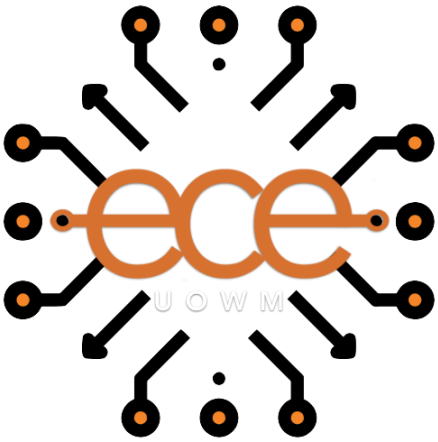


ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Προσαρμογή παραμετρικών καμπυλών σε 2D νέφη σημείων με χρήση μεθόδων τεχνητής νοημοσύνης

Πολυμενίδης Ιωάννης

Επιβλέποντες: Δρ. Αντώνιος Πρωτοψάλτης

Καθ. Νικόλαος Σαπίδης

Οκτώβριος 2021, Κοζάνη

ΠΡΟΣΑΡΜΟΓΗ ΠΑΡΑΜΕΤΡΙΚΩΝ ΚΑΜΠΥΛΩΝ ΣΕ 2D ΝΕΦΗ ΣΗΜΕΙΩΝ ΜΕ ΧΡΗΣΗ ΜΕΘΟΔΩΝ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ

Πολυμενίδης Ιωάννης

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Επιβλέποντες: Δρ. Αντώνιος Πρωτοψάλτης, Καθ. Νικόλαος Σαπίδης

Οκτώβριος 2021, Κοζάνη

Δήλωση Πνευματικών Δικαιωμάτων

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ. 3 του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο

"ΠΡΟΣΑΡΜΟΓΗ ΠΑΡΑΜΕΤΡΙΚΩΝ ΚΑΜΠΥΛΩΝ ΣΕ 2D ΝΕΦΗ ΣΗΜΕΙΩΝ ΜΕ ΧΡΗΣΗ ΜΕΘΟΔΩΝ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ"

καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Αντώνιος Πρωτοψάλτης αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή / και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή.

Υπογραφή Φοιτητή:

Περίληψη

Στην παρούσα διπλωματική εργασία ασχοληθήκαμε με το πρόβλημα προσαρμογής παραμετρικών καμπύλων Bezier σε 2Δ νέφη σημείων, με τη χρήση μηχανικής μάθησης, και με την αξιολόγηση τους απέναντι στη μέθοδο ελάχιστων τετραγώνων. Πιο συγκεκριμένα, αναλύεται σε βάθος η μορφή και η σχεδίαση των παραμετρικών καμπυλών Bezier και των καμπύλων B-spline καθώς και οι τρόποι σχεδίασης τους. Γίνεται μια εισαγωγή στην έννοια του νέφους σημείων και το πως παράγεται. Έπειτα αναλύεται ο ορισμός της μηχανικής μάθησης και δίνεται βάση στα νευρωνικά δίκτυα. Για την προσαρμογή καμπύλης σε 2Δ νέφος σημείων, υλοποιήθηκαν δύο μέθοδοι: η μέθοδος ελαχίστων τετραγώνων και η μέθοδος νευρωνικών δικτύων. Τέλος γίνεται σύγκριση των δυο μεθόδων και εξάγονται χρήσιμα συμπεράσματα.

Abstract

In the present thesis we have dealt with the problem of fitting Bezier parametric curves to 2D point clouds, using machine learning, and evaluating them against the least square method. More specifically, the shape and design of the Bezier parametric curves and the B-spline curves are analyzed in depth. An introduction is made to the concept of point clouds and how they are produced. The definition of machine learning is then analyzed. We implement two methods for fitting Bezier curve to 2D point clouds: Least squares and neural networks. Finally, the two methods are compared, and useful conclusions are drawn.

Περιεχόμενα

Κατάλογος Εικόνων

ΚΕΦΑΛΑΙΟ: 1 Εισαγωγή

Το πρόβλημα της προσαρμογής καμπύλης σε νέφη σημείων είναι από τα πιο βασικά ερευνητικά θέματα σε πληθώρα επιστημονικών πεδίων όπως των γραφικών υπολογιστών, της σχεδίασης με υπολογιστή (CAD – Computer Aided Design), της επεξεργασίας εικόνας, της εξόρυξης δεδομένων κλπ.. Σε κάθε εφαρμογή, το είδος της καμπύλης που θα προσαρμοστεί παίζει σημαντικό ρολό για να καλυφθούν οι απαιτήσεις του συστήματος. Η παρούσα εργασία έχει εστιάζει στις παραμετρικές καμπύλες Bezier και στις διάφορες εκδοχές τους. Πάρα το γεγονός ότι οι καμπύλες Bezier υστερούν σε ευελιξία σε σχέση με πιο πρόσφατες και πιο περίπλοκες καμπύλες, προσφέρουν καλή κατανόηση των βασικών αρχών της προσαρμογής καμπύλης.

Οι καμπύλες στον χώρο των γραφικών υπολογιστών μας είναι ιδιαίτερα χρήσιμες γιατί επιτρέπουν την σχεδίαση περίπλοκων σχημάτων που δεν είναι δυνατό να περιγράφουν μόνο από ευθείες γραμμές και μαθηματικές καμπύλες. Μερικά παραδείγματα εφαρμογών που χρησιμοποιούνται καμπύλες είναι ο σχεδιασμός επιφανειών οχημάτων, (σασί αυτοκίνητων, ύφαλα πλοίων, ατράκτους αεροσκαφών), η αρχιτεκτονική κτιρίων και γενικότερα στη μοντελοποιήση και αναπαράσταση τρισδιάστατων αντικειμένων στον υπολογιστή.

Εφαρμογές που υποβοηθούν τους επιστήμονες και τους επαγγελματίες στην σχεδίαση ονομάζονται συστήματα CAD (Computer-Aided Design) ή CAGD (Computer-Aided Geometric Design) οι οποίες χρησιμοποιούν μεθόδους και τεχνικές των γραφικών υπολογιστών για να προβάλουν καμπύλες και επιφάνειες.

1.1 Κίνητρο

Οι μέθοδοι της προσαρμογής καμπυλών περιλαμβάνουν συνήθως την τεχνική βελτιστοποίησης της παλινδρόμησης (regression) [1] χρησιμοποιώντας τη μέθοδο των ελάχιστων τετράγωνων [2], η οποία είναι μια κλασική τεχνική για αυτού του είδους τα προβλήματα. Η συγκεκριμένη μέθοδος προσπαθεί να υπολογίσει, μέσα από ένα σύνολο δεδομένων, τη βέλτιστη συνάρτηση που τα αναπαριστά ελαχιστοποιώντας μια συνάρτηση λάθους. Με την ραγδαία ανάπτυξη της μηχανικής μάθησης είναι δυνατή η χρήση τεχνικών τεχνητής νοημοσύνης και πιο συγκεκριμένα μέσω των νευρωνικών δικτύων για να επιτευχθεί ο ίδιος σκοπός με ακριβέστερα αποτελέσματα για το εν λόγω πρόβλημα. Τα νευρωνικά δίκτυα έχουν τη δυνατότητα να αποδίδουν τη βέλτιστη συνάρτηση αφού έχει προηγηθεί η κατάλληλη εκπαίδευση τους. Το γεγονός αυτό τα κάνει πολύ ελκυστικά για τέτοια προβλήματα. Επίσης για την σωστή

αξιολόγηση αυτών των μεθόδων θα πρέπει να χρησιμοποιηθεί ένας αλγόριθμος που υπολογίζει την μικρότερη απόσταση από των δεδομένων από την καμπύλη.

1.2 Αναγκαιότητα της προτεινόμενης έρευνας

Η προσαρμογή καμπύλης σε δεδομένο σύνολο τιμών, είναι ένα μια πολύ σημαντική διαδικασία για την επίλυση προβλημάτων με μεγάλο όγκο δεδομένων. Η σημαντικότητα της έγκειται στη δυνατότητά της να δημιουργεί ένα μαθηματικό μοντέλο για την ακριβή πρόβλεψη μελλοντικών τιμών. Η προσαρμογή καμπύλης βρίσκει μεγάλη εφαρμογή σε τομείς όπως η ιατρική, η βιολογία, η μηχανική, η χρηματοοικονομική κτλ.. Αν γνωρίζουμε την δομή των δεδομένων, η οποία προκύπτει από το πρόβλημα στο οποίο εφαρμόζεται η προσαρμογή, τότε η εύρεση της καμπύλης γίνεται ευκολότερη.

1.3 Σκοπός της εργασίας

Ο στόχος αυτής της εργασίας είναι να εξετάσει μεθόδους παλινδρόμησης πάνω στη προσαρμογή παραμετρικών καμπυλών για μεγάλο όγκο σημείων. Πιο συγκεκριμένα ερευνώνται μέθοδοι προσαρμογής καμπυλών Bezier σε νέφη σημείων με τη χρήση του αλγορίθμου των ελαχίστων τετραγώνων. Παράλληλα το ίδιο πρόβλημα προσεγγίζεται και με πιο σύγχρονες μεθόδους αυτές της τεχνητής νοημοσύνης. Τα νευρωνικά δίκτυα τα οποία υπάγονται στον τομέα της μηχανικής μάθησης έχουν την δυνατότητα για επιλύουν τέτοια προβλήματα προσαρμογής αλλά με την κατάλληλη εκπαίδευση. Τελικά η εργασία θα καταλήξει σε κάποιο συμπέρασμα για την χρησιμότητα και τον τρόπο χρήσης της κάθε μεθόδου, λαμβάνοντας υπόψιν τον χρόνο υπολογισμού και την απόκλιση από τα πραγματικά δεδομένα.

1.4 Δομή της εργασίας

Στο 2^{ο} κεφάλαιο της παρούσας διπλωματικής εργασίας θα αναλυθούν λεπτομερώς έννοιες και δομές των παραμετρικών καμπυλών Bezier και B-spline καθώς και στους αλγορίθμους σχεδίασής τους. Επίσης, θα αναλυθεί η έννοια του νέφους σημείων και το πως παράγεται. Στο 3^{ο} κεφάλαιο θα γίνει μια εισαγωγή για την μηχανική μάθηση και ειδικότερα τα νευρωνικά δίκτυα, θα αναλυθούν λεπτομερώς και αναφερθούν πολλά είδη αυτών. Στο 4^{ο} κεφάλαιο θα εξεταστεί η θεωρία της προσαρμογής καμπύλων και επίσης τρόποι επίλυσης της όπως η γραμμική ανάλυση παλινδρόμησης με των αλγόριθμο ελαχίστων τετραγώνων. Στο 5^{ο} κεφάλαιο θα εξηγηθεί η υλοποίησης της μεθόδου των ελαχίστων τετραγώνων πάνω

στις καμπύλες Bezier καθώς και η προσέγγιση της παρούσας εργασίας με χρήση των νευρωνικών δικτύων. Τέλος το 6ο κεφάλαιο αναφέρεται στις δυνατότητες επέκτασης και βελτίωσης των μεθόδων που προτείνονται.

ΚΕΦΑΛΑΙΟ: 2 Θεωρία καμπυλών

2.1 Τρισδιάστατη αναπαράσταση αντικειμένων

Με την ταχύτατη ανάπτυξη της τεχνολογίας των υπολογιστών η ανάγκη για σχεδίαση πραγματικών αντικειμένων στον υπολογιστή έχει αυξηθεί εξίσου. Μεγάλη γκάμα μεθόδων για την σχεδίαση αντικειμένων έχει αναπτυχθεί που προσεγγίζουν τα χαρακτηριστικά τους με διαφορετικό τρόπο. Πολλές φορές συνδυασμός τέτοιων μεθόδων χρησιμοποιείται για να αποτυπωθεί η μοναδικότητα του κάθε αντικειμένου στην οθόνη.

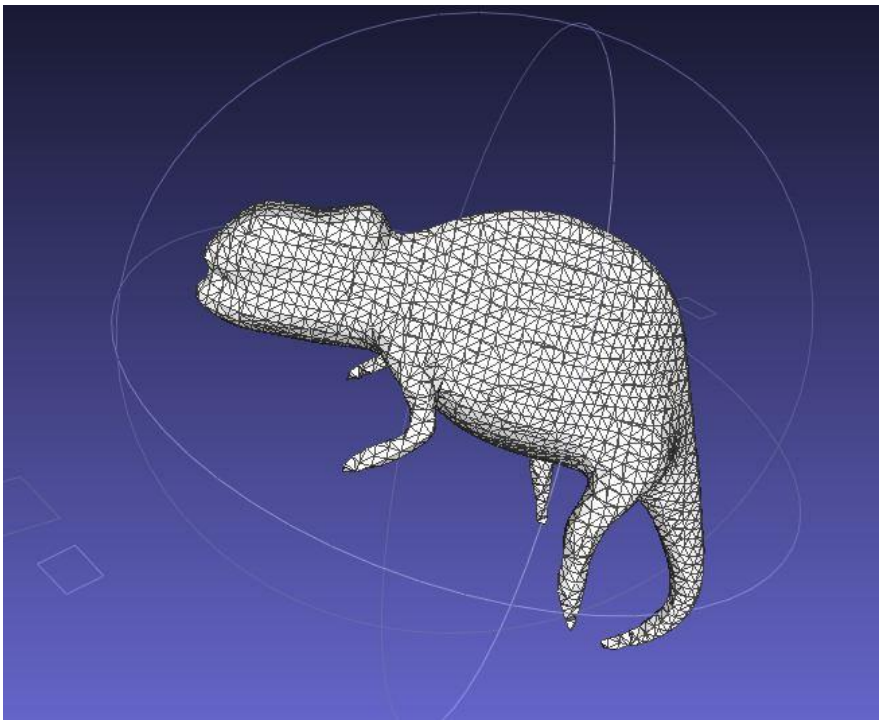
Για την αναπαράσταση τρισδιάστατων αντικειμένων χρησιμοποιούνται πλέγματα επίπεδων σχημάτων τα οποία ορίζουν την επιφάνεια του αντικειμένου. Αυτά τα σχήματα ονομάζονται τυπικά αντικείμενα γραφικών [3] (standard graphical objects) τα οποία είναι επίπεδα δυο διαστάσεων όπως τρίγωνα, παραλληλόγραμμα, εξάγωνα κτλ. Αυτά τα σχήματα είναι η βάση για την αναπαράσταση τρισδιάστατων στερεών μοντέλων. Με την ένωση αυτών των επίπεδων σχημάτων μπορούμε να δημιουργήσουμε τρισδιάστατα κανονικά σχήματα όπως τετράεδρα (ή πυραμίδα), εξάεδρα (ή κύβος), οκτάεδρα , εικοσάεδρο κτλ.

Για την αναπαράσταση στερεών που αποτελούνται από επιφάνειες μεγαλύτερου βαθμού (πχ. σφαίρες, ελλειψοειδή, δακτυλίους κτλ.) απαιτούνται εύκαμπτες μαθηματικές καμπύλες και επιφάνειες που περιγράφονται από παραμετρικές εξισώσεις.

2.2 Νέφη σημείων

Το νέφος σημείων είναι μια αναπαράσταση της ολική εξωτερικής επιφάνειας ενός 3D αντικειμένου και ορίζεται από ένα πλήθος σημείων στο καρτεσιανό σύστημα συντεταγμένων πού το καθένα παίρνει μια τιμή για κάθε έναν άξονα x,y,z. Η δημιουργία του νέφους σημείων είναι αποτέλεσμα 3D σάρωσης του πραγματικού αντικειμένου και προσεγγίζει το σχήμα του αντικειμένου μέσω μεμονωμένων σημείων που μετατρέπονται σε επιφάνειες μετά από τον αλγόριθμο ανακατασκευής επιφανειών. Οι μετρήσεις αρχικά παίρνονται από αισθητήρες (κάμερες, λέιζερ) σε σύστημα πολικών συντεταγμένων και έπειτα το σύστημα τις μετατρέπει σε τρις-ορθογώνιο καρτεσιανό σύστημα συντεταγμένων για κάθε ένα από τα σημεία.

Τα νέφη σημείων μπορούν να μετατραπούν σε τρισδιάστατα μοντέλα με την χρήση τεχνικών ανακατασκευής επιφανειών σχεδίασης [4], [5]. Για να ολοκληρωθεί αυτή η διαδικασία πρέπει να μετατραπεί σε τριγωνικό ή πολυγωνικό πλέγμα ή σε μοντέλα επιφανειών NURBS ή και σε άλλα ειδή ορισμού επιφάνειας. Η διαδικασία αυτή ονομάζεται ανασυγκρότηση επιφάνειας (wrapping). Υπάρχουν πολλές τεχνικές για αυτή την διαδικασία όπως η Delaunay[5] η οποία κατασκευάζει ένα δίκτυο τριγώνων με κορυφές τα υπάρχουσα σημεία του νέφους [Εικόνα 2.1]. Τελικά το παραγόμενο μοντέλο απεικονίζει την επιφάνεια του αρχικού αντικειμένου μέσω ενός μεγάλου πλήθους τριγώνων χρησιμοποιώντας τα σημεία του νέφους ως κορυφές.



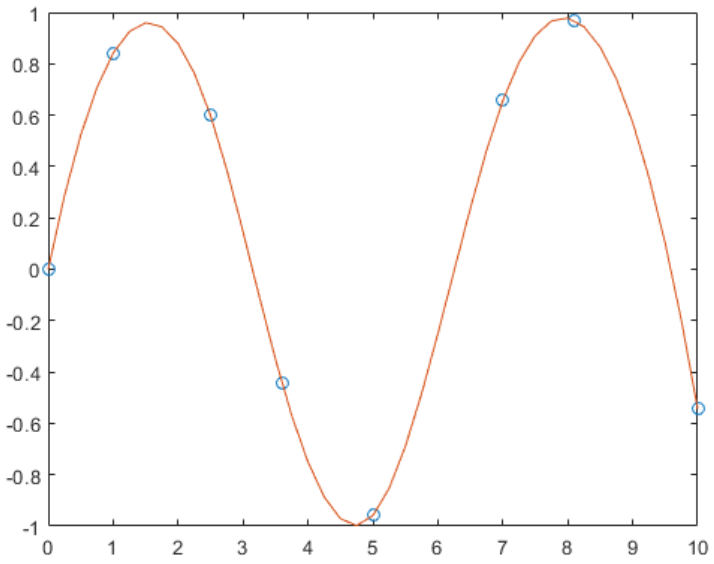
Εικόνα 2.1 Ανασυγκρότηση επιφάνειας με τη χρήση του MeshLab[6]

2.3 Εύκαμπτες καμπύλες

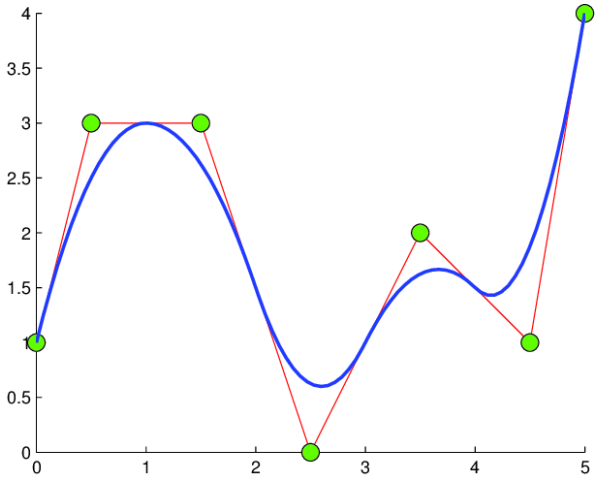
Στον χώρο των γραφικών μια εύκαμπτη καμπύλη θεωρείται οποιαδήποτε συνθέτη καμπύλη που σχηματίζεται από τμήματα πολυωνύμου ικανοποιεί κάποιες συνθήκες συνέχειας και κάποιους μαθηματικούς περιορισμούς. Υπάρχουν πολλοί διαφορετικοί ορισμοί για την περιγραφή των εύκαμπτων καμπυλών. Το κάθε είδος αναφέρεται σε διαφορετικό τύπο πολυωνύμου και με διαφορετικούς περιορισμούς. Ο βαθμός του πολυωνύμου επηρεάζει άμεσα το σχήμα και την πορεία της καμπύλης. Όσο ο βαθμός του

πολυωνύμου γίνεται μεγαλύτερος δίνει στην καμπύλη την δυνατότητα να έχει πιο περίπλοκο σχήμα αλλά να γίνεται επίσης πιο περιπλοκή και στον υπολογισμό. Ένα χαρακτηριστικό των ευκάμπτων καμπύλων είναι η ομαλή και λεία μετάβαση σε όλο το μήκος της καμπύλης

Μια εύκαμπτη καμπύλη (spline) ορίζεται από ένα σύνολο σημείων ελέγχου (control points) τα οποία μας δίνουν μια γενική εικόνα για την καμπύλη [3], [7]. Αυτά τα σημεία ελέγχου ορίζονται από τον χρήστη και δημιουργείται μια ομαλή καμπύλη που μπορεί να ελεγχθεί μέσο αυτών των σημείων. Σε μερικά είδη συναρτήσεων καμπυλών τα σημεία ελέγχου αποτελούν μέρος της καμπύλης άλλα σε μερικά όμως όχι. Στην $1^{η}$ περίπτωση λέγεται πως η καμπύλη που δημιουργείται αποτελεί παρεμβολή των σημείων ελέγχου [Εικόνα 2.2] . Ενώ στην $2^{η}$ περίπτωση λέγεται ότι η καμπύλη προσεγγίζει τα σημεία ελέγχου [Εικόνα 2.3].



Εικόνα 2.2 Εύκαμπτη καμπύλη με παρεμβολή



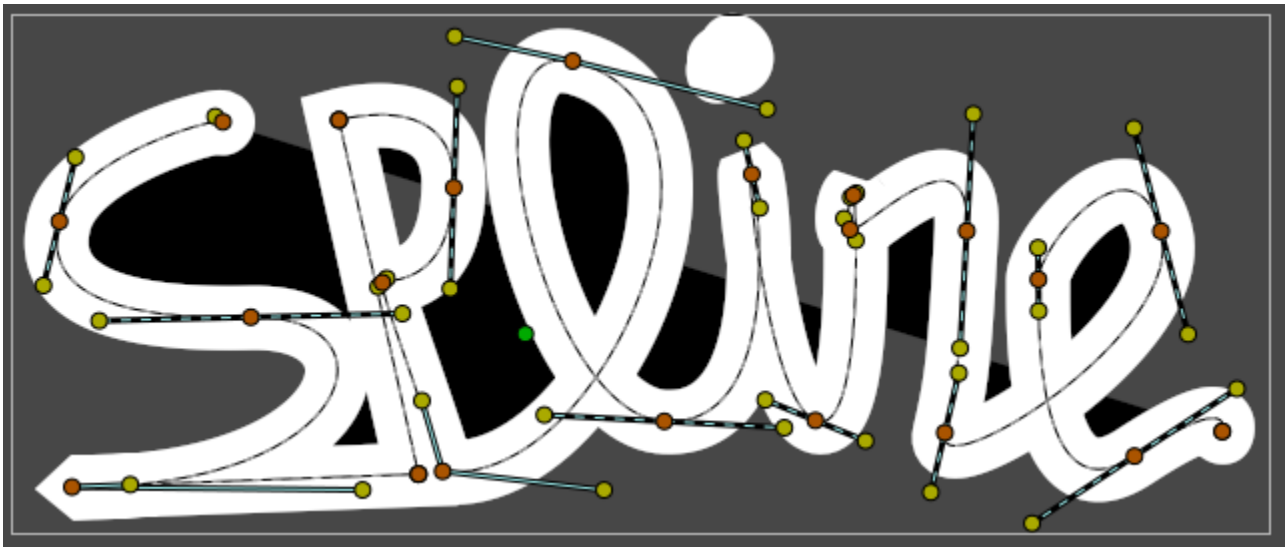
Εικόνα 2.3 Εύκαμπτη καμπύλη με προσέγγιση

Ένας τρόπος για να ορίσουμε την θέση την εύκαμπτης καμπύλης στο χώρο είναι το κυρτό περίβλημα (convex hull) , όταν αναφερόμαστε σε δισδιάστατη καμπύλη, που ορίζεται από τα σημεία ελέγχου της [3]. Ενώ όταν αναφερόμαστε σε τρισδιάστατη καμπύλη τότε λέγεται κυρτό πολύεδρο. Επίσης από τα σημεία ελέγχου της καμπύλης μπορεί να οριστεί το γράφημα ελέγχου ή αλλιώς πολύγωνο ελέγχου ή χαρακτηριστικό πολύγωνο. Αυτό το πολύγωνο μας δίνει μια οπτική προσέγγιση για την πορεία που θα ακολουθήσει η καμπύλη και αποτελείται από ευθύγραμμα τμήματα των διασυνδεδεμένων σημείων ελέγχου σε διαδοχική σειρά [3].

Οι εύκαμπτες καμπύλες είναι ιδιαίτερα χρήσιμες σε προγράμματα σχεδίασης γραφικών καθώς βοηθάνε να υπάρχει καλύτερος έλεγχος των καμπύλων γραμμών κατά την σχεδίαση εικόνων [Εικόνα 2.4] καθώς και στην σχεδίαση 3D μοντέλων. Επίσης μπορούν να βοηθήσουν στην δημιουργία κάποιου animation

σχεδιάζοντας την πορεία που πρέπει να ακολουθήσει το αντικείμενο του animation. Μια ακόμα εφαρμογή αυτών των καμπύλων είναι η σχεδίαση καλλιτεχνικών γραμματοσειρών και γραμματοσειρών για περίπλοκα σχήματα [Εικόνα 2.5].

