**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ**

**ΣΧΟΛΗ ΠΟΛΙΤΙΚΩΝ ΜΗΧΑΝΙΚΩΝ**

ΤΟΜΕΑΣ ΜΕΤΑΦΟΡΩΝ ΚΑΙ ΣΥΓΚΟΙΝΩΝΙΑΚΗΣ ΥΠΟΔΟΜΗΣ

Διπλωματική Εργασία

# Διερεύνηση Ειδικών Μορφών Οριζοντιογραφικών Καμπύλων

**Πονηράκη Αικατερίνη**

*Επιβλέπων: Σ. Μαυρομάτης, Επίκουρος Καθηγητής Ε.Μ.Π.*

Αθήνα, Ιούλιος 2020

**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ**

**ΣΧΟΛΗ ΠΟΛΙΤΙΚΩΝ ΜΗΧΑΝΙΚΩΝ**

ΤΟΜΕΑΣ ΜΕΤΑΦΟΡΩΝ ΚΑΙ ΣΥΓΚΟΙΝΩΝΙΑΚΗΣ ΥΠΟΔΟΜΗΣ

Διπλωματική Εργασία

# Διερεύνηση Ειδικών Μορφών Οριζοντιογραφικών Καμπύλων

**Πονηράκη Αικατερίνη**

*Επιβλέπων: Σ. Μαυρομάτης, Επίκουρος Καθηγητής Ε.Μ.Π.*

Αθήνα, Ιούλιος 2020

**NATIONAL TECHNICAL UNIVERSITY OF ATHENS**

**SCHOOL OF CIVIL ENGINEERING**

DEPT. OF TRANSPORTATION PLANNING AND ENGINEERING

Diploma Thesis

# Investigation of Special Forms of Horizontal Curves

**Poniraki Aikaterini**

*Supervisor: S. Mavromatis, Assistant Professor, N.T.U.A.*

Athens, July 2020

# ΕΥΧΑΡΙΣΤΙΕΣ

Με την περάτωση της παρούσας διπλωματικής εργασίας, η οποία σηματοδοτεί το τέλος των σπουδών μου και την απόκτηση του πτυχίου μου, θα ήθελα να ευχαριστήσω όσους με βοήθησαν και συντέλεσαν με διαφορετικό τρόπο ο καθένας, άμεσα ή έμμεσα, στην ολοκλήρωση των σπουδών μου.

Αρχικά, επιθυμώ να ευχαριστήσω θερμά τον κύριο Μαυρομάτη Στέργιο, Καθηγητή του Τομέα Μεταφορών και Συγκοινωνιακής Υποδομής της Σχολής Πολιτικών Μηχανικών ΕΜΠ, για τη βοήθεια που μου προσέφερε στη σύλληψη του θέματος της παρούσας εργασίας, για την εμπιστοσύνη που μου έδειξε με την ανάθεση αυτής και για την καθοδήγησή του καθ' όλη την διάρκεια.

Θα ήθελα, επίσης, να ευχαριστήσω τον κύριο Φώτη Μερτζάνη, πρώην Επιστημονικό Συνεργάτη της Σχολής Πολιτικών Μηχανικών ΕΜΠ, για την στήριξη και καθοδήγησή του, καθώς και για την ηθική του συμπαράσταση και την αμέριστη υποστήριξη που μου παρείχε.

Επιπλέον, θα ήθελα να εκφράσω τις ευχαριστίες μου στον κύριο Στάμο Αθανάσιο, ΕΔΙΠ-Διδάκτορα του τομέα Δομοστατικής, για τη βοήθεια του στο προγραμματιστικό κομμάτι της διπλωματικής μου.

Τέλος, επιθυμώ να ευχαριστήσω την οικογένεια μου και τους φίλους μου για την ανιδιοτελή στήριξη και συμπαράσταση που μου προσφέρουν όλα αυτά τα χρόνια, σε οποιοδήποτε εγχείρημά μου.

# ΣΥΝΟΨΗ

## Διερεύνηση Ειδικών Μορφών Οριζοντιογραφικών Καμπύλων

**Πονηράκη Αικατερίνη**

**Επιβλέπων: Σ. Μαυρομάτης, Επίκουρος Καθηγητής Ε.Μ.Π.**

Αντικείμενο της παρούσας Διπλωματικής Εργασίας είναι η διερεύνηση μιας αποτελεσματικής και χρηστικής μεθόδου υπολογισμού και σχεδιασμού των πιο διαδεδομένων σύνθετων οριζοντιογραφικών καμπυλών.

Η έρευνα περιλαμβάνει βιβλιογραφική ανασκόπηση των συνηθέστερων τύπων οριζοντιογραφικών καμπυλών προκειμένου να αιτιολογηθεί η επιλογή των πιο κρίσιμων.

Οι καμπύλες που αναλύθηκαν είναι η Σιγμοειδής και η Ωοειδής, οι οποίες έχουν ιδιαίτερα αυξημένο φόρτο υπολογισμών και σχεδίασης.

Στην συνέχεια, για την κάθε καμπύλη, δημιουργήθηκαν αναλυτικές εξισώσεις, οι οποίες υπολογίζουν τα βασικά γεωμετρικά χαρακτηριστικά τους, και διερευνήθηκαν οι παράμετροι που πρέπει να ικανοποιούνται με βάση τις υφιστάμενες οδηγίες σχεδιασμού οδών.

Για τις ίδιες καμπύλες σε επόμενο στάδιο, διερευνήθηκε η πληρότητα που παρέχουν τα σχεδιαστικά και υπολογιστικά λογισμικά που υπάρχουν στην αγορά. Από τη διαδικασία αυτή προέκυψε ότι τα υπόψη λογισμικά δεν είναι προσαρμοσμένα στις ανάγκες του μηχανικού, δεδομένου ότι έχουν δημιουργηθεί με «μαθηματικό» τρόπο προσέγγισης.

Προκειμένου να περιοριστεί αυτή η έλλειψη στοχευμένων λογισμικών, δημιουργήθηκε κατάλληλο λογισμικό σε περιβάλλον Ηλεκτρονικού Υπολογιστή, το οποίο προσαρμόστηκε στις ανάγκες του μηχανικού, καθιστώντας το έτσι ένα πολύ χρήσιμο και εύχρηστο εργαλείο. Το εν λόγω εργαλείο, αποτελεί μια αυτοματοποιημένη διαδικασία υπολογισμών και σχεδίασης η οποία προσφέρει στον μηχανικό δυνατότητες οι οποίες μέχρι τώρα δεν είχαν αναλυθεί σε προγραμματιστικό περιβάλλον.

**Λέξεις – κλειδιά**

Οριζοντιογραφία, Σιγμοειδής Καμπύλη, Ωοειδής Καμπύλη, Python, GUI, AutoCAD

# ABSTRACT

## Investigation of Special Forms of Horizontal Curves

**Poniraki Aikaterini**

**Supervisor: S. Mavromatis, Assistant Professor N.T.U.A.**

This diploma thesis aims to explore an effective and useful method of calculating and designing the most widely used types of horizontal curves.

The conducted research includes the literature review of the most common types of horizontal curves in order to validate and justify the study of the most critical ones. The analysed road curves are the S-curve and the Oval-curve, which take significant effort to be calculated and designed.

Subsequently, for each of the curves, analytical equations were created in order to compute their basic geometry characteristics. Additionally, the parameters that need to be accommodated based on the road designing guidelines were studied.

Afterwards, an investigation was conducted regarding the plenitude provided by existed calculating and designing operating systems. The outcome of this investigation proved that those operating systems are not adjusted to the engineering needs given that they were approached as a typical mathematical problem.

In order to reduce the absence of targeted operating systems, an appropriate software was developed, which was adjusted to engineering needs n order for it to be a very useful and practical tool. This tool is actually an automated calculating and designing operating system which provides capabilities that had never been developed by any software to date.

**Keywords**

Horizontal Alignment, S-Curve, Oval-Curve, Python, GUI, AutoCAD

# *ΠΕΡΙΕΧΟΜΕΝΑ*

# ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

# ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

# ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

# 1. ΕΙΣΑΓΩΓΗ

## 1.1 ΓΕΝΙΚΗ ΑΝΑΣΚΟΠΗΣΗ

Οδός είναι η στενή σχετικά λωρίδα εδάφους, η οποία έχει διαμορφωθεί έτσι, ώστε να είναι δυνατή επάνω σε αυτή η κυκλοφορία ανθρώπων και οχημάτων. Η οδοποιία μελετά τον τρόπο κατασκευής της οδού έτσι ώστε να ανταποκρίνεται πληρέστερα στον προορισμό της.

Ο πρώτος ακούσιος οδοποιός ήταν εκείνος ο οποίος καταδιωκόμενος ή διώκοντας, με τους αγκώνες του και το σώμα του άνοιξε την πρώτη οδό μέσα στο δάσος. Η πρώτη αυτή ατραπός χρησιμοποιήθηκε και από δεύτερο και τρίτο και έτσι δημιουργήθηκε η πρώτη βατή οδός, χωρίς βέβαια ο άνθρωπος να επέμβει για την παραπέρα διαμόρφωσή της. Σύμφωνα με τον Ηρόδοτο, η αρχαιότερη λιθόστρωτη οδός κατασκευάστηκε το 3000 π.Χ. στην Αίγυπτο από το βασιλιά Χέοπα, για τη μεταφορά υλικών προς κατασκευή της μεγάλης πυραμίδας.

Τις οδούς δημιούργησε η ανάγκη για την μεταξύ των ανθρώπων μετακίνηση, με σκοπό την εύρεση τροφής, την επιβολή κυριαρχίας, την επικοινωνία και τη συναλλαγή.

Οι προσπάθειες των ανθρώπων για την δημιουργία των πρώτων βατών οδών περιορίζονταν μόνο στο να συντομεύσουν τις αποστάσεις ή στο να δημιουργήσουν λιγότερο κοπιώδη και ακίνδυνη διαδρομή. Την πρόοδο όμως και την ανάπτυξη του πολιτισμού ακολούθησε πιστά και η τεχνική για την ανάπτυξη και κατασκευή οδών. Άρχισε δηλαδή η εφαρμογή μελέτης, έστω και στοιχειώδους, για την κατασκευή οδών. Η χρησιμοποίηση τροχηλάτων αμαξών είχε ως επακόλουθο οι πρώτοι αυτοσχέδιοι οδοποιοί να κατασκευάσουν οδούς με ελαττωμένες κλίσεις και να επινοήσουν πιο τέλειο και πιο στέρεο οδόστρωμα. Με την εξέλιξη όμως του πολιτισμού και με τις συνεχώς αυξανόμενες απαιτήσεις της ζωής, η αναζήτηση και η μελέτη συντομότερων κατευθύνσεων, η εκτέλεση χωματουργικών έργων, η κατασκευή γεφυρών και γενικά τεχνικών έργων ήταν επιτακτική ανάγκη. Έτσι, για να αναπτύσσονται μεγαλύτερες ταχύτητες στις οδούς έπρεπε να δημιουργηθούν ανθεκτικές και λείες επιφάνειες κυλίσεων. Με τον τρόπο αυτό άρχισε η συστηματικότερη μελέτη της οδού.

Με την πάροδο των χρόνων όμως και την εξέλιξη της αυτοκινητοβιομηχανίας, κατασκευάστηκαν ταχύτερα οχήματα τα οποία δημιουργήσαν περισσότερες ανάγκες για πιο εμπεριστατωμένες μελέτες. Αυτό είχε ως αποτέλεσμα αναγκαιότητα βελτίωσης των οδικών υποδομών ως προς την παρεχόμενη ασφάλεια καθώς και την καλύτερη ποιότητα κυκλοφοριακής ροής.

Η ποιότητα των οδικών υποδομών μιας χώρας, εκφράζει σε μεγάλο βαθμό το δείκτη ανάπτυξης και ευημερίας της. Τα οδικά δίκτυα αποτελούν το μεγαλύτερο τμήμα της συγκοινωνιακής υποδομής, όπου μέσω του γεωμετρικού και λειτουργικού σχεδιασμού οδών πρέπει να εξυπηρετούνται συγκεκριμένοι στόχοι με κυριότερους τους εξής:

- Ασφάλεια
- Εύρυθμη κυκλοφοριακή ροή
- Διαφύλαξη ποιότητας φυσικού και ανθρωπογενούς περιβάλλοντος
- Εξοικονόμηση φυσικών διαθεσίμων

Από την παραπάνω ιεράρχηση πρέπει να γίνει σαφές ότι πρακτικά είναι αδύνατη η πλήρης επίτευξη του ενός στόχου χωρίς να υπάρχουν αρνητικές συνέπειες στην προσπάθεια υλοποίησης των υπόλοιπων. Δηλαδή, με άλλα λόγια, η όποια προσπάθεια ταυτόχρονης επίτευξης των παραπάνω στόχων διά μέσου της "βέλτιστης" όδευσης είναι ανέφικτη, δεδομένου ότι το βασικό εργαλείο υλοποίησής τους, ο γεωμετρικός σχεδιασμός, αποβλέπει στην εύρεση μιας αποδεκτής συμβιβαστικής λύσης, όπου σε κάθε περίπτωση, η ασφάλεια των χρηστών αποτελεί το κύριο μέλημα.

Σε ό,τι αφορά στην ασφάλεια μιας οδού, οι καθοριστικοί παράγοντες επιρροής της είναι η γεωμετρία της οδού καθώς και ο γενικότερος παρόδιος σχεδιασμός, το όχημα κίνησης και τέλος ο χρήστης. Μεταξύ των παραγόντων αυτών υπάρχει σχέση αλληλεξάρτησης και προκειμένου ο σχεδιασμός και η λειτουργία ενός κυκλοφοριακού έργου να θεωρηθεί επιτυχής, απαιτείται η επιτυχής σύμπραξη και των τριών.

Ενώ όμως η γεωμετρία μιας οδού αλλά και το όχημα κίνησης σε αυτή είναι μεταβλητές προβλέψιμες στις οποίες η παρέμβαση, τουλάχιστον σε εμπειρικό επίπεδο, δύναται να δώσει θετικά αποτελέσματα, η συμπεριφορά του χρήστη (οδηγού) είναι απρόβλεπτη και μόνο ως στοχαστική μεταβλητή μπορεί να προσομοιωθεί. Ως εκ τούτου η μελέτη της υπόψη παραμέτρου επιβάλλει τη συνδρομή και άλλων επιστημών (πχ. της ψυχολογίας).

Σε ό,τι αφορά στο χρήστη, τα τελευταία χρόνια ακολουθείται μια πιο ολοκληρωμένη προσέγγιση, σύμφωνα με την οποία ο σχεδιασμός της οδού πρέπει να προσαρμόζεται και να ικανοποιεί τις ανάγκες καθώς και τις προσδοκίες του. Επιπλέον, αυτή η διαδικασία πρέπει να λαμβάνει χώρα με τέτοιο τρόπο, ώστε να μην απαιτείται εκ μέρους του ιδιαίτερη προσπάθεια, διατηρώντας, όσο είναι δυνατό, σε χαμηλά επίπεδα το φόρτο ενασχόλησής του (workload). Σε αντίθετη περίπτωση, αυξάνονται οι πιθανότητες λανθασμένης εκτίμησης του οδικού περιβάλλοντος και ως επακόλουθο ο χρήστης είναι εκτεθειμένος σε οδηγικούς ελιγμούς οι οποίοι δύνανται να προκαλέσουν ατύχημα.

Κρίσιμη παράμετρος για την ασφάλεια του χρήστη είναι η επιλογή της ταχύτητας κίνησης, η οποία, μέσω κατάλληλου γεωμετρικού σχεδιασμού, πρέπει να παραμένει κατά το δυνατόν σταθερή και κυρίως να αποφεύγονται οι απότομες εναλλαγές στην τιμή της.

Ο γεωμετρικός σχεδιασμός οδού αναφέρεται σε τρισδιάστατη (3D) επιφάνεια η οποία όμως απεικονίζεται και μελετάται σε τρία επιμέρους στάδια-επίπεδα δισδιάστατης (2D) απεικόνισης. Συγκεκριμένα, τα 2D αυτά στάδια, εκτός από διαφορετικά είναι και ανεξάρτητα μεταξύ τους με την έννοια ότι η αλληλεπίδραση των επιμέρους παραμέτρων βασίζεται κυρίως σε εμπειρικές προσεγγίσεις. Τα επιμέρους στάδια-επίπεδα δισδιάστατης (2D) απεικόνισης είναι τα εξής:

- Οριζοντιογραφία (οριζόντια χάραξη), η οποία αποτελεί την κατακόρυφη προβολή της οδού στο οριζόντιο επίπεδο και περιλαμβάνει αλληλουχία ευθύγραμμων και καμπύλων τμημάτων
- Μηκοτομή (κατακόρυφη χάραξη) ή κατά μήκος τομή, η οποία ορίζεται ως η πλάγια προβολή στο κατακόρυφο επίπεδο του "ανοιγμένου" μήκους (αναπτύγματος) του άξονα της οδού και συνίσταται από καμπύλα τμήματα (κυρτά και κοίλα) καθώς και τμήματα σταθερής κατά μήκος κλίσης

• Διατομή, η οποία αφορά στην προβολή μιας τομής της οδού, κάθετης προς τον άξονά της, στο κατακόρυφο επίπεδο, περιλαμβάνει εγκάρσιες κλίσεις (επικλίσεις) και ως προς το έδαφος μπορεί να είναι σε όρυγμα, σε επίχωμα ή μικτή.

Κατά την διαδικασία του γεωμετρικού σχεδιασμού, εξίσου μεγάλη έμφαση δίνεται στην επίτευξη της ομοιογένειας και της συνοχής των επιμέρους στοιχείων. Είναι απαραίτητο να αποφεύγονται απότομες αλλαγές τόσο στην γεωμετρία της οδού όσο και στην λειτουργική ταχύτητα των οχημάτων ώστε να μειωθεί ο φόρτος ενασχόλησης του οδηγού και να εξαλειφθεί το αίσθημα του «αιφνιδιασμού». Είναι σημαντικό η γεωμετρία της οδού να ανταποκρίνεται στις προσδοκίες του οδηγού, αποφεύγοντας πιθανούς απότομους ελιγμούς οι οποίοι δύνανται να οδηγήσουν σε συγκρούσεις.

Όπως γίνεται κατανοητό, η οδός αποτελεί ένα σύνθετο τεχνικό έργο με το οποίο διασφαλίζεται η αναγκαία ομαλή επιφάνεια για την ασφαλή, άνετη, οικονομική και γρήγορη κυκλοφορία οχημάτων και πεζών. Για τον λόγο αυτό η μελέτη και η χάραξη των οδών απαιτεί προσοχή στη λεπτομέρεια, ακρίβεια, γνώση του αντικειμένου και των οδηγιών σε βάθος και σχολαστικότητα. Ωστόσο, δεδομένου ότι πρέπει να λαμβάνονται υπόψιν πολλές παράμετροι ταυτόχρονα, ειδικά σε έργα μεγάλου μεγέθους, πλέον κρίνεται αναγκαία η χρήση εξειδικευμένων λογισμικών που αυτοματοποιούν την διαδικασία του σχεδιασμού.

Στα επόμενα κεφάλαια παρουσιάζεται η προσπάθεια δημιουργίας ενός τέτοιου προγράμματος το οποίο αυτοματοποιεί την διαδικασία μελέτης και σχεδίασης της οδού.

## 1.2 ΣΤΟΧΟΙ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Η εκπόνηση της παρούσας διπλωματικής εργασίας έχει ως κύριο στόχο την αυτοματοποίηση ως προς τις διαδικασίες υπολογισμών και σχεδιασμού σύνθετων οριζοντιογραφικών καμπυλών. Αυτό κρίθηκε αναγκαίο, καθώς στις περισσότερες περιπτώσεις, οι καμπύλες αυτές, αν και επιφέρουν σταδιακή προσαρμογή στις αναμενόμενες ταχύτητες των οχημάτων και άρα βελτιώνουν σημαντικά την ποιότητα της χάραξης από άποψη συνοχής και ομοιογένειας των επιμέρους στοιχείων, δεδομένου του αυξημένου υπολογιστικού φόρτου, δεν εφαρμόζονται.

Οι καμπύλες αυτές γενικά εφαρμόζονται σε συνθήκες περιορισμένου διαθέσιμου χώρου και είναι η Σιγμοειδής που χρησιμοποιείται όπου χρειάζεται σύνδεση δύο αντίρροπων κυκλικών τόξων και η Ωοειδής που χρησιμοποιείται όπου απαιτείται σημαντική αλλά ομαλή αλλαγή στην καμπυλότητα της οδού. Στο πλαίσιο της εργασίας αυτής δημιουργήθηκαν δύο υπολογιστικοί αλγόριθμοι, οι οποίοι διευκολύνουν την υπολογιστική διαδικασία για την εύρεση των χαρακτηριστικών σημείων και μηκών των επιμέρους τμημάτων αυτών των δύο καμπυλών.

Επίσης, στην εν λόγω διπλωματική εργασία, παρατίθενται οι σχετικοί κανονισμοί ΟΜΟΕ-Χ 2001 , RAL 2012, RAA 2008 με σκοπό τον προσδιορισμό των οριζοντιογραφικών παραμέτρων και των δεσμεύσεων των δύο αυτών καμπυλών. Στο πλαίσιο της ολοκληρωμένης μελέτης χάραξης της οδού έγινε προσπάθεια σύνδεσης με το πρόγραμμα FM19 του επιστημονικού συνεργάτη του Εθνικού Μετσόβιου Πολυτεχνείου κ. Φώτη Μερτζάνη, το οποίο διευκόλυνε την διερεύνηση και πέρα από τα πλαίσια της οριζοντιογραφικής μόνο προσέγγισης.

Τέλος ένας από τους πιο σημαντικούς στόχους ήταν η συσχέτιση με αντίστοιχα λογισμικά σχεδιασμού καμπυλών με σκοπό την αξιολόγηση των νέων προγραμμάτων αλλά και την εξακρίβωση της αναγκαιότητας ύπαρξής τους. Με αυτό τον τρόπο, το τελικό προϊόν της παρούσας εργασίας καθίσταται ένα χρήσιμο εργαλείο για το μελετητή οδικών έργων, καθώς διευκολύνεται η διαδικασία υπολογισμών και σχεδιασμού οδών, σε όλα τα στάδια μελέτης.

## 1.3 ΔΟΜΗ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Στο υπό-κεφάλαιο παρουσιάζεται η δομή της διπλωματικής εργασίας και μια συνοπτική περιγραφή των κεφαλαίων που την απαρτίζουν.

ΚΕΦΑΛΑΙΟ 1
Στο πρώτο κεφάλαιο πραγματοποιείται μια σύντομη εισαγωγή. Καταγράφεται ο αντικειμενικός σκοπός και οι στόχοι της εργασίας. Επίσης παρουσιάζεται η δομή της και τα κεφάλαια που την απαρτίζουν.

ΚΕΦΑΛΑΙΟ 2
Το δεύτερο κεφάλαιο περιλαμβάνει το θεωρητικό υπόβαθρο της χάραξης της οδού και συγκεκριμένα της οριζοντιογραφίας. Παρουσιάζονται οι πιο διαδεδομένοι τύποι οριζοντιογραφικών καμπυλών με ανάλυση της γεωμετρίας τους αλλά και πλήρη παρουσίαση των οδηγιών που οφείλουν να ακολουθούν.

ΚΕΦΑΛΑΙΟ 3
Στο τρίτο κεφάλαιο επιλέγονται οι πιο διαδεδομένες σύνθετες καμπύλες προς περαιτέρω ανάλυση και παρουσιάζεται ο τρόπος ανάπτυξης και λειτουργίας του προγράμματος καθώς και τα εξαγόμενα αποτελέσματά του. Το κεφάλαιο αυτό παρουσιάζει βήμα προς βήμα την διαδικασία που ακολουθήθηκε, από τον καθορισμό των δεδομένων εισόδου, την δημιουργία του κώδικα μέχρι και την παρουσίαση και περιγραφή των δεδομένων εξόδου. Στο τέλος αυτού του κεφαλαίου αναλύεται η σύνδεση του προγράμματος με το λογισμικό FM19, ο σκοπός που έγινε αυτή η σύνδεση και τα πλεονεκτήματα που δημιουργούνται μέσω αυτής.

ΚΕΦΑΛΑΙΟ 4
Στο τέταρτο κεφάλαιο γίνεται η συσχέτιση των λογισμικών που αναλύθηκαν στο κεφάλαιο 3 με αντίστοιχα λογισμικά που υπάρχουν στην αγορά. Αυτό πραγματοποιείται μέσω ενός παραδείγματος Σιγμοειδούς και ένα Ωοειδούς Καμπύλης.

ΚΕΦΑΛΑΙΟ 5
Στο πέμπτο κεφάλαιο παρέχεται στον αναγνώστη συνοπτική εικόνα των συμπερασμάτων της παρούσας εργασίας καθώς και προτάσεις για περεταίρω έρευνα σχετικά με το αντικείμενο της χάραξης.

ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ
Στο παρόν κεφάλαιο παρατίθεται κατάλογος της πλήρους βιβλιογραφίας που χρησιμοποιήθηκε για την εκπόνηση της παρούσας διπλωματικής εργασίας.

ΠΑΡΑΡΤΗΜΑ
Τέλος, το παράρτημα περιλαμβάνει τον κώδικα του προγράμματος, που αναπτύχθηκε στην παρούσα διπλωματική εργασία, σε γλώσσα προγραμματισμού Python. Αποτελείται από τέσσερα μέρη τα οποία αφορούν τις συναρτήσεις που χρησιμοποιεί το κάθε πρόγραμμα.

# *ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ*

## 2.1 Η ΧΑΡΑΞΗ ΤΗΣ ΟΔΟΥ

Με την έννοια «χάραξη» εννοείται η λεπτομερής μελέτη και κατασκευή της πορείας του άξονα του κυκλοφοριακού δρόμου. Η χάραξη της οδού περιλαμβάνει τη μελέτη:

- Οριζοντιογραφίας
- Μηκοτομής
- Κατά πλάτος τομών - Διατομών (πλάτος, διαπλατύνσεις, επικλίσεις)
- Ορατότητας
- Οδού στο χώρο

και πραγματοποιείται με την εναρμόνιση διαφόρων στοιχείων της μελέτης, τα οποία πρέπει να καθορίζονται με βάση την ταχύτητα μελέτης.

## 2.2 ΟΙ ΘΕΜΕΛΕΙΩΔΕΙΣ ΑΡΧΕΣ ΤΗΣ ΧΑΡΑΞΗΣ

Κατά την χάραξη επιδιώκεται πορεία της οδού κατάλληλη για την ομαλή ροή της κυκλοφορίας, ευνοϊκή για την ασφαλή και ομαλή κίνηση των οχημάτων και ορθή από την άποψη της ψυχολογίας των οδηγών. Απαραίτητα συνεπώς είναι τα κριτήρια τα οποία επιτρέπουν την κατασκευή κατάλληλων, ασφαλών οδών επαρκούς κυκλοφοριακής ικανότητας με λογικό κόστος.

Μία οδός πρέπει να διαμορφώνεται κατασκευαστικά με βάση τις ίδιες θεμελιώδεις αρχές ώστε να παρουσιάζει ενιαία χαρακτηριστικά. Η πορεία του άξονα πρέπει να είναι ομαλή και η διατομή πρέπει να διατηρεί τις ίδιες διαστάσεις κατά τα τμήματα ενός συνολικού οδικού έργου. Αντίστοιχα, οι αρχές διάταξης και διαμόρφωσης κόμβων, συμβολής και προσβάσεων στα παρόδια γήπεδα, πρέπει να διατηρούνται αμετάβλητες. Τέλος η πορεία του άξονα, η διατομή, η διαμόρφωση των κόμβων και ο έλεγχος των προσβάσεων πρέπει να εναρμονίζονται μεταξύ τους. Με αυτόν τον τρόπο ο οδηγός είναι σε θέση να ρυθμίσει κατάλληλα την ταχύτητα του οχήματος του, χωρίς να υποχρεώνεται σε συνεχείς αλλαγές ταχύτητας εξαιτίας των μεταβαλλόμενων οδικών και κυκλοφοριακών συνθηκών.

## 2.3 ΣΤΟΙΧΕΙΑ ΜΕΛΕΤΗΣ ΚΑΤΑ ΤΗΝ ΟΡΙΖΟΝΤΙΟΓΡΑΦΙΑ

Η οριζοντιογραφία της οδού συνίσταται από τα εξής τρία στοιχεία μελέτης:

- Ευθυγραμμίες
- Κυκλικά τόξα
- Τόξα συναρμογής

### 2.3.1 Ευθυγραμμία

Η ευθυγραμμία αποτελεί το πιο απλό και διαδεδομένο ιστορικά στοιχείο μελέτης οδών. Η συντομότερη καμπύλη μεταξύ δύο σημείων είναι η ευθεία. Η ιδιότητα αυτή ήταν ο λόγος για τον οποίον πίστευαν για πολύ χρόνο ότι η χάραξη πρέπει κατά το δυνατό να χρησιμοποιεί ευθείες. Ο άλλος λόγος ήταν η προσπάθεια να μειωθούν κατά το δυνατό στο ελάχιστο οι απαιτούμενοι χειρισμοί του τιμονιού από τον οδηγό ώστε να μην χρειάζεται ένταση της προσοχής.

Σήμερα η ευθεία χρησιμοποιείται στην οδοποιία σε μικρή έκταση, επειδή η αυστηρή της γεωμετρία συχνά επιβάλλει άχαρες χαράξεις, οι οποίες δύσκολα προσαρμόζονται στο τοπίο. Επίσης οι μεγάλες ευθυγραμμίες κουράζουν τον οδηγό, αυξάνουν την νύχτα τον κίνδυνο θάμβωσης από τα φώτα και δυσχεραίνουν την εκτίμηση των αποστάσεων και ταχυτήτων άλλων οχημάτων.

Η χρήση ευθυγραμμίας προσφέρεται σαν στοιχείο χάραξης υπό ορισμένες συνθήκες, όπως σε εκτεταμένα επίπεδες περιοχές ή σε στενές κοιλάδες. Επίσης καμμιά φορά χρειάζεται σε τμήματα απεμπλοκής της κυκλοφορίας, σε περιοχές ισόπεδων και ανισόπεδων κόμβων προκειμένου να ικανοποιηθούν πολεοδομικές δεσμεύσεις ή για την εξασφάλιση των αναγκαίων μηκών ορατότητας για στάση και προσπέραση. Τέλος η ευθυγραμμία υπερτερεί σε περιοχές προσαρμογής της χάραξης με σιδηροδρομικές γραμμές, αγωγούς καθώς και άλλα μεγάλα τεχνικά έργα.

Για τους λόγους που προαναφέρθηκαν τα μήκη των ευθυγραμμιών έχουν αυστηρά όρια. Συγκεκριμένα, οι μέγιστες τιμές μηκών ευθυγραμμιών με σταθερή κατά μήκος κλίση κατά ΟΜΟΕ-Χ, 2001 είναι:

$$maxL_{ευθ} = 20 * V_e \qquad (2.1)$$

όπου :
$maxL_{ευθ}$ : μέγιστο μήκος ευθυγραμμίας (m)
$V_e$ : ταχύτητα μελέτης οδού (km/h)

Επίσης, μεταξύ ομόρροπων καμπυλών, προκειμένου ο οδηγός να αντιλαμβάνεται επαρκώς την μετάβαση από την πρώτη καμπύλη στη δεύτερη, πρέπει να αποφεύγονται οι μικρές ευθυγραμμίες. Το ελάχιστο μήκος ευθυγραμμίας κατά ΟΜΟΕ-Χ, 2001 σε αυτές τις περιπτώσεις είναι:

$$minL_{ευθ} = 6 * V_e \qquad (2.2)$$

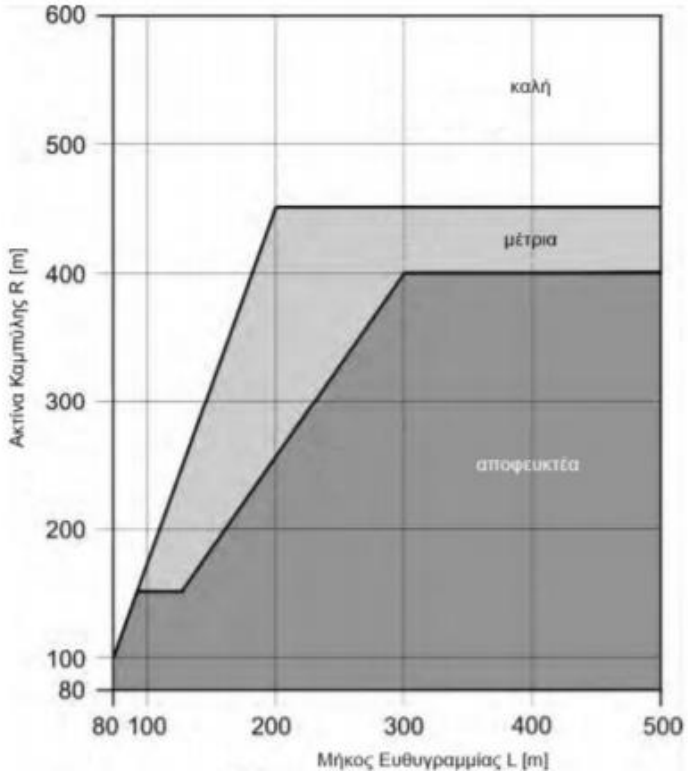
όπου :
$minL_{ευθ}$ : ελάχιστο μήκος ευθυγραμμίας μεταξύ ομόρροπων καμπυλών (m)
$V_e$ : ταχύτητα μελέτης της οδού (km/h)
Στις Γερμανικές οδηγίες υπεραστικών οδών (RAL, 2012) και αυτοκινητοδρόμων (RAA, 2008) αντίστοιχες τιμές δίνονται με βάση το ζεύγος Κατηγορία οδού (κλάση) – Ταχύτητα (Πίνακας 2.1).

Με βάση πιο σύγχρονες αντιλήψεις στο γεωμετρικό σχεδιασμό οδών, η ευθυγραμμία θεωρείται δυναμικό στοιχείο μελέτης. Δηλαδή το μήκος της ευθυγραμμίας επηρεάζει άμεσα την ταχύτητα κίνησης των οχημάτων και άρα την παρεχόμενη ασφάλεια της οδού. Έρευνες έχουν δείξει ότι τόσο η αποφυγή απότομων εναλλαγών στην ταχύτητα κίνησης, όσο και η συμμόρφωση των οδηγών με τα όρια ταχυτήτων είναι περισσότερο εμφανείς πρακτικές όταν δεν σχεδιάζονται μεγάλα μήκη ευθυγραμμιών, ιδιαίτερα όταν αυτά προηγούνται από τόξα μεγάλης καμπυλότητας. Άρα οι ευθυγραμμίες αποτελούν βασικό στοιχείο για την επίτευξη μιας ομοιογένειας στις τιμές ταχυτήτων μεταξύ διαδοχικών στοιχείων μελέτης. Προς την κατεύθυνση αυτή οι οδηγίες ΟΜΟΕ-Χ έχουν υιοθετήσει συγκεκριμένα ποσοτικά κριτήρια (Κριτήρια Ασφάλειας), μέσω των οποίων αξιολογείται το μήκος της ευθυγραμμίας.

| | RAA, 2008 | | | RAL, 2012 | | |
|---|---|---|---|---|---|---|
| V (km/h) | Κλάση Οδού | $L_{min}$ (m) | $L_{max}$ (m) | Κλάση Οδού | $L_{min}$ (m) | $L_{max}$ (m) |
| 130 | ΕΚΑ 1A | 400 | 2000 | | | |
| 120 | ΕΚΑ 1B | 400 | 2000 | | | |
| 110 | | | | ΕΚL 1 | 600 | 1500 |
| 100 | ΕΚΑ 2 | 400 | 2000 | ΕΚL 2 | 600 | 1500 |
| 90 | | | | ΕΚL 3 | 600 | 1500 |
| 80 | ΕΚΑ 3 | 400 | 2000 | | | |
| 70 | | | | ΕΚL 4 | 400 | 1500 |

Πίνακας 2.1: Μέγιστα και ελάχιστα μήκη ευθυγραμμιών με σταθερή κατά μήκος κλίση ($L_{max}$) και μεταξύ ομόρροπων καμπυλών ($L_{min}$) αντίστοιχα, σύμφωνα με τις Γερμανικές οδηγίες RAA και RAL.

Η συσχέτιση μεταξύ μήκους ευθυγραμμίας και ακτίνας κυκλικού τόξου σύμφωνα με τις Γερμανικές οδηγίες RAL, 2012 δίνεται στο επόμενο σχήμα (Σχήμα 5.1)
Στις Γερμανικές οδηγίες RAA, 2008 για αυτοκινητόδρομους, αναγνωρίζοντας ότι τα μεγάλα μήκη ευθυγραμμιών, ιδιαίτερα όταν δεν υπάρχει περιορισμός της επιτρεπόμενης ταχύτητας, έχουν ως αποτέλεσμα πολύ μεγάλες ταχύτητες κίνησης, ορίζεται ότι μετά από ευθυγραμμίες μήκους L > 500 m θα πρέπει να τηρείται ελάχιστη ακτίνα Rmin=1300m.




Σχήμα 2.1. Σχέση μεταξύ ακτίνας καμπύλης και μήκους ευθυγραμμίας.

### 2.3.2 Κυκλικό Τόξο

Από την πρώτη εποχή της οδοποιίας, το κυκλικό τόξο αποτελούσε το συνηθισμένο μέσο σύνδεσης δύο ευθειών όταν απαιτούνταν μεταβολή κατεύθυνσης. Η ακτίνα του κυκλικού τόξου επιλέγεται με βάση την ταχύτητα μελέτης με τέτοιο τρόπο ώστε η οδός να προσαρμόζεται κατά μορφή και μέγεθος με το ανάγλυφο. Γενικά θα πρέπει να αποφεύγεται η επιλογή οριακών τιμών ακτινών.
Σύμφωνα με τις OMOE-X, 2001 τα κυκλικά τόξα πρέπει επίσης να έχουν μήκος τουλάχιστον ίσο με εκείνο που διανύεται σε 2 δευτερόλεπτα από όχημα κινούμενο με την ταχύτητα μελέτης. Δηλαδή ισχύει:

$$minb = \frac{2}{3.6} * V_e \qquad (2.3)$$

όπου :
minb : ελάχιστο μήκος κυκλικού τόξου (m)
$V_e$ : ταχύτητα μελέτης οδού (km/h)

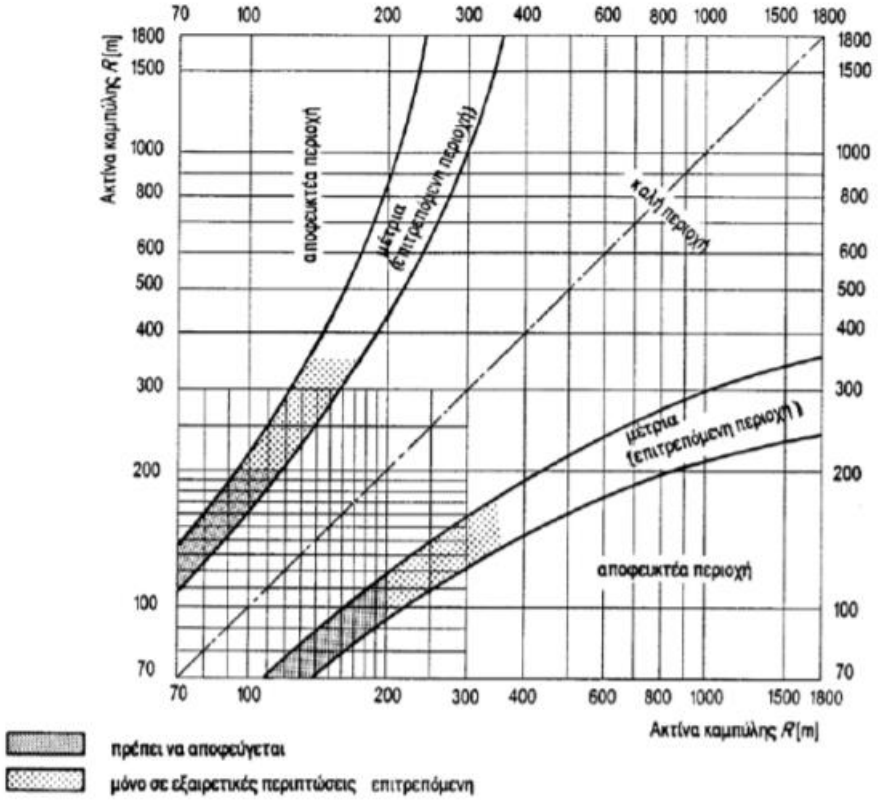
Στον Πίνακα 2.2 δίδονται τα ελάχιστα μήκη κυκλικού τόξου (minb) κατά RAA (2008), RAL (2012) και OMOE-X (2001).

| | RAA, 2008 | | RAL, 2012 | | OMOE-X, 2001 |
|---|---|---|---|---|---|
| V (km/h) | Κλάση Οδού | minb (m) | Κλάση Οδού | minb (m) | minb (m) |
| 130 | ΕΚΑ 1A | 75 | | | 72 |
| 120 | ΕΚΑ 1B | 75 | | | 67 |
| 110 | | | ΕΚL 1 | 70 | 61 |
| 100 | ΕΚΑ 2 | 55 | ΕΚL 2 | 60 | 56 |
| 90 | | | ΕΚL 3 | 50 | 50 |
| 80 | ΕΚΑ 3 | 55 | | | 44 |
| 70 | | | ΕΚL 4 | 40 | 39 |

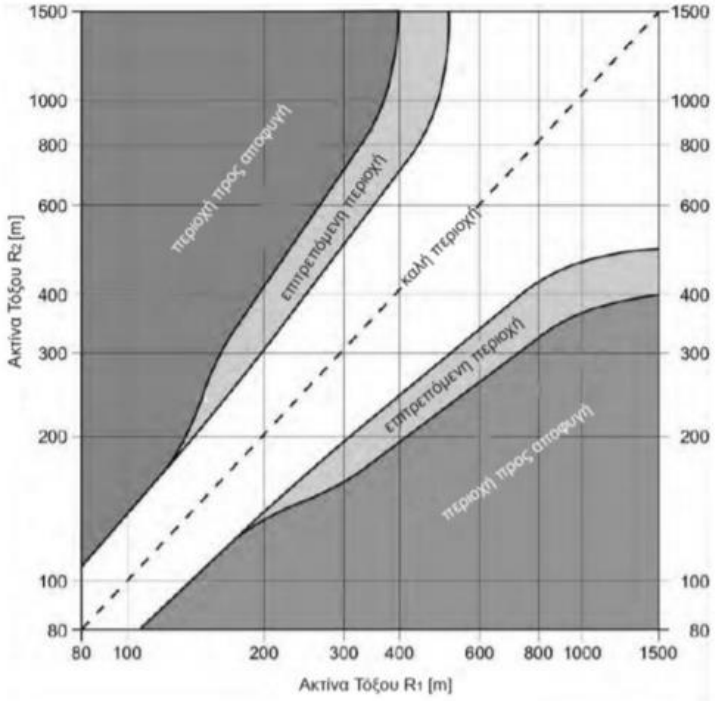
Πίνακας 2.2: Ελάχιστα μήκη κυκλικού τόξου κατά RAA, RAL και OMOE-X.

Για λόγους κυκλοφοριακής ασφάλειας, οι τιμές ακτινών μεταξύ διαδοχικών ομόρροπων ή αντίρροπων καμπυλών πρέπει να είναι μεταξύ τους εναρμονισμένες. Η απαίτηση αυτή οφείλεται στο γεγονός ότι δεν πρέπει να μεταβάλλεται απότομα η λειτουργική ταχύτητα V85 μεταξύ δύο διαδοχικών καμπυλών και συνεπώς υπό αυτήν την έννοια να ενισχύεται ένας ασφαλής τρόπος οδήγησης. Η επιτρεπόμενη σχέση μεταξύ των ακτινών διαδοχικών κυκλικών τόξων κατά OMOE-X, 2001 δίδεται στο Σχήμα 2.2.
Με βάση τις κατηγορίες οδού καθώς και τις λειτουργικές τους βαθμίδες επιλέγεται η περιοχή που πρέπει να βρίσκεται η αλληλουχία των ακτινών.
Στις οδηγίες RAA, 2008 συνιστάται στις διαδοχικές ακτίνες κυκλικών τόξων να τηρείται η προϋπόθεση $R1/R2 \leq 1.5$, εφόσον $R1 \leq 1500$ m.

Η αντίστοιχη αλληλουχία ακτινών διαδοχικών καμπυλών σύμφωνα με τις Γερμανικές οδηγίες RAL, 2012 δίδεται στο Σχήμα 2.3, όπου οι υπόψη τιμές των ακτινών πρέπει επίσης να βρίσκονται στην "καλή" περιοχή.



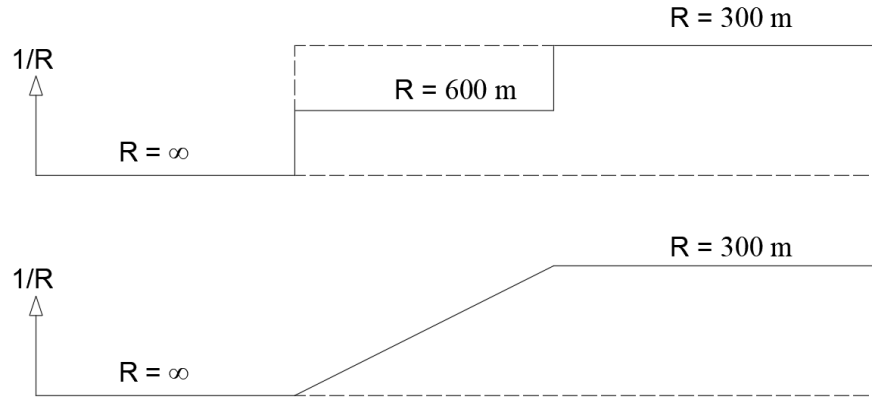

Σχήμα 2.2: Σχέση διαδοχικών ακτινών κυκλικών τόξων κατά OMOE-X, 2001.




