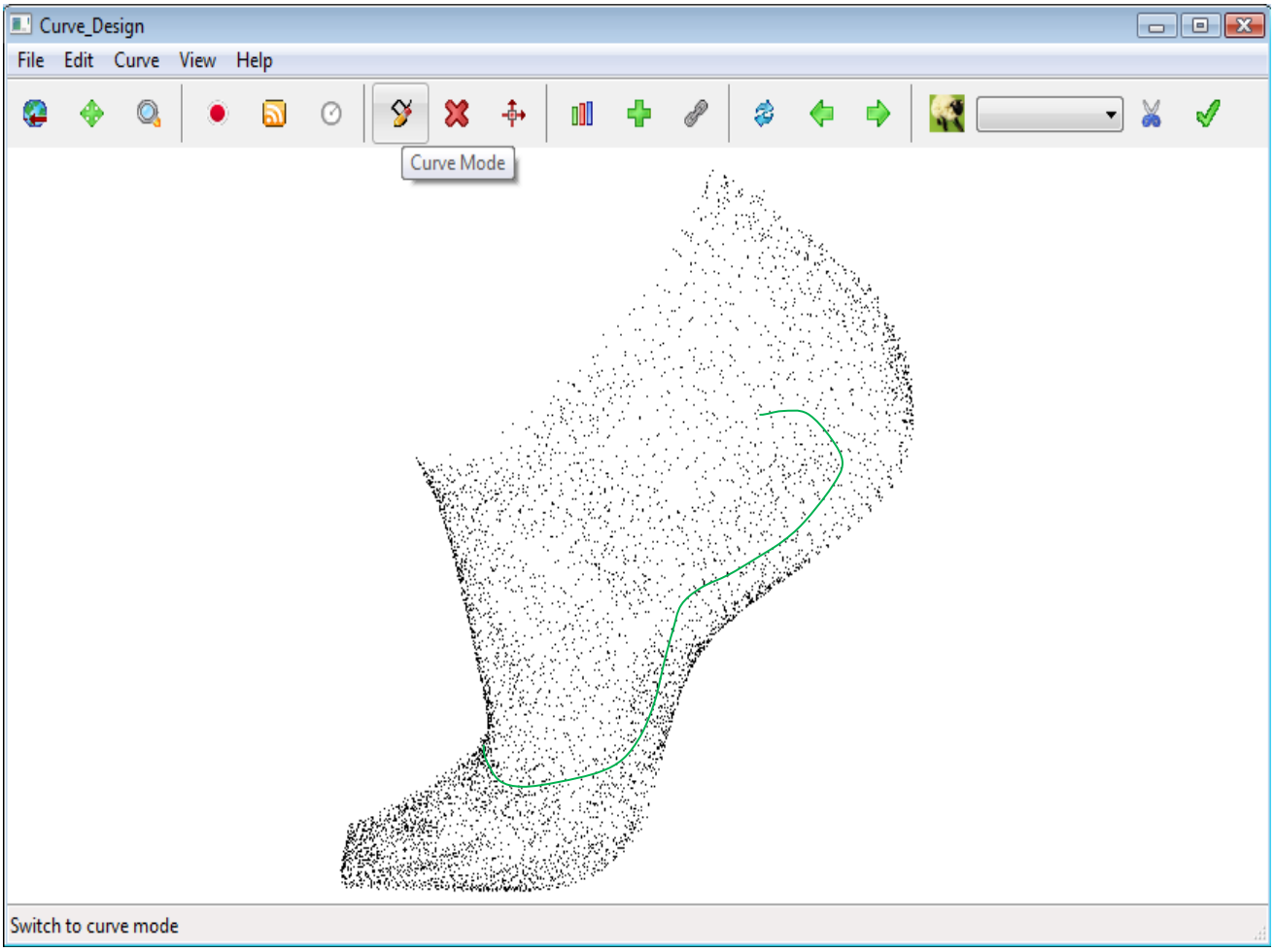
Μια άλλη εξίσου σημαντική εντολή στην εφαρμογή που υλοποιήθηκε είναι η Curve Estimation. Η εντολή αυτή διαδραματίζει εξίσου βασικό ρολό με τις εντολές που προαναφέρθηκαν. Θα πρέπει να σημειώσουμε ότι ο ρόλος της εντολής Curve Estimation είναι να σχηματίζει τις καμπύλες τρίτου βαθμού Bezier που προκύπτουν ύστερα από τα σημεία που έχει δώσει ο χρήστης.

CurveEstimation

CurveEstimation(p_i,cb_i)
Input :
p_i : Σημεία που έχουν προβληθεί στο νέφος και συνιστούν δείγματα για την κατασκευή καμπύλη

Output :
cb_i : Υπολογισμένες καμπύλες Bezier $3^{ου}$ βαθμού

Τοπικές Μεταβλητές :
T_i : Εφαπτόμενα διανύσματα στα σημεία p_i (εξ.[19],[20],[21],[22])
Begin
Υπολόγισε τα T_i από εξίσωση(εξ.[20])
for(k=0; k<size(p_i)-1; k++)
{
$t = T_k + T_{k+1}$;
$p = p_{i,k+1} – p_{i,k}$;
$a = 16 – t^2$;
$b = 12 * p \cdot t$;
$c = p^2 – 36$;
$r = (-b \pm sqrt(b^2 – 4ac)) / (2*a)$;
$cp_a = p_{i,k} + T_k * r * 1/3$;
$cp_b = p_{i,k+1} - T_{k+1} * r * 1/3$;
$cb_{i,k}$ = CubicBezier ($p_{i,k}$, cp_a , cp_b , $p_{i,k+1}$);
}
End



Σχήμα(20) Ενδεικτική μετατροπή των Points σε Curves

9.4 Project Curve /SmoothSegProjection

Στη συνέχεια ακολουθεί η εντολή Project Curve, η οποία ευθύνεται για την προβολή της καμπύλης που έχει οριστεί. Εδώ μπορούμε να συμπεριλάβουμε και την SmoothSegProjection,η οποία με τεχνικές εξομαλύνσεις που έχουμε ορίσει, οδηγεί σε ένα πιο ομαλό αποτέλεσμα στην καμπύλη που θα προκύψει.

Project Curve

Project Curve(cb_i, C_N, γ, ds_i)
Input :
cb_i : Η καμπύλη προς προβολή
C_N : Το νέφος σημείων
γ : Παράμετρος εξομάλυνσης

Output :
ds_i : Ακολουθία σημείων που ορίζουν την προβεβλημένη καμπύλη στο νέφος

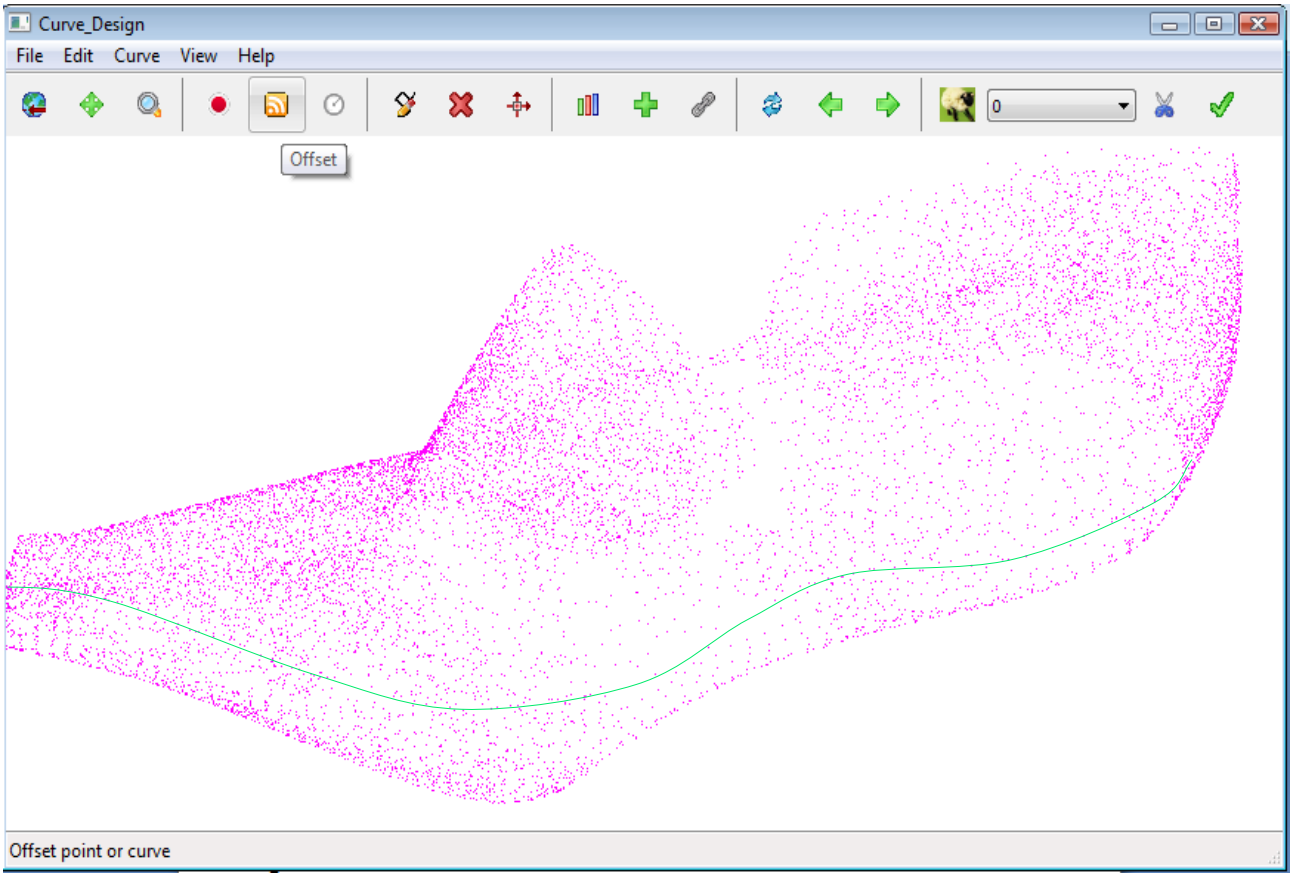
Τοπικές Μεταβλητές :
cbp_i : Ακολουθία σημείων δειγματοληψίας από την καμπύλη cb_i
step : Βήμα δειγματοληψίας (ανάλογο της επιθυμητής πυκνότητας)

Begin
for(t = 0.0 ; t≤1.0 ; t+=step)
{
bp = Bezier Point(cb_i, t); Από Εξίσωση(eq.[3])
Add(cbp_i, bp);
$cbp = (1-t)^3 P_0 + 3(1-t)^2 tP_1 + 3(1-t) t^2 P_2 + t^3 P_3$
}
SmoothSegProjection(cbp_i, C_N, γ, ds_i)
End

