ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΗΣ

ΑΝΑΠΤΥΞΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΓΡΑΦΙΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ ΔΙΕΠΑΦΗΣ ΓΙΑ ΤΗΝ ΕΚΤΕΛΕΣΗ ΕΡΓΑΣΤΗΡΙΑΚΩΝ ΑΣΚΗΣΕΩΝ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ ΜΕ ΕΝΑ ΣΥΣΤΗΜΑ 3 ΔΕΞΑΜΕΝΩΝ

υπό

Παπαγεωργίου Ιωάννα

Χανιά, 2012

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 ΕΙΣΑΓΩΓΗ

Στην παρούσα πτυχιακή εργασία παρουσιάζεται η μοντελοποίηση ενός πραγματικού συστήματος 3 δεξαμενών για τη διδασκαλία συστημάτων ελέγχου που έχει κατασκευαστεί στο Τμήμα Ηλεκτρονικής του ΤΕΙ Κρήτης σε περιβάλλον προσομοίωσης εικονικής πραγματικότητας. Αναπτύχθηκαν πρωτότυπα περιβάλλοντα διεπαφής που επιτρέπουν την προσομοίωση του φυσικού συστήματος και την ταυτόχρονη γραφική απεικόνιση της συμπεριφοράς του.

Η πτυχιακή εργασία δομείται ως εξής:

Στο 2^0 κεφάλαιο, παρουσιάζονται διάφορα συστήματα δεξαμενών που έχουν αναπτυχθεί για τη διδασκαλία συστημάτων αυτομάτου ελέγχου.

Στο 3^0 κεφάλαιο, περιγράφεται ο τρόπος ανάπτυξης αντικειμένων και εικονικών κόσμων με τη βοήθεια της γλώσσας VRML, καθώς και ο τρόπος σύνδεσής τους με το λογισμικό MATLAB.

Στο 4^0 κεφάλαιο, παρουσιάζονται τα διαφορετικά γραφικά περιβάλλοντα που έχουν αναπτυχθεί με σκοπό την κατανόηση και τη μελέτη διαφόρων θεωρητικών θεμάτων και εννοιών στα συστήματα αυτομάτου ελέγχου.

ΚΕΦΑΛΑΙΟ 2

ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ ΔΕΞΑΜΕΝΩΝ ΚΑΙ ΕΚΠΑΙΔΕΥΤΙΚΑ ΛΟΓΙΣΜΙΚΑ ΕΛΕΓΧΟΥ

2.1 ΕΙΣΑΓΩΓΗ

Η εκπαίδευση νέων μηχανικών στα Συστήματα Αυτομάτου Ελέγχου, είναι δυνατόν να γίνει με ένα συνδυασμό τεχνικών που βασίζονται τόσο στην προσομοίωση όσο και στη χρήση πραγματικών διατάξεων. Ενδεικτικές εκπαιδευτικές διατάξεις επιτρέπουν τον έλεγχο διαφορετικών φυσικών παραμέτρων μεταβλητών όπως η στάθμη, η θερμοκρασία, η πίεση, η ταχύτητα, κ.α.. Μία εξαιρετικά δημοφιλής διάταξη για τη μελέτη και τη διδασκαλία συστημάτων αυτομάτου ελέγχου είναι και τα συστήματα ελέγχου στάθμης υγρών με τη χρήση μίας ή και περισσότερων δεξαμενών. Με τη μελέτη ενός τέτοιου συστήματος ο φοιτητής μπορεί να αποκομίσει γνώσεις για τα συστήματα ελέγχου τόσο σε θεωρητικό όσο και σε πρακτικό επίπεδο, αφού για τη σωστή λειτουργία του, απαιτείται η επιλογή των κατάλληλων υλικών (hardware) αλλά και του κατάλληλου λογισμικού (software).

Στο κεφάλαιο αυτό, αναλύονται κάποιες από αυτές τις πολυάριθμες επιλογές συστημάτων ελέγχου δεξαμενών. Παρουσιάζονται δύο αξιόπιστα και φιλικά προς τον εκάστοτε χρήστη παραδείγματα, που του δίνουν τη δυνατότητα να εισάγει τις τιμές των μεταβλητών και να μελετήσει τα αποτελέσματα και την συμπεριφορά του συστήματος. Με αυτόν τον τρόπο κατανοεί καλύτερα τις έννοιες που σχετίζονται με τα συστήματα αυτομάτου ελέγχου.

2.2 ΣΥΣΤΗΜΑΤΑ ΕΛΕΓΧΟΥ ΣΤΗΝ ΕΚΠΑΙΔΕΥΣΗ

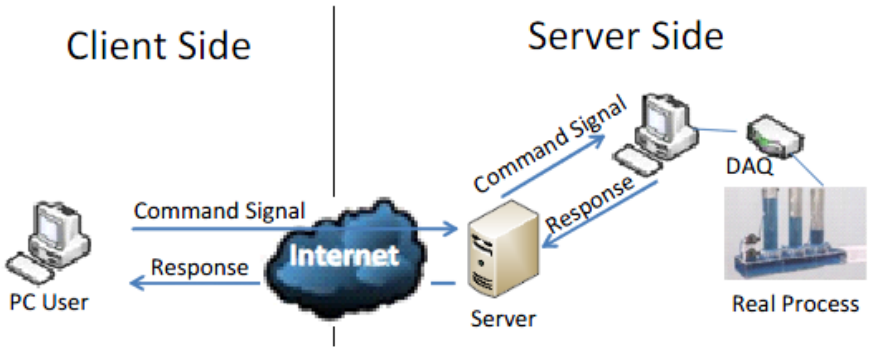
Υπάρχουν πολλών ειδών συστήματα ελέγχου που χρησιμοποιούνται για εκπαιδευτικούς σκοπούς και διαφέρουν μεταξύ τους ως προς την ακρίβεια των μετρήσεων, το περιβάλλον εργασίας (Interface) αλλά και τον τρόπο με τον οποίο παρουσιάζουν τα

αποτελέσματα της διαδικασίας ελέγχου. Ενδεικτικά στην συνέχεια παρουσιάζονται τα ακόλουθα συστήματα:

Το πρότυπο συστήματος εκπαίδευσης Πανεπιστημίου τριών δεξαμενών λαμβάνοντας υπόψιν την ανάγκη δημιουργίας εικονικών εργαστηρίων αλλά και ευέλικτες πλατφόρμες κατάλληλες για πειράματα ελέγχου, σχεδιάστηκε ένα νέο σύστημα μάθησης, η συνδυαστική εκπαίδευση (Blended learning ή πιο απλά b-learning), με πλούσιο εκπαιδευτικό υλικό, πρακτικό, ευέλικτο και εύκολο στην πρόσβαση αφού οι φοιτητές μπορούν να πραγματοποιήσουν τα πειράματά τους μέσω ενός ηλεκτρονικού υπολογιστή συνδεδεμένο στο Internet ή σε ένα τοπικό δίκτυο Ethernet (LAN), για να έχουν πρόσβαση είτε στα εικονικά είτε στα πραγματικά συστήματα.

Τα πειράματα πραγματοποιούνται με τη χρήση πραγματικών συστημάτων που ελέγχονται μέσω ενός φυλλομετρητή ιστού (Web Browser) και με τη βοήθεια μίας πλατφόρμας. Ο χρήστης έχει πλέον τη δυνατότητα, στο εικονικό και ειδικά διαμορφωμένο εργαστήριο να κάνει ρυθμίσεις και να παρατηρήσει τη συμπεριφορά και την απόδοση του συστήματος που μελετά, είτε πρόκειται για πραγματικά, είτε για εικονικά πειράματα.

Τα εκπαιδευτικά συστήματα του Πανεπιστημίου της Coimbra που αναλύονται παρακάτω, υλοποιούνται με τη βοήθεια μίας ειδικής πλατφόρμας RVL@DEI-UC, βασισμένης στο διαδίκτυο προσφέροντας στο χρήστη μεγάλη άνεση αλλά και δυνατότητα να ρυθμίζει και να παρατηρεί τη συμπεριφορά του συστήματος ελέγχου, όπως άλλωστε και τα περισσότερα συστήματα αυτού του είδους. Το νέο αυτό υβριδικό σύστημα προσφέρει πολύ μεγάλη δύναμη στο χρήστη, δεδομένου ότι μπορεί να ορίσει τους ελεγκτές του ο ίδιος ακόμη και αν πρόκειται για σύνθετους χωρίς να επηρεάζεται ο διακομιστής. Οι παράμετροι του ελεγκτή είναι συντονισμένοι σε απευθείας σύνδεση από το χρήστη, ενώ ο διακομιστής (Server) μεταδίδει την απόκριση του συστήματος. Η αρχιτεκτονική του RLV@DEI-UC βασίζεται στην αλληλεπίδραση χειριστή και διακομιστή (Σχήμα 2.1) και αποτελείται από τέσσερις βασικές εφαρμογές. Ο πελάτης εγκαθιστά την εφαρμογή στον ηλεκτρονικό του υπολογιστή και συνδέεται με το διακομιστή του εργαστηρίου στο οποίο εκτελείται η διαδικασία ελέγχου. [1]



Σχήμα 2.1 Αλληλεπίδραση πελάτη (Client) και διακομιστή (Server) κατά το έλεγχο πειραματικής διαδικασίας

Τα αποτελέσματα που δέχονται υλοποιούνται σε ένα τρίτο υπολογιστή που καθορίζει την αλληλεπίδρασή τους είτε στο πραγματικό σύστημα είτε στο εικονικό. Αυτό πραγματοποιείται με χρήση του λογισμικού JAVA ή του MATLAB μέσω συναρτήσεων που καθιστούν εφικτή την επικοινωνία μεταξύ εισόδου και εξόδου (Input-Output) με κάρτες τύπου DAQ.

Στην περίπτωση που η πλατφόρμα έχει ρυθμιστεί ως το εικονικό εργαστήριο, δηλαδή το εικονικό περιβάλλον είναι αυτό που μελετάται, τότε χρησιμοποιείται το μοντέλο προσομοίωσης (Simulation Model) του MATLAB και του Real-Time Toolbox. Στη συνέχεια ο χρήστης λαμβάνει από το διακομιστή τα πραγματικά και εικονικά συστήματα, επιλέγει αυτό που επιθυμεί και προχωρά στη ρύθμιση των παραμέτρων του ελεγκτή που τον ενδιαφέρει, όπως το χρόνο δειγματοληψίας, το συνολικό χρόνο που θα διαρκέσει το πείραμα και το είδος του κύκλου ελέγχου. Όσο το πείραμα βρίσκεται σε εξέλιξη ο χρήστης λαμβάνει τις τιμές των παραμέτρων που εξετάζει χωρίς να επηρεάζεται η διαδικασία εκτέλεσης του πειράματος.

Για να χρησιμοποιηθεί το RVL, θα πρέπει ο χρήστης να συνδεθεί στην ιστοσελίδα της πλατφόρμας (website) μέσω της οποίας καθορίζεται ή επιλέγεται ο τύπος της διαδικασίας (πραγματικό ή εικονικό) και οι διαθέσιμες διεργασίες της βάσης δεδομένων ενώ παράλληλα υποδεικνύει εκείνες που είναι έτοιμες προς χρήση.

Σχήμα 2.2 Περιβάλλον απεικόνισης ελέγχου συστήματος τριών δεξαμενών

Κατά τη διάρκεια του ελέγχου του συστήματος των δεξαμενών, ο χρήστης έχει πρόσβαση σε ένα παράθυρο διαλόγου από το οποίο μπορεί να παρακολουθεί τα

γραφήματα που προκύπτουν αλλά και τις τιμές των μεταβλητών, όπως των εξόδων του συστήματος. Όπως φαίνεται στο Σχήμα 2.2, το περιβάλλον απεικόνισης του ελέγχου προσφέρει στο χρήστη την αριθμητική αναπαράσταση των σημάτων που συνδέονται με τη διαδικασία και το δίκτυο επικοινωνίας, γραφική απεικόνιση, σε πραγματικό χρόνο, των εισόδων και των εξόδων, επιτρέποντας έτσι την παρατήρηση και την ανάλυση της προόδου της διαδικασίας ελέγχου. Επιπλέον, παρουσιάζει την εικονική αναπαράσταση του συστήματος, τις τιμές που σχετίζονται με αυτή τη διαδικασία και τον έλεγχό της σε ζωντανή μετάδοση βίντεο (δισδιάστατης ή τρισδιάστατης μορφής) αλλά και ήχου.

Η εικονική αναπαράσταση του συστήματος ελέγχου και τα αποτελέσματα που προκύπτουν είναι σημαντικά για το εικονικό περιβάλλον όπου η προσομοίωση γίνεται σε τρισδιάστατη μορφή.

Μεγάλος αριθμός ελέγχου είτε πραγματικών είτε εικονικών συστημάτων μπορούν να πραγματοποιηθούν χρησιμοποιώντας την RVL @ DEI-UC πλατφόρμα και συνήθως αναλύονται και προσομοιώνονται στο περιβάλλον του Matlab με τη βοήθεια του Simulink. Ενδεικτικά αναφέρονται τα παρακάτω παραδείγματα:

Το σύστημα δύο δεξαμενών (μοντέλο PCT9), (Σχήμα 2.3), που χρησιμοποιεί μία μηχανοκίνητη βαλβίδα για τον έλεγχο του ρυθμού ροής του νερού που αντλείται από μία κεντρική δεξαμενή προς τις άλλες δύο, που είναι τοποθετημένες στο πάνω μέρος του συστήματος. Σε αυτές είναι τοποθετημένος ένας αισθητήρας, ο οποίος μετρά τη στάθμη του νερού. Μάλιστα, ο χρήστης μπορεί να επιλέξει διαφορετικές εξόδους με διαφορετικό ρυθμό ροής.