Εικόνα 2.4 Καμπύλες σε προγράμματα σχεδίασης γραφικών



Εικόνα 2.5 Γραμματοσειρά από εύκαμπτες καμπύλες

Στη συνέχεια θα αναλυθούν με περισσότερη λεπτομέρεια οι καμπύλες Bezier και B-spline ενώ θα γίνει και μια εισαγωγή στις ρητές καμπύλες και πιο συγκεκριμένα στις ρητές καμπύλες Bezier. Τέλος, θα αναφερθούν και οι επιφάνειες Bezier ως επέκταση των αντίστοιχων καμπυλών.

Παρακάτω δίνεται η γενική μορφή της εξίσωσης εύκαμπτων καμπυλών με τη χρήση πολυωνύμου και οριακών συνθηκών για την μεταβλητή της συνάρτησης:

$$x(u) = \sum_{k=0}^{n} \mathrm{g_k} BF(u)$$

Τα πολυώνυμα BF ονομάζονται συναρτήσεις μίξης (blending functions) ή συναρτήσεις βάσης (basis functions) της εύκαμπτης καμπύλης. Ο όρος g_k ορίζει ένα γεωμετρικό περιορισμό σε σχέση με τα σημεία ελέγχου της καμπύλης [3].

2.4 Καμπύλες Bezier

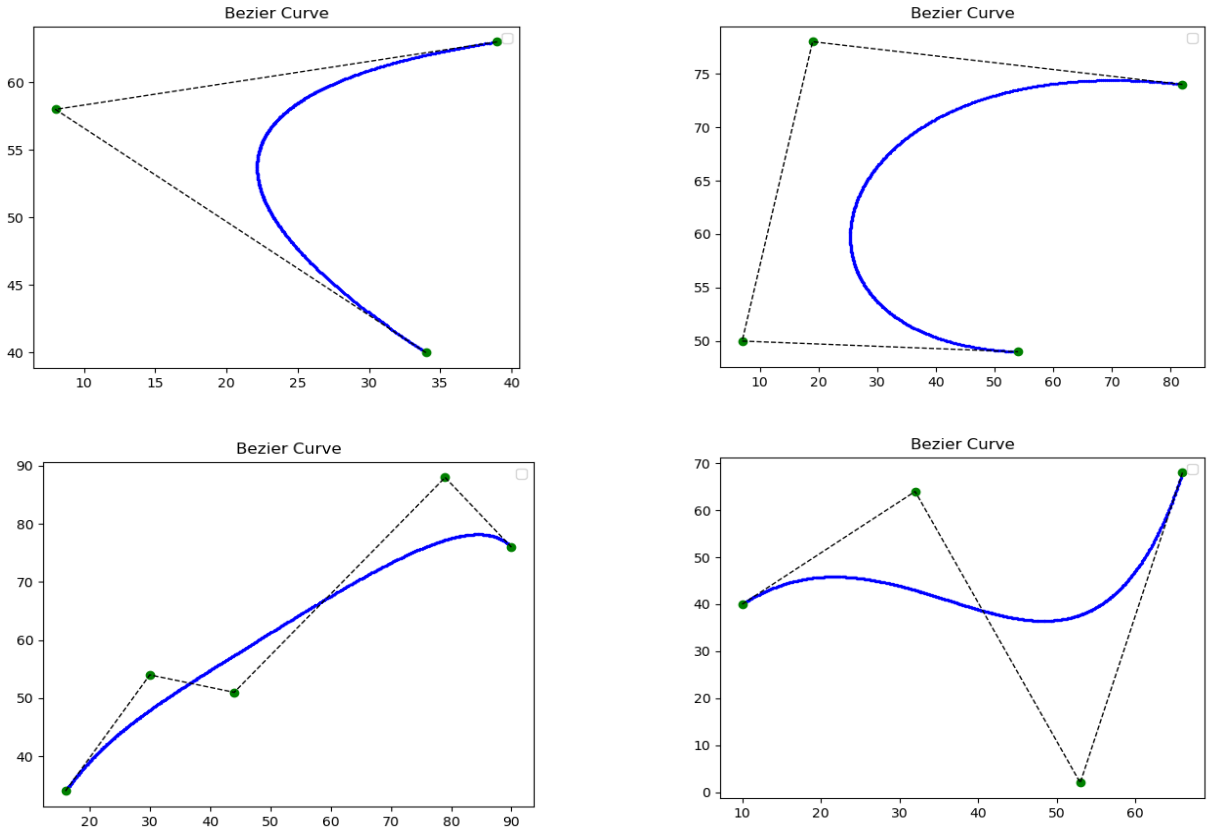
Οι καμπύλες Bezier δημιουργήθηκαν ταυτόχρονα από τον Γάλλο μαθηματικό Pierre E. Bezier και τον Paul de Casteljau το 1960 για να βοηθήσουν στον σχεδιασμό σκαφών αυτοκινήτων στις εταιρείες Renault και Citroen αντίστοιχα [3], [8], [9]. Είναι από τις πιο διαδεδομένες καμπύλες στον χώρο των γραφικών υπολογιστών λόγω του εύκολου χειρισμού τους και του μικρού κόστους σχεδίασης σε σχέση με τις παραμετρικές καμπύλες που χρησιμοποιούσαν εκείνη την εποχή. Αλλιώς ονομάζονται και εύκαμπτες καμπύλες Bezier (Bezier spline) γιατί ανήκουν στην γενική κατηγορία των εύκαμπτων καμπύλων και αποτελούν ειδική περίπτωση των καμπύλων Basis spline (ή B-spline για συντομία). Οι καμπύλες Bezier χρησιμοποιούνται κυρίως από προγράμματα σχεδίασης γραφικών όπως Photoshop, Inkscape , GIMP κλπ. για να περιγράψουν πιο πολύπλοκα δισδιάστατα σχήματα. Επίσης μπορούν να επεκταθούν σε περισσότερες διαστάσεις και περιγράψουν κυρτές ή κοίλες επιφάνειες και τότε ονομάζονται επιφάνειες Bezier.

Η καμπύλη Bezier ορίζεται από μια παραμετρική εξίσωση [1] η οποία χρησιμοποιεί τα πολυώνυμα Bernstein ως συνάρτηση βάσης και περιέχει ένα σύνολο σημείων ελέγχου P_0 έως P_{n+1}. Οι καμπύλες Bezier ξεκινάνε από το πρώτο σημείο ελέγχου και καταλήγουν στο τελευταίο και η πορεία της καμπύλης επηρεάζεται από ενδιάμεσα σημεία ελέγχου.

Ο βαθμός του πολυωνύμου *n* προκύπτει από τον αριθμό των σημείων ελέγχου (n =1 γραμμική, n=2 τετραγωνική, n=3 κυβική). Σπάνια χρησιμοποιούνται καμπύλες Bezier βαθμού μεγαλύτερου από 5 διότι το κόστος σχεδίασης αυξάνεται αναλογικά με τον βαθμό του πολυωνύμου.
Στην γενική της μορφή για n+1 σημείων ελέγχου η εξίσωση έχει την έξης μορφή [3]:

$$\mathrm{B}ez(t) = \sum_{i=0}^{n} B\,{}_i^n(t) Pi$$

Όπου B(t) είναι τα πολυώνυμα Bernstein και P τα σημεία ελέγχου



Εικόνα 2.6 Παραδείγματα καμπυλών Bezier

2.4.1 Πολυώνυμα Bernstein

Τα πολυώνυμα Bernstein είναι ένα πολύ χρήσιμο μαθηματικό εργαλείο που δημιουργήθηκε από τον μαθηματικό Sergei Natanovich Bernstein. Αυτά τα πολυώνυμα χρησιμοποιούνται αρκετά στην επιστήμη των υπολογιστών διότι έχουν την ιδιότητα να ορίζονται και υπολογίζονται ευκολά, ακόμα έχουν την ικανότητα να παριστάνουν ένα μεγάλο πλήθος συναρτήσεων.

Τα πολυώνυμα Bernstein στην γενική μορφή τους έχουν την εξής εξίσωση [10]:

$$B_i^n(t) = \binom{n}{i} \frac{(t-a)^i (b-t)^{n-i}}{(b-a)^n}, \qquad i = 0,1,\dots,n \quad \kappa\alpha\iota \quad t \in [a,b]$$

Αν θεωρήσουμε α = 0 και b = 1 τότε η παραπάνω έκφραση παίρνει την ακόλουθη μορφή:

$$B_i^n(t) = \binom{n}{i}(t)^i(1-t)^{n-i}, \qquad i = 0,1,\dots,n \quad και \quad t \in [0,1]$$

Ο διωνύμικος συντελεστής ορίζεται ως εξής:

$$\binom{n}{i} = \frac{n!}{i!\,(n-i)!}$$

Αυτή η μορφή του πολυωνύμου ορίζεται ως συνάρτηση βάσης για την για την καμπύλη Bezier.Μια πολύ σημαντική ιδιότητα των πολυωνύμων Bernstein που παίζει μεγάλο ρόλο στον υπολογισμό της καμπύλης είναι ότι το άθροισμα τους ισούται πάντα με 1. Αυτή η ιδιότητα ονομάζεται partition of unity. Η ένωση πολλών καμπυλών Bezier ονομάζεται Bezier spline.

$$\sum_{i=0}^{n} B_i^n = \sum_{i=0}^{n} \binom{n}{i}(t)^t(1-t)^{n-i} = \big(x + (1-x)\big)^n = 1$$

Το πολυώνυμο Bernstein τρίτου βαθμού ορίζεται από 4 συναρτήσεις η οποίες αθροίζονται και είναι:

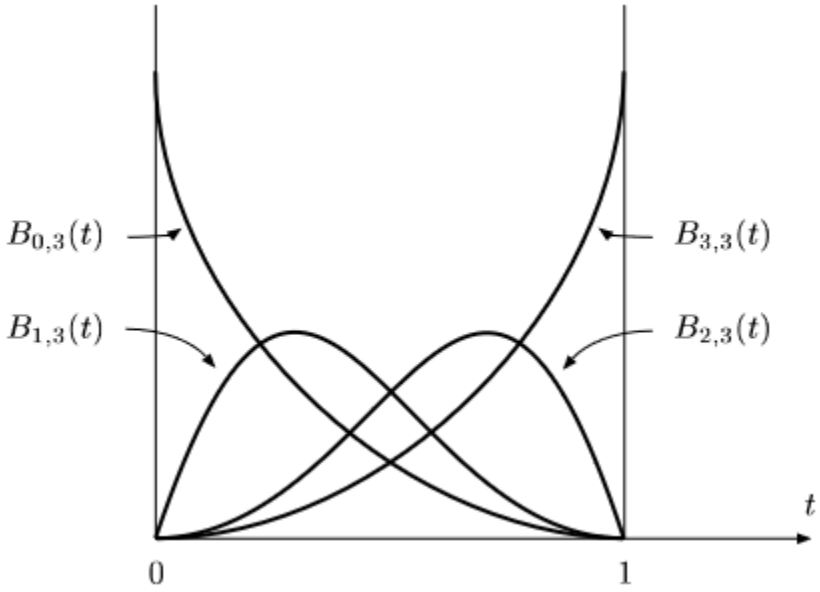
$$B_{0,3}(t) = (1-t)^3$$

$$B_{1,3}(t) = 3t(1-t)^2$$

$$B_{2,3}(t) = 3t^2(1-t)$$

$$B_{3,3}(t) = t$$

Η γραφική παράσταση τους για το διάστημα [0,1] προκύπτει στην [Εικόνα 2.7]



Εικόνα 2.7 Πολυώνυμο Bernstein τρίτου βαθμού

2.4.2 Αλγόριθμος De Casteljau

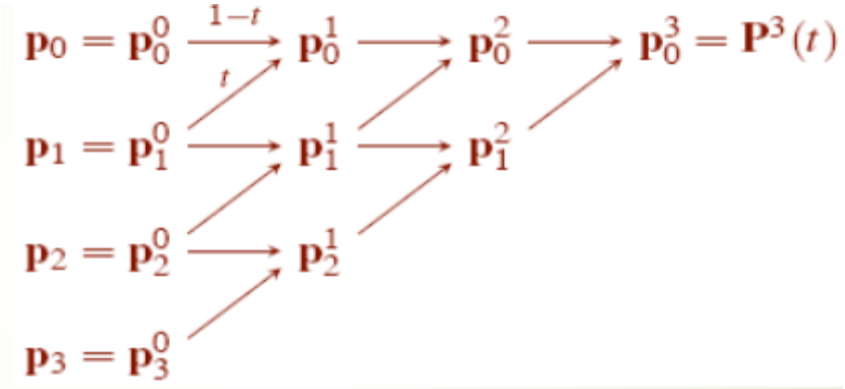
Η καμπύλη Bezier μπορεί επίσης να οριστεί μέσω γραμμικών παρεμβολών. Η γραμμική παρεμβολή ορίζεται ανάμεσα σε 2 σημεία και ένα ποσοστό καθορίζει το σημείο της παρεμβολής. Το ποσοστό αυτό μας δίνει σε ποιο σημείο βρισκόμαστε ανάμεσα στα δύο άκρα αυτού του ευθύγραμμου τμήματος. Αν αυτό το ποσοστό είναι 40% τότε αναφερόμαστε στο σημείο που απέχει στο 40% της απόστασης από το πρώτο σημείο και στο 60% της απόστασης από το δεύτερο. Η γραμμική παραβολή χρειάζεται να εφαρμοστεί πολλές φορές ανάμεσα στα σημεία ελέγχου της καμπύλης ώστε τελικά να υπολογιστεί το σημείο της καμπύλης.

Αυτός ο τρόπος υπολογισμού ονομάζεται αλγόριθμος de Casteljau διότι επινοήθηκε από τον μαθηματικό Paul de Casteljau ως ένας τρόπος υπολογισμού των πολυωνύμων Bernstein και των καμπύλων Bezier [2] . Ο αλγόριθμος αυτός είναι ένας πολύ αποδοτικός τρόπος εύρεσης των σημείων της καμπύλης Bezier χωρίς να απαιτούνται πολύπλοκες πράξεις με αρκετούς επεξεργαστικούς κύκλους.

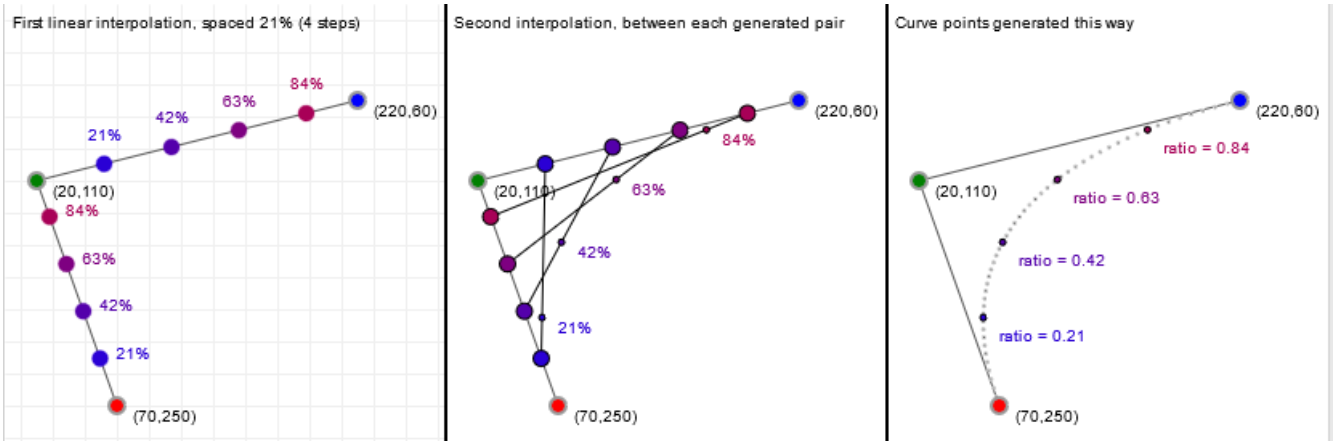
Για υπολογίσουμε τα σημεία της καμπύλης πρέπει να εφαρμοστούν γραμμικές παρεμβολές σε κάθε ευθύγραμμο τμήμα του πολυγώνου ελέγχου [Εικόνα 2.9 αριστερά] καθώς και στα τμήματα που δημιουργούνται μεταξύ των σημείων της παρεμβολής [Εικόνα 2.9 μέση]. Το τελευταίο σημείο που απομένει είναι το σημείο της καμπύλης [Εικόνα 2.9 δεξιά]. Ο αριθμός των παρεμβολών που θα χρειαστούν είναι και ο βαθμός της καμπύλης και ορίζεται από τον αριθμό των σημείων ελέγχου της καμπύλης.

Για να υπολογίσουμε την καμπύλη θα πρέπει να οριστεί ένα βήμα για την παρεμβολή το οποίο θα διατρέχει το διάστημα ορισμού της καμπύλης [0,1] . Όσο μικρότερο είναι το βήμα τόσο πιο πολλά σημεία υπολογίζονται σχεδιάζοντας την καμπύλη με μεγαλύτερη ακρίβεια.

Για την πιο εύκολη κατανόηση του αλγορίθμου μπορεί να δημιουργηθεί ένα τρίγωνο με τα σημεία που περιλαμβάνονται στο αλγόριθμο [Εικόνα 2.8] και ο τρόπος υπολογισμού του κάθε σημείου φαίνεται στην [Εικόνα 2.9]. Για παράδειγμα μια κυβική καμπύλη Bezier θα χρειαστεί 3 επίπεδα παρεμβολών στο πρώτο επίπεδο τα σημεία $P_{0,0}$ με $P_{0,1}$, $P_{0,1}$ με $P_{0,2}$, $P_{0,2}$ με $P_{0,3}$. Στο δεύτερο επίπεδο η παρεμβολή θα γίνει μεταξύ των σημείων του δημιουργήθηκαν δηλαδή $P_{1,0}$ με $P_{1,1}$, $P_{1,1}$ με $P_{1,2}$. Στο τελευταίο επίπεδο θα γίνει μεταξύ $P_{2.0}$ και $P_{2,1}$ και το αποτέλεσμα θα είναι το σημείο την καμπύλης.



Εικόνα 2.8 Αλγόριθμος De Casteljau για κυβική καμπύλη Bezier



Εικόνα 2.9 Οπτική αναπαράσταση του αλγορίθμου De Casteljau

2.4.3 Ιδιότητες των καμπυλών Bezier

Οι καμπύλες Bezier διαθέτουν αρκετές χρήσιμες ιδιότητες[3]:

1. Για κάθε πολυωνυμική καμπύλη οποιουδήποτε βαθμού υπάρχει μια καμπύλη Bezier που μπορεί να την αντιπροσωπεύσει.
2. Το σχήμα της καμπύλης Bezier δεν επηρεάζεται από μετασχηματισμούς στα σημεία ελέγχου της. Για να μετασχηματιστεί η καμπύλη αρκεί να εφαρμοστεί ο μετασχηματισμός στα σημεία ελέγχου της και να ξανά υπολογιστεί η καμπύλη.
3. Η καμπύλη Bezier παραμένει αμετάβλητη σε μετασχηματισμούς της παραμετρική της μεταβλητής, δηλαδή αν το πεδίο ορισμού του t αλλάξει από $t \in [0,1]$ σε $t \in [\alpha, \beta]$
4. Η καμπύλη Bezier είναι συμμετρική ως προς τα σημεία ελέγχου: δηλαδή η εξίσωση μπορεί να εφαρμοστεί στα σημεία με αντίστροφη σειρά χωρίς να επηρεαστεί το σχήμα της καμπύλης. Επομένως η καμπύλη μπορεί να διασχιστεί και με αντίθετη κατεύθυνση.
5. Η καμπύλη Bezier έχει γραμμική ακρίβεια, δηλαδή αν τα σημεία ελέγχου είναι συνευθειακά τότε η καμπύλη θα έχει το σχήμα ευθύγραμμου τμήματος.
6. Για κάθε καμπύλη Bezier ισχύει η ιδιότητα φθίνουσας διακύμανσης: δηλαδή μια ευθεία μπορεί να τέμνει την καμπύλη σε όχι περισσότερες φορές από ότι τέμνει το πολύγωνο ελέγχου της. Επίσης αν μια καμπύλη έχει κυρτό πολύγωνο ελέγχου τότε και η ίδια είναι κυρτή χωρίς να ισχύει πάντα το αντίθετο.
7. Η καμπύλη Bezier ορίζεται ανάμεσα στο πρώτο και στο τελευταίο σημείο ελέγχου και αυτά είναι πάντα τα δύο άκρα της καμπύλης.

$$\gamma\iota\alpha\ t = 0 \quad Bez(0) = \sum_{i=0}^{n} B\,{}_i^n(t)Pi = P_0$$

$$\gamma\iota\alpha\ t = 1 \quad Bez(1) = \sum_{i=0}^{n} B\,{}_i^n(t)Pi = P_n$$

Τα ενδιάμεσα σημεία ελέγχου βρίσκονται πάνω στην καμπύλη εκτός αν τα σημεία ελέγχου είναι συνευθειακά.

8. Η πρώτη παράγωγος μιας καμπύλης Bezier στο πρώτο και στο τελευταίο σημείο (t=0 και t=1) ταυτίζεται με το πολύγωνο ελέγχου στα συγκεκριμένα σημεία.

$$Bez'(0) = -nP_0 + nP_1$$

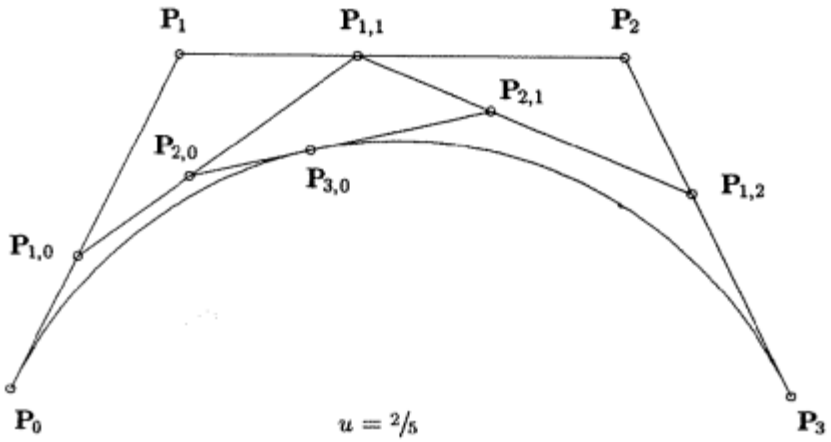
$$Bez'(1) = -nP_{n-1} + nP_n$$

9. Από την παραπάνω ιδιότητα μπορεί να υπολογιστεί η κλίση της καμπύλης ανάμεσα στα δυο πρώτα σημεία ελέγχου και στα δυο τελευταία.
10. Για κάθε τιμή της παραμετρικής μεταβλητής t, το άθροισμα των συναρτήσεων μίξης, δηλαδή των πολυωνύμων Bernstein, είναι πάντα ίσο με 1.
11. Αφού οι συναρτήσεις μίξης μιας καμπύλης Bezier είναι πάντα θετικές και το άθροισμα τους πάντα ισούται με 1 η καμπύλη θα βρίσκεται πάντα μέσα στο κλειστό πολύγωνο που ορίζουν τα σημεία ελέγχου της.
12. Στις καμπύλες Bezier δεν μπορεί να εφαρμοστεί τοπικός έλεγχος δηλαδή: ένα σημείο της καμπύλης να επηρεάζεται μόνο από μερικά γειτονικά σημεία ελέγχου και η αλλαγή θέσης ενός σημείου ελέγχου να επηρεάζει τοπικά τα σημεία της καμπύλης. Αυτό ισχύει διότι τα πολυώνυμα Bernstein δεν είναι μηδενικά σε κανένα σημείο του διαστήματος [0,1] της καμπύλης.
13. Η μετακίνηση ενός σημείου ελέγχου έχει ως αποτέλεσμα την τροποποίηση του σχήματος όλης της καμπύλης
14. Υπάρχει εύκολη πρόβλεψη για το σχήμα της καμπύλης κατά της μετακίνηση των σημείων ελέγχου.

2.4.4 Υποδιαίρεση των καμπύλων Bezier

Η λογική πίσω από την υποδιαίρεση των καμπύλων είναι η διαχώριση μια καμπύλης Bezier σε δυο τμήματα Bezier ίδιου βαθμού με νέα σημεία ελέγχου [3]. Η υποδιαίρεση καμπυλών αποδεικνύεται πολύ χρήσιμη διαδικασία διότι ελαχιστοποιεί το υπολογιστικό κόστος σχεδίασης ακόμα και για πιο σύνθετα σχήματα καθώς το κόστος υπολογισμού της καμπύλης Bezier αυξάνεται αναλογικά με τον βαθμό του πολυωνύμου. Η διαδικασία της υποδιαίρεσης μπορεί να εφαρμοστεί σε μια καμπύλη αναδρομικά πολλές φορές. Παραδείγματα εφαρμογής της υποδιαίρεσης καμπυλών είναι ο υπολογισμός της τομής δυο καμπυλών Bezier, και η σχεδίαση των καμπυλών στην οθόνη.

Ο αλγόριθμος De Casteljau μας δίνει όλες τις απαραίτητες πληροφορίες για τον υπολογισμό των νέων σημείων ελέγχου. Στην [Εικόνα 2.10] φαίνεται μια κυβική καμπύλη Bezier. Οποιαδήποτε παραμετρική τιμή χωρίζει της καμπύλη σε δυο τμήματα.