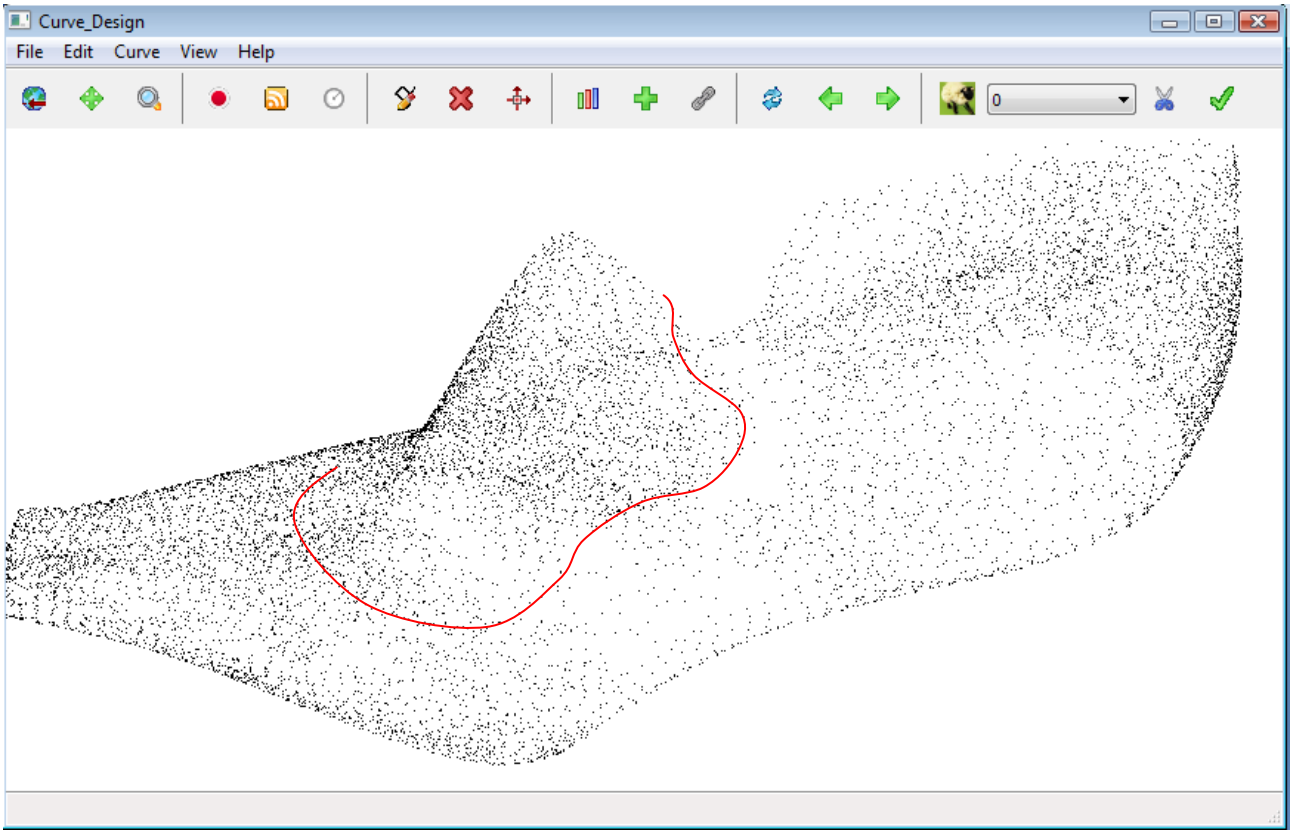
SmoothSegProjection

SmoothSegProjection ($\hat{p}_I$, C_N, γ, $p^*_{i,k}$)
Input:
$\hat{p}_I$ είναι το d-segment που θα προβληθεί από το q_i
$C_N \in R^{Nx3}$ το δοσμένο νέφος σημείων
$\gamma \in [0,1]$ συντελεστής ομαλότητας
Output:
$p^*_{i,k}$ τα σημεία του p-segment q^*_i

Τοπικοί μεταβλητές:
$c \in R^{k(i)-2}$ το διάνυσμα που υπολογίζεται από την εξ.(20)
$b \in R^{k(i)-2}$ το διάνυσμα που υπολογίζεται από την εξ.(21)
$t \in R^{k(i)-2}$ το αποτέλεσμα του υπολογίζεται από την εξ.(17)
Begin
//compute solution, από τον τύπο της εξ.(16) και υπολόγιζε την λύση του t από την εξ.(17)
//compute projection $p^*_{i,k}$
for (κ=1;κ<κ(i)-1;κ++)
$p^*_{i,k} = p_{i,k} + t * n_{i,k}$;
End



Σχήμα(21) Ενδεικτική εικόνα Project Curve



Σχήμα(22) Ενδεικτική εικόνα SmoothSegProjection

9.6 Offset Curve

Η Offset Curve είναι μια εντολή που τη συναντάμε σχεδόν σε όλες τις σχεδιαστικές εφαρμογές και θεωρείται πολύ βασική γιατί είναι συνήθως απαραίτητη η μεταφορά κάποιου στοιχείου που έχουμε σχεδιάσει. Βέβαια στην εφαρμογή αυτή η λειτουργία της Offset Curve διαφοροποιείται σε σχέση με άλλα σχεδιαστικά προγράμματα. Αυτό συμβαίνει γιατί την απόσταση της μετακίνησης της αρχικής (μητρικής) καμπύλης ο χρήστης-σχεδιαστής την ορίζει μεν, ωστόσο, όχι με αριθμητικά δεδομένα αλλά κατά εκτίμηση για το που αυτός θεωρεί ότι είναι η πιο κατάλληλη θέση πάνω στο νέφος σημείων. Οπότε η Offset Curve παίρνει την αρχική καμπύλη και την προβάλει ξανά σε κάποιο νέο σημείο που ο σχεδιαστής θα ορίσει.

Offset Curve

Offset Curve(p)
Input :
cIn : Η καμπύλη εισόδου
p: Το σημείο στο οποίο θα μετακινηθεί το κέντρο της καμπύλης

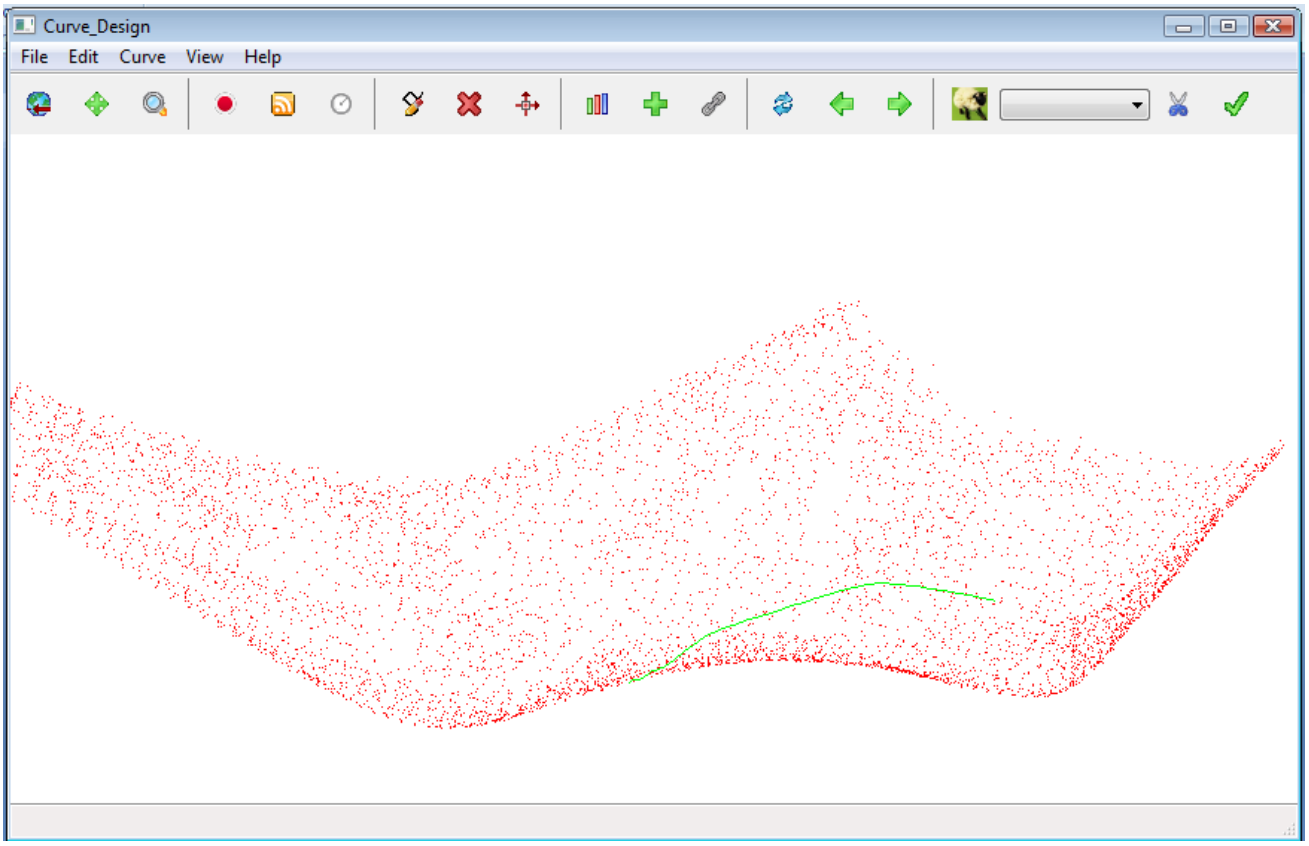
Output :
cOut : Η καμπύλη εξόδου
dsOut : Η ακολουθία των προβεβλημένων σημείων της νέας καμπύλης

Τοπικές Μεταβλητές :
C : Το κέντρο της καμπύλης εισόδου
V : Το διάνυσμα μεταφοράς

Begin
C = (0,0,0);
for(every p ∈ Cout)
{
totalv++
C += r + p;
}

C *= 1/totalv;
V = p - center;

cOut = cIn.Move(V);
cIn.Erase();
ProjectCurve(cOut, C_N, γ, dsOut);
Add cOut to curve display list.
End



Σχήμα(23) Ενδεικτική εικόνα για της Offset Curve

9.7 Clone Curve

Η Clone Curve είναι μια εντολή που από το όνομα της ακόμα αντιλαμβανόμαστε πως πρόκειται για εντολή χρήσιμη για την αντιγραφή καμπύλης. Μια εξίσου βασική εντολή γιατί όπως και η Offset Curve έτσι και η Clone Curve θεωρούνται βασικά εργαλεία για την σχεδίαση κάποιου προϊόντος. Η Clone Curve αντιγραφεί την αρχική (μητρική) καμπύλη, όμως σε αντίθεση με την Offset Curve δεν την διαγράφει αλλά προβάλει ξανά αντίγραφο της σε νέο σημείο που ορίζει ο χρήστης. Έτσι έχουμε ένα πολύ δυνατό σχεδιαστικό εργαλείο για να κάνουμε αυτό που όλοι έχουμε συνηθίσει και ονομάζουμε Copy-Paste.