Αυτό το υπολογιστικό μοντέλο ARX ελέγχου των δύο δεξαμενών, εφαρμόζονται στο πραγματικό σύστημα, που απεικονίζεται στο Σχήμα 2.3, με PID ελεγκτές για την ενεργοποίηση και τον έλεγχο διαταραχών στις εξόδους του, σε πραγματικό χρόνο. [1]

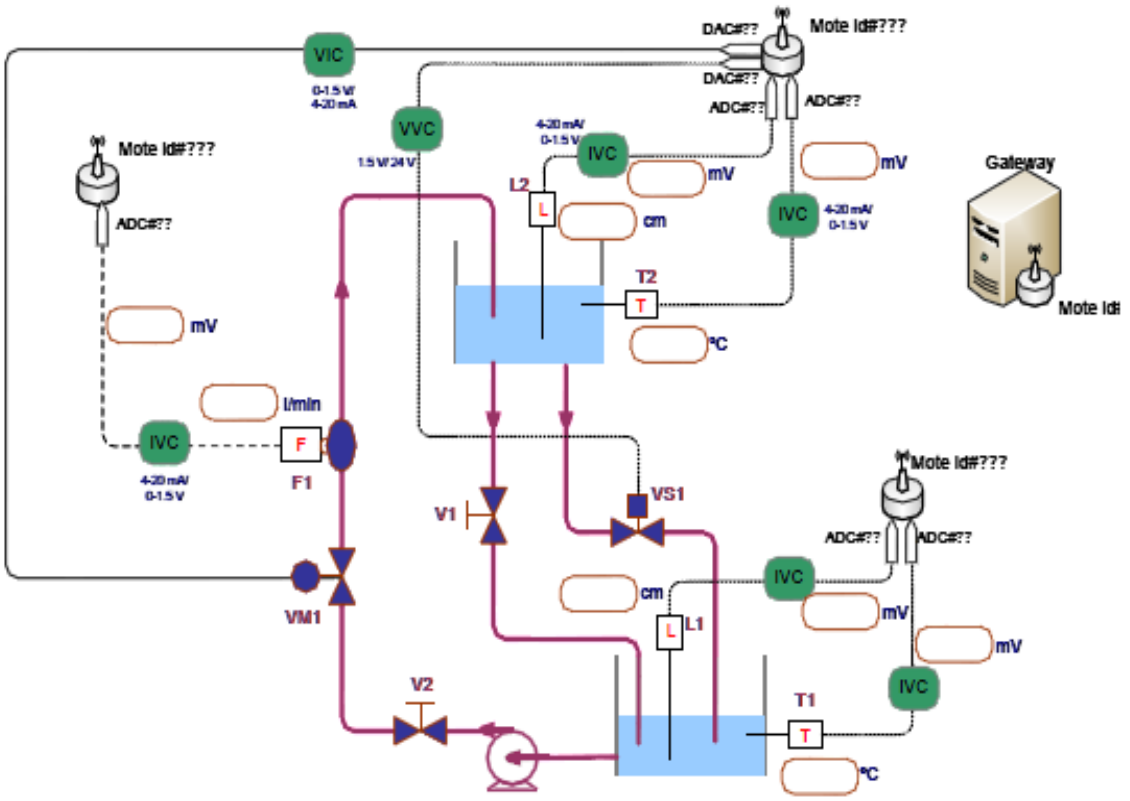
Σχήμα 2.3 Σύστημα δύο δεξαμενών (μοντέλο PCT9)

Ένα άλλο παράδειγμα του Πανεπιστημίου της Coimbra, είναι ενός συστήματος ελέγχου που έχει ως στόχο την μη-αποκλειστική επικοινωνία μεταξύ του δικτύου και του χρήστη με σκοπό τη δημιουργία ενός δικτυωμένου περιβάλλοντος, είναι το σύστημα τριών δεξαμενών (μοντέλο 200 DTS), όπως φαίνεται στο Σχήμα 2.4. Συγκεκριμένα, είναι ένα σύστημα που θα μπορούσε να χρησιμοποιηθεί για την εφαρμογή ενός δικτυακού ελέγχου (Networked Control Systems) στάθμης δεξαμενών. Πρόκειται για ένα μη γραμμικό σύστημα πολλαπλών εισόδων- πολλαπλών εξόδων (Multi Input – Multi Output, MIMO) με τρεις αισθητήρες για τον έλεγχο της στάθμης, ένα σε κάθε δεξαμενή και δύο αντλίες για τον έλεγχο της ροής του υγρού. Η χρήση του διαδικτύου για τον έλεγχο ενός πραγματικού συστήματος και μάλιστα από απόσταση, μπορεί να υπολογίσει από τις καθυστερήσεις και τον αριθμό των σφαλμάτων για να προσδιοριστεί κατά πόσο αυτό είναι αξιόπιστο. [1]

Σχήμα 2.4 Σύστημα τριών δεξαμενών (μοντέλο 200 DTS)

Άλλο ένα παράδειγμα συστήματος ελέγχου δεξαμενών του Πανεπιστημίου της Coimbra, είναι αυτό της εποπτείας και του ελέγχου κατανεμημένου συστήματος ελέγχου (Distributed Control System, DCS), το οποίο έχει ενσωματωμένο ένα ασύρματο αισθητήρα και Actuator Network (WSAN).

Το διάγραμμα που απεικονίζεται στο Σχήμα 2.5 δείχνει ένα κατανεμημένο σύστημα ελέγχου που βασίζεται στο σύστημα δύο δεξαμενών, που αναφέρεται σε προηγούμενο παράδειγμα (μοντέλο PCT9) και σε ένα WSAN που έχει τη δυνατότητα να διαβάζει θερμοκρασίες. Οι αισθητήρες και ο μηχανισμός κίνησης (αντλίες) συνδέονται μέσω των κόμβων του δικτύου σε ένα υπολογιστή. Ο χρήστης μπορεί να επιλέξει να υλοποιηθεί ο έλεγχος μέσω του δικτύου ή μέσω ενός τηλεχειριστηρίου που αλληλεπιδρά με το σύστημα [1].



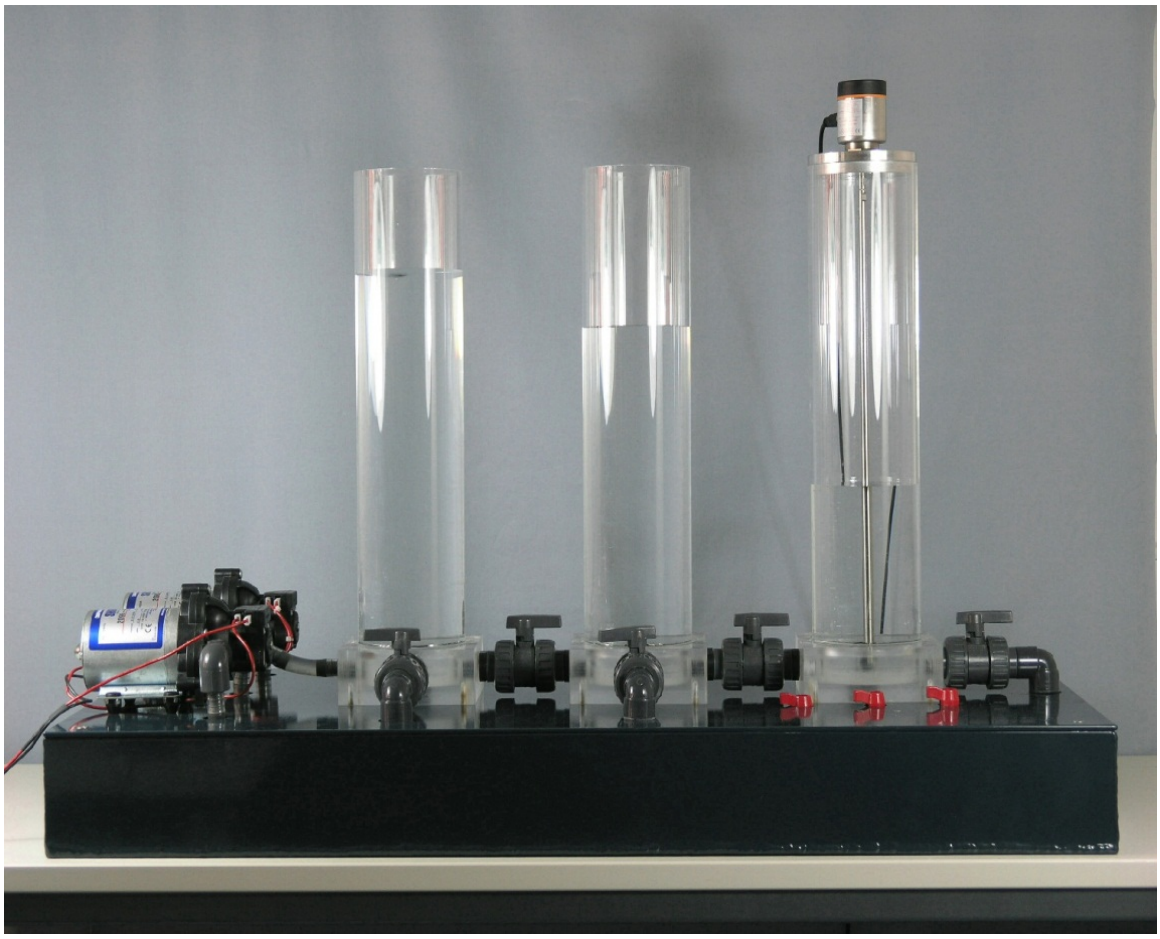
Σχήμα 2.5 Διάγραμμα του συστήματος κατανεμημένου ελέγχου με βάση το σύστημα των δύο δεξαμενών (PCT9) και ένα ασύρματο δίκτυο αισθητήρων

Τέλος, ένα ακόμη παράδειγμα συστήματος ελέγχου τριών δεξαμενών είναι αυτό της PendCon. Πρόκειται για ένα πρότυπο σύστημα ελέγχου που σκοπό έχει τη διατήρηση της στάθμης του υγρού των δεξαμενών στο επιθυμητό επίπεδο.

Η ροή εξόδου μπορεί να παράγεται από μια βαλβίδα ή μια αντλία. Με αυτό τον τρόπο, είναι δυνατόν να γίνουν διάφορες διαμορφώσεις με διαφορετικό βαθμό πολυπλοκότητα του συστήματος.

Η συσκευή για τη ροή εξόδου προσδιορίζει τον τύπο του συστήματος. Με τη βαλβίδα εξόδου, το σύστημα είναι τύπου μηδέν, ενώ αν χρησιμοποιηθεί μια αντλία για την παραγωγή ροής στην έξοδο, το σύστημα είναι τύπου ένα.

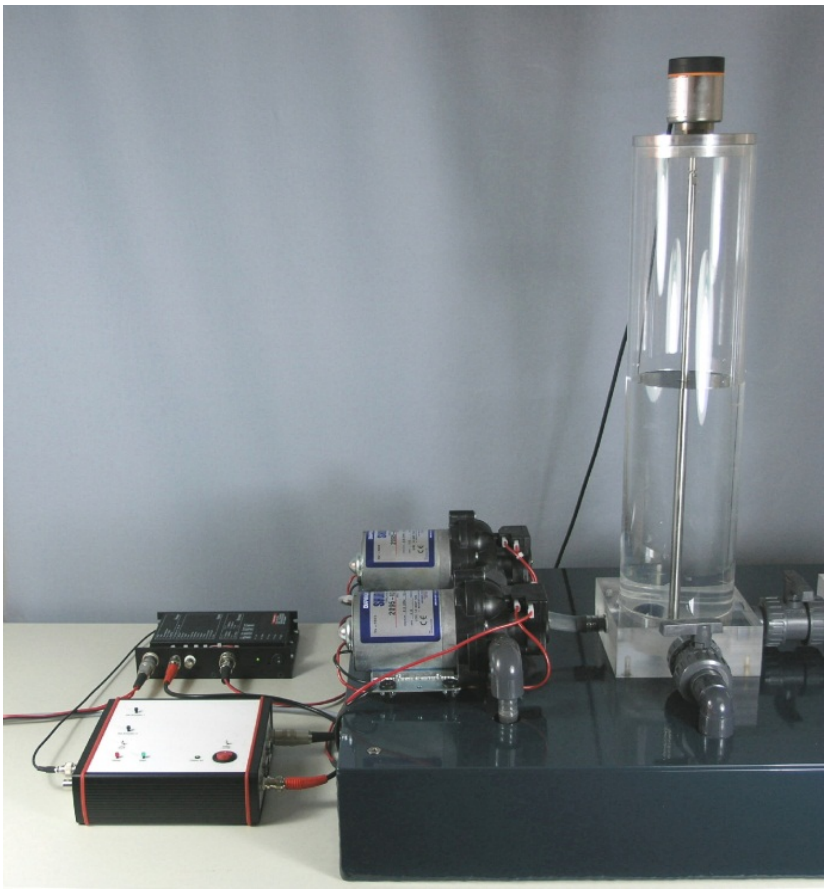
Ο σωλήνας μεταξύ των δεξαμενών μπορεί να κλείσει ή να ανοίξει για να συνδεθούν οι τρεις δεξαμενές μεταξύ τους ή μόνο οι δύο ή μόνο μία από αυτές. Με αυτό τον τρόπο, δημιουργούνται έξι διαφορετικά συστήματα. Κάθε αντλία ενεργοποιείται από ένα ενισχυτή, που αποτελεί μέρος του ελέγχου. Η στάθμη του υγρού υπολογίζεται με ένα υπερηχητικό αισθητήρα. Μπορεί να πραγματοποιηθεί και αναλογικός (PendCon Analogue) και ψηφιακός (PendCon Digital) έλεγχος [2].



Σχήμα 2.6 Έλεγχος επιπέδου στάθμης συστήματος ελέγχου τριών δεξαμενών

Ο κύριος εκπαιδευτικός στόχος του συστήματος αυτού είναι να εισαγάγει τους φοιτητές στα βασικά θέματα και τις ιδέες του σχεδιασμού ενός συστήματος ελέγχου, όπως για παράδειγμα τον υπολογισμό του βήματος και της απόκρισης συχνότητας, τη γραμμική μορφοποίηση, τον PID ελεγκτή και το σχεδιασμό του βάσει του Θεωρήματος δειγματοληψίας Nyquist.

Εάν χρησιμοποιείται μόνο μία ή δύο δεξαμενές (Σχήμα 2.7), ο σχεδιασμός του συστήματος ελέγχου μπορεί να γίνει με τη χρήση απλών τύπων. Στην απλούστερη περίπτωση, το σύστημα αποτελείται από μία δεξαμενή. Με μία αντλία εξόδου, το σύστημα είναι γραμμικό (τύπου 1), ενώ με μια βαλβίδα εξόδου, είναι μη γραμμικό (τύπου 0). Και στις δύο περιπτώσεις, τα συστήματα ελέγχου μπορούν να σχεδιαστούν με τη βοήθεια απλών εξισώσεων αλλά και πειραματικά. Σε αυτή τη περίπτωση δίνεται έμφαση όχι τόσο στο πόσο εξελιγμένο είναι σχεδιαστικά το σύστημα ελέγχου, αλλά περισσότερο στην κατανόηση των βασικών εννοιών αφού άλλωστε εφαρμόζεται για εκπαιδευτικούς σκοπούς.



Σχήμα 2.7 Σύστημα ελέγχου μιας δεξαμενής

2.3 ΣΥΣΤΗΜΑΤΑ ΕΛΕΓΧΟΥ ΓΙΑ ΕΛΕΓΧΟ ΠΡΑΓΜΑΤΙΚΩΝ ΣΥΣΚΕΥΩΝ

Μεγάλος αριθμός συστημάτων δεξαμενών έχουν κατασκευαστεί και διατεθεί στο εμπόριο. Η διάταξή τους, οι διαστάσεις τους, τα εξαρτήματα τους, ο έλεγχός τους καθώς επίσης και το λογισμικό που χρησιμοποιείται κάθε φορά, διαφοροποιούνται και εξαρτώνται από την εφαρμογή για την οποία κατασκευάστηκαν. Ενδεικτικά αναφέρονται τα παρακάτω:

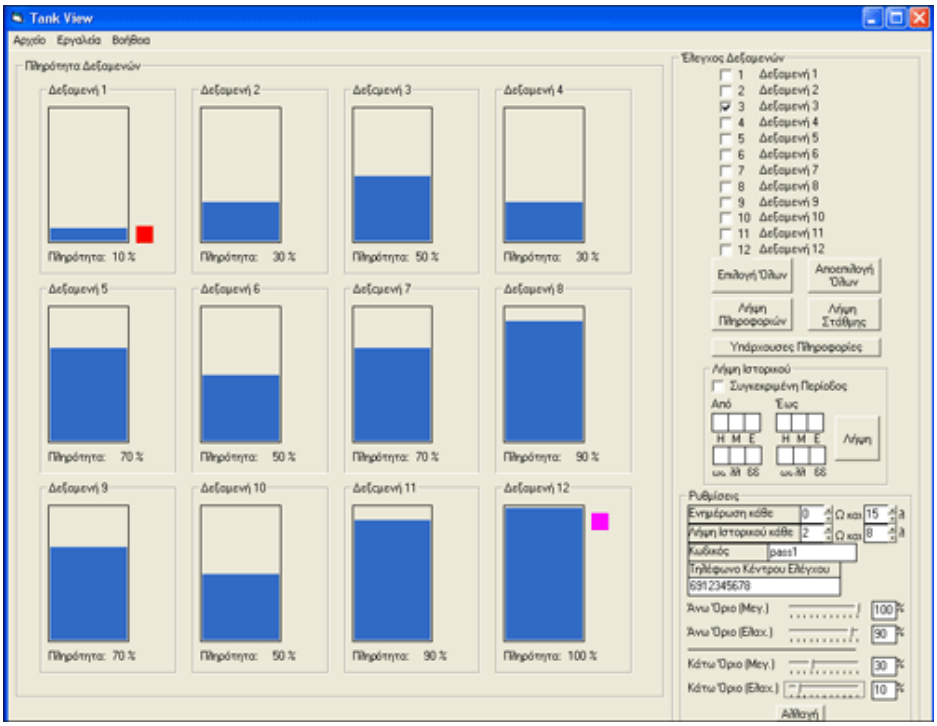
Το Σύστημα Ελέγχου και Επιτήρησης Ύδρευσης "TankViewer" αποτελείται από τις Συσκευές "TankTalk" (Σχήμα 2.8), οι οποίες αφού εγκατασταθούν στις δεξαμενές, συνδέονται με το Κέντρο Ελέγχου των Δεξαμενών έτσι ώστε να επιστρέφουν πληροφορίες σχετικά με τη στάθμη των δεξαμενών αλλά και των άλλων μεταβλητών που έχουν οριστεί σύμφωνα με τις ανάγκες του χρήστη. Ακόμα διαθέτει μία βάση δεδομένων και τη δυνατότητα επίβλεψης της στάθμης αλλά και των μεταβλητών της εκάστοτε δεξαμενής μέσω ενός ειδικού λογισμικού ελέγχου (Σχήμα 2.9) [3].