Εικόνα 2.10 Υποδιαίρεση των καμπυλών Bezier με χρήση του αλγορίθμου De Casteljau

Τα σημεία ελέγχου για το αριστερό τμήμα είναι P_0, $P_{1,0}$, $P_{2,0}$ και $P_{3,0}$. Ενώ τα σημεία για το δεξί τμήμα είναι $P_{3,0}$,$P_{2,1}$, $P_{1,2}$, P_3

2.4.5 Συνθήκες συνέχειας

Στις περισσότερες περιπτώσεις, όταν αναφερόμαστε σε νέφη σημείων πραγματικών αντικειμένων, η αναπαράσταση των καμπυλωτών τμημάτων τους δεν είναι δυνατή με εύκαμπτες καμπύλες χαμηλού βαθμού [3]. Αυτό γιατί η καμπύλες χαμηλού βαθμού έχουν περιορισμένες δυνατότητες ευελιξίας και τοπικό έλεγχο. Η αύξηση του βαθμού του πολυωνύμου μια παραμετρικής καμπύλης δίνει από την μια περισσότερο έλεγχο πάνω στην καμπύλη αλλά από την άλλη κάνει τον υπολογισμό της αρκετά πιο πολύπλοκο και χρονοβόρο. Το πρόβλημα αυτό είναι δυνατόν να ξεπεραστεί με τη χρήση και ένωση πολλών διαδοχικών τμημάτων χαμηλόβαθμων καμπυλών. Η ένωση διαδοχικών τμημάτων μπορεί να επιτευχθεί με τον ορισμό συνθηκών συνέχειας ώστε να υπάρχει μια ομαλή μετάβαση μεταξύ των δυο διαδοχικών τμημάτων.

Για την εξασφάλιση ομαλής ένωσης δυο καμπύλων πρέπει να οριστούν συνθήκες συνέχειας στο σημείο τομής των δυο τμημάτων [3], [11] . Αυτές οι συνθήκες ονομάζονται συνθήκες παραμετρικής συνέχειας και δείχνουν πόσο ομαλή είναι η ένωση των τμημάτων. Εφόσον οι εύκαμπτες καμπύλες ορίζονται από παραμετρικές εξισώσεις, για κάθε συντεταγμένη στο σημείο επαφής, ορίζεται μια συνάρτηση της παραμετρικής μεταβλητής. Οι παράγωγοι των σημείων αυτών και από τα δυο τμήματα ορίζουν τον βαθμό την παραμετρικής συνέχειας. Η συνέχεια μηδενικής τάξης C_0 σημαίνει ότι οι τιμές των συναρτήσεων στο σημείο τομής είναι ίσες.

$$Bez_1(t_1) = Bez_2(t_2)$$

Η συνέχεια πρώτης τάξης C_1 προϋποθέτει την C_0 και επιπλέον οι πρώτες παράγωγοι των συναρτήσεων είναι ίσες δηλαδή οι εφαπτόμενες των δύο συναρτήσεων ταυτίζονται.

$$Bez_1'(t_1) = Bez_2'(t_2)$$

Η συνέχεια δεύτερης τάξης C_2 προϋποθέτει την C_1 και επιπλέον οι δεύτερες παράγωγοι είναι ίσες. Ως αποτέλεσμα, η συνέχεια C_2 σημαίνει πως η καμπυλότητα της συνάρτησης συνεχίζει να είναι η ίδια μεταξύ των δυο τμημάτων.

$$Bez_1''(t_1) = Bez_2''(t_2)$$

Συνήθως η συνέχεια πρώτης τάξης είναι αρκετή για την δημιουργία μιας ομαλής καμπύλης άλλα υπάρχουν εφαρμογές που χρησιμοποιούν και δεύτερης τάξης.

Εκτός από την παραμετρική συνέχεια υπάρχει και η γεωμετρική συνέχεια για την οποία δεν απαιτείται ισότητα για τις παραγώγους άλλα μόνο αναλογία μεταξύ τους. Η γεωμετρική συνέχεια μηδενικής τάξης G_0 είναι ταυτόσημη με την παραμετρική συνέχεια μηδενικής τάξης. Η γεωμετρική συνέχεια πρώτης τάξης G_1 G_1 προϋποθέτει την G_0 και επιπλέον οι παράγωγοι των συναρτήσεων έχουν την ίδια μορφή (ίσους συντελεστές) άλλα όχι απαραίτητα ίσες τιμές. Το αντίστοιχο ισχύει και για την γεωμετρική συνέχεια δεύτερης τάξης G_2

2.5 Εύκαμπτες καμπύλες B-spline

Οι εύκαμπτες καμπύλες B-spline [7] αποτελούν γενίκευση των καμπυλών Bezier. Οι καμπύλες αυτές βρίσκουν μεγαλύτερη εφαρμογή σε σχέση με τις καμπύλες Bezier και είναι πιο εκτεταμένη η χρήση τους σε προγράμματα σχεδίασης γραφικών (CAD). Όπως όλες οι εύκαμπτες καμπύλες έτσι και οι εύκαμπτες καμπύλες B-spline παράγονται με την χρήση σημείων ελέγχου. Οι καμπύλες B-Spline πλεονεκτούν σε σχέση με τις καμπύλες Bezier καθώς ο βαθμός του πολυωνύμου καμπύλης δεν εξαρτάται από των αριθμό των σημείων ελέγχου και επιτρέπουν καλύτερο και πιο στοχευμένο έλεγχο πάνω στην καμπύλη. Τα πλεονεκτήματα αυτά τις κάνουν πιο αποτελεσματικές, αποδοτικές και χρήσιμες στον χώρο της σχεδίασης γραφικών όμως ο ορισμός τους και η χρήση τους είναι εμφανώς πιο περίπλοκος.

2.5.1 Ορισμός των συναρτήσεων μίξης

Οι καμπύλες B-spline παράγονται από την συνένωση παραμετρικών καμπυλών με περιορισμούς συνέχειας. Αποτελούνται από πολυωνυμικά τμήματα βαθμού k συνενωμένα με συνέχεια C_{k-1}. Ο βαθμός k των τμημάτων είναι επίσης ο βαθμός της καμπύλης B-spline. Οι καμπύλες B-spline ορίζονται από σημεία ελέγχου P_0, P_1, …, P_n, το πλήθος των οποίων (n+1) είναι ανεξάρτητο από τον βαθμό της καμπύλης και σχετίζεται με το πλήθος των πολυωνυμικών τμημάτων. Τα πολυωνυμικά τμήματα ορίζονται σε διαδοχικά παραμετρικά υποδιαστήματα $[u_i, u_{i+1}]$. Ως εκ τούτου το πεδίο ορισμού της καμπύλης B-spline είναι η ένωση όλων των υπόδιαστημάτων $[u_{min}, u_{max}]$.

Οι καμπύλες B-Spline έχουν σημαντικές διαφορές στον ορισμό τους με τις Bezier. Οι συναρτήσεις μίξης έχουν ως παράμετρο το u που ανήκει στο διάστημα $[u_{min}, u_{max}]$. Το συγκεκριμένο διάστημα ονομάζεται knot vector και τα άκρα του προκύπτουν από τις υπόλοιπες μεταβλητές της καμπύλης. Αντιθέτως, η παράμετρος t της Bezier ανήκει πάντα στο διάστημα [0,1]. Επίσης, το γεγονός ότι οι συναρτήσεις μίξης μπορεί να είναι μηδενικές έχει ως αποτέλεσμα τον πολύ καλό τοπικό έλεγχο.

Κάθε άκρο του ενός διαστήματος του u ονομάζεται κόμβος (knot) και το σύνολο των άκρων ονομάζεται διάνυσμα κόμβων (knot vector). Επομένως, οι κόμβοι διαιρούν όλο το διάστημα σε υπό διαστήματα. Όταν η απόσταση μεταξύ των κόμβων είναι σταθερή τότε η καμπύλη ονομάζεται ομοιόμορφη (uniform) σε αντίθετη περίπτωση ονομάζεται ανομοιόμορφη (non-uniform).

Ο βαθμός του πολυωνύμου της συνάρτησης μίξης N(u) είναι d, η παράμετρος βαθμού (degree parameter). Το εύρος τιμών της παραμέτρου d κυμαίνεται από 1 μέχρι το πλήθος των σημείων ελέγχου. Για να επιτευχθεί καλύτερος έλεγχος, οι καμπύλες B-spline χρησιμοποιούν διαφορετική συνάρτηση μίξης σε υπό διαστήματα του u.

$$N_{k,0}(u) = \begin{cases} 1 & u \in [u_k, u_{k+1}) \\ 0 \ στις \ υπόλοιπες \ περιπτωσεις \end{cases}$$

$$N_{k,d}(u) = \frac{u - u_k}{u_{k+d} - u_k} \cdot N_{k,d-1}(u) + \frac{u_{k+d+1} - u}{u_{k+d+1} - u_{k+1}} \cdot N_{k+1,d-1}(u)$$

Η παραπάνω εξίσωση είναι γνώστη ως αναδρομική φόρμουλα Cox-de Boor [7]. Πάρα την περιπλοκή εμφάνιση της αυτό που μας δείχνει η πρώτη εξίσωση είναι πως για συνάρτηση μίξης μηδενικού βαθμού (d=0), η συνάρτηση μίξης είναι ίση με 1 στο διάστημα με αύξοντα αριθμό k ενώ είναι ίση με 0 σε όλα

τα υπόλοιπα υποδιαστήματα. Η δεύτερη εξίσωση εφαρμόζεται για συνάρτησης μίξης μεγαλύτερου βαθμού από 0 και για τον υπολογισμό του χρειάζονται οι τιμές $N_{k,d-1}(u)$ και $N_{k+1,d-1}(u)$ οι οποίες είναι οι συναρτήσεις του προηγουμένου βαθμού.

2.5.2 Ορισμός της καμπύλης B-spline

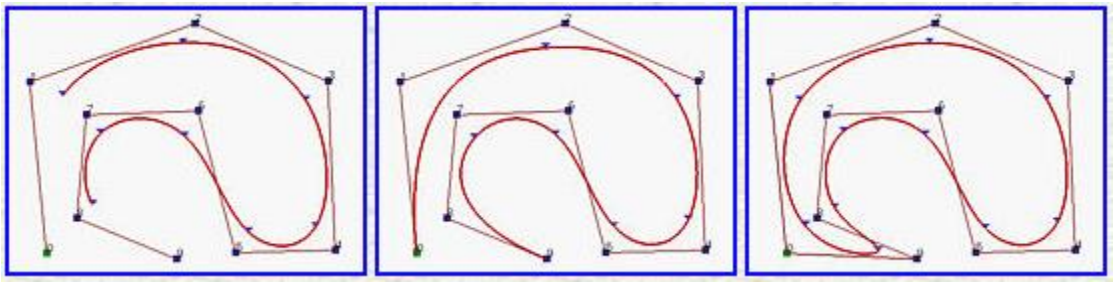
Η γενική εξίσωση υπολογισμού της καμπύλης B-spline ορίζεται χρησιμοποιώντας συνάρτηση μίξης.

$$\mathrm{P(u)} = \sum_{k=0}^{n} P_k \cdot N_{k,d}(u) \quad \mathrm{u} \in [u_{min}, \mathrm{u_{max}}] \qquad ,2 < d < n+1$$

Όπου P_k είναι το σύνολο εισόδου των n+1 σημείων ελέγχου και d είναι ο βαθμός της συνάρτησης μίξης.

Ο καμπύλες B-spline είναι παρόμοιας λογικής με τις καμπύλες Bezier. Για να σχεδιαστεί μια καμπύλη B-spline απαιτούνται περισσότερα στοιχεία όπως: n+1 σημεία ελέγχου, m+1 διανύσματα κόμβων (knot vector) και τον βαθμό d των συντελεστών για το κάθε σημείο ελέγχου. Ο τρόπος συσχέτισης αυτών των στοιχείων πρέπει να ικανοποιεί της σχέση m = n + d + 1, που σημαίνει ότι για μια καμπύλη με βαθμό d και n+1 σημεία ελέγχου πρέπει να του προσδιοριστούν n + d + 2 κόμβοι $(u_0, u_1, \dots, u_{n+d+2})$.Το σημείο της καμπύλης που βρίσκεται ο κόμβος ονομάζεται σημείο κόμβου (knot point). Ως εκ τούτου τα σημεία κόμβου χωρίζουν την καμπύλη σε τμήματα πού το καθένα ορίζεται μέσα στο αντίστοιχο διάνυσμα κόμβου. Καθένα από αυτά τα τμήματα μπορεί να αποδειχθεί πως είναι μια καμπύλη Bezier βαθμού d.

Το τελικό σχήμα των καμπύλων B-spline μπορεί να έχει διαφορές ανάλογα το πλήθος των κόμβων και των σημείων ελέγχου που έχουν επιλεχθεί. Επομένως, αν μία καμπύλη δεν διαθέτει στο τέλος και στην αρχή του διανύσματος των κόμβων d+1 κόμβοι τότε η καμπύλη δεν θα εφάπτεται στο πρώτο και τελευταίο σημείο ελέγχου [Εικόνα 2.11 αριστερά]. Στην περίπτωση που μια καμπύλη έχει d+1=4 κόμβους στην αρχή και στο τέλος του διανύσματος κόμβων, τότε οι πρώτοι 4 και οι 4 τελευταίοι κόμβοι πρέπει να είναι ίδιοι μεταξύ τους [Εικόνα 2.11 μέση]. Όταν το πρώτο σημείο ελέγχου είναι διαφορετικό από το τελευταίο τότε δημιουργείται ανοιχτή καμπύλη, ενώ όταν τα δυο αυτά σημεία ταυτίζονται δημιουργείται κλειστή καμπύλη [Εικόνα 2.11 δεξιά].



Εικόνα 2.11 Ανοικτή , clamped και κλειστή B-spline

2.6 Ρητές εύκαμπτες καμπύλες

Οι ρητές καμπύλες [3] είναι μια ισχυρότερη μορφή καμπύλων καθώς έχουν δυνατότητα να περιγράφουν καμπύλες δευτέρου βαθμού όπως οι κύκλοι και οι ελλείψεις. Ως αποτέλεσμα, προτιμώνται από σχεδιαστικά προγράμματα καθώς μπορούν να αναπαραστήσουν περισσότερα είδη καμπυλών

Μια ρητή (rational) εύκαμπτη καμπύλη είναι ορίζεται ως ο λόγος 2 συναρτήσεων καμπύλης.

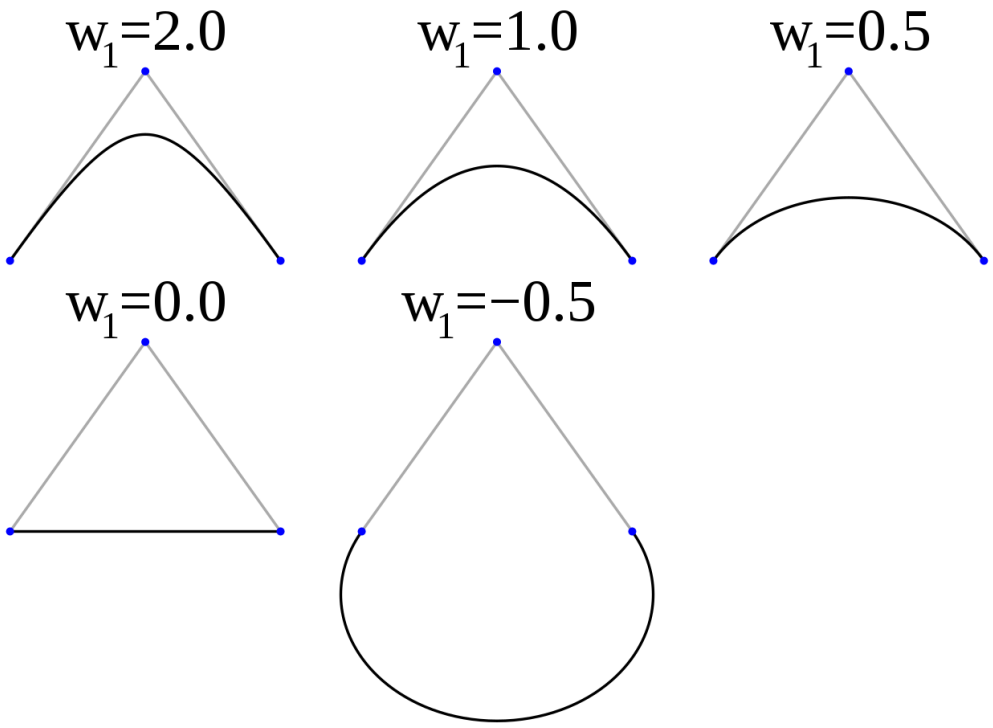
Η ρητή συνάρτηση Bezier ορίζεται ως:

$$RBez(t) = \frac{\sum_{k=0}^{n} w_k * \mathrm{P_k} * \mathrm{B(t)}}{\sum_{k=0}^{n} w_k * \mathrm{B(t)}}$$

Οι συναρτήσεις ρητών καμπυλών χρησιμοποιούν παραμέτρους w_k οι οποίοι είναι παράγοντες αντιστάθμισης των σημείων ελέγχου. Το μέτρο της παραμέτρου w_k επηρεάζει τον βαθμό έλξης της καμπύλης από το συγκεκριμένο σημείο ελέγχου. Αν όλες η παράμετροι w_k είναι ίσες με 1 ή έχουν όλες την ίδια τιμή θα δημιουργηθεί η κανονική καμπύλη Bezier καθώς το άθροισμα των συναρτήσεων μίξης στον παρονομαστή θα είναι ίσο με 1.

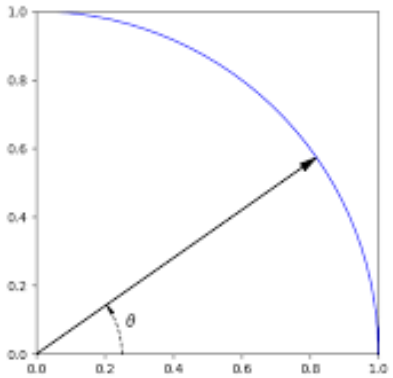
Στο παρακάτω παράδειγμα [Εικόνα 2.12] όλες οι καμπύλες περιγράφονται από τα ίδια σημεία ελέγχου αλλά με διαφορετικούς παράγοντες αντιστάθμισης για το ενδιάμεσο σημείο ελέγχου:

$$w_0 = w_2 = 1 \text{και } w_1 = r$$



Εικόνα 2.12 Κωνική τομή με ρητή καμπύλη Bezier

Επίσης, με τη χρήση ρητής καμπύλης Bezier μπορεί να παραχθεί ένα τεταρτοκύκλιο [Εικόνα 2.13] θέτοντας τα $w_0 = 1$, $w_1 = cos\theta$, $w_2 = 1$ και επιλέγοντας τα σημεία ελέγχου $P_0(0,1)$, $P_1(1,1)$, $P_2(1,0)$ [3]. Για την κατασκευή ολόκληρου κύκλου θα χρειαστούν άλλες 3 καμπύλες σε αντίστοιχη θέση στα υπόλοιπα 3 τεταρτημόρια των καρτεσιανών συντεταγμένων.



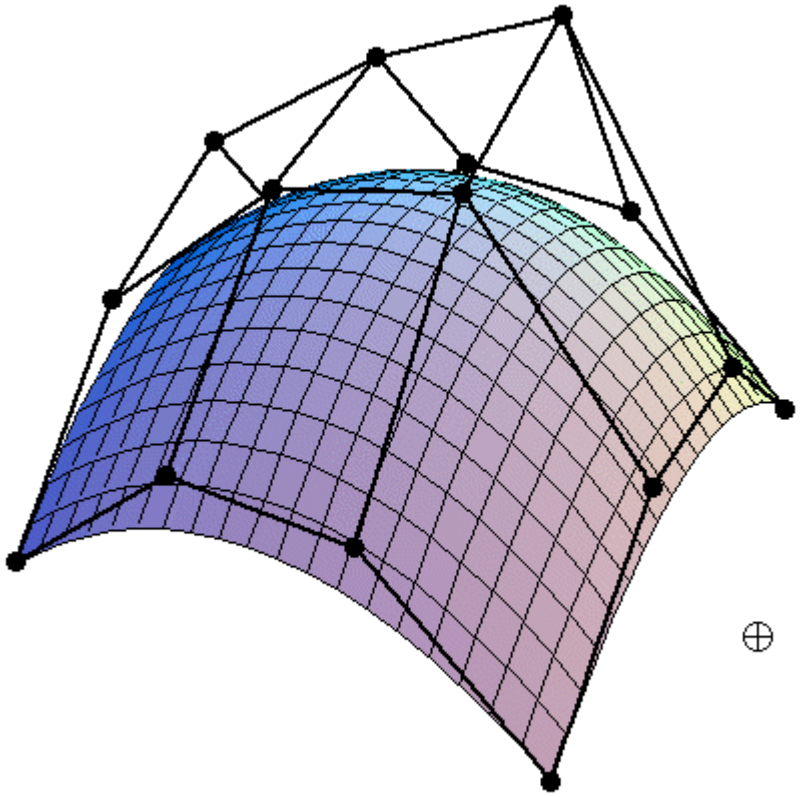
Εικόνα 2.13 Τεταρτημόριο κύκλου με ρητή καμπύλη Bezier

2.7 Επιφάνειες Bezier

Μία επιφάνεια Bezier [3] ορίζεται με τη χρήση δυο συνόλων από καμπύλες Bezier και μπορεί να χρησιμοποιηθεί για να οριστεί μια επιφάνεια ενός αντικειμένου. Η παραμετρική συνάρτηση για την επιφάνεια Bezier εμπλέκει το γινόμενο δυο συναρτήσεων μίξης Bernstein:

$$BezSurf(u,v) = \sum_{j=0}^{m} \sum_{i=0}^{n} P_{i,j} B_i^n(u) B_j^m(v)$$

Οι παράμετροι u,v ορίζουν την επιφάνεια η προς τις δυο κατεύθυνσης στην οποία σχεδιάζεται η επιφάνεια. Ο όρος Pi,j ορίζει την τοποθεσία των (m+1) επί (n+1) σημείων ελέγχου Οι επιφάνειες Bezier έχουν τις ίδιες ιδιότητες με τις καμπύλες Bezier και είναι μια καλή αναπαράσταση επιφανειών. Για να μπορέσουμε να τις φανταστούμε η επιφάνεια δημιουργεί ένα πλέγμα από τα σημεία ελέγχου [Εικόνα 2.1]. Τα σημεία ελέγχου, όπως και στις καμπύλες Bezier ,δεν βρίσκονται πάνω στην επιφάνεια εκτός από τις 4 γωνίες της επιφάνειας.



Εικόνα 2.14 Επιφάνεια Bezier

ΚΕΦΑΛΑΙΟ: 3 Μηχανική μάθηση

3.1 Εισαγωγή

Η μηχανική μάθηση είναι μια υποκατηγορία της τεχνητή νοημοσύνη (artificial intelligence) . Η πιο εντατική της μελέτη της ξεκίνησε μετά τον δεύτερο παγκόσμιο πόλεμο. Ο όρος όπως τον γνωρίζουμε σήμερα πρωτοεμφανίστηκε τα 1956 [12] . Από τότε μέχρι σήμερα έχει αναπτυχθεί και εφαρμόζεται σε πολλούς τομείς όπως η ιατρική, μηχανική, τηλεπικοινωνίες κ.α.. Ο σκοπός της είναι να μπορέσει να δημιουργήσει ένα πρόγραμμα στον υπολογιστή όπου να έχει λογική και να μπορεί να λάβει αποφάσεις αυτόνομα.

Στην μηχανική μάθηση χρησιμοποιείται η οντότητα του πράκτορα. Ένας πράκτορας έχει στόχο να αντιλαμβάνεται το περιβάλλον του και να ενεργεί πάνω σε αυτό ώστε να πετύχει ένα σκοπό.

3.2 Μηχανική Μάθηση

Η μηχανική μάθηση είναι η εφαρμογή των αλγορίθμων και των τεχνικών της τεχνητής νοημοσύνης έτσι ώστε να δημιουργηθεί ένα σύστημα που μπορεί να μάθει από τα δεδομένα που του δίνονται χωρίς να χρειάζεται ο προγραμματισμός του ενός ειδικού προγράμματος για κάθε ξεχωριστή εφαρμογή.

Στόχος της μηχανικής μάθησης είναι η δημιουργία αυτόνομων συστημάτων που είναι σε θέση να θυμούνται τις προηγούμενες προσπάθειες και να χρησιμοποιούνε τις εμπειρίες αυτές για να επιτύχουν τον σκοπό τους. Πιο συγκεκριμένα χρησιμοποιούν μαθηματικές συναρτήσεις και μοντέλα για να εξάγουν τις σχέσεις μεταξύ των εισόδων και των εξόδων τους ώστε να αντιληφθούν το πρόβλημα και στην συνέχεια να το βελτιστοποιήσουν. Μετά από αυτήν την διαδικασία είναι σε θέση να επιλύσουν το συγκεκριμένο πρόβλημα ακόμα και για διαφορετικές αρχικές παραμέτρους και μεταβλητές.

Η λογική πίσω από την μάθηση είναι πως οι πράκτορες δεν πρέπει να βασίζονται μόνο στις αισθήσεις τους για να πάρουν μια απόφαση άλλα να μπορούν να χρησιμοποιούν τις εμπειρίες που έχουν αποκτήσει από προηγούμενες επαναλήψεις. Η εμπειρία που αποκτούν από μια διαδικασία μάθησης τους βοηθάει να λάβουν πιο σωστές και βέλτιστες αποφάσεις την επομένη φορά που θα αντιμετωπίσουν παρόμοιο πρόβλημα. Το κάθε πρόβλημα πρέπει να αναλυθεί και να καθοριστούν οι συνιστώσες που ελέγχει ο πράκτορας κατά την μάθηση.