Σχήμα 2.3: Σχέση διαδοχικών ακτινών κυκλικών τόξων κατά RAL, 2012.

### 2.3.3 Τόξο Συναρμογής

Από την πρώτη εποχή της οδοποιίας, η ευθεία και ο κύκλος χρησιμοποιήθηκαν σαν μέσο συνένωσης δύο ευθειών κατά την αλλαγή κατεύθυνσης. Όταν όμως τα οχήματα έγιναν ταχύτερα, η δυναμική της κυκλοφορίας έκανε αναγκαία μια βαθμιαία μετάβαση με σταθερή μεταβολή της καμπυλότητας από την ευθεία στον κύκλο. Το πρώτο είδος τόξου συναρμογής ήταν κυκλικό τόξο συναρμογής , διπλάσιας ακτίνας από το τελικό, δεν παρείχε όμως ομαλή μεταβολή της καμπυλότητας αλλά μόνο κλιμάκωσή της. Η απαίτηση ομοιόμορφης μεταβολής της καμπυλότητας ικανοποιείται με ακρίβεια μόνον από την κλωθοειδή (Σχήμα 2.4).




Σχήμα 2.4: Διάταξη κυκλικού τόξου συναρμογής με R=600m και διάταξη κλωθοειδούς μεταξύ ευθείας και τόξου κύκλου με R=300m

Το τόξο συναρμογής έχει τρεις στόχους. Κατά την μετάβαση από μία καμπυλότητα σε άλλη επιτρέπει συνεχή μεταβολή της φυγόκεντρης επιτάχυνσης που εμφανίζεται κατά την κίνηση σε καμπύλες. Συγχρόνως χρησιμοποιείται ως διαδρομή συναρμογής των δύο οδοστρωμάτων για την μεταβολή της επίκλισης. Τέλος εξασφαλίζει με την προοδευτική μεταβολή της καμπυλότητας μια ομαλή χάραξη.

Η κλωθοειδής καμπύλη σε πολλά συγγράμματα αναφέρεται ως σπειροειδής καμπύλη (spiral curve). Σε κάθε περίπτωση, η κλωθοειδής είναι μια καμπύλη με σταθερά μεταβαλλόμενη καμπυλότητά από 1/R=0 εώς 1/R=∞. Στην οδοποιία χρησιμοποιείται μόνο το πρώτο τμήμα της κλωθοειδούς.

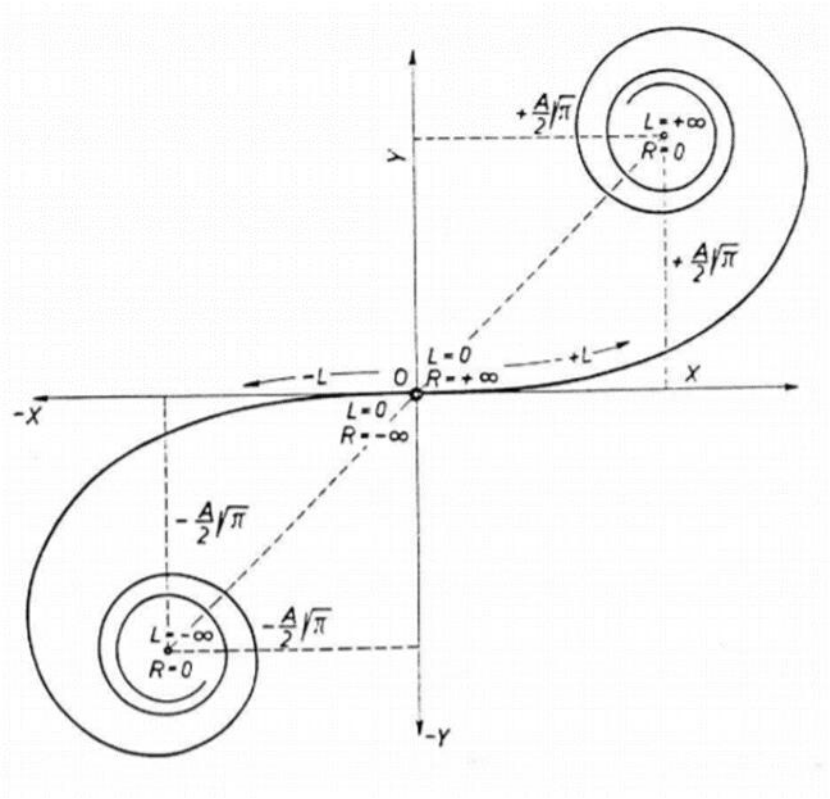
Η κλωθοειδής περιγράφεται από τον απλό νόμο:

της έκφραση έχει ως εξής:

$$A^2 = R * L \qquad (2.4)$$

όπου :
A : παράμετρος κλωθοειδούς (m)
R : ακτίνα καμπυλότητας στο πέρας του τμήματος της κλωθοειδούς (m)
L : μήκος του τόξου της κλωθοειδούς από την αρχή του (R=∞) μέχρι την ακτίνα R (m)

Η καμπυλότητα της κλωθοειδούς μεταβάλλεται γραμμικά με το μήκος της και ταυτόχρονα το γινόμενο μήκους – καμπυλότητας είναι σταθερό ($A^2$ ), έτσι ώστε η αύξηση του μήκους της καμπύλης να επιφέρει ανάλογη αύξηση στην καμπυλότητά της. Κατά συνέπεια, το μέγιστο θεωρητικό μήκος της καμπύλης προκύπτει όταν η ακτίνα καμπυλότητας τείνει να μηδενιστεί. Το πλήρες ανάπτυγμα της κλωθοειδούς καμπύλης φαίνεται στο σχήμα 2.5, όπου είναι προφανές ότι στην οδοποιία χρησιμοποιείται τμήμα της.




Σχήμα 2.5: Πλήρες ανάπτυγμα της κλωθοειδούς καμπύλης.

Η κλωθοειδής καμπύλη, όντας μαθηματική καμπύλη, επιλύεται στη βάση αναλυτικών σχέσεων. Στο σχήμα 2.6 φαίνεται σε καρτεσιανό σύστημα αξόνων στοιχειώδες τμήμα dL της καμπύλης σε σημείο Σ(x,y) με σημειακή ακτίνα R. Θεωρώντας ως dτ τη γωνιακή μεταβολή του στοιχειώδους τμήματος dL, ισχύει:

$$dL = R\, d\tau \qquad (2.5)$$

Λαμβάνοντας υπόψη την Εξίσωση (2.4) και ολοκληρώνοντας τελικά προκύπτει:

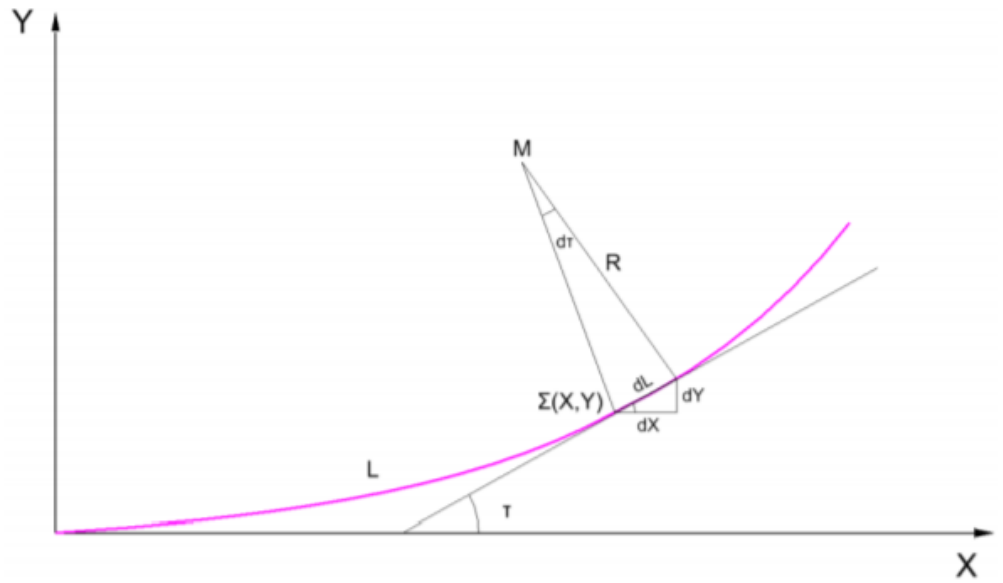
$$\frac{L^2}{2} = A^2 * \tau + C \qquad (2.6)$$

όπου :
C : η σταθερά της ολοκλήρωσης η οποία είναι 0 καθώς για L=0, τ=0

Άρα τελικά προκύπτει:

$$\tau = \frac{L}{2R} = \frac{A^2}{2R^2} = \frac{L^2}{2A^2} \qquad (2.7)$$

όπου :
τ : γωνία αλλαγής διεύθυνσης κλωθοειδούς (rad)
A : παράμετρος κλωθοειδούς (m)
R : ακτίνα καμπυλότητας στο πέρας του τμήματος της κλωθοειδούς (m)
L : μήκος του τόξου της κλωθοειδούς από την αρχή του (R=∞) μέχρι την ακτίνα R (m)




Σχήμα 2.6: Τμήμα κλωθοειδούς καμπύλης και στοιχειώδες τμήμα dL αυτής στο σημείο Σ(x,y).

Σύμφωνα με το Σχήμα 2.6, ο υπολογισμός της στοιχειώδους τετμημένης και τεταγμένης dx και dy αντίστοιχα στο σημείο Σ(x,y), λαμβάνοντας υπόψη την Εξίσωση (2.7), έχει ως εξής:

$$dx = dL * cos\tau = dL * cos\left(\frac{L^2}{2A^2}\right) \qquad (2.8)$$

$$dy = dL * sin\tau = dL * sin\left(\frac{L^2}{2A^2}\right) \qquad (2.9)$$

Η ολοκλήρωση της εξίσωσης (2.8) και εξίσωσης (2.9), λαμβάνοντας υπόψη την ανάπτυξη του ημιτόνου και συνημιτόνου σε σειρές Taylor με βάση τις γνωστές σχέσεις:

$$cosa = 1 - \frac{a^2}{2!} + \frac{a^4}{4!} - \frac{a^6}{6!} + \cdots \qquad (2.10)$$

$$sina = a - \frac{a^3}{3!} + \frac{a^5}{5!} - \frac{a^7}{7!} + \cdots \qquad (2.11)$$

τελικά δίνει:

$$x = L - \frac{L^5}{40A^4} + \frac{L^9}{3456A^8} - \cdots \quad (2.12)$$

$$y = \frac{L^3}{6A^2} + \frac{L^7}{336A^6} + \frac{L^{11}}{42240A^{10}} - \cdots \quad (2.13)$$

Στο Σχήμα 2.7 φαίνεται και πάλι τμήμα κλωθοειδούς καμπύλης, απ' όπου προκύπτουν σχέσεις υπολογισμού κάποιων βοηθητικών μεγεθών. Συγκεκριμένα είναι:

$$T_L = x - \frac{y}{tan\tau} \quad (2.14)$$

$$T_K = \frac{y}{sin\tau} \quad (2.15)$$

$$x_m = x - R * sin\tau \quad (2.16)$$

$$y_m = y + R * cos\tau \quad (2.17)$$

$$\Delta R = y_m - R \quad (2.18)$$

όπου :

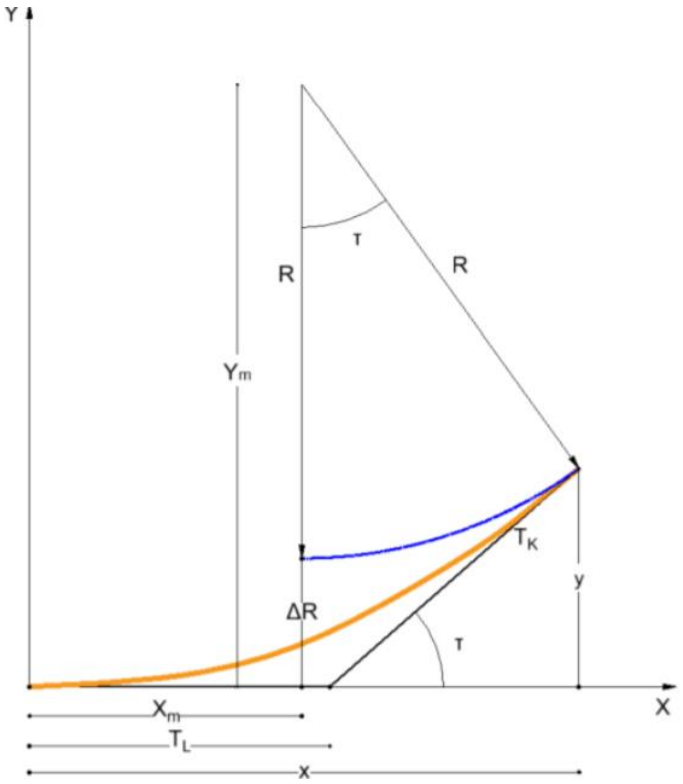
$T_L$ : κύρια εφαπτόμενη κλωθοειδούς (m)

$T_K$ : δευτερεύουσα εφαπτόμενη κλωθοειδούς (m)

$X_m$ : τετμημένη του κέντρου του κύκλου καμπυλότητας ακτίνας R όπου περατώνεται η καμπύλη

$Y_m$ : τεταγμένη του κέντρου του κύκλου καμπυλότητας ακτίνας R όπου περατώνεται η καμπύλη

ΔR : εκτροπή κλωθοειδούς (ελάχιστη απόσταση μεταξύ της καμπυλότητας ακτίνας R όπου περατώνεται η καμπύλη και του άξονα των τετμημένων)




Σχήμα 2.7. Βοηθητικά μεγέθη κλωθοειδούς καμπύλης.

Με βάση εμπειρικά δεδομένα, προκειμένου η κλωθοειδής καμπύλη να γίνει αντιληπτή από τον οδηγό, πρέπει η γωνία αλλαγής της διεύθυνσής της να είναι ίση τουλάχιστον με 3.50 g. Γεγονός το οποίο σημαίνει ότι:

$$A_{min} = \frac{R}{3} \qquad (2.19)$$

Η ελάχιστη τιμή της παραμέτρου Α, υπόκειται και σε έναν επιπλέον περιορισμό, ο οποίος αφορά στη χρήση της κλωθοειδούς ως μήκος συναρμογής για τη μεταβολή της επίκλισης μεταξύ ευθυγραμμίας και κυκλικού τόξου ή μεταξύ διαδοχικών κυκλικών τόξων. Συγκεκριμένα πρέπει το μήκος της κλωθοειδούς να είναι επαρκές ώστε η μεταβολή της επικλίσης να γίνεται ομαλά μέσα στο τόξο συναρμογής, με την έννοια να μην υπάρχει υπέρβαση στη μέγιστη επιτρεπόμενη τιμή της πρόσθετης κατά μήκος κλίσης των οριογραμμών.
Για λόγους ασφάλειας της κυκλοφορίας, ως μέγιστη τιμή της παραμέτρου της κλωθοειδούς λαμβάνεται:

$$A_{max} = R \qquad (2.20)$$

Η τιμή αυτή αντιστοιχεί σε γωνία αλλαγής της διεύθυνσής στο τέλος του τόξου συναρμογής ίσο με 31.8g.

## 2.4 ΤΥΠΟΙ ΟΡΙΖΟΝΤΙΟΓΡΑΦΙΚΩΝ ΚΑΜΠΥΛΩΝ ΜΕ ΕΦΑΡΜΟΓΗ ΚΛΩΘΟΕΙΔΩΝ

Με βάση την εφαρμογή διάφορων μορφών τόξων συναρμογής προκύπτουν αντίστοιχοι τύποι οριζοντιγραφικών καμπυλών. Στα επόμενα εδάφια περιγράφονται συνοπτικά οι πιο χαρακτηριστικοί τύποι οριζοντιγραφικών καμπυλών.

### 2.4.1 Τυπική Καμπύλη

Η τυπική καμπύλη συνίσταται από αλληλουχία ευθυγραμμίας – κλωθοειδούς (εισόδου) – κυκλικού τόξου – κλοθωειδούς (εξόδου) – ευθυγραμμίας και αποτελεί την πιο συνηθισμένη μορφή καμπύλης στην οδοποιία. Οι κλωθοειδείς εισόδου και εξόδου μπορούν να εφαρμόζονται με τις ίδιες ή διαφορετικές παραμέτρους οπότε η καμπύλη είναι συμμετρική ή ασύμμετρη αντίστοιχα.
Στο Σχήμα 2.8 φαίνεται παράδειγμα τυπικής καμπύλης με γωνία στροφής γ αποτελούμενης από κυκλικό τόξο ακτίνας R και κλωθοειδείς με ασύμμετρες παραμέτρους κλωθοειδών (A1 ≠ A2), τα κυριότερα γεωμετρικά χαρακτηριστικά της οποίας υπολογίζονται με βάση τις εξισώσεις (2.4, 2.7 και 2.12 – 2.18). Στις περιπτώσεις εφαρμογής ασύμμετρων κλωθοειδών, γενικά είναι επιθυμητό A1 ≤ 1.50 A2.
Ο υπολογισμός των κύριων εφαπτόμενων της συνολικής καμπύλης (Τ1 και Τ2), οι οποίες αποτελούν την πολυγωνική της καμπύλης, προκύπτει μέσω του σχήματος 2.9 καθώς και των εξής σχέσεων:

$$\gamma^{grad} = {\tau_1}^{grad} + {\tau_2}^{grad} + \ a^{grad} \quad (2.21)$$

$$t_1 = (R + \Delta R_1) tan\left(\frac{\gamma}{2}\right) \quad (2.22)$$

$$t_2 = (R + \Delta R_2) tan\left(\frac{\gamma}{2}\right) \quad (2.23)$$

Τελικά ισχύει:

$$T_1 = x_{m1} + t_1 - d \quad (2.24)$$

$$T_2 = x_{m2} + t_2 + d \quad (2.25)$$

Το μέγεθος d προκύπτει από το βοηθητικό σχήμα (μεγέθυνση) στην περιοχή του σημείου K της πολυγωνικής (Σχήμα 2.9), όπου λαμβάνοντας υπόψη ότι:

$$2\varphi \ + \ \gamma \ = \ 200grad \quad (2.26)$$

προκύπτει:

$$d = \frac{\Delta R_1 - \Delta R_2}{sin\gamma} \quad (2.27)$$

Είναι προφανές ότι όταν οι κλωθοειδείς έχουν ίδιες παραμέτρους (A1 = A2), τότε d=0, τα σημεία K, C και B συμπίπτουν και η καμπύλη είναι συμμετρική ως προς τον κάθετο άξονα που αποτελεί και τη διχοτόμο της γωνίας 200-γ.

Τέλος, το μήκος του κυκλικού τόξου b προκύπτει με βάση την Εξίσωση (2.21) ως εξής:

$$b = R * (\gamma^{grad} - {\tau_1}^{grad} - {\tau_2}^{grad}) \frac{\pi}{200} \quad (2.28)$$

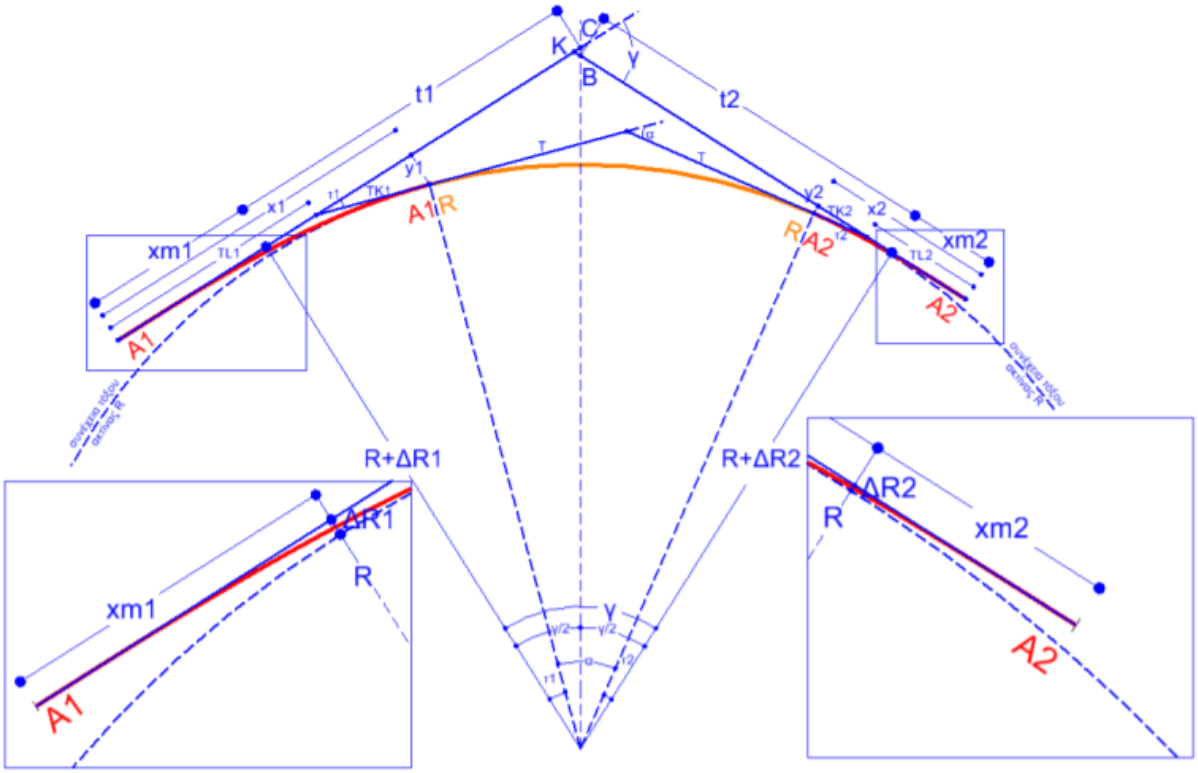
Στην περίπτωση εφαρμογής της τυπικής καμπύλης σε οδούς ενιαία επιφάνεια κυκλοφορίας, η σχέση που δίνει την ελικτότητα, αναγκαία για τον προσδιορισμό της λειτουργικής ταχύτητας V85 είναι:

$$K_E = \frac{\gamma}{L} * 1000 \ = \frac{{\tau_1}^{grad} + {\tau_2}^{grad} + a^{grad}}{L_1 + b + L_2} * 1000 = \frac{\left(\frac{{A_1}^2}{2R^2} + \frac{b}{R} + \frac{{A_2}^2}{2R^2}\right) * \frac{200}{\pi}}{\frac{{A_1}^2}{R} + b + \frac{{A_2}^2}{R}} * 1000 \quad (2.29)$$

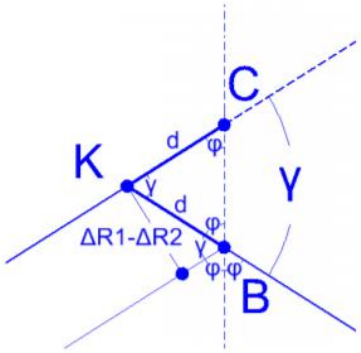
όπου:

KE: ελικτότητα καμπύλης (grad/km)

γ: γωνιακή μεταβολή συνολικής καμπύλης (grad)




Σχήμα 2.8. Τυπική καμπύλη με ασύμμετρες παραμέτρους κλωθοειδών (A1 ≠ A2).




Σχήμα 2.9. Προσδιορισμός απόστασης d στην περιοχή του σημείου τομής της πολυγωνικής.

### 2.4.2 Καμπύλη Τύπου S

Η υποκατάσταση των τόξων συναρμογής μεταξύ δύο αντίρροπων κυκλικών τόξων από ευθεία είναι ιδιαίτερα ανεπαρκής μιας και δημιουργεί ασυνεχής μεταβολή της καμπυλότητας. Για αυτόν τον λόγο διαμορφώνονται συνήθως τόξα συναρμογής με δύο αντίρροπους κλάδους κλωθοειδούς χωρίς ενδιάμεση ευθυγραμμία. Καθένας από τους κλάδους της κλωθοειδούς υπόκειται στους περιορισμούς του απλού τόξου συναρμογής της τυπικής καμπύλης. Οι δύο κλάδοι πρέπει να έχουν περίπου την ίδια παράμετρο για λόγους αρμονίας της χάραξης και ομοιόμορφης πρόσθετης κλίσης των οριογραμμών. Κατά OMOE-X και RAL, όταν δεν μπορεί να είναι ίσες οι δύο παράμετροι A1 και A2, για A2 ≤ 200m (A2 ≤ 300m κατά RAA, 2008) πρέπει να ισχύει η σχέση:

$$A1 \leq 1.50 * A2 \quad (2.30)$$

όπου:
A1: η μεγαλύτερη παράμετρος (m)
A2: η μικρότερη παράμετρος (m)

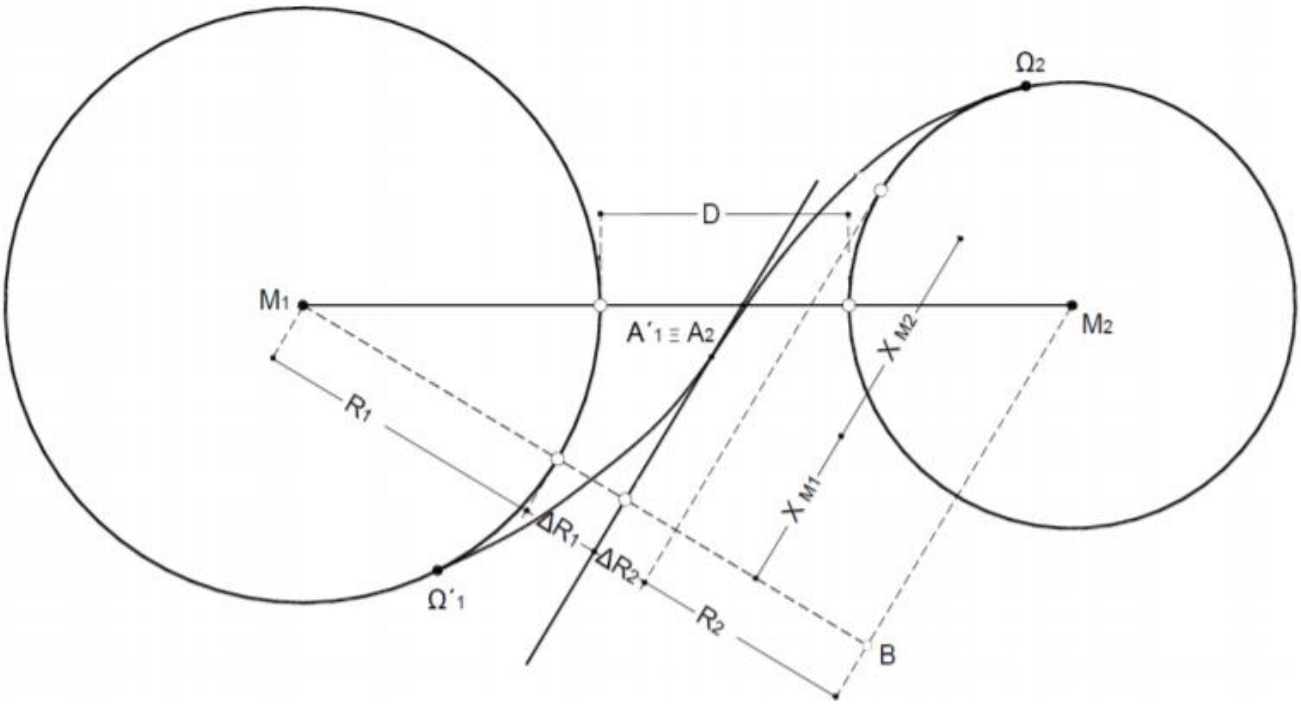
Η σχέση των ακτινών των δύο καμπυλών καθορίζεται από τα όρια του σχήματος 2.10. Η καμπύλη τύπου S συνίσταται από αλληλουχία ευθυγραμμίας – κλωθοειδούς (εισόδου) – κυκλικού τόξου – S-Καμπύλης– κυκλικού τόξου – κλωθοειδούς (εξόδου) – ευθυγραμμίας.

Επιπλέον, στις οδηγίες OMOE-X και RAL αναφέρεται ότι προκειμένου να είναι δυνατή η προσαρμογή σε τοπικά υποχρεωτικά σημεία, επιτρέπεται η διάταξη ευθυγραμμίας μικρού μήκους μεταξύ των δύο σημείων αρχής των κλωθοειδών. Προκειμένου όμως να διατηρηθεί η οπτική εντύπωση της S-καμπύλης ώστε να μην θεωρούνται ανεξάρτητες καμπύλες, πρέπει η τιμή του μήκους της ενδιάμεσης ευθυγραμμίας να μην υπερβαίνει την τιμή:

$$Lz \leq 0.08\ (A1 + A2) \quad (2.31)$$

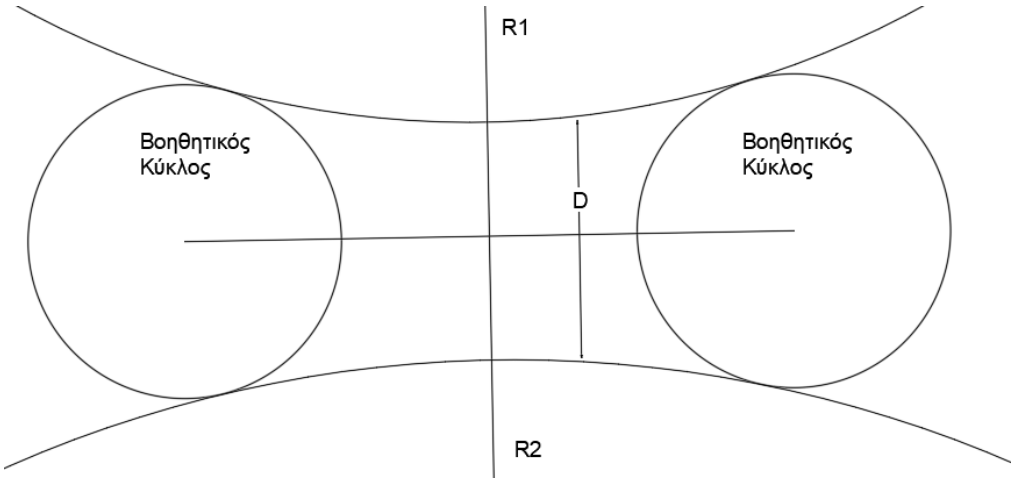
όπου:
A1, A2: παράμετροι κλωθοειδούς (m)
Lz: μήκος ενδιάμεσης ευθυγραμμίας (m)

Η σχετική θέση δύο κύκλων καθορίζεται μονοσήμαντα από την απόσταση των κέντρων τους $M_1M_2$, η οποία αποτελείται από τις δύο ακτίνες και την περιφερειακή απόσταση D. Δηλαδή:
$M_1M_2 = R_1 + D + R_2$ (2.32)




Σχήμα 2.10: Σιγμοειδής καμπύλη.

Τα κέντρα που χρησιμοποιούνται στην πράξη είναι συνήθως απρόσιτα και κατά την σχεδίαση και κατά την πασσάλωση. Στο παρελθόν, όπου ο σχεδιασμός των καμπυλών γινόταν χωρίς τη βοήθεια σχεδιαστικών λογισμικών (CAD), όταν τα κέντρα των κύκλων ήταν απρόσιτα, η διάκεντρος $M_1M_2$ σχεδιαζόταν με την βοήθεια βοηθητικών κύκλων. Ο βοηθητικός κύκλος σχεδιαζόταν σε ιδιαίτερο φύλλο χαρτιού, διαφανές, και μετακινούταν πάνω στο βασικό σχέδιο μέχρις ότου έρθει σε επαφή και με τα 2 κυκλικά τόξα. Η ίδια διαδικασία ακολουθούταν και από την άλλη πλευρά της διακέντρου. Η μεσοκάθετος της διακέντρου των δύο ίσων βοηθητικών κύκλων αποτελεί την ζητούμενη διάκεντρο, οπότε ορίζεται και η απόσταση D.




Σχήμα 2.11: Προσδιορισμός της περιφερειακής απόστασης D των τόξων μέσω βοηθητικών κύκλων.

Με γνωστά τώρα τα $R_1$, $R_2$, D αναζητείται η κοινή παράμετρος A.
Από το τρίγωνο $M_1M_2B$ ισχύει:

$$(R_1 + R_2 + D)^2 = (x_{m1} + x_{m2})^2 + (R_1 + \Delta R_1 + R_2 + \Delta R_2)^2 \qquad (2.33)$$

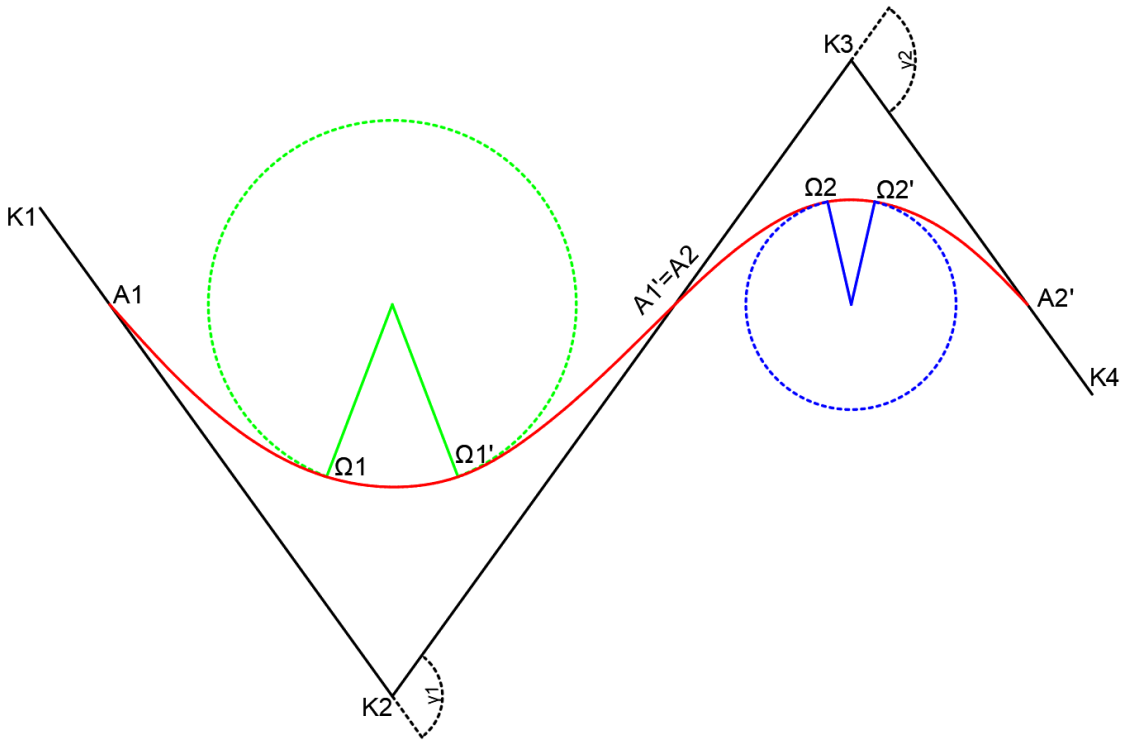
όπου (i=1,2):
$x_{mi}$, $y_{mi}$ : υπολογίζονται από την εξίσωση (2.16) και την εξίσωση (2.17)
$\Delta R_i$ : υπολογίζεται από την εξίσωση (2.18)

Η γωνία στροφής τ στην εξίσωση (2.16) και εξίσωση (2.17) υπολογίζεται από την εξίσωση (2.7), ενώ τα μεγέθη xi και yi στις παραπάνω σχέσεις υπολογίζονται ως εξής:

$$x_i = \frac{{A_i}^2}{R_i} - \frac{\left(\frac{{A_i}^2}{R_i}\right)^5}{40{A_i}^4} + \frac{\left(\frac{{A_i}^2}{R_i}\right)^9}{3456{A_i}^8} \qquad (2.34)$$

$$y_i = \frac{\left(\frac{{A_i}^2}{R_i}\right)^3}{6{A_i}^2} + \frac{\left(\frac{{A_i}^2}{R_i}\right)^7}{336{A_i}^6} + \frac{\left(\frac{{A_i}^2}{R_i}\right)^{11}}{42240{A_i}^{10}} \qquad (2.35)$$

Η επίλυση της Εξίσωσης (2.33) αν και ενός αγνώστου είναι αρκετά σύνθετη. Γενικά σε εξισώσεις αυτού του τύπου η λύση προκύπτει με διαδοχικές προσεγγίσεις ή παρόμοιες πρακτικές. Ένας από τους τρόπους επίλυσης αναφέρεται στο κεφάλαιο 3.1.1.




Σχήμα 2.12: Σιγμοειδής Καμπύλη.

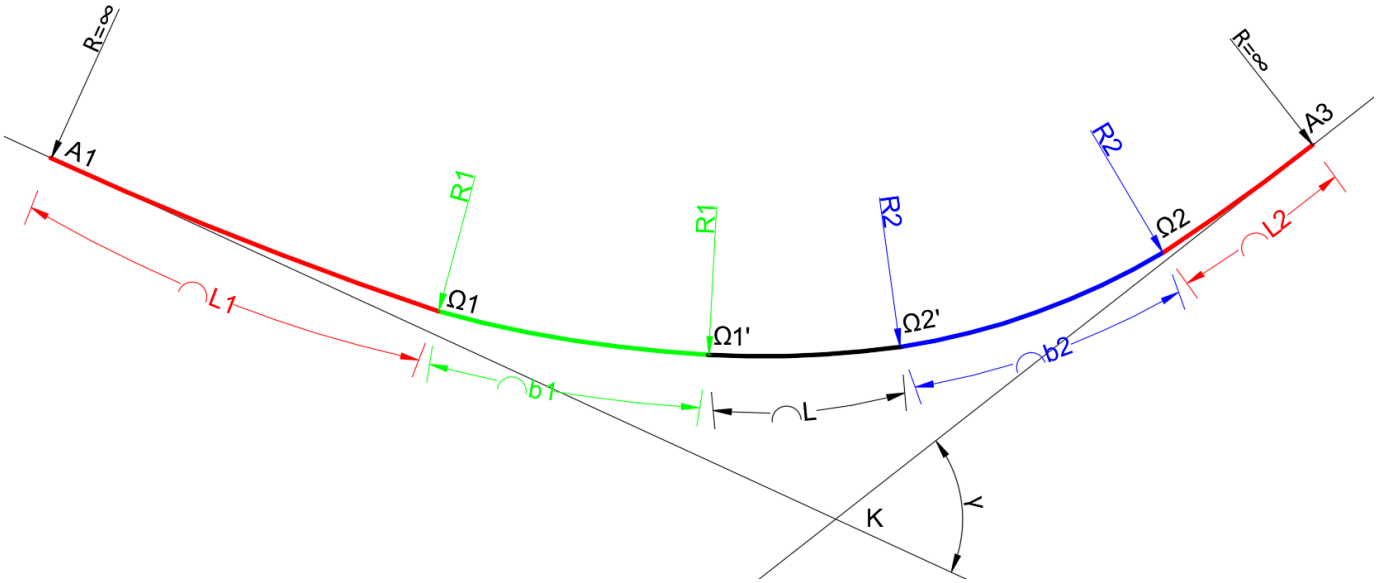
### 2.4.3 Ωοειδής Καμπύλη

Συχνά, τοπογραφικά εμπόδια, οικισμοί ή ειδικές χαράξεις (πχ. σε συνδετήριους κλάδους αυτοκινητόδρομων) επιβάλλουν τη μόρφωση καμπυλών με μεταβαλλόμενη καμπυλότητα. Η διάταξη όμως κυκλικών τόξων σε άμεση αλληλουχία δημιουργεί τόξο κανίστρου, δηλαδή ασυνεχή μεταβολή της καμπυλότητας, η οποία είναι ιδιαίτερα αισθητή όταν οι ακτίνες είναι μικρότερες. Η διαμόρφωση συνεχούς μεταβολής καμπυλότητας απαιτεί τη διάταξη τόξου συναρμογής και μάλιστα ωοειδούς.

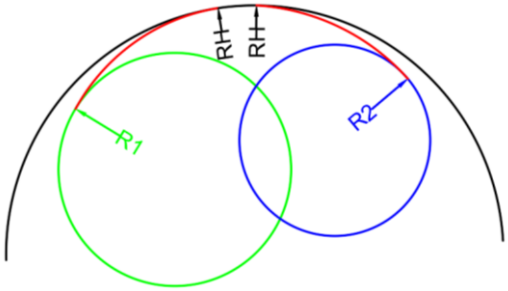
Η ωοειδής καμπύλη είναι τμήμα κλωθοειδούς και χρησιμοποιείται ως τόξο συναρμογής μεταξύ δύο ομόρροπων κυκλικών τόξων με διαφορετικές ακτίνες. Προκειμένου μια καμπύλη να θεωρείται ωοειδής πρέπει:

- Ο ένας κύκλος να βρίσκεται στο εσωτερικό του άλλου
- Οι δύο κύκλοι να μην τέμνονται
- Οι δύο κύκλοι να μην είναι ομόκεντροι

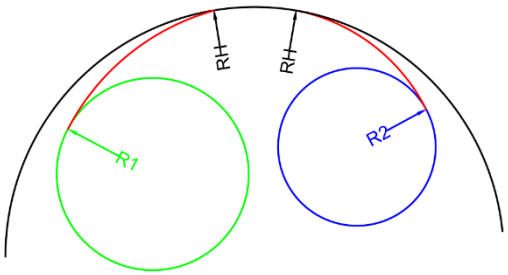
Στην ωοειδή καμπύλη η ακολουθία των στοιχείων χάραξης έχει ως εξής: ευθυγραμμία – κλωθοειδής - κυκλικό τόξο – ωοειδής – κυκλικό τόξο – κλωθοειδής – ευθυγραμμία. Η μεγαλύτερη ακτίνα σημειώνεται με R1 και η μικρότερη με R2 (Σχήμα 2.12). Αν οι δύο κύκλοι τέμνονται ή ο ένας κύκλος δεν βρίσκεται στο εσωτερικό του άλλου, τότε πρέπει να χρησιμοποιηθεί βοηθητικό κυκλικό τόξο και προκύπτει η εξής σειρά στοιχείων χάραξης (διπλή ωοειδής): ευθυγραμμία – κλωθοειδής - κυκλικό τόξο – ωοειδής – κυκλικό τόξο (βοηθητικό) – ωοειδής - κυκλικό τόξο - κλωθοειδής – ευθυγραμμία. Η ακτίνα του πρόσθετου κύκλου συμβολίζεται με $R_H$. (Σχήματα 2.13-2.16)



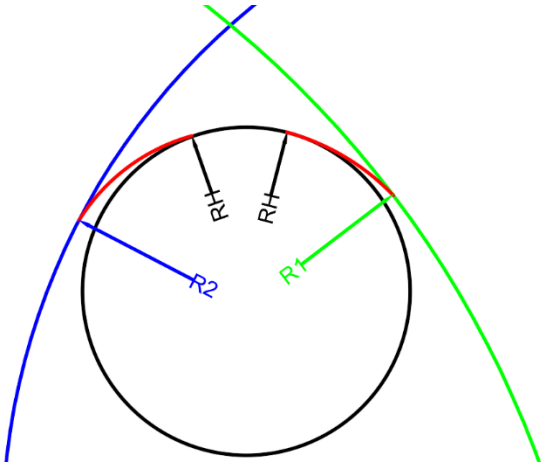

Σχήμα 2.13: Ακολουθία στοιχείων χάραξης στην ωοειδή καμπύλη.




Σχήμα 2.14: Τα κυκλικά τόξα τέμνονται και η διπλή ωοειδής σχηματίζεται με τη βοήθεια πρόσθετου κύκλου ο οποίος δεν εφάπτεται με κανένα από τα κυκλικά τόξα.




Σχήμα 2.15: Οι R1 R2 δεν έχουν κοινό σημείο ούτε ο ένας είναι εσωτερικός του άλλου. Η διπλή ωοειδής σχηματίζεται με τη βοήθεια πρόσθετου εξωτερικού κύκλου.




Σχήμα 2.16: Οι R1, R2 τέμνονται. Η διπλή ωοειδής σχηματίζεται με τη βοήθεια εσωτερικού πρόσθετου κύκλου.

Γενικά η χρήση ωοειδούς συνιστάται σε περιπτώσεις όπου απαιτείται σημαντική αλλαγή στην καμπυλότητα της οδού σε περιορισμένο χώρο (πχ. ορισμένοι συνδετήριοι κλάδοι ανισόπεδων κόμβων). Σε αυτές τις περιπτώσεις προτιμάται η εφαρμογή ωοειδούς καμπύλης έναντι αλληλουχίας τόξων κανίστρων (ομόρροπα διαδοχικά κυκλικά τόξα), καθώς η φυγόκεντρη επιτάχυνση μεταβάλλεται γραμμικά (κατά μήκος της ωοειδούς καμπύλης) και όχι σημειακά και άρα η μεταβολή στη λειτουργική ταχύτητα των οχημάτων είναι πιο ομοιόμορφη.