Clone Curve

Clone Curve(p)
Input :
cIn : Η καμπύλη εισόδου
p : Το σημείο στο οποίο θα εμφανιστεί το κέντρο της κλωνοποιημένης καμπύλης

Output :
cOut : Η καμπύλη εξόδου
dsOut : Η ακολουθία των προβεβλημένων σημείων της νέας καμπύλης

Τοπικές Μεταβλητές :
C : Το κέντρο της καμπύλης εισόδου
V : Το διάνυσμα μεταφοράς

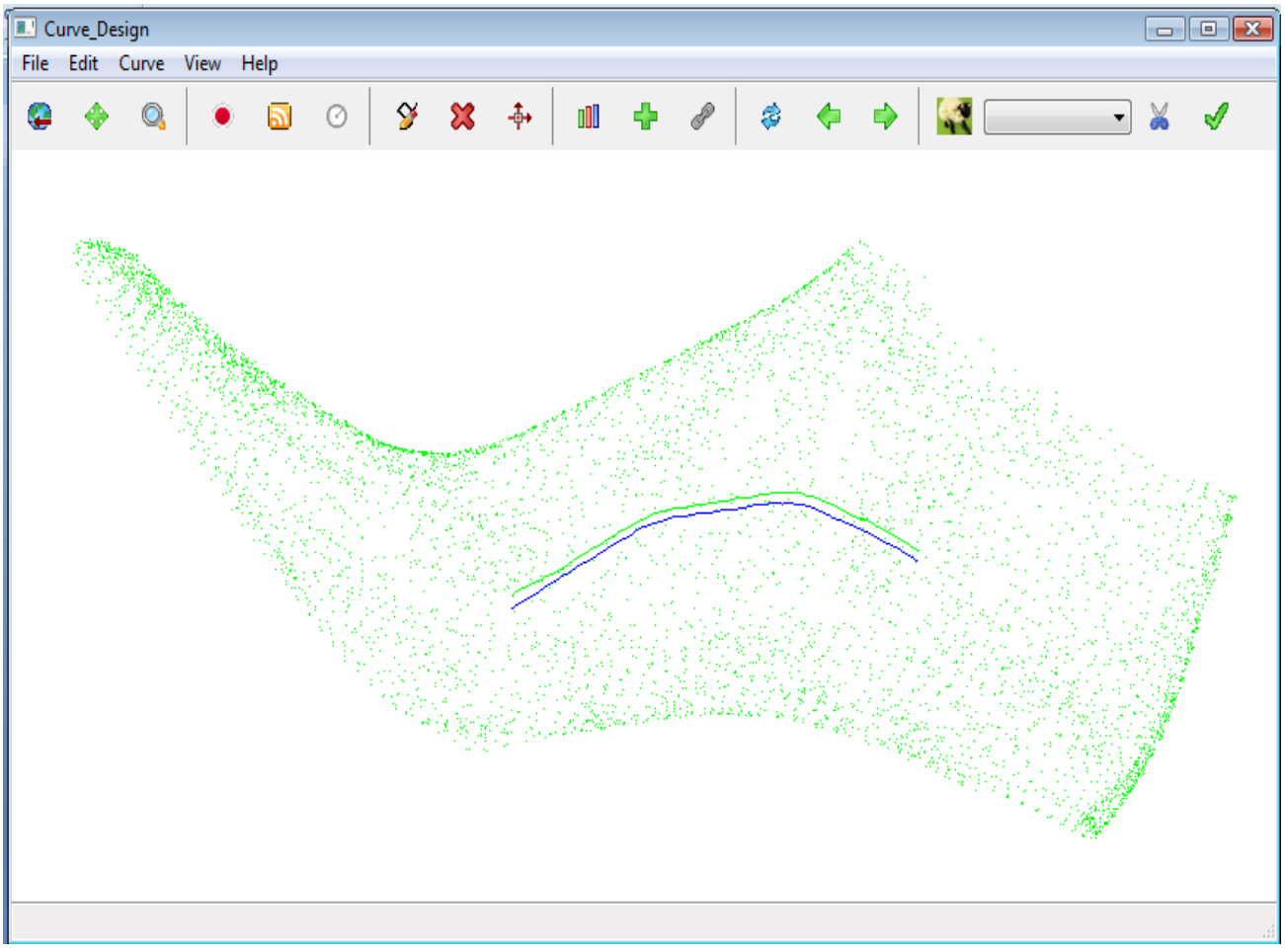
Begin

```
C = (0,0,0);
for(every p ∈ Cout)
{
        totalv++
        C += r + p;
}

C *= 1/totalv;
V  = p - center;

cOut = cIn.Move(V);
ProjectCurve(cOut, C_N, γ, dsOut);
Add cOut to curve display list.
```

End



Σχήμα(24) Ενδεικτική εικόνα για τη Clone Curve

9.8 Circle

Ο κύκλος είναι ένα από τα σχήματα-εργαλεία που η εφαρμογή μπορεί να σχεδιάσει. Για τη σχεδίαση του, ο χρήστης θα πρέπει αρχικά να ορίσει το κέντρο του και έπειτα την ακτίνα που θα επιθυμούσε να έχει ο σχεδιαζόμενος κύκλος. Παρά το γεγονός ότι η σχεδίαση αυτή υστερεί σε μεγάλο βαθμό σε σχέση με άλλα σχεδιαστικά προγράμματα που έχουμε συνηθίσει, θα πρέπει να λάβουμε υπόψη μας το 3D χώρο άρα και τους αντίστοιχους περιορισμούς που υπάρχουν. Ο κύκλος που σχηματίζεται είναι μια ακολουθία από πολλά περιμετρικά σημεία τα οποία προβάλλονται στο νέφος σημείων με βάση τη διαδικασία της προβολής.

Circle

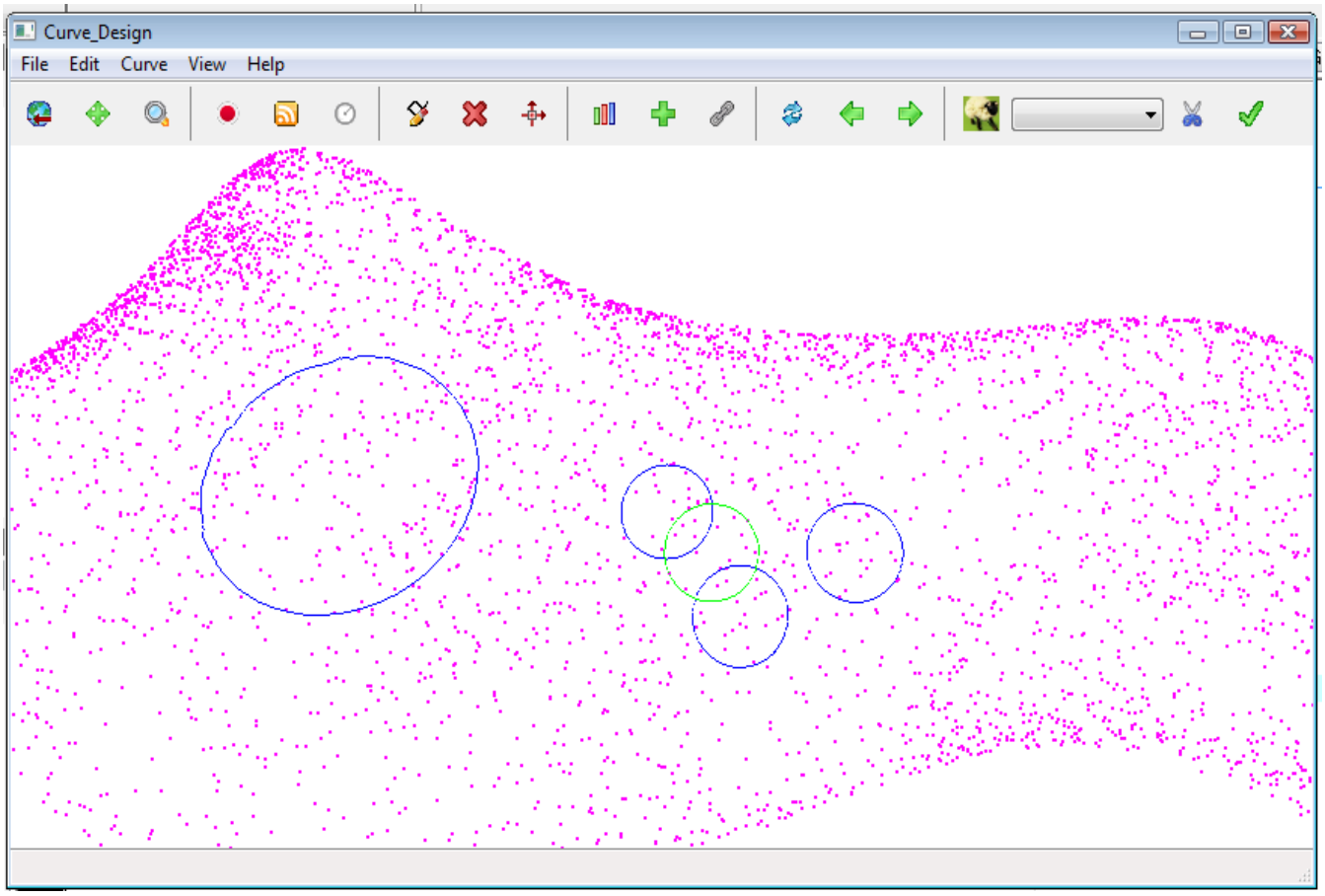
Circle(p, n, r)
Input :
p : Το σημείο που ορίζεται ως κέντρο του κύκλου
n : Το διάνυσμα του παρατηρητή
r : Η ακτίνα του κύκλου
Output :
cOut : Η καμπύλη εξόδου
dsOut : Η ακολουθία των προβεβλημένων σημείων της νέας καμπύλης

Τοπικές Μεταβλητές :
p_i : Η ακολουθία των σημείων στη περίμετρο του κύκλου

Begin
Περιστροφή του p στους άξονες Y, X κατά γωνίες θ, φ ώστε το n να ταυτίζεται με τον άξονα Z+
i = 0
for(angle = 0.0; angle <= 360.0; angle += 5.0)
{
double x = r * cos(PI/180.0 * angle) + p.x;
double y = r * sin(PI/180.0 * angle) + p.y;
double z = p.z;

pTemp = Point(x, y, z);
Αντίστροφή περιστροφή του pTemp κατά φ, θ στους άξονες X, Y
p_{i++} = ProjectToCloud(pTemp, C_N, n);
}

CurveEstimation(p_i, cOut);
ProjectCurve(cOut, C_N. γ, dsOut);
Add cOut to curve display list.
End



Σχήμα(25) Ενδεικτική εικόνα για τις Circle μαζί με Clone Curve

9.9 Trim

Τέλος η εντολή Trim είναι αυτή που θα μας δίνει την τομή που υπάρχει μεταξύ δυο καμπυλών. Η Trim υπολογίζει την τομή που υπάρχει μεταξύ της πρώτης καμπύλης και την τομή που θα προκύψει από τη δεύτερη καμπύλη. Με αυτό τον τρόπο, μετά από υπολογισμούς και πράξεις που γίνονται στον αλγόριθμο καθώς εκτελείται, ο χρήστης αποκτά τη δυνατότητα να διαλέξει μια από τις δυο καμπύλες που έχουν προκύψει μετά το κόψιμο της δεύτερης. Ουσιαστικά, η πρώτη καμπύλη θεωρείται οδηγός και οποιαδήποτε άλλη που την τέμνει μπορεί να χωριστεί από τον οδηγό σε δύο άλλες.