Υπάρχουν 3 κατηγορίες, αλληλεπίδρασης με τα δεδομένα, για την μάθηση των συνιστωσών [2], [13]: η επιβλεπόμενη μάθηση (supervised learning), η μη επιβλεπόμενη μάθηση (unsupervised learning) και η ενισχυτική μάθηση (reinforced learning).

- Στην **επιβλεπόμενη μάθηση** η εκπαίδευση προκύπτει από την μελέτη παραδειγμάτων εισόδων και εξόδων. Ο πράκτορας μαθαίνει να αντιστοιχίζει ένα μοτίβο εισόδου με ένα μοτίβο στην έξοδο δηλαδή τις δράσεις που πρέπει να κάνει. Αυτή η μέθοδος χρησιμοποιείται σε συστήματα που είναι πλήρως γνωστό το περιβάλλον και τα αποτελέσματα των πράξεων του.

- Αντιθέτως στην **μη επιβλεπόμενη μάθηση** παρέχονται πρότυπα εισόδων χωρίς να δίνονται συγκεκριμένες τιμές για την έξοδο. Παρόλο το σύστημα δεν μπορεί να μάθει μόνο του, μπορεί να δημιουργήσει συμπεράσματα από τις εισόδους του και να παράξει πιθανοτικές εξόδους. Πρέπει να αναγνωρίσει πιθανά μοτίβα μέσα στα δεδομένα εισόδου. Το είδος αυτών των μοτίβων δεν είναι γνωστό στον αλγόριθμο αλλά θα πρέπει να καθοριστεί από αυτόν. Μερικά από τα πιο συνηθισμένα προβλήματα μη επιβλεπόμενης μάθησης είναι το clustering και το anomaly detection. Για το πρόβλημα του clustering ο αλγόριθμος πρέπει να αντιστοιχίσει τα δεδομένα εισόδου του σε clusters με κοινά χαρακτηριστικά ενώ για το πρόβλημα του anomaly detection ο αλγόριθμος πρέπει να ξεχωρίσει τα στοιχεία που δεν ταιριάζουν στο υπόλοιπο σύνολο των δεδομένων.

- Η **ενισχυτική μάθηση** είναι πιο γενική κατηγορία από τις άλλες δυο καθώς η μάθηση γίνεται μέσα από τα αποτελέσματα των ενεργειών του πράκτορα συγκρίνοντας τα αν είναι θετικά ή αρνητικά για το αποτέλεσμα που στοχεύει. Ο αλγόριθμος δεν έχει κάποια σωστή ενέργεια που πρέπει να ακολουθήσει αλλά έχει ένα γενικό σκοπό. Χρησιμοποιεί τα αποτελέσματα των πράξεων του για να πλησιάσει των στόχο του. Αυτό το είδος μάθησης είναι πιο κοντά στην μάθηση του ανθρώπινου εγκεφάλου γιατί χρησιμοποιεί τεχνικές όπως trial and error που είναι συνηθισμένη τεχνική για τον άνθρωπο.

Ένα παράδειγμα επιβλεπόμενης μάθηση είναι η επαγωγική μάθηση [12]. Σε αυτό το είδος μάθησης η είσοδος που δίνεται στον αλγόριθμο του πράκτορα είναι ένα δείγμα σωστών αποτελεσμάτων με τις αντίστοιχες εισόδους μιας συνάρτησης. Το ζητούμενο είναι να βρεθεί αυτή η συνάρτηση έτσι ώστε το πρόβλημα να γενικευτεί για οποιαδήποτε είσοδο. Η δυσκολία στην υλοποίηση αυτής της μεθόδου είναι πως δεν μπορούμε να γνωρίζουμε την μορφή της συνάρτησης, δηλαδή τον τύπο της ή των αριθμό των συνιστωσών που θα χρειαστούν. Έτσι δεν είναι σίγουρο πως ο αλγόριθμος θα παράξει ικανοποιητικά

καλή έξοδο. Σε αυτό το είδος ανήκει και αλγόριθμος που εξετάζεται σε αυτήν την εργασία με μορφή συνάρτησης εκπαίδευσης αυτή της συνάρτησης Bezier.

3.3 Στατιστικές μέθοδοι μάθησης

Αυτές οι μέθοδοι δίνουν μια πιθανοτική προσέγγιση στο τομέα της τεχνητής νοημοσύνης [12]. Δημιουργήθηκαν και μελετήθηκαν με σκοπό να δώσουν λύση σε προβλήματα που επικρατεί αβεβαιότητα όπως τα πραγματικά προβλήματα στο φυσικό κόσμο. Αυτές οι μέθοδοι εξετάζουν δεδομένα εισόδου και παράγουν μια υπόθεση με μορφή πιθανότητας. Στο τέλος μιας επιτυχημένης πρόβλεψης, η αληθής υπόθεση πρέπει να κυριαρχεί σε σχέση με τις υπόλοιπες.

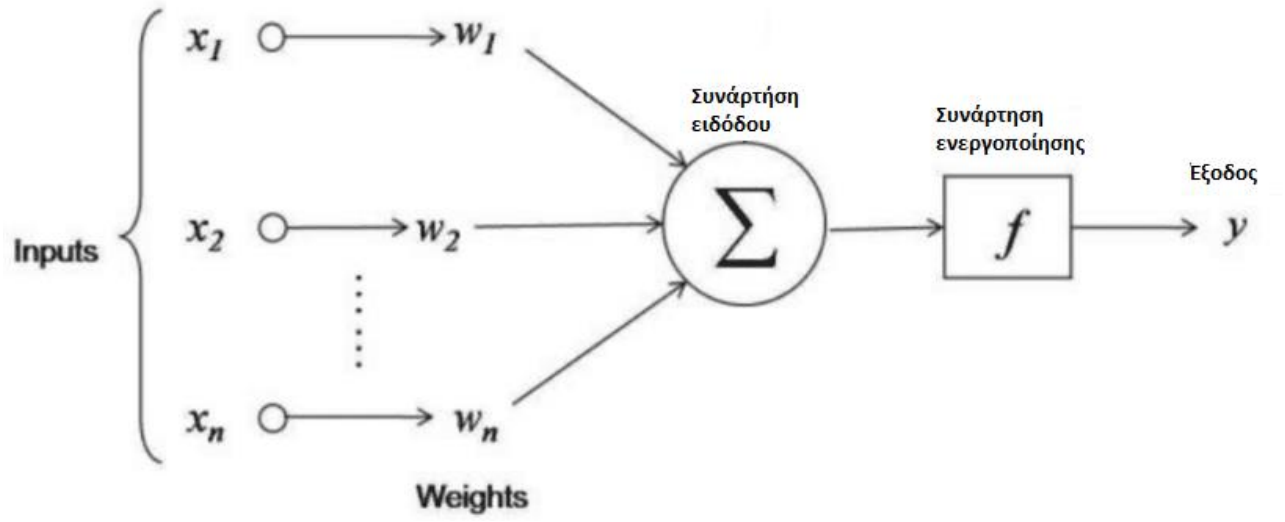
3.4 Τεχνητά νευρωνικά δίκτυα

Τα τεχνητά νευρωνικά δίκτυα (artificial neural networks) πήραν το όνομα τους από τους βιολογικούς νευρώνες του ανθρώπινου εγκεφάλου. Στην βιολογία ο νευρώνας είναι κύτταρο του νευρικού συστήματος που βρίσκεται στον εγκέφαλο και η λειτουργία του είναι η επεξεργασία και διάδοση ηλεκτρικών σημάτων [3]. Η δυνατότητα της σκέψης απορρέει από ένα δίκτυο διασυνδεδεμένων νευρώνων (νευρωνικά δίκτυα, neural network). Πάνω στη συγκεκριμένη λογική στηρίχθηκαν οι πρώτες μελέτες στην τεχνητή νοημοσύνη που προσπαθούσαν να προσομοιώσουν την λειτουργία του εγκεφάλου χρησιμοποιώντας συστήματα τεχνητών διασυνδεδεμένων νευρώνων [2]. Αυτή η προσέγγιση είχε μεγάλη δυνατότητα βελτίωσης λόγω της παράλληλης και κατανεμημένης επεξεργασίας που μπορεί να υποστηρίξει ένα νευρωνικό δίκτυο. Το 1943 επινοήθηκε το πρώτο νευρωνικό σύστημα από τον McCulloch και Pitts [12] το οποίο ήταν πολύ απλό σε σχέση με τις μεταγενέστερες εκδοχές του αλλά έβαλε τα θεμέλια για την μετέπειτα εντατική μελέτη τους. Τα επόμενα νευρωνικά δίκτυα που κατασκευάστηκαν ήταν πιο λεπτομερή και ρεαλιστικά στην προσομοίωση του εγκεφάλου. Σημαντικό χαρακτηριστικό τους είναι η ανοχή σε εισόδους με θόρυβο και η δυνατότητα τους να μαθαίνουν.

3.5 Μονάδες στα νευρωνικά δίκτυα (Νευρώνας)

Ο νευρώνας είναι μια μαθηματική συνάρτηση που μπορεί να έχει μια ή περισσότερες εισόδους και να παράγει μια έξοδο [12]. Τα νευρωνικά δίκτυα αποτελούνται από πολλούς διασυνδεδεμένους κόμβους ή μονάδες (units) που είναι τα θεμέλια όλου του δικτύου. Οι διασυνδέσεις αυτές έχουν κατεύθυνση η οποία συμβολίζει το πως θα γίνει η ενεργοποίηση του κάθε κόμβου. Κάθε κόμβος υπολογίζει το άθροισμα των εισόδων του μαζί με ένα αριθμητικό βάρος $W_{i,j}$ για την κάθε είσοδο για να υπολογίσει την έξοδο του Το βάρος $W_{j,i}$ της κάθε εισόδου δείχνει το πόσο η συγκεκριμένη είσοδος θα επηρεάσει το αποτέλεσμα του υπολογισμού και είναι οι μεταβλητές που εκπαιδεύονται σε ένα νευρωνικό δίκτυο. Έπειτα αυτό το άθροισμα εφαρμόζεται στην συνάρτηση ενεργοποίησης (activation function) και παράγεται η έξοδος του κόμβου. Στην εικόνα [Εικόνα 3.1] φαίνεται η δομή ενός νευρώνα [12].

$$a = g(in) = g\left(\sum W_{j,i} * a_i\right)$$



Εικόνα 3.1 Δομή μιας μονάδας των νευρωνικών δικτύων

3.5.1 Συνάρτηση ενεργοποίησης (activation function)

Η συνάρτηση ενεργοποίησης [12] παρέχει δυο καταστάσεις για έναν συγκεκριμένο κόμβο, μετατρέπει την έξοδο της μονάδας σε μια κλίμακα και αν είναι πάνω ή κάτω από μια τιμή κατωφλίου μπορεί

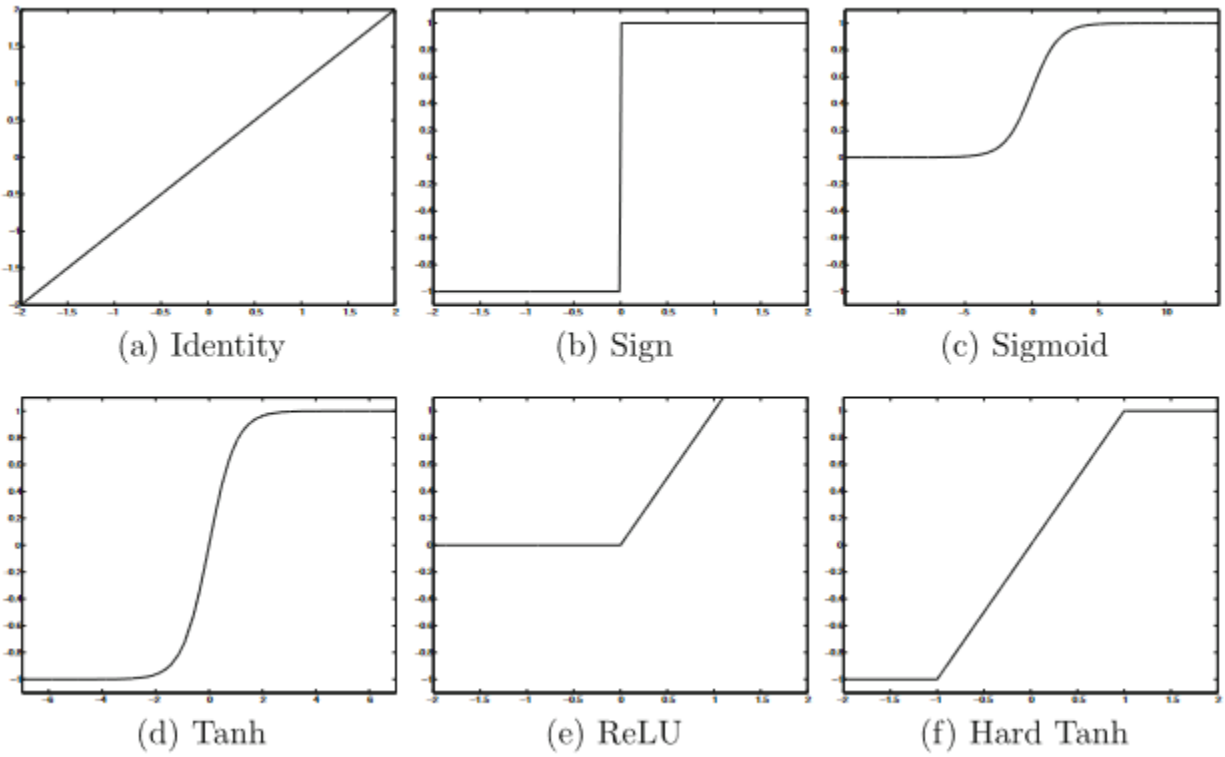
να χαρακτηρίσει την μονάδα ενεργή ή ανενεργή. Δηλαδή όταν οι είσοδοι του κόμβου μας κατευθύνουν προς την σωστή λύση τότε θέλουμε η συνάρτηση ενεργοποίησης να μας δίνει τιμές κοντά στην τιμή 1 δηλαδή ο κόμβος να είναι ενεργός. Αντίθετα όταν οι είσοδοι μας κατευθύνουν προς την λανθασμένη λύση τότε να μας δίνει τιμές κοντά στο 0, τότε λέμε πως ο κόμβος είναι ανενεργός.

Όσο αφορά την έξοδο του δικτύου η επιλογή της συνάρτησης ενεργοποίησης παίζει σημαντικό ρόλο [3]. Για παράδειγμα μπορεί χρειάζεται να προβλέψει διακριτές καταστάσεις που σημαίνει ότι μια δυαδική συνάρτηση ενεργοποίησης θα είναι αρκετή, όπως η συνάρτηση του δυαδικού βήματος [Εικόνα 3.2 (b)]. Όμως είναι δυνατόν να υπάρχουν και άλλες περιπτώσεις όπου οι προβλέψεις του δικτύου να είναι συνεχείς. Για παράδειγμα αν η προβλεπόμενη τιμή είναι πραγματικός αριθμός δηλαδή είναι το αποτέλεσμα μιας συνάρτηση τότε θα πρέπει να χρησιμοποιηθεί μια συνάρτηση ενεργοποίησης που οι έξοδοι της να περνούν τιμές σε όλο το διάστημα των πραγματικών αριθμών όπως η συνάρτηση identity [Εικόνα 3.2 (a)]. Αν η έξοδος του δικτύου μας θέλουμε να είναι πιθανότητα μιας κλάσης τότε η σιγμοειδής συνάρτηση είναι ιδανική γιατί μετατρέπει την προβλεπόμενη τιμή σε μια πιθανότητα για την συγκεκριμένη κλάση.

Για δίκτυα πολλών επιπέδων η συνάρτηση ενεργοποίησης θα πρέπει να είναι μη γραμμική διότι διαφορετικά όλο το νευρωνικό δίκτυο θα εκφυλίζεται γραμμικά. Κάποιες από τις συνήθης επιλογές για την συνάρτηση ενεργοποίησης είναι η σιγμοειδής συνάρτηση (sigmoid function) [14] ή το ReLU (Rectified Linear Unit) αλλά υπάρχουν και άλλες συναρτήσεις που εξυπηρετούν τον ίδιο ή και διαφορετικό σκοπό [Εικόνα 3.2]. Επίσης για την πρόβλεψη μιας επιλογής μέσα από ένα σύνολο από πιθανές επιλογές χρησιμοποιείται η συνάρτηση ενεργοποίησης SoftMax [15].

Η τιμή που υπολογίζεται πριν την εφαρμογή της συνάρτησης ενεργοποίησης ονομάζεται pre-activation value, ενώ η τιμή που υπολογίζεται μετρά την συνάρτηση ενεργοποίησης ονομάζεται post-activation value.

$$Σιγμοειδής\ συνάρτηση\ \sigma(z) = \frac{1}{(1 + e^{-z})}$$



Εικόνα 3.2 Διάφορα διαγράμματα συναρτήσεων ενεργοποίησης

3.5.2 Συνάρτηση λάθους (loss function)

Η συνάρτηση λάθους είναι απαραίτητη για την εκπαίδευση του δικτύου. Το αποτέλεσμά της είναι ένας δείκτης για το ποσό μακριά είναι η προβλεπόμενη τιμή του δικτύου από τον στόχο που έχει οριστεί [12]. Την χρησιμοποιούμε για να υπολογίσουμε τα gradients που θα μας οδηγήσουν προς την επιθυμητή έξοδο. Για την επιλογή της συνάρτηση λάθους πρέπει να ληφθεί υπόψη η εφαρμογή στην οποία εισάγεται. Για παράδειγμα, σε ένα πρόβλημα παλινδρόμησης, η τετραγωνική διαφορά της προβλεπόμενης τιμής από την πραγματική είναι δυνατόν να ορίσει το λάθος του δικτύου.

3.5.3 Εκπαίδευση των νευρωνικών δικτύων

Τα νευρωνικά δίκτυα κατατάσσονται όσον αφορά την μάθηση στην κατηγορία της επιβλεπόμενης μάθησης [12]. Έτσι ένα σύνολο από δεδομένα, αντιστοιχισμένα ήδη με την επιθυμητή έξοδο, χρησιμοποιείται με σκοπό την εκπαίδευση του δικτύου.. Για την σωστή αξιολόγηση του μοντέλου είναι απαραίτητο να υπάρχει και ένα άλλο τέτοιο σύνολο από δεδομένα για να υπάρχει δυνατότητα αξιολόγησης σε δεδομένα διαφορετικά από τα δεδομένα εκπαίδευσης ώστε να γίνει αντιληπτή η υπερπροσαρμογή.

Κατά την διάρκεια εκπαίδευσης του νευρωνικού δικτύου, τα βάρη των εισόδων της κάθε μονάδας αλλάζουν τιμή με σκοπό την ελαχιστοποίηση της συνάρτησης λάθους, μετρώντας την διαφορά της εξόδου του δικτύου από την επιθυμητή έξοδο. Αυτή η αλλαγή στα βάρη του νευρωνικού δικτύου γίνεται μέσω μια τεχνικής που λέγεται back propagation καθώς το λάθος ελαχιστοποιείται.

Σε ένα δίκτυο πολλών κόμβων το πλήθος των βαρών που πρέπει να μεταβληθούν είναι μεγάλο, αυτό κάνει το πρόβλημα της εκπαίδευσης αρκετά περίπλοκο. Για τα νευρωνικά δίκτυα η συνάρτηση λάθους είναι μια σύνθεση όλων των βαρών. Ο αλγόριθμος back-propagation [14] αξιοποιεί τους κανόνες του διαφορικού λογισμού για να υπολογίσει τα gradient error, σε μορφή αθροίσματος, διάφορων μονοπατιών από ένα κόμβο προς την έξοδο. . Τελικός σκοπός είναι να υπολογίσει τις τελικές τιμές των βαρών της κάθε μονάδας έτσι ώστε να ελαχιστοποιηθεί η συνάρτηση λάθους και το πρόβλημα να είναι πιο κοντά στην λύση του.

Ο αλγόριθμος αποτελείται από δυο κύριες φάσεις που ονομάζονται forward phase και backward phase. Η forward phase χρειάζεται να υπολογίσει τις εξόδους και τις τοπικές παραγώγους διαφόρων κόμβων και η backward phase χρειάζεται να συνδυάσει τα γινόμενα αυτών των τοπικών μεταβλητών για να δημιουργήσει ένα μονοπάτι από τους κόμβους έως την έξοδο.

- **Forward phase**: Σε αυτήν την φάση, εισάγονται στην είσοδο του νευρωνικού τα δεδομένα εισόδου. Έπειτα ακολουθούν οι υπολογισμοί τις κάθε μονάδας διαδοχικά δηλαδή η εκτέλεση του δικτύου με τις τρέχουσες τιμές των βαρών. Τέλος το αποτέλεσμα συγκρίνεται με το πραγματικό αποτέλεσμα που έχουμε για εκπαίδευση και υπολογίζεται η συνάρτηση λάθους καθώς και η παράγωγος της με βάση την έξοδο. Η παράγωγος με βάση τα βάρη του δικτύου υπολογίζεται στην επόμενη φάση
- **Backward phase**: Ο στόχος αυτής της φάσης είναι να υπολογιστούν τα σωστά gradient της συνάρτησης λάθους με βάση τα βάρη. Αυτό επιτυγχάνεται χρησιμοποιώντας την παράγωγο της συνάρτησης ως προς το κάθε βάρος. Έπειτα τα gradients χρησιμοποιούνται για να ενημερώσουν τα βάρη σε νέες τιμές που ελαχιστοποιούν την συνάρτηση λάθους. Ο λόγος που ονομάζεται backward phase είναι γιατί αυτή η διαδικασία ξεκινάει από του τελευταίους κόμβους και κατευθύνεται στους αρχικούς.

3.5.4 Optimizers

Ο αλγόριθμος back-propagation είναι απαραίτητος για εκμάθηση και γενίκευση του προβλήματος από το νευρωνικό δίκτυο [14]. Για αυτό θα πρέπει να είναι βελτιστοποιημένος και αποδοτικός. Δεν υπάρχει κάποιος αλγόριθμος βελτιστοποίησης που να εφαρμόζεται και να επιλύει όλα τα προβλήματα. Για κάθε ένα ξεχωριστό πρόβλημα και δεδομένα πρέπει να εφαρμοστεί διαφορετικός αλγόριθμος βελτιστοποίησης. Για κάθε ένα ξεχωριστό πρόβλημα και δεδομένα πρέπει να εφαρμοστεί διαφορετικός αλγόριθμος βελτιστοποίησης. Ο ρόλος του αλγορίθμου είναι να βελτιστοποιήσει την διαδικασία μάθησης εισάγοντας μαθηματικές τεχνικές για την ενημέρωση των μεταβλητών του νευρωνικού και να ελαχιστοποιήσει τη συνάρτηση λάθους.

3.5.5 Αρχιτεκτονική Τεχνητών Νευρωνικών Δικτύων

Η δομή του δικτύου είναι διατεταγμένη σε επίπεδα (layers). Σε κάθε επίπεδο υπάρχει ένας αριθμός από μονάδες του νευρωνικού δικτύου [12], [13]. Η κάθε μονάδα ενός επιπέδου διαδίδει την έξοδο του σε κάθε μονάδα του επόμενου επιπέδου και έτσι δημιουργούνται οι διασυνδέσεις στα νευρωνικά δίκτυα. Τα επίπεδα που βρίσκονται ανάμεσα στο επίπεδο εισόδου και στο επίπεδο εξόδου ονομάζονται κρυφά επίπεδα (hidden layers). Ο αριθμός αυτών των κρυφών επίπεδων και ο αριθμός των μονάδων που έχει το κάθε επίπεδο αυξάνει την πολυπλοκότητα όλου δικτύου και το πλήθος των βαρών που πρέπει να υπολογιστούν. Έτσι το νευρωνικό δίκτυο έχει μεγαλύτερο χώρο προς εκπαίδευση με αποτέλεσμα να μπορεί να προσαρμοστεί σε πιο δύσκολα προβλήματα. Ένα μεγάλο δίκτυο όμως έχει τον κίνδυνο της υπεπροσαρμογής (overfitting), όπου το πρόβλημα δεν γενικεύεται επαρκώς με αποτέλεσμα η λύση του να στοχεύει μόνο στα δεδομένα της εκπαίδευσης.