Στα κυκλικά τόξα ισχύει η απαίτηση αλληλουχίας των ακτινών. Για να είναι οπτικά εμφανής η ωοειδής καμπύλη, θα πρέπει η γωνία στροφής κατά μήκος της να είναι $\tau \geq 3.5\,\text{grad}$. Το ελάχιστο μήκος ωοειδούς καμπύλης, καθώς και οι οριακές τιμές των εκατέρωθεν της ωοειδούς κυκλικών τόξων κατά OMOE-X, 2001 δίδονται στον Πίνακα 2.3.

| Τιμές Μικρότερης Ακτίνας R2 (m) | Επιτρεπόμενη Σχέση R1/R2 | Ελάχιστο Μήκος Ωοειδούς $L_v$ (m) |
|---|---|---|
| R2 ≤ 100 | 1.5 | $\frac{V_e\left(\text{km}/\text{h}\right)}{3.6}$ |
| 100 < R2 ≤ 500 | 2.0 | |
| 500 < R2 | Χωρίς περιορισμό | |

Πίνακας 2.3: Οριακές Τιμές Ωοειδούς Καμπύλης

Στο Σχήμα 2.17 φαίνεται η παράδειγμα ωοειδούς καμπύλης με παράμετρο Α μεταξύ των εκατέρωθεν αυτής κυκλικών τόξων με ακτίνες R1 και R2, όπου προκύπτουν οι εξής σχέσεις:

$$\tau = \tau_2 - \tau_1 = \frac{A^2}{2R_2^2} - \frac{A^2}{2R_1^2} \quad (2.36)$$

$$L = L_2 - L_1 = \frac{A^2}{R_2} - \frac{A^2}{R_1} \quad (2.37)$$

$$T_L = \frac{(T_{L2} - T_{L1})}{sin\tau} sin\tau_2 - T_{K1} \quad (2.38)$$

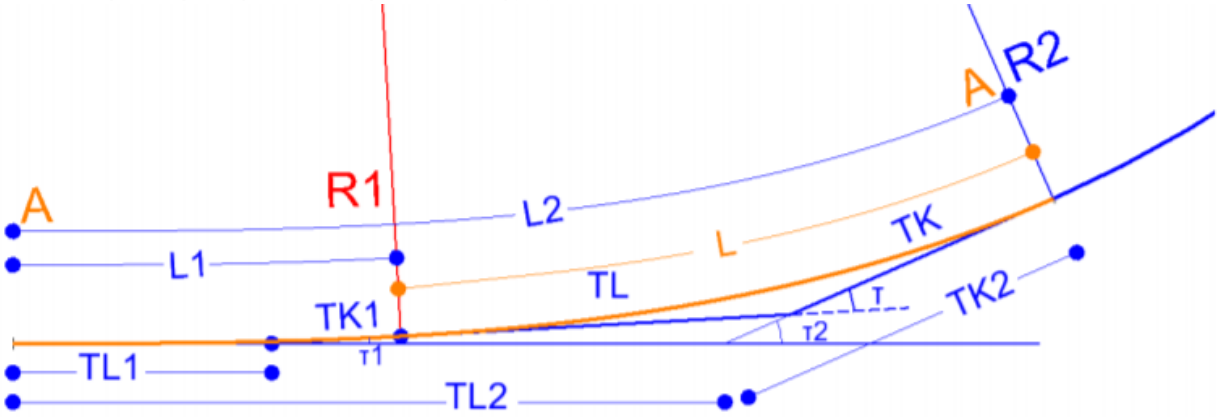
$$T_K = T_{K2} - \frac{(T_{L2} - T_{L1})}{sin\tau} sin\tau_1 \quad (2.39)$$

όπου:
τ: γωνία στροφής ωοειδούς καμπύλης (rad)
L: μήκος ωοειδούς καμπύλης (m)
TL: κύρια εφαπτόμενη ωοειδούς καμπύλης (m)
TK: δευτερεύουσα εφαπτόμενη ωοειδούς καμπύλης (m)

Στην περίπτωση εφαρμογής ωοειδούς καμπύλης σε οδούς ενιαία επιφάνεια κυκλοφορίας, η σχέση που δίνει την ελικτότητα, προκειμένου να προσδιοριστεί η λειτουργική ταχύτητα V85 είναι:

$$K_E = \frac{\gamma}{L} 1000 = \frac{\tau_1 + \alpha_1 + \tau + \alpha_2 + \tau_2}{L_1 + b_1 + L + b_2 + L_2} 100 = \frac{\left(\frac{A_1^2}{2R_1^2} + \frac{b_1}{R_1} + \frac{A^2}{2R_2^2} - \frac{A^2}{2R_1^2} + \frac{b_2}{R_2} + \frac{A_2^2}{2R_2^2}\right)\frac{200}{\pi}}{\frac{A_1^2}{R_1} + b_1 + \frac{A^2}{R_2} - \frac{A^2}{R_1} + b_2 + \frac{A_2^2}{R_2}} 1000 \quad (2.40)$$

όπου:
ΚΕ: ελικτότητα καμπύλης (grad/km)
γ: γωνιακή μεταβολή συνολικής καμπύλης (grad)




Σχήμα 2.17: Παράμετροι ωοειδούς καμπύλης.

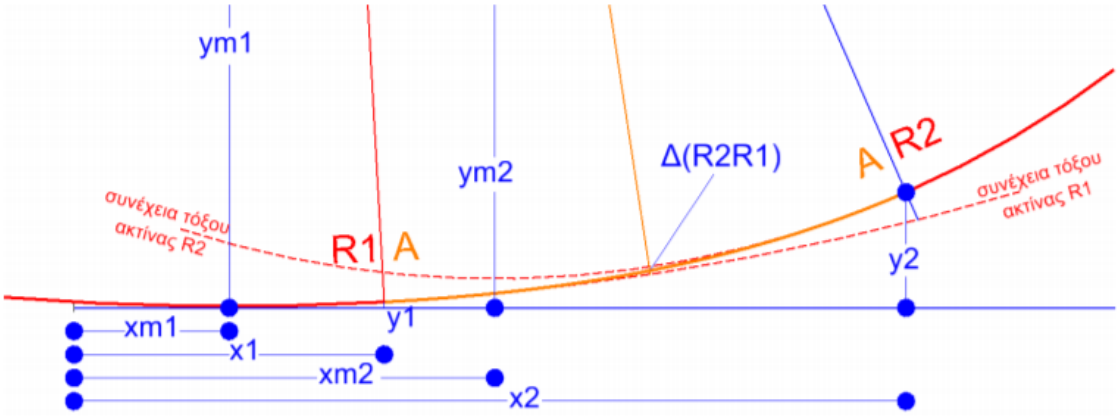
Στο σχήμα 2.18 φαίνεται και πάλι η ίδια καμπύλη όπου δίδονται σχέσεις υπολογισμού του ελάχιστου μήκους μεταξύ των εκατέρωθεν της ωοειδούς κυκλικών τόξων με ακτίνες R1 και R2 αντίστοιχα. Ισχύει:

$$K_1K_2 = \sqrt{(x_{m1} - x_{m2})^2 + (y_{m1} - y_{m2})^2} \qquad (2.41)$$

$$\Delta(R_1R_2) = R_1 - K_1K_2 - R_2 \qquad (2.42)$$

όπου:
K1K2: απόσταση των εκατέρωθεν της ωοειδούς κέντρων κύκλων με ακτίνες R1 και R2 αντίστοιχα (m)
Δ(R2R1): ελάχιστο μήκους μεταξύ των εκατέρωθεν της ωοειδούς κυκλικών τόξων

Η επίλυση της εξίσωσης (2.42) και πάλι έχει μονοσήμαντη λύση και προκύπτει με διαδοχικές προσεγγίσεις ή παρόμοιες πρακτικές. Ο ζητούμενος άγνωστος τώρα είναι η παράμετρος Α. Μεγέθυνση στην περιοχή της απόστασης Δ(R2R1) δίδεται στο σχήμα 2.17.

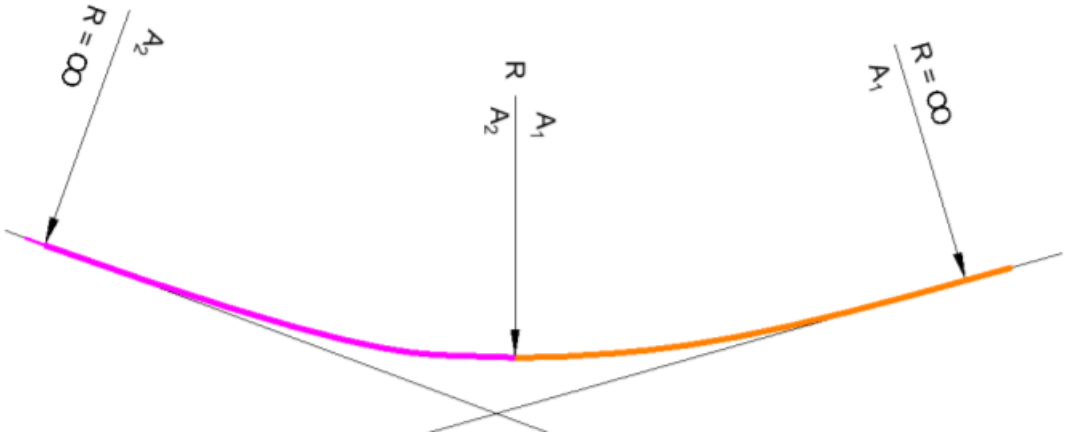



Σχήμα 2.18: Ελάχιστο απόσταση Δ(R2R1) μεταξύ των εκατέρωθεν της ωοειδούς κυκλικών τόξων.

### 2.4.4 Κλωθοειδής Κορυφής

Αποτελείται από δύο κλωθοειδείς χωρίς ενδιάμεσο κύκλο. Σχηματίζεται δηλαδή από δύο κλωθοειδείς ομόρροπες με κοινό άκρο (και όχι αρχή). Εφαρμόζεται μόνο όταν χρειάζεται ανοικτή καμπύλη, αλλιώς αποφεύγεται, επειδή η άμεση διαδοχή καμπύλης αυξανόμενης καμπυλότητας με καμπύλη μειούμενης καμπυλότητας προκαλεί δυσκολίες στην κίνηση των οχημάτων.

Οι παράμετροι των κλωθοειδών πρέπει κατά το δυνατόν να είναι ίσες (A1 = A2). Στον Πίνακα 5.8 δίδονται οι ελάχιστες τιμές της κοινής ακτίνας σύμφωνα με τις οδηγίες ΟΜΟΕ-Χ, 2001. Στις οδηγίες RAL, 2012 απαραίτητη προϋπόθεση για την εφαρμογή κλωθοειδούς κορυφής είναι η τιμή της γωνίας στροφής της καμπύλης η οποία πρέπει να είναι μικρότερη των 10 g. Το ελάχιστο μήκος της καμπύλης σε αυτές τις περιπτώσεις φαίνεται στον Πίνακα 5.9.




Σχήμα 2.19: Κλωθοειδής κορυφής.

| ΟΜΟΕ-Χ, 2001 | |
|---|---|
| Ομάδα Οδού | $R_{s\ min}$ (m) |
| A | 500 |
| B | 260 |

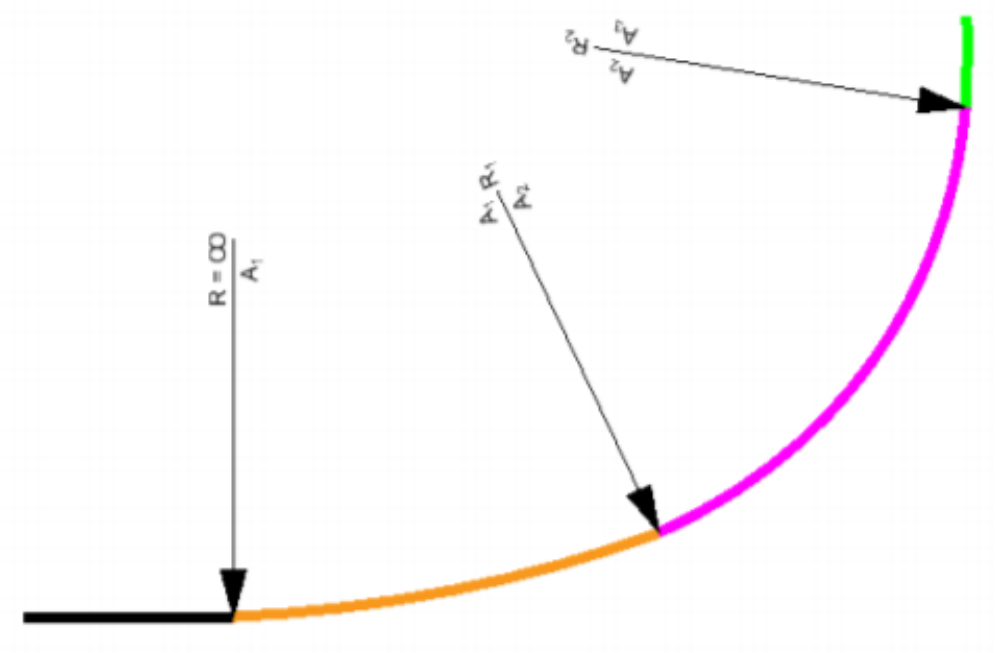
Πίνακας 2.4: Ελάχιστες τιμές της κοινής ακτίνας σε κλωθοειδή κορυφής κατά ΟΜΟΕ-Χ, 2001.

| | RAL, 2012 | |
|---|---|---|
| V (km/h) | Κλάση Οδού | $L_{min}$ (m) |
| 110 | EKL 1 | 200 |
| 100 | EKL 2 | 200 |
| 90 | EKL 3 | 150 |
| 70 | EKL 4 | 100 |

Πίνακας 2.5: Ελάχιστα μήκη κλωθοειδούς κορυφής κατά RAL, 2012.

### 2.4.5 Κλωθοειδής Κανίστρου

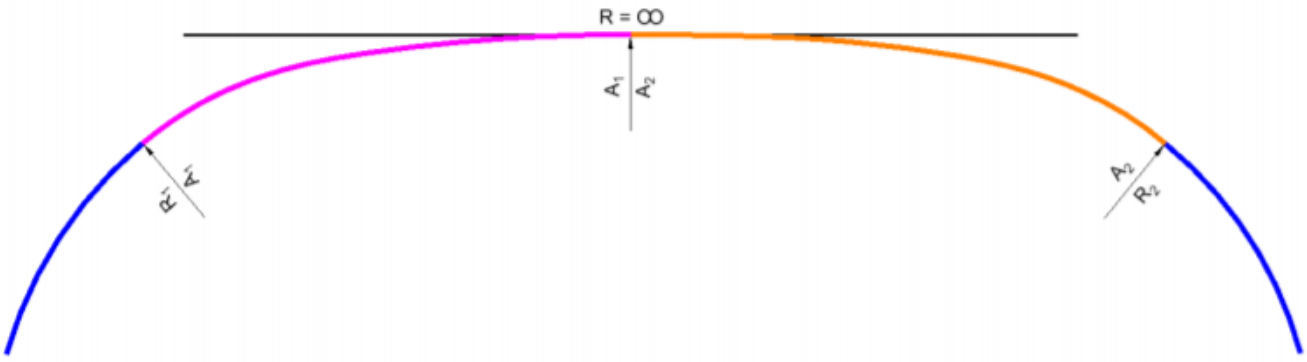
Αποτελείται από αλληλουχία ομόρροπων τμημάτων κλωθοειδών με διαφορετικές παραμέτρους. Στα σημεία επαφής των επιμέρους κλωθοειδών, οι ακτίνες είναι ίσες και οι εφαπτόμενες κοινές. Συνήθως επιβάλλεται από ιδιαίτερα δεσμευτικές συνθήκες. Χρησιμοποιούνται το πολύ τριμελείς κλωθοειδείς κανίστρου. Κατά RAAL-L για λόγους οδικής ασφάλειας πρέπει να αποφεύγονται, εφόσον είναι δυνατό.




Σχήμα 2.20: Κλωθοειδής Κανίστρου.

### 2.4.6 Κλωθοειδής Τύπου C

Αποτελείται από αλληλουχία δύο ομόρροπων κλωθοειδών σε επαφή κατά τα αρχικά σημεία τους (Σχήμα 5.21). Επειδή η περιοχή μικρής καμπυλότητας με $\tau < 3.5\,\text{grad}$ διαφέρει ελάχιστα από την ευθεία, προκαλείται η οπτική εντύπωση ενδιάμεσης ευθυγραμμίας. Για αυτό το λόγο πρέπει να αποφεύγεται η χρήση της.




Σχήμα 2.21: Κλωθοειδής τύπου C.

# *ΕΦΑΡΜΟΓΗ ΜΕΘΟΔΟΛΟΓΙΑΣ-ΑΠΟΤΕΛΕΣΜΑΤΑ*

## 3.1 ΕΙΣΑΓΩΓΗ

Όπως αναφέρθηκε στο Κεφάλαιο 2, δύο από τις πιο διαδεδομένες καμπύλες είναι η Σιγμοειδής και η Ωοειδής. Η Σιγμοειδής αποτελεί μια καμπύλη που χρησιμοποιείται κατά κόρων στην οριζοντιογραφική χάραξη ενώ η Ωοειδής χρησιμοποιείται σε ειδικές περιπτώσεις, όπως για τον σχεδιασμό κλάδων ανισόπεδων κόμβων ή γενικά όπου απαιτείται ομαλή αύξηση της καμπυλότητας χωρίς ενδιάμεσες ευθυγραμμίες.

Ένα συχνό φαινόμενο που καλούνται να αντιμετωπίσουν οι μελετητές είναι η ομαλή διαχείριση της ταχύτητας σε συνθήκες περιορισμένου χώρου. Για παράδειγμα έστω υπάρχουσα χάραξη οδού που αποτελείται από δύο δεσμευτικά τόξα. Μια συχνή επίλυση αυτού του προβλήματος, η οποία χρησιμοποιείται ακόμα και στις μέρες μας από πολλούς μηχανικούς, είναι η δημιουργία κανιστροειδούς τόξου αποτελούμενο από δύο διαφορετικές καμπυλότητες με κοινό σημείο (εφαπτόμενη) επαφής. Αυτή όμως δεν είναι μια ασφαλής προσέγγιση καθώς στο κοινό σημείο, η καμπυλότητα και η πλευρική επιτάχυνση μεταβάλλονται στιγμιαία, δημιουργώντας πλευρική αστάθεια και εν συνεχεία δυσκολία στην διατήρηση του οχήματος στη θεωρητική του τροχιά. Αυτό έχει ως αποτέλεσμα μία απότομη μεταβολή στην ταχύτητα του οχήματος, η οποία με την σειρά της αναγκάζει τον οδηγό σε απότομους ελιγμούς που πολλές φορές είναι μη ασφαλείς. Η σωστή αντιμετώπιση αυτού του προβλήματος είναι μόνο με την χρήση τόξου συναρμογής ωοειδούς καμπύλης ώστε αυτή η μεταβολή στην πλευρική επιτάχυνση να γίνεται γραμμικά.

Ένα άλλο συχνό παράδειγμα είναι η ανάγκη ένωσης δύο σημείων αντίρροπων τόξων. Για την ομαλή ένωση των δύο αυτών σημείων είναι απαραίτητη η χρήση τόξου συναρμογής σιγμοειδούς καμπύλης με σκοπό και πάλι την γραμμική μεταβολή της πλευρικής επιτάχυνσης και της καμπυλότητας καθώς και την ελαχιστοποίηση της ενδιάμεσης ευθυγραμμίας.

Στην αγορά υπάρχουν διάφορα λογισμικά σχεδιασμού καμπυλών όπως για παράδειγμα το Civil 3D από την Autodesk ή το Inroads από την Bentley. Παρόλα αυτά, τα λογισμικά αυτά είναι προσαρμοσμένα στην «μαθηματική» λύση, μιας και δεν είναι δομημένα από μηχανικούς. Έτσι τα δεδομένα εισόδου που χρειάζεται να εισάγει ο χρήστης είναι προσαρμοσμένα στη χάραξη της καμπύλης ανεξάρτητα από περιορισμούς που επιβάλλονται από τις συνθήκες σχεδιασμού (πχ. διαθέσιμος χώρος, ανάγλυφο εδάφους, κλπ.). Με βάση τα παραπάνω, κρίθηκε αναγκαίο να αναπτυχθούν αλγόριθμοι για τον σχεδιασμό δύο εκ των πιο συνηθισμένων καμπυλών, της Σιγμοειδούς και Ωοειδούς. Βασικός σκοπός είναι η δημιουργία εργαλείων που έχουν ως δεδομένα εισόδου στοχευμένες μεταβλητές, τις οποίες καλείται να επιλέξει ο μηχανικός, για τον σχεδιασμό μονοσήμαντης λύσης. Με τον τρόπο αυτόν, διευκολύνεται και επιταχύνεται η διαδικασία του βέλτιστου σχεδιασμού.

## 3.2 ΜΕΘΟΔΟΛΟΓΙΑ ΚΑΙ ΑΝΑΠΤΥΞΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Από το προηγούμενο κεφάλαιο, έγινε κατανοητή η σημασία εφαρμογής κανονισμών κατά την οριζόντια χάραξη ενός δρόμου. Συγκεκριμένα πρέπει να πληρούνται οι απαραίτητες προϋποθέσεις ώστε να υπάρχει αρμονία στην χάραξη και γραμμική μεταβολή της φυγόκεντρης επιτάχυνσης έτσι ώστε η λειτουργική ταχύτητα των οχημάτων να είναι το δυνατό πιο ομοιόμορφη.

Στο παρόν κεφάλαιο περιγράφεται η προσπάθεια δημιουργίας προγράμματος, σε γλώσσα προγραμματισμού Python, για τον υπολογισμό αλλά και την σχεδίαση σε μορφή CAD δύο εκ των τύπων Οριζοντιογραφικών Καμπυλών με Εφαρμογή Κλωθοειδών που αναφέρθηκαν στο κεφάλαιο 2.4, της Σιγμοειδούς και της Ωοειδούς καμπύλης. Τα εν λόγω προγράμματα ονομάστηκαν S-Curve Calculator και Oval-Curve Calculator.
Αφού τέθηκε ο κυρίαρχος στόχος για τη δημιουργία του προγράμματος, στη συνέχεια δίδεται το γενικότερο πλαίσιο και παρουσιάζονται οι εργασίες που γίνονται σε αυτή την κατεύθυνση ώστε το ζήτημα που ερευνάται να περιγραφεί καταλλήλως. Παρακάτω παρουσιάζεται η πορεία σκέψης και τα βήματα:

i. Προσδιορισμός Δεδομένων

Σε πρώτη φάση είναι απαραίτητο να καταγραφούν τα εξής: α)τα δεδομένα που θα απαιτηθούν για την κατασκευή του προγράμματος και β)να προσδιοριστούν οι παράμετροι που θα ορίζει ο χρήστης ως είσοδο για τη λειτουργία του προγράμματος. Ήδη στο κεφάλαιο 2 παρουσιάστηκαν τα δεδομένα που χρειάζονται για τη δημιουργία του κώδικα, κοινώς οι κανονισμοί. Οι παράμετροι που ο χρήστης καλείται να ορίσει θα αναφερθούν στην συνέχεια ξεχωριστά για κάθε είδος καμπύλης.

ii. Καθορισμός Ζητούμενων και Αποτελεσμάτων

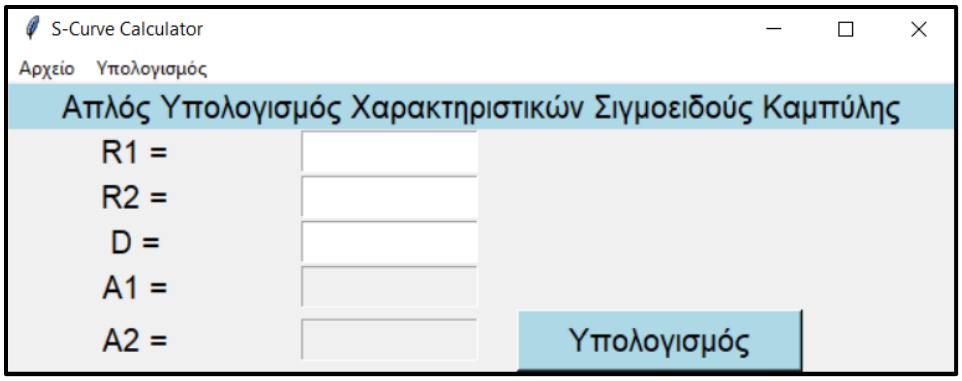
Για τη δημιουργία του προγράμματος σημαντικό είναι να οριστούν τα δεδομένα και τα αποτελέσματα τα οποία είναι προσαρμοσμένα στις ανάγκες των μηχανικών. Κύριος στόχος του προγράμματος, όπως έχει ήδη αναφερθεί, είναι ο αυτοματοποιημένος υπολογισμός των χαρακτηριστικών της κάθε καμπύλης αλλά και η δημιουργία αρχείου CAD με το σχέδιο του άξονα της οδού. Συνεπώς, ζητούμενο είναι αφού ο χρήστης ορίσει τα στοιχεία εισόδου και εκτελεστεί το πρόγραμμα, να δημιουργεί αρχεία σε μορφή αναγνώσιμη από την εφαρμογή Σημειωματάριο ή Notepad που περιέχουν τα βασικά χαρακτηριστικά της εκάστοτε καμπύλης και τις συντεταγμένες των σημείων του άξονα της οδού, καθώς και αρχείο CAD που περιέχει το σχέδιο του άξονα της οδού. Πολύ σημαντικό κριτήριο αποτέλεσε η δημιουργία ενός γραφικού περιβάλλοντος το οποίο να είναι εύκολα κατανοητό από τον χρήστη και όσο το δυνατόν πιο εύκολο στην χρήση.

iii. Καταμερισμός των εργασιών

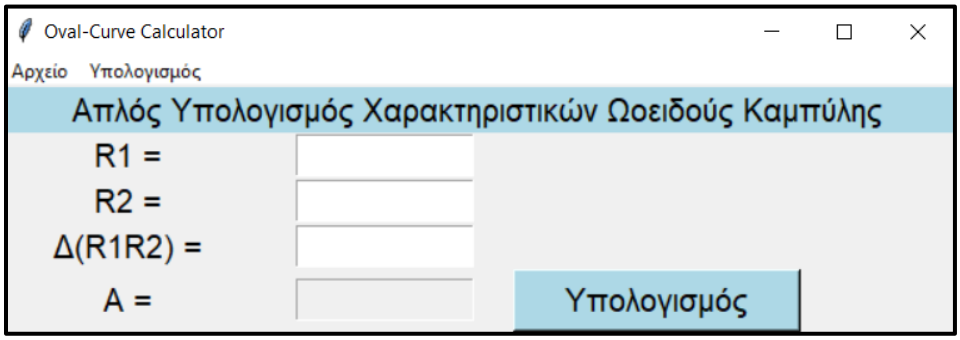
Μετά τον καθορισμό των δεδομένων και των ζητούμενων του προβλήματος διερευνώνται οι επιμέρους εργασίες που πρέπει να γίνουν για να καταστεί εφικτή η λύση του. Σε πρώτη φάση, θα συνταχθεί ο κώδικας μέσω του οποίου θα γίνονται οι έλεγχοι και θα δημιουργούνται τα αρχεία που προαναφέρθηκαν. Έπειτα θα γίνει προσπάθεια σύνδεσης του προγράμματος αυτού με το πρόγραμμα FM19, το οποίο δίνει την δυνατότητα στον χρήστη να εμβαθύνει την μελέτη της χάραξης της οδού μιας και το FM19 προσφέρει την δυνατότητα υπολογισμού και δημιουργίας σχεδίων όχι μόνο οριζοντιογραφίας αλλά και μηκοτομής, διατομών κλπ.

## 3.3 ΔΗΜΙΟΥΡΓΙΑ ΠΡΟΓΡΑΜΜΑΤΟΣ

Όπως προαναφέρθηκε, στόχος της παρούσας διπλωματικής εργασίας αποτελεί η δημιουργία ενός γραφικού περιβάλλοντος το οποίο υπολογίζει τα βασικά χαρακτηριστικά και σχεδιάζει τον άξονα της Σιγμοειδούς και Ωοειδούς καμπύλης.
Λόγω του ότι οι διαφορές των δύο αυτών τύπων καμπύλης είναι πολλές, η προσέγγιση έγινε ξεχωριστά για την καθεμία. Συγκεκριμένα δημιουργήθηκαν δύο διαφορετικά προγράμματα, ένα για την κάθε καμπύλη (Εικόνα 3.1 , 3.2), που όμως ακολουθούν την ίδια ιδεολογία και αντίστοιχες πρακτικές. Για τον λόγο αυτό θα γίνει η ανάλυση της διαδικασίας σε διαφορετικά πεδία για την κάθε καμπύλη στην συνέχεια.

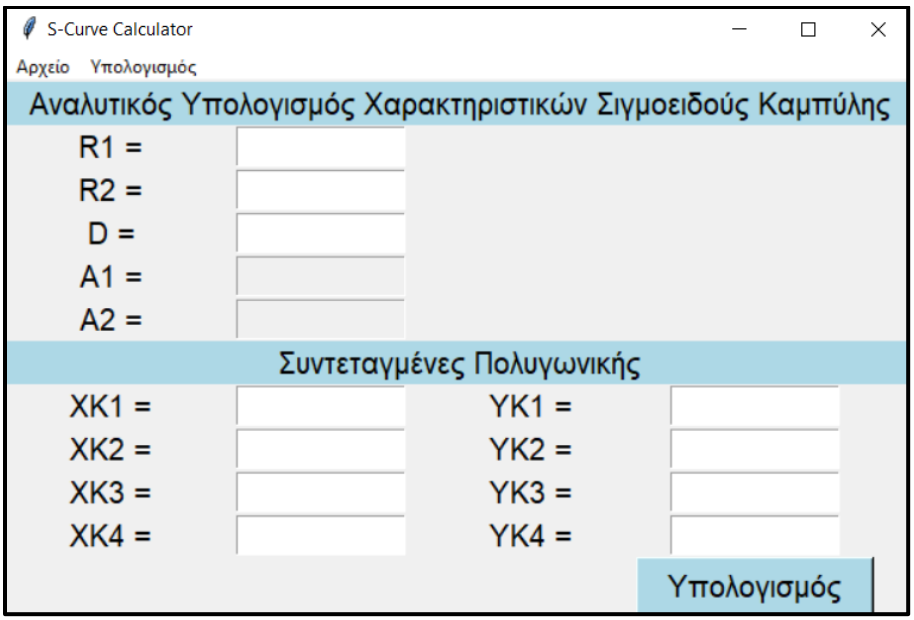



Εικόνα 3.1: Γραφικό Περιβάλλον Απλού Υπολογισμού Σιγμοειδούς Καμπύλης.

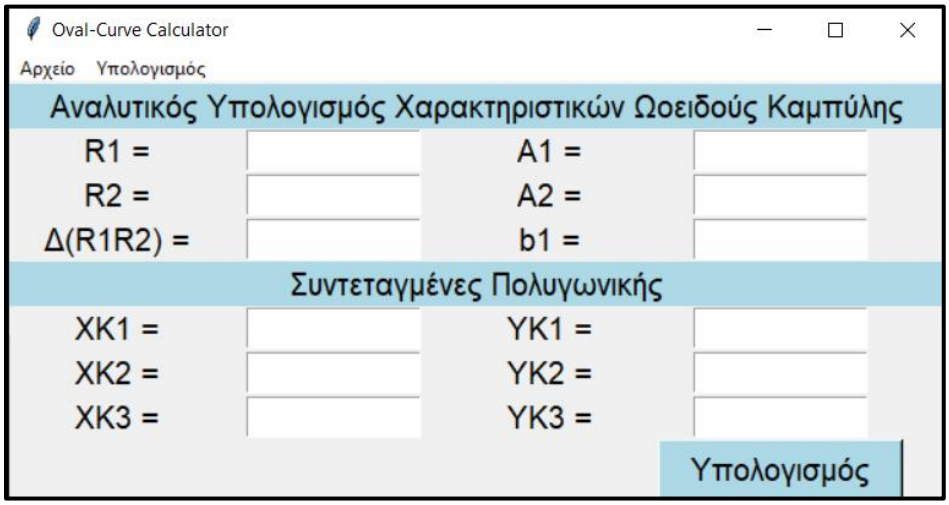



Εικόνα 3.2: Γραφικό Περιβάλλον Απλού Υπολογισμού Ωοειδούς Καμπύλης.

Επίσης δεδομένου ότι ο υπολογισμός της παραμέτρου Α τόσο για την Σιγμοειδή όσο και για την Ωοειδή καμπύλη αποτελεί από μόνο του ένα πρόβλημα, αποφασίστηκε η δημιουργία δύο λειτουργιών (Modes) σε κάθε πρόγραμμα. Στην Πρώτη λειτουργία-Απλός Υπολογισμός (mode 1) ο χρήστης εισάγει ως δεδομένα μόνο τα απαραίτητα για τον υπολογισμό και την εύρεση της παραμέτρου Α, και έχει ως αποτελέσματα την τιμή της παραμέτρου Α στην οθόνη του καθώς και αρχείο κειμένου που περιέχει τα βασικά χαρακτηριστικά της καμπύλης (Εικόνα 3.1 , 3.2). Στην Δεύτερη λειτουργία-Αναλυτικός Υπολογισμός (mode 2) ο χρήστης καλείται να εισάγει περισσότερα στοιχεία, ανάμεσα σε αυτά και οι συντεταγμένες της πολυγωνικής, έχοντας τώρα ως αποτέλεσμα τρία διαφορετικά αρχεία τα οποία θα αναλυθούν περαιτέρω παρακάτω (Εικόνες 3.3 , 3.4).

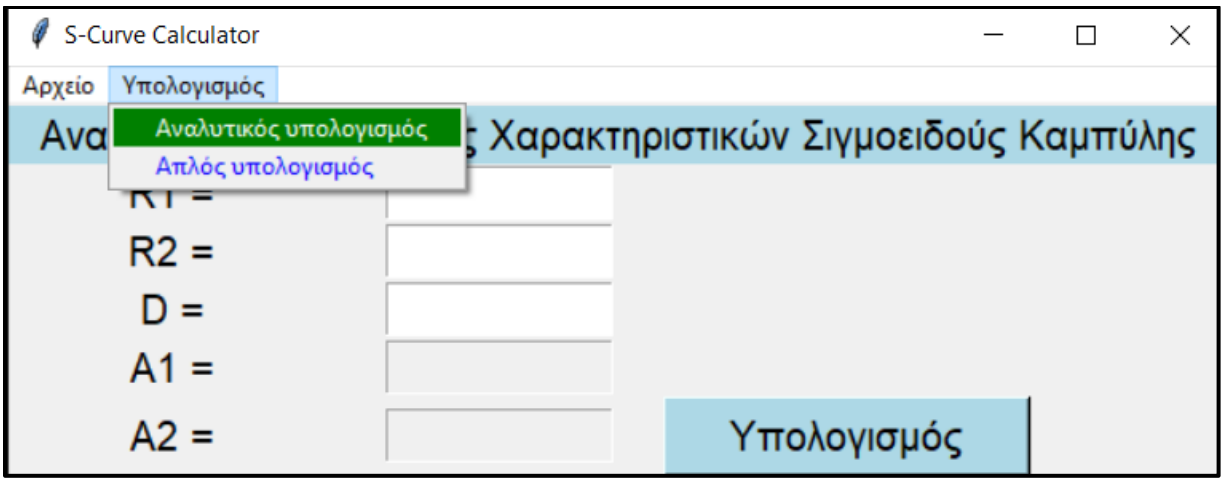



Εικόνα 3.3: Γραφικό Περιβάλλον Αναλυτικού Υπολογισμού Σιγμοειδούς Καμπύλης.



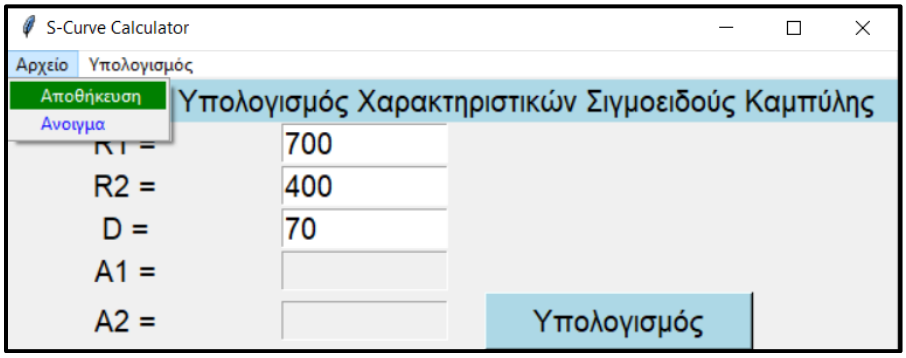

Εικόνα 3.4: Γραφικό Περιβάλλον Αναλυτικού Υπολογισμού Ωοειδούς Καμπύλης.

Ο χρήστης έχει την δυνατότητα ανά πάσα στιγμή να διαλέξει ποια από τις δύο λειτουργίες θέλει να χρησιμοποιήσει ανάλογα τις ανάγκες του (Εικόνα 3.5).

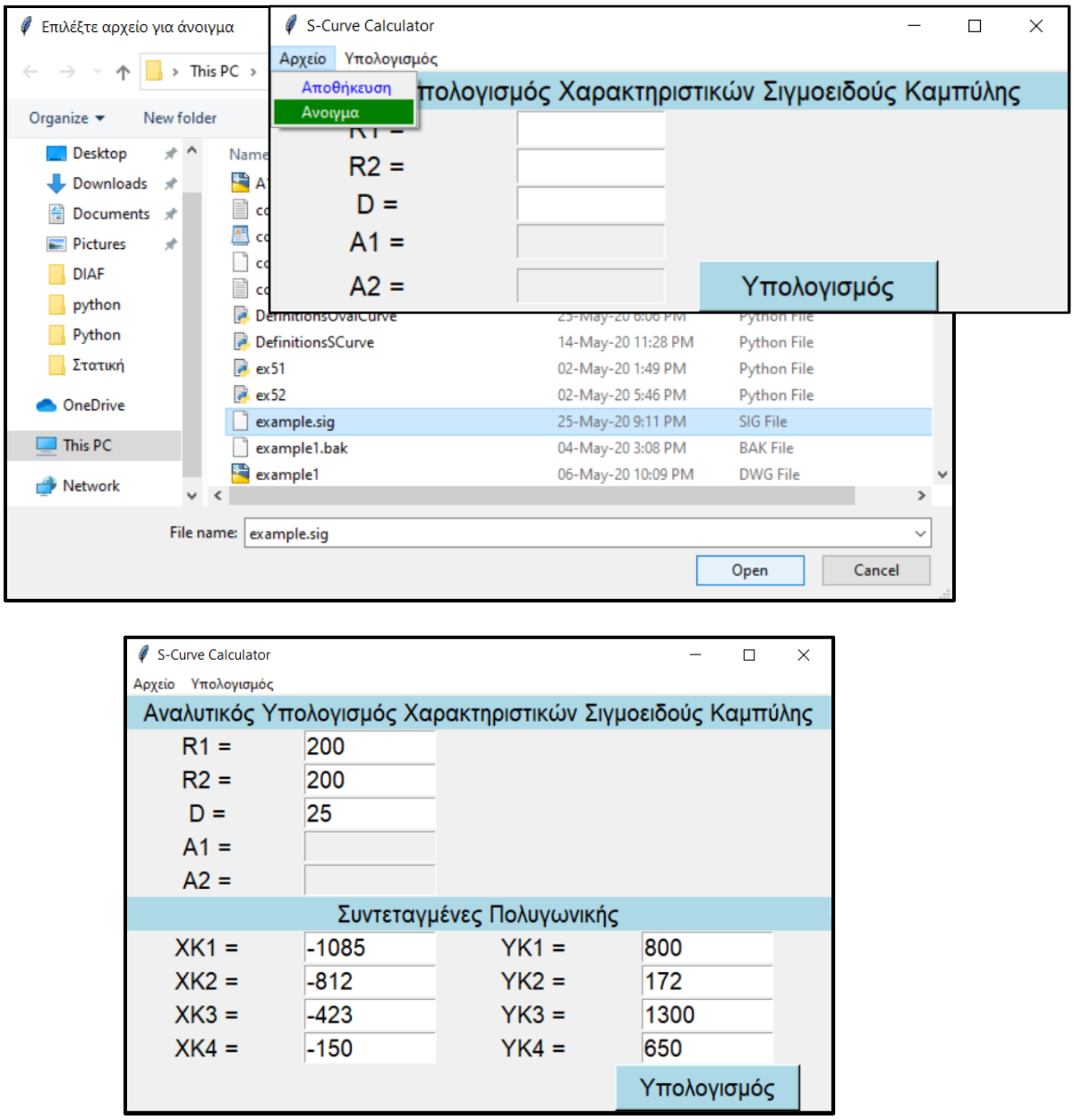



Εικόνα 3.5: Επιλογή λειτουργίας (mode) από το Μενού του προγράμματος.

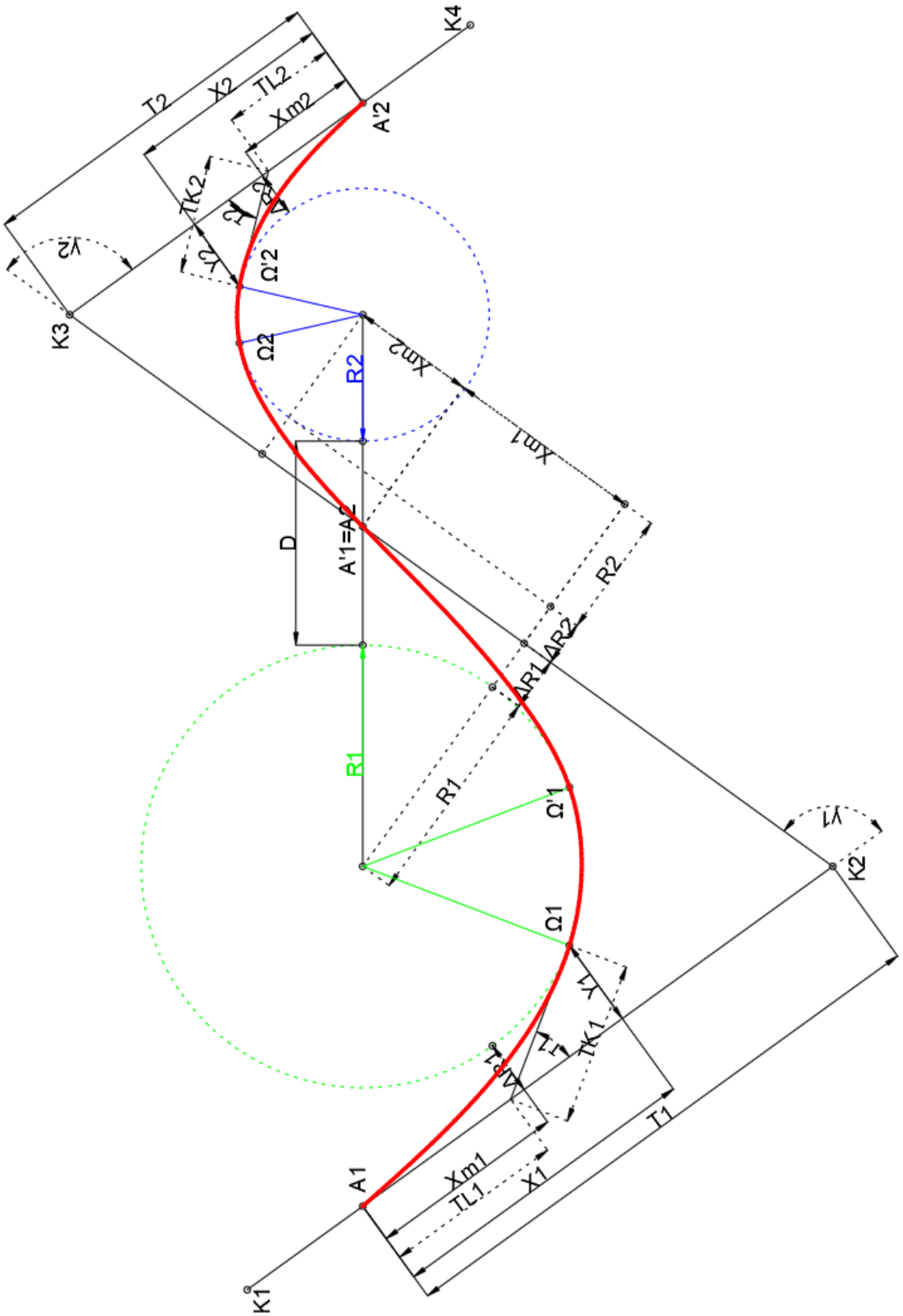
Για λόγους φιλικότητας προς τον χρήστη προστέθηκε στο πρόγραμμα και η δυνατότητα αποθήκευσης των δεδομένων που εισάγει ο χρήστης κάθε φορά, ώστε να μπορεί σε μετέπειτα φάση να χρησιμοποιήσει τα ίδια δεδομένα αν τα χρειαστεί. Για τον ίδιο ακριβώς λόγο, ο χρήστης έχει και την δυνατότητα να ανοίξει ένα ήδη αποθηκευμένο αρχείο δεδομένων μέσω του προγράμματος και αυτόματα τα δεδομένα αυτά να εισαχθούν στο πρόγραμμα.




Εικόνα 3.6: Δυνατότητα αποθήκευσης αρχείου δεδομένων από το Μενού του προγράμματος για μετέπειτα χρήση.




Εικόνα 3.7: Δυνατότητα ανοίγματος ήδη αποθηκευμένου αρχείου δεδομένων από το Μενού του προγράμματος.