Trim

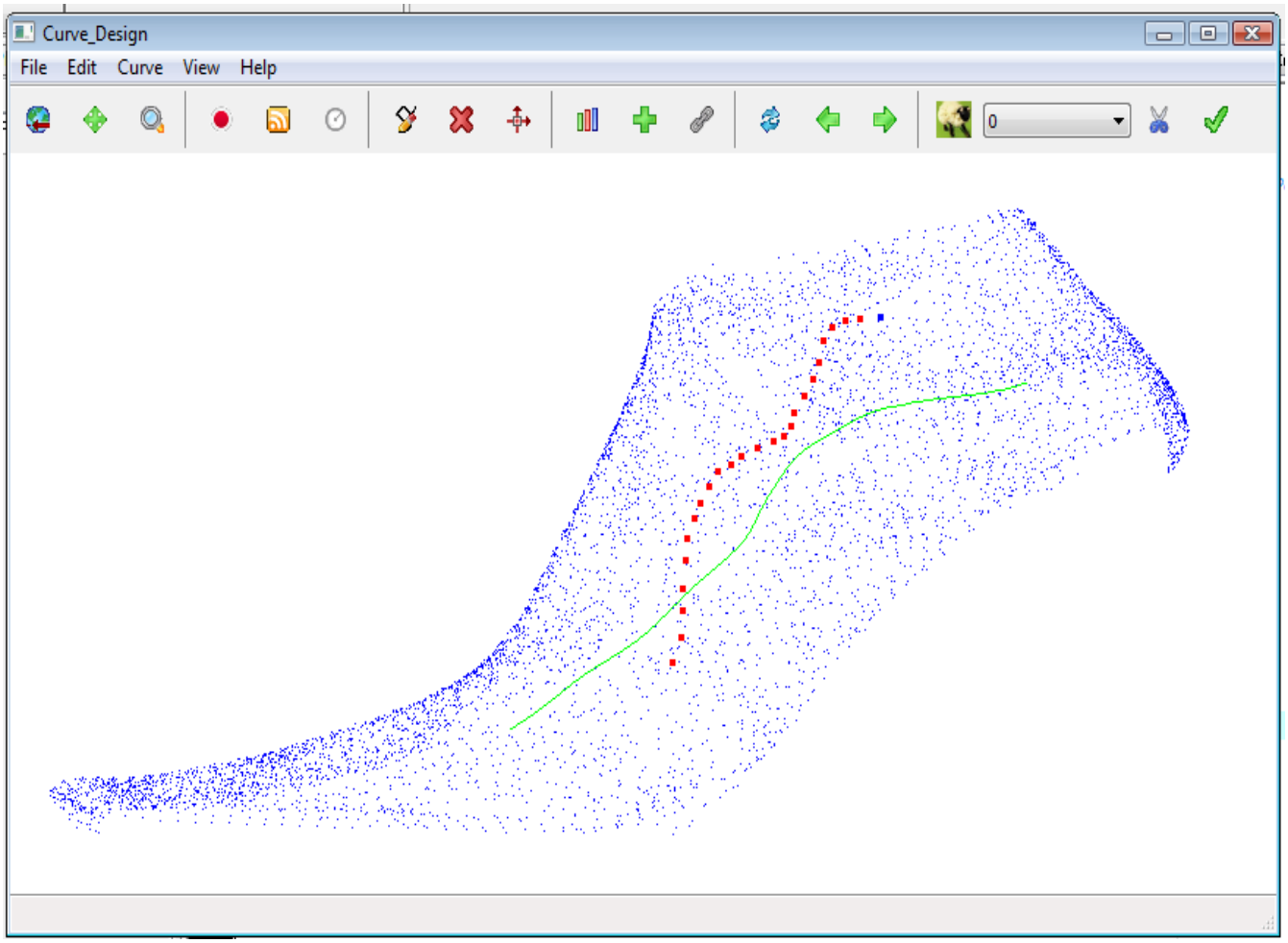
Trim Curves($cb_{i,1}$, $ds_{i,1}$, $cb_{i,2}$, $ds_{i,2}$, C_N, γ, $tcb_{i,1}$, $tds_{i,1}$, $tcb_{i,2}$, $tds_{i,2}$)
Input :
c $b_{i,1/2}$: Οι καμπύλες για trimming
$ds_{i,1/2}$: Οι ακολουθίες τον προβεβλημένων σημείων των αντίστοιχων καμπυλών
C_N : Το νέφος σημείων
γ : Παράμετρος εξομάλυνσης

Output :

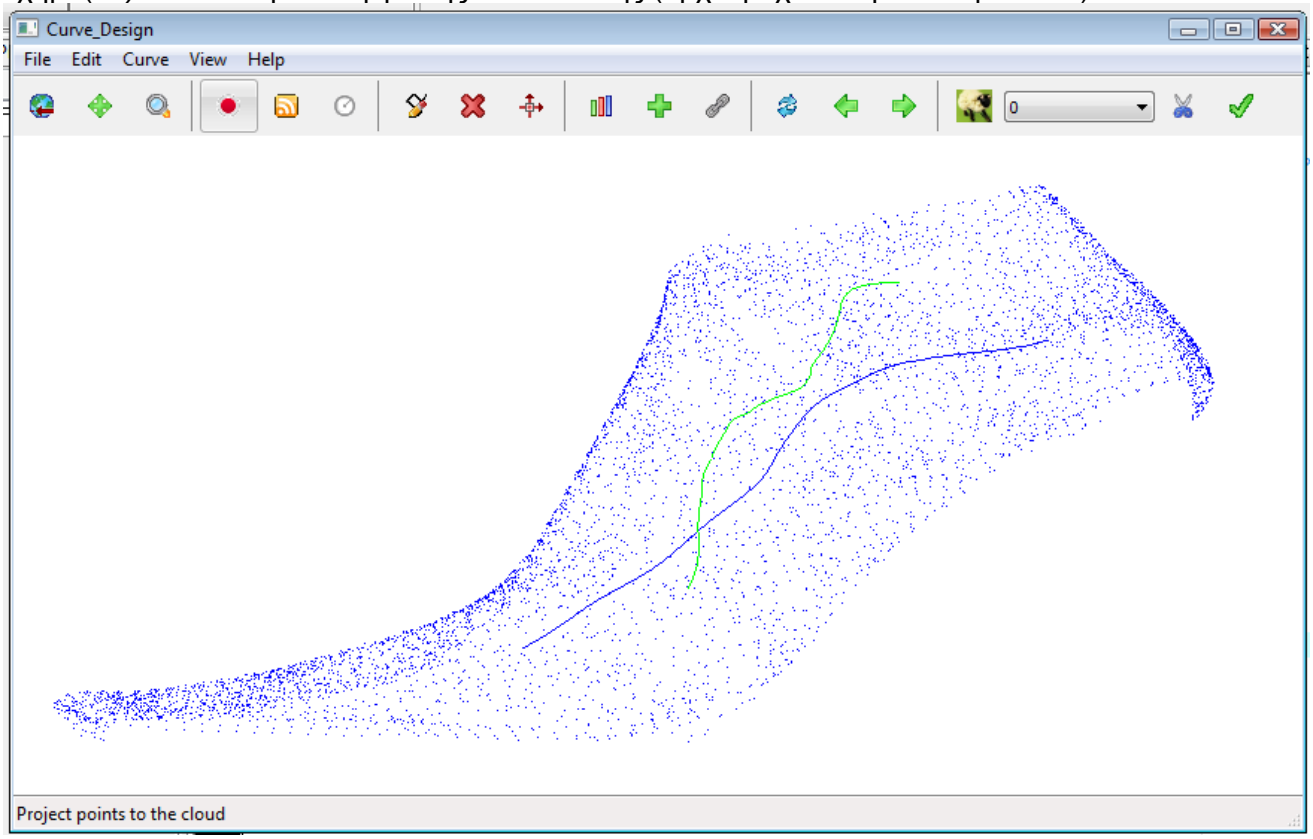
$tcb_{i,1/2}$: Οι καμπύλες που προκύπτουν μετά το trimming
$tds_{i,1/2}$: Οι ακολουθίες τον προβεβλημένων σημείων των αντίστοιχων trimmed καμπυλών

Τοπικές Μεταβλητές :
tp : Το σημείο τομής των καμπυλών εισόδου
cpt_1 : Το σημείο ελέγχου της $1^{ης}$ καμπύλης που προηγείται του σημείου τομής *tp*
cpt_2 : Το σημείο ελέγχου της $2^{ης}$ καμπύλης που έπεται του σημείου τομής *tp*