Η Συσκευή "TankTalk" εγκαθίσταται στη δεξαμενή και διαθέτει ένα GSM modem, για τη μεταφορά δεδομένων που αφορούν στη δεξαμενή καθώς επίσης και ένα σύστημα αισθητήρων στάθμης του νερού με δυνατότητα ενεργοποίησης συναγερμού (alarm) σε περίπτωση υπερχείλισης ή άδειασμα της δεξαμενής. Η ειδοποίηση σε αυτή την περίπτωση πραγματοποιείται μέσω γραπτού μηνύματος (sms) στο Κέντρο Ελέγχου ή σε κάποιο άλλο αριθμό κινητού τηλεφώνου.

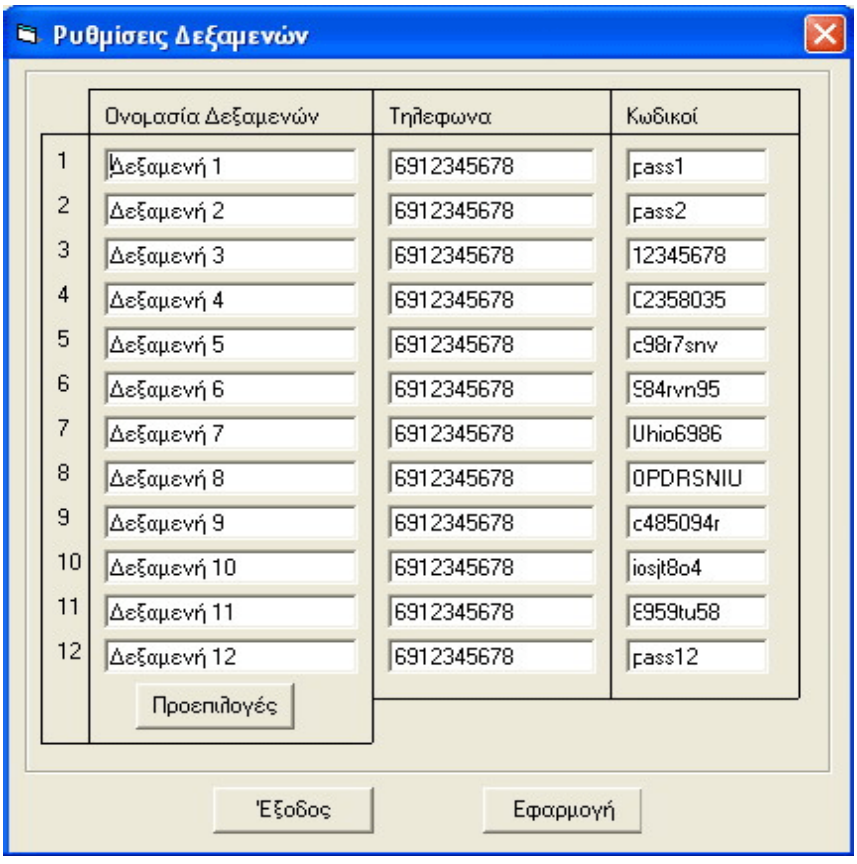
Από το Κέντρο Ελέγχου ρυθμίζονται οι συνθήκες στάθμης του νερού αλλά και άλλοι παράγοντες, όπως το μέγεθος των δεδομένων που θα λαμβάνονται, το διάστημα που μεσολαβεί μεταξύ της αποστολής των ενημερώσεων μέσω γραπτού κειμένου, καθώς επίσης ο αριθμός του τηλεφώνου του Κέντρου Ελέγχου αλλά και ο κωδικός πρόσβασης της συσκευής.

Σχήμα 2.8 Συσκευή Δεξαμενής TankTalk

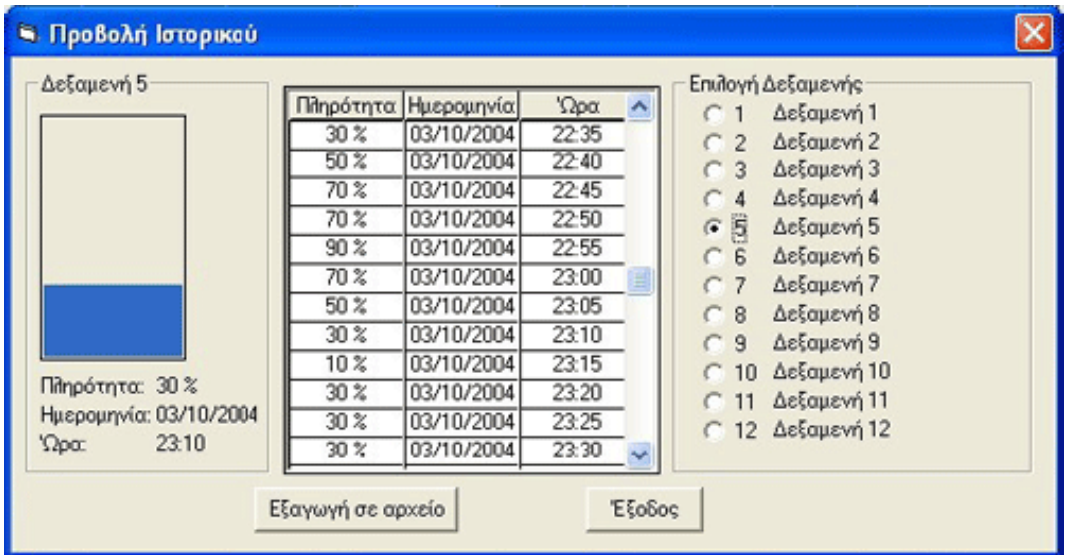
Το Κέντρο Ελέγχου Δεξαμενών Ύδρευσης "TankView" (Σχήμα 2.9) λειτουργεί με τη βοήθεια ενός Ηλεκτρονικού υπολογιστή στον οποίο έχει εγκατασταθεί το ειδικό λογισμικό ελέγχου του TankView μέσω σειριακής θύρας. Η επικοινωνία της συσκευής "TankTalk" πραγματοποιείται με τη σύνδεση του GSM modem με τον ηλεκτρονικό υπολογιστή. Έτσι, το Κέντρο Ελέγχου μπορεί να ζητάει άμεσα την στάθμη της δεξαμενής στέλνοντας ένα SMS μήνυμα. Υπάρχει επίσης η δυνατότητα εμφάνισης της στάθμης των δεξαμενών (Σχήμα 2.9), τις δεξαμενές σε κατάσταση συναγερμού (alarm) αλλά και του ιστορικού (Σχήμα 2.11) δηλαδή των στοιχείων της δεξαμενής που έχουν επιλεγεί και που αποθηκεύονται στη βάση δεδομένων που διαθέτει, συμπεριλαμβανομένων και των πληροφοριών που έχουν καταγραφεί. Τέλος, είναι δυνατή η αλλαγή των ρυθμίσεων των συσκευών που είναι συνδεδεμένες με τις δεξαμενές (Σχήμα 2.10)



Σχήμα 2.9 Λογισμικό κέντρου ελέγχου δεξαμενών TankView



Σχήμα 2.10 Ρυθμίσεις Επικοινωνίας των Δεξαμενών



Σχήμα 2.11 Προβολή Ιστορικού Δεξαμενής

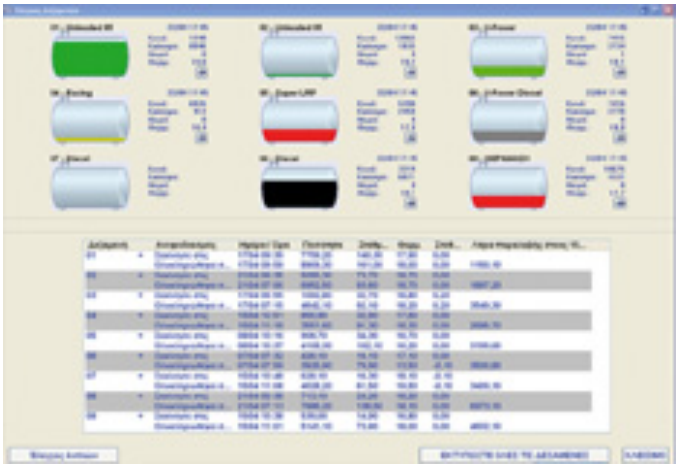
Ένα άλλο σύστημα ελέγχου δεξαμενών είναι της OPW. Τα συστήματα μέτρησης στάθμης "iTouch" και "iSite" της OPW χαρακτηρίζονται από την υψηλή ακρίβεια των μετρήσεών τους, καθιστώντας τα έτσι σημαντικά εργαλεία για τον υπολογισμό των παραμέτρων, που απαιτεί η εκάστοτε εφαρμογή (Σχήμα 2.12) [4].

Σχήμα 2.12 Σύστημα ελέγχου iTouch

Συγκεκριμένα, το σύστημα "iTouch" είναι κατάλληλο για τον έλεγχο μέχρι και 16 δεξαμενών με διάμετρο 3,80 μέτρα ενώ στην περίπτωση που απαιτείται μεγαλύτερος αριθμός δεξαμενών χρησιμοποιείται το σύστημα "iSite" που μπορεί να καλύψει έως και 128 δεξαμενές με διάμετρο έως 15 μέτρων.

Τα συστήματα αυτά είναι πολύ υψηλών προδιαγραφών και διαθέτουν μεγάλη ακρίβεια στη μέτρηση της στάθμης. Η αξιοπιστία τους ενισχύεται ακόμη πιο πολύ με τους συχνούς ελέγχους που πραγματοποιεί η OPW στα εργαστήρια διακριβώσεων που διαθέτει πιστοποιημένα κατά ISO-17025.

Το σύστημα ελέγχου "iTouch" διαθέτει αυτόματο έλεγχο διαρροών και σύστημα συναγερμού (alarm) σε περίπτωση που παρουσιαστεί μη φυσιολογική αλλαγή της στάθμης των δεξαμενών. Επιπλέον, στην περίπτωση που αλλάξει η πυκνότητα του περιεχομένου τους, για παράδειγμα αν σε δεξαμενές υγρού καυσίμου διαπιστωθεί ύπαρξη νερού αλλά και αν η θερμοκρασία του υγρού παρατηρηθεί ότι είναι εκτός ορίων. Γενικότερα αν παρουσιαστεί οποιαδήποτε απόκλιση από τα όρια που έχουν οριστεί από το χρήστη, ειδοποιείται ώστε να επέμβει άμεσα. Συγκεκριμένα μπορεί να έχει πλήρη εικόνα σχετικά με το αν υπάρχει διαρροή αλλά και για όλες τις μεταβλητές του συστήματος μέσω του ειδικού λογισμικού του συστήματος της OPW (Σχήμα 2.13).



Σχήμα 2.13 Αναλυτικά στοιχεία δεξαμενών

Ο ελεγκτής (controller) μπορεί να συνδεθεί είτε με το σύστημα κονσόλας των αντλιών είτε μέσω του ειδικού λογισμικού της OPW σε οποιονδήποτε ηλεκτρονικό υπολογιστή διαθέτει η εγκατάσταση.

ΚΕΦΑΛΑΙΟ 3

ΓΛΩΣΣΑ ΜΟΝΤΕΛΟΠΟΙΗΣΗΣ ΕΙΚΟΝΙΚΗΣ ΠΡΑΓΜΑΤΙΚΟΤΗΤΑΣ (VRML – VIRTUAL REALITY MODELING LANGUAGE)

3.1 ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο περιγράφεται διεξοδικά η Γλώσσα Μοντελοποίησης Εικονικής Πραγματικότητας VRML (Virtual Reality Modeling Language), τα βασικά στοιχεία της, όπως επίσης και η άμεση συνάφειά της με το λογισμικό Matlab. Η VRML χρησιμοποιείται για την περιγραφή εικονικών κόσμων.

3.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Η VRML αποτελεί μία γλώσσα προγραμματισμού για την περιγραφή τρισδιάστατων αντικειμένων και τη δημιουργία εικονικών κόσμων στον παγκόσμιο ιστό WWW (World Wide Web). Είναι γλώσσα γραπτού κειμένου ASCII και χρησιμοποιεί τους HTTP βοηθούς/φορείς. Οι εφαρμογές που δημιουργούνται παρουσιάζονται σε οποιοδήποτε πρόγραμμα περιήγησης (browser). Έτσι επιτυγχάνεται η ανάγνωση ενός VRML αρχείου και η εύκολη πλοήγηση σε ένα τρισδιάστατο εικονικό κόσμο [5].

Με τη χρήση της VRML, ο προγραμματιστής μπορεί να σχεδιάσει ένα εικονικό περιβάλλον και έχει την ικανότητα να δραστηριοποιείται μέσα σε αυτό.

Η VRML χρησιμοποιείται σε διάφορους τομείς, όπως στις επιχειρήσεις, τις διαφημίσεις στον παγκόσμιο ιστό (e-commerce), την ψυχαγωγία, την τέχνη, την εκπαίδευση, την επικοινωνία, την αρχιτεκτονική κλπ.

Μία γλώσσα προγραμματισμού όπως η VRML δίνει τη δυνατότητα σχεδιασμού τρισδιάστατων (3D) αντικειμένων, όπου μπορούν να τους προσδοθούν τα επιθυμητά χαρακτηριστικά χωρίς να απαιτείται η σύνδεσή τους με κάποιο συγκεκριμένο λογισμικό και η παρουσίαση μέσω φυλλομετρητή (browser) [6].

Αυτό είναι ένα μεγάλο πλεονέκτημα γιατί δίνεται η δυνατότητα αναπαράστασης πολλών εικονικών αντικειμένων είτε πρόκειται για απλά, είτε για ποιο σύνθετα με τη βοήθεια της VRML, κάνοντας εύκολη τη διαχείριση των εικόνων. Τα συγκεκριμένα αντικείμενα δεν είναι αρχεία μεγάλου μεγέθους, γεγονός που διευκολύνει τη διαχείριση τους σε πραγματικό χρόνο. Ο χρήστης έχει διαθέσιμο ένα μεγάλο όγκο πληροφοριών για τα αντικείμενα που συνθέτουν τον εικονικό κόσμο και μπορεί εύκολα να τα επεξεργαστεί και να τα τροποποιήσει.