Σχήμα 3.1: Σιγμοειδής Καμπύλη

### 3.3.1 Σιγμοειδής Καμπύλη

Βήμα 1: Υπολογισμός της παραμέτρου A1 και A2 (Λειτουργία = Απλός Υπολογισμός)
Δεδομένα: R1, R2 , D (Σχήμα 3.1)
Ζητούμενα: A1, A2

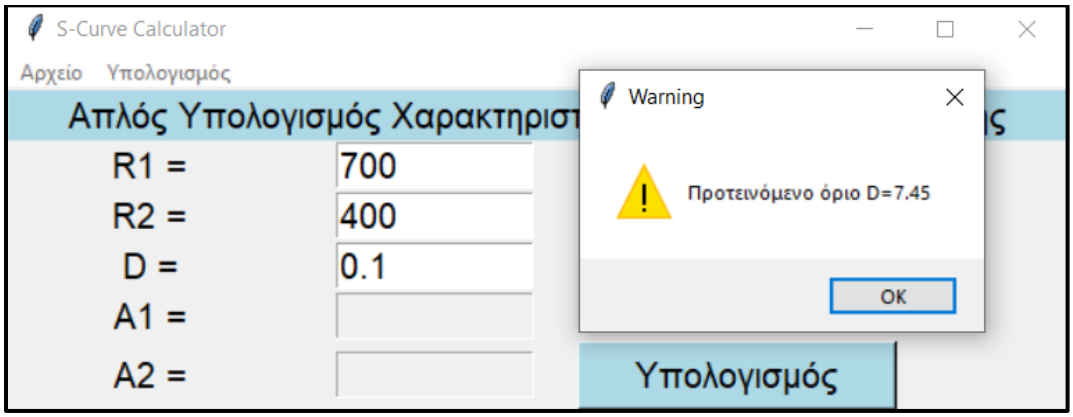
Όπως αναφέρθηκε και στην βιβλιογραφική ανασκόπηση το αρχικό ζητούμενο για τον υπολογισμό της Σιγμοειδούς καμπύλης είναι η εύρεση της κοινής παραμέτρου A=A1=A2 για τις δύο καμπύλες. Αυτή μπορεί να καθοριστεί μονοσήμαντα βάσει της εξίσωσης (2.23) η οποία αναφέρεται και παρακάτω:

$$s = 0 = -(R_1 + R_2 + D)^2 + (x_{m1} + x_{m2})^2 + (R_1 + \Delta R_1 + R_2 + \Delta R_2)^2 \qquad (3.1)$$

Με αντικατάσταση των εξισώσεων (2.7), (2.16), (2.17), (2.18) στην (2.23) προκύπτει μία εξίσωση ενός μόνο αγνώστου, της παραμέτρου A. Η επίλυση αυτής όμως, είναι αρκετά δύσκολη οπότε χρησιμοποιήθηκαν διαδοχικές δοκιμές. Συγκεκριμένα, γνωρίζοντας τα επιτρεπτά όρια της παραμέτρου A σύμφωνα με τις εξισώσεις (2.19) και (2.20):

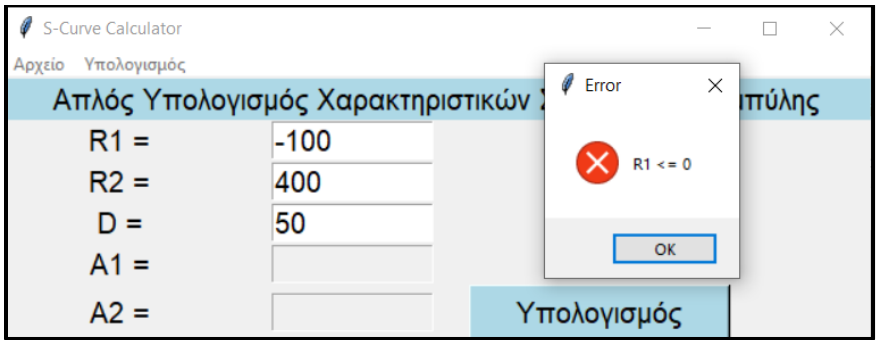
$$\frac{R}{3} \leq A_i \leq R \qquad (3.2)$$

δημιουργήθηκε ένας βρόχος ο οποίος λαμβάνει ως αρχική τιμή η μικρότερη δυνατή τιμή του A, δηλαδή $\frac{R_{min}}{3}$, και υπολογίζει την εξίσωση (3.1). Αν η τιμή του s είναι διάφορη του μηδενός, αυξάνει την τιμή του A με βήμα 0.05 μέχρις ότου να βρεθεί η κατάλληλη τιμή του A που μηδενίζει την εξίσωση. Ο βρόχος τερματίζει στην μεγαλύτερη δυνατή τιμή του A, δηλαδή $R_{max}$ . Σε περίπτωση που δεν βρεθεί κοινή παράμετρος A=A1=A2 που να μηδενίζουν την εξίσωση (3.1), αναζητούνται παράμετροι που να ικανοποιούν την εξίσωση (2.30) $A1 \leq 1.50 * A2$. Συγκεκριμένα δημιουργήθηκε εξωτερικός βρόχος του προηγούμενου, ο οποίος λαμβάνει ως αρχικό λόγο A1/A2=1, τελικό λόγο A1/A2=1.5 και με βήμα 0.01 επαναλαμβάνει τη διαδικασία μέχρις ότου βρεθούν οι παράμετροι A1,A2. Σε περίπτωση που δεν βρεθεί και πάλι παράμετρος A1,A2 που να ικανοποιούν την εξίσωση (3.1), ο χρήστης ειδοποιείται με μήνυμα στην οθόνη του, το οποίο προτείνει το μέγιστο ή ελάχιστο όριο που μπορεί να έχει η περιφερειακή απόσταση των 2 κύκλων D (Εικόνα 3.8).



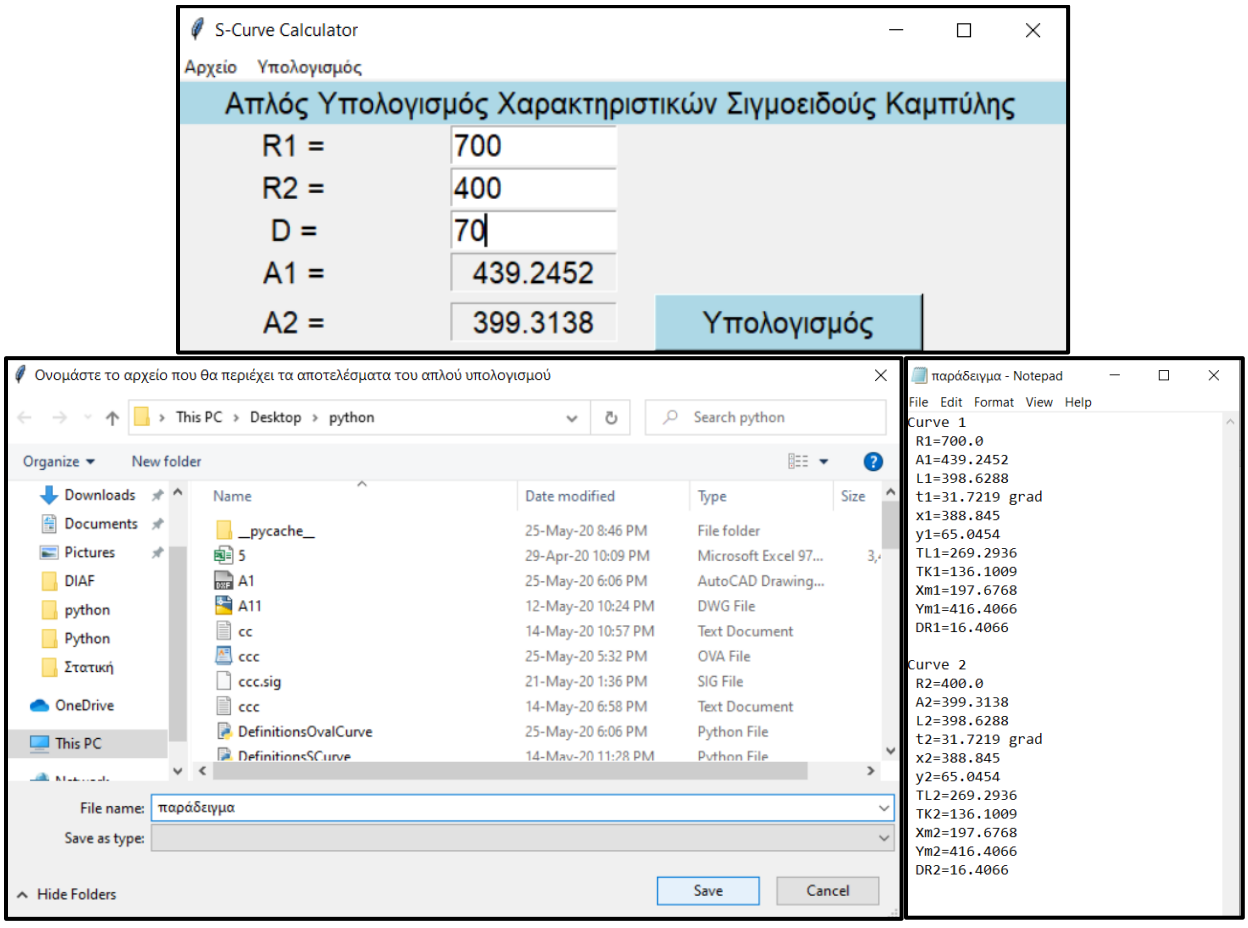
Εικόνα 3.8: Ειδοποίηση χρήστη για προτεινόμενη τιμή της περιφερειακής απόστασης D.

Όπως αναφέρθηκε και προηγουμένως, μεγάλη βάση δόθηκε στην φιλικότητα προς τον χρήστη έτσι σε περίπτωση εισαγωγής λάθος δεδομένων, υπάρχει ενημέρωση στην οθόνη του (Εικόνα 3.9).




Εικόνα 3.9: Ειδοποίηση χρήστη για λάθος εισαγωγή δεδομένων.

Με το πάτημα του κουμπιού «Υπολογισμός» ο χρήστης μπορεί να δει στην οθόνη του το ζητούμενο και του δίνεται η δυνατότητα αν θέλει να αποθηκεύσει τα αποτελέσματα του Απλού Υπολογισμού σε αρχείο κειμένου με όνομα της επιλογής του (Εικόνα 3.10).




Εικόνα 3.10: Προβολή αποτελέσματος στην οθόνη του χρήστη και εξαγωγή περεταίρω χαρακτηριστικών Σιγμοειδούς καμπύλης σε αρχείο κειμένου.

Για τον υπολογισμό των χαρακτηριστικών της κάθε καμπύλης χρησιμοποιήθηκαν οι τύποι που αναφέρθηκαν στο κεφάλαιο 2. Παρόλα αυτά παρουσιάζονται και στον επόμενο πίνακα συγκεντρωτικά για διευκόλυνση.

| |
|---|
| $L = \frac{A^2}{R}$ |
| $\tau^{rad} = \frac{L}{2R}$ |
| $X = L - \frac{L^5}{40A^4} + \frac{L^9}{3456A^8} - \cdots$ |
| $Y = \frac{L^3}{6A^2} + \frac{L^7}{336A^6} + \frac{L^{11}}{42240A^{10}} - \cdots$ |
| $X_m = X - R * sin\tau$ |
| $Y_m = Y + R * cos\tau$ |
| $T_L = x - \frac{y}{tan\tau}$ |
| $T_K = \frac{y}{sin\tau}$ |
| $\delta = \frac{(R + \Delta R)}{cos\left(\frac{\gamma}{2}\right) - R}$ |
| $\Delta R = y_m - R$ |
| $t = (R + \Delta R) tan\left(\frac{\gamma}{2}\right)$ |
| $T = t + X_m$ |
| $\alpha^{rad} = \frac{\gamma * \pi}{200} - \frac{L}{R}$ |
| $\Omega\Omega' = R * \alpha^{rad}$ |
| $A\Omega\Omega'\Delta' = 2L + \Omega\Omega'$ |

Πίνακας 3.1: Τύποι υπολογισμού οριζόντιας καμπύλης.

Βήμα 2: Εύρεση διευθύνσεων πολυγωνικής (Λειτουργία = Αναλυτικός υπολογισμός)

Για τον καθορισμό των διευθύνσεων χρησιμοποιήθηκε διανυσματική γεωμετρία. Συγκεκριμένα, θεωρώντας τρεις διαδοχικές κορυφές της πολυγωνικής K1, K2, K3 όπου K2, τα μοναδιαία διανύσματα στις πλευρές K1K2 και K3K2 είναι:

$$\overrightarrow{t_{12}} = \left[\frac{X_{K2}-X_{K1}}{\sqrt{(X_{K2}-X_{K1})^2+(Y_{K2}-Y_{K1})^2}}, \frac{Y_{K2}-Y_{K1}}{\sqrt{(X_{K2}-X_{K1})^2+(Y_{K2}-Y_{K1})^2}}\right] \quad (3.3)$$

$$\overrightarrow{t_{32}} = \left[\frac{X_{K2}-X_{K3}}{\sqrt{(X_{K2}-X_{K3})^2+(Y_{K2}-Y_{K3})^2}}, \frac{Y_{K2}-Y_{K3}}{\sqrt{(X_{K2}-X_{K3})^2+(Y_{K2}-Y_{K3})^2}}\right] \quad (3.4)$$

Τα αντίστοιχα μοναδιαία κάθετα διανύσματα στις πλευρές είναι:

$$\overrightarrow{n12} = [-t_{12}(2), t_{12}(1)] \quad (3.5)$$

$$\overrightarrow{n32} = [-t_{32}(2), t_{32}(1)] \quad (3.6)$$

Η φορά των κάθετων διανυσμάτων μπορεί να είναι προς το εσωτερικό της καμπύλης ή προς το εξωτερικό. Για να έχουν τη σωστή φορά (προς το εσωτερικό), υπολογίζεται διάνυσμα παράλληλο με τη διχοτόμο το οποίο έχει πάντα φορά προς το εσωτερικό της καμπύλης:

$$\vec{\delta} = \vec{t}_{21} + \vec{t}_{23} = -\vec{t}_{12} - \vec{t}_{32} \quad (3.7)$$

Έτσι αν το εσωτερικό γινόμενο ενός από τα κάθετα διανύσματα επί το διάνυσμα διχοτόμου είναι θετικό, τότε η φορά του είναι σωστή, αλλιώς πρέπει να γίνει αντίθετη:

Αν $\vec{\delta} * \overrightarrow{n12} < 0$ τότε τίθεται $\overrightarrow{n12} = -\overrightarrow{n12}$ (3.8)

Αν $\vec{\delta} * \overrightarrow{n32} < 0$ τότε τίθεται $\overrightarrow{n32} = -\overrightarrow{n32}$ (3.9)

Έχοντας λοιπόν τα μοναδιαία διανύσματα μπορεί πολύ εύκολα να υπολογιστεί η εσωτερική γωνία β των δύο ευθειών και έπειτα η γωνία αλλαγής διεύθυνσης γ:

$$\vec{t}_{21} * \vec{t}_{23} = |\vec{t}_{21}| * |\vec{t}_{23}| * cos\beta => \beta = cos^{-1}(\vec{t}_{21} * \vec{t}_{23}) \quad (3.10)$$

$$\gamma = \pi - \beta \quad (3.11)$$

Με τον ίδιο τρόπο υπολογίστηκαν και οι δύο γωνίες γ1, γ2.

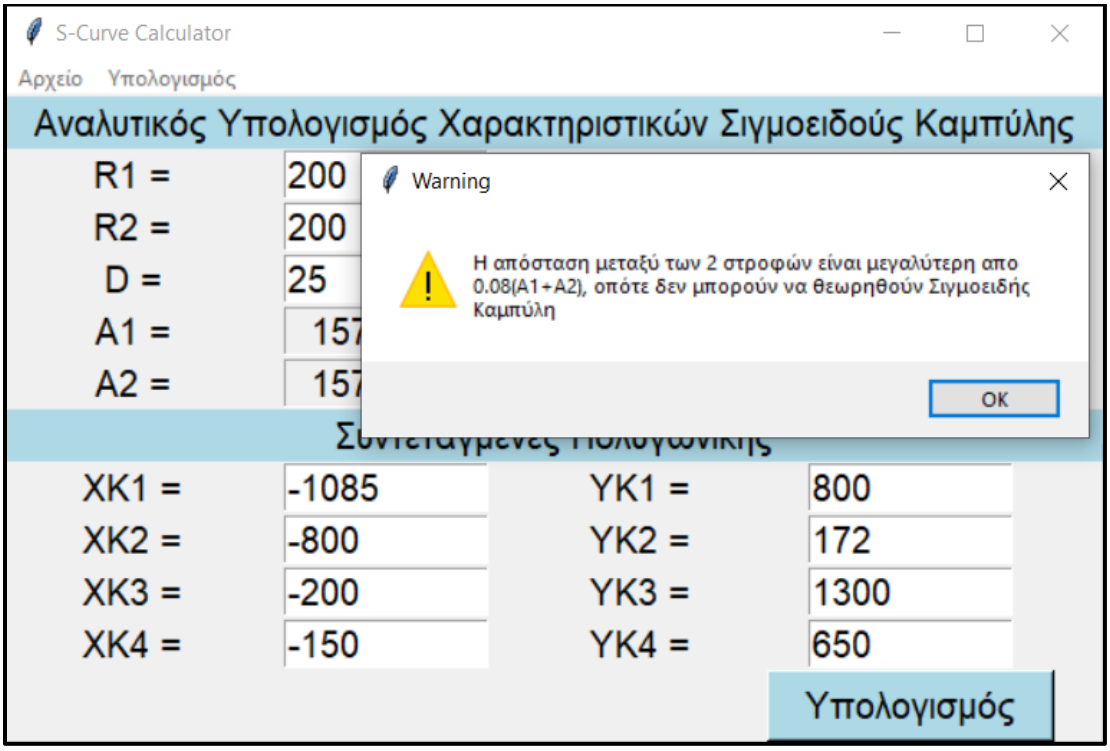
Βήμα 3: Έλεγχος αν η καμπύλη χωράει στην πολυγωνική (Λειτουργία = Αναλυτικός υπολογισμός)

Ένας από τους πιο σημαντικούς ελέγχους για την Σιγμοειδή Καμπύλη είναι αν υπάρχει επαρκές μήκος για την ανάπτυξη της. Στα άκρα K1, K4 δεν υπάρχει πρόβλημα μιας και αν δεν επαρκεί το μήκος K1K2 ή K3K4 βάσει διανυσματικής γεωμετρίας τα σημεία A1 και A2΄ θα υπολογιστούν κανονικά, απλά θα βρίσκονται εκτός του τμήματος K1K2 ή K3K4 αντίστοιχα αλλά θα έχουν την σωστή διεύθυνση. Το τμήμα που δεν μπορεί να μεταβληθεί όμως είναι το K2K3. Μέσα σε αυτό το τμήμα πρέπει να χωράνε να αναπτυχθούν και οι δύο καμπύλες. Έτσι λοιπόν το πρόγραμμα χρησιμοποιώντας την παρακάτω σχέση:

$$L_Z = (K1K2) - T_1 - T_2 \quad (3.12)$$

ενημερώνει τον χρήστη για το αν επαρκεί η ευθυγραμμία ανάμεσα στα K2K3. Όπως αναφέρθηκε και στο κεφάλαιο 2.4.2 η εξίσωση (3.12) έχει κάποια όρια. Συγκεκριμένα αν:

- Lz < 0 ο χρήστης ενημερώνεται ότι οι καμπύλες δεν χωράνε
- Lz < 0.08(A1+A2) η διαδικασία συνεχίζει κανονικά χωρίς κάποια ειδοποίηση
- Lz > 0.08(A1+A2) ο χρήστης ενημερώνεται πως η απόσταση είναι μεγαλύτερη από το όριο οπότε δεν μπορεί να θεωρηθεί Σιγμοειδής καμπύλη (Εικόνα 3.11).




Εικόνα 3.11: Ενημέρωση χρήστη σε περίπτωση που ο έλεγχος της ευθυγραμμίας δεν είναι στα αποδεκτά όρια.

<u>Βήμα 4: Υπολογισμός Συντεταγμένων Άξονα οδού (Λειτουργία = Αναλυτικός υπολογισμός)</u>
Για την εύρεση των σημείων A1, A1΄, A2, A2΄ χρησιμοποιείται όπως προαναφέρθηκε διανυσματική γεωμετρία. Έχοντας δηλαδή την απόσταση Ti από την κάθε κορυφή υπολογίζονται οι συντεταγμένες των σημείων Ai. Για παράδειγμα:

$$X_{A1} = X_{K1} + \overrightarrow{t12}[1] * ((K1K2) - T1) \qquad (3.13)$$

$$Y_{A1} = Y_{K1} + \overrightarrow{t12}[2] * ((K1K2) - T1) \qquad (3.14)$$

Οι συντεταγμένες τυχαίου σημείου μεταξύ Α και Ω, σε απόσταση s (s ≤ L) από το Α δίνονται από τους τύπους:

$$\bar{X} = s - \frac{s^5}{40A^4} + \frac{s^9}{3456A^8} - \cdots = s + \sum_{k=4,8,12,16,\ldots} \left[ \frac{(-1)^{k/4} * s^{k+1}}{2^{k/2} * (k+1) * (k/2)! * A^k} \right] \qquad (3.15)$$

$$\bar{Y} = \frac{s^3}{6A^2} + \frac{s^7}{336A^6} + \frac{s^{11}}{42240A^{10}} - \cdots \sum_{k=2,6,10,14,\ldots} \left[ \frac{(-1)^{(k-2)/4} * s^{k+1}}{2^{k/2} * (k+1) * (k/2)! * A^k} \right] \qquad (3.16)$$

Στην ειδική περίπτωση που s=L, οι συντεταγμένες ταυτίζονται με τις συντεταγμένες του σημείου Ω. Οι συντεταγμένες τυχαίου σημείου μεταξύ Ω και Δ, σε απόσταση s (L≤ s ≤M/2) από το Α δίνονται από τις σχέσεις:

$$\bar{X} = Xm + R * sin\left(\tau + \frac{s-L}{R}\right) \quad (3.17)$$

$$\bar{Y} = \Delta R + R - R * cos\left(\tau + \frac{s-L}{R}\right) \quad (3.18)$$

Όλες οι συντεταγμένες αναφέρονται σε τοπικό σύστημα με άξονα $\bar{X}$ τον αριστερό κλάδο ΑΚ (ή δεξί κλάδο Α'Κ), αρχή αξόνων το σημείο Α (ή Α') και άξονα $\bar{Y}$ κάθετο στον άξονα και με φορά προς το εσωτερικό της καμπύλης (της γωνίας ΑΚΑ').

Έχοντας τα μοναδιαία διανύσματα, η μετατροπή τυχαίου σημείου με συντεταγμένες $\bar{X}, \bar{Y}$ στο τοπικό σύστημα του αριστερού κλάδου της καμπύλης μετατρέπονται στο γενικό σύστημα ως εξής:

$$\vec{P} = \bar{X}\,\overrightarrow{t_{12}} + \bar{Y}\,\overrightarrow{n12}\,,\ X = X_A + P(1)\ ,\ Y = Y_A + P(2) \quad (3.19)$$

Για το δεξιό κλάδο:

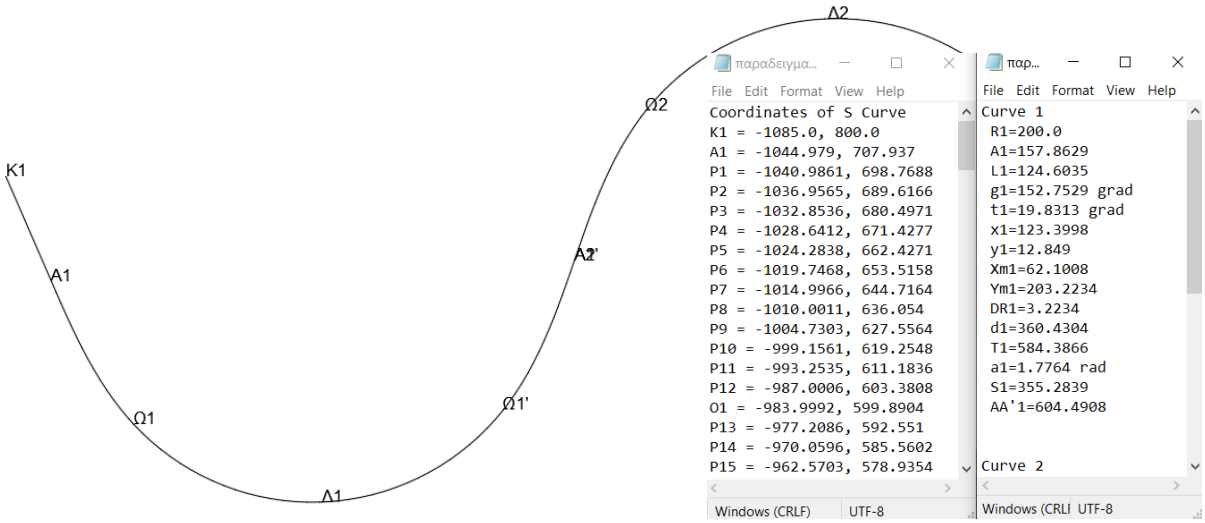
$$\vec{P} = \bar{X}\,\overrightarrow{t_{32}} + \bar{Y}\,\overrightarrow{n32}\,,\ X = X_A + P(1)\ ,\ Y = Y_A + P(2) \quad (3.20)$$

Βήμα 5: Εξαγωγή Αποτελεσμάτων (Λειτουργία = Αναλυτικός υπολογισμός)

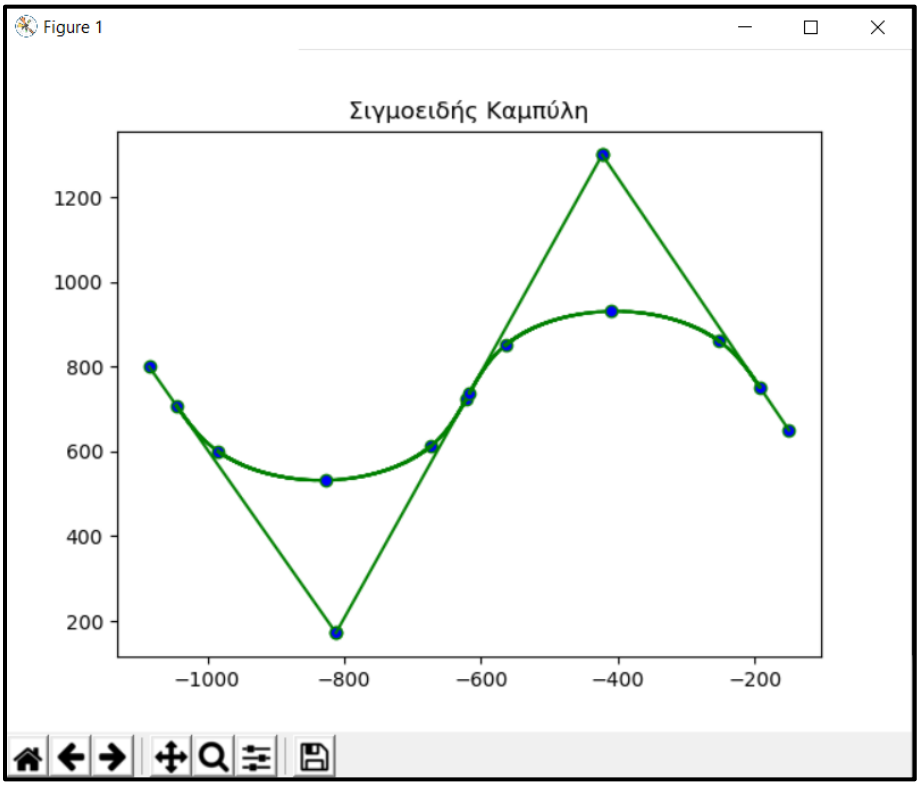
Τα αποτελέσματα αποθηκεύονται, αν θέλει ο χρήστης σε αρχεία με ονομασία που επιλέγει ο ίδιος. Συγκεκριμένα μπορούν να παραχθούν τα εξής αρχεία (Εικόνα 3.12):

- <πρόθεμα>.sig, το οποίο περιλαμβάνει τα βασικά χαρακτηριστικά της Σιγμοειδούς Καμπύλης
- <πρόθεμα>.txt , το οποίο περιλαμβάνει τις συντεταγμένες του άξονα της Σιγμοειδούς Καμπύλης
- <πρόθεμα>.dxf , το οποίο περιλαμβάνει το σχέδιο με τον άξονα της Σιγμοειδούς Καμπύλης και τα βασικά της σημεία

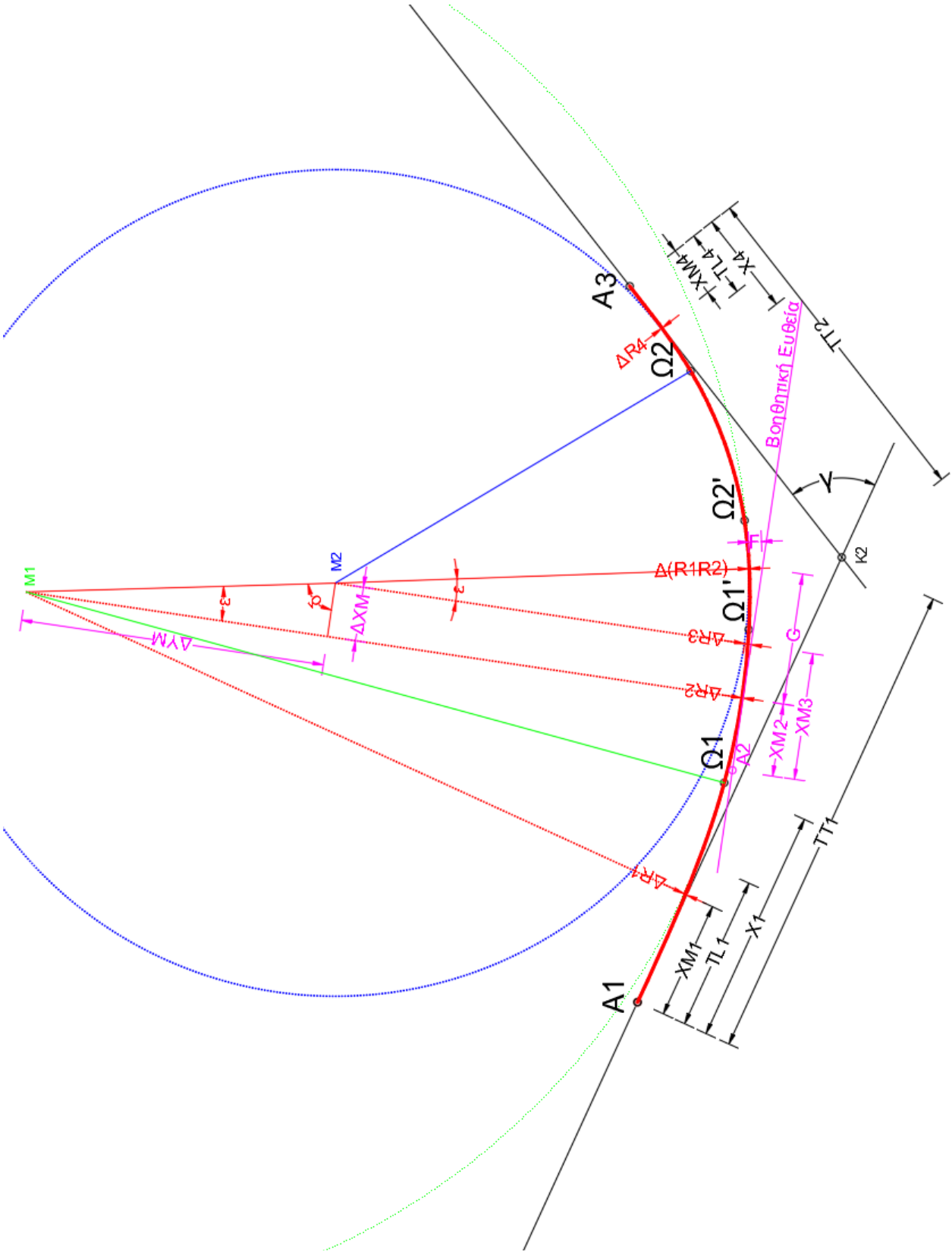
Είτε αποθηκεύσει τα αρχεία του είτε όχι, στο τέλος του προγράμματος εμφανίζεται στην οθόνη του χρήστη ένα γράφημα (Plot) με το σχέδιο της Πολυγωνικής, της Σιγμοειδούς καμπύλης καθώς και τα βασικά της σημεία (Εικόνα 3.13).




Εικόνα 3.12: Εξαγωγή αποτελεσμάτων Σιγμοειδούς καμπύλης σε αρχεία κειμένου και αρχείο CAD.




Εικόνα 3.13: Εξαγωγή σκαριφήματος Σιγμοειδούς καμπύλης στην οθόνη του χρήστη.




Σχήμα 3.2: Ωοειδής Καμπύλη

### 3.3.2 Ωοειδής Καμπύλη

Βήμα 1: Υπολογισμός της παραμέτρου A (Λειτουργία = Απλός Υπολογισμός)
Δεδομένα: R1, R2 , Δ(R1, R2) (Σχήμα 3.2)
Ζητούμενο: A

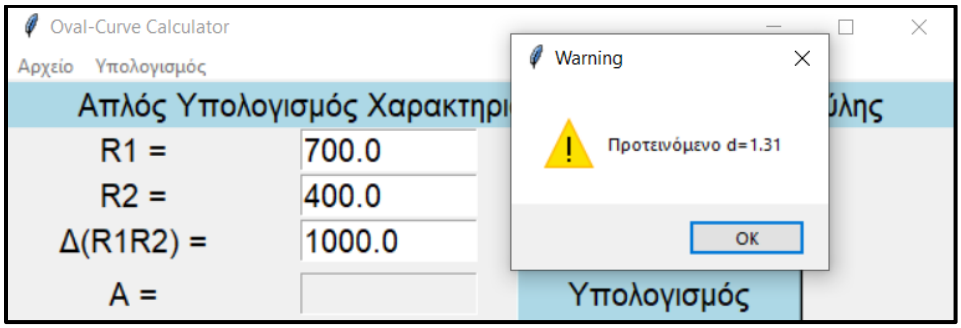
Όπως αναφέρθηκε και στην βιβλιογραφική ανασκόπηση το αρχικό ζητούμενο για τον υπολογισμό της Ωοειδούς καμπύλης είναι η εύρεση της κοινής παραμέτρου A. Αυτή μπορεί να καθοριστεί μονοσήμαντα βάσει της εξίσωσης (2.42) η οποία αναφέρεται και παρακάτω:

$$s = -\Delta(R_1R_2) + R_1 - K_1K_2 - R_2 \qquad (3.21)$$

όπου K1K2 υπολογίζεται από την εξίσωση (2.41) η οποία αναφέρεται και παρακάτω:

$$K_1K_2 = \sqrt{(x_{m1} - x_{m2})^2 + (y_{m1} - y_{m2})^2} \qquad (3.22)$$

Με αντικατάσταση των εξισώσεων (2.7), (2.16), (2.17) στην (2.42) προκύπτει μία εξίσωση ενός μόνο αγνώστου, της παραμέτρου A. Η επίλυση αυτής όμως, είναι αρκετά δύσκολη οπότε χρησιμοποιήθηκαν διαδοχικές δοκιμές. Συγκεκριμένα, γνωρίζοντας τα επιτρεπτά όρια της παραμέτρου A σύμφωνα με τις εξισώσεις (2.19) και (2.20), δημιουργήθηκε ένας βρόχος ο οποίος λαμβάνει ως αρχική τιμή η μικρότερη δυνατή τιμή του A, δηλαδή $\frac{R_{min}}{3}$, και υπολογίζει την εξίσωση (3.21). Αν η τιμή του s είναι διάφορη του μηδενός, αυξάνει την τιμή του A με βήμα 0.05 μέχρις ότου να βρεθεί η κατάλληλη τιμή του A που μηδενίζει την εξίσωση. Ο βρόχος τερματίζει στην μεγαλύτερη δυνατή τιμή του A, δηλαδή $R_{max}$. Σε περίπτωση που δεν βρεθεί κοινή παράμετρος A=A1=A2 που να μηδενίζει την εξίσωση (3.21), ο χρήστης ειδοποιείται με μήνυμα στην οθόνη του, το οποίο προτείνει το μέγιστο ή ελάχιστο όριο που μπορεί να έχει το ελάχιστο μήκος μεταξύ των εκατέρωθεν της ωοειδούς κυκλικών τόξων Δ(R1R2) (Εικόνα 3.14).

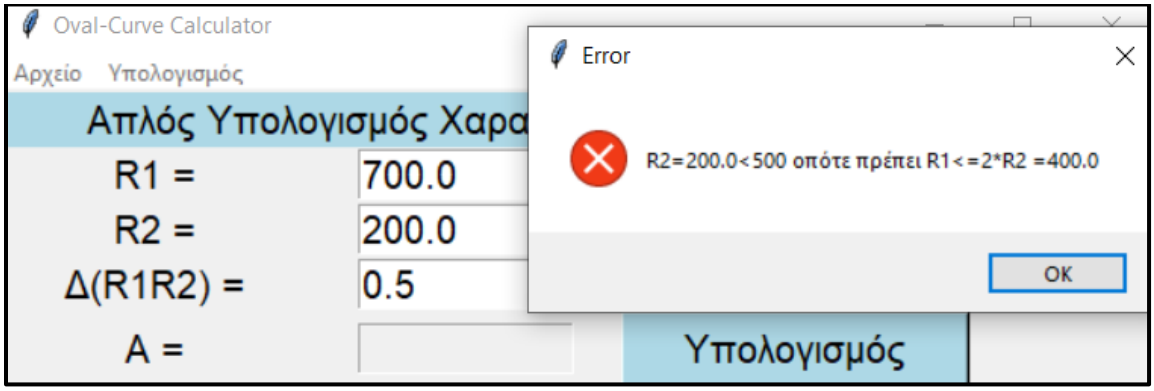



Εικόνα 3.14: Ειδοποίηση χρήστη για προτεινόμενη τιμή του ελάχιστου μήκους μεταξύ των εκατέρωθεν της ωοειδούς κυκλικών τόξων Δ(R1R2).

Όπως αναφέρθηκε και προηγουμένως, μεγάλη βάση δόθηκε στην φιλικότητα προς τον χρήστη έτσι σε περίπτωση εισαγωγής λάθος δεδομένων, υπάρχει ενημέρωση στην οθόνη του. Πέρα από τον βασικό έλεγχο δεδομένων που εισάγει ο χρήστης (τιμές >0), στην Ωοειδή καμπύλη πρέπει να ελεγχθεί και η σχέση μεταξύ των ακτινών. Συγκεκριμένα, όπως αναφέρθηκε και στο

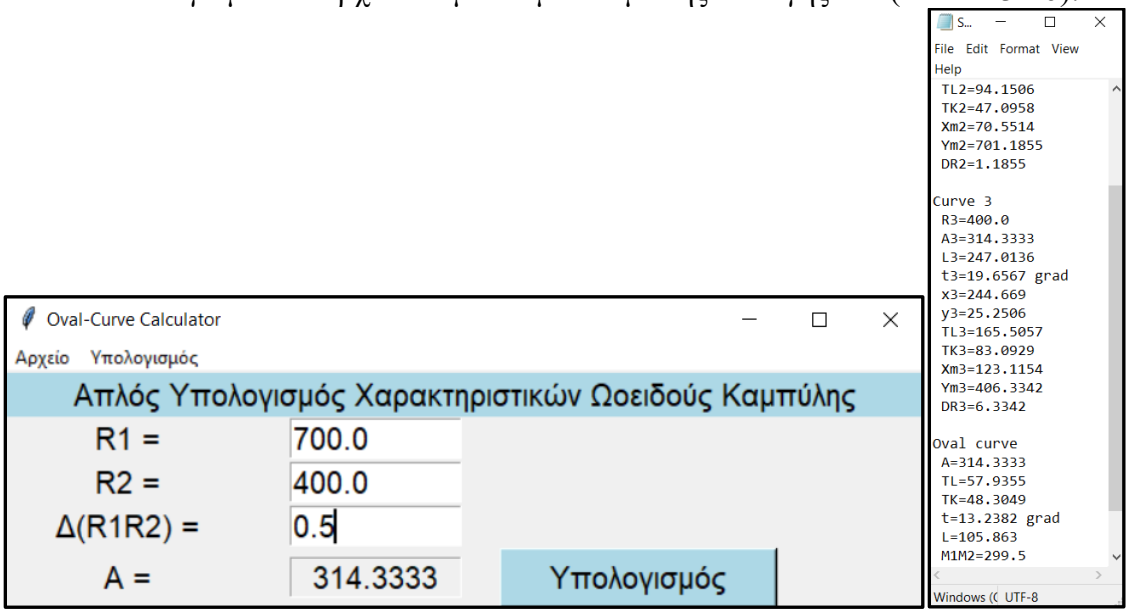
κεφάλαιο 2.4.3, το πρόγραμμα ενημερώνει τον χρήστη σε περίπτωση εισόδου λάθος τιμών για τις ακτίνες R1, R2. Συγκεκριμένα, με βάση τον πίνακα (2.3), αν:

- $R2 \leq 100$ και $R1>1.5* R2$ ο χρήστης ενημερώνεται για την μέγιστη τιμή του R1
- $100 < R2 \leq 500$ και $R1>2.0* R2$ ο χρήστης ενημερώνεται για την μέγιστη τιμή του R1 (Εικόνα 3.15)
- Σε κάθε άλλη περίπτωση η διαδικασία συνεχίζει κανονικά χωρίς κάποια ειδοποίηση




Εικόνα 3.15: Ενημέρωση χρήστη σε περίπτωση που ο έλεγχος των ακτινών R1, R2 δεν είναι στα αποδεκτά όρια.

Με το πάτημα του κουμπιού «Υπολογισμός» ο χρήστης και πάλι μπορεί να δει στην οθόνη του το ζητούμενο και του δίνεται η δυνατότητα αν θέλει να αποθηκεύσει τα αποτελέσματα του Απλού Υπολογισμού σε αρχείο κειμένου με όνομα της επιλογής του (Εικόνα 3.16).




Εικόνα 3.16: Προβολή αποτελέσματος στην οθόνη του χρήστη και εξαγωγή περεταίρω χαρακτηριστικών Ωοειδούς καμπύλης σε αρχείο κειμένου.

Για τον υπολογισμό των χαρακτηριστικών της κάθε καμπύλης χρησιμοποιήθηκαν οι τύποι που αναφέρθηκαν στο κεφάλαιο 2. Παρόλα αυτά παρουσιάζονται και στον πίνακα 3.1 συγκεντρωτικά για διευκόλυνση. Για την Ωοειδή χρησιμοποιήθηκαν οι εξισώσεις 2.36 - 2.39.