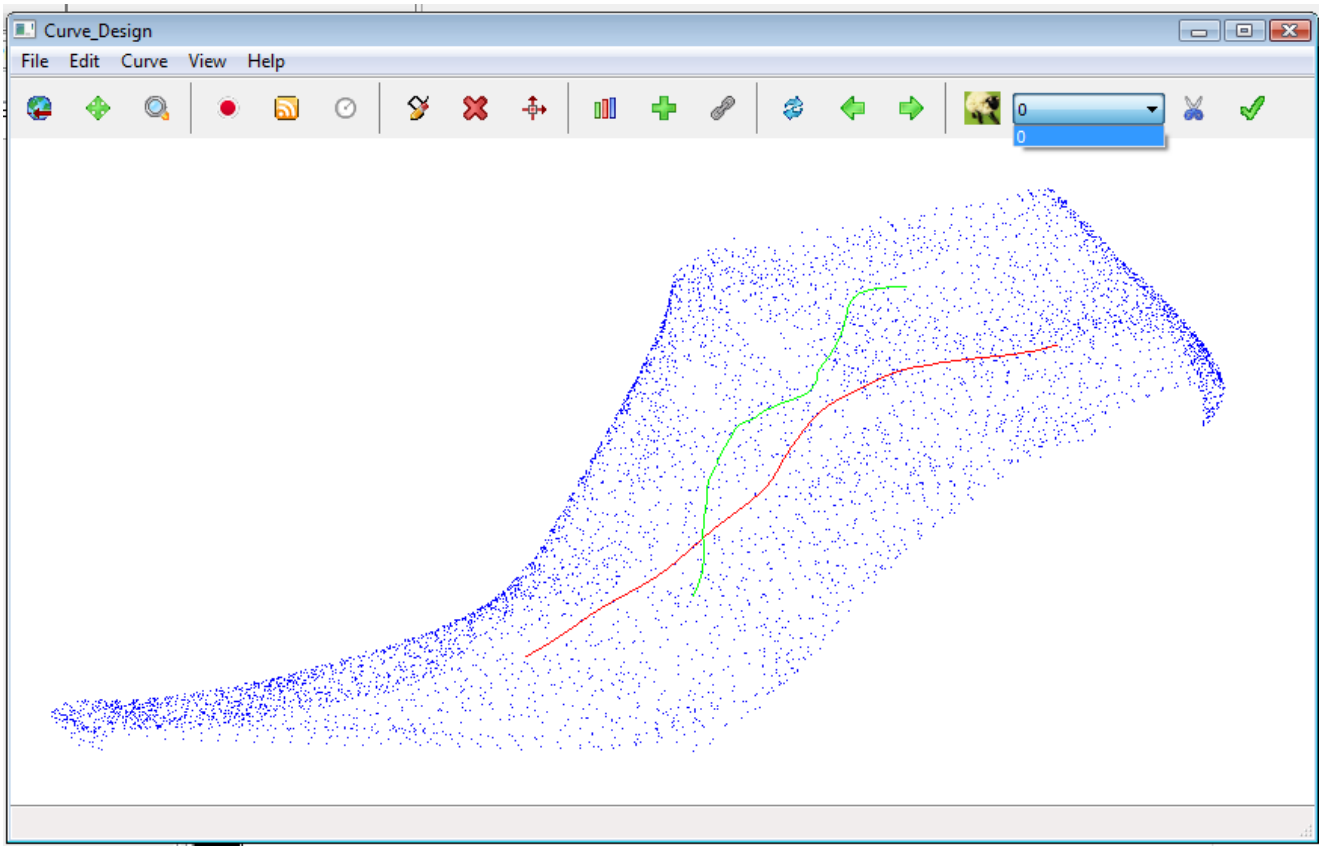
Begin
for(k = 0 ; k<size($ds_{i,1}$)-1 ; k++)
{
    for(l = 0 ; l<size($ds_{i,2}$)-1 ; l++)
    {
        if(Intersect($ds_{i,1}$(k,k+1), $ds_{i,2}$(l,l+1), tp))
            break;
    }
}
GetControlPoints($cb_{i,1}$, $cb_{i,2}$, tp , cpt_1, cpt_2) ;
$tcb_{i,1}$ = Bezier Curve(start($cb_{i,1}$), cpt_1) ;
$tcb_{i,2}$ = Bezier Curve(cpt_2, end($cb_{i,2}$)) ;
Project Curve($tcb_{i,1}$, C_N, γ, $tds_{i,1}$) ;
Project Curve($tcb_{i,2}$, C_N, γ, $tds_{i,2}$) ;
End



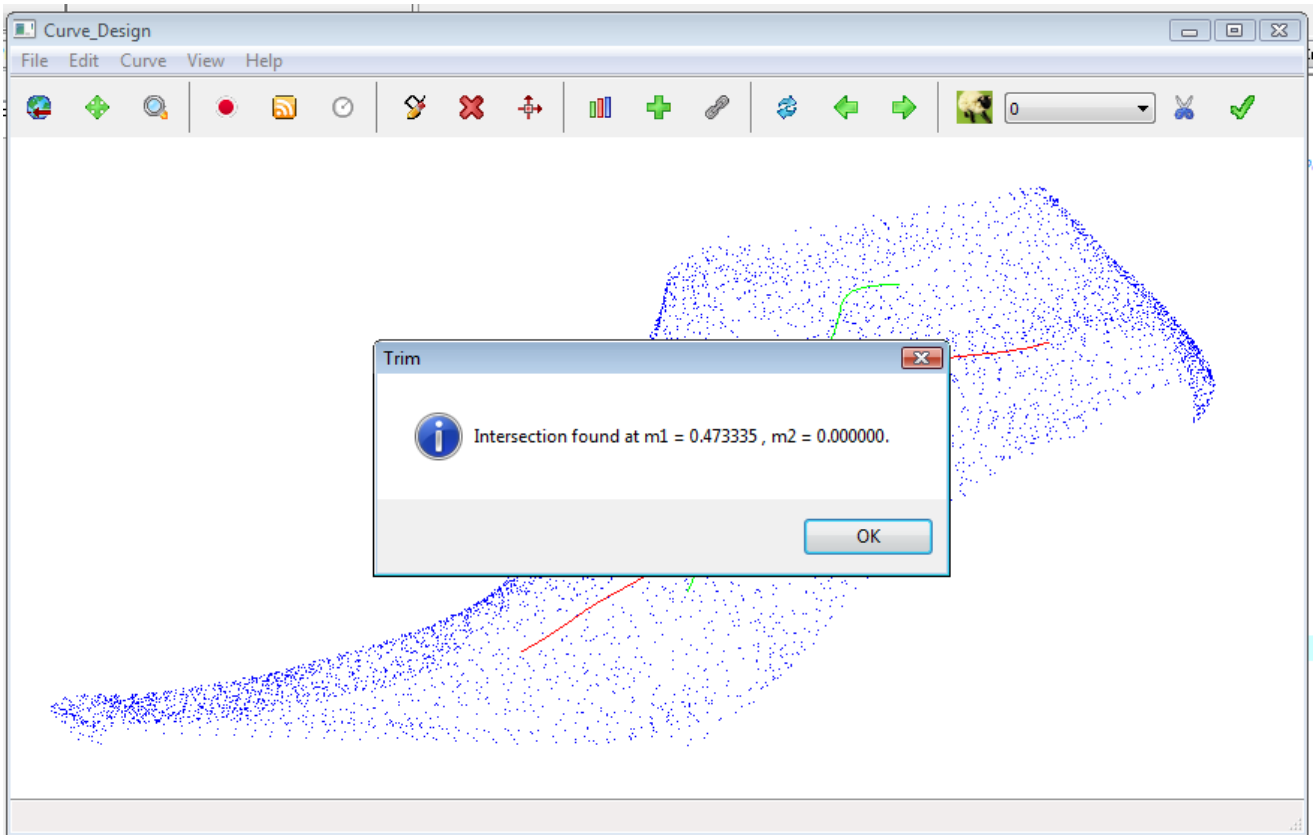
Σχήμα(26) Ενδεικτική λειτουργία της Trim εντολής (αρχική σχεδίαση δυο καμπυλών).



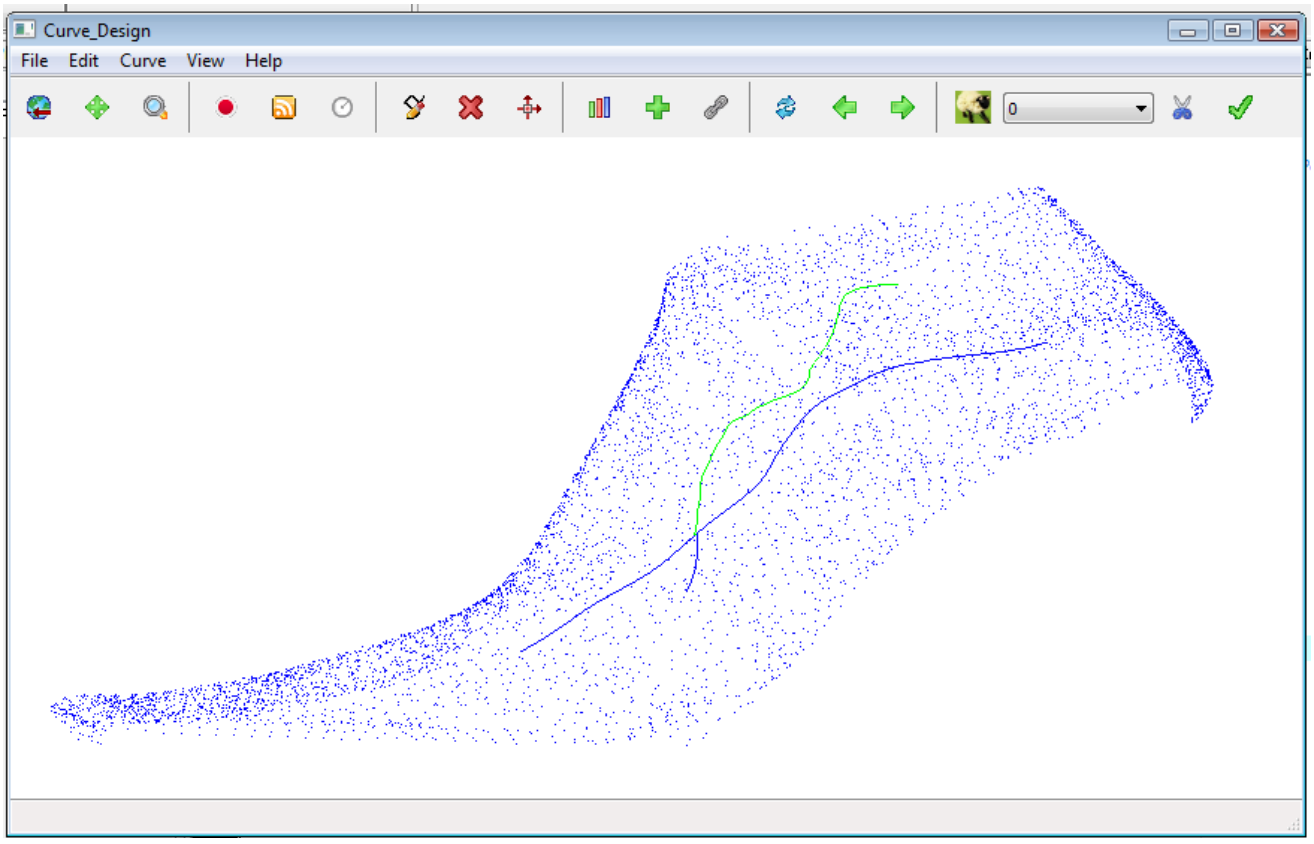
Σχήμα(27) Ενδεικτική λειτουργία της Trim εντολής (σχεδίαση δυο καμπυλών).



Σχήμα(29) Ενδεικτική λειτουργία της Trim εντολής (επιλογή της καμπύλης οδηγού).



Σχήμα(29) Ενδεικτική λειτουργία της Trim εντολής (ένδειξη του σημείου τομής).



Σχήμα(30) Ενδεικτική λειτουργία της Trim εντολής (η καμπύλη έχει χωριστεί σε δυο τμήματα).

10.Εφαρμογές και Παραδείγματα σχεδίασης προϊόντων.