Η VRML βρίσκει εφαρμογές σε διάφορους τομείς στο διαδίκτυο, όπως διαδικτυακή ψυχαγωγία (π.χ. παιχνίδια), διαδραστικές προσομοιώσεις στην εκπαίδευση, οπτικοποιήσεις αντικειμένων επιστημονικού ενδιαφέροντος, ψηφιοποίηση μνημείων και έργων τέχνης, προσομοιώσεις στη μηχανική, στην αρχιτεκτονική και πολλά άλλα [7].

Η πρώτη προσπάθεια δημιουργίας της VRML παρουσιάστηκε το 1994 σε ένα συνέδριο στη Γενεύη και αρχικά χρησιμοποιήθηκε ο όρος τιμή γλώσσας εικονικής πραγματικότητας (Virtual Reality Markup Language). Το 1995 παρουσιάστηκε η πρώτη έκδοση της VRML (VRML 1.0). Ένα χρόνο μετά ακολούθησε η VRML 1.1, μία βελτιωμένη εκδοχή της προηγούμενης έκδοσης , ενώ το 1996 παρουσιάστηκε η VRML 2.0 η οποία μετονομάστηκε μέσω κατάλληλων τροποποιήσεων από VRML 97 σε VRM 97.

Η VRML περιγράφει ένα σύνολο από αντικείμενα διατεταγμένα στο χώρο σε τρισδιάστατη αναπαράσταση που ορίζεται από το γεωμετρικό σχήμα, τις ιδιότητες της επιφάνειας (χρωματικές υφές - textures) και τη θέση τους στο χώρο (viewpoints). Μέσω της VRML περιγράφεται και ένα απλό αντικείμενο αλλά και ιδιαίτερα πολύπλοκες σκηνές. Η κατάληξε ενός εικονικού κόσμου, δηλαδή ενός VRML αρχείου είναι *".wrl*" (ή ακόμα "*.wrz*" ή "*.wrz.gz*" όταν αυτό είναι συμπιεσμένο). Τα αρχεία αυτά μπορούν να εγγραφούν σε οποιοδήποτε κειμενογράφο (text editor) ή με τη βοήθεια κάποιου εργαλείου τρισδιάστατης μοντελοποίησης, όπως 3D Studio Max, V-Realm Builder, FormZ, κλπ. Όταν ένας φυλλομετρητής ιστού (Web Browser) ακολουθεί ένα σύνδεσμο, ανακτά ένα αρχείο από τον παγκόσμιο ιστό WWW (World Wide Web) και διαβάζει την πληροφορία που βρίσκεται σε αυτό. Για την απεικόνιση της πληροφορίας που λαμβάνει από το πρόγραμμα πλοήγησης (Browser) χρειάζεται μία υποβοηθητική εφαρμογή, (plug-in). Εφόσον εγκατασταθεί και συσχετιστεί με το πρόγραμμα πλοήγησης (Browser) που χρησιμοποιείται δίνεται η δυνατότητα στο χρήστη να κινηθεί στον τρισδιάστατο εικονικό κόσμο.

Το βασικό δομικό στοιχείο της VRML είναι ένα δέντρο κόμβων (VRML Tree) που επιτρέπει την αλληλεπίδραση των επιμέρους τμημάτων. Η VRML διαθέτει μία ομάδα αντικειμένων που μπορούν να χρησιμοποιηθούν για τη δημιουργία τρισδιάστατων γραφικών. Τα αντικείμενα αυτά που ονομάζονται κόμβοι (nodes) και αποτελούνται από

πεδία (fields) που περιγράφουν τις ιδιότητές τους. Οι κόμβοι έχουν τα εξής χαρακτηριστικά:

- Σχήμα (Shape) το οποίο περιγράφει τρισδιάστατα αντικείμενα, όπως σφαίρα, κώνος, κύβος, κ.α.
- Ιδιότητα (Property), που καθορίζει τα χρωματικά χαρακτηριστικά, όπως Materials [8].

Ομάδας (Group), που περιέχουν μετασχηματισμούς (Transforms), δηλαδή δημιουργείται ένα σύστημα συντεταγμένων που δίνει τη δυνατότητα μετακίνησης του αντικειμένου στο χώρο και ενσωματωμένο (Inlines), δηλαδή περιγράφει που είναι τοποθετημένα τα διάφορα στοιχεία, κ.α.

3.3 ΕΝΑΛΛΑΚΤΙΚΑ ΕΡΓΑΛΕΙΑ ΜΟΝΤΕΛΟΠΟΙΗΣΗΣ ΤΡΙΣΔΙΑΣΤΑΤΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ

Εκτός από τη VRML υπάρχουν και άλλες γλώσσες που θα μπορούσαν να χρησιμοποιηθούν για την αναπαράσταση 3D αντικειμένων. Ενδεικτικά αναφέρονται τα παρακάτω:

- Η **3DMLW** (**3D M**ark-up **L**anguage for **W**eb) [9] είναι γλώσσα ανοιχτού κώδικα (open-source). Είναι κατάλληλη για την αναπαράσταση τρισδιάστατων και δισδιάστατων αντικειμένων, με δυνατότητες αλληλεπίδρασης, με τον παγκόσμιο ιστό και στηρίζεται σε αρχεία τύπου XML.
- Η **O3D** [10] είναι μια γλώσσα ανοιχτού κώδικα (open-source) και δημιουργήθηκε από τη Google για τη δημιουργία τρισδιάστατων εφαρμογών με δυνατότητες αλληλεπίδρασης, έτσι ώστε οι τρισδιάστατες εφαρμογές, να λειτουργούν σε ένα φυλλομετρητή ιστού (web browser) ή σε μια XUL εφαρμογή. Όπως η VRML έτσι και η O3D χρειάζεται μία βοηθητική εφαρμογή, ένα Plug-in έτσι ώστε να λαμβάνει από το πρόγραμμα πλοήγησης πληροφορίες για την απεικόνιση του κόσμου.
- Η **COLLADA** (**COLLA**borative **D**esign **A**ctivity) [11] καθιερώθηκε ως διεθνές πρότυπο γλώσσας προγραμματισμού με σκοπό τη δημιουργία 3D εφαρμογών στο Διαδίκτυο, ενώ στηρίζεται σε ένα XML σχήμα.
- Η **Universal 3D** (**U3D**) [12] Η "3D Industry Forum" όρισε το συγκεκριμένο πρότυπο συμπιεσμένου αρχείου στοχεύοντας στη διευκόλυνση της ανταλλαγής δεδομένων 3D γραφικών. Είναι ένα συμπιεσμένο πρότυπο αρχείου, που χρησιμοποιείται για δεδομένα 3D γραφικών. Η "Ecma International" το κατοχύρωσε τον Αύγουστο του 2005 ως ECMA-363, προκειμένου να αποτελέσει πρότυπο για κάθε είδους 3D δεδομένα. Με την U3D παρέχεται η δυνατότητα να τοποθετηθούν τα 3D αντικείμενα στο περιβάλλον της σε μορφή PDF και να παρατηρηθούν μέσω Acrobat Reader.

- Η **X3D** [13] είναι μια γλώσσα προγραμματισμού, κατοχυρωμένη με ISO (ISO/IEC 19775-1), και θα μπορούσε να θεωρηθεί ως η εξέλιξη της VRML, ενώ εξυπηρετεί στην αναπαράσταση εικονικών κόσμων με τη χρήση αρχείων τύπου XML, όπως τον Ανοιχτό Εφευρέτη της VRML97. Humanoid Animation, NURBS και GeoVRML, είναι κάποιες από τις επιπλέον δυνατότητες, που προσφέρει συμβάλλοντας στην κωδικοποίηση της εικόνας, αλλά και ενισχύοντας το περιβάλλον προγραμματισμού των εφαρμογών (Application Programming Interface - APIs).

Οι λόγοι που η VRML υπερέχει του προτύπου X3D είναι οι εξής:

- Για το σχεδιασμό τρισδιάστατων αντικειμένων με χρήση του X3D απαιτείται γνώση της VRML..
- Μια επιπλέον γλώσσα, η XML, είναι απαραίτητη προκειμένου να χρησιμοποιηθεί το πρότυπο X3D, δυσκολεύοντας τη δουλειά του χρήστη.
- Η πρόσβαση από εργαλεία μοντελοποίησης σε γλώσσα προγραμματισμού με τη VRML μετατρέπεται σε απλή διαδικασία, καθώς ο έλεγχος γίνεται σε πραγματικό χρόνο [14].

3.4 ΣΧΕΔΙΑΣΜΟΣ ΕΙΚΟΝΙΚΩΝ ΚΟΣΜΩΝ ΜΕ ΧΡΗΣΗ ΤΟΥ V-REALM BUILDER

Το πρόγραμμα μοντελοποίησης εικονικών κόσμων V-Realm Builder (Version 2.0) είναι κατάλληλο για το σχεδιασμό VRML αντικειμένων. Πρόκειται για ένα προγραμματιστικό εργαλείο για τη δημιουργία τρισδιάστατων (3D) αντικειμένων και εικονικών κόσμων με την υποστήριξη οποιουδήποτε φυλλομετρητή (browser) συμβατού με το VRML 2.0.

Το V-Realm Builder σε συνδυασμό με τη VRML έχει τα πλεονεκτήματα του δικτυακού περιβάλλοντος, όπως το μικρό μέγεθος των αρχείων, ακόμη και όταν πρόκειται για περιβάλλον με σύνθετα αντικείμενα.

Το V-Realm Builder είναι ένα εύχρηστο εργαλείο, αφού δεν είναι απαραίτητη η γνώση γλωσσών προγραμματισμού και η συγγραφή κώδικα. Προσφέρει τη δυνατότητα σχεδιασμού και ελέγχου ενός εικονικού κόσμου μόνο με τη χρήση ενός ποντικιού και ελάχιστη χρήση πληκτρολογίου.

Χάρη στο ειδικά προσαρμοσμένο στη VRML γραφικό του περιβάλλον (Graphical User Interface - GUI), απλοποιείται ο σχεδιασμός εικονικών κόσμων, καθώς η δυνατότητα επέμβασης και διόρθωσης είναι άμεση [15].

3.5 ΕΓΚΑΤΑΣΤΑΣΗ ΚΑΙ ΑΡΧΙΚΟΠΟΙΗΣΗ ΤΟΥ V-REALM BUILDER

Για τον σχεδιασμό εικονικών κόσμων που αποτελούνται από τρισδιάστατα αντικείμενα χρησιμοποιείται το V-Realm Builder, ενώ μέσω του λογισμικού Matlab, τα αντικείμενα αυτά μπορούν αν ελεγχθούν. Ο V-Realm Builder εμπεριέχεται στην αρχική εγκατάσταση του Matlab. Προκειμένου να ενεργοποιηθεί και να είναι δυνατή η χρήση του

πληκτρολογείται στο Command Window του λογισμικού Matlab η εντολή: **vrinstall ('-install','editor')** και στη συνέχεια εμφανίζεται το μήνυμα:

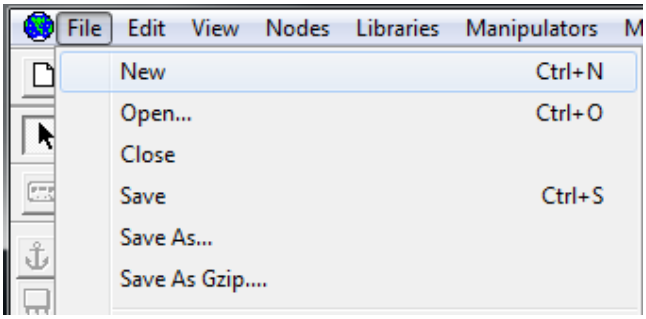
```
Starting editor installation ...
Done.
```

Για τον έλεγχο της εγκατάστασης του V-Realm Builder πληκτρολογείται η εντολή: **vrinstall('-check')** και στη συνέχεια εμφανίζεται το μήνυμα:

```
External VRML viewer:    installed
    VRML editor: installed
```

Για την εκκίνηση του V-Realm Builder πρέπει να εκτελεστεί το αρχείο "vrbuild2.exe", που βρίσκεται εγκατεστημένο στην εργαλειοθήκη (toolbox) του λογισμικού Matlab [16].

Από το μενού του V-Realm Builder επιλέγεται το File και στη συνέχεια το New για τη δημιουργία ενός εικονικού κόσμου. Εναλλακτικά πληκτρολογείται η συντόμευση Ctrl+N (Σχήμα 3.1) ή επιλέγεται το εικονίδιο *New*  .



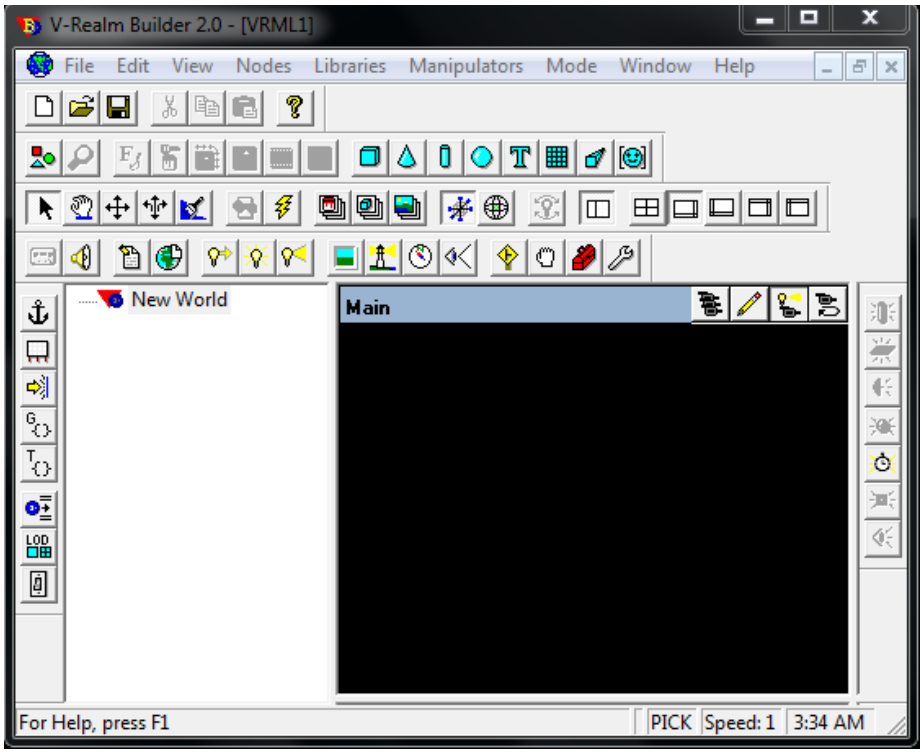
Σχήμα 3.1 Μενού έναρξης του V-Realm Builder

Στην οθόνη εμφανίζεται ένα περιβάλλον (interface) που απαρτίζει το V-Realm Builder και που αποτελείται από ένα λευκό παράθυρο στα αριστερά και ονομάζεται δέντρο κόμβων (Node Tree View).

Στο σημείο αυτό εμφανίζονται ιεραρχικά όλα τα στοιχεία που έχουν επιλεγεί για την αναπαράσταση του εικονικού κόσμου. Δεξιά εμφανίζεται το βασικό παράθυρο του V-Realm Builder το Main View που αναπαριστά τον κόσμο όπως είναι υπό κατασκευή. Το V-Realm Builder διαθέτει πολλά ακόμη εργαλεία δίνοντας τη δυνατότητα επεξεργασίας των χαρακτηριστικών των αντικειμένων, όπως το σχήμα τους αλλά και των λειτουργιών τους.