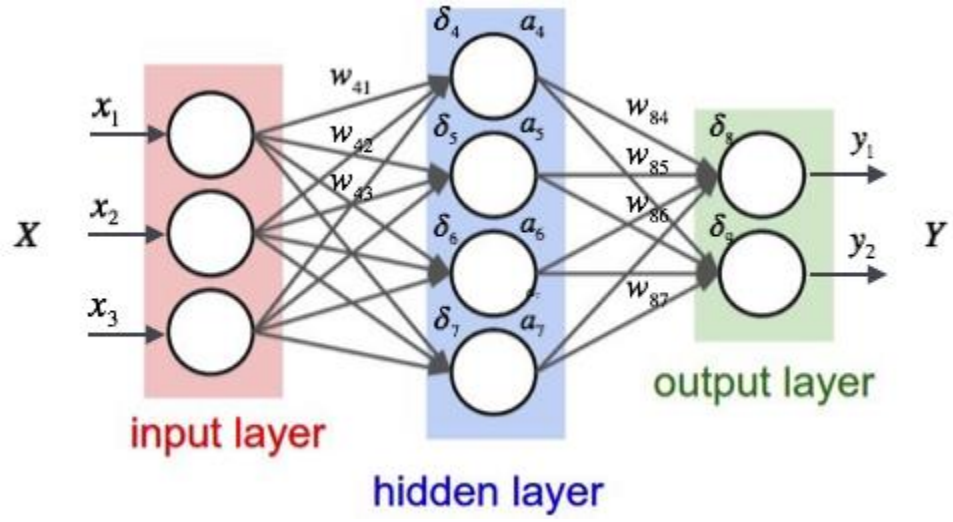
Υπάρχουν δυο κατηγορίες νευρωνικών δικτύων όσον αφορά την κατεύθυνση μετάδοσης των μονάδων [12]. Η πρώτη είναι η προς τα εμπρός τροφοδότηση του σήματος (feed forward networks) και η δεύτερη είναι τα αναδρομικά δίκτυα (recurrent networks). Επιπλέον, υπάρχουν τα συνελικτικά νευρωνικά δίκτυα που ειδικεύονται σε εφαρμογές αναγνώρισης εικόνας.

3.5.5.1 Προς τα εμπρός τροφοδότηση του σήματος (feed-forward networks)

Σε αυτό το είδος δικτύου [12] όπως συμπεραίνεται και από το όνομα του, η κάθε μονάδα μεταδίδει την έξοδο της μόνο προς τους κόμβους του επόμενου επιπέδου. Αυτό το είδος ονομάζεται και δίκτυο αισθητήρων (perceptrons). Όταν πρόκειται για δίκτυο με ένα επίπεδο μονάδων τότε ονομάζεται αισθητήρας (perceptron). Οι μονάδες είναι διατεταγμένες σε επίπεδα (layers) τα οποία έχουν μια σειρά

μετάδοσης, ανάλογα με το επίπεδο που βρίσκονται [Εικόνα 3.3]. Δηλαδή το πρώτο επίπεδο μεταδίδει την έξοδο του στο δεύτερο ,το δεύτερο στο τρίτο κ.ο.κ. Όταν όλες οι μονάδες ενός επιπέδου είναι διασυνδεδεμένες με το επόμενο επίπεδο τότε λέμε πως το δίκτυο είναι πλήρες διασυνδεδεμένο. Το επίπεδο εισόδου του δικτύου δεν εκτελεί κανέναν υπολογισμό του πάρα μόνο στέλνει την τιμή της κάθε μονάδας του στο επόμενο επίπεδο. Για αυτό το λόγο δεν μετριέται ως επίπεδο. Από την άλλη η έξοδος του δικτύου είναι το αποτέλεσμα του προηγουμένου επιπέδου. Το κάθε επίπεδο μπορεί να έχει το δικό του πλήθος από μονάδες έτσι ένα δίκτυο που φαίνεται εξωτερικά να έχει λίγες εξόδους μπορεί εσωτερικά να αποτελείται από μεγάλο αριθμό μονάδων.

Δυο από τα πιο συνηθισμένα προβλήματα που μπορούν να επιλυθούν από τα δίκτυα προς-τα-εμπρός-τροφοδότηση είναι η ταξινόμηση (classification) και η παλινδρόμηση (regression). Ταξινόμηση εννοούμε την αντιστοίχιση της εισόδου με μια κατηγορία στην έξοδο που ορίζουμε εμείς. Δηλαδή αν έχουμε να κατατάξουμε τις εισόδους σε x κατηγορίες τότε πρέπει να έχουμε x εξόδους στο νευρωνικό δίκτυο έτσι ώστε η κάθε κατηγορία να αντιστοιχίζεται σε μια έξοδο. Το πρόβλημα της παλινδρόμησης είναι η πρόβλεψη των τιμών μια συνάρτησης έχοντας κάποιο δείγμα από αυτήν.



Εικόνα 3.3 Ιεραρχία διασυνδέσεων σε ένα feed-forward δίκτυο[16]

Τα προς-τα-εμπρός-τροφοδότηση δίκτυα μπορούν να χωριστούν σε δύο κατηγορίες. Αν ένα δίκτυο αποτελείται από ένα επίπεδο μονάδων και οι είσοδοι είναι απευθείας συνδεδεμένοι με τις εξόδους τότε το δίκτυο ονομάζεται δίκτυο ενός επιπέδου (single layer neural network) ή δίκτυο αισθητήρα μονού

επιπέδου (single layer perceptron). Αν το δίκτυο αποτελείται από πολλά επίπεδα και είσοδοι του τροφοδοτούνται μέσω πολλών ενδιάμεσων κόμβων τότε το δίκτυο ονομάζεται αισθητήρας πολλών επιπέδων (multilayer perceptrons).

3.5.5.2 Αναδρομικά δίκτυα (recurrent networks)

Στα αναδρομικά δίκτυα [12] η κάθε μονάδα τροφοδοτεί την έξοδο της πίσω στις εισόδους της. Έτσι υπάρχει αμφίδρομη επικοινωνία μεταξύ μονάδων. Αυτό έχει ως αποτέλεσμα το σύστημα να διατηρεί μια βραχυπρόθεσμη μνήμη. Επίσης δημιουργεί μια πιο ρεαλιστική εκδοχή του εγκεφάλου και έχει μεγαλύτερη προοπτική για μάθηση.

3.5.5.3 Συνελικτικά νευρωνικά δίκτυα

Τα συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks) [12], [16] συνήθως χρησιμοποιούνται για αναγνώριση εικόνας. Σε ένα απλό νευρωνικό δίκτυο οι μονάδες του ίδιου επιπέδου δεν έχουν μεταξύ τους διασυνδέσεις. Έτσι δεν λειτουργούν με αποδοτικό και αποτελεσματικό τρόπο σε εφαρμογές αναγνώρισης εικόνας, καθώς κάθε pixel της εικόνας που αντιστοιχίζεται σε μια είσοδο του δικτύου δεν μπορεί να συσχετιστεί με τα γειτονικά του pixels.

Τα συνελικτικά νευρωνικά δίκτυα καταφέρνουν να ξεπεράσουν αυτήν την αδυναμία διατηρώντας της γειτονικές σχέσεις μεταξύ pixels. Για να το επιτύχουν αυτό τροφοδοτούν τμήματα της εισόδου τους σε συγκεκριμένους μόνο κόμβους του επόμενου επιπέδου για να διατηρηθεί η γειτονική συσχέτιση. Έτσι γίνονται πιο αποδοτικά και αποτελεσματικά σε τέτοιου είδους εφαρμογές.

ΚΕΦΑΛΑΙΟ: 4 Μέθοδοι προσαρμογής καμπυλών

4.1 Εισαγωγή

Η προσαρμογή καμπύλης (curve fitting) στον τομέα των γραφικών είναι η διαδικασία κατασκευής μιας καμπύλης ή μια μαθηματικής συνάρτησης που ταιριάζει καλύτερα σε ένα σύνολο σημείων είτε στον χώρο είτε στο επίπεδο [14]. Με πιο μαθηματικούς όρους η προσαρμογή καμπύλης είναι μια διαδικασία ανάλυσης δεδομένων που προσπαθεί να δημιουργήσει μια γραμμική ή μη γραμμική συνάρτηση που μοντελοποιεί αυτά τα δεδομένα. Υπάρχουν τρία σενάρια για αυτήν την συνάρτηση:

- Πρώτο: Η δομή της συνάρτηση να είναι θεωρητικά ή εμπειρικά γνωστή αλλά όχι οι παράμετροι που την χαρακτηρίζουν (π.χ. γνωρίζουμε πως η συνάρτηση είναι μια ευθεία γραμμή αλλά όχι την κλίση της ή/και την μετατόπιση της). Σε αυτήν την περίπτωση το πρόβλημα μπορεί να λυθεί αλγεβρικά μέσω μαθηματικών τεχνικών.
- Δεύτερο: Η δομή της συνάρτηση να μη είναι γνωστή αλλά να μπορεί γίνει μια υπόθεση για αυτήν. Σε τέτοιες περιπτώσεις εξετάζονται οι δομές που πιθανώς να καλύπτουν τα δεδομένα και έπειτα προκύπτει το συμπέρασμα για τη δομή που ταιριάζει καλύτερα σε αυτά.
- Τρίτο: Η δομή της συνάρτηση να είναι άγνωστη και δεν μπορεί να γίνει καμία υπόθεση για αυτήν. Σε αυτήν την περίπτωση οι μέθοδοι του deep learning και συγκεκριμένα της μηχανικής μάθησης είναι ιδανικές.

Για τις δύο πρώτες κατηγορίες η προσαρμογή καμπύλης μπορεί να επιτευχθεί με τη μέθοδο της παλινδρόμησης. Σε προβλήματα παλινδρόμησης στόχος είναι η πρόβλεψη της πραγματικής τιμής για κάθε τιμή εισόδου, δηλαδή ο ορισμός συνάρτησης μεταξύ εισόδου και εξόδου, μέσα από ένα σύνολο δειγμάτων. Υπάρχουν πολλές μέθοδοι για προσαρμογή καμπύλων που κατατάσσονται σε αυτήν την τεχνική, όπως η μέθοδος των ελαχίστων τετραγώνων (least squares) [17] που εντάσσεται στην γραμμική παλινδρόμηση. Αναλόγως το είδος των συναρτήσεων που θέλουμε να εφαρμόσουμε, το πρόβλημα μπορεί να είναι είτε γραμμικό είτε μη γραμμικό και σε κάθε περίπτωση πρέπει να επιλυθεί με διαφορετικό τρόπο.

Επίσης υπάρχουν και μέθοδοι τεχνητής νοημοσύνης όπως δέντρα αποφάσεων (decision tree) [18] και νευρωνικά δίκτυα (neural networks) [19] που μπορούν να λύσουν το πρόβλημα ακόμα και αν η δομή

της συνάρτησης είναι τελείως άγνωστη. Επίσης, ένα είδος μηχανικής μάθησης είναι οι γενετικοί αλγόριθμοι (genetic algorithms) [20] οι οποίοι έχουν τη δυνατότητα να αναζητήσουν μια καμπύλη μέσα σε ένα νέφος σημείων.

4.2 Αλγόριθμος ελαχίστων τετραγώνων

Ο αλγόριθμος των ελαχίστων τετραγώνων ανήκει στην γενική κατηγορία της παλινδρόμησης (regression analysis) και χρησιμοποιείται κυρίως στον τομέα των πιθανοτήτων και της στατιστικής. Η ανάλυση της παλινδρόμησης μας επιτρέπει να προβλέψουμε τα αποτελέσματα μιας διαδικασίας έχοντας πολλά δεδομένα-δείγματα από αυτή [21]. Τα δεδομένα-δείγματα περιέχουν ένα σύνολο από ζεύγη μεταβλητών X,Y όπου μόνο η μια μεταβλητή θεωρείται εξαρτημένη. Η ανάλυση παλινδρόμησης περιλαμβάνει μαθηματικές συναρτήσεις για να μοντελοποιήσει και να διερευνήσει την εξάρτηση μιας μεταβλητής Y, η οποία θεωρείται εξαρτημένη από μια άλλη μεταβλητή X, που θεωρείται ανεξάρτητη.

Στην λογική της παλινδρόμησης η μεταβλητή X αντιμετωπίζεται ως σταθερά. Η εξίσωση της παλινδρόμησης αποτελείται από δυο μέρη: το συστηματικό στο οποίο είναι την μαθηματική σχέση που περιγράφει καλύτερα το δείγμα-δεδομένα , και το στοχαστικό μέρος u. Το u αναπαριστά το τυχαίο σφάλμα που υπάρχει μέσα στα δεδομένα εισόδου και στόχος της μεθόδου είναι να το ελαχιστοποιήσει.

Ένα παράδειγμα τέτοιας εξίσωσης με χρήση του u είναι:

$$f(x) = bx + u$$

Για την επίτευξη καλής προσέγγιση πρέπει να ελαχιστοποιηθεί το κατάλοιπο της συνάρτησης το οποίο υπολογίζεται από την σχέση:

$$y_i - f(x_i)$$

Η τελική σχέση που πρέπει να ελαχιστοποιηθεί για μια ικανοποιητική προσέγγιση είναι το άθροισμα όλων των αποστάσεων μεταξύ των δεδομένων και της τιμής της συνάρτησης.

$$error = \sum_{i=a}^{n} \big(y_i - f(x_i)\big)^2 = \sum_{i=a}^{n} (y_i - (bx_i + u))^2$$

Αυτή είναι η σχέση του αλγορίθμου των ελαχίστων τετραγώνων και όσο πιο κοντά στο μηδέν είναι το αποτέλεσμα αυτής τόσο καλύτερη είναι η προσέγγιση.

Πρέπει να τηρούνται κάποιες προϋποθέσεις ώστε ο αλγόριθμος να παράγει σωστά αποτελέσματα. Αυτές οι προϋποθέσεις είναι τρεις:

α) Η μεταβλητή X να προσδιορίζεται από την αρχή ,για κάθε τιμή της X να υπάρχει μια τιμή Y που να ακολουθεί μια κατανομή πιθανότητας γ.

β) Το σφάλμα u να είναι ανεξάρτητο από την X και να ακολουθεί τυχαία κανονική κατανομή με μέση τιμή μηδέν και σταθερή διακύμανση *σ*.

γ) Η διακύμανση της κατανομής *σ* να είναι σταθερή για όλα τα ζεύγη μεταβλητών, αυτή η ιδιότητα είναι γνωστή και ως **ομοσκεδαστικότητα** [21].

Έπειτα για την ελαχιστοποίηση του error χρησιμοποιούνται τεχνικές του διαφορικού λογισμού, συγκεκριμένα η ιδιότητά πως μια τετραγωνική καμπύλη έχει την ελάχιστη τιμή της στο σημείο που παράγωγος της έχει την τιμή μηδέν. Έτσι παίρνοντας την μερική παράγωγο της ως προς b και u και εξισώνοντας με μηδέν μπορούν να προκύψουν οι συντελεστές της ευθείας b και u που ελαχιστοποιούν το error.

$$\frac{\partial error}{\partial b} = 0$$

$$\frac{\partial error}{\partial u} = 0$$

Όταν οι εξαρτημένες μεταβλητές για τον αλγόριθμο είναι γραμμικές το πρόβλημα ονομάζεται Linear Least Squares και η λύση του προκύπτει από την εξίσωση της παραγωγού του αθροίσματος στο μηδέν. Όταν οι εξαρτημένες μεταβλητές του προβλήματος δεν είναι γραμμικές τότε το πρόβλημα ονομάζεται nonlinear least squares και η λύση του προβλήματος δεν είναι τόσο απλή. Σε αυτή την περίπτωση τα προβλήματα αυτά χρειάζονται διαφορετική προσέγγιση για να λυθούν, κάποιες από τις μεθόδους που χρησιμοποιούνται για επίλυση τέτοιων προβλημάτων είναι η μέθοδος Gradient descent και η μέθοδος Gauss-Newton.

Υπάρχουν επίσης άλλες παραλλαγές του αλγορίθμου ελαχίστων. Μια από αυτές είναι τα αλγόριθμος των ελαχίστων τετραγώνων με βάρη (Weighted least squares (WLS)) [22] τα οποία χρησιμοποιούνται όταν υπάρχει ετεροσκεδαστικότητα μεταξύ των λαθών. Ενώ υπάρχει και η μέθοδος των γενικευμένων ελαχίστων τετραγώνων (Generalized Least Squares) [22] τα οποίο αποτελούν γενίκευση του απλού αλγορίθμου των ελαχίστων τετραγώνων

4.3 Μη γραμμικοί αλγόριθμοι παλινδρόμησης

Οι μη γραμμικοί αλγόριθμοι παλινδρόμησης υπάγονται στη κατηγορία της παλινδρόμησης αλλά χρησιμοποιούνται όταν οι παράμετροι της δεν αποτελούν μια γραμμική συνάρτηση της εξόδου της συνάρτησης μοντέλου [14]. Αυτοί οι αλγόριθμοι ελαχιστοποιούν την συνάρτηση που εκφράζει το λάθος σε επαναληπτικό ρυθμό και με κάθε επανάληψη καταφέρνουν να προσεγγίζουν καλύτερα το αρχικό νέφος σημείων.

Έστω για ένα σημείο X_k και $P(t_k)$ είναι το πλησιέστερο σημείο της καμπύλης που θέλουμε να προσαρμόσουμε. Η απόσταση μεταξύ του σημείου X_k και $P(t_k)$ είναι $|X_k - P(t_k)|$.Τότε η συνάρτηση που θέλουμε να ελαχιστοποιήσουμε είναι:

$$\min = \sum_{k=1}^{n} |X_k - P(t_k)|$$

Μερικά παραδείγματα αλγορίθμων επίλυσης προβλημάτων μη γραμμικής παλινδρόμησης είναι οι εξής:

- Ο αλγόριθμος **Gauss-Newton** [23] αποτελεί μια παραλλαγή της μεθόδου του Newton για την εύρεση ριζών διαφορικών εξισώσεων και χρησιμοποιείται για την εύρεση της ελάχιστης τιμής μια συνάρτησης. Ο αλγόριθμος βασίζεται σε επαναλαμβανόμενους υπολογισμούς μέσα από τυχαίες εικασίες για βρει τη σωστή λύση.
- Ο αλγόριθμος **Gradient Descent** [23] ή αλλιώς ονομάζεται και steepest descent διότι αυτό που προσπαθεί να κάνει ο αλγόριθμος είναι να εντοπίσει στην τοπική περιοχή την κατεύθυνση με την πιο απότομη κάθοδο ώστε να κατευθυνθεί προς αυτή και να βρει το ελάχιστο μιας συνάρτησης. Προϋπόθεση για την εκτέλεση του αλγορίθμου είναι η συνάρτηση να είναι διαφορίσημη σε όλο το διάστημα ορισμού της. Αυτός ο αλγόριθμος παρότι μπορεί να βρει ένα τοπικό ελάχιστο αρκετά γρήγορα μπορεί να κολλήσει σε αυτό και μην βρει το ολικό ελάχιστο. Αυτός ο αλγόριθμος χρησιμοποιείται επίσης από τα νευρωνικά δίκτυα για των υπολογισμό των νέων βαρών.
- Ο αλγόριθμος **Levenberg–Marquardt** είναι ο πιο γνωστός και πιο πολύ χρησιμοποιημένος αλγόριθμος βελτιστοποίησης [23]. Έχει καλύτερη απόδοση σε σχέση με τον απλό αλγόριθμο gradient descent και από τις άλλες εκδοχές του. Ο αλγόριθμος Levenberg–Marquardt αποτελεί μια μίξη των αλγορίθμων Gradient Descent και Gauss-Newton.

4.4 Προσαρμογή καμπύλης με μηχανική μάθηση

Όλες οι μέθοδοι της μηχανικής μάθησης εφαρμόζονται σε ζεύγη από διανύσματα εισόδου και εξόδου και χωρίς να γνωρίζουμε την συνάρτηση που τα συνδέει μεταξύ τους [2], [24]. Οι αλγόριθμοι της μηχανικής μάθησης προσπαθούν μέσω αυτών των διανυσμάτων να προσεγγίσουν την αρχική συνάρτηση που τα δημιούργησε. Εξετάζοντας την διαφορά μεταξύ εισόδου και εξόδου μαθαίνουν τις παραμέτρους του συστήματος τους και μπορούν να την αναπαράγουν προσεγγιστικά χωρίς να έχουν καμία γνώση για την αρχική συνάρτηση.

Για την προσαρμογή καμπύλης τέτοιοι αλγόριθμοι έχουν την δυνατότητα να παράγουν μια πολύ ικανοποιητική προσέγγιση. Συνήθως η εκμάθηση του αλγορίθμου γίνεται με την χρήση ενός καταλοίπου (η απόσταση του πραγματικού σημείου με το σημείο που υπολογίζει ο αλγόριθμος, συνήθως γίνεται η χρήση του Root Mean Square Error) το οποίο προσπαθεί να ελαχιστοποιήσει. Οι αλγόριθμοι αυτοί έχουν ένα βασικό μειονέκτημα καθώς δεν μπορούν να επεκταθούν σε γενικότερη περίπτωση. Έτσι αν τροποποιήσουμε ή αλλάξουμε το διάνυσμα εισόδου το αποτέλεσμα δεν θα είναι καθόλου αντιπροσωπευτικό για την αρχική συνάρτηση. [2].

ΚΕΦΑΛΑΙΟ: 5 Υλοποίηση αλγορίθμων προσαρμογής καμπύλης

Σε αυτό το κεφάλαιο αναλύονται οι δυο τρόποι προσαρμογής καμπύλης Bezier που εξετάστηκαν στα πλαίσια αυτής της εργασίας. Πρώτα θα εξεταστεί ο αλγόριθμος γραμμικής παλινδρόμησης με την μέθοδο των ελαχίστων τετραγώνων (παράγραφος 5.4) και ακολούθως δύο εναλλακτικοί τρόποι με χρήση νευρωνικών δικτύων (παράγραφος 5.5). Τέλος θα παρουσιαστεί μια σύγκριση μεταξύ των δυο τεχνικών (παράγραφος 5.6), όσον αφορά την ακρίβεια τους και τον χρόνο εκτέλεσης τους ενώ εξάγονται τα ανάλογα συμπεράσματα.

5.1 Δεδομένα υλοποίησης

Τα νέφη σημείων που χρησιμοποιήθηκαν έχουν σημεία που ορίζονται σε δύο διαστάσεις, αλλά μπορεί να γίνει επέκταση των αλγορίθμων σε περισσότερες διαστάσεις χωρίς να αλλάξει η λογική της επίλυσης. Η προσαρμογή καμπύλης γίνεται για αριθμό σημείων του νέφους κατά πολύ μεγαλύτερο από το βαθμό της καμπύλης διαφορετικά η καμπύλη θα ταιριάζει απόλυτα για κάθε τέτοιο νέφος. Επίσης α σημεία αυτά που θα χρησιμοποιηθούν δημιουργήθηκαν από συναρτήσεις και εξισώσεις καμπυλών στα οποία όμως θα έχει εισαχθεί κάποιος θόρυβος. Έτσι οι αλγόριθμοι δοκιμάζονται για δεδομένα που είναι κοντά και εύκολα επεκτάσιμα στα δεδομένα ενός 3D scanner. Τέλος τα νέφη σημείων είναι διατεταγμένα (ordered) δηλαδή βρίσκονται ήδη ταξινομημένα σε αύξουσα σειρά με βάση τον οριζόντιο άξονα x.

5.2 Κριτήριο αξιολόγησης της κάθε μεθόδου

Για την σωστή και συνεπή αξιολόγηση της κάθε μεθόδου θα πρέπει να δημιουργηθεί ένας αλγόριθμος που να υπολογίζει μία τιμή λάθους που θα ισοδυναμεί με την συνολική απόσταση των 2Δ σημείων από την προσαρμοσμένη καμπύλη. Αυτή η τιμή λάθους θα είναι ανεξάρτητη από τις μεταβλητές του κάθε αλγορίθμου που εξετάζεται. Με αυτόν τον τρόπο δημιουργείται μια γενική συγκριτική εικόνα για κάθε τρόπο προσέγγισης. Η ποιότητα και χρησιμότητα της κάθε προσέγγισης δεν θα εξαρτάται εξ ολοκλήρου από αυτή την τιμή αλλά θα μας δίνει την ακρίβεια του πάνω στα συγκεκριμένα δεδομένα εισόδου.

5.2.1 Ο αλγόριθμος για την διεξαγωγή του κριτηρίου αξιολόγησης

Η πιο απλή και αξιόπιστη τιμή λάθους είναι το άθροισμα της κάθετης απόστασης του κάθε σημείου από την εξίσωση της καμπύλης δηλαδή η προβολή αυτού του σημείου πάνω στην καμπύλη [Εικόνα 5.1]. Για τον υπολογισμό της τιμής: αρχικά υπολογίζουμε την παραγωγό της καμπύλης για να πάρουμε την εξίσωση της εφαπτομένης και με χρήση του εσωτερικού γινομένου βρίσκουμε την ευθεία που είναι κάθετη στην εφαπτομένη. Έπειτα βρίσκουμε το σημείο επαφής των δυο ευθείων για να βρούμε το σημείο πάνω στην καμπύλη και υπολογίζουμε την απόσταση μεταξύ αυτού και του αρχικού σημείου. Έτσι έχουμε την μικρότερη απόσταση του σημείου από την καμπύλη. Επαναλαμβάνουμε αυτή την διαδικασία για κάθε σημείου του αρχικού νέφους σημείων για να υπολογίσουμε την μέση τετραγωνική απόσταση. Έτσι έχουμε το κριτήριο αξιολόγησης μας για τον κάθε τρόπο.