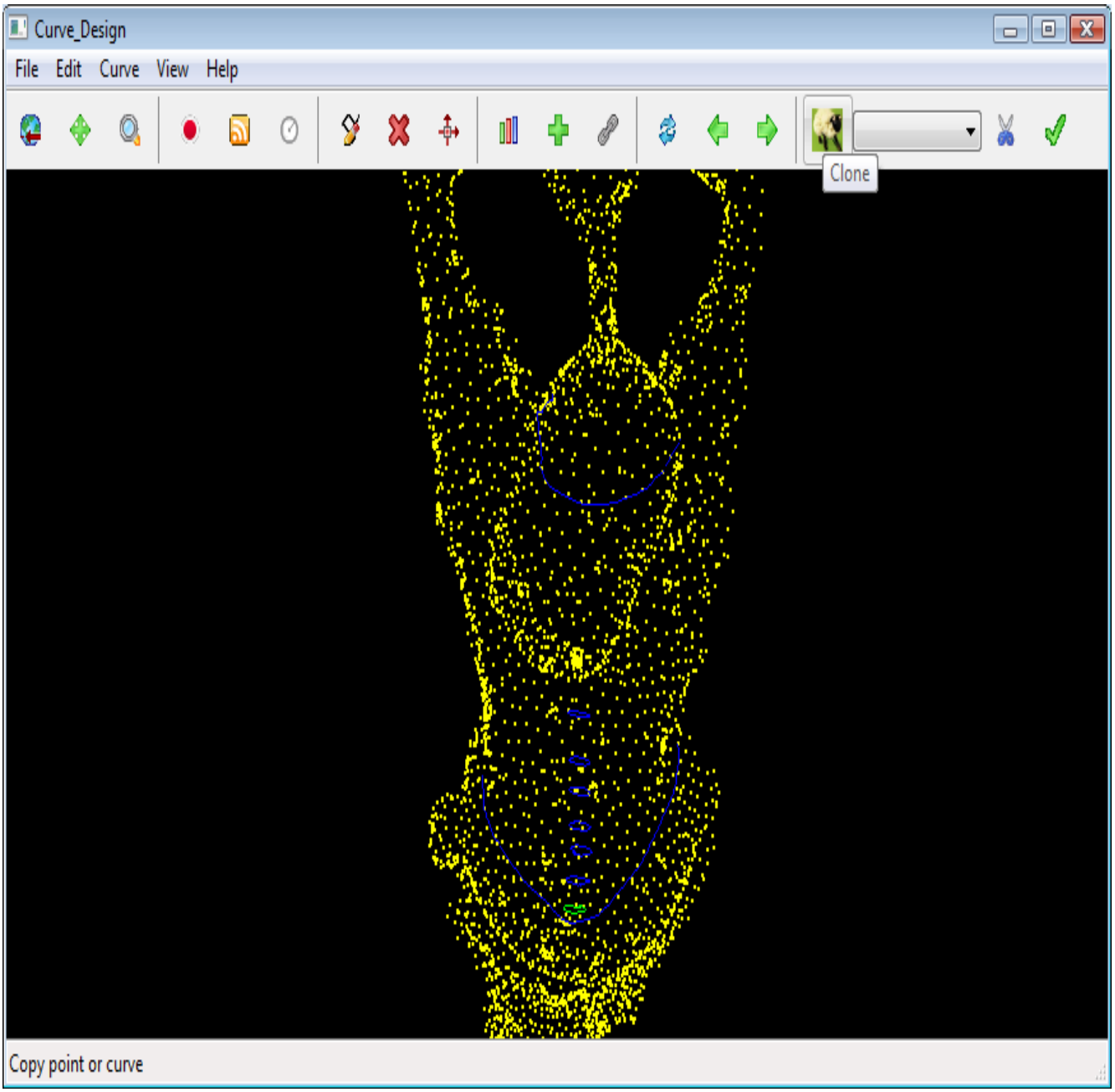
Οι εφαρμογές που μπορεί να έχει ένα νέφος σημείων (point cloud) εκτός από αυτές που έχουν αναφερθεί στην παράγραφο 2.6, εξαρτώνται και από τις ανάγκες των απαιτήσεων της εκάστοτε εφαρμογής. Ενδεικτικά μερικά από αυτά μπορεί να είναι.

- Δημιουργία 3D μοντέλων επιφανειών
- Τοπογραφικές αποτυπώσεις μεγάλης κλίμακας
- Μεταλλεία
- Αποτύπωση και παρακολούθηση μεγάλων τεχνικών έργων (δρόμοι, γέφυρες, φράγματα κλπ)
- Αρχαιολογικές εφαρμογές
- Αρχιτεκτονικές εφαρμογές
- Εφαρμογές εικονικής πραγματικότητας
- Βιομηχανικές εφαρμογές

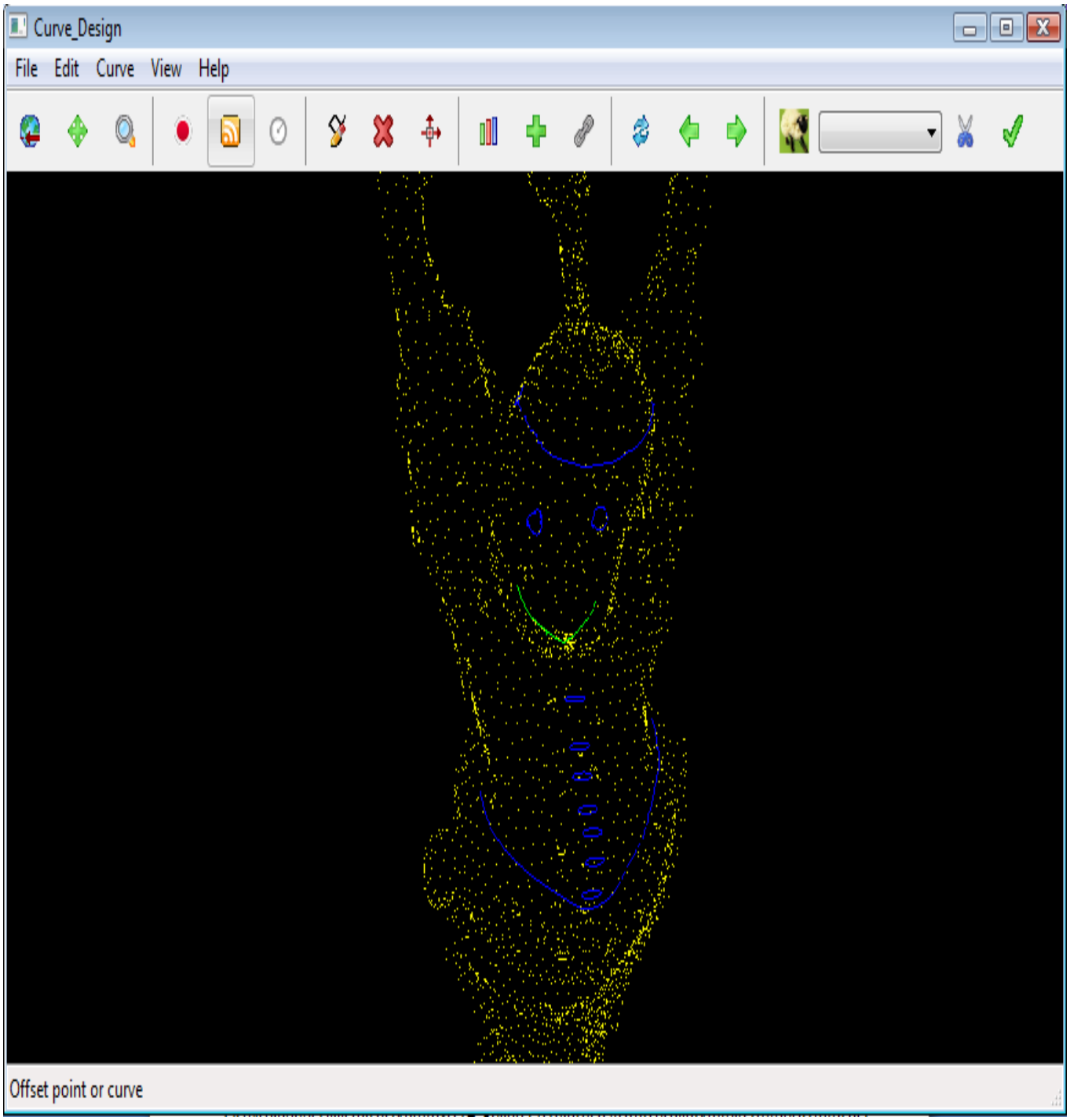
Μερικά από τα άμεσα πλεονεκτήματα που μπορεί να έχει μια τέτοια εφαρμογή είναι.

- Άμεσος υπολογισμός μηκών, υψών, γωνιών, ακτινών, όγκων.
- Αυτόματη εξαγωγή χαρακτηριστικών, όπως τομές, γραμμές δρόμων.
- Εξαγωγή δεδομένων απ' ευθείας σε μορφή DXF, IGES κλπ.
- Αυτοματοποιημένη δυνατότητα σύνθεσης του νέφους σημείων από πολλαπλές στάσεις χωρίς τη χρήση ειδικών στόχων.