Βήμα 2: Καθορισμός απαραίτητων Δεδομένων που εισάγει ο χρήστης (Λειτουργία = Αναλυτικός υπολογισμός)

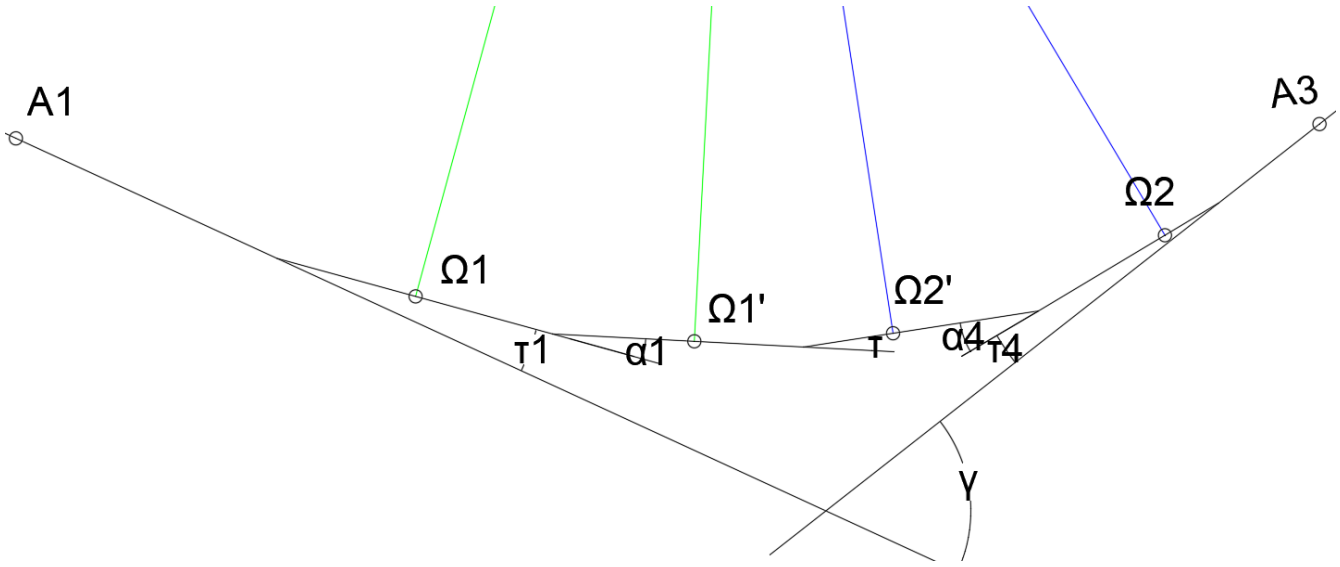
Στην Λειτουργία = Αναλυτικός Υπολογισμός της Ωοειδούς Καμπύλης, πέραν από τις συντεταγμένες της πολυγωνικής χρειαζόμαστε και άλλα δεδομένα. Σίγουρα χρειάζονται τα A1, A2, δηλαδή οι παράμετροι των κλωθοειδών που βρίσκονται πριν και μετά την ενδιάμεση κλωθοειδή-ωοειδή καμπύλη. Επίσης χρειάζεται άλλο ένα δεδομένο το οποίο να μας καθιστά την λύση μονοσήμαντη. Πιθανό δεδομένο εισόδου θα ήταν η απόσταση που έχει το σημείο A1 από το σημείο K1, αλλά ο χρήστης, γενικά ο μηχανικός, δεν είναι εύκολο να γνωρίζει εξ αρχής αυτό το δεδομένο. Άλλο ένα πιθανό δεδομένο εισόδου θα μπορούσε να ήταν οι συντεταγμένες του κέντρου ενός από τους δύο κύκλους. Όμως όπως αναφέρθηκε και στο κεφάλαιο 2 συνήθως τα κέντρα των κύκλων είναι απρόσιτα κατά την σχεδίαση και κατά την πασσάλωση. Έγινε λοιπόν η προσπάθεια χρησιμοποίησης ενός δεδομένου που ξέρει ο χρήστης και που έχει μια συχνή χρήση στην πραγματικότητα. Συγκεκριμένα, ένα συχνό πρόβλημα που έχει να επιλύσει ένας μηχανικός είναι το να υπάρχει ήδη ένα κομμάτι της οριζοντιογραφίας, πχ το κομμάτι εισόδου στην στροφή από A1 έως Ω1΄, και να θέλει αυτό το τμήμα να το «ενώσει» με μία οδό που έχει την διεύθυνση της K2K3. Σε αυτή την περίπτωση λοιπόν ο μηχανικός γνωρίζει ακριβώς το μήκος $b_1$ του πρώτου κυκλικού τμήματος. Για αυτόν τον λόγο λοιπόν στο πρόγραμμα που δημιουργήθηκε χρησιμοποιήθηκε ως επιπρόσθετο δεδομένο εισόδου το μήκος του πρώτου κυκλικού τόξου ($b_1$) (Εικόνα 3.4). Τώρα η λύση της ωοειδούς είναι μονοσήμαντη και μάλιστα ο υπολογισμός του μήκους του δεύτερου κυκλικού τόξου ορίζεται άμεσα και μονοσήμαντα από την σχέση:

$$\alpha_4 = \gamma - \tau_1 - \tau_4 - \tau - \alpha_1 \qquad (3.23)$$

όπου:

$$\alpha_1 = b_1/R_1 \qquad (3.24)$$

$\tau_1$ από εξίσωση (2.7) για A1, R1
$\tau_4$ από εξίσωση (2.7) για A2, R2
$\tau = \tau_3 - \tau_2$ από εξίσωση (2.36) για A, R1, R2




Σχήμα 3.3: Λεπτομέρεια γωνιών Ωοειδούς.

Βήμα 3: Εύρεση διευθύνσεων πολυγωνικής και συντεταγμένες σημείων Α1, Α3 (Λειτουργία = Αναλυτικός υπολογισμός)

Για τον καθορισμό των διευθύνσεων της πολυγωνικής χρησιμοποιήθηκε και πάλι διανυσματική γεωμετρία. Υπολογίζονται και πάλι τα μοναδιαία διανύσματα στις πλευρές Κ1Κ2 και Κ3Κ2 με βάση τις εξισώσεις (3.3-3.10) και τέλος υπολογίζεται η γωνία αλλαγής διεύθυνσης γ σύμφωνα με την σχέση (3.11).

Για την εύρεση των σημείων Α1, Α3 που βρίσκονται πάνω στα ευθύγραμμα τμήματα Κ1Κ2, Κ2Κ3 αρκεί να υπολογιστούν οι αποστάσεις που απέχει το κάθε σημείο από το σημείο Κ2. Ο υπολογισμός αυτός έγινε με την βοήθεια των νόμων ημιτόνων και συνημιτόνων. Παρακάτω παρατίθενται οι τύποι που χρησιμοποιήθηκαν καθώς και τα σχήματα 3.4, 3.5 για την κατανόηση των υπολογισμών:

$$Ti = R_i * \tan\left(\frac{a_i}{2}\right) \quad (2.25)$$

$$T1_{prov} = \sqrt{(TK_1 + T1)^2 + (TL + T1)^2 - 2 * (TK_1 + T1) * (TL + T1) * cos(\pi - \alpha_1)} \quad (2.26)$$

$$T2_{prov} = \sqrt{(TK_4 + T4)^2 + (T\kappa + T4)^2 - 2 * (TK_4 + T4) * (T\kappa + T4) * cos(\pi - \alpha_4)} \quad (2.27)$$

$$\omega_1 = asin\left(\frac{(TL+T1)*\sin(\alpha_1)}{T1_{prov}}\right) \quad (2.28)$$

$$\omega_2 = asin\left(\frac{(TK+T4)*\sin(\alpha_4)}{T2_{prov}}\right) \quad (2.29)$$

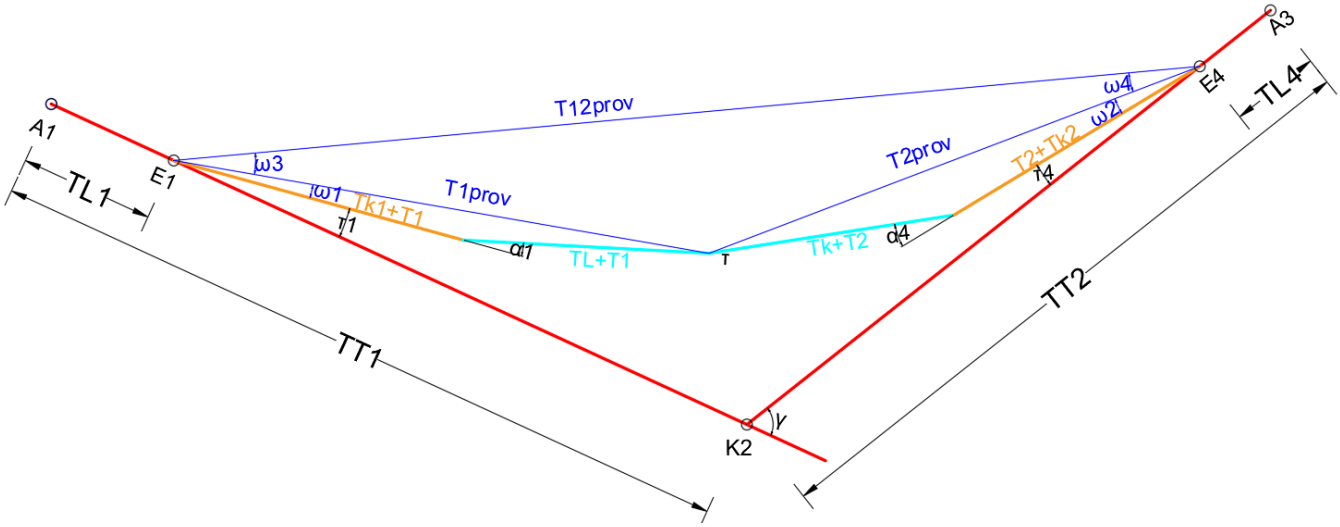
$$T12_{prov} = \sqrt{T1_{prov}{}^2 + T2_{prov}{}^2 - 2 * T1_{prov}T2_{prov} * cos(\pi - \tau - \alpha_1 + \omega_1 - \alpha_4 + \omega_2)} \quad (2.30)$$

$$\omega_3 = asin\left(\frac{T2_{prov}*\sin(\pi-\tau-\alpha_1+\omega_1-\alpha_4+\omega_2)}{T12_{prov}}\right) \quad (2.31)$$

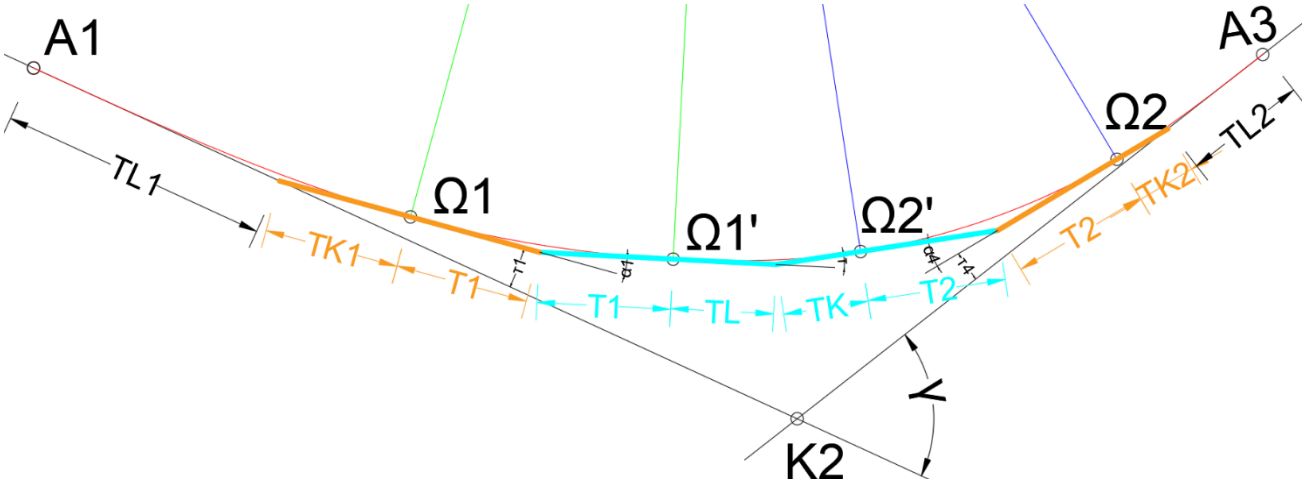
$$\omega_4 = asin\left(\frac{T1_{prov}*\sin(\pi-\tau-\alpha_1+\omega_1-\alpha_4+\omega_2)}{T12_{prov}}\right) \quad (2.32)$$

$$\mathrm{TT1} = \sin(\tau_4 + \omega_2 + \omega_4) * \frac{T12_{prov}}{\sin(\gamma)} + \mathrm{T}_{L1} \quad (2.33)$$

$$\mathrm{TT2} = \sin(\tau_1 + \omega_1 + \omega_3) * \frac{T12_{prov}}{\sin(\gamma)} + \mathrm{T}_{L4} \quad (2.34)$$

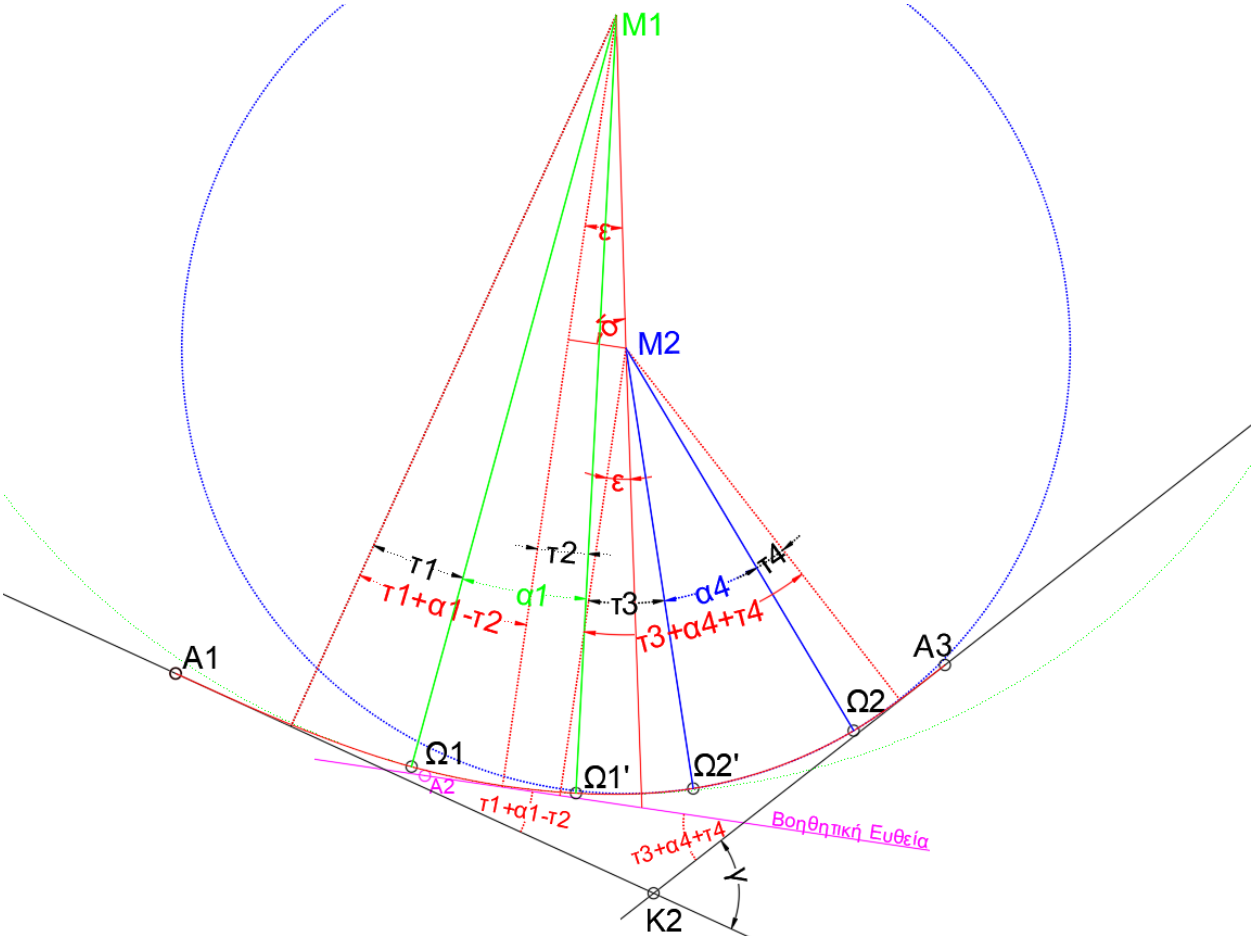



Σχήμα 3.4: Λεπτομέρεια Ωοειδούς για εύρεση απόστασης σημείων Α1, Α3 από σημείο Κ2 [1].




Σχήμα 3.5: Λεπτομέρεια Ωοειδούς για εύρεση απόστασης σημείων A1, A3 από σημείο K2 [2].

Έχοντας τώρα δεδομένα και την διεύθυνση και την απόσταση από το σημείο K2, με διανυσματική γεωμετρία υπολογίζονται οι συντεταγμένες των σημείων A1, A3. Στην συνέχεια παρατίθεται σχέδιο (Σχήμα 3.6) με λεπτομέρειες των γωνιών της Ωοειδούς καμπύλης για επιπλέον κατανόηση της μορφής και της γεωμετρίας της.




Σχήμα 3.6: Λεπτομέρεια γωνιών Ωοειδούς καμπύλης.

Σε αυτό το σημείο είναι πολύ σημαντικό να τονιστεί η χρησιμότητα των δύο εφαπτόμενων ευθύγραμμων τμημάτων που περιβάλλουν την Ωοειδή (TT1 και TT2). Συγκεκριμένα, έχοντας μια πολυγωνική (K1K2K3) που σχηματίζει γωνία γ, καθώς και τη θέση και τις ακτίνες των κυκλικών τόξων, η εύρεση της Ωοειδούς αποτελεί μονοσήμαντη λύση. Όπως έγινε κατανοητό, τα εφαπτόμενα αυτά τμήματα καθορίζουν την αρχή και το τέλος της Ωοειδούς άρα και κατ' επέκταση το αν «χωράει» ή όχι στην παρούσα πολυγωνική χάραξη. Αυτή είναι μια πολύ χρήσιμη πληροφορία για τον μηχανικό μιας και συνήθως ο χώρος που έχει στην διάθεσή του για την χάραξη είναι περιορισμένος. Έτσι λοιπόν συγκρίνοντας αυτά τα μήκη με το διαθέσιμο μήκος, μπορεί να γνωρίζει αμέσως αν με αυτά τα χαρακτηριστικά που επέλεξε (R1, A1, R2, A2, Δ(R1R2), b1) μπορεί να σχεδιαστεί Ωοειδής καμπύλη ή αν πρέπει να αλλάξει κάποια ή κάποιες από τις μεταβλητές.

Βήμα 4: Εύρεση διεύθυνσης Βοηθητικής ευθείας και συντεταγμένες σημείου Α2 (Λειτουργία = Αναλυτικός υπολογισμός)

Η Ωοειδής καμπύλη, όπως έγινε κατανοητό από το κεφάλαιο 2.4.3, αποτελείται από ένα μέρος κλωθοειδούς το οποίο ξεκινάει με ακτίνα R1 και τελειώνει με ακτίνα R2. Για να μπορέσει να σχεδιαστεί αυτό το τμήμα πρέπει να βρεθεί το αρχικό σημείο της κλωθοειδούς αυτής καθώς και η διεύθυνση της, πρέπει δηλαδή να βρεθεί το σημείο Α2 και η διεύθυνση της βοηθητικής ευθείας. Το χαρακτηριστικό της ευθείας αυτής είναι ότι είναι κάθετη ταυτόχρονα και στα δύο ευθύγραμμα τμήματα M1E2, M2E3 (Σχήμα 3.7).

Επίσης είναι γνωστά τα μήκη αυτών των ευθύγραμμων τμημάτων και ίσα με:

$$(M1E2) = Y_{M2}\,,\ (M2E3) = Y_{M3} \qquad (3.23)$$

Από την κατασκευή της Ωοειδούς και το σχήμα (3.5) προκύπτουν οι εξής σχέσεις:

$$\Delta X_M = X_{M3} - X_{M2} \qquad (3.24)$$

$$\Delta Y_M = Y_{M2} - Y_{M1} = (R_2 + \Delta R_2) - (R_1 + \Delta R_1) \qquad (3.25)$$

$$\tan(\varepsilon) = \frac{\Delta X_M}{\Delta Y_M} \qquad (3.26)$$

$$(M1M2) = \sqrt{\Delta X_M^2 + \Delta Y_M^2} \qquad (3.27)$$

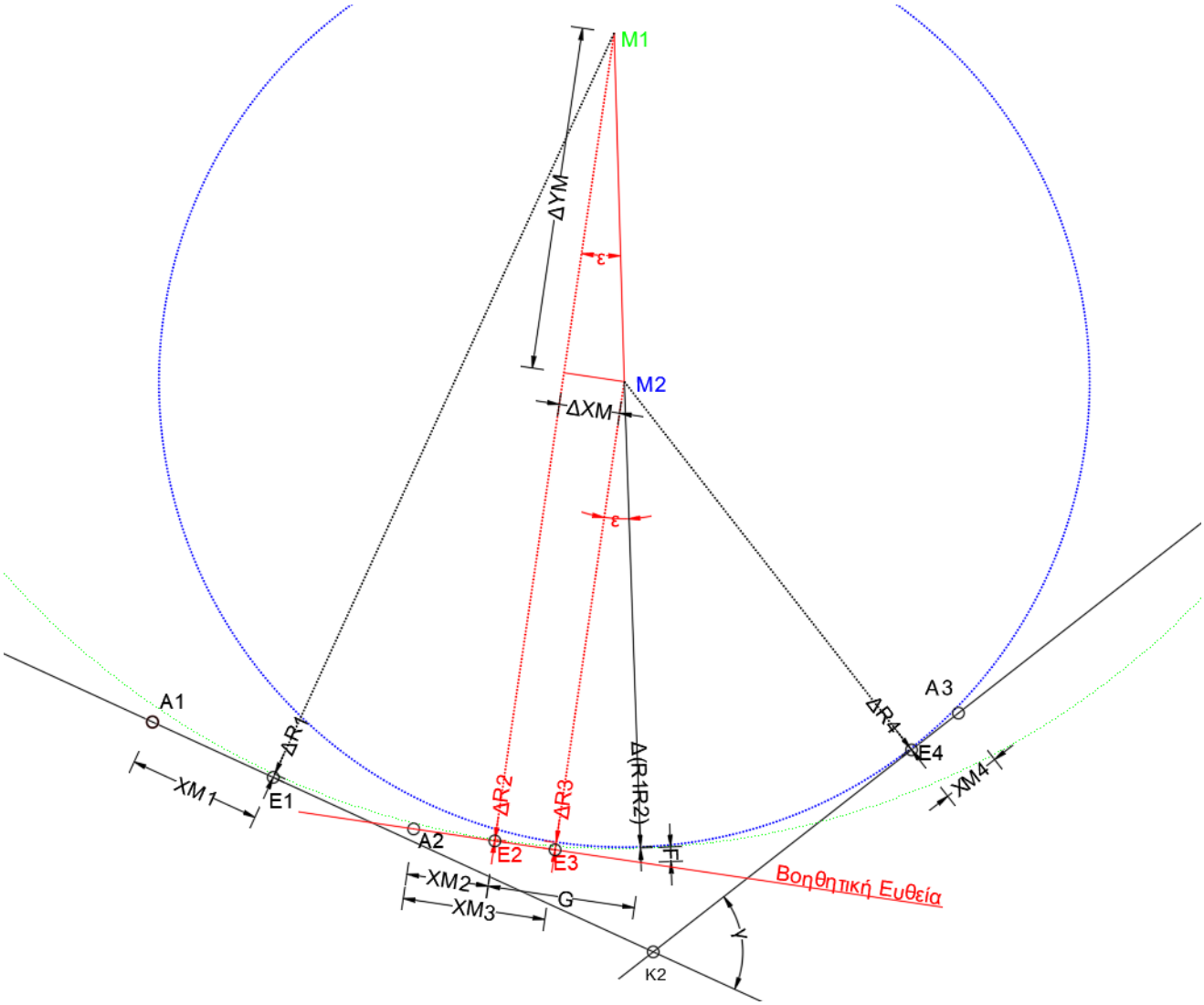
$$\Delta(R_1R_2) = R_1 - R_2 - M \qquad (3.28)$$

$$F = \frac{Y_{M3}}{\cos(\varepsilon)} - \left(R_2 + \Delta(R_1R_2)\right) \qquad (3.29)$$

$$G = Y_{M2} * \tan(\varepsilon) \qquad (3.30)$$

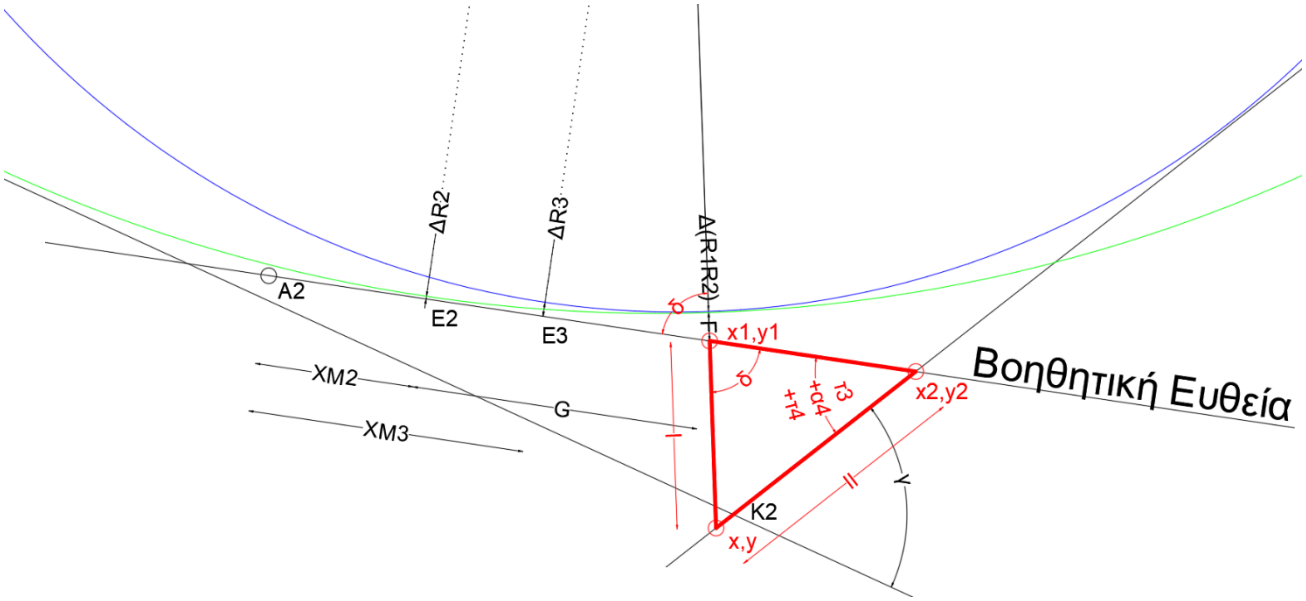
Όπως φαίνεται λοιπόν και από το σχήμα, για να μπορέσει να βρεθεί η βοηθητική ευθεία πρέπει για αρχή να βρεθούν οι θέσεις των κέντρων των κύκλων M1, M2.Για τον υπολογισμό των συντεταγμένων των κύκλων, μιας και από το βήμα 2 είναι πλέον γνωστές οι συντεταγμένες των A1, A3 μπορούμε να βρούμε τα σημεία E1, E4 με την βοήθεια των $\vec{t}_{21}, \vec{t}_{23}$ και έπειτα μπορούμε να υπολογίσουμε και τις συντεταγμένες των M1, M2 με την βοήθεια των $\vec{n}_{21},\ \vec{n}_{23}$. Με την χρήση της εξίσωσης (3.26) υπολογίζεται το ε και από τις συντεταγμένες των M1 M2 η διεύθυνση και φορά της ευθείας που ορίζουν. Ανάλογα με το αν η στροφή είναι δεξιόστροφη ή αριστερόστροφη, το ε προστίθεται ή αφαιρείται αντίστοιχα από την κλίση της ευθείας που σχηματίζουν τα M1M2. Στο παράδειγμα που δίνεται στο σχήμα 3.5, η στροφή είναι δεξιόστροφη άρα το ε προστίθεται. Γνωρίζοντας τώρα την διεύθυνση (κοινή) των δύο ευθύγραμμων τμημάτων (M1E2) και (M2E3), τις συντεταγμένες των M1, M2 και την απόσταση αυτών από την βοηθητική ευθεία βρίσκουμε τις συντεταγμένες των σημείων E2, E3. Πλέον ορίζεται η

βοηθητική ευθεία εφόσον είναι γνωστά δύο σημεία που την επαληθεύουν. Τέλος οι συντεταγμένες του Α2 βρίσκονται όπως και προηγουμένως με διανυσματική γεωμετρία.




Σχήμα 3.7: Λεπτομέρεια Βοηθητικής Ευθείας Ωοειδούς Καμπύλης.

Παρόλα αυτά, για να αποφευχθεί η ιδιαιτερότητα την πρόσθεσης ή αφαίρεσης της γωνίας ε ανάλογα με την φορά της καμπύλης, η διεύθυνση της βοηθητικής ευθείας καθώς και το σημείο Α2 μπορεί να βρεθεί και με άλλους τρόπους. Στο σχήμα 3.8 παρουσιάζεται η λεπτομέρεια του υπολογισμού που χρησιμοποιήθηκε. Συγκεκριμένα βρέθηκε η διεύθυνση του ευθύγραμμου τμήματος M1M2 ( $\overrightarrow{t_{M1M2}}$ ) και ύστερα υπολογίστηκε το σημείο με συντεταγμένες (x1,y1) το οποίο απέχει από το Μ2 απόσταση R2+Δ(R1R2)+F. Ύστερα βρέθηκε το κοινό σημείο (x,y) της ευθείας που σχηματίζουν τα M1M2 και της ευθείας K2K3. Υπολογίστηκε η απόσταση των δύο αυτών σημείων ( l ) και έπειτα με τον νόμο των ημιτόνων βρέθηκε η απόσταση ( ll ). Έπειτα βρέθηκε το σημείο (x2,y2) που απέχει από το Κ2 απόσταση ( ll ). Τέλος με γνωστά δύο σημεία που ορίζουν την βοηθητική ευθεία βρέθηκε η διεύθυνσή της και υπολογίστηκε η θέση του Α2 σε απόσταση $G+X_{M2}$ από το (x1,y1).



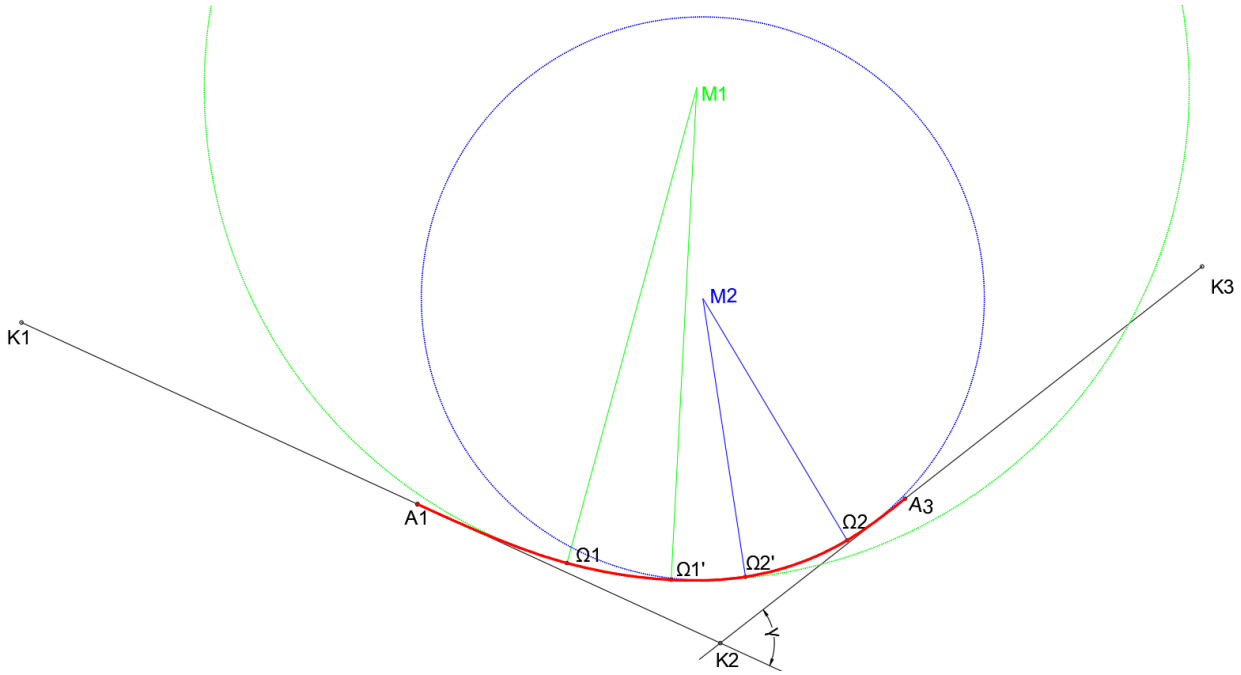

Σχήμα 3.8: Λεπτομέρεια Ωοειδούς Καμπύλης για την εύρεση της Βοηθητικής Ευθείας.

Βήμα 5: Υπολογισμός Συντεταγμένων Άξονα οδού (Λειτουργία = Αναλυτικός υπολογισμός)
Για την εύρεση των συντεταγμένων του άξονα οδού χρησιμοποιείται όπως και πριν διανυσματική γεωμετρία. Σημαντικό είναι για το τμήμα της δεύτερης κλωθοειδούς (Α2-Ω2΄) να χρησιμοποιηθεί μόνο το τμήμα της που αφορά ακτίνες R1 μέχρι R2. Δηλαδή κατά τον υπολογισμό των σημείων της κλωθοειδούς με βάση τις εξισώσεις 3.15, 3.16 το s παίρνει τιμές από $l_2$ εώς $l_3$. Στο σχήμα 3.8 φαίνεται η τελική μορφή της Ωοειδούς καμπύλης καθώς και τα βασικά σημεία της.

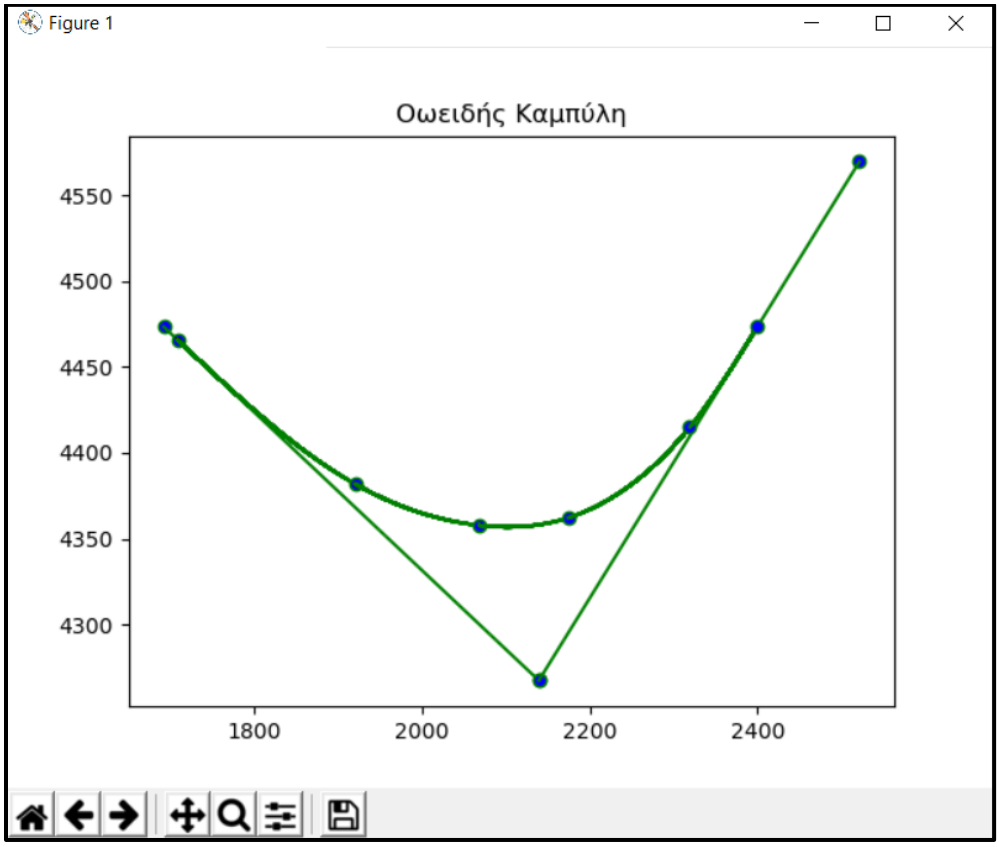
Βήμα 6: Εξαγωγή Αποτελεσμάτων (Λειτουργία = Αναλυτικός υπολογισμός)
Τα αποτελέσματα αντίστοιχα με πριν αποθηκεύονται, αν θέλει ο χρήστης σε αρχεία με ονομασία που επιλέγει ο ίδιος. Συγκεκριμένα μπορούν να παραχθούν τα εξής αρχεία (Εικόνα 3.18):

- <πρόθεμα>.ova, το οποίο περιλαμβάνει τα βασικά χαρακτηριστικά της Ωοειδούς Καμπύλης
- <πρόθεμα>.txt , το οποίο περιλαμβάνει τις συντεταγμένες του άξονα της Ωοειδούς Καμπύλης
- <πρόθεμα>.dxf , το οποίο περιλαμβάνει το σχέδιο με τον άξονα της Ωοειδούς Καμπύλης και τα βασικά της σημεία.

Είτε αποθηκεύσει τα αρχεία του είτε όχι, στο τέλος του προγράμματος εμφανίζεται στην οθόνη του χρήστη ένα γράφημα (Plot) με το σχέδιο της Πολυγωνικής, της Ωοειδούς καμπύλης καθώς και τα βασικά της σημεία (Εικόνα 3.17).




Σχήμα 3.9: Ωοειδής Καμπύλη και τα χαρακτηριστικά της σημεία.




Εικόνα 3.17: Εξαγωγή σκαριφήματος Ωοειδούς καμπύλης στην οθόνη του χρήστη.

παραδειγμ...

File Edit Format View Help

```
Coordinates of Oval Curve
K1 = 1691.7981, 4473.2589
A1 = 1708.2976, 4465.6696
P1 = 1709.2061, 4465.2517
P2 = 1710.1146, 4464.8338
P3 = 1711.0231, 4464.416
P4 = 1711.9316, 4463.9981
P5 = 1712.8401, 4463.5803
P6 = 1713.7487, 4463.1625
P7 = 1714.6572, 4462.7447
P8 = 1715.5658, 4462.327
P9 = 1716.4744, 4461.9093
P10 = 1717.383, 4461.4917
P11 = 1718.2917, 4461.0741
P12 = 1719.2003, 4460.6566
P13 = 1720.109, 4460.2392
P14 = 1721.0178, 4459.8218
P15 = 1721.9266, 4459.4045
P16 = 1722.8354, 4458.9873
P17 = 1723.7442, 4458.5702
P18 = 1724.6531, 4458.1532
P19 = 1725.5621, 4457.7363
P20 = 1726.4711, 4457.3195
```

Windows (CRLF) ANSI

*παραδειγμ...

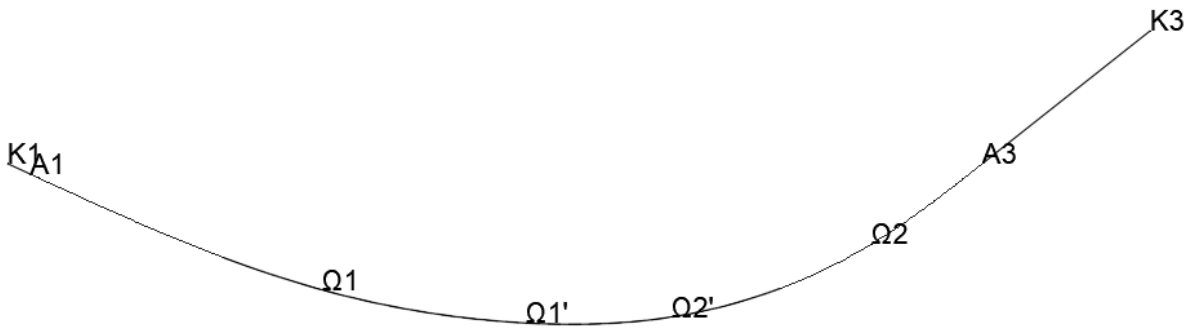
File Edit Format View He

```
Curve 1
 R1=700.0
 A1=400.0
 L1=228.5714
 τ1=10.3938 grad
 X1=227.9629
 Y1=12.4156
 TL1=152.5943
 TK1=76.3844
 Xm1=114.1842
 Ym1=703.1069
 DR1=3.1069
 T1=75.2883
 α1=0.2143 rad
 b1=150.0

Oval Curve
 TT1= 474.3542
 TT2 = 332.5372
 γ = 70.0

Curve 2
 R2=700.0
```

Windows (CRLF) ANSI

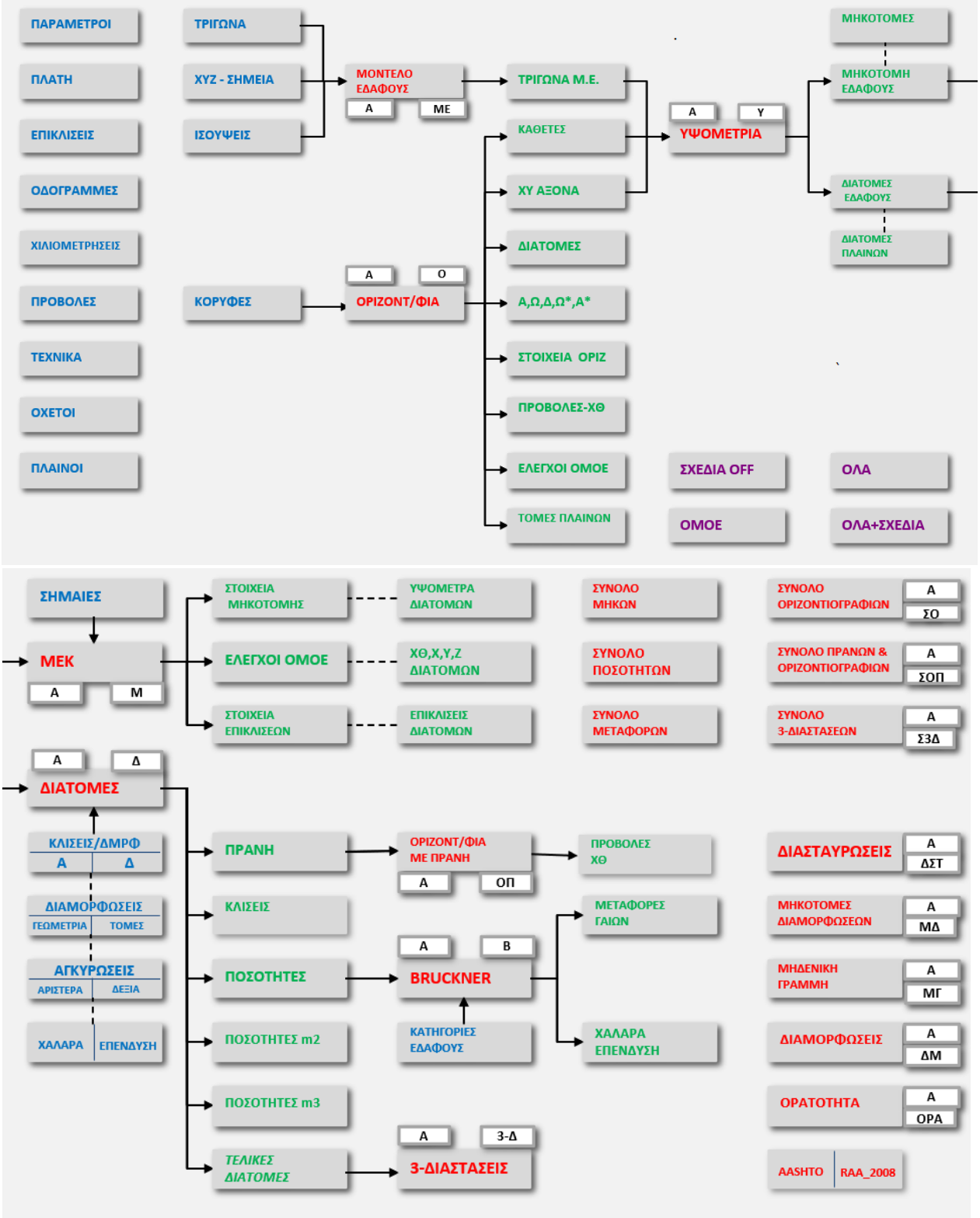



Εικόνα 3.18: Εξαγωγή αποτελεσμάτων Ωοειδούς καμπύλης σε αρχεία κειμένου και αρχείο CAD.

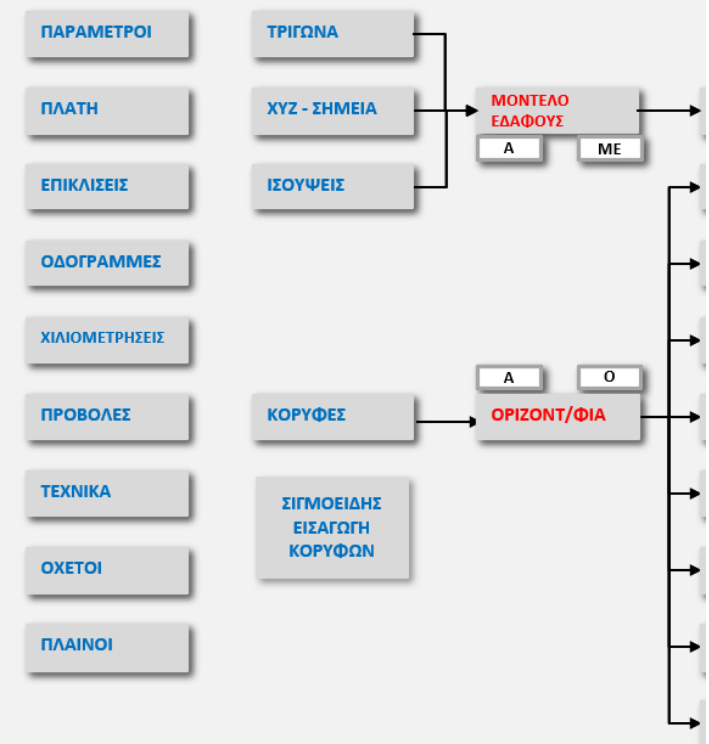
## 3.4 ΣΥΝΔΕΣΗ ΜΕ ΤΟ ΛΟΓΙΣΜΙΚΟ FM19

Το FM19 είναι ένα λογισμικό το οποίο έχει αναπτυχθεί από τον Φ. Μερτζάνη, για εκπαιδευτική χρήση στο πλαίσιο των μαθημάτων Οδοποιίας που διδάσκονται στο Εθνικό Μετσόβιο Πολυτεχνείο. Οι χρήστες του λογισμικού, έχοντας βασικές γνώσεις οδοποιίας και λειτουργίας ηλεκτρονικού υπολογιστή, μπορούν να μελετήσουν μια οδό και να παραγάγουν τα βασικά σχέδιά της.
Το λογισμικό λειτουργεί στο περιβάλλον του προγράμματος Microsoft Excel, στα φύλλα του οποίου εισάγονται οι παράμετροι σχεδιασμού της υπό μελέτη οδού (Εικόνα 3.19). Στη συνέχεια, εκτελούνται τα επιμέρους προγράμματα και εξάγονται τα αποτελέσματα (υπολογιζόμενα στοιχεία της οδού) και τα τελικά σχέδια της μελέτης. Τα αποτελέσματα διαβάζονται μέσω της εφαρμογής Notepad (Σημειωματάριο)και τα σχέδια της οδού μπορούν να ανοιχτούν από διάφορα λογισμικά σχεδίασης, όπως είναι το AutoCAD της Autodesk.
Σε αντίθεση με τα προγράμματα που αναλύθηκαν στο κεφάλαιο 3.5, που παράγουν μόνο στοιχεία οριζοντιογραφίας, το FM19 κάνει ολοκληρωμένη Μελέτη Χάραξης. Για αυτόν τον λόγο θεωρήθηκε σκόπιμο να γίνει σύνδεση των δύο προγραμμάτων. Συγκεκριμένα τα προγράμματα του κεφαλαίου 3.3 παράγουν ένα επιπλέον αρχείο το οποίο έπειτα χρησιμοποιεί το FM19 για να παράξει και τα υπόλοιπα κομμάτια της χάραξης.
Για το κομμάτι της Σιγμοειδούς Καμπύλης η διαδικασία σύνδεσης είναι αρκετά εύκολη μιας και μπορεί να θεωρηθεί δύο συνεχόμενες καμπύλες με συγκεκριμένα χαρακτηριστικά. Η σύνδεση έγινε μέσω ενός κουμπιού στο ήδη υπάρχον FM19 το οποίο εισάγει αυτά τα δεδομένα κατευθείαν στο σύστημα και έπειτα ο χρήστης μπορεί να κάνει πλήρη χρήση του FM19 (Εικόνα 3.25). Κάποιες από τις Βασικές Δυνατότητες- Προγράμματα Υπολογισμών είναι οι εξής:

- Μοντέλο Εδάφους: Δημιουργείται το ψηφιακό μοντέλο του εδάφους της περιοχής μελέτης.
- Οριζοντιογραφία: Δημιουργείται η οριζοντιογραφία της οδού.
- Υψομετρία: Λαμβάνονται τα υψόμετρα του εδάφους κατά μήκος του άξονα της οδού, καθώς και εγκάρσια του άξονα της οδού, στις θέσεις των διατομών.
- ΜΕΚ: Δημιουργείται η μηκοτομή της οδού και το διάγραμμα επικλίσεων, και πραγματοποιείται ο έλεγχος της χάραξης σύμφωνα με τις οδηγίες μελέτης οδών που έχουν επιλεγεί.
- Διατομές: Παράγονται οι διατομές της οδού.
- BRUCKNER: Κατασκευάζεται το διάγραμμα Bruckner(διάγραμμα κίνησης εκχωμάτων).
- Οριζοντιογραφία με πρανή: Παράγεται η τελική οριζοντιογραφία, η οποία περιλαμβάνει τις οριογραμμές του οδοστρώματος, τις διαμορφώσεις και τα πρανή της οδού.
- 3-Διαστάσεις: Δημιουργείται το τρισδιάστατο προσομοίωμα (μοντέλο) της οδού και του περιβάλλοντα χώρου, με χρήση της μεθόδου του τριγωνισμού.