Μαθηματικά αυτό υπολογίζεται ως εξής [25]: Έχοντας ένα σημείο p και μια καμπύλη Bezier Bez(t) βαθμού n το πρόβλημα της προβολής αυτού του σημείου πάνω στην καμπύλη, θέλοντας να βρεθεί t^* που να ελαχιστοποιεί αυτήν την απόσταση, μπορεί να γραφτεί:

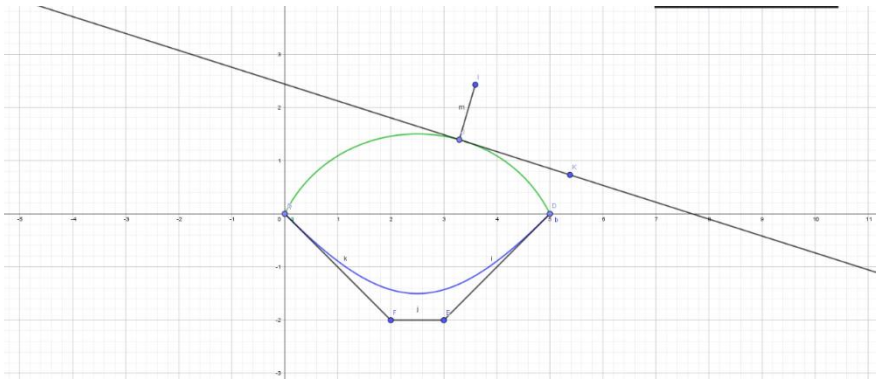
$$\|p - Bez(t^*)\| = \min\{\|p - Bez(t)\| \quad | \quad t \in [0,1]\}$$

Παίρνοντας την παράγωγο της παραπάνω σχέση βάση την μεταβλητή t μπορεί να προκύψει η παρακάτω εξίσωση [25]:

$$\big(p - b(t)\big) * Bez'(t) = 0$$

Για να υπολογιστεί το πλησιέστερο σημείο της καμπύλης αρκεί να βρούμε την ρίζα της παραπάνω εξίσωσης. Αυτή η πολυωνυμική εξίσωση δεν είναι απαραίτητο πως θα έχει πραγματική λύση αρά η απόσταση δεν μπορεί να βρεθεί πάντα με αυτόν τον τρόπο.

Υπάρχουν πολλοί αλγόριθμοι που μπορούν να βρουν ρίζες πολυωνυμικής εξίσωσης σε διάφορες βιβλιοθήκες που παρέχουν προγραμματιστικά εργαλεία αλλά σε αυτήν την εργασία έχει χρησιμοποιηθεί η συνάρτηση roots της βιβλιοθήκης NumPy.



Εικόνα 5.1 Καθετή απόσταση σημείου από την καμπύλη

5.3 Εργαλεία και βιβλιοθήκες υλοποίησης

Για την υλοποίηση νευρωνικού δικτύου χρησιμοποιήθηκαν τα παρακάτω προγραμματιστικά εργαλεία:

1. **Python**: Η γλώσσα προγραμματισμού python είναι μια πιο σύγχρονες και πλέον γνωστές γλώσσες. Είναι μια αντικειμενοστραφής γλώσσα χωρίς να χρειάζεται μεταγλωττιστή (compiler) για την εκτέλεση του κώδικα και έχει την δυνατότητα να τρέχει σε διάφορες πλατφόρμες. Έχει σχεδιαστεί ώστε να είναι εύκολη στην μάθηση και απλή στην χρήση χωρίς να έχει σοβαρά μειονεκτήματα σε σχέσεις με άλλες γλώσσες προγραμματισμού. Η Python παρέχει αναρίθμητα πακέτα στην διάθεση του προγραμματιστή.

2. **TensorFlow**: Το TensorFlow είναι περιβάλλον ανοικτού κώδικα που περιέχει μαθηματικά μοντέλα και μεγάλη ποικιλία από διάφορες βιβλιοθήκες για την δημιουργία προγραμματιστικών εφαρμογών. Η πιο διαδεδομένη χρήση του είναι μέσω της python όμως παρέχεται η δυνατότητα χρήσης του και από άλλες γλώσσες προγραμματισμού. Πιο συχνά χρησιμοποιείται για προβλήματα μηχανικής μάθησης και υλοποίηση νευρωνικών δικτύων. Επίσης έχει δυνατότητα χρήσης της κάρτας γραφικών (GPU acceleration) για επεξεργαστικά βαριές διαδικασίες μέσω CUDA.

3. **Keras**: Το Keras είναι μια διεπαφή προγραμματισμού εφαρμογών (API) που ανήκει στο περιβάλλον του TensorFlow 2.0 και είναι σχεδιασμένο να υλοποιεί προβλήματα μηχανικής μάθησης και deep learning. Είναι σχεδιασμένο για προγραμματισμό σε γλώσσα python και έχει απλή χρήση και ελαχιστοποιεί τον αριθμό των γραμμών που χρειάζονται να γραφτούν για τις πιο συχνές λειτουργίες του. Επίσης παρέχει βοηθητικές προτροπές λάθους για την εύκολη αντιμετώπιση των προβλημάτων.

4. **NumPy**: Η βιβλιοθήκη NumPy είναι από της πιο βασικές βιβλιοθήκη της Python. Προσφέρει καλή διαχείριση πινάκων πολλών διαστάσεων και πράξεις μεταξύ αυτών. Μπορεί να δημιουργήσει πίνακες με σταθερό μέγεθος δυνατότητα που δεν υπάρχει στην Python από μόνη της. Επίσης έχει δυνατότητα για μαθηματικές πράξεις, λογικές πράξεις, πράξεις γραμμικής άλγεβρας και πολλά άλλα.

5. **Matplotlib**: Μια ακόμα από τις βασικές βιβλιοθήκες τις Python είναι η Matplotlib που προσφέρει μια οπτικοποίηση δεδομένων μέσω διαγραμμάτων. Χρησιμοποιεί τις περιγραφές πινάκων τις NumPy για την είσοδο των συναρτήσεων της και έχει μεγάλη ποικιλία σε διαγράμματα έτσι ώστε να καλύπτει της απαιτήσεις των περισσότερων χρηστών.

5.4 Εκτίμηση των συντελεστών της παλινδρόμησης με μέθοδο των ελαχίστων τετραγώνων

Ο τρόπος επίλυσης του προβλήματος της προσαρμογής με την χρήση πολυωνυμικών συναρτήσεων ως μοντέλο της παλινδρόμησης είναι παρόμοιος με αυτόν της γραμμικής παλινδρόμησης. Η καμπύλη Bezier ως μια πολυωνυμική καμπύλη μπορεί να λυθεί ως ένα τέτοιο πρόβλημα βελτιστοποίησης. Αυτή η επίλυση μπορεί να γενικευτεί και για όλες της πολυωνυμικές εύκαμπτες καμπύλες [20].

Για την επίλυση του προβλήματος της προσαρμογής καμπύλης θα πρέπει να προσαρμόσουμε την εξίσωση της καμπύλης Bezier στην λογική της παλινδρόμησης. Στην περίπτωση αυτή, στο συστηματικό μέρος της εξίσωσης ορίζουμε την εξίσωση της καμπύλης Bezier και στον ρόλο των συντελεστών είναι τα σημεία ελέγχου (control points) τα οποία είναι και το ζητούμενο του προβλήματος. Οι εξαρτημένες μεταβλητές Y αντιστοιχίζονται στα $Bez_x(t)$ και $Bez_y(t)$, ενώ η ανεξάρτητη μεταβλητή X θεωρείται η παραμετρική μεταβλητή t. Η εξαρτημένη μεταβλητή , που είναι τα σημεία ελέγχου , αποτελούν γραμμικό συνδυασμό για την έξοδο στην εξίσωση καμπύλης Bezier οπότε το πρόβλημα μπορεί να λυθεί ως γραμμικό πρόβλημα παλινδρόμησης.

Για παράδειγμα θα χρησιμοποιήσουμε την κυβική καμπύλη Bezier η οποία ορίζεται:

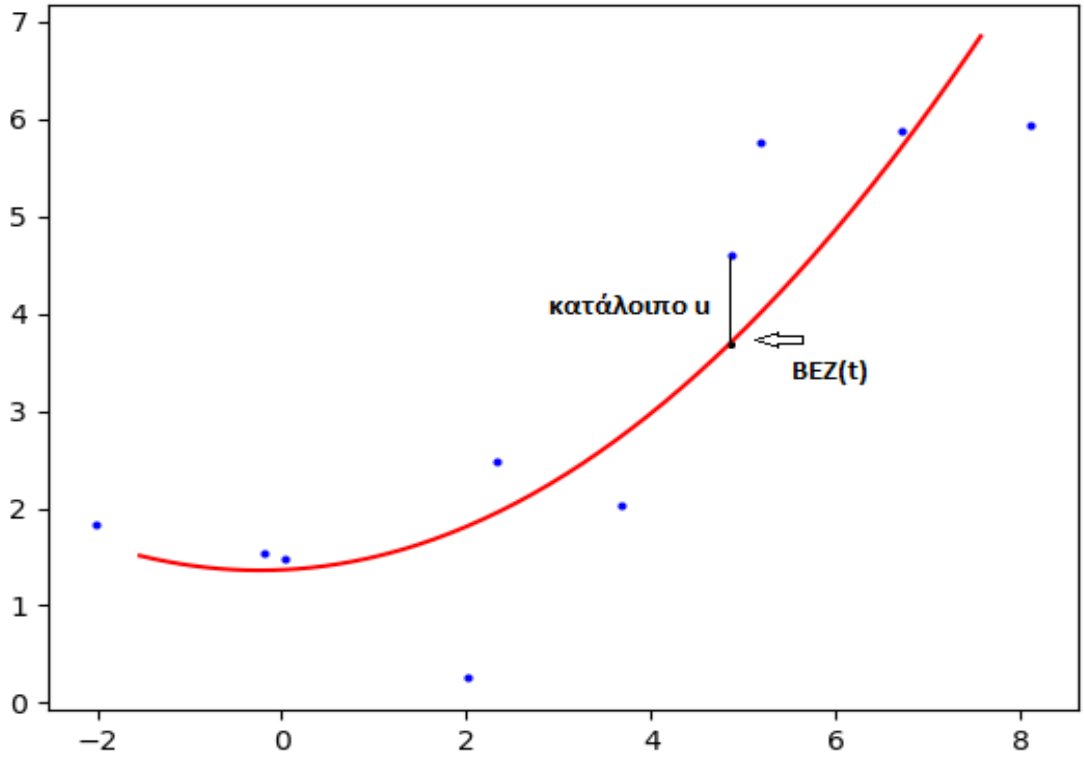
$$Bez(t) = P_0(1-t)^3 + 3P_1t(1-t)^2 + 3P_2t^2(1-t) + P_3t^3$$

P_0, P_1, P_2, P_3 είναι τα σημεία ελέγχου της καμπύλης και οι συντελεστές στην ανάλυση παλινδρόμησης, t ορίζεται από [0,1].

Αν στην εξίσωση εισάγουμε τον όρο u, που αντιπροσωπεύει το σφάλμα, τότε θα πάρει την έξης μορφή:

$$Bez_{error}(t) = Bez(t) + u$$

Αν υποθέσουμε πως έχουμε ένα δείγμα σημείων όπως φαίνεται στην [Εικόνα 5.2] τότε η διάφορα της εκτίμησης και της πραγματικής τιμής είναι το σφάλμα u της εξίσωσης.



Εικόνα 5.2 Σφάλμα εκτίμησης στην καμπύλη

Έτσι αν τα P_0, P_1, P_2, P_3 οι εκτιμητές της εξίσωσης τότε το άθροισμα των τετραγώνων των διαφορών μεταξύ εκτιμώμενων τιμών και των πραγματικών δίνεται από την εξίσωση:

$$\sum_{i=1}^{n} u^2 = \sum_{i=1}^{n} \left(y_i - Bez(t)\right)^2$$

Η μέθοδος των ελαχίστων τετραγώνων ή γνωστή ως LS (Least square) επιλέγει τους εκτιμητές της ώστε να ελαχιστοποιεί την απόσταση όλων των σημείων από την καμπύλη στον κάθετο άξονα y. Για να προσδιορίσουμε αυτούς τους συντελεστές βρίσκουμε την παράγωγο του αθροίσματος ως προς το

κάθε σημείο ελέγχου και το εξισώνουμε με το μηδέν. Έτσι προκύπτει ένα σύστημα εξισώσεων που μας επιτρέπουν να προσδιορίσουμε τα σημεία ελέγχου [20].

5.4.1 Παράδειγμα του αλγορίθμου

Σε αυτό το παράδειγμα θα χρησιμοποιηθεί η κυβική καμπύλη Bezier η οποία ορίζεται:

$$Bez(t) = P_0(1-t)^3 + 3P_1t(1-t)^2 + 3P_2t^2(1-t) + P_3t^3$$

Για να διευκολυνθούν οι πράξεις θα μετατρέψουμε την εξίσωση Bezier σε πράξεις μεταξύ πινάκων.

$$\mathrm{M} = \begin{vmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{vmatrix}$$

$$T = |t^3 \quad t^2 \quad t \quad 1|$$

$$C = \begin{vmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{vmatrix}$$

Η εξίσωση της καμπύλης Bezier τρίτου βαθμού με χρήση των πινάκων γίνεται:

$$Bez_{array}(t) = TMC$$

Για να μπορέσουμε να υπολογίσουμε το σφάλμα σωστά θα πρέπει πρώτα να αντιστοιχίσουμε τα σημεία του συνόλου με την μεταβλητή t, έτσι ώστε να υπάρχει μια τιμή t για κάθε ένα σημείο που θέλουμε να προσαρμόσουμε. Πρώτα θα βρούμε το άθροισμα των αποστάσεων του κάθε σημείου με το επόμενο για να βρούμε το μέγιστο μονοπάτι των σημείων.

$$\mathrm{d_i} = \sum_{j=2}^{n} |P_j - P_{j-1}|$$

Έπειτα θα αντιστοιχίσουμε το κάθε σημείο με ένα ποσοστό πάνω στο μονοπάτι αυτό.

$$t_i = \frac{|\mathrm{d_i} - d_{i-1}|}{\sum_{j=2}^{n} |d_j - d_{j-1}|}$$

Η συνάρτηση λάθους σε αυτό το πρόβλημα την ορίζεται να είναι το άθροισμα του τετράγωνου της απόστασης από το σημείο στην καμπύλη:

$$E(t) = \sum_{i=0}^{n} \left(\mathrm{y_i} - Bez(t_i)\right)^2$$

Όπου y_i είναι η y συντεταγμένη του κάθε σημείου

Τώρα ξανά ορίζουμε την συνάρτηση λάθους χρησιμοποιώντας πίνακες

$$E(C_y) = \left(y - TMC_y\right)^T (y - TMC_y)$$

Έπειτα, υπολογίζουμε την παράγωγο της παραπάνω έκφρασης και την εξισώνοντας με το μηδέν. Έτσι βρίσκουμε το μέγιστο της συνάρτησης.

$$\frac{\partial E}{\partial C} = 0 \Rightarrow -2T^T\left(y - TMC_y\right) = 0$$

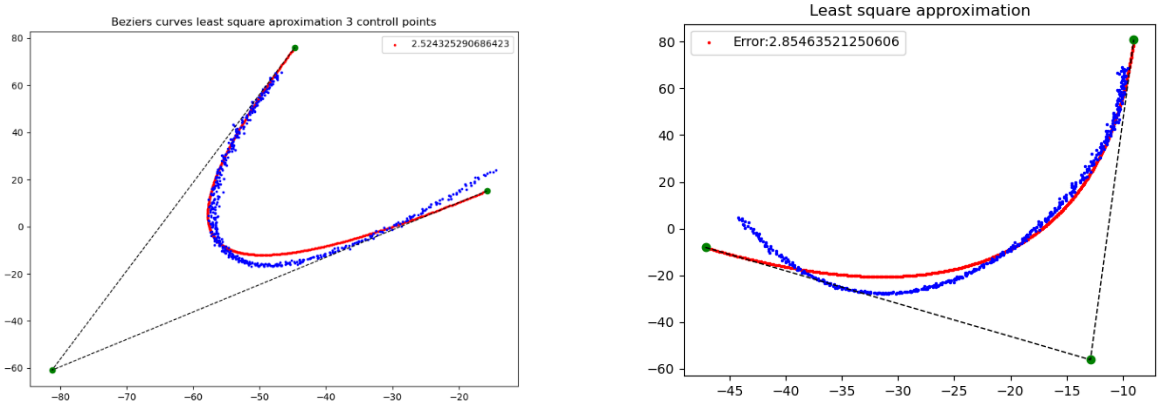
Λύνουμε ως προς τα σημεία ελέγχου.

$$C_y = M^{-1}(T^T T)^{-1} T^T Y$$

Το ίδιο ακριβώς ισχύει και για τις x συντεταγμένες των σημείων ελέγχου δηλαδή

$$C_x = M^{-1}(T^T T)^{-1} T^T X$$

Τα αποτελέσματα αυτής της μεθόδου φαίνονται στην εικόνα [Εικόνα 5.3]



Εικόνα 5.3 Αποτελέσματα της γραμμικής παλινδρόμησης με χρήση least square

5.5 Προσαρμογή της καμπύλης με χρήση μηχανικής μάθησης

Το πρόβλημα της προσαρμογής καμπύλης μπορεί να αντιμετωπιστεί επίσης με νευρωνικά δίκτυα τα οποία εντάσσονται στην κυρίως στην κατηγορία της επιβλεπόμενης μάθησης (supervised learning). Ο αλγόριθμος θα έχει ως βάση του την εξίσωση Bezier και θα προσπαθεί να δημιουργήσει την καμπύλη που είναι πιο κοντά στα επιθυμητά αποτελέσματα δηλαδή το νέφος σημείων προσαρμόζοντας τα σημεία ελέγχου. Δηλαδή ως συντελεστές του προβλήματος κατά την μάθηση θα είναι οι συντεταγμένες των σημείων ελέγχου. Παρακάτω εξετάζονται 2 τρόποι (παράγραφοι 5.5.1 ,5.5.2) που ένα νευρωνικό δίκτυο μπορεί να επιτύχει αυτόν τον σκοπό αλλά κυρίως δίνεται βάση στον δεύτερο γιατί αξιοποιεί καλύτερα τις δυνατότητες της μηχανικής μάθησης και των νευρωνικών δικτύων.

5.5.1 Πρώτη προσέγγιση της προσαρμογής καμπύλης με χρήση νευρωνικού δικτύου

Η προσαρμογή καμπύλης σε 2Δ νέφος σημείων υλοποιήθηκε με τη χρήση ενός νευρωνικού δικτύου με προς-τα-εμπρός-τροφοδότηση. Αυτό το νευρωνικό δίκτυο αποτελείται από ένα επίπεδο κόμβων έτσι ώστε η είσοδος να συνδέεται απευθείας στους κόμβους εξόδου. Αν και αυτό το επίπεδο αποτελείται από μόνο έναν κόμβο, η pre-activation function έχει οριστεί έτσι ώστε να ταυτίζεται με την εξίσωση καμπύλης Bezier ενός συγκεκριμένου βαθμού.

Ο κάθε συντελεστής βάρους αναπαριστά καθένα από τα σημεία ελέγχου της καμπύλης. Με αυτόν τον τρόπο το νευρωνικό δίκτυο μέσα από την διαδικασία της εκπαίδευσης να προσπαθεί να βρει τις κατάλληλες τιμές των βαρών που ταυτίζονται με τους συντελεστές της καμπύλης Bezier. Δηλαδή το νευρωνικό δίκτυο προσπαθεί διορθώνοντας τα σημεία ελέγχου να προσαρμόσει την καμπύλη στα δεδομένα που του παρέχουμε.

Οι εξισώσεις Bezier είναι παραμετρικές και έτσι ορίζονται με διαφορετική συνάρτηση για τον υπολογισμό των x συντεταγμένων και διαφορετική για των υπολογισμό των y συντεταγμένων. Η μόνη διαφορά τους είναι η συντεταγμένη των σημείων ελέγχου δηλαδή για τον υπολογισμό των x πρέπει να πάρουμε τις x συντεταγμένες των σημείων ελέγχου και το αντίστοιχο για τα y. Για τον λόγο αυτό έχουμε δυο εξισώσεις που πρέπει να προσαρμόσουμε στο αρχικό νέφος σημείων και για να το πετύχουμε αυτό θα πρέπει να επαναλάβουμε την εκμάθηση μια φορά για την κάθε συντεταγμένη των σημείων ελέγχου.

Ο κάθε κόμβος του μοντέλου έχει στην pre-activation function του με έναν αριθμό μεταβλητών (weights) [Εικόνα 5.4] που κάθε τέτοιο βάρος αντιπροσωπεύει ένα σημείο ελέγχου της καμπύλης Bezier. Αυτές οι μεταβλητές αρχικοποιούνται σε τυχαίες τιμές και με την εκπαίδευση πλησιάζουν στις σωστές για την κάθε καμπύλη. Το δίκτυο εκπαιδεύεται πάνω στα ίδια δεδομένα του αρχικού νέφους και με την κάθε εποχή καταφέρνει να προσεγγίσει καλύτερα αυτά τα δεδομένα.

Η είσοδος αυτού του δικτύου είναι οι τιμές της παραμετρικής μεταβλητής t. Η έξοδος του δικτύου είναι τα σημεία της καμπύλης που προσαρμόζεται. Για κάθε ένα σημείο του νέφους υπάρχει μια τιμή t που το αντιστοιχίζει. Για την επιλογή της τιμής του t το πεδίο ορισμού του $t \in [0,1]$ διαιρείται σε τόσα διαστήματα όσα και τα σημεία του νέφους .

```
class MyModel(tf.Module):
    def __init__(self, **kwargs):
        super().__init__(**kwargs)
        # Initialize the weights
        # These should be randomly initialized
        self.w = tf.Variable(np.random.uniform(0, 100))
        self.w1 = tf.Variable(np.random.uniform(0, 100))
        self.w2 = tf.Variable(np.random.uniform(0, 100))

    def __call__(self, t):
        return (1 - t) ** 2 * self.w + 2 * (1 - t) * t * self.w1 + (t ** 2) * self.w2
```

Εικόνα 5.4 Μοντέλο εκμάθηση για καμπύλη Bezier δευτέρου βαθμού

Ως συνάρτηση σφάλματος [Εικόνα 5.5] έχει οριστεί η μέση τετραγωνική διαφορά του κάθε σημείου που έχει δημιουργήσει ο αλγόριθμος με τα δεδομένα του νέφους που του έχουν δοθεί ως είσοδο. Αυτή η απόσταση μας δίνει μια πολύ καλή προσέγγιση για το σφάλμα καθώς επίσης μπορεί να υπολογιστεί ταχύτατα μειώνοντας τον χρόνο εκπαίδευσης κατά πολύ. Όταν το σφάλμα φτάσει σε ικανοποιητικά μικρή τιμή τότε από εξάγουμε από την pre activation function τις τιμές των βαρών του νευρωνικού. Με αυτές τις τιμές κατασκευάζουμε την καμπύλη Bezier ως τα σημεία ελέγχου.

```
# This computes a single loss value for an entire batch
def loss(target_y, predicted_y):
    return tf.reduce_mean(tf.square(target_y - predicted_y))
```

Εικόνα 5.5 Συνάρτηση λάθους του δικτύου

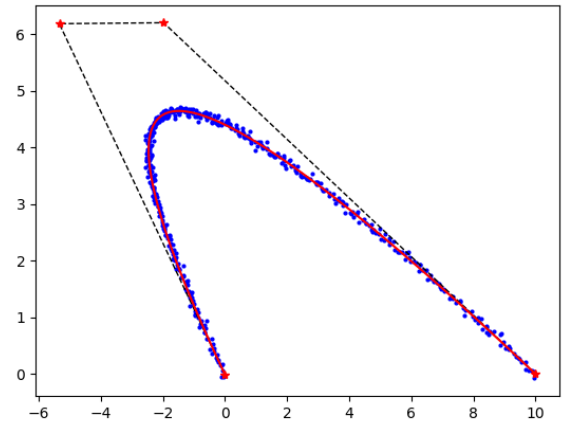
Έπειτα, το μοντέλο εκπαιδεύεται πάνω στα δεδομένα για συγκεκριμένες επαναλήψεις και για κάθε επανάληψη ξανά υπολογίζει το σφάλμα [Εικόνα 5.6], δηλαδή την απόκλιση από τα πραγματικά δεδομένα και τροποποιεί τις μεταβλητές του ώστε να πλησιάσει τα δεδομένα με την μέθοδο gradient descent. Αυτή η μέθοδος προσπαθεί με μικρές μεταβολές των αρχικών παραμέτρων να βρει ένα τοπικό ελάχιστο στην συνάρτηση λάθους και έτσι ένα ικανοποιητικό αποτέλεσμα για το πρόβλημα μας.

```
def train(model, x, y, learning_rate):
    with tf.GradientTape() as t:
        # Υπολογίζει το λαθος για την καθε επαναληψη
        current_loss = loss(y, model(x))

    # Υπολογίζει την αλλαγη που πρεπει να γινει σε καθε μεταβλητή
    dw, dw1, dw2 = t.gradient(current_loss, [model.w, model.w1, model.w2])

    # κανει την αλλαγη στις μεταβλητες μειωμενες απο τον ρυθμο εκμαθησης
    model.w.assign_sub(learning_rate * dw)
    model.w1.assign_sub(learning_rate * dw1)
    model.w2.assign_sub(learning_rate * dw2)
```

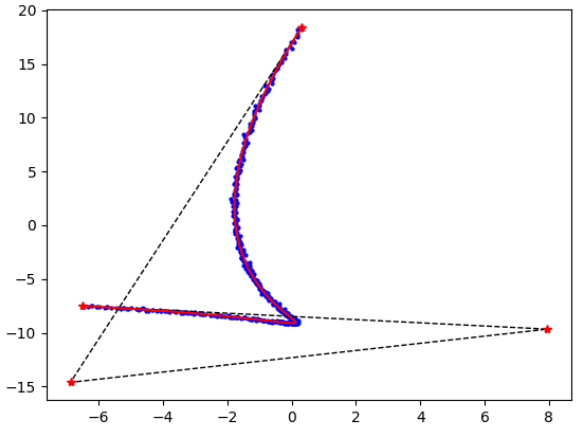
Εικόνα 5.6 Συνάρτηση/ακολουθία εκπαίδευσης

Αυτή η υλοποίηση παρότι μπορεί να δημιουργήσει αρκετά αντιπροσωπευτική καμπύλη για ένα νέφος σημείων απαιτεί να επαναλαμβάνεται η διαδικασία της εκπαίδευσης του δικτύου για κάθε νέφος που θέλουμε να προσεγγίσουμε με αποτέλεσμα να έχει πολύ μεγαλύτερο χρόνο εκτέλεσης. Αυτή η μέθοδος έχει πολλά κοινά στοιχεία με επαναλαμβανόμενους τρόπους επίλυσης που είδαμε στο Κεφάλαιο 4 αλλά χρησιμοποιεί την λογική και την διεπαφή των νευρωνικών δικτύων.