Οι κόμβοι αποτελούν τα βασικά συστατικά ενός VRML αρχείου και περιγράφουν τα γεωμετρικά χαρακτηριστικά και τις ιδιότητες κάθε αντικειμένου που σχηματίζει τον εικονικό κόσμο. Συγκεκριμένα περιγράφουν το σχήμα, το χρώμα, την επιφανειακή υφή (texture) αλλά και το πως προσανατολίζονται στον τρισδιάστατο κόσμο. Τα

ομαδοποιημένα στοιχεία κάθε κόμβου σχηματίζουν μία ομάδα (group**).** Το σημείο που ενώνονται σύνθετα σχήματα με τη χρήση κόμβων λέγεται parent node και το βασικό αντικείμενο γονέας (parent). Τα σχήματα που προστίθενται στην ομάδα λέγονται παιδιά (children) και μοιράζονται τα ίδια χαρακτηριστικά του γονέα (parent node).



Σχήμα 3.2 Γραφικό περιβάλλον του V-Realm Builder

3.6 ΕΙΣΑΓΩΓΗ ΚΑΙ ΕΠΕΞΕΡΓΑΣΙΑ ΑΝΤΙΚΕΙΜΕΝΩΝ ΣΤΟ V-REALM BUILDER

Η VRML έχει πρωτογενή γεωμετρικά σχήματα όπως ο κύβος, ο κώνος, ο κύλινδρος και η σφαίρα, κ.α. Τα σχήματα αυτά επιλέγονται από τη Geometry Node Toolbar και εισάγονται στο Main View. Έτσι, αναπτύσσεται στο Node Tree View το δέντρο με όλα τα χαρακτηριστικά του αντικειμένου που έχει επιλεγεί, όπως το σχήμα του (Shape) και η θέση του (Transform) στον υπό επεξεργασία κόσμο.

Σχήμα 3.3 Εργαλειομπάρα γεωμετρικών σχημάτων

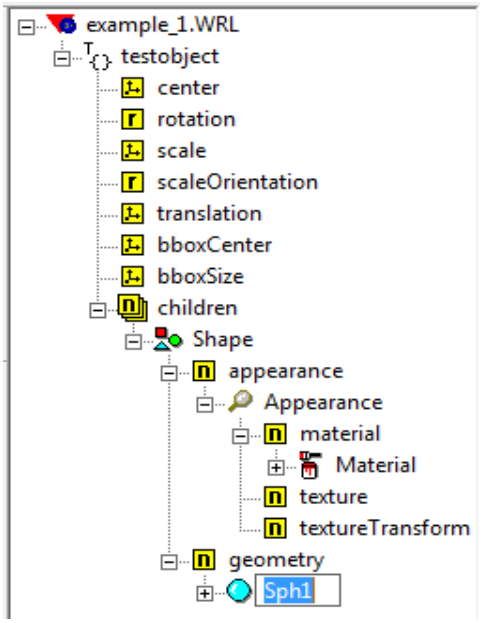
Για την εισαγωγή, για παράδειγμα μίας σφαίρας επιλέγεται από το Geometry Toolbar το κουμπί Insert Sphere. Σε αυτή την περίπτωση εμφανίζεται στο Node Tree View του

VRML, η ομάδα (group) του αντικειμένου με όλες τις πληροφορίες που διαθέτει. Εφόσον απαιτείται να συνδεθεί με ένα άλλο αντικείμενο, θα πρέπει να επιλεγεί ο κόμβος children και στη συνέχεια επιλέγεται το δεύτερο αντικείμενο, πάλι από το Geometry Toolbar. Από τις ιδιότητες που εμφανίζονται γίνονται οι ανάλογες ρυθμίσεις, έτσι ώστε να αλληλεπιδρούν μεταξύ τους, όπως επιθυμεί κάθε φορά ο χρήστης.

Από το δέντρο που αναπτύσσεται καθίσταται δυνατή η επεξεργασία των διαστάσεων των επιλεγμένων αντικειμένων από τα ακόλουθα σημεία:

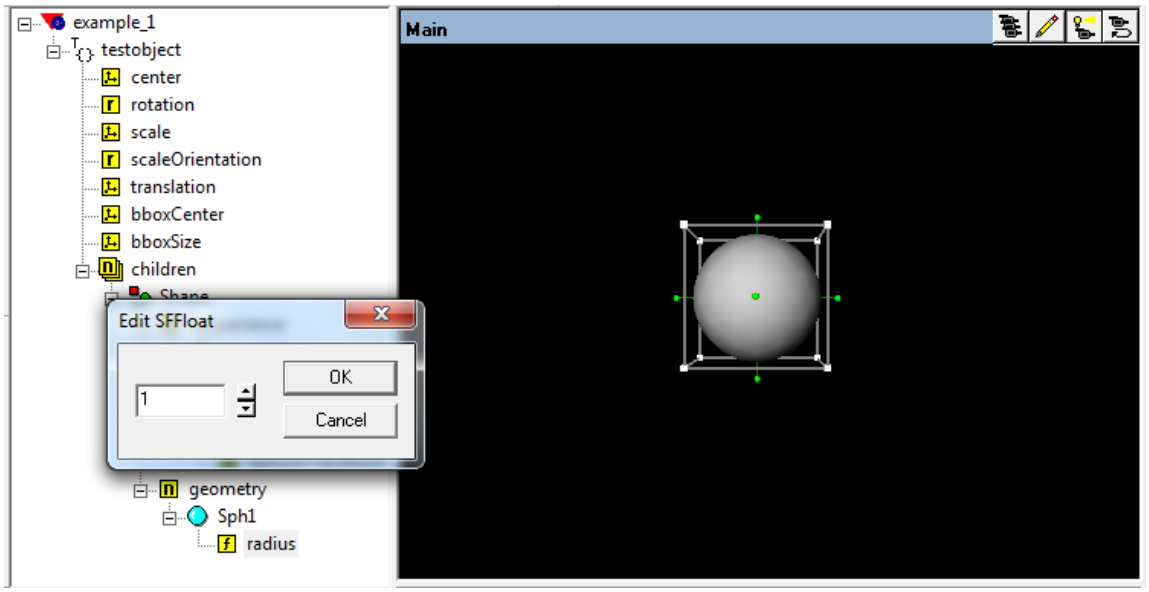
- Ο κόμβος children, με τα χαρακτηριστικά του σχήματος του αντικειμένου.
- Ο κόμβος Shape, που χωρίζεται σε δύο τομείς την όψη (appearance) και τη γεωμετρία (geometry).
- Ο κόμβος Appearance, ορίζει τα οπτικά χαρακτηριστικά του αντικειμένου, όπως το σχήμα και το υλικό.
- Ο κόμβος Geometry, που περιέχει το γεωμετρικό σχήμα.
- Ο κόμβος Sphere, που είναι το επιλεγμένο σχήμα.
- Ο κόμβος Radius, που ορίζει την ακτίνα της σφαίρας.

Για να καταστεί δυνατός ο έλεγχος των αντικειμένων ενός εικονικού κόσμου, είναι απαραίτητο να τους δοθεί κάποιο όνομα αναφοράς. Για τη μετονομασία του προς επεξεργασία αντικειμένου επιλέγεται το Transform και στη συνέχεια επιλέγεται ξανά ώστε να μπορεί να μετονομαστεί με κάποιο άλλο όνομα της επιλογής του χρήστη όπως για παράδειγμα σε “testobject”. Το όνομα του εικονικού κόσμου (New World) αλλάζει κατά την αποθήκευσή του αρχείου (Save as).



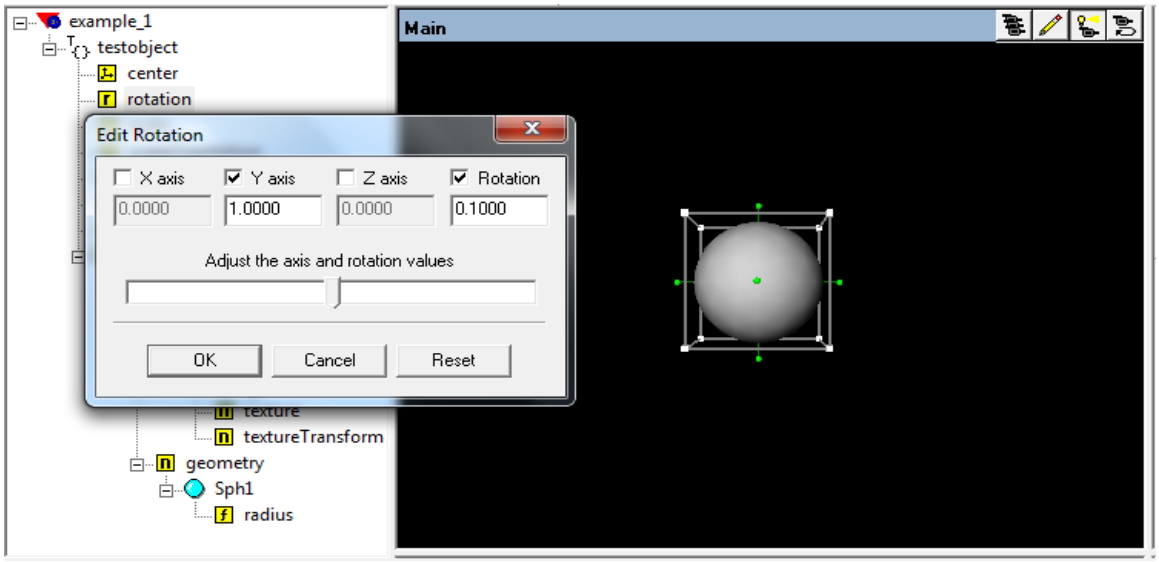
Σχήμα 3.4 Μετονομασία στοιχείων στο V-Realm Builder

Από την επιλογή του κόμβου Radius εμφανίζεται το παράθυρο διαλόγου "Edit SFFloat", το οποίο δίνει τη δυνατότητα να μεταβληθούν οι διαστάσεις της σφαίρας, όπως για παράδειγμα να αυξομειωθεί η ακτίνα της.



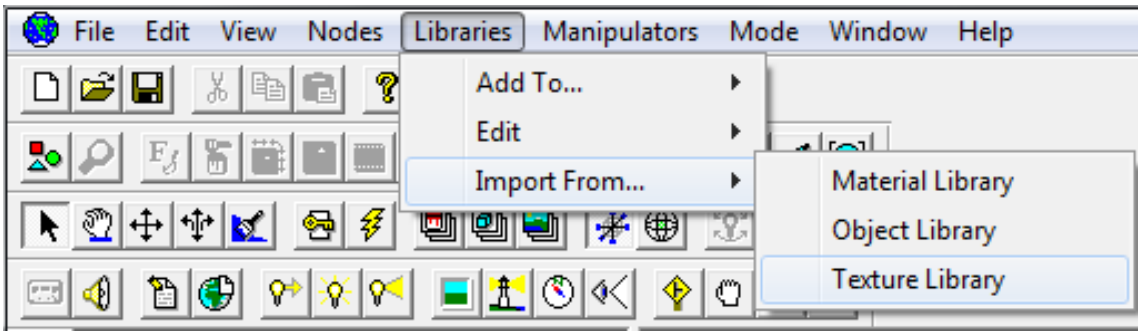
Σχήμα 3.5 Επεξεργασία ακτίνας αντικειμένου στο V-Realm Builder

Ο άξονας περιστροφής της σφαίρας μεταβάλλεται από τον κόμβο Transform ("testobject"), και τότε από το rotation (Transform→rotation) εμφανίζεται το "Editor Rotation". Από αυτό το παράθυρο διαλόγου ορίζεται η γωνία της σφαίρας και οι άξονας περιστροφής της (X, Y και Z).



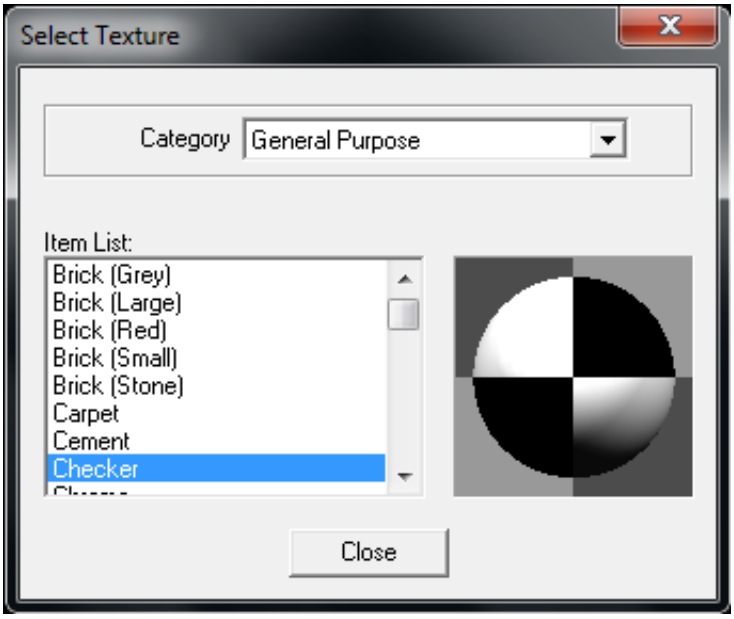
Σχήμα 3.6 Επεξεργασία γωνίας και άξονα περιστροφής αντικειμένου στο V-Realm Builder

Από τη Mode View ToolBar, επιλέγεται η Βιβλιοθήκη Υλικών (Material Library), για να αντικατασταθεί το προεπιλεγμένο γκρι χρώμα της σφαίρας, με κάποιο από αυτά που παρουσιάζονται στο παράθυρο διαλόγου που ανοίγει (Select Material). Αντίστοιχα, από τη Βιβλιοθήκη Υφής (Texture Library), δίνεται μία σειρά επιλογών χρωματικών σχεδίων υφής (Texture).



Σχήμα 3.7 Ορισμός σχεδίου υφής το V-Realm Builder

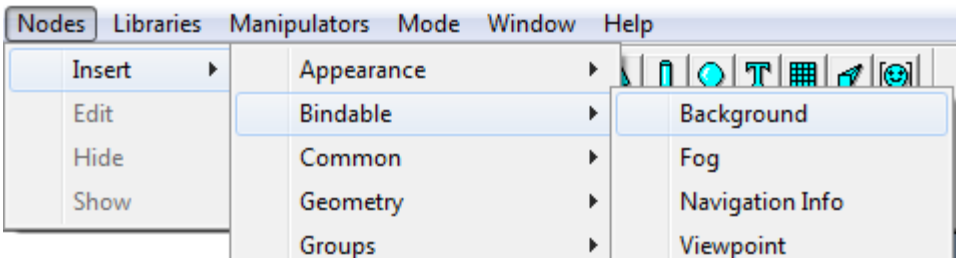
Η "Item List:" παρουσιάζει μία σειρά διαθέσιμων σχεδίων. Αφού επιλεγεί το κατάλληλο σχέδιο για το αντικείμενο, με τη βοήθεια της σφαίρας – δειγματολόγιο, τοποθετείται στο αντικείμενο με τη μέθοδο μεταφέρω και αποθέτω (drag and drop).



Σχήμα 3.8 Εισαγωγή σχεδίου υφής το V-Realm Builder

Εναλλακτικά, η επιλογή του Texture μπορεί να γίνει από τον κόμβο Appearance→ texture→ Insert Image Texture. Με τη μέθοδο drag and drop επιλέγεται το κατάλληλο σχέδιο υφής από την "Item List:" και εφαρμόζεται στη σφαίρα.