Εικόνα 3.19: Μορφή κεντρικού μενού λογισμικού FM19 (καρτέλα MENU).




Εικόνα 3.20: Σύνδεση με το λογισμικό FM19 με την χρήση ενός κουμπιού.

Για το κομμάτι της Ωοειδούς Καμπύλης η διαδικασία σύνδεσης δεν είναι εφικτή μιας και το FM19 είναι προγραμματισμένο να σχεδιάζει απλές καμπύλες της μορφής Κλωθοειδής-Κυκλικό Τόξο-Κλωθοειδής και όχι Κλωθοειδής-Κυκλικό Τόξο-Ωοειδής-Κυκλικό Τόξο-Κλωθοειδής.

# *ΣΥΣΧΕΤΙΣΗ ΜΕ ΑΛΛΑ ΛΟΓΙΣΜΙΚΑ*

## 4.1 ΕΙΣΑΓΩΓΗ

Η χάραξη των οδών αποτελεί μια αρκετά πολύπλοκη και χρονοβόρα διαδικασία που απαιτεί προσοχή στη λεπτομέρεια, ακρίβεια, γνώση του αντικειμένου και των κανόνων εις βάθος και σχολαστικότητα. Ωστόσο τα περιθώρια ανθρώπινου λάθους είναι πολλά, ειδικά σε έργα μεγάλου μεγέθους και για αυτό κρίνεται αναγκαία η χρήση προγραμμάτων που αυτοματοποιούν την διαδικασία αυτή. Στην αγορά υπάρχουν διάφορα τέτοια σχεδιαστικά προγράμματα όπως το Civil 3D από την Autodesk, Inroads από την Bentley, το FM19 του κύριου Μερτζάνη κ.α.

Στο παρόν κεφάλαιο θα παρουσιαστεί ένα παράδειγμα Σιγμοειδούς Καμπύλης το οποίο θα σχεδιαστεί με την βοήθεια:

1. του προγράμματος S-Curve Calculator που αναλύθηκε στο Κεφάλαιο 3,
2. του προγράμματος Civil 3D,
3. του προγράμματος FM19

Επίσης θα παρουσιαστεί ένα παράδειγμα Ωοειδούς Καμπύλης το οποίο θα σχεδιαστεί με την βοήθεια:

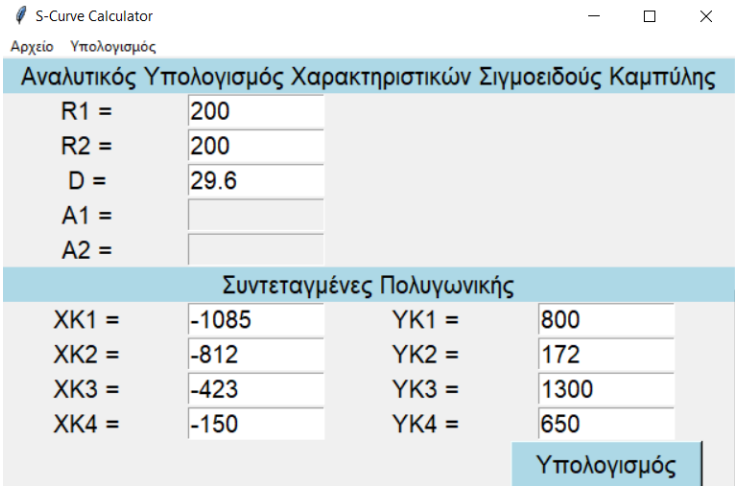
1. του προγράμματος που Oval-Curve Calculator αναλύθηκε στο Κεφάλαιο 3,
2. του προγράμματος Civil 3D.

Σκοπός αυτού του κεφαλαίου είναι για αρχή να συγκριθούν τα αποτελέσματα των προγραμμάτων μεταξύ τους ώστε να διασταυρωθεί η ορθότητα και να αξιολογηθεί η επίδοση των νέων προγραμμάτων που αναλύθηκαν στο κεφάλαιο 3. Πολύ σημαντικό είναι να εντοπιστούν οι ομοιότητες και οι διαφορές που υπάρχουν μεταξύ των σχεδιαστικών προγραμμάτων. Με αυτόν τον τρόπο γίνεται κατανόηση σε βάθος αυτών των μορφών καμπύλης και τονίζονται τα προτερήματα και τα μειονεκτήματα του κάθε προγράμματος.

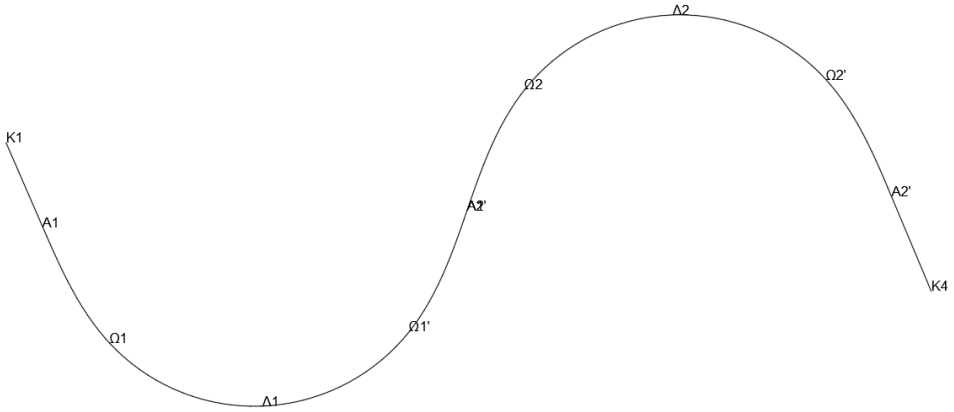
## 4.2 ΠΑΡΑΔΕΙΓΜΑ ΣΙΓΜΟΕΙΔΟΥΣ ΚΑΜΠΥΛΗΣ

### 4.2.1 Επίλυση με την βοήθεια του προγράμματος S-Curve Calculator

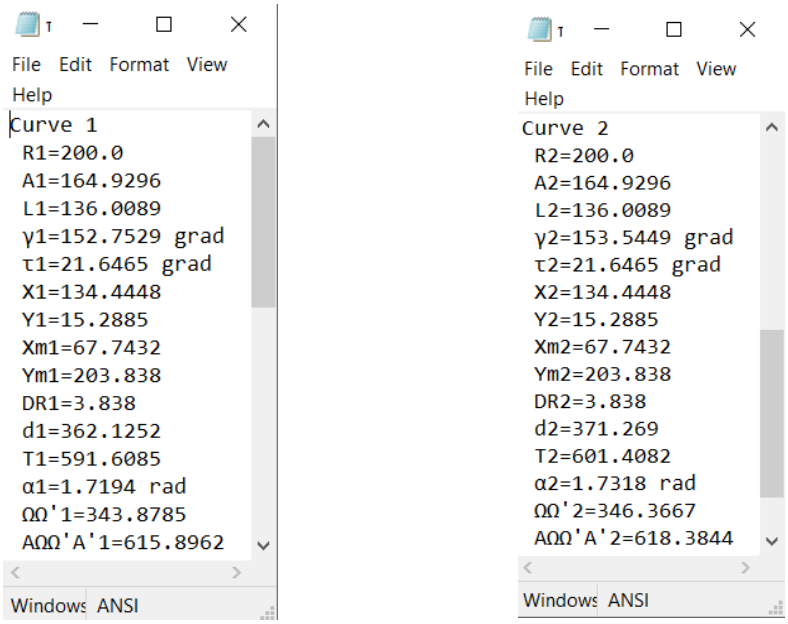
Όπως αναφέρθηκε στα προηγούμενα κεφάλαια, το πιο βασικό χαρακτηριστικό της Σιγμοειδούς Καμπύλης είναι η περιφερειακή απόσταση D, δηλαδή η απόσταση μεταξύ των κέντρων των κύκλων πλην την τιμή των δύο ακτινών. Αυτή η απόσταση καθορίζει τόσο τις τιμές των παραμέτρων A όσο και γενικά την Σιγμοειδή Καμπύλη. Για αυτόν τον λόγο και δημιουργήθηκε το πρόγραμμα S-Curve Calculator, το οποίο λαμβάνει ως δεδομένο εισόδου αυτήν ακριβώς την απόσταση, η οποία πρακτικά αποτελεί και το βασικό στοιχείο που έχει ένας μελετητής.
Στην εικόνα 4.1 παρουσιάζονται τα δεδομένα εισόδου του παραδείγματος, στην εικόνα 4.2 παρουσιάζεται το σχέδιο της οριζοντιογραφίας και τέλος στην εικόνα 4.3 τα αποτελέσματα.




Εικόνα 4.1: S-Curve Calculator – Εισαγωγή δεδομένων.




Εικόνα 4.2: S-Curve Calculator - Σχέδιο παραδείγματος Οριζοντιογραφίας.




Εικόνα 4.3: S-Curve Calculator - Αποτελέσματα Παραδείγματος Οριζοντιογραφίας.

Τα προγράμματα που υπάρχουν στην αγορά δεν έχουν την δυνατότητα να σχεδιάσουν μια Σιγμοειδή Καμπύλη. Παρόλα αυτά, αν θεωρηθεί ως δυο ξεχωριστές καμπύλες είναι δυνατό να σχεδιαστεί. Για να γίνει αυτό θα χρειαστούν κάποια δεδομένα εισόδου, τα οποία λαμβάνονται από τα στοιχεία εξόδου που παρέχει το Scalculator. Παρακάτω παρουσιάζεται ο πίνακας ο οποίος περιλαμβάνει όλα τα πιθανά δεδομένα εισόδου που θα χρειαστούν τα επόμενα σχεδιαστικά προγράμματα με βάση το παράδειγμα που επιλύθηκε στο κεφάλαιο 4.2.1.

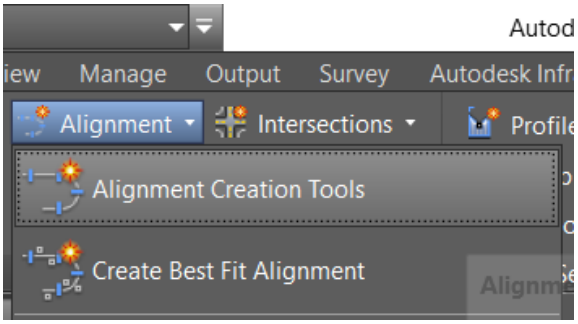
| Συντεταγμένες Πολυγωνικής | | | |
|---|---|---|---|
| Xk1 | -1085 | Yk1 | 800 |
| Xk2 | -812 | Yk2 | 172 |
| Xk3 | -423 | Yk3 | 1300 |
| Xk4 | -150 | Yk4 | 650 |

| Δεδομένα Πρώτης Καμπύλης | | | | Δεδομένα Δεύτερης Καμπύλης | | | |
|---|---|---|---|---|---|---|---|
| Δεδομένα Κλωθοειδούς εισόδου | | | | Δεδομένα Κλωθοειδούς εισόδου | | | |
| L1 | 136.0089 | A1 | 164.9296 | L2 | 136.0089 | A2 | 164.9296 |
| Δεδομένα Κυκλικού Τόξου | | | | Δεδομένα Κυκλικού Τόξου | | | |
| | R1 | 200 | | | R2 | 200 | |
| Δεδομένα Κλωθοειδούς εισόδου | | | | Δεδομένα Κλωθοειδούς εισόδου | | | |
| L1 | 136.0089 | A1 | 164.9296 | L2 | 136.0089 | A2 | 164.9296 |

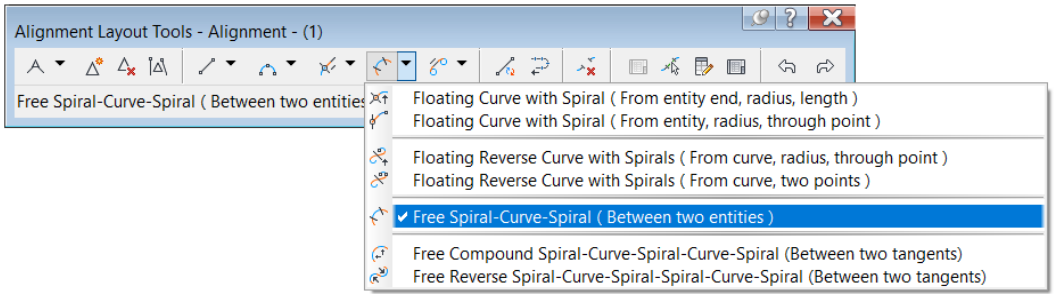
Πίνακας 4.1: Δεδομένα Παραδείγματος Σιγμοειδούς Καμπύλης.

### 4.2.2 Επίλυση με την βοήθεια του προγράμματος Civil 3D

Ο σχεδιασμός μίας καμπύλης δρόμου μέσω του προγράμματος Civil 3D γίνεται με την δημιουργία Alignment (Εικόνα 4.4). Εφόσον σχεδιαστεί η πολυγωνική με βάση τις συντεταγμένες του παραδείγματος, μέσω του Alignment Creation Tools πρέπει να δημιουργηθεί μία αλληλουχία κλωθοειδής εισόδου, κυκλικό τόξο, κλωθοειδής εξόδου (SCS) για την κάθε καμπύλη (Εικόνα 4.5). Τονίζεται ότι οι δύο Καμπύλες θεωρούνται ξεχωριστές και όχι ενιαίες με μορφή Σιγμοειδούς Καμπύλης.



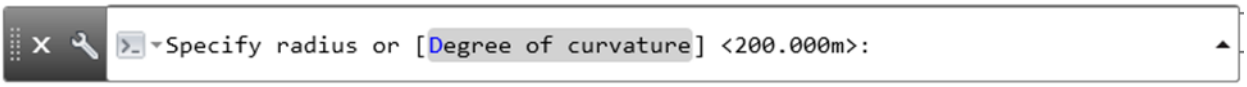

Εικόνα 4.4: Civil 3D - Δημιουργία Οδού.




Εικόνα 4.5: Civil 3D - Δημιουργία απλής Καμπύλης.

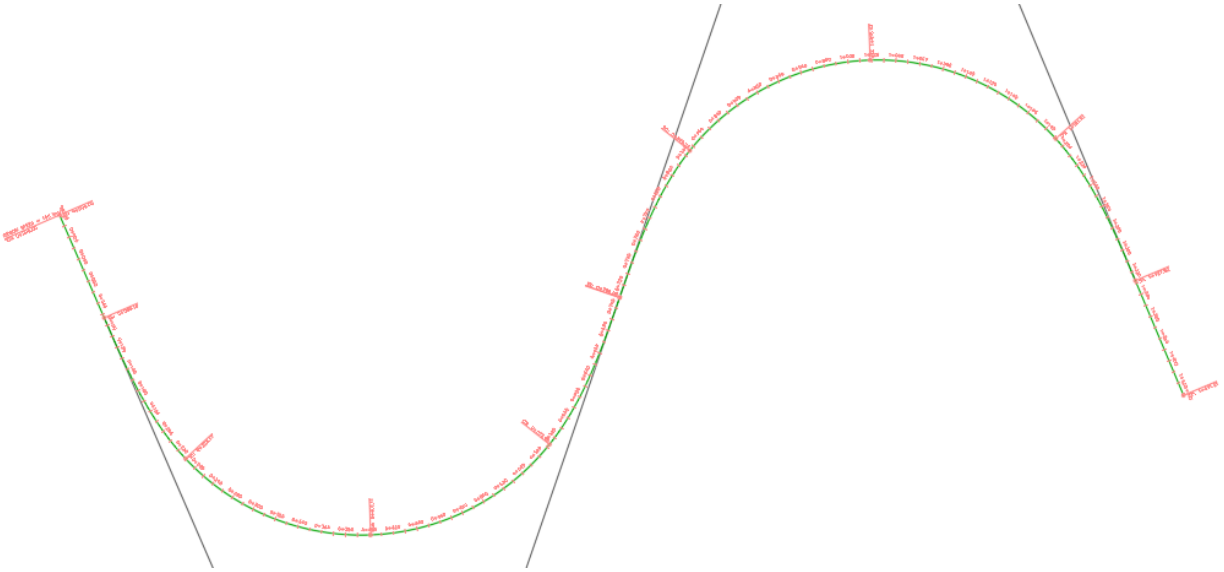
Επόμενο βήμα είναι ο προσδιορισμός των χαρακτηριστικών της καμπύλης που θέλουμε να σχεδιάσουμε. Από το Civil 3D ζητούνται τα εξής δεδομένα εισόδου-παράμετροι:

- Επιλογή των εφαπτόμενων της καμπύλης που θα σχεδιαστεί
- Προσδιορισμός ακτίνας ή καμπυλότητα (degree of curvature-Εικόνα 4.6)
- Προσδιορισμός παραμέτρου Α ή μήκος κλωθοειδούς εισόδου
- Προσδιορισμός παραμέτρου Α ή μήκος κλωθοειδούς εξόδου




Εικόνα 4.6: Civil 3D - Παράδειγμα τρόπου εισαγωγής δεδομένων.

Έπειτα σχεδιάζεται η καμπύλη και ο χρήστης μπορεί να επιλέξει αν θέλει τον προσδιορισμό της χιλιομετρικής θέσης στο σχέδιο καθώς και των βασικών σημείων της καμπύλης ως σημαίες. Το παράδειγμα της οριζοντιογραφίας δίνεται στην εικόνα 4.7.



Εικόνα 4.7: Civil 3D - Σχέδιο παραδείγματος Οριζοντιογραφίας.

Ο χρήστης έχει πρόσβαση σε δεδομένα εξόδου και λεπτομέρειες σχεδίου μέσω της επιλογής του Alignment Grid View. Στην συνέχεια παρουσιάζονται τα αποτελέσματα που προέκυψαν από το παραπάνω παράδειγμα (Εικόνα 4.8).

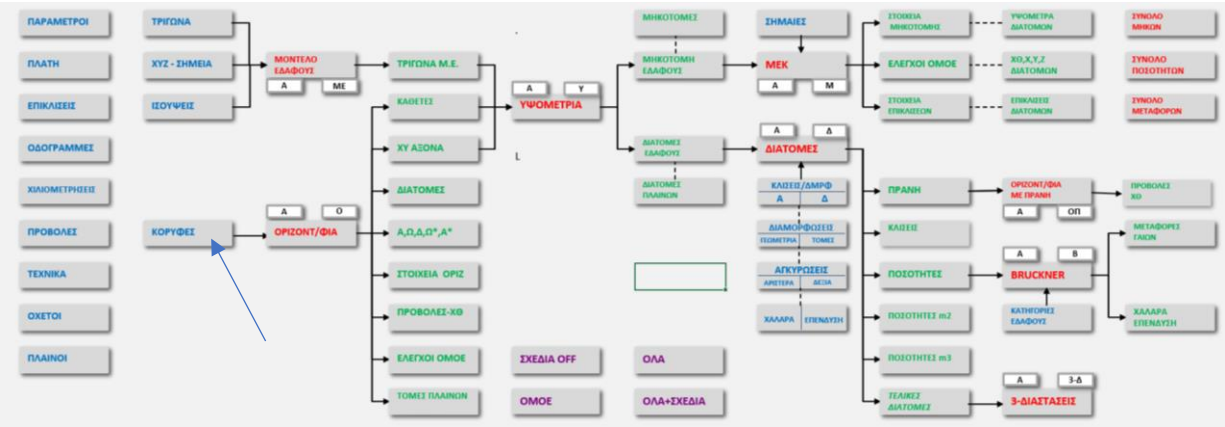
| No. | Type | Param... | Length | Radius | Direction | Start Station | End Station |
|---|---|---|---|---|---|---|---|
| 1 | Line | Two p... | 93.164m | | S23° 29' 43"E | 0+000.00m | 0+093.16m |
| 2.1 | Spiral-Cur... | Sp1 A ... | 136.009m | | | 0+093.16m | 0+229.17m |
| 2.2 | Spiral-Cur... | Sp1 A ... | 343.878m | 200.000m | | 0+229.17m | 0+573.05m |
| 2.3 | Spiral-Cur... | Sp1 A ... | 136.010m | | | 0+573.05m | 0+709.06m |
| 3 | Line | Two p... | 0.174m | | N19° 01' 38"E | 0+709.06m | 0+709.23m |
| 4.1 | Spiral-Cur... | Sp1 A ... | 136.009m | | | 0+709.23m | 0+845.24m |
| 4.2 | Spiral-Cur... | Sp1 A ... | 346.367m | 200.000m | | 0+845.24m | 1+191.61m |
| 4.3 | Spiral-Cur... | Sp1 A ... | 136.009m | | | 1+191.61m | 1+327.62m |
| 5 | Line | Two p... | 103.595m | | S22° 46' 57"E | 1+327.62m | 1+431.21m |

| No. | Delta angle | Chord length | Start Point | Center Point | A | Degree of Curv... |
|---|---|---|---|---|---|---|
| 1 | | | (-1085.0000m,800.0000... | | | |
| 2.1 | 19.4818 (d) | | (-1047.8582m,714.5602... | | 164.930m | |
| 2.2 | 98.5138 (d) | 303.058m | (-980.2377m,597.3569m... | (-833.9124m,733.6980m... | | 8.7319 (d) |
| 2.3 | 19.4819 (d) | | (-677.4105m,609.1702m... | | 164.930m | |
| 3 | | | (-619.1258m,731.2857m... | | | |
| 4.1 | 19.4818 (d) | | (-619.0690m,731.4503m... | | 164.930m | |
| 4.2 | 99.2268 (d) | 304.676m | (-560.7846m,853.5653m... | (-404.2825m,729.0377m... | | 8.7319 (d) |
| 4.3 | 19.4818 (d) | | (-256.2723m,863.5479m... | | 164.930m | |
| 5 | | | (-190.1152m,745.5124m... | | | |

Εικόνα 4.8: Civil 3D - Αποτελέσματα Παραδείγματος Οριζοντιογραφίας.

### 4.2.3 Επίλυση με την βοήθεια του προγράμματος FM19

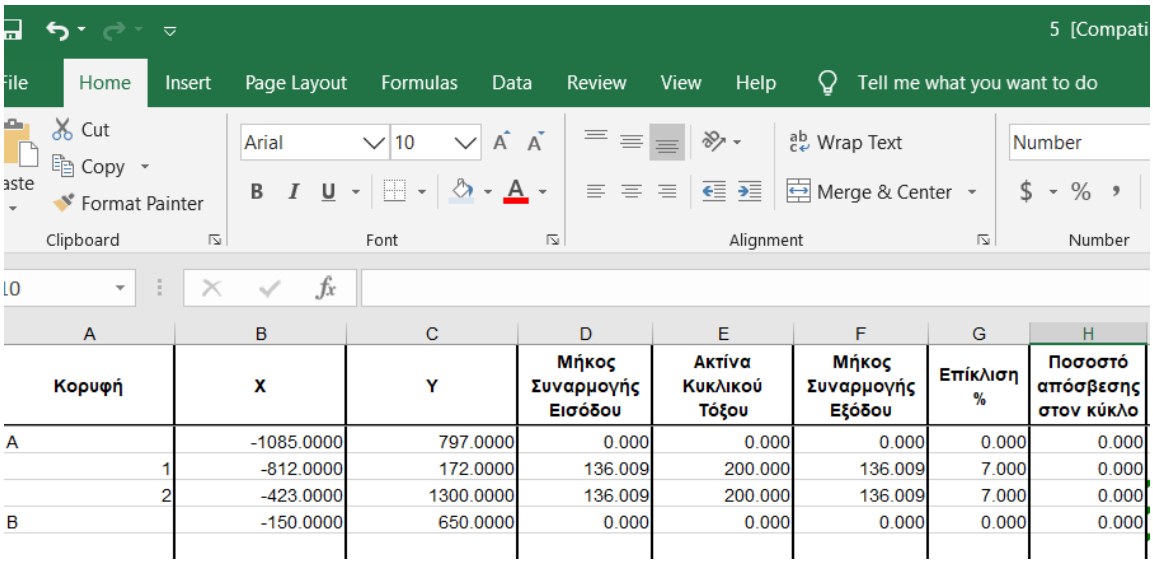
Ο σχεδιασμός μίας καμπύλης δρόμου μέσω του προγράμματος FM19 γίνεται από το κουμπί ΚΟΡΥΦΕΣ (Εικόνα 4.9). Τονίζεται ότι και εδώ, οι δύο Καμπύλες θεωρούνται ξεχωριστές.




Εικόνα 4.9: FM19 – Εισαγωγή δεδομένων από το κουμπί ΚΟΡΥΦΕΣ.

Επόμενο βήμα είναι ο προσδιορισμός των χαρακτηριστικών της καμπύλης που θέλουμε να σχεδιάσουμε. Από το FM19 ζητούνται τα εξής δεδομένα εισόδου-παράμετροι (Εικόνα 4.10):

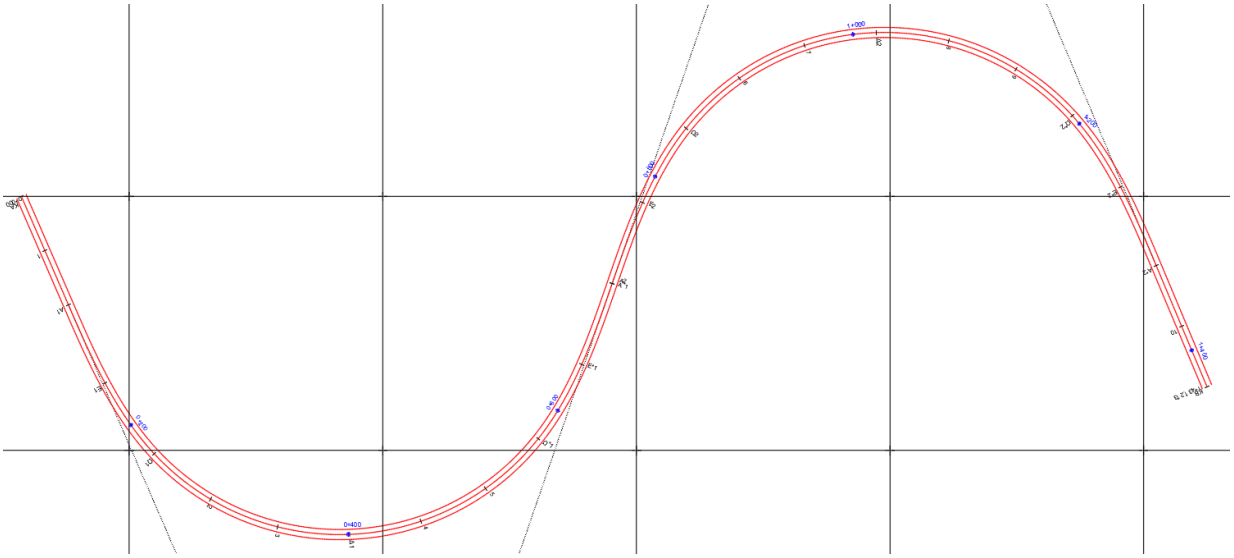
- Συντεταγμένες Πολυγωνικής
- Μήκος Συναρμογής Εισόδου
- Ακτίνα Κυκλικού Τόξου
- Μήκος Συναρμογής Εξόδου




| A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|
| **Κορυφή** | **X** | **Y** | **Μήκος Συναρμογής Εισόδου** | **Ακτίνα Κυκλικού Τόξου** | **Μήκος Συναρμογής Εξόδου** | **Επίκλιση %** | **Ποσοστό απόσβεσης στον κύκλο** |
| A | -1085.0000 | 797.0000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 1 | -812.0000 | 172.0000 | 136.009 | 200.000 | 136.009 | 7.000 | 0.000 |
| 2 | -423.0000 | 1300.0000 | 136.009 | 200.000 | 136.009 | 7.000 | 0.000 |
| B | -150.0000 | 650.0000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |

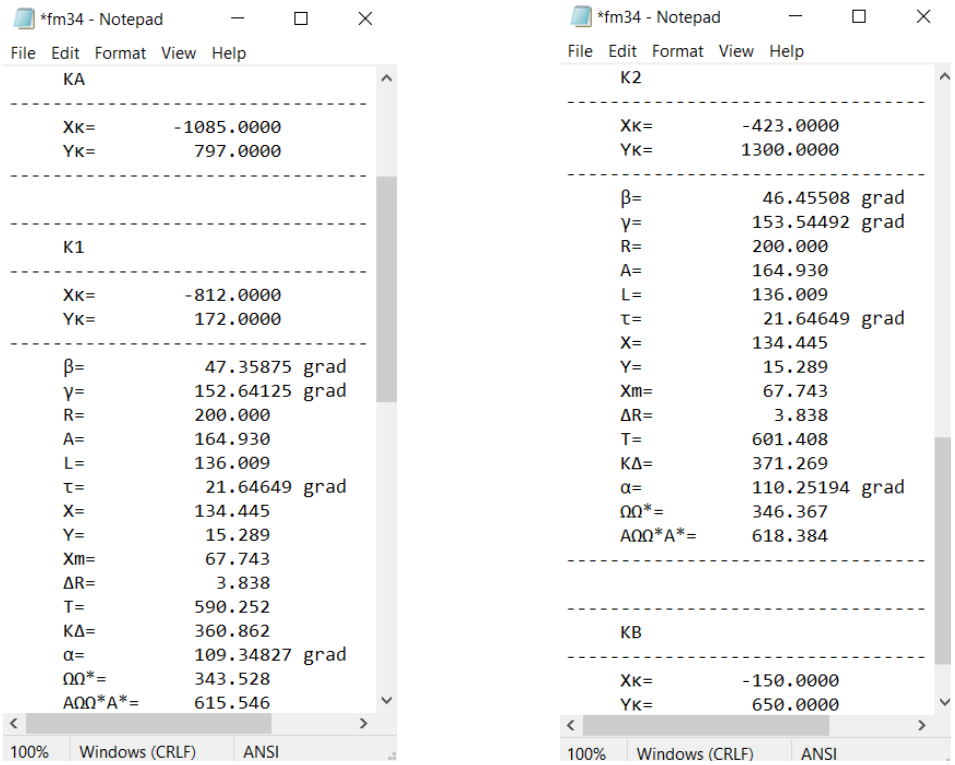
Εικόνα 4.10: FM19 – Εισαγωγή δεδομένων.

Έπειτα σχεδιάζεται η καμπύλη. Το παράδειγμα της οριζοντιογραφίας δίνεται στην εικόνα 4.11.



Εικόνα 4.11: FM19 - Σχέδιο παραδείγματος Οριζοντιογραφίας.

Ο χρήστης έχει πρόσβαση σε δεδομένα εξόδου και λεπτομέρειες σχεδίου μέσω του κουμπιού {ΣΤΟΙΧΕΙΑ ΟΡΙΖ}. Στην συνέχεια παρουσιάζονται τα αποτελέσματα που προέκυψαν από το παραπάνω πρόβλημα (Εικόνα 4.12).



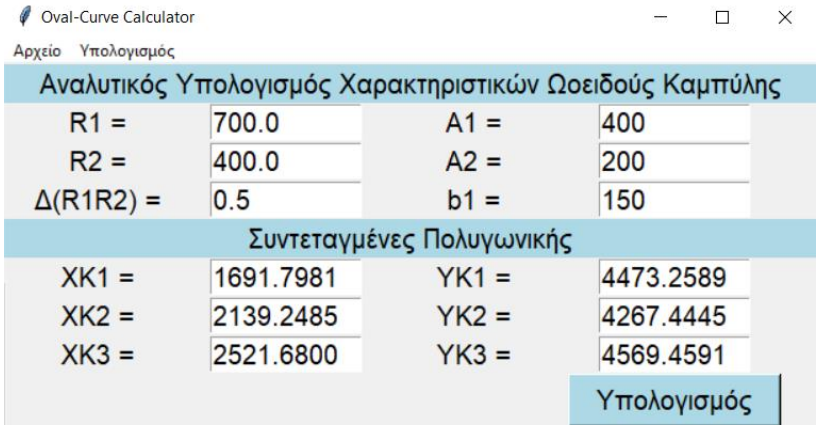

Εικόνα 4.12: FM19 - Αποτελέσματα Παραδείγματος Οριζοντιογραφίας.

## 4.3 ΠΑΡΑΔΕΙΓΜΑ ΩΟΕΙΔΟΥΣ ΚΑΜΠΥΛΗΣ

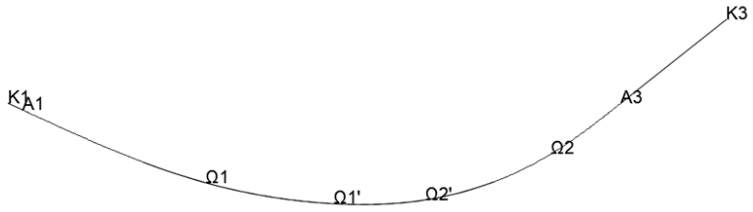
### 4.3.1 Επίλυση με την βοήθεια του προγράμματος Oval-Curve Calculator

Όπως αναφέρθηκε στα προηγούμενα κεφάλαια, το πιο βασικό χαρακτηριστικό της Ωοειδούς Καμπύλης είναι το ελάχιστο μήκος μεταξύ των εκατέρωθεν της ωοειδούς κυκλικών τόξων Δ(R1R2). Αυτή η απόσταση καθορίζει τόσο την τιμή της κοινής παραμέτρου Α όσο και γενικά την Ωοειδή Καμπύλη. Για αυτόν τον λόγο και δημιουργήθηκε το πρόγραμμα Oval-Curve Calculator, το οποίο λαμβάνει ως δεδομένο εισόδου αυτήν ακριβώς την απόσταση, η οποία πρακτικά αποτελεί και το βασικό στοιχείο που έχει ένας μελετητής.

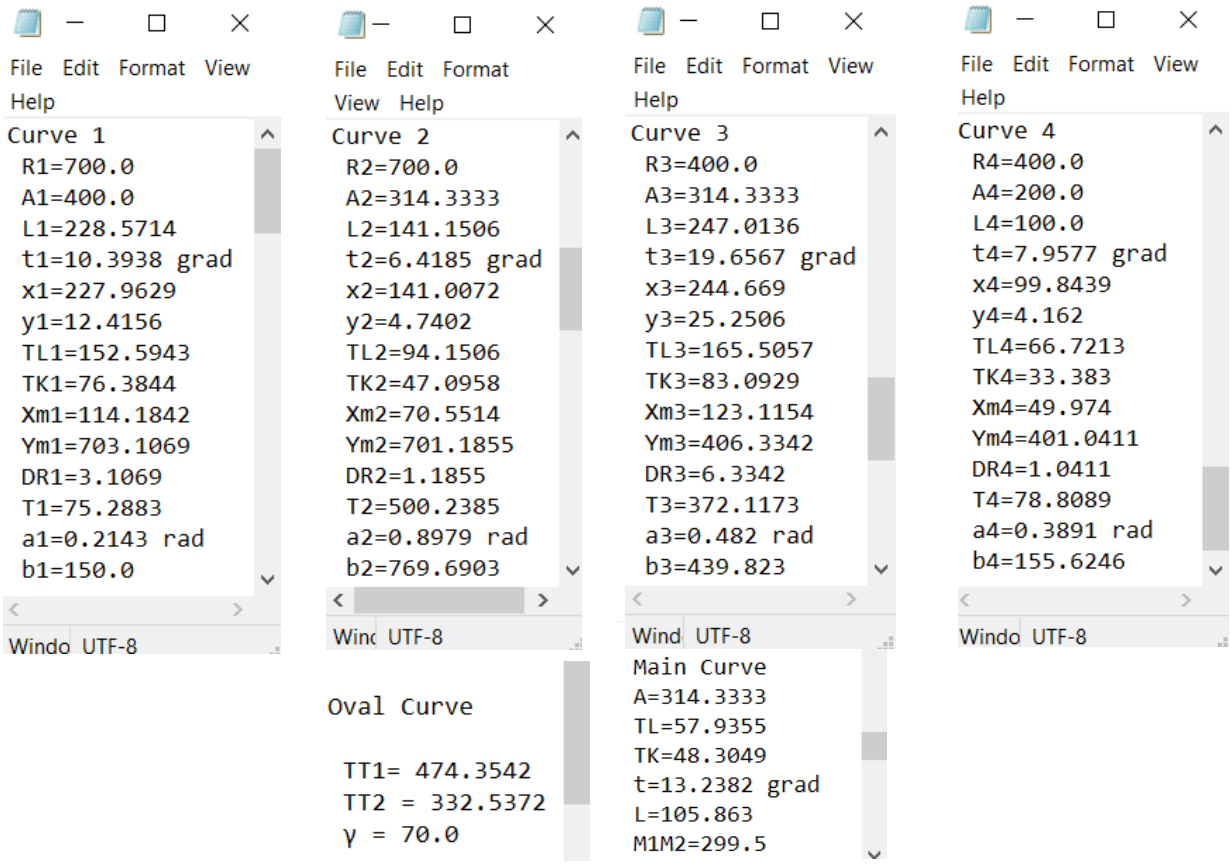
Στην εικόνα 4.13 παρουσιάζονται τα δεδομένα εισόδου του παραδείγματος, στην εικόνα 4.14 παρουσιάζεται το σχέδιο της οριζοντιογραφίας και τέλος στην εικόνα 4.15 τα αποτελέσματα.




Εικόνα 4.13: Ovalcalculator – Εισαγωγή δεδομένων.




Εικόνα 4.14: Oval-Curve Calculator - Σχέδιο παραδείγματος Οριζοντιογραφίας.




Εικόνα 4.15: Oval-Curve Calculator – Αποτελέσματα Παραδείγματος Οριζοντιογραφίας.

Τα λογισμικά που υπάρχουν στην αγορά δεν είναι προσαρμοσμένα να σχεδιάζουν μια Ωοειδή Καμπύλη, αλλά διάφορα είδη καμπυλών ή αλληλουχίες καμπυλών με αυθαίρετη επιλογή των γεωμετρικών παραμέτρων. Για να πληρούν όμως τις προϋποθέσεις και τους περιορισμούς από τις σχετικές οδηγίες, θα πρέπει πρώτα να γίνει μια δύσκολη διαδικασία υπολογισμών από τον μηχανικό ώστε να καθοριστούν τα απαιτούμενα δεδομένα εισόδου που χρειάζονται για το πρόγραμμα. Για να σχεδιαστεί λοιπόν αυτή η καμπύλη πρέπει να θεωρηθεί ως αλληλουχία κλωθοειδής εισόδου, κυκλικό τόξο, κλωθοειδής, κυκλικό τόξο, κλωθοειδής εξόδου (SCSCS). Τα δεδομένα εισόδου που θα χρειαστούν για τον σχεδιασμό της λαμβάνονται από τα στοιχεία εξόδου που παρέχει το Ovalcalculator. Παρακάτω παρουσιάζεται ο πίνακας ο οποίος περιλαμβάνει όλα τα πιθανά δεδομένα εισόδου που θα χρειαστούν με βάση το παράδειγμα που επιλύθηκε στο κεφάλαιο 4.3.1.

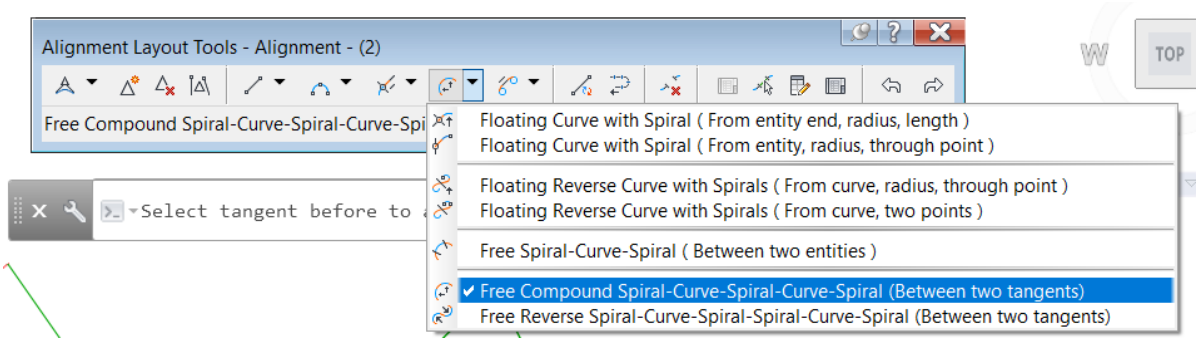
| Συντεταγμένες Πολυγωνικής | | | |
|---|---|---|---|
| Xk1 | 1691.798 | Yk1 | 4473.259 |
| Xk2 | 2139.249 | Yk2 | 4267.445 |
| Xk3 | 2521.68 | Yk3 | 4569.459 |

| Δεδομένα Πρώτης Καμπύλης | | | | Δεδομένα Δεύτερης Καμπύλης | | | | Δεδομένα Τρίτης Καμπύλης | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Δεδομένα Κλωθοειδούς εισόδου | | | | Δεδομένα Κλωθοειδούς εισόδου | | | | Δεδομένα Κλωθοειδούς εισόδου | | | |
| L1 | 228.5714 | A1 | 400 | L | 105.863 | A | 314.3333 | L4 | 100 | A4 | 200 |
| Δεδομένα Κυκλικού Τόξου | | | | Δεδομένα Κυκλικού Τόξου | | | | Δεδομένα Κυκλικού Τόξου | | | |
| R1 | 700 | b1 | 150 | | Δ(R1R2) | 0.5 | | R2 | 400 | b4 | 155.6246 |
| Δεδομένα Κλωθοειδούς εισόδου | | | | Δεδομένα Κλωθοειδούς εισόδου | | | | Δεδομένα Κλωθοειδούς εισόδου | | | |
| L1 | 228.5714 | A1 | 400 | L | 105.863 | A | 314.3333 | L4 | 100 | A4 | 200 |

Πίνακας 4.2: Δεδομένα Παραδείγματος Ωοειδούς Καμπύλης.

### 4.3.2 Επίλυση με την βοήθεια του προγράμματος Civil 3D

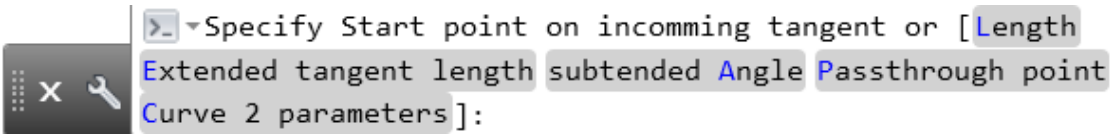
Ο σχεδιασμός μίας καμπύλης δρόμου μέσω του προγράμματος Civil 3D γίνεται με την δημιουργία Alignment (Εικόνα 4.4). Εφόσον σχεδιαστεί η πολυγωνική, μέσω του Alignment Creation Tools πρέπει να δημιουργηθεί μία αλληλουχία κλωθοειδής εισόδου, κυκλικό τόξο, κλωθοειδής, κυκλικό τόξο, κλωθοειδής εξόδου (SCSCS) (Εικόνα 4.16).