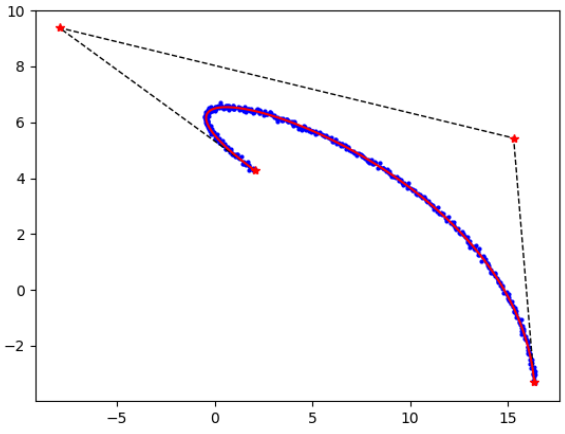
Στις παρακάτω εικόνες [Εικόνα 5.10, Εικόνα 5.9, Εικόνα 5.8, Εικόνα 5.7] φαίνονται τα αποτελέσματα αυτής της μεθόδου για τέσσερα 2Δ νέφη σημείων που προσεγγίζονται με καμπύλες Bezier 3^{ου} βαθμού.



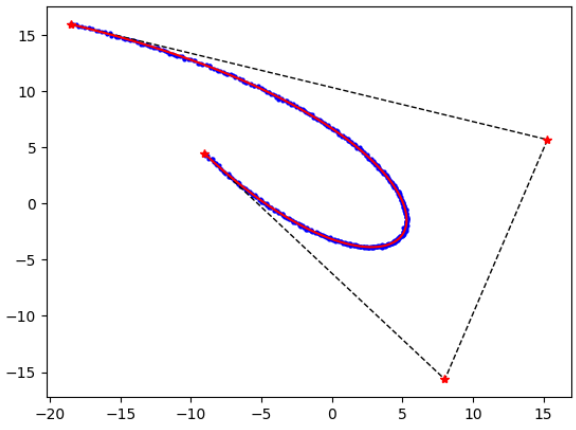
Εικόνα 5.10 Αποτέλεσμα 1



Εικόνα 5.9 Αποτέλεσμα 2



Εικόνα 5.8 Αποτέλεσμα 3



Εικόνα 5.7 Αποτέλεσμα 4

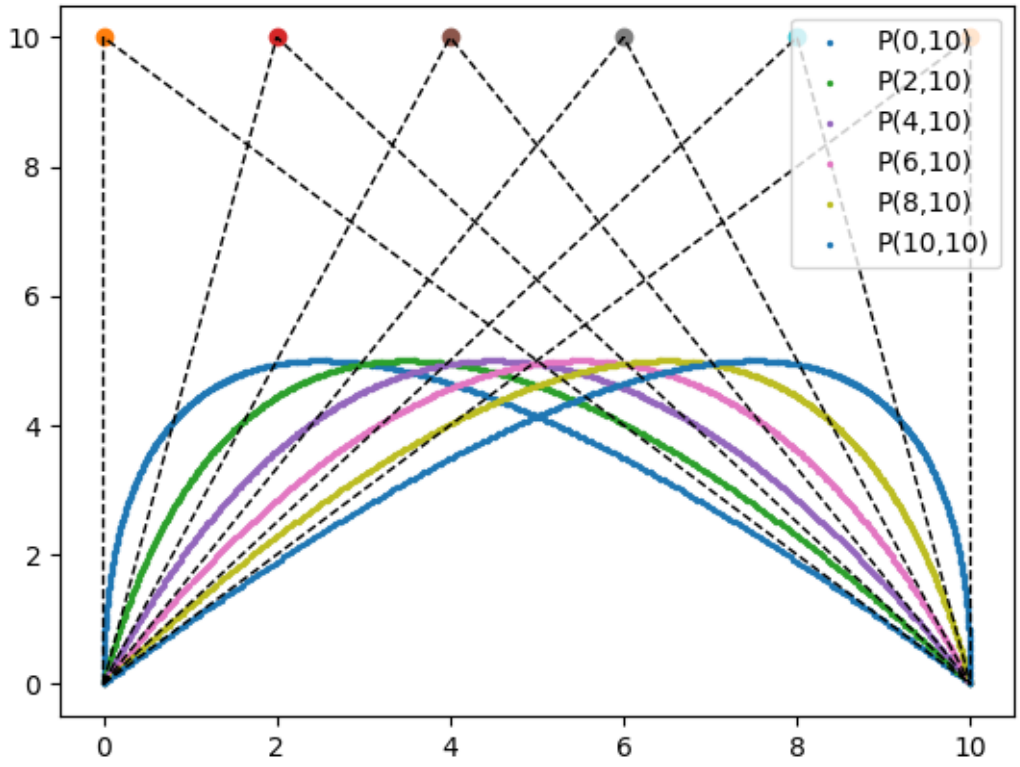
5.5.2 Δεύτερη προσέγγιση της προσαρμογής καμπύλης με χρήση νευρωνικού δικτύου

Σε αυτή την περίπτωση το πρόβλημα της προσαρμογής καμπύλης αντιμετωπίστηκε ως πρόβλημα ταξινόμησης, όπου η βέλτιστη καμπύλη αναζητάτε με τη χρήση νευρωνικού δικτύου το οποίο έχει εκπαιδευτεί νωρίτερα από δεδομένα νέφη και τις αντίστοιχες καμπύλες τους. Αυτή η πλεονεκτεί έναντι της προηγούμενης, όσον αφορά το χρόνο εκτέλεσης καθώς η εκπαίδευση του νευρωνικού δικτύου γίνεται μία φορά. Το δίκτυο μπορεί να αναγνωρίζει καμπύλες μέσα σε ένα συγκεκριμένο εύρος τιμών στο οποίο θα πρέπει να ορίζονται οι καμπύλες. Αυτό το εύρος μπορεί να γίνει όσο μεγάλο επιθυμούμε αλλά με κόστος την διάρκεια εκπαίδευσης και την τελική ταχύτητα του δικτύου. Ένα φανερό μειονέκτημα είναι ότι δεν μπορεί αν εφαρμοστεί για όλες τις καμπύλες λόγω του συγκεκριμένου πεδίου ορισμού του.

Το νευρωνικό δίκτυο έχει να εκπαιδευτεί για να προσεγγίζει καμπύλες Bezier δευτέρου βαθμού. Υπάρχει δυνατότητα επέκτασης και σε μεγαλύτερες τάξης καμπύλης ή ακόμα και σε διαφορετικές παραμετρικές καμπύλες. Η καμπύλης Bezier δευτέρου βαθμού αποτελείται από 3 σημεία ελέγχου. Τα δυο άκρα της καμπύλης όπου είναι και τα σημεία ελέγχου έχουν θεωρηθεί σταθερά ενώ το τρίτο που βρίσκεται ενδιάμεσα στα δυο άκρα είναι ελεύθερο και η θέση του υπολογίζεται από το νευρωνικό δίκτυο.

Στην [Εικόνα 5.11] απεικονίζεται ένα παράδειγμα δεδομένων καμπυλών που χρησιμοποιούνται για την εκπαίδευση του νευρωνικού δικτύου. Σε αυτό το παράδειγμα οι τετραγωνικές καμπύλες Bezier έχουν δημιουργηθεί κρατώντας σταθερά τα ακριανά σημεία ελέγχου ενώ το μεσαίο σημείο ελέγχου μετακινείται ανά μικρές αποστάσεις στο άξονα x. Έτσι δημιουργούνται δέκα απλές κατηγορίες τετραγωνικών καμπυλών Bezier. Πάνω σε αυτές τις καμπύλες έχουν υπολογιστεί σημεία τα οποία θα χρησιμοποιηθούν ως νέφη σημείων για την εκπαίδευση του νευρωνικού δικτύου. Έτσι, δημιουργούνται πολλά δείγματα καμπυλών με τα αντίστοιχα νέφη σημείων. Τα δείγματα αυτά χρησιμοποιούνται για την εκπαίδευση του δικτύου και τη δημιουργία κατηγοριών καμπυλών. Έτσι, το νευρωνικό δίκτυο, θα πρέπει να είναι σε θέση να κατατάξει το ζητούμενο νέφος σημείων σε μια από τις κατηγορίες για τις οποίες έχει

εκπαιδευτεί. Η κατάταξη του ζητούμενου νέφους σημείων επιτυγχάνεται με τον υπολογισμό πιθανοτήτων για την κάθε κατηγορία του νευρωνικού δικτύου.



Εικόνα 5.11 Ένα παράδειγμα κατηγοριοποίησης βάση του μεσαίου σημείου ελέγχου

Αντίστοιχα δείγματα μπορούν να δημιουργηθούν μετακινώντας το μεσαίο σημείο ελέγχου στον άξονα y ή και στους δύο άξονες.

5.5.2.1 Σχεδιασμός του αλγορίθμου

Το πρώτο βήμα σε εφαρμογές νευρωνικών δικτύων είναι να ορίσουμε το μοντέλο του δικτύου. Στην περίπτωση μας θέλουμε ένα δίκτυο που θα έχει ως είσοδο όλα τα δεδομένα ενός νέφους σημείων. Ως έξοδο θα υπολογίζει μία πιθανότητα για κάθε μία από τις κατηγορίες καμπυλών στις οποίες το δίκτυο έχει εκπαιδευτεί. Ακολούθως θα μπορεί να υπολογιστεί το μεσαίο σημείο ελέγχου της τετραγωνικής καμπύλης Bezier. Το δίκτυο θα δέχεται έναν δισδιάστατο πίνακα, με τις συντεταγμένες x,y των σημείων

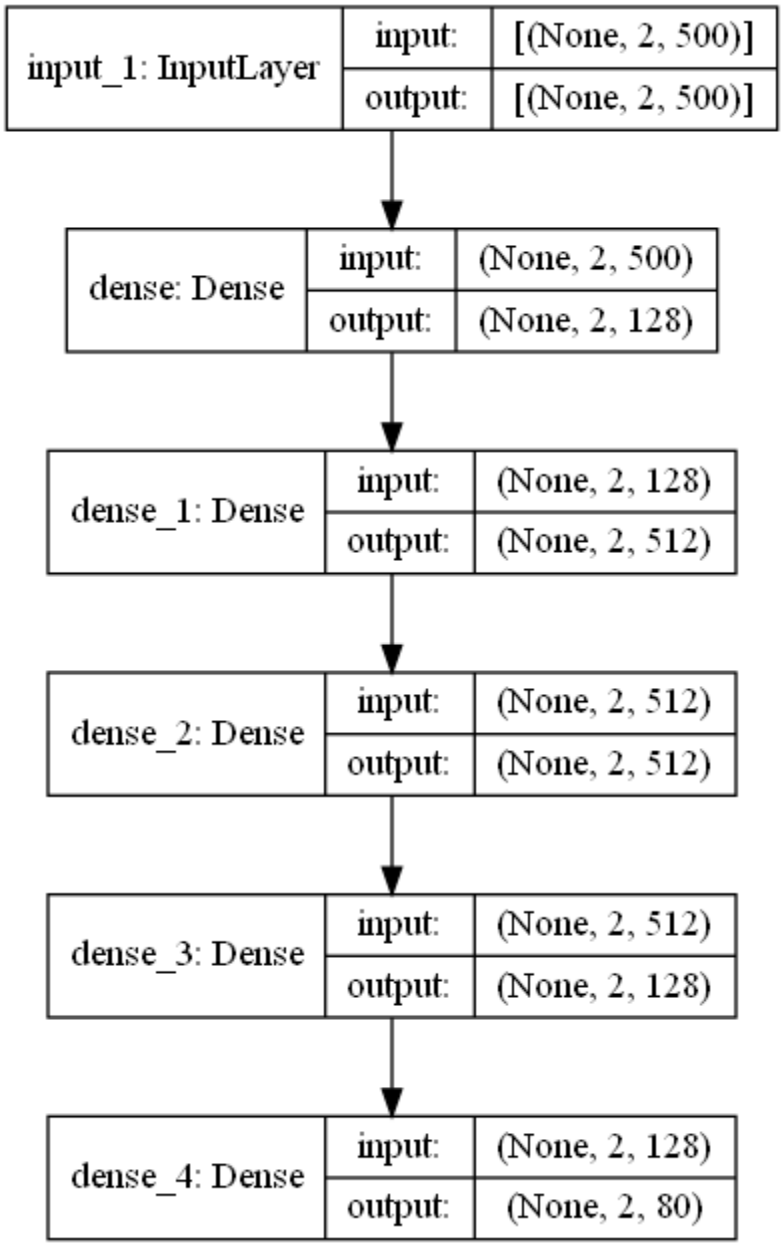
του νέφους ως είσοδο και θα παράγει έναν δισδιάστατο πίνακα, με τις πιθανότητες κάθε καμπύλης, ως έξοδο. Πιο συγκεκριμένα, ο πίνακας εξόδου είναι της μορφής $k \times 2$ και περιέχει τις πιθανότητες για κάθε κατηγορία καμπύλης. Η κάθε καμπύλη είναι συσχετισμένη με ένα συγκεκριμένο σημείο ελέγχου του οποίου οι x,y συντεταγμένες κυμαίνονται σε ένα εύρος $[\alpha, \beta]$. Οι τιμές των α, β ορίζονται σε σχέση και με την απόσταση των ακραίων σημείων ελέγχου ώστε το δίκτυο να εκπαιδεύεται σε μεγάλη ποικιλία καμπυλών. Διαδοχικά σημεία ελέγχου (από διαδοχικές καμπύλες) απέχουν μεταξύ τους σταθερή απόσταση m. Αυτό το εύρος μπορεί να τροποποιηθεί και να αυξηθεί ανάλογα με την κλίμακα που επιθυμούμε να εκπαιδεύσουμε το δίκτυο. Οι k πιθανές τιμές για την κάθε συντεταγμένη x,y του μεσαίου σημείου ελέγχου παράγουν k^2 καμπύλες για την εκπαίδευση του δικτύου. Περεταίρω αύξηση των δειγμάτων k, και τελικά της ακρίβειας των αποτελεσμάτων, μπορεί να επιτευχθεί μειώνοντας το βήμα m μεταξύ των συντεταγμένων των διαδοχικών σημείων ελέγχου.

Η [Εικόνα 5.13] δείχνει μια οπτική αναπαράσταση της δομής του δικτύου σε επίπεδο εισόδων/εξόδων και επίπεδα κόμβων ενώ η [Εικόνα 5.12] δείχνει το μοντέλο σε μορφή κώδικα.

```
def build_model():
    model1 = keras.Sequential([
        keras.Input(shape = (2,N)),
        layers.Dense(128, activation='relu'),
        layers.Dense(512, activation='relu'),
        layers.Dense(512, activation='relu'),
        layers.Dense(128, activation='relu'),
        layers.Dense(units=80,activation=softmax)
    ])

    return model1
```

Εικόνα 5.12 Μοντέλο του νευρωνικού δικτύου

Για να επιτευχθεί η σωστή ενεργοποίηση των κόμβων του δικτύου πρέπει να οριστεί η κατάλληλη συνάρτηση ενεργοποίησης. Έτσι ως συνάρτηση ενεργοποιήσεις των μονάδων των κρυφών στοιβάδων του νευρωνικού δικτύου χρησιμοποιήθηκε η συνάρτηση ReLU. Ενώ για την τελευταία στοιβάδα η συνάρτηση SoftMax η οποία είναι κατάλληλη για να πιθανοτικές εξόδους και εφαρμογές κατηγοριοποίησης καθώς κανονικοποιεί την έξοδο της στοιβάδας σε κατανομή πιθανοτήτων μεταξύ κλάσεων.



Εικόνα 5.13 Νευρωνικό δίκτυο σε μορφή διαγράμματος

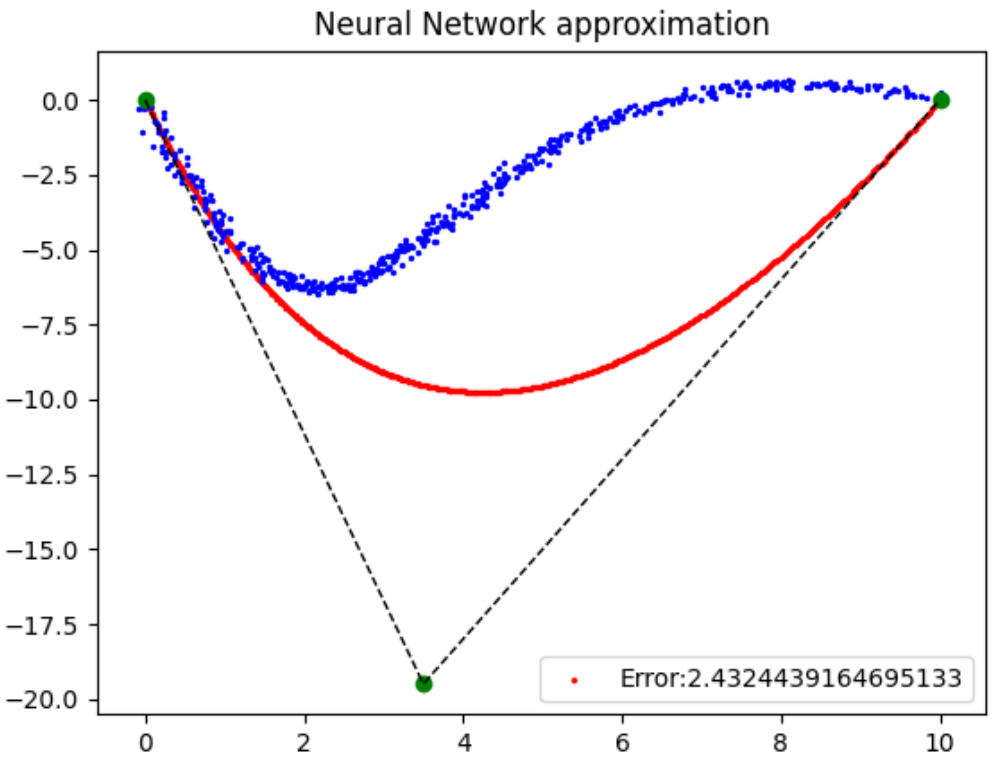
Το επόμενο βήμα στην δημιουργία ενός νευρωνικού δικτύου είναι η δημιουργία μια συνάρτησης λάθους για να μπορέσει να γίνει η εκπαίδευση του. Τα πακέτα υλοποίησης μας δίνουν πρόσβαση σε γενικευμένες συναρτήσεις λάθους και βελτιστοποιητές που αρκούν για την επίλυση των περισσότερων προβλημάτων νευρωνικών δικτύων, αλλά υπάρχει και δυνατότητα δημιουργίας προσαρμοσμένης συνάρτησης για το κάθε πρόβλημα. Η συνάρτηση που επιλέχθηκε είναι η categorical cross-entropy και ως αλγόριθμο βελτιστοποίησης τον Adaptive Moment Estimation (Adam) που υπάρχουν ήδη υλοποιημένοι

στην βιβλιοθήκη Keras. Ο αλγόριθμος Adam χρησιμοποιείται ως βελτιστοποίηση για να επίλυση προβλημάτων επιβλεπόμενης μάθησης και βοηθάει στην εύρεση τις ελάχιστης τιμής της συνάρτηση λάθους δίνοντας έναν συντελεστή ορμής προς την κατεύθυνση που ελαχιστοποιείται η συνάρτηση. Επίσης σε αυτόν τον αλγόριθμο μπορεί να οριστεί μια μεταβλητή που να ελέγχει την ταχύτητα μάθησης του δικτύου (learning rate). Στη συγκεκριμένη υλοποίηση επιλέχθηκε μια μικρή τιμή τις τάξης 10^{-3} διότι θέλουμε να προσεγγίζει την ελάχιστη τιμή της συνάρτησης λάθους με μικρά βήματα ώστε να μην την ξεπεράσει και αρχίσει να ταλαντεύεται γύρω από την ελάχιστη τιμή. Στην [Εικόνα 5.14] φαίνονται αυτοί οι παράμετροι του δικτύου υλοποιημένοι σε μορφή κώδικα.

```
opt = keras.optimizers.Adam(learning_rate=0.001)
loss = keras.losses.SparseCategoricalCrossentropy()
model1.compile(
    optimizer=opt,  # Optimizer
    loss=loss,      # Loss function to minimize
    metrics=['accuracy']
)
plot_model(model1,to_file='mode_plot_1.png',show_shapes=True, show_layer_names=True)
```

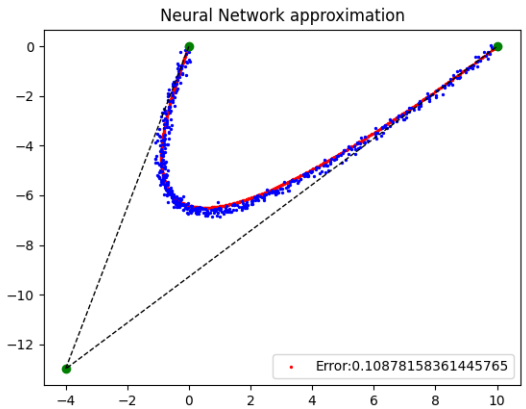
Εικόνα 5.14 Παράμετροι του τελικού μοντέλου δικτύου

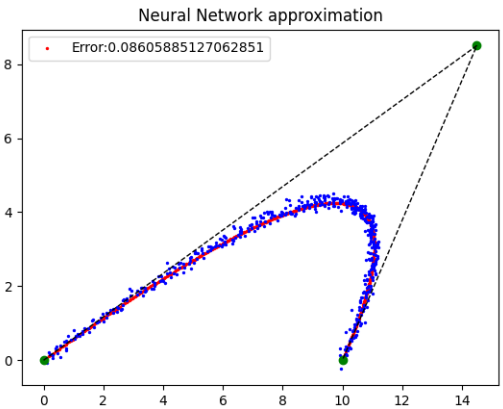
Οι καμπύλες Bezier μικρού βαθμού δεν μπορούν να ορίσουν καμπύλες με μεγάλη πολυπλοκότητα. Οπότε για περιπτώσεις όπως στην [Εικόνα 5.15] αν ο βαθμός της καμπύλης είναι μικρός και η πολυπλοκότητα των δεδομένων είναι μεγάλη τότε η λύση του προβλήματος περιορίζεται από την εξίσωση της καμπύλης και όχι από τον αλγόριθμο προσαρμογής. Ο αλγόριθμος βρίσκει την βέλτιστη προσαρμογή αλλά δεν είναι κατάλληλη για τα συγκεκριμένα δεδομένα.

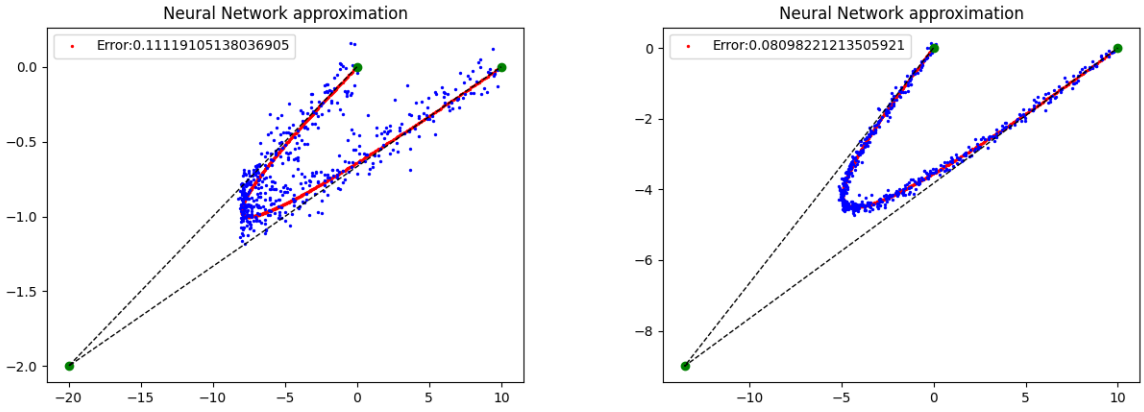


Εικόνα 5.15 Επιλογή της σωστής καμπύλης

Τα αποτελέσματα του δικτύου για τον σωστό βαθμό καμπύλης και για διαφορά επίπεδα θορύβου φαίνονται στην [Εικόνα 5.16]. Το δίκτυο παράγει πολύ ικανοποιητικά αποτελέσματα χωρίς να επηρεάζεται δραστικά από το επίπεδο του θορύβου.







Εικόνα 5.16 Αποτέλεσμα δεύτερης προσέγγισης

5.6 Σύγκριση αποτελεσμάτων και συμπεράσματα

Για την σύγκριση των δυο μεθόδων χρησιμοποιηθήκαν 5 περιπτώσεις χρήσης νεφών σημείων που δημιουργήθηκαν από καμπύλες Bezier δευτέρου βαθμού στις οποίες έχει εισαχθεί θόρυβος. Ο θόρυβος που εισάχθηκε είναι 2 ειδών: α) θόρυβος που ακολουθεί κανονική κατανομή με μέση τιμή 0 και διακύμανση που εξαρτάται από το μέγεθος της καμπύλης και εφαρμόζεται σε όλα τα σημεία της καμπύλης, και β) θόρυβος που τα μετατοπίζει τα τυχαία σημεία μακριά από την αρχική τους θέση.