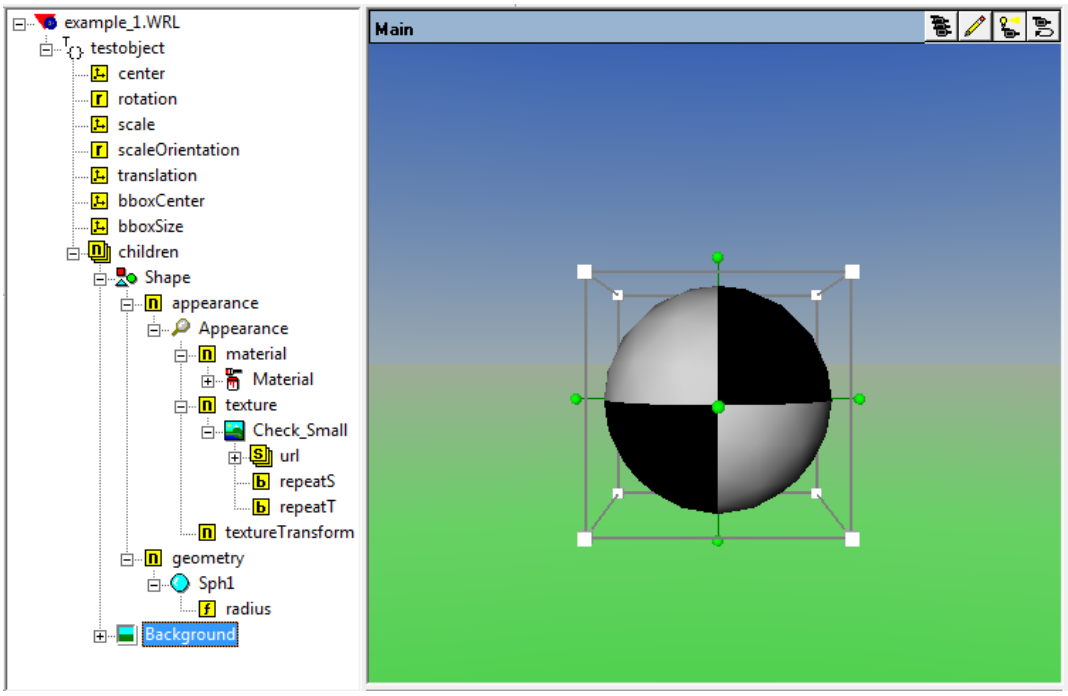
Επιπλέον, μπορεί να αλλάξει το φόντο του Main View με την επιλογή του Insert→Bindable→Background.



Σχήμα 3.9 Επιλογή φόντου στο Main View του V-Realm Builder

Εναλλακτικά, επιλέγεται από τη γενική εργαλειομπάρα (Common Node Toolbar) το εικονίδιο  Εισαγωγή Φόντου (Insert Background).

Έτσι, το προεπιλεγμένο μαύρο φόντο του Main Menu μπορεί να αντικατασταθεί από ένα άλλο φόντο, με αποχρώσεις ουρανού και Γης (μπλε και πράσινο).



Σχήμα 3.10 Εισαγωγή φόντου στο Main View του V-Realm Builder

Κατά αυτόν τον τρόπο δημιουργείται ένα αρχείο με την προεπιλεγμένη ονομασία "vrml1.wrl" ή μπορεί να δοθεί με κάποιο άλλο όνομα της επιλογής του χρήστη, όπως "example_1.wrl". Η κατάληξη ".wrl" προέρχεται από τη λέξη world.

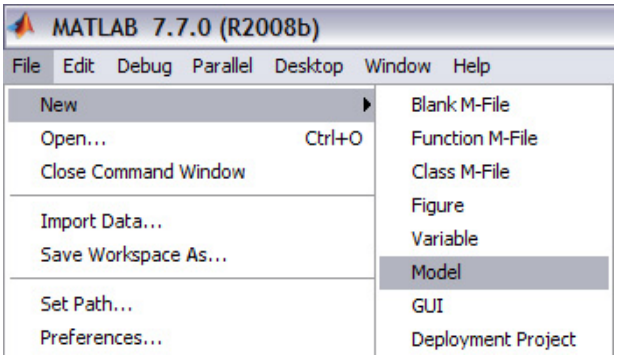
3.7 ΕΛΕΓΧΟΣ ΕΙΚΟΝΙΚΩΝ ΚΟΣΜΩΝ ΜΕΣΩ SIMULINK

Με τη βοήθεια του V-Realm Builder είναι δυνατή η σχεδίαση ενός εικονικού κόσμου. Ο έλεγχός του, μπορεί να πραγματοποιηθεί μέσω του λογισμικού Simulink. Πρόκειται για ένα παρακλάδι του λογισμικού Matlab με τη διαφορά ότι διαθέτει ένα διαδραστικό γραφικό περιβάλλον (Graphical User Interface - GUI) που δίνει τη δυνατότητα στο χρήστη να δημιουργήσει εξολοκλήρου μοντέλα. Ο V-Realm Builder διαθέτει γραφικό περιβάλλον μοντελοποίησης και προσομοίωσης γραμμικών και μη γραμμικών συστημάτων.

Το Simulink, μέσω της βιβλιοθήκης εικονικής πραγματικότητας (Virtual Reality Toolbox library) που διαθέτει, καθιστά δυνατή την αλληλεπίδραση του GUI με τους εικονικούς κόσμους. Η σύνδεση του λογισμικού προσομοίωσης με τους εικονικούς κόσμους πραγματοποιείται με τη βοήθεια των διαγραμμάτων (block) της Virtual Reality Toolbox library.

Από το κεντρικό μενού του λογισμικού Matlab επιλέγεται το File →New→Model έτσι ώστε να δημιουργηθεί ένα μοντέλο προσομοίωσης (Simulink Model).

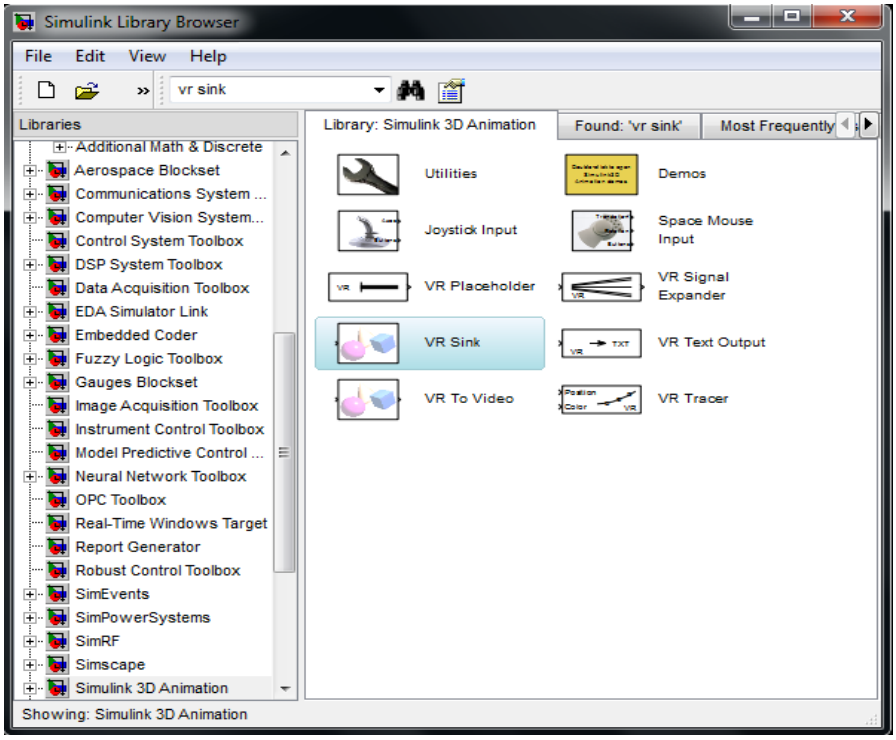
Το Simulink Model που δημιουργείται αποθηκεύεται από το μενού File και στη συνέχεια επιλέγεται το Save as. Από το παράθυρο διαλόγου που εμφανίζεται πληκτρολογείται το όνομα "demo_1.mdl". Η κατάληξη .mdl προέρχεται από την αγγλική λέξη model.



Σχήμα 3.11 Δημιουργία Μοντέλου προσομοίωσης στο λογισμικό Matlab

Πληκτρολογώντας τη λέξη Simulink στο Command Window του Matlab, ανοίγει το παράθυρο διαλόγου Simulink Library Browser, το οποίο αποτελεί τη βιβλιοθήκη του λογισμικού Simulink με όλα τα διαθέσιμα εργαλεία για το σχεδιασμό του μοντέλου προσομοίωσης.

Από το σημείο αναζήτησης πληκτρολογείται η λέξη κλειδί “VR Sink” και στη συνέχεια επιλέγεται από το παράθυρο Library: Simulink 3D Animation έτσι ώστε να καταστεί δυνατή η μεταφορά του διαγράμματος VR Sink από το μοντέλο προσομοίωσης στον εικονικό κόσμο.



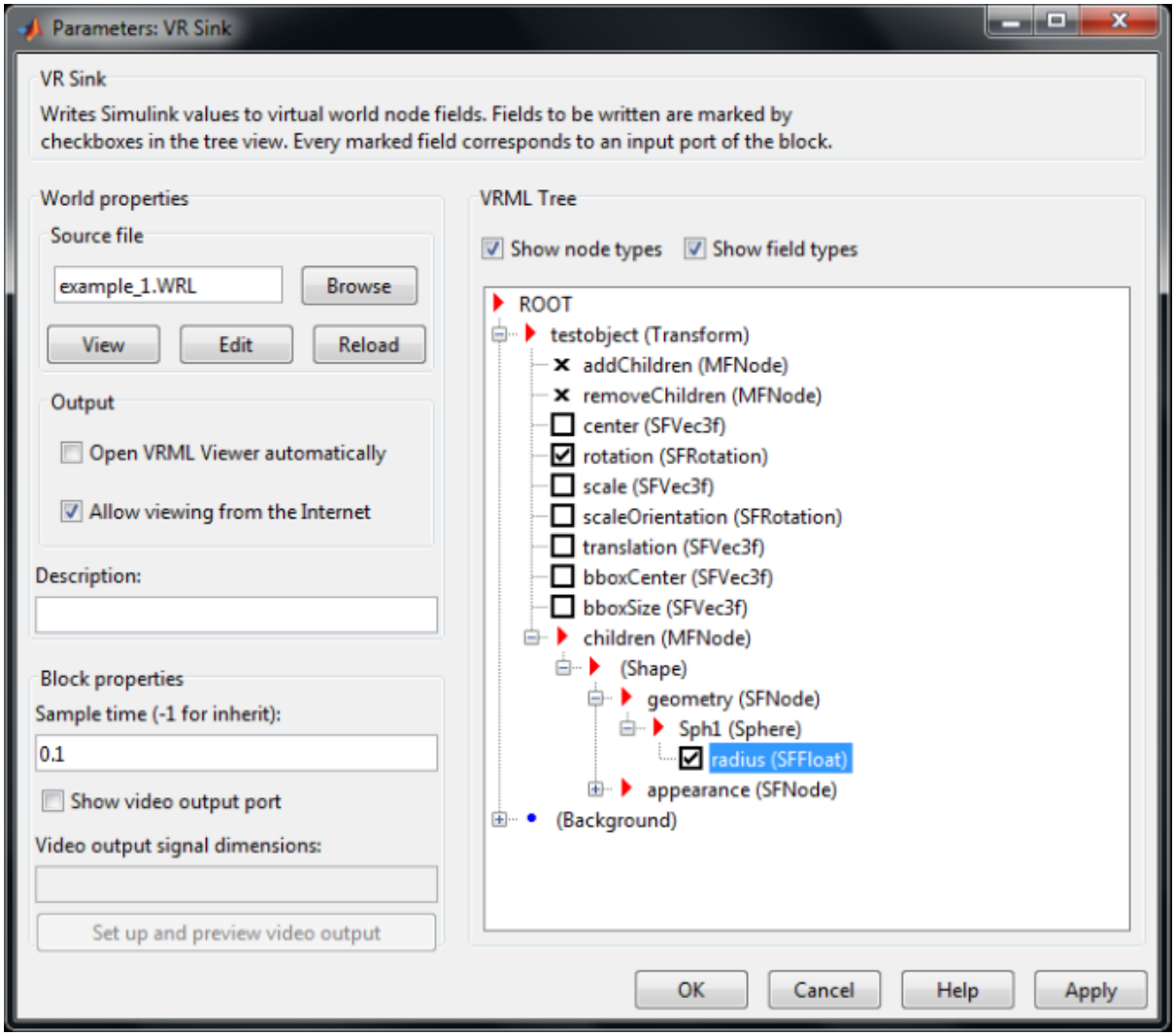
Σχήμα 3.12 Άνοιγμα του διαλόγου του λογισμικού Simulink από το λογισμικό Matlab

Αφού επιλεγούν οι παράμετροι ελέγχου από το περιβάλλον προσομοίωσης (Simulink) ενημερώνεται το διάγραμμα (VR Sink) και την ίδια στιγμή τα σήματα εισόδου και εξόδου συνδέονται κατάλληλα με τους αντίστοιχους κόμβους του εικονικού κόσμου. Με αυτό τον τρόπο είναι δυνατός ο έλεγχος του Simulink μέσω ενός λογισμικού αναπαράστασης VRML (VRML viewer).

Οι ιδιότητες των VRML κόμβων ανοίγουν ιεραρχικά. Με την εισαγωγή του VR Sink ο εικονικός κόσμος αναζητά στο λογισμικό VRML τους διαθέσιμους κόμβους που μπορούν να οριστούν ως είσοδοι διαγράμματος και να παράγουν τα σήματα του λογισμικού Simulink.

Ο έλεγχος της σφαίρας που δημιουργήθηκε στο VRML, πραγματοποιείται με το VR Sink. Αν ανοιχτεί το VR Sink, εμφανίζεται το παράθυρο των ιδιοτήτων “Parameters: VR Sink”. Από το πλαίσιο (panel) με το όνομα “World properties” που εμφανίζεται στα αριστερά του παραθύρου, γίνεται αναζήτηση του προς επεξεργασία αρχείου με τη βοήθεια του κουμπιού “Browse”. Από το πλαίσιο “VRML Tree” ξεδιπλώνεται το VRML

δέντρο του εικονικού κόσμου, το οποίο εμπεριέχει όλες τις παραμέτρους ελέγχου. Σε αυτό το σημείο επιλέγεται το "rotation (SFRotation)".



Σχήμα 3.13 Παράθυρο ιδιοτήτων του VR Sink διαγράμματος

Η κίνηση της σφαίρας μπορεί να πραγματοποιηθεί με τη βοήθεια ορισμένων δομικών τμημάτων, τα οποία συνδέονται με τις εισόδους του διαγράμματος VR Sink και δίνουν στη σφαίρα τις τιμές που ορίζονται κάθε φορά.

Για την περιστροφική κίνηση και την αυξομείωση της ακτίνας της σφαίρας, απαιτείται η χρήση δύο "Slider Gain" διαγραμμάτων, που μπορούν να πολλαπλασιάσουν το σήμα εισόδου καθώς κινείται η μπάρα ολίσθησης (Slider).