Εικόνα 4.16: Civil 3D- Δημιουργία Καμπύλης Τύπου SCSCS.

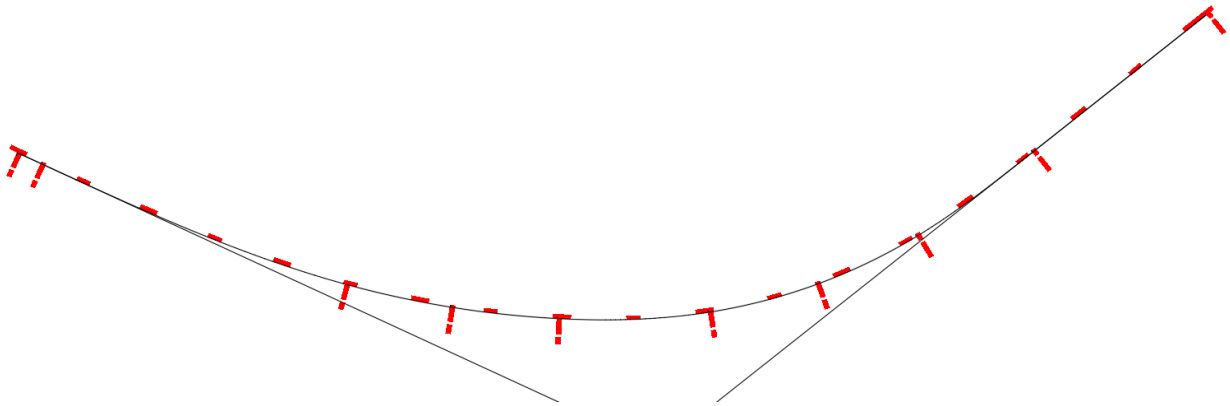
Επόμενο βήμα είναι ο προσδιορισμός των χαρακτηριστικών της καμπύλης που θέλουμε να σχεδιάσουμε. Από το Civil 3D ζητούνται τα εξής δεδομένα εισόδου-παράμετροι:

- Επιλογή των εφαπτόμενων της καμπύλης που θα σχεδιαστεί
- Προσδιορισμός ακτίνας R1 ή καμπυλότητα (degree of curvature)
- Προσδιορισμός ακτίνας R2 ή καμπυλότητα (degree of curvature)
- Προσδιορισμός παραμέτρου A1 ή μήκος κλωθοειδούς εισόδου L1
- Προσδιορισμός παραμέτρου A2 ή μήκος ενδιάμεσης κλωθοειδούς L
- Προσδιορισμός παραμέτρου A3 ή μήκος κλωθοειδούς εξόδου L2
- Προσδιορισμός σημείου αρχής ή μήκος κυκλικού τόξου b1 ή b2 ή κάποιου άλλου στοιχείου (Εικόνα 4.17).




Εικόνα 4.17: Civil 3D-Παράδειγμα τρόπου εισαγωγής δεδομένων.

Έπειτα σχεδιάζεται η καμπύλη και ο χρήστης μπορεί να επιλέξει αν θέλει τον προσδιορισμό της χιλιομετρικής θέσης στο σχέδιο καθώς και των βασικών σημείων της καμπύλης ως σημαίες. Το παράδειγμα της οριζοντιογραφίας δίνεται στην εικόνα 4.18.



Εικόνα 4.18: Civil 3D-Σχέδιο παραδείγματος Οριζοντιογραφίας.

Ο χρήστης έχει πρόσβαση σε δεδομένα εξόδου και λεπτομέρειες σχεδίου μέσω της επιλογής του Alignment Grid View. Στην συνέχεια παρουσιάζονται τα αποτελέσματα που προέκυψαν από το παραπάνω πρόβλημα (Εικόνα 4.19).

| No. | Type | Param... | Length | Radius | Direction | Start Station | End Station |
|---|---|---|---|---|---|---|---|
| 1 | Line | Two p... | 18.161' | | S65° 17' 56.12"E | 0+00.00' | 0+18.16' |
| 2.1 | Spiral-Cur... | Spiral... | 228.571' | | | 0+18.16' | 2+46.73' |
| 2.2 | Spiral-Cur... | Spiral... | 150.000' | 700.000' | | 2+46.73' | 3+96.73' |
| 2.3 | Spiral-Cur... | Spiral... | 105.863' | | | 3+96.73' | 5+02.60' |
| 2.4 | Spiral-Cur... | Spiral... | 155.625' | 400.000' | | 5+02.60' | 6+58.22' |
| 2.5 | Spiral-Cur... | Spiral... | 100.000' | | | 6+58.22' | 7+58.22' |
| 3 | Line | Two p... | 154.768' | | N51° 42' 03.85"E | 7+58.22' | 9+12.99' |

| No. | Delta angle | Chord length | Start Point | Center Point | A | Degree of Curv... |
|---|---|---|---|---|---|---|
| 1 | | | (1691.7981',4473.2589',... | | | |
| 2.1 | 009.3544 (d) | | (1708.2976',4465.6696',... | | 400.000' | |
| 2.2 | 012.2777 (d) | 149.713' | (1920.5903',4381.6871',... | (2105.8513',5056.7266'... | | 008.1851 (d) |
| 2.3 | 011.9144 (d) | | (2068.3743',4357.7306',... | | 314.333' | |
| 2.4 | 022.2916 (d) | 154.645' | (2173.9571',4362.0565',... | (2112.4497',4757.2993'... | | 014.3239 (d) |
| 2.5 | 007.1620 (d) | | (2319.2839',4414.9258',... | | 200.000' | |
| 3 | | | (2400.2197',4473.5392',... | | | |

Εικόνα 4.19: Civil 3D-Αποτελέσματα Παραδείγματος Οριζοντιογραφίας.

# *ΣΥΜΠΕΡΑΣΜΑΤΑ*

## 5.1 ΓΕΝΙΚΑ

Ο βασικός σκοπός της παρούσας Διπλωματικής Εργασίας ήταν η μελέτη και αυτοματοποίηση της οριζοντιογραφικής χάραξης δύο εκ των πιο διαδεδομένων καμπυλών. Για την επίτευξη του σκοπού αυτού απαιτούνται τα κατάλληλα εργαλεία και για αυτό το λόγο ήταν αναγκαία η δημιουργία κατάλληλου λογισμικού σε περιβάλλον ηλεκτρονικού υπολογιστή, το οποίο θα είναι σε θέση να εξάγει τα αποτελέσματα εύκολα, γρήγορα και αξιόπιστα.

Παρόλα αυτά, τα αποτελέσματα που εξάγονται από το λογισμικό δεν αρκούν για την πλήρη μελέτη χάραξης της οδού. Όπως αναφέρθηκε και στα προηγούμενα κεφάλαια η υπόψη διαδικασία είναι πολύ δύσκολη και χρονοβόρα στην οποία δεν υπάρχει μονοσήμαντη λύση μιας και οι παράγοντες που πρέπει να ληφθούν υπόψιν είναι πολλοί και αντικρουόμενοι. Έτσι, ο μελετητής καλείται να συμπεριλάβει όλους αυτούς τους παράγοντες και τις παραμέτρους ώστε να εξάγει το τελικό αποτέλεσμα. Παρόλα αυτά, με την βοήθεια του λογισμικού που δημιουργήθηκε του παρέχονται τα εργαλεία για να πετύχει το καλύτερο δυνατό αποτέλεσμα στον μικρότερο δυνατό χρόνο.

Σε αυτό το κεφάλαιο παρουσιάζονται τα βασικά συμπεράσματα αξιολόγησης της Διπλωματικής εργασίας, όπως αυτά εξήχθησαν από τα διάφορα στάδια αυτής. Τέλος, έχοντας ως γνώμονα την αντιμετώπιση και αυτοματοποίηση της πλήρης χάραξης της οδού, παρουσιάζονται διάφορες προτάσεις για περαιτέρω έρευνα και βελτίωσης της προσπάθειας που έγινε σε αυτό το τεύχος.

## 5.2 ΒΑΣΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ ΑΞΙΟΛΟΓΗΣΗΣ

Με το πέρας της παρούσας διπλωματικής εργασίας δημιουργήθηκαν δύο λογισμικά μέσω των οποίων επιτεύχθηκαν τα παρακάτω:

- Δημιουργία ενός πολύ χρήσιμου εργαλείου το οποίο έχει προσαρμοστεί στις ανάγκες του μηχανικού, σε αντίθεση με τα ήδη υπάρχοντα λογισμικά τόσο της Ελληνικής αλλά και παγκόσμιας αγοράς. Τα δεδομένα εισόδου επιλέχθηκαν με πολύ προσοχή και έχοντας ως κύριο γνώμονα την βοήθεια του μηχανικού σε προβλήματα που καλείται να αντιμετωπίσει.

- Διευκόλυνση του μηχανικού κατά τη διαδικασία του υπολογισμού και σχεδιασμού ειδικών μορφών καμπυλών, οι οποίες πολλές φορές αποφεύγονταν λόγω του υπολογιστικού φόρτου που απαιτούσαν.

- Δημιουργία λογισμικών φιλικών προς το χρήστη μιας και δεν χρειάζονται κάποιες ιδιαίτερες γνώσεις για τον χειρισμό τους.

- Γένεση δίαυλου επικοινωνίας με άλλα σχεδιαστικά προγράμματα. Επιλέχθηκαν τα αρχεία προς εξαγωγή με ιδιαίτερη σύνεση ώστε να περιλαμβάνουν όλες τις πιθανές πληροφορίες που μπορεί χρειαστεί ο μηχανικός σε μετέπειτα στάδιο μελέτης χάραξης.

## 5.3 ΠΡΟΤΑΣΕΙΣ ΓΙΑ ΠΕΡΕΤΑΙΡΩ ΕΡΕΥΝΑ

Η παρούσα εργασία αποτελεί την αφετηρία σε μία προσπάθεια με στόχο την αυτοματοποίηση της χάραξης της οδού. Στο πλαίσιο αυτό, τα στοιχεία που μπορούν να αποτελέσουν σημεία για περαιτέρω έρευνα είναι τα ακόλουθα:

- Ανάλυση περισσότερων καμπυλών (πχ διπλή ωοειδής) και δημιουργία αντίστοιχου λογισμικού που υπολογίζει και σχεδιάζει τις καμπύλες αυτές. Δεδομένου ότι το συγκεκριμένο λογισμικό που δημιουργήθηκε, πραγματεύεται μόνο τις δύο πιο διαδεδομένες καμπύλες, την Σιγμοειδή και την Ωοειδή, σε επόμενο στάδιο θα ήταν χρήσιμη η μελέτη περισσότερων μορφών καμπύλης.

- Επέκταση λογισμικού ώστε να παράγει πλήρη μελέτη της χάραξης. Η παρούσα εργασία είχε ως αντικείμενο την οριζοντιογραφική προσέγγιση της χάραξης. Θα μπορούσε λοιπόν να γίνει προσπάθεια υπολογισμού-σχεδιασμού και της κατακόρυφης χάραξης (μηκοτομές, διατομές, επικλίσεις κλπ.).

- Επέκταση λογισμικού ώστε να μπορεί να σχεδιάσει μια ενιαία χάραξη. Τα λογισμικά που δημιουργήθηκαν υπολογίζουν και σχεδιάζουν μόνο αυτόνομες τις δύο αυτές καμπύλες. Σε μετέπειτα στάδιο θα ήταν δόκιμο να γίνει προσπάθεια για δημιουργία λογισμικού που αντιμετωπίζει την οδό σαν μία ενιαία χάραξη, αποτελούμενη από διαφορετικές μορφές καμπυλών όπου η μία διαδέχεται την άλλη.

- Διερεύνηση ορατότητας σε οριζοντιογραφία και μηκοτομή. Ένα από τα βασικά κριτήρια κατά τη μελέτη κατασκευής μιας οδού ώστε να παρέχεται η απραραίτητη οδική ασφάλεια είναι η εξασφάλιση επαρκούς μήκους ορατότητας. Ιδιαίτερα στην Ελλάδα, χώρα με έντονο ανάγλυφο, είναι εξίσου σημαντική η διερεύνηση ορατότητας και σε μηκοτομή μιας και η δημιουργία κυρτής καμπύλης παρέχει περιορισμένη ορατότητα.

# *ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ*


1. Μαυρομάτης, "Σημειώσεις για το Μάθημα Γεωμετρικός Σχεδιασμός Οδών", Αθήνα, ΕΜΠ, 2018.
2. Pietzsch, "Σχεδιασμός και χάραξη των οδών", Dusseldorf, 1976
3. Krenz, Ostreloh, "Tables de clothoides pour l'etablissement des projects et le piquetage", Berlin 1960
4. Kasper, Schurba, Lorenz ,1956. "Die Klothoide als Trassierungselement". Bonn, Germany 1968
5. Κανελλαΐδης Γ., Φ. Μερτζάνης, Γ. Γλαρός, και Σ. Βαρδάκη, 2017. "Σημειώσεις Γεωμετρικού Σχεδιασμού των Οδών",
6. Υπουργείο Υποδομών και Μεταφορών, 2001. Γενική Γραμματεία Δημοσίων Έργων. Διεύθυνση Μελετών Έργων Οδοποιίας. Οδηγίες Μελετών Οδικών Έργων. (ΟΜΟΕ). Τεύχος 3: Χαράξεις (ΟΜΟΕ - Χ).
7. Υπουργείο Υποδομών και Μεταφορών, 2013. Γενική Γραμματεία Δημοσίων Έργων. Διεύθυνση Μελετών Έργων Οδοποιίας. Οδηγίες Μελετών Οδικών Έργων. (ΟΜΟΕ). Τεύχος 3: Χαράξεις, Έκδοση 2 - Αναθεώρηση (ΟΜΟΕ - Χ). 7.
8. Γ. Κανελλαΐδης - Φ. Μερτζάνης - Γ. Γλαρός - Σ. Βαρδάκη, "Σημειώσεις Γεωμετρικού Σχεδιασμού των Οδών", Αθήνα, ΕΜΠ, 2017.
9. Γιαννής Γιώργος, "Οδηγίες Για την Ετοιμασία Της Διπλωματικής Εργασίας", Αθήνα, ΕΜΠ, 2010.
10. Γιώτης, Α., Κανελλαΐδης, Γ., Μαλέρδος, Γ. (1990). Γεωμετρικός Σχεδιασμός των Οδών. Αθήνα: Συμεών.
11. INTEGRATED SPEED MANAGEMENT STRATEGIES IN LOCAL COMMUNITIES IN SOUTH EAST EUROPE, George Yannis , Stergios Mavromatis , Alexandra Laiou
12. http://portal.survey.ntua.gr/main/labs/roads/Roads-g_files/Edu-g_files/Edu-41-g_files/Horizontal%20Curves.pdf
13. http://library.tee.gr/digital/m2070/m2070_iliou3.pdf
14. https://www.geeksforgeeks.org/python-programming-language/
15. https://docs.python.org/3/library/
16. https://realpython.com/
17. https://www.w3schools.com/python/
18. https://www.programiz.com/python-programming

# *ΠΑΡΑΡΤΗΜΑ*

## Κώδικες Προγραμματισμού

## 7.1 Σιγμοειδής Καμπύλη – Κώδικας Συναρτήσεων Python

```
import sys
from math import sin, cos, pi, tan, floor, sqrt, atan, acos
import matplotlib.pyplot as plt
import numpy as np
import sympy as sym
from dxfwrite import DXFEngine as dxf

def sigma(r1,r2,d):
    "See if we can fit a sigmoid and have distance d."
    c = 1.0
    for c in frangec(1, 1.5, 0.01):
        a1, s , dd= prosarm1(r1, r2, d, c)
        if s >= 0.0: break
    a1,a2,dd= prosarm2(r1, r2, d, c, a1)
    return round(a1,4),round(a1/c,4),s, dd

def check(r1, r2, d):
    "Check if input is correct"
    if (r1>0 and r2>0 and r1>=r2 and d>=0): return
    raise ValueError('Please check input: R1>0 R2>0 R1>R2 D>=0')

def omega(r1, a1):
    "Compute coordinates of omega point."
    x1 =(a1**2/r1)-((a1**2/r1)**5)/(40*a1**4)+((a1**2/r1)**9)/(3456*a1**8)
    y1= ((a1**2/r1)**3)/(6*a1**2)-((a1**2/r1)**7)/(336*a1**6)+
        ((a1**2/r1)**11)/(42240*a1**10)
    return x1, y1

def prosarm1(r1, r2, d, c):
    "Find a1 so that citerion is met; s>=0 is successful, s<0 unsuceessful."
    for a1 in frangec(r1/3.0, min(r1, c*r2), 0.05):
        a2=a1/c
        x1, y1 = omega(r1, a1)
        t1= ((a1**2)/(2*r1**2))
        xm1= x1-r1*sin(t1)
        ym1=y1+r1*cos(t1)
        dr1=ym1-r1

        x2, y2 = omega(r2, a2)
```

```
        t2= ((a2**2)/(2*r2**2))
        xm2= x2-r2*sin(t2)
        ym2=y2+r2*cos(t2)
        dr2=ym2-r2

        s=(r1+r2+dr1+dr2)**2+(xm1+xm2)**2-(r1+r2+d)**2
        dd=sqrt((r1+r2+dr1+dr2)**2+(xm1+xm2)**2)-r1-r2
        if s >= 0.0: break
    return a1, s, dd

def prosarm2(r1, r2, d, c, a1):
    "Find a1 so that citerion is met; s>=0 is successful, s<0 unsuceessful."
    for a1 in frangec((a1-0.05), (a1+0.05), 0.0001):
        a2=a1/c
        x1, y1 = omega(r1, a1)
        t1= ((a1**2)/(2*r1**2))
        xm1= x1-r1*sin(t1)
        ym1=y1+r1*cos(t1)
        dr1=ym1-r1

        x2, y2 = omega(r2, a2)
        t2= ((a2**2)/(2*r2**2))
        xm2= x2-r2*sin(t2)
        ym2=y2+r2*cos(t2)
        dr2=ym2-r2

        s=(r1+r2+dr1+dr2)**2+(xm1+xm2)**2-(r1+r2+d)**2
        dd=sqrt((r1+r2+dr1+dr2)**2+(xm1+xm2)**2)-r1-r2
        if s >= 0.0: break
    return a1, a2, dd

def frangec(start, end, step=1, tol=0.0):
    """Implements float xrange - which includes start and end; tol is tolerance ratio.
      if the last step before end is within tol*step distance from end it is skipped."""
    assert tol>=0.0, "xfrangec(): tolerance shouble a non negative ratio."
    start = float(start); end = float(end); step = float(step); tol = float(tol)
    a = start
    if step > 0:
        if start > end: return
        while a < end-tol*step:
            yield a
            a += step
    else:
        if start < end: return
```

```
        while a > end-tol*step:  #Note tol>=0 and step < 0
            yield a
            a += step
    yield end

def rad2grad(x):
    "Convert angle from radians to gradians."
    return x*200/pi

def ggg(t21,t23):
    "Find the angle between 2 lines"
    b=acos(vmul(t21,t23))
    return pi-b

def frac(r,a,g):
    "Compute all parameters of klothoide"
    l=(a**2)/r
    t=l/(2*r)
    x, y=omega(r,a)
    xm=x-(r*sin(t))
    ym=y+(r*cos(t))
    dr=y-(r*(1-cos(t)))
    d=((r+dr)/(cos(g/2)))-r
    T=((r+dr)*tan(g/2))+xm
    o=(g)-(l/r)
    s=r*o
    m=(2*l)+s
    return l,t,x,y,xm,ym,dr,d,T,o,s,m

def frac1(r,a):
    "Compute all parameters of klothoide"
    l=(a**2)/r
    t=l/(2*r)
    x, y=omega(r,a)
    tl=x-(y/tan(t))
    tk=y/sin(t)
    xm=x-(r*sin(t))
    ym=y+(r*cos(t))
    dr=y-(r*(1-cos(t)))
    return l,t,x,y,tl,tk,xm,ym,dr

def fact(n):
    "Compute the factorial of a number"
    fact = 1
```

```
    for i in range(1,n+1):
        fact = fact * i
    return fact

def orosx(k,s,a):
    "Compute the k repetition of the s point of the klothoide"
    ox = (((-1)**(k/4))*(s**(k+1)))/((2**(k/2))*(k+1)*(fact(floor(k/2)))*(a**k))
    return ox

def orosy(k,s,a):
    "Compute the k repetition of the s point of the klothoide"
    oy=(((-1)**((k-2)/4))*(s**(k+1)))/((2**(k/2))*(k+1)*(fact(floor(k/2)))*(a**k))
    return oy

def kloth1(s,a):
    "Compute the coordinates of one point in klothoide"
    x=s
    y=0
    for k in range(2, 14, 4):
        y=y+orosy(k,s,a)
        x=x+orosx((k+2),s,a)
    return x,y

def kloth(l,a):
    "Create a list of  the coordinates of every point in klothoide by 10m, even numbers are x -
odd
     numbers are y" "ex. x[0],x[1] =x1,y1 etc"
    klothoide=[]
    for s in frangec(0, l, 10):
        x, y = kloth1(s,a)
        klothoide.append(x)
        klothoide.append(y)
    return klothoide

def circ1(i,xm,dr,r,t,l):
    "Compute the i repetition of the s point of the coordinates of the circle"
    x=xm+(r*(sin(t+((i-l)/r))))
    y=dr+r-(r*(cos(t+((i-l)/r))))
    return x,y

def circle(xm,dr,r,t,m,l):
    "Create a list of  the coordinates of every point in circle  by 100m,  even numbers are x - odd
     numbers are y" "ex. x[0],x[1] =x1,y1 etc"
    circle=[]
```

```
    for i in frangec(l,m/2,10):
        x,y=circ1(i,xm,dr,r,t,l)
        circle.append(x)
        circle.append(y)
    return circle

def unitvector(xk1,yk1,xk2,yk2):
    "Compute the unit vectors of the polygonic"
    x=xk2-xk1
    y=yk2-yk1
    z=sqrt(x**2+y**2)
    t=[x/z,y/z]
    return t

def vertical(t):
    "Compute the vertical unit vector to a unit vector"
    n=[-t[1],t[0]]
    return n

def bisector(t12,t32):
    "Find the parallel vector to the bisector"
    d=[]
    for i in range(0,2):
        d.append(-t12[i]-t32[i])
    return (d)

def vmul(a,b):
    "Compute the dot product of two vectors"
    c=a[0]*b[0]+a[1]*b[1]
    return (c)

def direction(d, n12):
    "Fix the direction of the vertical vectors towards the center of the curve"
    if vmul(d, n12)<0:
        n12=[-n12[0],-n12[1]]
    else:
        n12==n12
    return n12

def pvector(a,t,n,i):
    "Compute the p vector"
    x=a[i]
    y=a[i+1]
    p=[x*t[0]+y*n[0],x*t[1]+y*n[1]]
```

```
    return p

def general(a,t,n,xa,ya):
    "Convert the local  x,y to the general x,y"
    c=[]
    for i in range(0,(len(a)-1),2):
        p=pvector(a,t,n,i)
        x=xa+p[0]
        y=ya+p[1]
        c.append(x)
        c.append(y)
    return c

def plot(xykloth1):
    "Plot the general x,y of the curve"
    x=[]
    y=[]
    for i in range(0,len(xykloth1),2):
        x.append(xykloth1[i])
        y.append(xykloth1[i+1])
        plt.plot(x, y, color='green', markersize=1)

def plotpolyg(xk1,yk1,xk2,yk2,xk3,yk3,xk4,yk4):
    "Plot the polygonic"
    c=[xk1,yk1,xk2,yk2,xk3,yk3,xk4,yk4]
    for i in range (0,5,2):
        plt.plot(c[i], c[i+1], color='green', linestyle='solid', linewidth = 3, marker='o', \
                 markerfacecolor='blue', markersize=6)
        plt.plot(c[i+2], c[i+3], color='green', linestyle='solid', linewidth = 3, marker='o', \
                 markerfacecolor='blue', markersize=6)
        x_values = [c[i], c[i+2]]
        y_values = [c[i+1],c[i+3]]
        plt.plot(x_values, y_values, color='green')

def writetxt(f, i,r,a,l,g,t,x,y,xm,ym,dr,d,T,o,s,m):
    "Write the mains of a curve in a txt"
    f.write("Curve {}".format(i))
    f.write("\n R{}={}".format(i,round(r,4)))
    f.write("\n A{}={}".format(i,round(a,4)))
    f.write("\n L{}={}".format(i,round(l,4)))
    f.write("\n g{}={} grad".format(i,round(rad2grad(g),4)))
    f.write("\n t{}={} grad".format(i,round(rad2grad(t),4)))
    f.write("\n x{}={}".format(i,round(x,4)))
    f.write("\n y{}={}".format(i,round(y,4)))
```

```
    f.write("\n Xm{}={}".format(i,round(xm,4)))
    f.write("\n Ym{}={}".format(i,round(ym,4)))
    f.write("\n DR{}={}".format(i,round(dr,4)))
    f.write("\n d{}={}".format(i,round(d,4)))
    f.write("\n T{}={}".format(i,round(T,4)))
    f.write("\n a{}={} rad".format(i,round(o,4)))
    f.write("\n S{}={}".format(i,round(s,4)))
    f.write("\n AA'{}={}".format(i,round(m,4)))
    f.write("\n\n\n")

def writetxt1(f,i,r,a,l,t,x,y,tl,tk,xm,ym,dr):
    "Write the mains of a curve in a txt"
    f.write("Curve {}".format(i))
    f.write("\n R{}={}".format(i,round(r,4)))
    f.write("\n A{}={}".format(i,round(a,4)))
    f.write("\n L{}={}".format(i,round(l,4)))
    f.write("\n t{}={} grad".format(i,round(rad2grad(t),4)))
    f.write("\n x{}={}".format(i,round(x,4)))
    f.write("\n y{}={}".format(i,round(y,4)))
    f.write("\n TL{}={}".format(i,round(tl,4)))
    f.write("\n TK{}={}".format(i,round(tk,4)))
    f.write("\n Xm{}={}".format(i,round(xm,4)))
    f.write("\n Ym{}={}".format(i,round(ym,4)))
    f.write("\n DR{}={} \n\n".format(i,round(dr,4)))

def writecoordinatesF(ff,xykloth1,k):
    "Write the Coordinates of the S Curve Farwards"
    for i in range (2,len(xykloth1)-2,2):
        ff.write('P{} = {}, {} \n'.format(k,round(xykloth1[i],4),round(xykloth1[i+1],4)))
        k+=1
    return k

def writecoordinatesB(ff,xykloth1,k):
    "Write the Coordinates of the S Curve Backwards"
    for i in range (len(xykloth1)-3,2,-2):
        ff.write('P{} = {}, {} \n'.format(k,round(xykloth1[i-1],4),round(xykloth1[i],4)))
        k+=1
    return k

def lend(xk2,yk2,xk3,yk3):
    "Calculate the length of K2K3"
    d=sqrt(((xk2-xk3)**2)+((yk2-yk3)**2))
    return d
```

```
def checkdist(d,T1,T2,a1,a2):
    "Check if the S-curve is possible."
    n=0
    if (d>=T1+T2):
        if ((d-T1-T2)>(0.08*(a1+a2))):
            n=1
    else:
        n=2
    return n

def tex(drawing, tex,poin):
    "Write text on dxf"
    text = dxf.text(tex, poin, height=10, rotation=0)
    text['color'] = 7
    drawing.add(text)
    drawing.add(dxf.point((poin)))

def pointsF(polyline, t):
    "Create polyline of the kloth or circle Forwards"
    for i in range(0,len(t)-1,2):
        c=(t[i],t[i+1])
        polyline.add_vertex(c)

def pointsB(polyline, t):
    "Create polyline of the kloth or circle Backwards"
    for i in range(len(t)-1,0,-2):
        c=(t[i-1],t[i])
        polyline.add_vertex(c)

def auto(fn, xk1,yk1,xa1,ya1,xykloth1,xycircle1,xycircle11,xykloth11,xaa1,yaa1,
            xa2,ya2,xykloth2,xycircle2,xycircle22,xykloth22,xaa2,yaa2,xk4,yk4):
    "Write all dxf"
    drawing = dxf.drawing(fn)
    polyline= dxf.polyline(linetype='CONTINUOUS')
    tex(drawing, "K1",(xk1, yk1))
    #drawing.add(dxf.point((xk1, yk1)))
    drawing.add(dxf.line((xk1, yk1), (xa1, ya1), color=7))
    tex(drawing, "A1",(xa1, ya1))
    #drawing.add(dxf.point((xa1, ya1)))
    pointsF(polyline, xykloth1)
    tex(drawing, 'Ω1',(xycircle1[0],xycircle1[1]))
    #drawing.add(dxf.point((xycircle1[0],xycircle1[1])))
    pointsF(polyline, xycircle1)
```

```
    tex(drawing, 'Δ1',(xycircle1[-2],xycircle1[-1]))
    #drawing.add(dxf.point((xycircle1[-2],xycircle1[-1])))
    pointsB(polyline, xycircle11)
    tex(drawing, "Ω1'",(xycircle11[0],xycircle11[1]))
    #drawing.add(dxf.point((xycircle11[0],xycircle11[1])))
    pointsB(polyline, xykloth11)
    tex(drawing, "A1'",(xaa1, yaa1))
    #drawing.add(dxf.point((xaa1, yaa1)))
    drawing.add(dxf.line((xaa1, yaa1), (xa2, ya2), color=7))
    tex(drawing, "A2",(xa2, ya2))
    #drawing.add(dxf.point((xa2, ya2)))
    pointsF(polyline, xykloth2)
    tex(drawing, 'Ω2',(xycircle2[0],xycircle2[1]))
    pointsF(polyline, xycircle2)
    tex(drawing, 'Δ2',(xycircle2[-2],xycircle2[-1]))
    pointsB(polyline, xycircle22)
    tex(drawing, "Ω2'",(xycircle22[0],xycircle22[1]))
    pointsB(polyline, xykloth22)
    tex(drawing, "A2'",(xaa2, yaa2))
    drawing.add(dxf.line((xaa2, yaa2), (xk4, yk4), color=7))
    tex(drawing, "K4",(xk4, yk4))
    drawing.add(polyline)
    drawing.save()

def pymain(xk1, yk1, xk2, yk2, xk3, yk3, xk4, yk4, r1, r2, a1, a2, f, ff, fff):
    'main program run'

    #find the vectors of curve 1
    t12= unitvector(xk1,yk1,xk2,yk2)
    t32= unitvector(xk3,yk3,xk2,yk2)
    n12= vertical(t12)
    n32= vertical(t32)
    d=bisector(t12,t32)
    n12= direction(d,n12)
    n32= direction(d,n32)

    #find angle of curve 1
    t21=[-t12[0],-t12[1]]
    t23=[-t32[0],-t32[1]]
    g1=ggg(t21,t23)

    #find the vectors of curve2
    t23= unitvector(xk2,yk2,xk3,yk3)
    t43= unitvector(xk4,yk4,xk3,yk3)
```

```
n23= vertical(t23)
n43= vertical(t43)
d=bisector(t23,t43)
n23= direction(d,n23)
n43= direction(d,n43)

#find angle of curve 2
t32=[-t23[0],-t23[1]]
t34=[-t43[0],-t43[1]]
g2=ggg(t32,t34)

#Compute various elements of the clothoid curve
l1,t1,x1,y1,xm1,ym1,dr1,d1,T1,o1,s1,m1=frac(r1,a1,g1)
l2,t2,x2,y2,xm2,ym2,dr2,d2,T2,o2,s2,m2=frac(r2,a2,g2)

#Compute length of line segment so the curve fits
dist=lend(xk2,yk2,xk3,yk3)
checkdist(dist,T1,T2,a1,a2)

#Coordinates of Points A1, A1'
xa1,ya1=xk1+t12[0]*(lend(xk1,yk1,xk2,yk2)-T1),yk1+t12[1]*(lend(xk1,yk1,xk2,yk2)-T1)
xaa1,yaa1=xk3+t32[0]*(lend(xk2,yk2,xk3,yk3)-T1),yk3+t32[1]*(lend(xk2,yk2,xk3,yk3)-T1)

#Coordinates of Points of Klothoide 1, 11
klothoide1=kloth(l1,a1)
xykloth1=general(klothoide1,t12,n12,xa1,ya1)
xykloth11=general(klothoide1,t32,n32,xaa1,yaa1)

#Coordinates of Points of Circle 1, 11
circle1=circle(xm1,dr1,r1,t1,m1,l1)
xycircle1=general(circle1,t12,n12,xa1,ya1)
xycircle11=general(circle1,t32,n32,xaa1,yaa1)

#Coordinates of points A2, A2'
xa2,ya2=xk2+t23[0]*(lend(xk2,yk2,xk3,yk3)-T2),yk2+t23[1]*(lend(xk2,yk2,xk3,yk3)-T2)
xaa2,yaa2=xk4+t43[0]*(lend(xk3,yk3,xk4,yk4)-T2),yk4+t43[1]*(lend(xk3,yk3,xk4,yk4)-T2)

#Coordinates of Points of Klothoide 2, 22
klothoide2=kloth(l2,a2)
xykloth2=general(klothoide2,t23,n23,xa2,ya2)
xykloth22=general(klothoide2,t43,n43,xaa2,yaa2)
```

```
#Coordinates of Points of Circle  2, 22
circle2=circle(xm2,dr2,r2,t2,m2,l2)
xycircle2=general(circle2,t23,n23,xa2,ya2)
xycircle22=general(circle2,t43,n43,xaa2,yaa2)

#Write txt file with the Details of the S Curve
if not(f == None) :
   f= open(f, "w")
   writetxt(f, 1,r1,a1,l1,g1,t1,x1,y1,xm1,ym1,dr1,d1,T1,o1,s1,m1)
   writetxt(f, 2,r2,a2,l2,g2,t2,x2,y2,xm2,ym2,dr2,d2,T2,o2,s2,m2)
   f.close()

#write txt file with the Coordinates of the Oval Curve
if not(ff==None) :
   ff= open(ff, "w")
   ff.write("Coordinates of S Curve \n")
   ff.write("K1 = {}, {} \n".format(round(xk1,4),round(yk1,4)))
   ff.write("A1 = {}, {} \n".format(round(xa1,4),round(ya1,4)))
   k=1
   k=writecoordinatesF(ff,xykloth1,k)
   ff.write("O1 = {}, {} \n".format(round(xycircle1[0],4),round(xycircle1[1],4)))
   k=writecoordinatesF(ff,xycircle1,k)
   ff.write("D1 = {}, {} \n".format(round(xycircle11[-2],4),round(xycircle11[-1],4)))
   k=writecoordinatesB(ff,xycircle11,k)
   ff.write("O1' = {}, {} \n".format(round(xycircle11[0],4),round(xycircle11[1],4)))
   k=writecoordinatesB(ff,xykloth11,k)
   ff.write("A1' = {}, {} \n".format(round(xaa1,4),round(yaa1,4)))
   ff.write("A2 = {}, {} \n".format(round(xa2,4),round(ya2,4)))
   k=writecoordinatesF(ff,xykloth2,k)
   ff.write("O2 = {}, {} \n".format(round(xycircle2[0],4),round(xycircle2[1],4)))
   k=writecoordinatesF(ff,xycircle2,k)
   ff.write("D2 = {}, {} \n".format(round(xycircle22[-2],4),round(xycircle22[-1],4)))
   k=writecoordinatesB(ff,xycircle22,k)
   ff.write("O2' = {}, {} \n".format(round(xycircle22[0],4),round(xycircle22[1],4)))
   k=writecoordinatesB(ff,xykloth22,k)
   ff.write("A2' = {}, {} \n".format(round(xaa2,4),round(yaa2,4)))
   ff.write("K4 = {}, {} \n".format(round(xk4,4),round(yk4,4)))
   ff.close()
```

```
    #Write in dxf the S curve
    if not(fff == None) :
       auto(fff, xk1,yk1,xa1,ya1,xykloth1,xycircle1,xycircle11,xykloth11,xaa1,yaa1,
            xa2,ya2,xykloth2,xycircle2,xycircle22,xykloth22,xaa2,yaa2,xk4,yk4)

    #plot the s curve
    plt.title(label='Σιγμοειδής Καμπύλη', fontdict=None, loc='center', pad=None)
    plotpolyg(xk1,yk1,xk2,yk2,xk3,yk3,xk4,yk4)
    plt.plot(xa1, ya1, color='green', linestyle='dashed', linewidth = 3, marker='o',\
            markerfacecolor='blue', markersize=6)
    plt.plot(xaa1, yaa1, color='green', linestyle='dashed', linewidth = 3, marker='o',\
            markerfacecolor='blue', markersize=6)
    plot(xykloth1)
    plt.plot(xycircle1[0], xycircle1[1], color='green', linestyle='dashed', linewidth = 3,\
             marker='o', markerfacecolor='blue', markersize=6)
    plot(xycircle1)
    plt.plot(xycircle11[0], xycircle11[1], color='green', linestyle='dashed', linewidth = 3,\
            marker='o', markerfacecolor='blue', markersize=6)
    plot(xycircle11)
    plt.plot(xycircle11[-2], xycircle11[-1], color='green', linestyle='dashed', linewidth = 3,\
            marker='o', markerfacecolor='blue', markersize=6)
    plot(xykloth11)
    plt.plot(xa2, ya2, color='green', linestyle='dashed', linewidth = 3, marker='o',\
            markerfacecolor='blue', markersize=6)
    plt.plot(xaa2, yaa2, color='green', linestyle='dashed', linewidth = 3, marker='o',\
            markerfacecolor='blue', markersize=6)
    plot(xykloth2)
    plt.plot(xycircle2[0], xycircle2[1], color='green', linestyle='dashed', linewidth = 3, marker='o',\
            markerfacecolor='blue', markersize=6)
    plot(xycircle2)
    plt.plot(xycircle22[0], xycircle22[1], color='green', linestyle='dashed', linewidth = 3,\
            marker='o', markerfacecolor='blue', markersize=6)
    plot(xycircle22)
    plt.plot(xycircle22[-2], xycircle22[-1], color='green', linestyle='dashed', linewidth = 3, \
            marker='o', markerfacecolor='blue', markersize=6)
    plot(xykloth22)
    plt.show()

def test():
    "Test the program."
    #data insert#
    check(r1, r2, d)
    a1,a2,s,dd=sigma(r1,r2,d)
```

```
pymain(xk1, yk1, xk2, yk2, xk3, yk3, xk4, yk4, r1, r2, a1, a2, f, ff, fff
```

## 7.2 Σιγμοειδής Καμπύλη – Κώδικας Γραφικού Περιβάλλοντος (GUI) Python

```
import tkinter as tk
from tkinter import font as tkfo
from math import sin, cos, pi, tan, sqrt, acos, floor, atan, asin
import tkinter.messagebox
from DefinitionsSCurve import
rad2grad,sigma,frac1,writetxt1,prosarm1,pymain,unitvector,ggg,lend,frac,checkdist
from tkinter import filedialog
import matplotlib.pyplot as plt
from pathlib import Path

class SCalculator:
    'Create a Spread Sheet'
    def __init__(self):
        self.makeWidgets()

    def setFonts(root):
        for t in 'TkDefaultFont', 'TkFixedFont', 'TkTextFont':
            font1 = tk.font.Font(name=t, exists=True)
            font1.config(size=15)
            font1.config(family='Arial')

    def setMenus(self):
        "Create the menu system."
        menuBar = tk.Menu(self.root, activebackground="green")
        menu = tk.Menu(menuBar, activebackground="green", tearoff=False)
        menuBar.add_cascade(label="Αρχείο", menu=menu, foreground="blue")
        self.root.config(menu=menuBar)
        menu.add_command(label="Αποθήκευση", command=self.saveAll, foreground="blue")
        menu.add_command(label="Ανοιγμα", command=self.readAll, foreground="blue")
        computemenu = tk.Menu(menuBar, activebackground="green", tearoff=False)
        menuBar.add_cascade(label="Υπολογισμός", menu=computemenu, foreground="blue")
        computemenu.add_command(label="Αναλυτικός υπολογισμός",
command=self.showmore, foreground="blue")
        computemenu.add_command(label="Απλός υπολογισμός", command=self.delete,
foreground="blue")
```

```
def makeWidgets(self):
    'Make the widgets'
    self.root = tk.Tk()
    self.root.title('S-Curve Calculator')
    self.setFonts()
    self.setMenus()
    self.save=False
    self.simple=True

    self.entries=[]
    self.labels=[]

    ir=0
    lab = tk.Label(self.root, text="Απλός Υπολογισμός Χαρακτηριστικών Σιγμοειδούς
                    Καμπύλης",activebackground='steelblue',bg='light blue',
                    width=55).grid(row=ir, column=0, columnspan=5)

    ir+=1
    temp = tk.Label(self.root, text="R1=",width=10)
    temp.grid(row=ir, column=0)
    self.ent = tk.Entry(self.root,width=10)
    self.ent.grid(row=ir,column=1)
    self.entries.append(self.ent)

    ir+=1
    temp = tk.Label(self.root, text="R2=",width=10).grid(row=ir, column=0)
    self.ent=tk.Entry(self.root,width=10)
    self.ent.grid(row=ir,column=1)
    self.entries.append(self.ent)

    ir+=1
    temp = tk.Label(self.root, text="D=",width=10).grid(row=ir, column=0)
    self.ent=tk.Entry(self.root,width=10)
    self.ent.grid(row=ir,column=1)
    self.entries.append(self.ent)

    ir+=1
    temp = tk.Label(self.root, text="A1=",width=10).grid(row=ir, column=0)
    self.temp1 = tk.Entry(self.root, width=10)
    self.temp1.grid(row=ir, column=1)
    self.temp1.configure(justify='center',state='readonly')
```

```
    ir+=1
    temp = tk.Label(self.root, text="A2 =",width=10).grid(row=ir, column=0)
    self.temp2 = tk.Entry(self.root, width=10)
    self.temp2.grid(row=ir, column=1)
    self.temp2.configure(justify='center',state='readonly')
    self.button=tk.Button(self.root, text='Υπολογισμός', activebackground='steelblue',
                          bg='light blue',command=self.calcu1,justify='center')
    self.button.grid(row=ir,column=2,sticky='we',columnspan=1)


    self.root.mainloop()

def calcu1(self):
    "Calculate the results of Simple S-curve"
    self.temp1.configure(state='normal')
    self.temp1.delete(0,'end')
    self.temp1.configure(state='readonly')
    self.temp2.configure(state='normal')
    self.temp2.delete(0,'end')
    self.temp2.configure(state='readonly')
    try: n = float(self.entries[0].get().replace(',','.'))
    except ValueError: n = None
    if n is None or n <= 0:
        tk.messagebox.showinfo("Error", "R1 <= 0 ", parent=self.root,
                               icon=tk.messagebox.ERROR)
        return
    try: n = float(self.entries[1].get().replace(',','.'))
    except ValueError: n = None
    if n is None or n <= 0:
        tk.messagebox.showinfo("Error", "R2 <= 0 ", parent=self.root,
                               icon=tk.messagebox.ERROR)
        return
    try: n = float(self.entries[2].get().replace(',','.'))
    except ValueError: n = None
    if n is None or n <= 0:
        tk.messagebox.showinfo("Error", "D <= 0 ", parent=self.root,
                               icon=tk.messagebox.ERROR)
        return
    r1=float(self.entries[0].get().replace(',','.'))
    r2=float(self.entries[1].get().replace(',','.'))
    d=float(self.entries[2].get().replace(',','.'))
    a1,a2,s,dd=sigma(r1,r2,d)
    if not(round(dd,2)==(float(self.entries[2].get()))):
        tk.messagebox.showinfo ("Warning","Προτεινόμενο όριο D={}".format(round(dd,2)),
```

```
                                parent=self.root, icon=tk.messagebox.WARNING)
            return
        self.temp1.configure(state='normal')
        self.temp1.insert(0,round(a1,4))
        self.temp1.configure(state='readonly')
        self.temp2.configure(state='normal')
        self.temp2.insert(0,round(a2,4))
        self.temp2.configure(state='readonly')

        l1,t1,x1,y1,tl1,tk1,xm1,ym1,dr1=frac1(r2,a2)
        l2,t2,x2,y2,tl2,tk2,xm2,ym2,dr2=frac1(r2,a2)
        if self.save==False and self.simple==True:
            f=tk.filedialog.asksaveasfile(parent=self.root, defaultextension='.txt',title="Ονομάστε το
                                        αρχείο που θα περιέχει τα αποτελέσματα του απλού
                                        υπολογισμού")
            if f is None: return #User cancelled
            writetxt1(f,1,r1,a1,l1,t1,x1,y1,tl1,tk1,xm1,ym1,dr1)
            writetxt1(f,2,r2,a2,l2,t2,x2,y2,tl2,tk2,xm2,ym2,dr2)
            f.close()

        return(r1,r2,round(a1,4),round(a2,4))

    def creategrid(self,a,b,i):
        "Create the grid of the coordinates of the Scurve"
        self.temp = tk.Label(self.root, text=a, width=10)
        self.temp.grid(row=i, column=0)
        self.labels.append(self.temp)
        self.ent=tk.Entry(self.root,width=10)
        self.ent.grid(row=i,column=1)
        self.entries.append(self.ent)
        self.temp = tk.Label(self.root, text=b, width=10)
        self.temp.grid(row=i, column=2)
        self.labels.append(self.temp)
        self.ent=tk.Entry(self.root,width=10)
        self.ent.grid(row=i,column=3)
        self.entries.append(self.ent)
```

```
def showmore(self):
  "Create the new grid of the analytical computation"
  if self.simple==True:
    self.button.grid_remove()
    lab = tk.Label(self.root, text="Αναλυτικός Υπολογισμός Χαρακτηριστικών Σιγμοειδούς
                Καμπύλης",activebackground='steelblue',bg='light blue',
                width=55).grid(row=0, column=0, columnspan=5)
    ir=6
    ir+=1
    self.temp = tk.Label(self.root, text="Συντεταγμένες Πολυγωνικής",
                        activebackground='steelblue',bg='light blue', width=55)
    self.temp.grid(row=ir, column=0, columnspan=5)
    self.labels.append(self.temp)
    ir+=1
    self.creategrid("XK1=", "YK1 =", ir)
    ir+=1
    self.creategrid("XK2=", "YK2 =", ir)
    ir+=1
    self.creategrid("XK3=", "YK3 =", ir)
    ir+=1
    self.creategrid("XK4=", "YK4 =", ir)

    ir+=1
    self.button1=tk.Button(self.root, text='Υπολογισμός', command=self.calcu2,
                          activebackground='steelblue',bg='light blue')
    self.button1.grid(row=ir,column=3,sticky='we',columnspan=1)
    self.simple=False

def delete(self):
  "Delete all the labels,entries of non simple calculation"
  if self.simple==False:
    self.button.grid()
    for i in range (len(self.labels)):
      self.labels[i].grid_remove()
    j=len(self.entries)
    for i in range (3,j):
      self.entries[i].grid_remove()
    self.button1.grid_remove()
    for i in range(3,j):
      self.entries.remove(self.entries[3])
    self.simple=True
```

```
    def saveAll(self):
        "Save all data to a user specified file."
        if self.simple==False:
            self.save=True

r1,r2,d,xk1,yk1,xk2,yk2,xk3,yk3,xk4,yk4=self.entries[0].get(),self.entries[1].get(),self.entries[2].g
et(),self.entries[3].get(),self.entries[4].get(),self.entries[5].get(),self.entries[6].get(),self.entries[
7].get(),self.entries[8].get(),self.entries[9].get(),self.entries[10].get()
            fw=tk.filedialog.asksaveasfile(parent=self.root, defaultextension='.sig',title="Ονομάστε
                                        το αρχείο που θα περιέχει τα δεδομένα σας")
            if fw is None: return #User cancelled
            fw.write('{}{}{}{}{}{}{}{}{}{}{}'.format(r1,r2,d,xk1,yk1,xk2,yk2,xk3,yk3,xk4,yk4))
            fw.close()
            self.save=False
        else:
            self.save=True
            r1,r2,d=self.entries[0].get(),self.entries[1].get(),self.entries[2].get()
            fw=tk.filedialog.asksaveasfile(parent=self.root, defaultextension='.sig',title="Ονομάστε
                                        το αρχείο που θα περιέχει τα δεδομένα σας")
            if fw is None: return #User cancelled
            fw.write('{}\n{}\n{}'.format(r1,r2,d))
            fw.close()
            self.save=False

    def readAll(self):
        "Read data from user specified file"
        fr=tk.filedialog.askopenfile(parent=self.root, defaultextension='.sig',title="Επιλέξτε αρχείο
                                    για άνοιγμα")
        if fr is None: return
        dlines= list(fr)
        fr.close()
        if len(dlines)>3 and self.simple==True:
            self.simple==False
            self.showmore()
        elif len(dlines)<=3 and self.simple==False:
            self.simple=True
            self.delete()
        i=0
        for ent in self.entries:
            ent.delete(0,'end')
            ent.insert(0, dlines[i])
            i+=1
```

```
def calcu2(self):
    "Calculate the results of the analytic calculation  of the S-Curve"
    self.simple=False
    r1,r2,a1,a2=self.calcu1()
    self.simple=True
    for i in range (3,len(self.entries)):
        try: n = float(self.entries[i].get().replace(',','.'))
        except ValueError: n = None
        if n is None:
            tk.messagebox.showinfo ("Warning","Οι τιμές των συντεταγμένων πρέπει να είναι
                                    πραγματικοί αριθμοί",parent=self.root,
                                    icon=tk.messagebox.WARNING)
            return
    r1=float(self.entries[0].get().replace(',','.'))
    r2=float(self.entries[1].get().replace(',','.'))
    d=float(self.entries[2].get().replace(',','.'))

    xk1=float(self.entries[3].get().replace(',','.'))
    yk1=float(self.entries[4].get().replace(',','.'))
    xk2=float(self.entries[5].get().replace(',','.'))
    yk2=float(self.entries[6].get().replace(',','.'))
    xk3=float(self.entries[7].get().replace(',','.'))
    yk3=float(self.entries[8].get().replace(',','.'))
    xk4=float(self.entries[9].get().replace(',','.'))
    yk4=float(self.entries[10].get().replace(',','.'))
    t21=unitvector(xk2,yk2,xk1,yk1)
    t23=unitvector(xk2,yk2,xk3,yk3)
    g1=ggg(t21,t23)
    t32=unitvector(xk3,yk3,xk2,yk2)
    t34=unitvector(xk3,yk3,xk4,yk4)
    g2=ggg(t32,t34)
    dist=lend(xk2,yk2,xk3,yk3)
    l1,t1,x1,y1,xm1,ym1,dr1,d1,T1,o1,s1,m1=frac(r1,a1,g1)
    l2,t2,x2,y2,xm2,ym2,dr2,d2,T2,o2,s2,m2=frac(r2,a2,g2)
    n=checkdist(dist,T1,T2,a1,a2)
    if n==0:
        f=tk.filedialog.asksaveasfile(parent=self.root, defaultextension='.txt',title="Ονομάστε το
                                      αρχείο που θα περιέχει τα αποτελέσματα του αναλυτικού
                                      υπολογισμού")
        if f is not None:
            f=f.name
            p=Path(f)
```

```
            ff=p.with_suffix('.syn')
            fff=p.with_suffix('.dxf')
        else: ff,fff=None,None
    elif n==1:
        tk.messagebox.showinfo ("Warning","Η απόσταση μεταξύ των 2 στροφών είναι
                                μεγαλύτερη απο 0.08(A1+A2), οπότε δεν μπορούν να
                                θεωρηθούν Σιγμοειδής Καμπύλη",parent=self.root,
                                icon=tk.messagebox.WARNING)
        return
    elif n==2:
        tk.messagebox.showinfo ("Warning","Οι Καμπύλες δεν χωράνε. Άλλαξε
                                δεδομένα",parent=self.root, icon=tk.messagebox.WARNING)
        return
    pymain(xk1, yk1, xk2, yk2, xk3, yk3, xk4, yk4, r1, r2, a1, a2, f, ff, fff)

SCalculator()
```