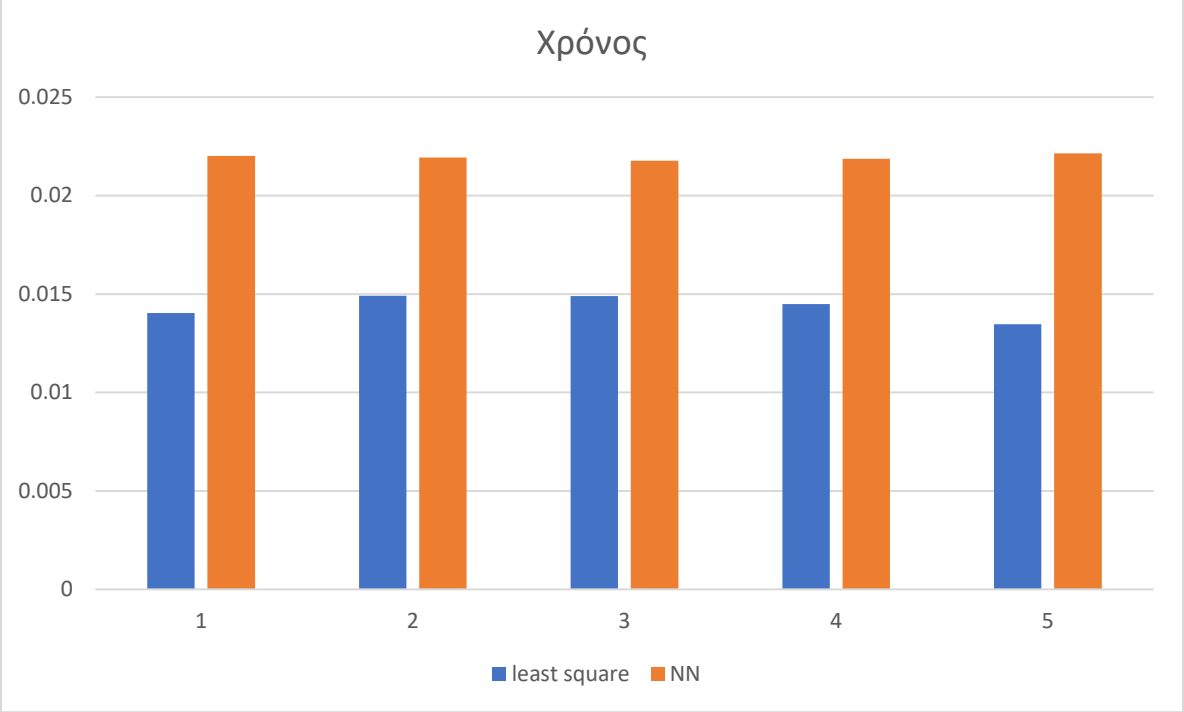
Στους παρακάτω πίνακες βλέπουμε την μέση απόσταση του κάθε σημείου από την υπολογισμένη καμπύλη και τον χρόνο που χρειάστηκε για τον υπολογισμό καθεμιάς από τις περιπτώσεις χρήσης. Στις εικόνες [Εικόνα 5.17, Εικόνα 5.18] φαίνονται τα οπτικοποιημένα αποτελέσματα των πειραμάτων.

Χρόνος(msec)	**Νέφος 1**	**Νέφος 2**	**Νέφος 3**	**Νέφος 4**	**Νέφος 5**
Least square	14.033	14.905	14.888	14.482	13.467
Νευρωνικό δίκτυο	22.02	21.94	21.776	21.861	22.138

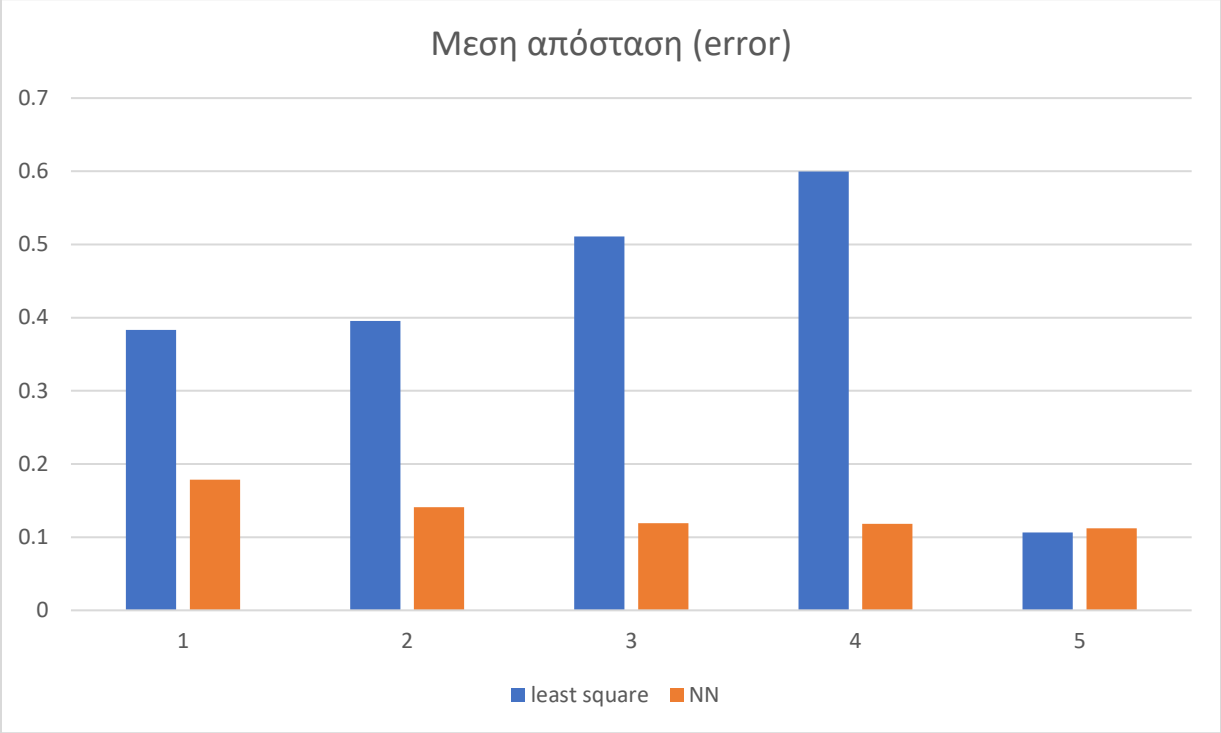
Μέση απόσταση(error)	**Νέφος 1**	**Νέφος 2**	**Νέφος 3**	**Νέφος 4**	**Νέφος 5**

Least square	0.383277	0.395352	0.510774	0.599861	0.106568
Νευρωνικό δίκτυο	0.178744	0.140806	0.119085	0.118264	0.112251

Μελετώντας τα παραπάνω αποτελέσματα παρατηρούμε ότι η μέθοδος των ελαχίστων τετραγώνων επηρεάζεται από τον θόρυβο του νέφους σημείων. Αυτό συμβαίνει διότι η συγκεκριμένη μέθοδος προσδίδει την ίδια βαρύτητα σε όλα τα σημεία του νέφους και προσπαθεί να υπολογίσει την βέλτιστη καμπύλη που μειώνει την τετραγωνική απόσταση όλων των σημείων από την καμπύλη. Έτσι, ο τετραγωνισμός της απόστασης των σημείων θορύβου ωθεί την καμπύλη να απομακρύνεται από τα υπόλοιπα σημεία του νέφους ώστε να πλησιάσει στα σημεία θορύβου.

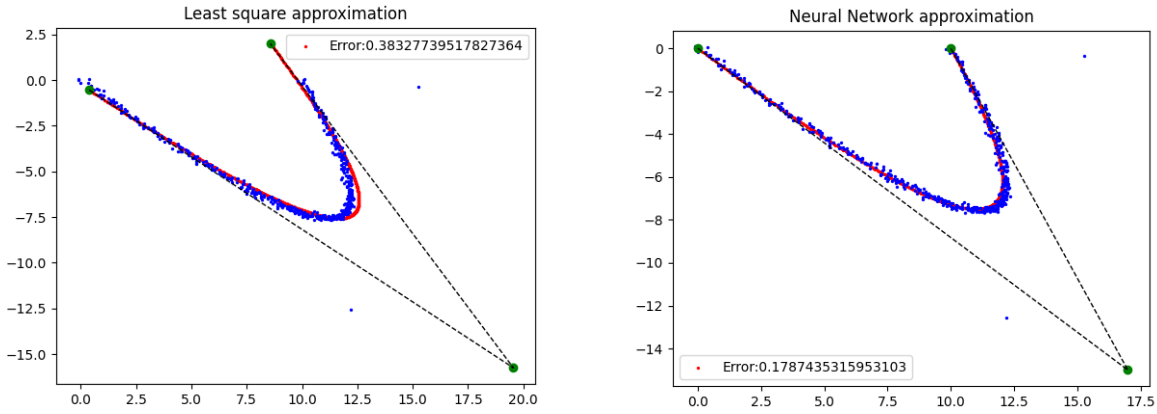


Εικόνα 5.17 Διάγραμμα χρόνου εκτέλεσης

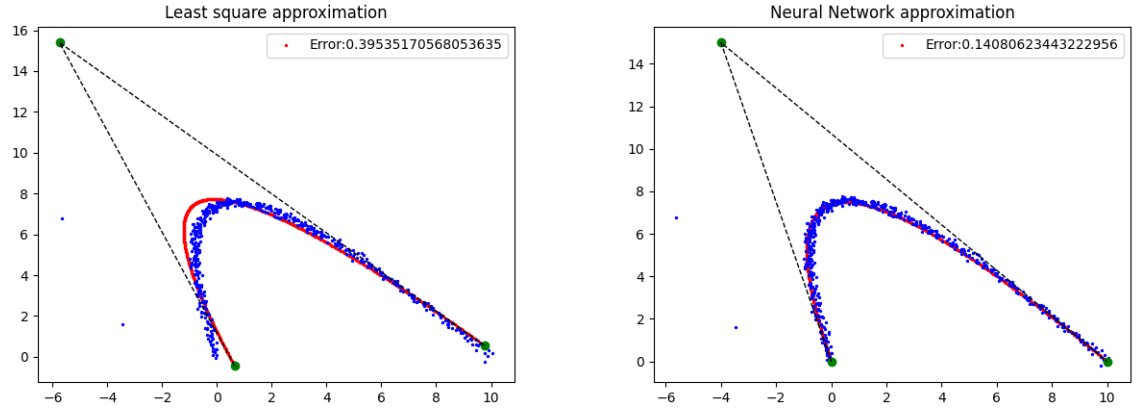


Εικόνα 5.18 Διάγραμμα μέσης απόστασης

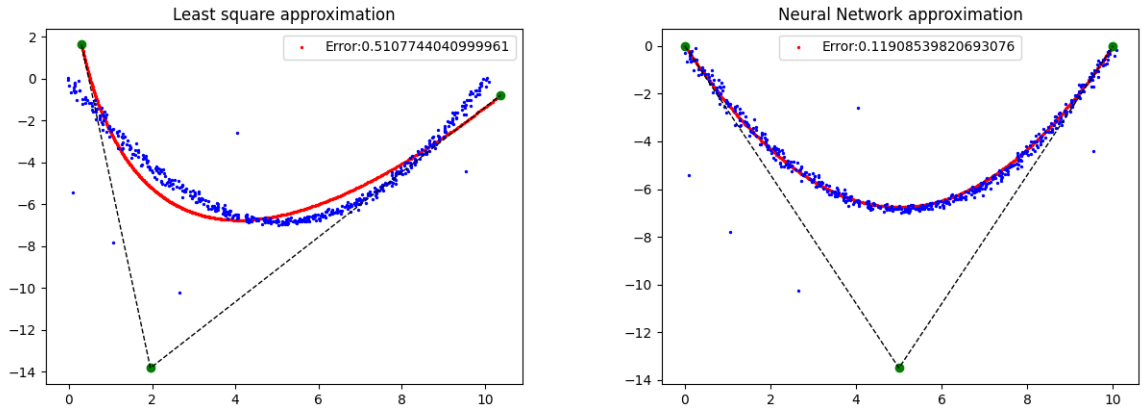
Μετά από σύγκριση των αλγορίθμων συμπεραίνουμε πως ο αλγόριθμος του νευρωνικού δικτύου παράγει καλύτερες καμπύλες σε νέφη σημείων που συμπεριλαμβάνουν θόρυβο ακόμα και αν είναι μόνο ένα σημείο. Από τα διαγράμματα παρατηρούμε πως για περιπτώσεις νεφών σημείων που αναπαρίστανται από τετραγωνικές καμπύλες, η μέση τιμή λάθους του νευρωνικού δικτύου είναι σταθερά μικρότερη ενώ ο χρόνος υπολογισμού είναι σταθερά μεγαλύτερος. Ενδιαφέρουσα περίπτωση θα ήταν η μελέτη περιπτώσεων χρήσης με νέφη σημείων που προσαρμόζονται σε μεγαλύτερου βαθμού καμπύλες. Το συμπέρασμα που εξάγεται είναι ότι ένα καλά εκπαιδευμένο νευρωνικό δίκτυο μπορεί να παράξει καλύτερα αποτελέσματα από την κλασική μέθοδο ελαχίστων τετραγώνων, και σε περιπτώσεις με θόρυβο.



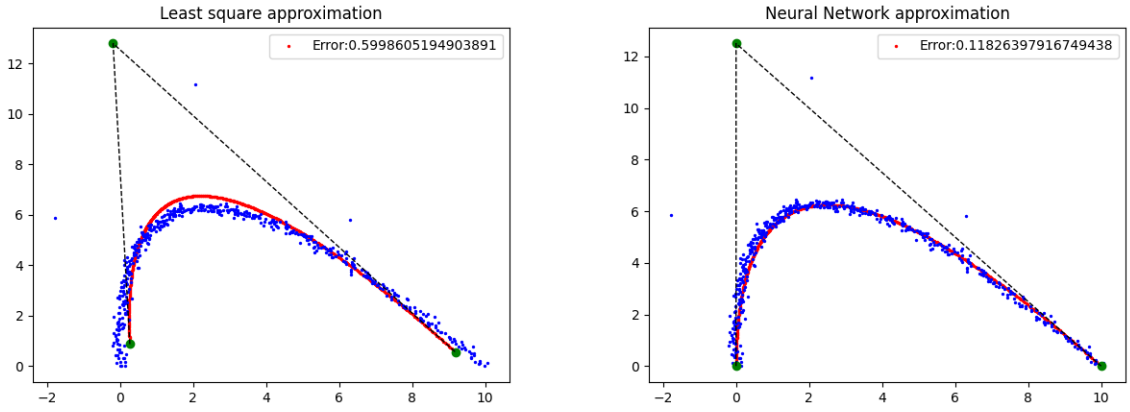
Εικόνα 5.19 Νέφος 1



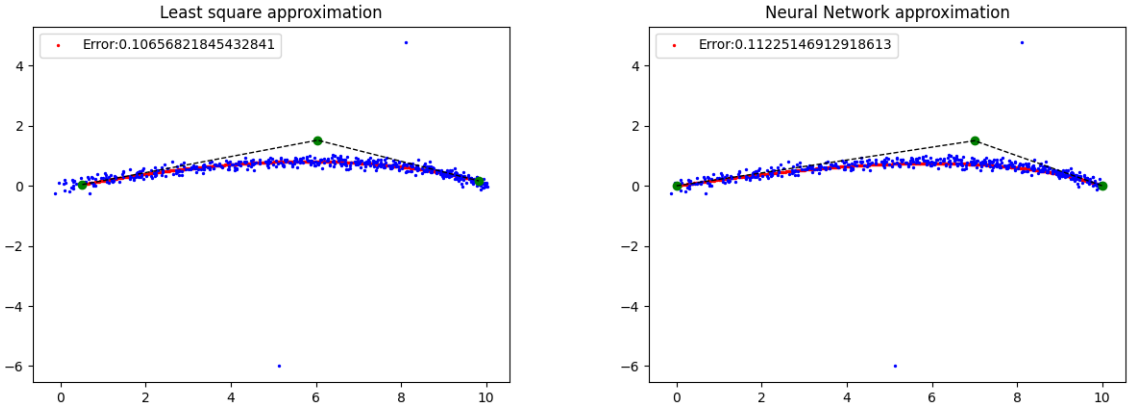
Εικόνα 5.20 Νέφος 2



Εικόνα 5.21 Νέφος 3



Εικόνα 5.22 Νέφος 4



Εικόνα 5.23 Νέφος 5

Εικόνα 5.24 Σύγκριση αποτελεσμάτων μεταξύ του αλγόριθμου least square και νευρωνικού δικτύου

ΚΕΦΑΛΑΙΟ: 6 Σύνοψη και μελλοντικές επεκτάσεις

Στην σχεδίαση με υπολογιστή (Computer-aided design CAD) η αποδοτικότητα και η ακρίβεια των μεθόδων προσαρμογής καμπυλών σε νέφη σημείων είναι απαραίτητη για την βέλτιστη παρουσίαση του τελικού αποτελέσματος. Οι παραδοσιακές μέθοδοι βελτιστοποίησης μπορούν λύσουν το πρόβλημα της προσαρμογής καμπυλών σε νέφη σημείων ως κάποιο βαθμό ακρίβειας. Για την επίτευξη μεγαλύτερης ακρίβειας απαιτούνται νέες σύγχρονες μέθοδοι για το πρόβλημα της προσαρμογής καμπυλών.

Στο πλαίσιο αυτής της διπλωματικής εξετάστηκε ο αλγόριθμος προσαρμογής καμπύλης σε 2Δ νέφος σημείων ελάχιστων τετράγωνων σε αντίστιξη με τον αντίστοιχο αλγόριθμο που κάνει χρήση νευρωνικού δικτύου. Οι δύο μέθοδοι εφαρμόστηκαν για την παραγωγή καμπυλών Bezier. Ο αλγόριθμος ελαχίστων τετραγώνων είναι ένας αλγόριθμος παλινδρόμησης και χρησιμοποιείται για να βρει την βέλτιστη λύση μέσω μιας συνάρτησης λάθους. Αντιθέτως, το νευρωνικό δίκτυο προσπαθεί να βρει μια καμπύλη που ταιριάζει καλύτερα στα δεδομένα του νέφους σημείων, μέσα από ένα σύνολο καμπυλών και σημείων τα οποία εκπαίδευσαν αρχικά το δίκτυο. Το συμπέρασμα που εξάγεται εν τέλει είναι πως το νευρωνικό δίκτυο παράγει ακριβέστερα αποτελέσματα σε περιπτώσεις με θόρυβο ενώ ο χρόνος εκτέλεσης των δύο μεθόδων είναι συγκρίσιμος.

Ο βαθμός της καμπύλης είναι ένας πολύ σημαντικός παράγοντας στο θέμα της προσαρμογής. Αυτό διότι επιτρέπει την προσαρμογή καμπύλης με μεγαλύτερη ακρίβεια πάνω στο σύνολο των δεδομένων. Έτσι, ως μελλοντική κατεύθυνση της συγκεκριμένης έρευνας η διεύρυνση του αλγορίθμου για προσαρμογή καμπύλών Bezier μεγαλύτερου βαθμού καθώς και η προσαρμογή μεγαλύτερου πλήθους από καμπύλες.

Ακόμα μια πολύ ενδιαφέρουσα μελλοντική κατεύθυνση είναι η επέκταση των τεχνικών που παρουσιαστήκαν στις3 διαστάσεις. Αυτό μπορεί να επιτευχθεί με την εισαγωγή της τρίτης συντεταγμένης στις τεχνικές που περιεγράφηκαν στην παρούσα διπλωματική. Τέλος, σημαντικό ερευνητικό ενδιαφέρον έχει η προσαρμογή επιφανειών Bezier σε 3Δ νέφος σημείων.

Βιβλιογραφία

[1] W. Zheng, P. Bo, Y. Liu, and W. Wang, 'Fast B-spline curve fitting by L-BFGS', *Comput. Aided Geom. Des.*, vol. 29, no. 7, pp. 448–462, Oct. 2012, doi: 10.1016/j.cagd.2012.03.004.

[2] A. Zielesny, *From Curve Fitting to Machine Learning: An Illustrative Guide to Scientific Data Analysis and Computational Intelligence*, vol. 18. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. doi: 10.1007/978-3-642-21280-2.

[3] D. Hearn and M. P. Baker, *Computer graphics with OpenGL*, 4th ed. Boston: Addison Wesley, 2011.

[4] H.-W. Lin, C.-L. Tai, and G.-J. Wang, 'A mesh reconstruction algorithm driven by an intrinsic property of a point cloud', *Comput.-Aided Des.*, vol. 36, no. 1, pp. 1–9, Jan. 2004, doi: 10.1016/S0010-4485(03)00064-2.

[5] P. Su and R. L. Scot Drysdale, 'A comparison of sequential Delaunay triangulation algorithms', *Comput. Geom.*, vol. 7, no. 5, pp. 361–385, Apr. 1997, doi: 10.1016/S0925-7721(96)00025-9.

[6] M. Tenney, 'Point Clouds to Mesh in "MeshLab"', *CAST GeoSpatial Methods & Visualization*, Jun. 04, 2012. https://gmv.cast.uark.edu/scanning/point-clouds-to-mesh-in-meshlab/ (accessed Sep. 10, 2021).

[7] W. J. Gordon and R. F. Riesenfeld, 'B-SPLINE CURVES AND SURFACES', in *Computer Aided Geometric Design*, R. E. Barnhill and R. F. Riesenfeld, Eds. Academic Press, 1974, pp. 95–126. doi: 10.1016/B978-0-12-079050-0.50011-4.

[8] S. R. Buss, *3-D computer graphics: a mathematical introduction with OpenGL*. New York: Cambridge University Press, 2003.

[9] 'A Primer on Bézier Curves', Jun. 13, 2013. https://pomax.github.io/bezierinfo (accessed Sep. 14, 2021).

[10] K. I. Joy, 'BERNSTEIN POLYNOMIALS', p. 13.

[11] B. A. Barsky and T. D. DeRose, 'Geometric continuity of parametric curves: three equivalent characterizations', *IEEE Comput. Graph. Appl.*, vol. 9, no. 6, pp. 60–69, Nov. 1989, doi: 10.1109/38.41470.

[12] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*, 2nd ed. Upper Saddle River, N.J: Prentice Hall/Pearson Education, 2003.

[13] R. Y. Choi, A. S. Coyner, J. Kalpathy-Cramer, M. F. Chiang, and J. P. Campbell, 'Introduction to Machine Learning, Neural Networks, and Deep Learning', *Neural Netw.*, p. 12.

[14] A. Zielesny, *From Curve Fitting to Machine Learning: An Illustrative Guide to Scientific Data Analysis and Computational Intelligence*, vol. 18. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. doi: 10.1007/978-3-642-21280-2.

[15] R. A. Dunne and N. A. Campbell, 'On The Pairing Of The Softmax Activation And Cross{Entropy Penalty Functions And The Derivation Of The Softmax Activation Function', p. 5, 1997.

[16] 'CS231n Convolutional Neural Networks for Visual Recognition'. https://cs231n.github.io/neural-networks-1/ (accessed Sep. 10, 2021).

[17] T. A. Pastva, 'Bezier Curve Fitting.', p. 75.

[18] M. Xu, P. Watanachaturaporn, P. Varshney, and M. Arora, 'Decision tree regression for soft classification of remote sensing data', *Remote Sens. Environ.*, vol. 97, no. 3, pp. 322–336, Aug. 2005, doi: 10.1016/j.rse.2005.05.008.

[19] D. F. Specht, 'A general regression neural network', *IEEE Trans. Neural Netw.*, vol. 2, no. 6, pp. 568–576, Nov. 1991, doi: 10.1109/72.97934.

[20] A. Gálvez, A. Iglesias, A. Cobo, J. Puig-Pey, and J. Espinola, 'Bézier Curve and Surface Fitting of 3D Point Clouds Through Genetic Algorithms, Functional Networks and Least-Squares Approximation', in *Computational Science and Its Applications – ICCSA 2007*, vol. 4706, O. Gervasi and M. L. Gavrilova, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 680–693. doi: 10.1007/978-3-540-74477-1_62.

[21] Γ. Ζιουτας, *Πιθανότητες και Στατιστική για Μηχανικούς Μεθοδοι - Εφαρμογές*, 3rd ed. σοφια.

[22] T. Strutz, *Data fitting and uncertainty: a practical introduction to weighted least squares and beyond*. New York, NY: Springer Berlin Heidelberg, 2015.

[23] H. P. Gavin, 'The Levenberg-Marquardt algorithm for nonlinear least squares curve-fitting problems', p. 19, 2020.

[24] C. M. Bishop and C. M. Roach, 'Fast curve fitting using neural networks', *Rev. Sci. Instrum.*, vol. 63, no. 10, pp. 4450–4456, Oct. 1992, doi: 10.1063/1.1143696.

[25] X. Chen, Y. Zho, Z. Shu, and H. Su, 'Improved Algebraic Algorithm on Point projection for Bezier curves', in *Second International Multi-Symposiums on Computer and Computational Sciences (IMSCCS 2007)*, Iowa City, IA, USA, Aug. 2007, pp. 158–163. doi: 10.1109/IMSCCS.2007.17.