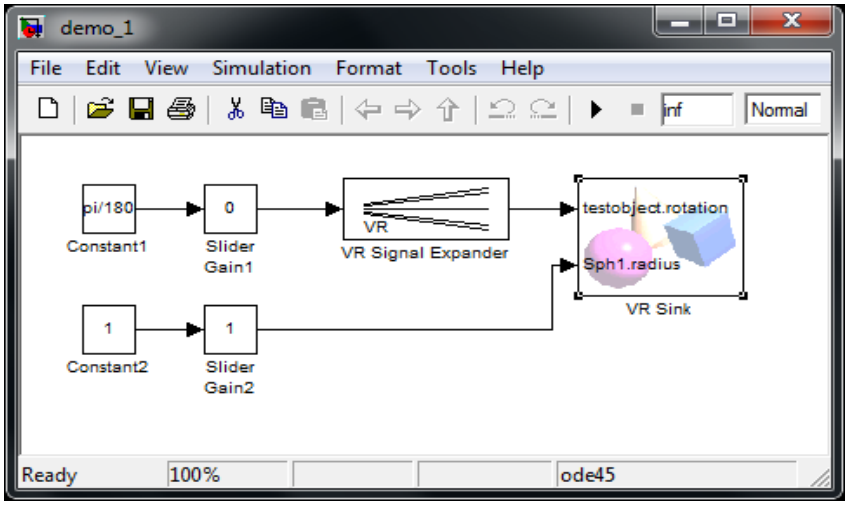
Για την επίτευξη μίας πλήρους περιστροφής (360°) της σφαίρας, δίνονται για το πρώτο Slider Gain (Slider Gain1) οι τιμές, που ρυθμίζονται από -180 έως 180.

Σχήμα 3.14 Μπάρα ολίσθησης για τον έλεγχο της περιστροφικής κίνησης της σφαίρας στο λογισμικό MATLAB

Σχήμα 3.15 Μπάρα ολίσθησης για τον έλεγχο της ακτίνας της σφαίρας στο λογισμικό MATLAB

Οι τιμές των Slider Gain είναι σε ακτίνια. Για την μετατροπή των ακτινίων σε μοίρες, τοποθετείται στην είσοδο του "Slider Gain1" διαγράμματος ένα "Constant" διάγραμμα ίσο με π/180 (pi/180), το οποίο έχει την ιδιότητα, να δίνει ένα σήμα εξόδου. Αυτό στη συνέχεια πολλαπλασιάζεται με το αποτέλεσμα του "Slider Gain1" διαγράμματος και οδηγείται στην έξοδό του. Πριν οδηγηθεί το σήμα στο VR Sink, τοποθετείται ανάμεσα στην έξοδο του "Slider Gain1" διαγράμματος και την είσοδο του VR Sink διαγράμματος, ένα "VR Signal Expander" για την επέκταση του σήματος.

Για την εξομοίωση της ακτίνας της σφαίρας, δίνονται για το δεύτερο Slider Gain (Slider Gain2) οι τυχαίες τιμές, που ρυθμίζονται από 1 έως 5.



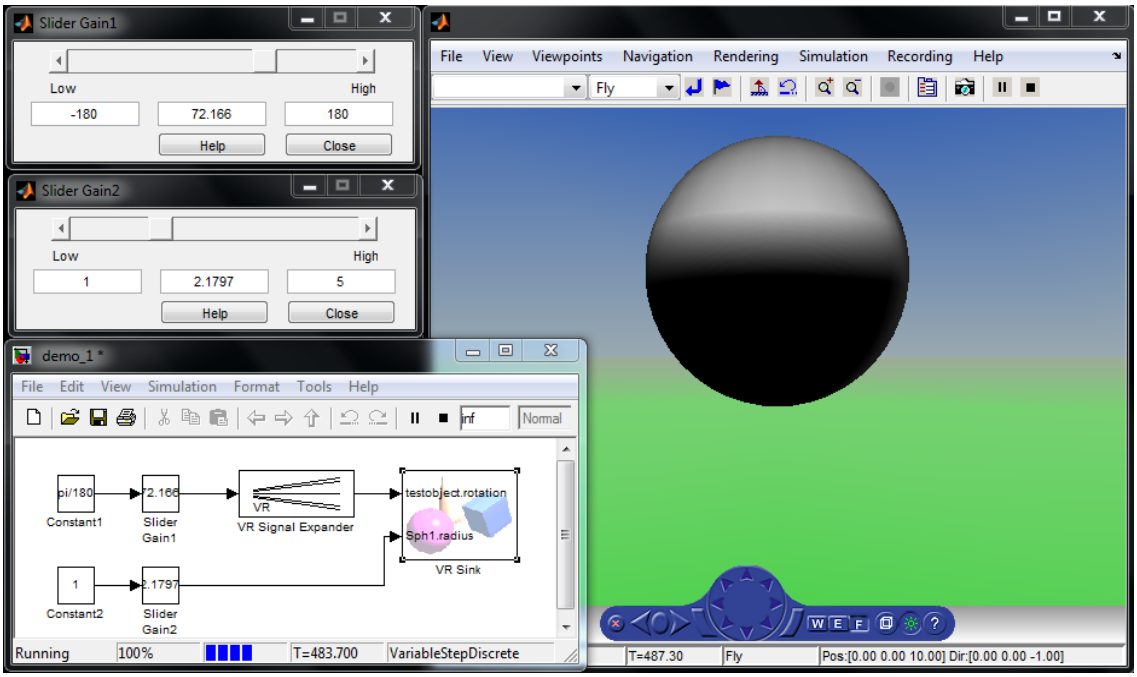
Σχήμα 3.16 Μοντέλο προσομοίωσης στο λογισμικό Matlab

Στην είσοδο “Slider Gain2” διαγράμματος, τοποθετείται ένα “Constant” διάγραμμα ίσο με 1. Αυτό στην συνέχεια πολλαπλασιάζεται με το αποτέλεσμα του “Slider Gain2” διαγράμματος και οδηγείται στην είσοδο του VR Sink διαγράμματος.

Για την έναρξη της προσομοίωσης πρέπει στο μενού του Simulink Model (μοντέλου προσομοίωσης) του λογισμικού Matlab, να επιλεγεί το Simulation και μετά το Start ή Ctrl+T ή να επιλεγεί το εικονίδιο Start simulation.

Για να εμφανιστεί ο εικονικός κόσμος, πρέπει να επιλεχθεί το VR Sink. Για τον έλεγχο του εικονικού κόσμου υπάρχουν τα Slider Gain, που ανοίγοντάς τα εμφανίζονται οι μπάρες ολίσθησης. Η περιστροφή της σφαίρας στον εικονικό κόσμο, είναι ανάλογη των τιμών της πρώτης μπάρας ολίσθησης ενώ η τιμή της ακτίνας της σφαίρας είναι ανάλογη των τιμών της δεύτερης μπάρας ολίσθησης.

Με την μπάρα ολίσθησης του παραθύρου “Slider Gain 1”, μπορεί να ελεγχθεί η περιστροφή της σφαίρας. Με την μπάρα ολίσθησης του παραθύρου “Slider Gain 2”, μπορεί να ελεγχθεί η τιμή της ακτίνας της σφαίρας (Σχήμα 3.17).



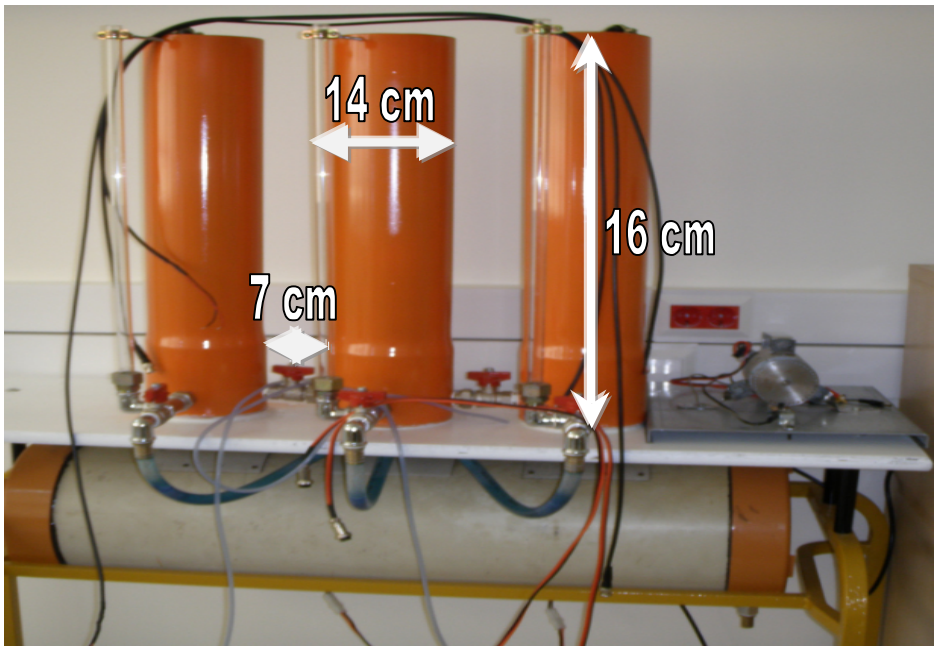
Σχήμα 3.17 Μπάρες ολίσθησης του μοντέλου προσομοίωσης και προβολή από ειδικό λογισμικό προσομοίωσης του λογισμικού Matlab.

ΚΕΦΑΛΑΙΟ 4

ΑΝΑΠΤΥΞΗ ΓΡΑΦΙΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ ΓΙΑ ΤΗΝ ΠΡΟΣΟΜΟΙΩΣΗ ΚΑΙ ΤΟΝ ΕΛΕΓΧΟ ΕΝΟΣ ΣΥΣΤΗΜΑΤΟΣ ΤΡΙΩΝ ΔΕΞΑΜΕΝΩΝ

4.1 ΕΙΣΑΓΩΓΗ

Το υδραυλικό σύστημα τριών δεξαμενών αποτελείται από τρεις δεξαμενές (T_1, T_2, T_3) ίσου ύψους 63 cm και 14 cm περιμέτρου (Σχήμα 4.1). Οι δεξαμενές απέχουν μεταξύ τους κατά 7 cm συνδέονται μέσω τριών αγωγών οι οποίοι έχουν βαλβίδες για τη χειροκίνητη μεταβολή των αντίστοιχων υδραυλικών αντιστάσεων. Οι αντλίες 1 και 2 τροφοδοτούν με ρευστό τις δεξαμενές T_1 και T_2, αντίστοιχα. Επιπλέον, κάθε δεξαμενή διαθέτει ένα πιεζοηλεκτρικό αισθητήρα για τη μέτρηση της στάθμης του περιεχομένου της [17].



Σχήμα 4.1 Πραγματικό υδραυλικό σύστημα τριών δεξαμενών

Το υδραυλικό σύστημα τριών δεξαμενών (Σχήμα 4.2) περιγράφεται από τις κάτωθι διαφορικές εξισώσεις:

$$A_1\rho\frac{dh_1}{dt} = q_1 - \frac{\rho g}{R_1}(h_1 - h_3)$$

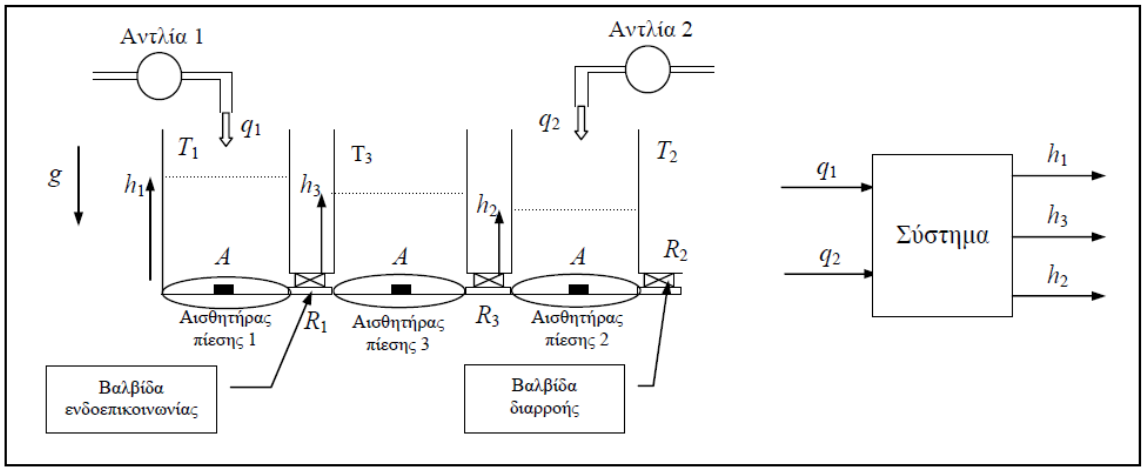
$$A_3\rho\frac{dh_3}{dt} = \frac{\rho g}{R_1}(h_1 - h_3) - \frac{\rho g}{R_3}(h_3 - h_2)$$

$$A_2\rho\frac{dh_2}{dt} = q_2 + \frac{\rho g}{R_3}(h_3 - h_2) - \frac{\rho g}{R_2}h_2$$

Αφού επιλεγούν ως μεταβλητές κατάστασης $z = [z_1\ z_2\ z_3\ ]^T = [h_1\ h_3\ h_2]$, θέτοντας $R_1=R_2=R_3=R$ και $A_1=A_2=A_3=A$. Ως είσοδοι του συστήματος ορίζονται οι παροχές q_1 και q_2 και ως έξοδοι τα ύψη h_1, h_2, h_3 και έτσι προκύπτουν οι ακόλουθες εξισώσεις κατάστασης:

$$\underline{A} = \begin{bmatrix} -g/AR & g/AR & 0 \\ g/AR & -2g/AR & g/AR \\ 0 & g/AR & -2g/AR \end{bmatrix} \quad \underline{B} = \begin{bmatrix} 1/A\rho & 0 \\ 0 & 0 \\ 0 & 1/A\rho \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \underline{D} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\underline{u} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \quad \underline{y} = \begin{bmatrix} h_1 \\ h_3 \\ h_2 \end{bmatrix}$$



Σχήμα 4.2 Υδραυλικό σύστημα τριών δεξαμενών

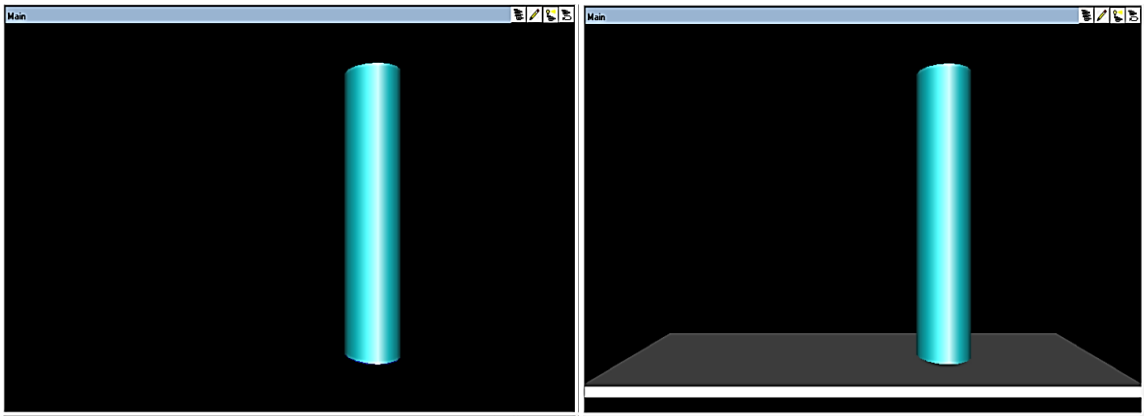
4.2 ΑΝΑΠΤΥΞΗ ΕΙΚΟΝΙΚΟΥ ΜΟΝΤΕΛΟΥ ΣΕ VRML

Για την ανάπτυξη του εικονικού συστήματος των τριών δεξαμενών χρησιμοποιήθηκε το λογισμικό V-Realm Builder. Αρχικά σχεδιάστηκε η βάση πάνω στην οποία θα στηριχθεί το σύστημα των τριών δεξαμενών (Σχήμα 4.3). Επιλέγεται λοιπόν από την εργαλειομπάρα γεωμετρικών σχημάτων (Geometry Node ToolBar), ένα αντικείμενο

σχήματος κουτιού (box) και διαμορφώνεται όπως φαίνεται παρακάτω προσαρμόζοντας ανάλογα τις διαστάσεις του.