## 7.3 Ωοειδής Καμπύλη – Κώδικας Συναρτήσεων Python

```
from math import sin, cos, pi, tan, sqrt, acos, floor, atan, asin
import matplotlib.pyplot as plt
from dxfwrite import DXFEngine as dxf

def rad2grad(x):
    "Convert angle from radians to gradians."
    return x*200/pi

def checkr(r1,r2):
    "Check R1,R2 based on their division"
    n=1
    if r2<100:
        if r1/r2<=1.5:return
        n=0
    elif r2>100 and r2<=500:
        if r1/r2<=2:return
        n=0
    return n

def fix (r1,r2):
    "Fix R1>R2"
    if r1<r2:
        temp1=r1
        r1=r2
        r2=temp1
    return r1,r2

def frangec(start, end, step=1, tol=0.0):
    """Implements float xrange - which includes start and end; tol is tolerance ratio.
    if the last step before end is within tol*step distance from end, it is skipped."""
    assert tol>=0.0, "xfrangec(): tolerance shouble a non negative ratio."
    start = float(start); end = float(end); step = float(step); tol = float(tol)
    a = start
    if step > 0:
        if start > end: return
        while a < end-tol*step:
            yield a
            a += step
    else:
        if start < end: return
```

```
        while a > end-tol*step:  #Note tol>=0 and step < 0
            yield a
            a += step
    yield end

def omega(r, a):
    "Compute coordinates of omega point."
    x =(a**2/r)-((a**2/r)**5)/(40*a**4)+((a**2/r)**9)/(3456*a**8)
    y= ((a**2/r)**3)/(6*a**2)-((a**2/r)**7)/(336*a**6)+((a**2/r)**11)/(42240*a**10)
    return x, y

def prosarm(r1, r2, d):
    "Find a so that citerion is met; s>=d is successful, s<d unsuceessful."
    for a in frangec(r1/3.0, r2, 0.05):
        x1, y1 = omega(r1, a)
        t1= ((a**2)/(2*r1**2))
        tl1=x1-(y1/tan(t1))
        tk1=y1/cos(t1)
        l1=(a**2)/r1
        xm1= x1-r1*sin(t1)
        ym1=y1+r1*cos(t1)
        dr1=ym1-r1

        x2, y2 = omega(r2, a)
        t2= ((a**2)/(2*r2**2))
        tl2=x2-(y2/tan(t2))
        tk2=y2/cos(t2)
        l2=(a**2)/r2
        xm2= x2-r2*sin(t2)
        ym2=y2+r2*cos(t2)
        dr2=ym2-r2

        s=r1-sqrt((xm2-xm1)**2+(ym2-ym1)**2)-r2
        if s >= d: break
    return a,s

def frac1(r,a):
    "Compute all parameters of klothoide"
    l=(a**2)/r
    t=l/(2*r)
    x, y=omega(r,a)
    tl=x-(y/tan(t))
    tk=y/sin(t)
    xm=x-(r*sin(t))
```

```
    ym=y+(r*cos(t))
    dr=y-(r*(1-cos(t)))
    return l,t,x,y,tl,tk,xm,ym,dr

def writetxt(f,i,r,a,l,t,x,y,tl,tk,xm,ym,dr):
    "Write the mains of a curve in a txt"
    f.write("Curve {}".format(i))
    f.write("\n R{}={}".format(i,round(r,4)))
    f.write("\n A{}={}".format(i,round(a,4)))
    f.write("\n L{}={}".format(i,round(l,4)))
    f.write("\n t{}={} grad".format(i,round(rad2grad(t),4)))
    f.write("\n x{}={}".format(i,round(x,4)))
    f.write("\n y{}={}".format(i,round(y,4)))
    f.write("\n TL{}={}".format(i,round(tl,4)))
    f.write("\n TK{}={}".format(i,round(tk,4)))
    f.write("\n Xm{}={}".format(i,round(xm,4)))
    f.write("\n Ym{}={}".format(i,round(ym,4)))
    f.write("\n DR{}={} \n\n".format(i,round(dr,4)))

def oval(t1,t2,l1,l2,tk1,tk2,tl1,tl2):
    "Compute the parametres of oval curve"
    t=t2-t1
    l=l2-l1
    tll=((tl2-tl1)*sin(t2))/sin(t)-tk1
    tkk=tk2-((tl2-tl1)*sin(t1)/sin(t))
    return tll,tkk,t,l

def unitvector(xk1,yk1,xk2,yk2):
    "Compute the unit vectors of the polygonic"
    x=xk2-xk1
    y=yk2-yk1
    z=sqrt(x**2+y**2)
    t=[x/z,y/z]
    return t

def vertical(t):
    "Compute the vertical unit vector to a unit vector"
    n=[-t[1],t[0]]
    return n
```

```
def bisector(t12,t32):
   "Find the parallel vector to the bisector"
   d=[]
   for i in range(0,2):
      d.append(-t12[i]-t32[i])
   return (d)

def vmul(a,b):
   "Compute the dot product of two vectors"
   c=a[0]*b[0]+a[1]*b[1]
   return (c)

def direction(d,n12):
   "Fix the direction of the vertical vectors towards the center of the curve"
   if vmul(d,n12)<0:
      n12=[-n12[0],-n12[1]]
   else:
      n12==n12
   return n12

def ggg(t21,t23):
   "Find the angle between 2 lines"
   b=acos(vmul(t21,t23))
   return pi-b

def frac(r,a,g):
   "Compute all parameters of klothoide"
   l=(a**2)/r
   t=l/(2*r)
   x, y=omega(r,a)
   tl=x-(y/tan(t))
   tk=y/sin(t)
   xm=x-(r*sin(t))
   ym=y+(r*cos(t))
   dr=y-(r*(1-cos(t)))
   d=((r+dr)/(cos(g/2)))-r
   T=((r+dr)*tan(g/2))+xm
   o=(g)-(l/r)
   s=r*o
   m=(l)+s
   return l,t,x,y,tl,tk,xm,ym,dr,d,T,o,s,m
```

```
def frac2(r,a,o,g):
    "Compute all parameters of oval curve"
    l=(a**2)/r
    t=l/(2*r)
    x, y=omega(r,a)
    xm=x-(r*sin(t))
    ym=y+(r*cos(t))
    dr=ym-r
    b=r*o
    T=r*tan(o/2)
    tl=x-(y/tan(t))
    tk=y/sin(t)
    return l,t,x,y,xm,ym,dr,b,T,tl,tk

def finda1a3(tk1,tk4,T1,T4,o1,o4,tk,tl,t,t1,t4,g,tl1,tl4,t12,t32,xk2,yk2):
    "Compute the coordinates of A1"
    t1prov=sqrt((tk1+T1)**2+(tl+T1)**2-2*(tk1+T1)*(tl+T1)*cos(pi-o1))
    t2prov=sqrt((tk4+T4)**2+(tk+T4)**2-2*(tk4+T4)*(tk+T4)*cos(pi-o4))
    w1=asin((tl+T1)*sin(o1)/t1prov)
    w2=asin((tk+T4)*sin(o4)/t2prov)
    t12prov=sqrt(t1prov**2+t2prov**2-2*t1prov*t2prov*cos(pi-t-o1+w1-o4+w2))
    w3=asin(t2prov*sin(pi-t-o1+w1-o4+w2)/t12prov)
    w4=asin(t1prov*sin(pi-t-o1+w1-o4+w2)/t12prov)
    TT=sin(t4+w2+w4)*t12prov/sin(g)+tl1
    TTT=sin(t1+w1+w3)*t12prov/sin(g)+tl4
    xa1,ya1=xk2-t12[0]*(TT),yk2-t12[1]*(TT)
    xa3,ya3=xk2-t32[0]*(TTT),yk2-t32[1]*(TTT)
    return xa1, ya1, xa3, ya3, TT, TTT

def
finda2(xm2,ym2,xm3,ym3,d,r2,xcircle1,ycircle1,xcircle2,ycircle2,t4,o4,t3,t32,n32,xk2,y
        k2,xk3,yk3):
    "Compute the coordinates of A2"
    dx=abs(xm3-xm2)
    dy=abs(ym2-ym3)
    e=atan(dx/dy)    # or asin(dx/m1m2)
    f=(ym3/cos(e))-r2-d
    g=ym2*tan(e)
    tc12= unitvector(xcircle1,ycircle1,xcircle2,ycircle2)
    x1,y1=xcircle2+tc12[0]*(r2+d+f),ycircle2+tc12[1]*(r2+d+f)
    a1,b1,c1=findline(xk2,yk2,xk3,yk3)
    a2,b2,c2=findline(xcircle1,ycircle1,xcircle2,ycircle2)
    x=(-c1+c2)/(a1-a2)
    y=a1*x+c1
```

```
    l=lend(x,y,x1,y1)
    ll=(sin((pi/2)-e)*l)/sin(t4+o4+t3)
    x2,y2=x-t32[0]*(ll),y-t32[1]*(ll)
    ta2s= unitvector(x1,y1,x2,y2)
    na2s= vertical(ta2s)
    dd=bisector(ta2s,t32)
    na2s= direction(dd,na2s)
    xa2,ya2=x1-ta2s[0]*(g+xm2),y1-ta2s[1]*(g+xm2)
    return (xa2,ya2,ta2s,na2s)

def coom1m2(xa1,ya1,t12,n12,xm1,dr1,r1,xa3,ya3,t32,n32,xm4,dr4,r2):
    "Compute the coordinates of M1 M2 - center of circles"
    x,y=xa1+t12[0]*(xm1),ya1+t12[1]*(xm1)
    xcircle1,ycircle1=x +(r1+dr1)*n12[0], y +(r1+dr1)*n12[1]
    x,y=xa3+t32[0]*(xm4),ya3+t32[1]*(xm4)
    xcircle2,ycircle2=x +(r2+dr4)*n32[0], y +(r2+dr4)*n32[1]
    return xcircle1,ycircle1,xcircle2,ycircle2

def fact(n):
    "Compute the factorial of a number"
    fact = 1
    for i in range(1,n+1):
        fact = fact * i
    return fact

def orosx(k,s,a):
    "Compute the k repetition of the s point of the klothoide"
    ox = (((-1)**(k/4))*(s**(k+1)))/((2**(k/2))*(k+1)*(fact(floor(k/2)))*(a**k))
    return ox

def orosy(k,s,a):
    "Compute the k repetition of the s point of the klothoide"
    oy=(((-1)**((k-2)/4))*(s**(k+1)))/((2**(k/2))*(k+1)*(fact(floor(k/2)))*(a**k))
    return oy

def kloth1(s,a):
    "Compute the coordinates of one point in klothoide"
    x=s
    y=0
    for k in range(2, 14, 4):
        y=y+orosy(k,s,a)
        x=x+orosx((k+2),s,a)
    return x,y
```

```
def kloth(l1,l2,a):
    "Create a list of the coordinates of every point in klothoide by 1m, even numbers are
      x - odd numbers are y" "ex. x[0],x[1] =x1,y1 etc"
    klothoide=[]
    for s in frangec(l1, l2, 1):
        x, y = kloth1(s,a)
        klothoide.append(x)
        klothoide.append(y)
    return klothoide

def circ1(i,xm,dr,r,t,l):
    "Compute the i repetition of the s point of the coordinates of the circle"
    x=xm+(r*(sin(t+((i-l)/r))))
    y=dr+r-(r*(cos(t+((i-l)/r))))
    return x,y

def circle(xm,dr,r,t,m,l):
    "Create a list of the coordinates of every point in circle by 1m, even numbers are x -
     odd numbers are y" "ex. x[0],x[1] =x1,y1 etc"
    circle=[]
    for i in frangec(l,m+l,1):
        x,y=circ1(i,xm,dr,r,t,l)
        circle.append(x)
        circle.append(y)
    return circle

def pvector(a,t,n,i):
    "Compute the p vector"
    x=a[i]
    y=a[i+1]
    p=[x*t[0]+y*n[0],x*t[1]+y*n[1]]
    return p

def general(a,t,n,xa,ya):
    "Convert the local x,y to the general x,y"
    c=[]
    for i in range(0,(len(a)-1),2):
        p=pvector(a,t,n,i)
        x=xa+p[0]
        y=ya+p[1]
        c.append(x)
        c.append(y)
    return c
```

```
def findline(xk2,yk2,xk3,yk3):
   "Find the line that 2 points make"
   a=(yk3-yk2)/(xk3-xk2)
   b=yk2-a*xk2
   #AX+BY+C=0#
   A,B,C=a,-1,b
   return A,B,C

def dpointline(A,B,C,xo,yo):
   "Compute the distance between a point from a line"
   d=abs(A*xo+B*yo+C)/sqrt(A**2+B**2)
   return d

def lend(xk2,yk2,xk3,yk3):
   "Calculate the distance between 2 points"
   d=sqrt(((xk2-xk3)**2)+((yk2-yk3)**2))
   return d

def plotpolyg(xk1,yk1,xk2,yk2,xk3,yk3):
   "Plot the polygonic"
   c=[xk1,yk1,xk2,yk2,xk3,yk3]
   for i in range (0,3,2):
      plt.plot(c[i], c[i+1], color='green', linestyle='solid', linewidth = 3, marker='o', \
            markerfacecolor='blue', markersize=6)
      plt.plot(c[i+2], c[i+3], color='green', linestyle='solid', linewidth = 3, marker='o', \
            markerfacecolor='blue', markersize=6)
      x_values = [c[i], c[i+2]]
      y_values = [c[i+1],c[i+3]]
      plt.plot(x_values, y_values, color='green')

def plot(xykloth1):
   "Plot the general x,y of the curve"
   x=[]
   y=[]
   for i in range(0,len(xykloth1),2):
      x.append(round(xykloth1[i],4))
      y.append(round(xykloth1[i+1],4))
      plt.plot(x, y, color='green', markersize=1)
```

```
def writetxt1(f, i,r,a,l,t,x,y,tl,tk,xm,ym,dr,T,o,b):
    "Write the details of a curve in a txt"
    f.write("Curve {}".format(i))
    f.write("\n R{}={}".format(i,round(r,4)))
    f.write("\n A{}={}".format(i,round(a,4)))
    f.write("\n L{}={}".format(i,round(l,4)))
    f.write("\n t{}={} grad".format(i,round(rad2grad(t),4)))
    f.write("\n x{}={}".format(i,round(x,4)))
    f.write("\n y{}={}".format(i,round(y,4)))
    f.write("\n TL{}={}".format(i,round(tl,4)))
    f.write("\n TK{}={}".format(i,round(tk,4)))
    f.write("\n Xm{}={}".format(i,round(xm,4)))
    f.write("\n Ym{}={}".format(i,round(ym,4)))
    f.write("\n DR{}={}".format(i,round(dr,4)))
    f.write("\n T{}={}".format(i,round(T,4)))
    f.write("\n a{}={} rad".format(i,round(o,4)))
    f.write("\n b{}={}".format(i,round(b,4)))
    f.write("\n")

def writetxt2 (f,r1,r2,a,l2,l3,t2,t3,x2,x3,y2,y3,tl2,tl3,tk2,tk3,xm2,xm3,ym2,ym3,
                    dr2,dr3,T2,T3,o2,o3,m2,m3,tl,tk,t,l,m1m2):
    "Write the details of a curve in a txt"
    f.write("\n\n\nOval Curve \n\n")
    writetxt1(f,2,r1,a,l2,t2,x2,y2,tl2,tk2,xm2,ym2,dr2,T2,o2,m2)
    f.write("\n Main Curve")
    f.write("\n A={}".format(round(a,4)))
    f.write("\n TL={}".format(round(tl,4)))
    f.write("\n TK={}".format(round(tk,4)))
    f.write("\n t={} grad".format(round(rad2grad(t),4)))
    f.write("\n L={}".format(round(l,4)))
    f.write("\n M1M2={} \n\n".format(round(m1m2,4)))
    writetxt1(f,3,r2,a,l3,t3,x3,y3,tl3,tk3,xm3,ym3,dr3,T3,o3,m3)
    f.write("\n")

def writecoordinatesF(ff,xykloth1,k):
    "Write the Coordinates of the Oval Curve"
    for i in range (2,len(xykloth1)-2,2):
        x=xykloth1[i]
        y=xykloth1[i+1]
        ff.write('P{} = {}, {} \n'.format(k,round(xykloth1[i],4),round(xykloth1[i+1],4)))
        k+=1
    return k
```

```
def writecoordinatesB(ff,xykloth1,k):
    "Write the Coordinates of the S Curve Backwards"
    for i in range (len(xykloth1)-3,2,-2):
        ff.write('P{} = {}, {} \n'.format(k,round(xykloth1[i-1],4),round(xykloth1[i],4)))
        k+=1
    return k

def tex(drawing, tex,poin):
    "Write text on dxf"
    text = dxf.text(tex, poin, height=15, rotation=0)
    text['color'] = 7
    drawing.add(text)
    drawing.add(dxf.point((poin)))

def pointsF(polyline, t):
    "Creates polyline of the kloth or circle  Forwards for dxf file"
    for i in range(0,len(t)-1,2):
        c=(t[i],t[i+1])
        polyline.add_vertex(c)

def pointsB(polyline, t):
    "Creates polyline of the kloth or circle  Backwards for dxf file"
    for i in range(len(t)-1,0,-2):
        c=(t[i-1],t[i])
        polyline.add_vertex(c)

def auto(fn, xk1,yk1,xa1,ya1,xykloth1,xycircle1,xykloth2,xycircle2,
            xykloth3,xa3,ya3,xk3,yk3,x1,y1):
    "Writes all  dxf"
    drawing = dxf.drawing(fn)
    polyline= dxf.polyline(linetype='CONTINUOUS')
    tex(drawing, "K1",(xk1, yk1))
    drawing.add(dxf.line((xk1, yk1), (xa1, ya1), color=7))
    tex(drawing, "A1",(xa1, ya1))
    pointsF(polyline, xykloth1)
    tex(drawing, 'Ω1',(xycircle1[0],xycircle1[1]))
    pointsF(polyline, xycircle1)
    tex(drawing, "Ω1'",(xycircle1[-2],xycircle1[-1]))
    pointsF(polyline, xykloth2)
    tex(drawing, "Ω2'",(xycircle2[-2],xycircle2[-1]))
    pointsB(polyline, xycircle2)
    tex(drawing, "Ω2",(xycircle2[0],xycircle2[1]))
    pointsB(polyline, xykloth3)
    tex(drawing, "A3",(xa3, ya3))
```

```
    drawing.add(dxf.line((xa3, ya3), (xk3, yk3), color=7))
    tex(drawing, "K3",(xk3, yk3))
    tex(drawing, "xy1",(x1, y1))
    drawing.add(polyline)
    drawing.save()

def pymain(xk1,yk1,xk2,yk2,xk3,yk3,r1,r2,a1,a2,b1,d,f,ff,fff):
    'main program run'
    a,d=prosarm(r1,r2,d)

    #find the vectors of K1K2 K2K3
    t12= unitvector(xk1,yk1,xk2,yk2)
    t32= unitvector(xk3,yk3,xk2,yk2)
    n12= vertical(t12)
    n32= vertical(t32)
    dd=bisector(t12,t32)
    n12= direction(dd,n12)
    n32= direction(dd,n32)

    #find angle between (K1K2) (K2K3)
    t21=unitvector(xk2,yk2,xk1,yk1)
    t23= unitvector(xk2,yk2,xk3,yk3)
    g=ggg(t21,t23)

    #compute details of oval curve
    l2,t2,x2,y2,tl2,tk2,xm2,ym2,dr2,d2,T2,o2,s2,m2=frac(r1,a,g)
    l3,t3,x3,y3,tl3,tk3,xm3,ym3,dr3,d3,T3,o3,s3,m3=frac(r2,a,g)
    tl,tk,t,l=oval(t2,t3,l2,l3,tk2,tk3,tl2,tl3)

    #compute details of klothoide and circles 1,4
    o1=(b1/r1)
    l1,t1,x1,y1,xm1,ym1,dr1,b1,T1,tl1,tk1=frac2(r1,a1,o1,g)
    o4=g-t1-((a2**2)/(2*r2**2))-t-o1
    l4,t4,x4,y4,xm4,ym4,dr4,b4,T4,tl4,tk4=frac2(r2,a2,o4,g)

    #compute x,y of A1 M1 M2 A2 A3
    xa1,ya1,xa3,ya3,TT,TTT=finda1a3(tk1,tk4,T1,T4,o1,o4,tk,tl,t,t1,t4,g,tl1,tl4,t12,t32,
                                    xk2,yk2)
    m1m2=-d+r1-r2
    xcircle1,ycircle1,xcircle2,ycircle2=coom1m2(xa1,ya1,t12,n12,xm1,dr1,r1,xa3,ya3,t32,
                                                n32,xm4,dr4,r2)
    xa2,ya2,ta2s,na2s=finda2(xm2,ym2,xm3,ym3,d,r2,xcircle1,ycircle1,xcircle2,ycircle2,
                             t4,o4,t3,t32,n32,xk2,yk2,xk3,yk3)
```

```
#compute x,y of klothoide1 and circle1
klothoide1=kloth(0,l1,a1)
xykloth1=general(klothoide1,t12,n12,xa1,ya1)
circle1=circle(xm1,dr1,r1,t1,b1,l1)
xycircle1=general(circle1,t12,n12,xa1,ya1)

#compute x,y of klothoide2-oval curve
klothoide2=kloth(l2,l3,a)
xykloth2=general(klothoide2,ta2s,na2s,xa2,ya2)

#compute x,y of klothoide3 and circle2
klothoide3=kloth(0,l4,a2)
xykloth3=general(klothoide3,t32,n32,xa3,ya3)
circle2=circle(xm4,dr4,r2,t4,b4,l4)
xycircle2=general(circle2,t32,n32,xa3,ya3)

#Write txt file with the Details of the Oval Curve
if not(f==None) :
   f= open(f, "w")
   writetxt1(f,1,r1,a1,l1,t1,x1,y1,tl1,tk1,xm1,ym1,dr1,T1,o1,b1)
   writetxt2(f,r1,r2,a,l2,l3,t2,t3,x2,x3,y2,y3,tl2,tl3,tk2,tk3,xm2,xm3,ym2,ym3,dr2,dr3,
             T2,T3,o2,o3,m2,m3,tl,tk,t,l,m1m2)
   writetxt1(f,4,r2,a2,l4,t4,x4,y4,tl4,tk4,xm4,ym4,dr4,T4,o4,b4)
   f.close()

#write txt file with the Coordinates of the Oval Curve
if not(ff==None) :
   ff= open(ff, "w")
   ff.write('Coordinates of Oval Curve \n')
   ff.write('K1 = {}, {} \n'.format(round(xk1,4),round(yk1,4)))
   ff.write('A1 = {}, {} \n'.format(round(xa1,4),round(ya1,4)))
   k=1
   k=writecoordinatesF(ff,xykloth1,k)
   ff.write('O1 = {}, {} \n'.format(round(xycircle1[0],4),round(xycircle1[1],4)))
   k=writecoordinatesF(ff,xycircle1,k)
   ff.write("O'1 = {}, {} \n".format(round(xycircle1[-2],4),round(xycircle1[-1],4)))
   k=writecoordinatesF(ff,xykloth2,k)
   ff.write("O'2 = {}, {} \n".format(round(xycircle2[-2],4),round(xycircle2[-1],4)))
   k=writecoordinatesB(ff,xycircle2,k)
   ff.write('O2 = {}, {} \n'.format(round(xycircle2[0],4),round(xycircle2[1],4)))
   k=writecoordinatesB(ff,xykloth3,k)
   ff.write('A3 = {}, {} \n'.format(round(xa3,4),round(ya3,4)))
   ff.write('K3 = {}, {} \n'.format(round(xk3,4),round(yk3,4)))
   ff.close()
```

```
    #write dxf file with the Coordinates of the Oval Curve
    if not(fff == None) :
       auto(fff, xk1,yk1,xa1,ya1,xykloth1,xycircle1,xykloth2,
            xycircle2,xykloth3,xa3,ya3,xk3,yk3,x1,y1)

    #Plot the Oval Curve
    plt.title(label='Ωωειδής Καμπύλη', fontdict=None, loc='center', pad=None)
    plotpolyg(xk1,yk1,xk2,yk2,xk3,yk3)
    plt.plot(xa1,ya1, color='green', linestyle='dashed', linewidth = 3, marker='o', \
             markerfacecolor='blue', markersize=6)
    plot(xykloth1)
    plt.plot(xycircle1[0], xycircle1[1], color='green', linestyle='dashed', linewidth = 3, \
             marker='o', markerfacecolor='blue', markersize=6)
    plot(xycircle1)
    plt.plot(xycircle1[-2], xycircle1[-1], color='green', linestyle='dashed', linewidth = 3, \
             marker='o', markerfacecolor='blue', markersize=6)
    plot(xykloth2)
    plt.plot(xycircle2[-2], xycircle2[-1], color='green', linestyle='dashed', linewidth = 3, \
             marker='o', markerfacecolor='blue', markersize=6)
    plot(xycircle2)
    plt.plot(xycircle2[0], xycircle2[1], color='green', linestyle='dashed', linewidth = 3, \
             marker='o', markerfacecolor='blue', markersize=6)
    plot(xykloth3)
    plt.plot(xa3,ya3, color='green', linestyle='dashed', linewidth = 3, marker='o', \
             markerfacecolor='blue', markersize=6)
    plt.show()

def test():
    pymain(xk1,yk1,xk2,yk2,xk3,yk3,r1,r2,a1,a2,b1,d,f,ff,fff)
```

## 7.4 Ωοειδής Καμπύλη – Κώδικας Γραφικού Περιβάλλοντος (GUI) Python

```
import tkinter as tk
from tkinter import font as tkfo
from math import sin, cos, pi, tan, sqrt, acos, floor, atan, asin
import tkinter.messagebox
from DefinitionsOvalCurve import rad2grad,checkr,fix,prosarm,frac1,writetxt,oval,pymain
from tkinter import filedialog
from pathlib import Path

class OvalCalculator:
    'Creates a Spread Sheet'
    def __init__(self):
        self.makeWidgets()

    def setMenus(self):
        "Create the menu system."
        menuBar = tk.Menu(self.root, activebackground="green")
        menu = tk.Menu(menuBar, activebackground="green", tearoff=False)
        menuBar.add_cascade(label="Αρχείο", menu=menu, foreground="blue")
        self.root.config(menu=menuBar)
        menu.add_command(label="Αποθήκευση", command=self.saveAll, foreground="blue")
        menu.add_command(label="Ανοιγμα", command=self.readAll, foreground="blue")
        computemenu = tk.Menu(menuBar, activebackground="green", tearoff=False)
        menuBar.add_cascade(label="Υπολογισμός", menu=computemenu, foreground="blue")
        computemenu.add_command(label="Αναλυτικός υπολογισμός",
                                command=self.showmore, foreground="blue")
        computemenu.add_command(label="Απλός υπολογισμός", command=self.delete,
                                foreground="blue")

    def setFonts(root):
        for t in 'TkDefaultFont', 'TkFixedFont', 'TkTextFont':
            font1 = tk.font.Font(name=t, exists=True)
            font1.config(size=15)
            font1.config(family='Arial')

    def makeWidgets(self):
        'make the widgets'
        self.root = tk.Tk()
        self.root.title('Oval-Curve Calculator')
        self.setFonts()
```

```
    self.setMenus()
    self.save=False
    self.simple=True

    temp = tk.Label(self.root, text="Απλός Υπολογισμός Χαρακτηριστικών Ωοειδούς
                        Καμπύλης",activebackground='steelblue',bg='light blue',
                         width=55).grid(row=0, column=0, columnspan=5)

    temp = tk.Label(self.root, text="R1 =",width=10)
    temp.grid(row=1, column=0)
    self.ent1 = tk.Entry(self.root,width=10)
    self.ent1.grid(row=1,column=1)

    temp = tk.Label(self.root, text="R2 =",width=10).grid(row=2, column=0)
    self.ent2=tk.Entry(self.root,width=10)
    self.ent2.grid(row=2,column=1)

    temp = tk.Label(self.root, text="Δ(R1R2) =",width=10).grid(row=3, column=0)
    self.ent3=tk.Entry(self.root,width=10)
    self.ent3.grid(row=3,column=1)

    temp = tk.Label(self.root, text="A =",width=10).grid(row=4, column=0)
    self.temp1 = tk.Entry(self.root, width=10)
    self.temp1.grid(row=4, column=1)
    self.temp1.configure(justify='center',state='readonly')
    self.button1=tk.Button(self.root, text='Υπολογισμός', activebackground='steelblue',
                                  bg='light blue', command=self.calcu1, justify='center')
    self.button1.grid(row=4,column=2,sticky='we',columnspan=1)

    self.root.mainloop()


def calcu1(self):
    "Calculate the results of the Simple Oval Curve"
    if self.simple==False:
        self.ent1.delete(0,'end')
        self.ent1.insert(0,self.entries[0].get())
        self.ent2.delete(0,'end')
        self.ent2.insert(0,self.entries[2].get())
        self.ent3.delete(0,'end')
        self.ent3.insert(0,self.entries[4].get())
        self.temp1.configure(state='normal')
        self.temp1.delete(0,'end')
        self.temp1.configure(state='readonly')
```

```
try: n = float(self.ent1.get().replace(',','.'))
except ValueError: n = None
if n is None or n <= 0:
   tk.messagebox.showinfo("Error", "R1 <= 0 ", parent=self.root,
                                 icon=tk.messagebox.ERROR)
   return
try: n = float(self.ent2.get().replace(',','.'))
except ValueError: n = None
if n is None or n <= 0:
   tk.messagebox.showinfo("Error", "R2 <= 0 ", parent=self.root,
                                 icon=tk.messagebox.ERROR)
   return
try: n = float(self.ent3.get().replace(',','.'))
except ValueError: n = None
if n is None or n <= 0:
   tk.messagebox.showinfo("Error", "Δ(R1R2) <= 0 ", parent=self.root,
                                 icon=tk.messagebox.ERROR)
   return
r1=float(self.ent1.get().replace(',','.'))
r2=float(self.ent2.get().replace(',','.'))
d=float(self.ent3.get().replace(',','.'))
r1,r2=fix(r1,r2)
self.ent1.delete(0,'end')
self.ent1.insert(0,r1)
self.ent2.delete(0,'end')
self.ent2.insert(0,r2)
self.ent3.delete(0,'end')
self.ent3.insert(0,d)

try: n=checkr(r1,r2)
except ValueError: n= 0
if r2<100:
   if r1/r2>1.5:
      tk.messagebox.showinfo ("Error","R2={}<100 οπότε πρέπει R1<=1.5*R2 \
                                 ={}".format(r2,1.5*r2), parent=self.root,
                                 icon=tk.messagebox.ERROR)
      return
elif r2>100  and r2<=500:
   if r1/r2>2:
      tk.messagebox.showinfo ("Error","R2={}<500 οπότε πρέπει R1<=2*R2 \
                                 ={}".format(r2,2*r2),parent=self.root,
                                icon=tk.messagebox.ERROR)
      return
a,d=prosarm(r1,r2,d)
```

```
    if not(round(d,2)==(float(self.ent3.get()))):
        tk.messagebox.showinfo ("Warning","Προτεινόμενο d={}".format(round(d,2)),
                                parent=self.root, icon=tk.messagebox.WARNING)
        self.ent3.delete(0,'end')
        self.ent3.insert(0,round(d,2))
    self.temp1.configure(state='normal')
    self.temp1.delete(0,'end')
    self.temp1.insert(0,round(a,4))
    self.temp1.configure(state='readonly')


    f= open("SimpleOvalResults.txt","w+")
    l2,t2,x2,y2,tl2,tk2,xm2,ym2,dr2=frac1(r1,a)
    l3,t3,x3,y3,tl3,tk3,xm3,ym3,dr3=frac1(r2,a)
    writetxt(f,2,r1,a,l2,t2,x2,y2,tl2,tk2,xm2,ym2,dr2)
    writetxt(f,3,r2,a,l3,t3,x3,y3,tl3,tk3,xm3,ym3,dr3)
    tll,tkk,t,l=oval(t2,t3,l2,l3,tk2,tk3,tl2,tl3)

    f.write("Oval curve")
    f.write("\n A={}".format(round(a,4)))
    f.write("\n TL={}".format(round(tll,4)))
    f.write("\n TK={}".format(round(tkk,4)))
    f.write("\n t={} grad".format(round(rad2grad(t),4)))
    f.write("\n L={}".format(round(l,4)))
    m1m2=-d+r1-r2
    f.write("\n M1M2={} \n\n\n".format(round(m1m2,4)))
    if self.save==False and self.simple==True:
        tk.messagebox.showinfo ("INFO", "Τα αποτελέσματα του απλού υπολογισμού
                                βρίσκονται στο αρχείο SimpleOvalResults.txt",
                                parent=self.root)
    return(r1,r2,round(a,4),round(d,2))

def creategrid(self,a,b,i):
    'Create grid'
    self.temp = tk.Label(self.root, text=a, width=10)
    self.temp.grid(row=i, column=0)
    self.labels.append(self.temp)
    self.ent=tk.Entry(self.root,width=10)
    self.ent.grid(row=i,column=1)
    self.entries.append(self.ent)
    self.temp = tk.Label(self.root, text=b, width=10)
    self.temp.grid(row=i, column=2)
    self.labels.append(self.temp)
    self.ent=tk.Entry(self.root,width=10)
```

```
    self.ent.grid(row=i,column=3)
    self.entries.append(self.ent)

def showmore(self):
    "Create the new grid of the analytical computation"
    if self.simple==True:
        self.entries=[]
        self.labels=[]
        self.button1.grid_remove()
        ic=0
        temp = tk.Label(self.root, text="Αναλυτικός Υπολογισμός Χαρακτηριστικών Ωοειδούς \
                                Καμπύλης",activebackground='steelblue',bg='light blue',
                                 width=55).grid(row=0, column=0, columnspan=5)
        ic+=1
        self.creategrid("R1 =", "A1 =", ic)
        ic+=1
        self.creategrid("R2 =", "A2 =", ic)
        ic+=1
        self.creategrid("Δ(R1R2) =", "b1 =", ic)


        ic+=1
        self.temp = tk.Label(self.root, text="Συντεταγμένες Πολυγωνικής",
                                    activebackground='steelblue',bg='light blue', width=55)
        self.temp.grid(row=ic, column=0, columnspan=5)
        self.labels.append(self.temp)
        ic+=1
        self.creategrid("XK1 =", "YK1 =", ic)
        ic+=1
        self.creategrid("XK2 =", "YK2 =", ic)
        ic+=1
        self.creategrid("XK3 =", "YK3 =", ic)

        ic+=1
        self.button2=tk.Button(self.root, text='Υπολογισμός', command=self.calcu2,
                                       activebackground='steelblue',bg='light blue')
        self.button2.grid(row=ic,column=3,sticky='we',columnspan=1)
        self.simple=False
```

```
def delete(self):
   "Delete all the labels,entries of non simple calculation"
   if self.simple==False:
      for i in range (len(self.labels)):
         self.labels[i].grid_remove()
      for i in range (len(self.entries)):
         self.entries[i].grid_remove()
      self.button2.grid_remove()
      self.button1.grid()
      self.simple=True

def saveAll(self):
   "Save all data to a user specified file."
   self.save=True
   if self.simple==False:
      fw=tk.filedialog.asksaveasfile(parent=self.root, defaultextension='.ova',title="Ονομάστε
                                        το αρχείο που θα περιέχει τα δεδομένα σας")
      if fw is None: return #User cancelled
      for i in range(0,12):
         fw.write('{}'.format(self.entries[i].get()))
   elif self.simple==True:
      r1,r2,d=self.ent1.get(),self.ent2.get(),self.ent3.get()
      fw=tk.filedialog.asksaveasfile(parent=self.root, defaultextension='.ova',title="Ονομάστε
                                        το αρχείο που θα περιέχει τα δεδομένα σας")
      if fw is None: return #User cancelled
      fw.write('{}\n{}\n{}'.format(r1,r2,d))
   self.save=False
   fw.close()

def readAll(self):
   "Read data from user specified file"
   fr=tk.filedialog.askopenfile(parent=self.root, defaultextension='.ova',title="Please select
                                     file to open")
   if fr is None: return
   dlines= list(fr)
   fr.close()
   if len(dlines)>3:
      if self.simple==True:
         self.showmore()
      for i in range(0,12):
         self.entries[i].delete(0,'end')
         self.entries[i].insert(0, dlines[i])
```

```
    elif len(dlines)<=3:
        if self.simple==False:
            self.delete()
        self.ent1.delete(0,'end')
        self.ent1.insert(0, dlines[0])
        self.ent2.delete(0,'end')
        self.ent2.insert(0, dlines[1])
        self.ent3.delete(0,'end')
        self.ent3.insert(0, dlines[2])

def calcu2(self):
    "Calculate the results of the analytical Oval Curve"
    r1,r2,a,d,xk1,yk1,xk2,yk2,xk3,yk3,a1,a2,b1= self.getvals()

    f=tk.filedialog.asksaveasfile(parent=self.root, defaultextension='.txt',title="Ονομάστε το
                                  αρχείο που θα περιέχει τα αποτελέσματα του αναλυτικού
                                  υπολογισμού")
    if f is not None:
        f=f.name
        p=Path(f)
        ff=p.with_suffix('.syn')
        fff=p.with_suffix('.dxf')
    else: ff,fff=None,None
    pymain(xk1,yk1,xk2,yk2,xk3,yk3,r1,r2,a1,a2,b1,d,f,ff,fff)

def getvals(self):
    "Check if entries are correct and get the values"
    trap=False
    for i in range (6,11):
        try: n = float(self.entries[i].get().replace(',','.'))
        except ValueError: n = None
        if n is None:
            if trap==False:
                tk.messagebox.showinfo ("Warning","Οι τιμές των συντεταγμένων πρέπει να είναι
                                        πραγματικοί αριθμοί",parent=self.root,
                                        icon=tk.messagebox.WARNING)
                trap=True
            return
    for i in range (0,5):
        try: n = float(self.entries[i].get().replace(',','.'))
        except ValueError: n = None
        if n is None or n<=0:
            if trap==False:
                tk.messagebox.showinfo ("Warning","Οι τιμές των R1,A1,R2,A2,Δ(R1R2),b1 πρέπει
```

```
                                        να είναι θετικές",parent=self.root,
                                        icon=tk.messagebox.WARNING)
            trap=True
        return
    r1=float(self.entries[0].get().replace(',','.'))
    r2=float(self.entries[2].get().replace(',','.'))
    d=float(self.entries[4].get().replace(',','.'))
    self.simple=False
    r1,r2,a,d=self.calcu1()
    self.simple=True
    self.entries[0].delete(0,'end')
    self.entries[0].insert(0,r1)
    self.entries[2].delete(0,'end')
    self.entries[2].insert(0,r2)
    self.entries[4].delete(0,'end')
    self.entries[4].insert(0,d)

    xk1=float(self.entries[6].get().replace(',','.'))
    yk1=float(self.entries[7].get().replace(',','.'))
    xk2=float(self.entries[8].get().replace(',','.'))
    yk2=float(self.entries[9].get().replace(',','.'))
    xk3=float(self.entries[10].get().replace(',','.'))
    yk3=float(self.entries[11].get().replace(',','.'))
    a1=float(self.entries[1].get().replace(',','.'))
    a2=float(self.entries[3].get().replace(',','.'))
    b1=float(self.entries[5].get().replace(',','.'))
    return r1,r2,a,d,xk1,yk1,xk2,yk2,xk3,yk3,a1,a2,b1

OvalCalculator()
```