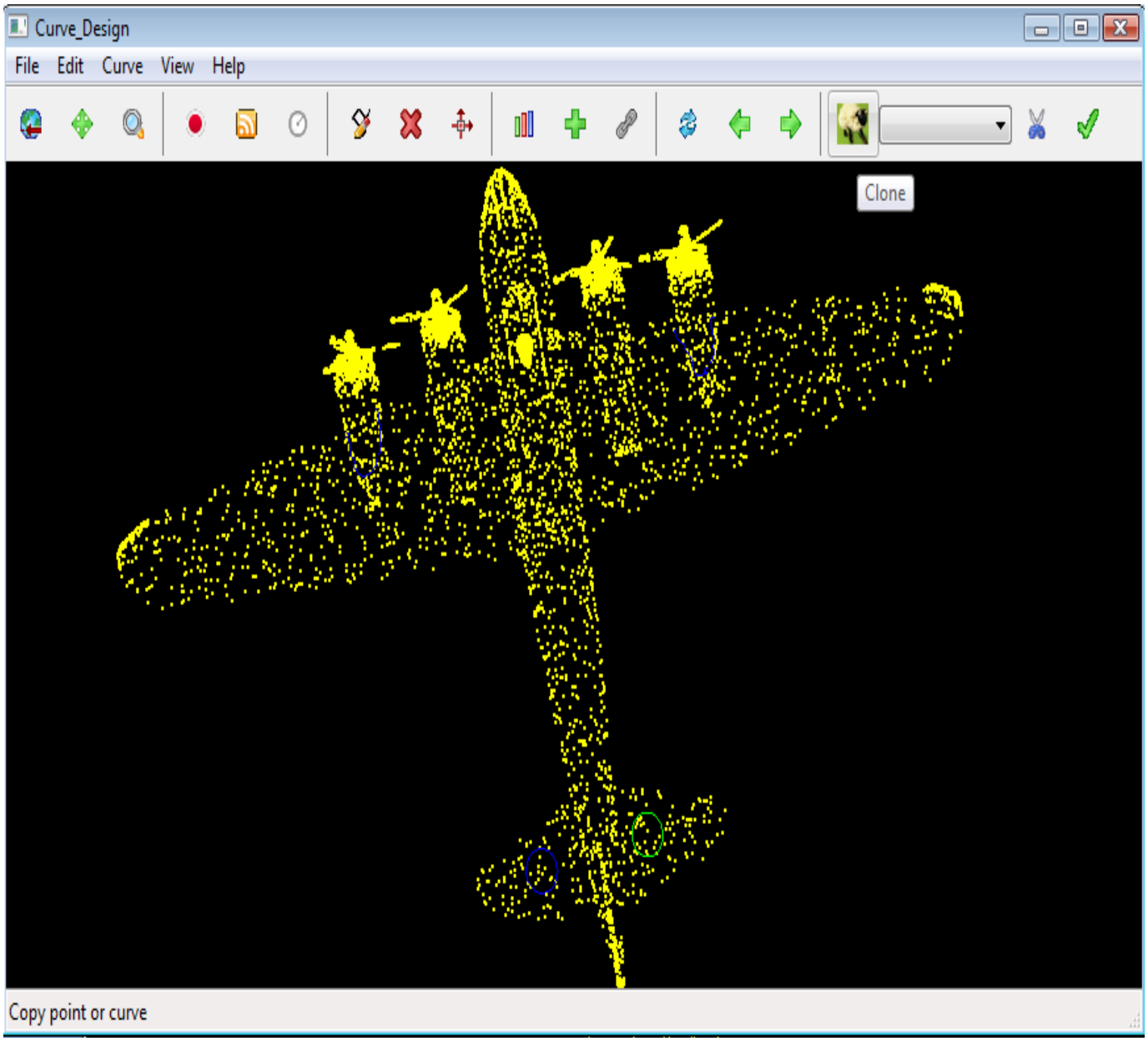
Μπορεί οι εφαρμογές αυτές να καλύπτουν ένα αρκετά μεγάλο σύνολο λειτουργιών, όμως αρχικά στις περισσότερες εφαρμογές τρισδιάστατης απεικόνισης, η οπτική πληροφορία που προσφέρει το νέφος σημείων από μόνο του συνήθως δεν επαρκεί, όσο πυκνό και αν είναι, αφού αυτό που απεικονίζεται στην οθόνη του υπολογιστή είναι ένα πλήθος κουκκίδων. Τα σημεία αυτά είτε χρησιμοποιούνται ως έχουν, είτε αποτελούν την πρώτη ύλη για την παραγωγή μιας μεγάλης σειράς δευτερογενών προϊόντων.



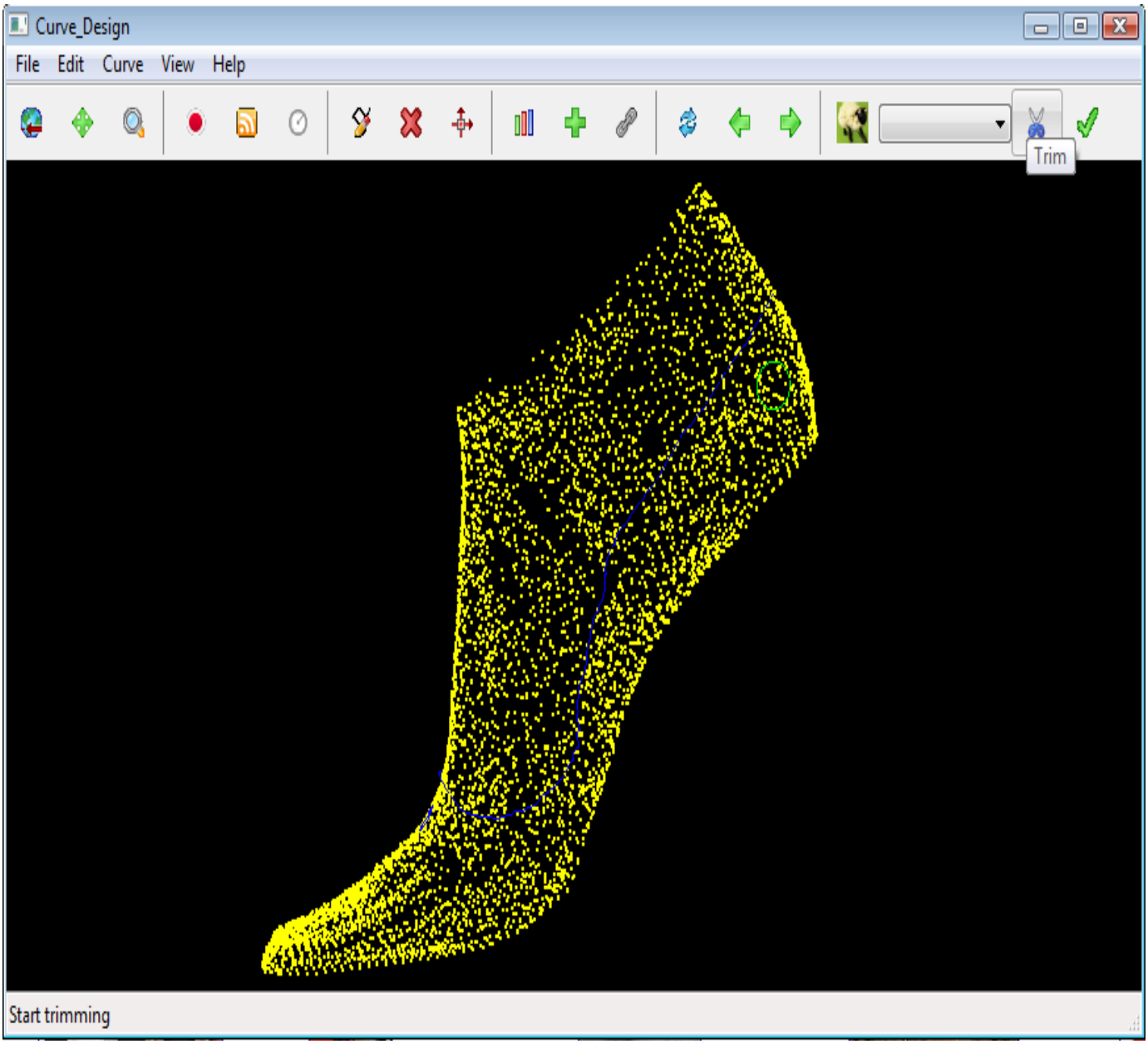
Σχήμα(31) Ενδεικτικές οθόνες από την σχεδίαση καμπυλών μέσα από την εφαρμογή.



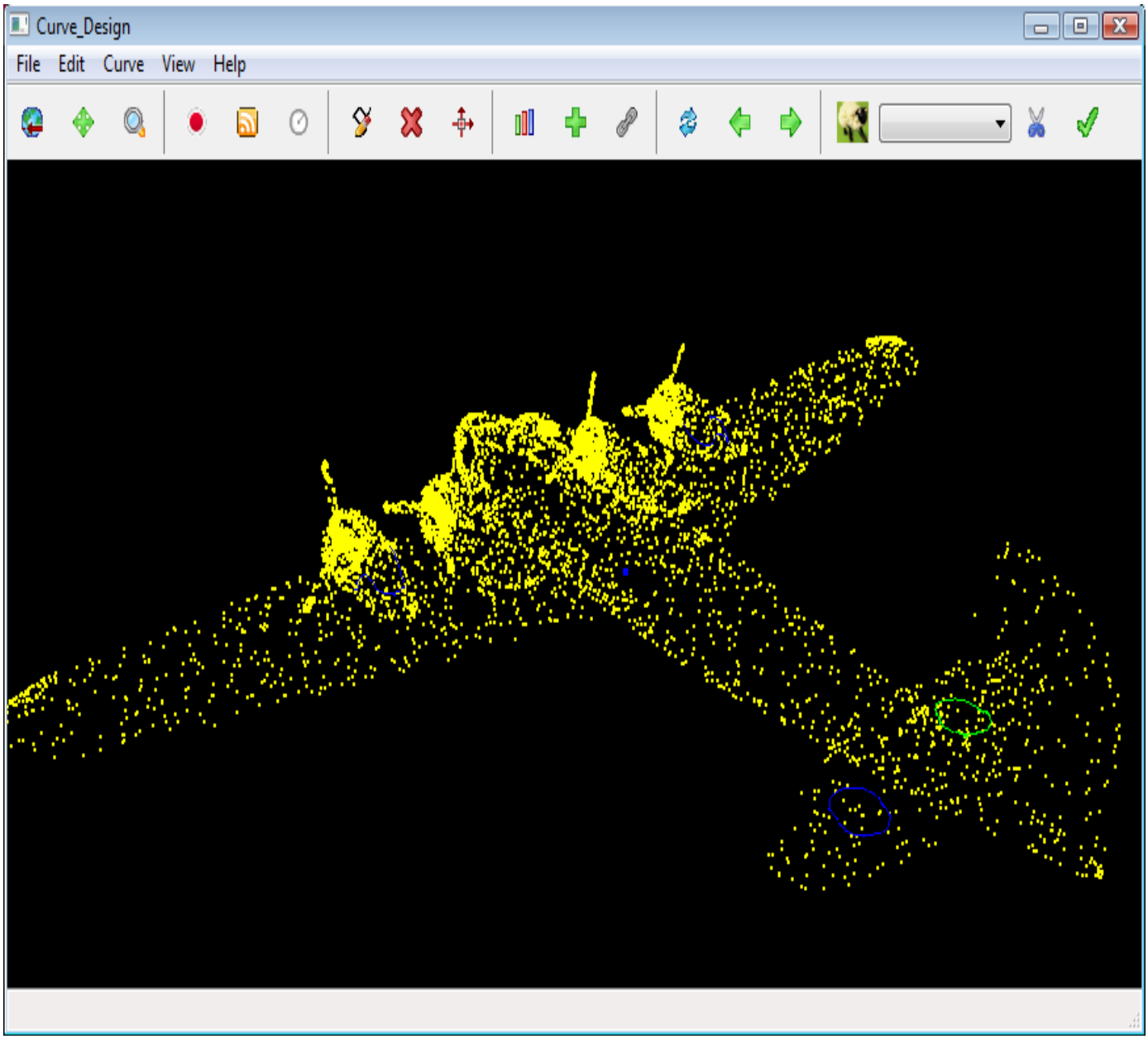
Σχήμα(32) Ενδεικτικές οθόνες από τη σχεδίαση καμπυλών μέσα από την εφαρμογή.



Σχήμα(33) Ενδεικτικές οθόνες από τη σχεδίαση καμπυλών μέσα από την εφαρμογή.



Σχήμα(34) Ενδεικτικές οθόνες από τη σχεδίαση καμπυλών μέσα από την εφαρμογή.



Σχήμα(35) Ενδεικτικές οθόνες από τη σχεδίαση καμπυλών μέσα από την εφαρμογή.

Το πρόγραμμα αυτό που έχει υλοποιηθεί όπως έχουμε αναφέρει αποτελεί ένα low level εργαλείο για το σχεδιασμό, τη μελέτη και υλοποίηση όλων αυτών. Ξεκινώντας από απλά σημεία, προχωρήσαμε στην υλοποίηση εργαλείων για την επεξεργασία και σχεδίαση πάνω στα σημεία αυτά. Ο σχεδιασμός γίνεται με καμπύλες B-Spline. Σε μια μετέπειτα ερευνά κάποιος μπορεί από τις καμπύλες αυτές να προχωρήσει και στη δημιουργία ακόμα και επιφανειών όπως και μια σειρά άλλων πολύπλοκων εργαλείων που θα αποτελέσουν κομμάτια για την υλοποίηση κάποιου αντικείμενου. Ακολουθεί μια σειρά από αντικείμενα που έχουν μελετηθεί και υλοποιηθεί από διάφορα εργαλεία που αποτελούν την εφαρμογή μας.

11. Βιβλιογραφία- URL Αναφορές

[1] Azariadis P. Sapidis N. Drawing curves onto a cloud of points for points-based modeling, Computer-Aided Design 37(1) (2005)

[2] Adamson A. Alexa M. In: Kobbelt L. Schorder P. Hoppe H. editors. Approximating and Intersecting surfaces from points, in eurographics symposium on geometry processing.2003. p.203-9.

[3] Alexa M. Behr J. Cohen-Or D. Fleishman S. Levin D. Silva C. Computing and rendering point set surfaces. IEEE TVCG 2003;9(1): 3-15

[4] Amenda M. Bern M. Kamvysselis M. A new Voronoi-based surface reconstruction algorithm. In: Proceeding of the SIGGRAPH 98. Computer Graphics Proceedings,Annual Conference Series; 1998. p 415-22.

[5] Jean-Daniel B. Frederic C. Smooth surface reconstruction via natural neighbor interpolation of distance function. Comput Geom 2000; 223-32

[6] Benko P. Kos G. Varady T. Andon L. Martin R. Constrained fitting in Reverse Engineering. Comput Aided Geom Des 2002; 19(3): 173-205.

[7] Benko P. Varady T. Segmentation methods for smooth point regions of conventional engineering objects. Comput-Aided Des 2004; 36(6): 511-23.

[8] Boonning R. Muller H. Interactive sculpturing and visualization of unbounded voxel volumes. Proceeding of the Seventh ACM Symposium on Solid Modeling and Applications; 2002. P 212-9.

[9] Benouamer M. Michelucci D. Bridging the gap between CSG and Brep via a triple ray representation. Proceeding of the Fourth ACM Symposium on Solid Modeling and Applications; 1997. p 68-79.

[10] Cripps R. Algorithms to support point-based cadcam. Int J Mach Tool Manufact 2003;43(4): 425-32.

[11] Corbo P. Germani M. Mandorli F. Aesthetic and functional analysis for product model validation in reverse engineering applications. Comput Aided Des 2004;36(1); 65-74

[12] Eck M. Jaspert R. In: Sapidis N. editor. Automatic fairing of point set, designing fair Curves and surfaces: shape quality in geometric modeling and computer aided design. Philadephia, PA: Society for Industrial and Applied Mathematics (SIAM); 1994 p.45.

[13] Erikson C. Manocha D. GAPS: general and automatic polygonal simplification. Seventh ACM Symposium on Solid Modeling and Application Tutorial T4: Handling Large Geometric Datasets; 2002.

[14] Felderman M. In: Sapidis N. editor. Tight string method to fair piecewise linear curves,designing fair curves and surfaces: shape quality in geometric modeling and computer aided design. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM) 1994. p. 61-72.

[15] Horvath I. Rusak Z. Collaboration virtual design environments: collaboration shape Conceptualization in virtual design environments. Commun ACM 2001; 44(12): 59-63.

[16] Jerard R. Angleton J. Drysdale R. Su P. The use of surface points sets for generation, simulation, verification and automatic correction of NC machining programs. Proceedingof the NSF Design and Manufacturing Systems Conference, Tempe, AZ, Society of Manufacturing Engineers; 1990. p. 143-8.

[17] Jansson J. Vergeest J. A discrete mechanics model for deformable bodies. Comput Aided Des 2002; 34(12): 913-28.

[18] Kobbelt L. Schroder P. A multiresolution framework for variational subdivision. ACM Trans Graph 1998; 17: 209-37.

[19] Lee I. Curve reconstruction from unorganized points. Comput Aided Des 2000;17(2) 161-77.

[20] Llamas I. Kim B. Gargus J. Rossignac J. Shaw C. Twister: a space-warp operator for the two-handed editing of 3D shapes. ACM Trans Graph 2003; 22(3): 663-8.

[21] Liu G. Wong Y. Zhang Y. Loh H. Error-based segmentation of cloud data for direct rapid Prototyping. Comput Aided Des 2002; 35(7): 633-45.

[22] Liu GH. Wong YS. Zhang YF. Loh HT. Adaptive fairing of digitized point data with discrete curvature. Comput Aided Des 2002; 34;(4): 309-20.

[23] McLaughlin W. Describing the surface: algorithms for geometric modeling. Comput Mech Eng 1986; 38-41.

[24] Menon J. Marisa R. Zagajac J. More powerful solid modeling through ray representations IEEE Comput Graph Appl 1994; 14(3): 22-35.

[25] Muller H. Surmann T. Stautner M. Albersmann F. Weinert K. Online sculpting and Visualization of multi-dexel volumes. Proceedings of the Eighth Symposium an Solid Modeling and Applications 2003; 258-61.

[26] Pottmann H. Randrup T. Rotational and helical surface approximation for reverse engineering Computing 1998; 60(4): 307-22

[27] Piegl L. Tiller W. The NURBS bokk. Berlin: Springer; 1997.

[28] Qin S. Harrison R. West A. Jordanov I. Wright D. A framework of web-based conceptual design. Comput Ind 2003; 50(2); 153-64

[29] Rusak Z. Horvath I. Kuczogi G. Akar E. Shape instance generation from domain distributed vague models. Proceeding of the DETC 02 ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, Montreal; 2002.

[30] Schweikardt E, Gross M. Digital Clay: deriving digital models from freehand sketches. Automat Construct 2000; 9(1): 107-15.

[31] Vergeest J. Horvath I. Spanjaard S. A methodology for reusing freeform shape content. Proceeding of the Design Theory and Methodology Conference, DETC 01/DTM-21708, NewYork: ASME; 2001

[32] Varady T. Martin RR. Cox J. Reverse engineering of geometric models-an introduction. Comput-Aided Des 1997; 29(4) 255-68.

[33] Woo H. Andor L. Renner G. Varady T. Advanced surface fitting techniques. Comput- Aided Geom Des 2002; 19(1) :19-42.

[34] Hoo H. Kang E. Wang S. Lee K. A new segmentation method for point cloud data. In J Mach Tools Manufact 2002; 42(2): 167-78

[35] Cho W. Takashi M. Nicholas PM. Topologically reliable approximation of composite Bezier curves. Comput-Aided Geom Des 1996; in press.

[36] Wu D. Sarma R. The incremental editing og faceted models in an integrated design environment. Comput-Aided Des 2004 ; in press.

[37] Wu Y. Wong Y. Loh H. Zhang Y. Modelling cloud data using an adaptive slicing approach. Comput-Aided Des 2004; 36(3) 231-40.

[38] Yin Z. Reverse engineering of a NURBS surface from digitized points subject to boundary conditions. Comput Graph 2004;28(2): 207-12.

[39] Yin Z. Jiang S. Automatic segmentation and approximation of digitized points for reverse engineering. Int J Product res 2003; 41(13): 3045-58.

[40] Zwicker M. Pauly M. Knoll O. Gross M. Pointshop 3D an interactive system for point-based surface editing. ACM Trans Graph 2002; 21(3):322-9. Special issue: Procceeding of ACM SIGGRAPH

[41] Παντελίδη Γ. Κραββαρίτη Δ. Νασοπούλου Β. Τσεκρέκου Π. “Γραμμική Αλγεβρα” Αθηνα (1992)

[42] Αζαριάδης Φ. “Σημειωσεις Γραφικων-OpenGL”

[43] Deitel H. Deitel P. “C++ how to program (3th Edition)” New Jersey (2001)

[44] Wright R. Lipchak B. Haemel N. OpenGL SuperBible: “Comprehensive Tutorial and Reference (4th Edition)” Boston (2005)

//[45] Wright R. Lipchak B. Haemel N.OpenGL SuperBible: “Comprehensive Tutorial and Reference (4th Edition)” Boston (2005)

[46] Johnson J “GUI Bloopers 2.0 Common User Interface Design Don’ts and Dos (2th Edition)” Elsevier (2008)

[47] L. Linsen, "Point cloud representation," CS Technical Report, University of Karlsruhe, 2001.

[48] Floater M.S, Reimers M, Meshless paramerization and surface reconstruction, Computer Aided Geometric Design 18(2) (2001) 77-92

[49] Azariadis P. Parameterization of cloud unorganized point using dynamic base surfaces. Comput-Aided Des 2004;36(7):607-23.

[50] Tamas Varady, Ralph R Martin* and Jordan Coxt “Reverse engineering of geometric models-an introduction” Elsevier Science 1997;256(1).

[51] Azariadis Philip, Sapidis Nickolas “Product design using point surfaces: A recursive subdivision technique for point parameterization” Elsevier 2007.

[52] Yu-Shen Liu, Jean-Claude Paul, Jun-Hai Yong, Pi-Qiang Yu, Hui Zhang, Jia-Guang Sun, Karthik Ramani “Automatic least-squares projection of points onto points cloud with applications in reverse engineering” Elsevier 2006.

[53] Ming-Cui Du. Yu-Shen Liu “An extension on robust directed projection of points onto clouds” ScienceDirect,2008.

[54] 3D Printer: http://www.emco.co.uk/rapid.htm

[55] What is Reverse Engineering: http://www.npd-solutions.com/reoverview.html

[56] Planar Curve Intersection: http://cagd.cs.byu.edu/~557/text/ch7.pdf

[57] Intersection: http://www.xtyler.com/code/104

[58] Projection: http://www.siggraph.org/education/materials/HyperGraph/viewing/view3d/perspect.htm

[59] Learning C++: http://www.roguewave.com/

[60] Vectors : http://emweb.unl.edu/math/mathweb/vectors/vectors.html

[61] Open GL: http://www.opengl.org/

[62] Surface approximation and Modeling: http://www.ams.sunysb.edu/~jsbm/surfapprox.html

[63] 3D Modeling & Rendering: http://courses.be.washington.edu/ARCH/481/00HomePage/0.default.html

[64] Axialis: http://www.axialis.com/

[65] HCI: http://www.hci-course.gr/

[66] Αλληλεπίδραση ανθρώπου μηχανής: http://meteora.csd.auth.gr/~dpolitis/hci/intro.htm

[67] University of the Aegean: http://www.syros.aegean.gr/users/kgp/hci.html

[68] IDE Program: http://c-ide1.software.informer.com/