Σχήμα 4.3 Σχεδιασμός βάσης συστήματος τριών δεξαμενών

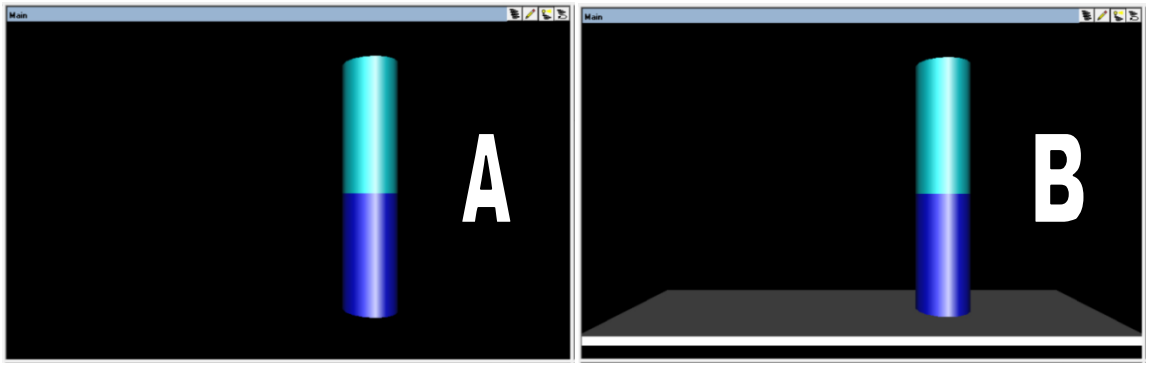
Στη συνέχεια, σχεδιάζεται η πρώτη δεξαμενή κυλινδρικού σχήματος, η οποία αναπαριστά τη δεξαμενή όταν αυτή είναι άδεια (Σχήμα 4.4). Για το σκοπό αυτό, επιλέγεται από την εργαλειομπάρα αντικείμενο κυλινδρικού σχήματος (cylinder) και προσαρμόζονται οι διαστάσεις του κατά τον ίδιο τρόπο.



Σχήμα 4.4 *Σχεδιασμός της πρώτης δεξαμενής*

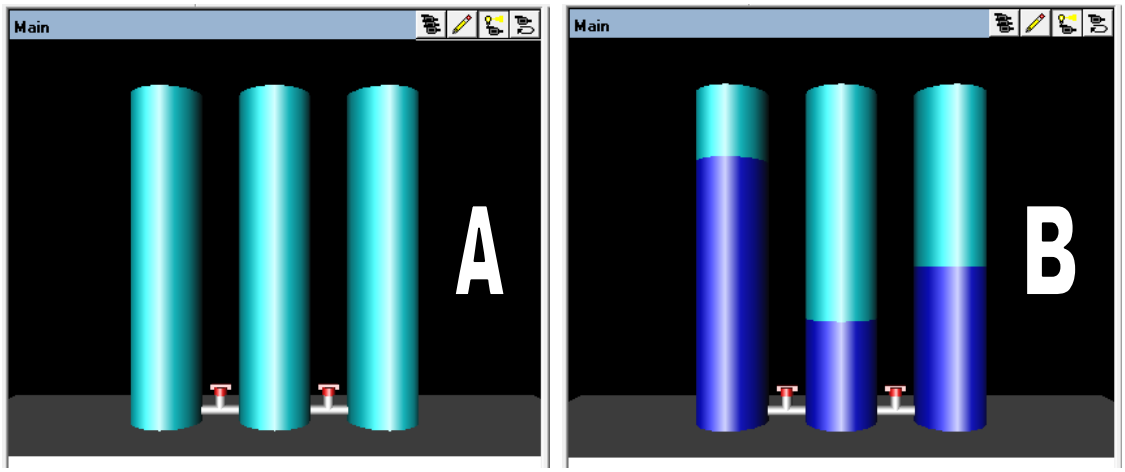
Έπειτα ακολουθώντας την ίδια διαδικασία επιλέγεται ένας δεύτερος κύλινδρος, ο οποίος απεικονίζει το νερό που βρίσκεται στη δεξαμενή, δηλαδή τη στάθμη της (Σχήμα 4.5.Α) και στη συνέχεια συνδέεται με τη βάση της δεξαμενής (Σχήμα 4.5.Β). Ο πρώτος κύλινδρος είναι πάντα σταθερός, ενώ ο δεύτερος ρυθμίζεται έτσι ώστε να μεταβάλλονται οι διαστάσεις του και προς τις δύο κατευθύνσεις και μετατοπίζεται κατά άξονα y. Ο

κύλινδρος αναπαριστά την αυξομείωση του ύψους της στάθμης της δεξαμενής, η τιμή του οποίου εξαρτάται από την τιμή που θα του δώσει ο χρήστης.



Σχήμα 4.5 Απεικόνιση της στάθμης νερού της δεξαμενής

Με τον ίδιο τρόπο επιλέγονται άλλα δύο αντικείμενα κυλινδρικού σχήματος και ίδιων διαστάσεων, τα οποία αναπαριστούν τις άλλες δύο δεξαμενές του συστήματος (Σχήμα 4.6.Α). Στη συνέχεια προστίθενται οι επόμενες δύο δεξαμενές που αναπαριστούν το ύψος της στάθμης των δεξαμενών (Σχήμα 4.6.Β). Τέλος, προστέθηκαν δύο βαλβίδες επικοινωνίας οι οποίες περιστρέφονται για μία πιο ρεαλιστική προσέγγιση της αναπαράστασης του συστήματος των τριών δεξαμενών (Σχήμα 4.6.Α και Σχήμα 4.6.Β).



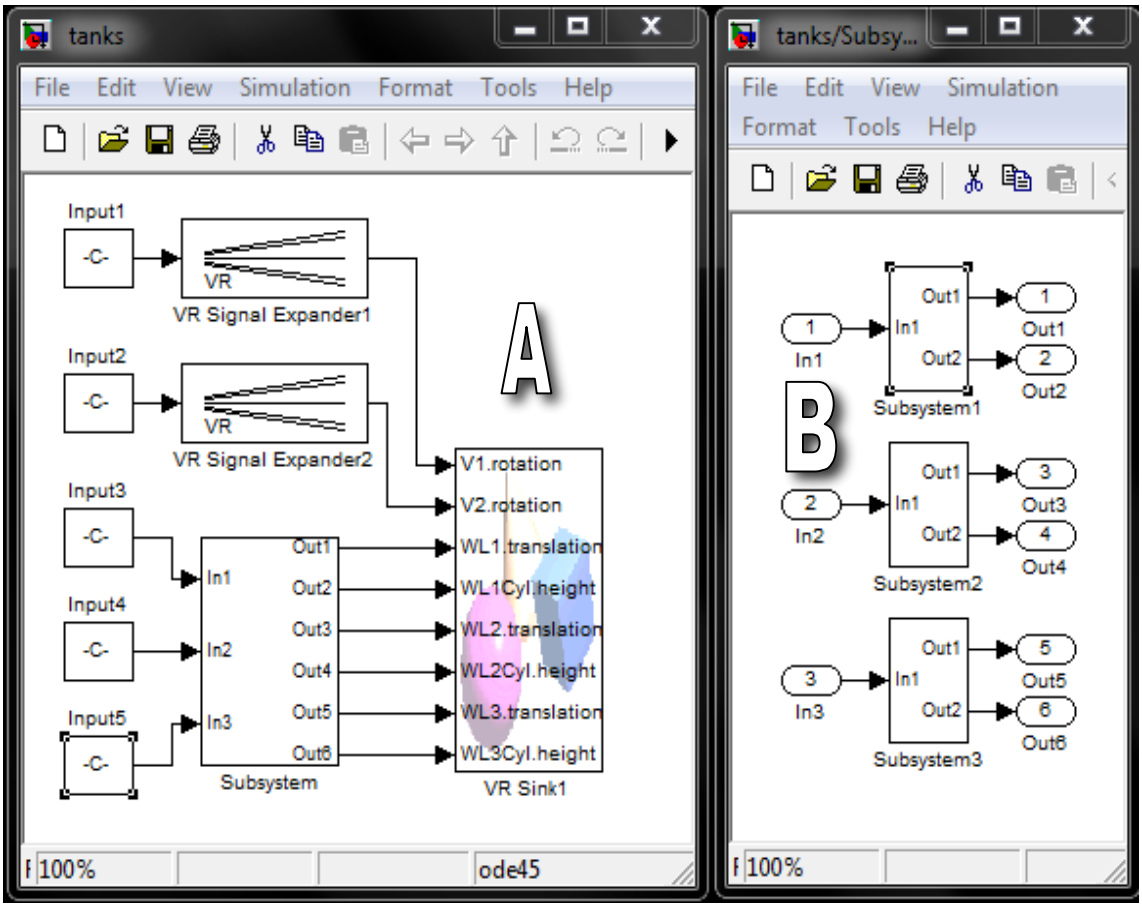
Σχήμα 4.6 Αναπαράσταση συστήματος τριών δεξαμενών στον εικονικό κόσμο

4.3 ΣΥΝΔΕΣΗ ΤΟΥ ΕΙΚΟΝΙΚΟΥ ΚΟΣΜΟΥ ΜΕ ΤΟ SIMULINK

Για να λειτουργήσει ο εικονικός κόσμος, είναι απαραίτητο να συνδεθεί με Simulink model, δηλαδή με το μοντέλο προσομοίωσης με χρήση του λογισμικού Simulink, όπως αναφέρθηκε στην παράγραφο 3.7. Αφού σχεδιαστεί ο εικονικός κόσμος μπορεί να

ελεγχθεί του μέσω του Simulink model, δηλαδή με ένα διαδραστικό περιβάλλον (Graphical User Interface - GUI) που διαθέτει το λογισμικό Matlab.

Το Simulink model έχει τη δυνατότητα να δέχεται έναν εικονικό κόσμο (vrml) αλλά και να δίνει τιμές σε αυτόν μεταβάλλοντας διάφορες μεταβλητές όπως το ύψος, την ακτίνα κ.α.. Τις τιμές αυτές τις παίρνει από το χρήστη μέσω του GUI που έχει σχεδιαστεί κατάλληλα και οι μεταβολές που προκύπτουν παρουσιάζονται στον εικονικό κόσμο. Το Simulink model που παρουσιάζεται παρακάτω (Σχήμα 4.7.Α) αποτελείται από τα εξής δομικά τμήματα, το VR Sink, τα VR Signal Expander, τα Constant (Input) και το Subsystem.



Σχήμα 4.7 Μοντέλο προσομοίωσης στο λογισμικό Matlab κα ανάλυση του κεντρικού υποσυστήματος

Αναλυτικότερα, με το VR Sink συνδέεται ο εικονικός κόσμος με το simulink, αφού μας δίνει τη δυνατότητα να εισάγουμε ένα αρχείο vrml με κατάληξη *.wrl και μέσω των επιλογών που διαθέτει να οριστεί ο αριθμός των εισόδων αλλά και τα χαρακτηριστικά τους.

Σε αυτή την περίπτωση έχουν επιλεγεί ως είσοδοι οι δύο βαλβίδες και τρεις στάθμες του νερού των δεξαμενών. Κάθε κύλινδρος μεταβάλει το ύψος του και τη θέση του ως προς

τους άξονες x, y και z, άρα έχει συνολικά οκτώ εισόδους.

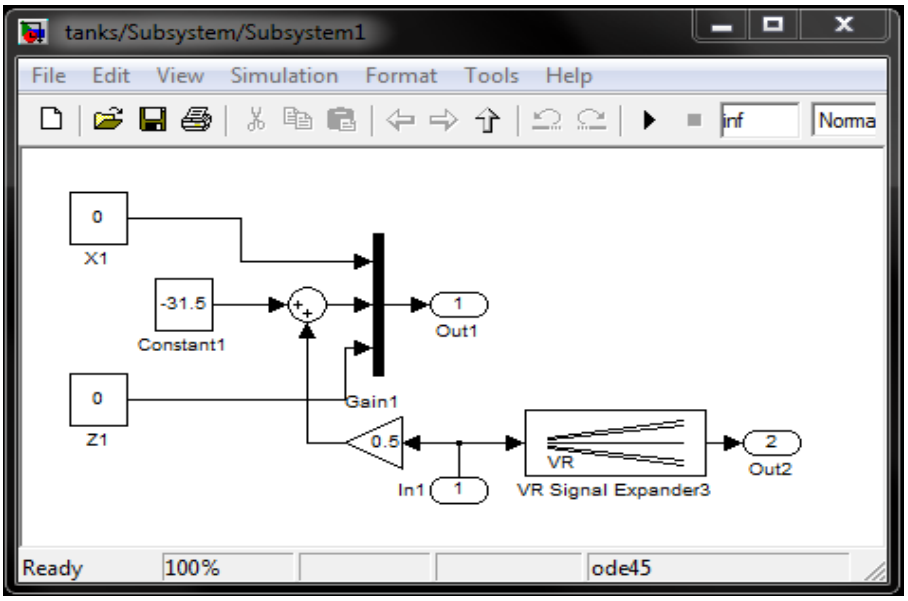
Οι σταθερές (Constant) που έχουμε μετονομάσει σε Input, είναι οι είσοδοι που παίρνουμε από το GUI. Έτσι συνδέονται το GUI με το Simulink model. Επιπλέον, μπορούμε να εισάγουμε τις μεταβλητές που χρειάζονται από το GUI (value_p1,value_p2,value_h1,value_h2 και value_h3) αλλά και άλλες σταθερές. Συγκεκριμένα για την περιστροφή της πρώτης βαλβίδας διαιρέθηκε η τιμή value_p1

με το 100, γιατί έχει οριστεί στο GUI να παίρνει τιμές από το 1 έως το 100 και πολλαπλασιάστηκε με το pi/2, όπου pi=3,14, για να γίνει η μετατροπή των ακτινίων σε δηλαδή (value_p1/100)*(pi/2). Η ίδια διαδικασία ακολουθήθηκε και για τη δεύτερη βαλβίδα δηλαδή (value_p2/100)*(pi/2).

Όσον αφορά το ύψος της πρώτης δεξαμενής ορίστηκε η τιμή value_h1 πολλαπλασιάζεται με το 100 για να είναι πιο εύκολος ο έλεγχος των αποτελεσμάτων αφού έχει οριστεί στο GUI να παίρνει τιμές από το 0 έως το 1 και στην συνέχεια πολλαπλασιάζεται με την τιμή 63/100 γιατί το ύψος των δεξαμενών του πραγματικού και του εικονικού συστήματος είναι 63cm και διαιρείται με το 100 ώστε να διευκολυνθεί η διαδικασία του ελέγχου των τιμών. Δηλαδή ορίστηκε value_h1*100)*(63/100) [στην προκειμένη περίπτωση θα μπορούσε να απλοποιηθεί η τιμή σε value_h1*63]. Ομοίως και για τις άλλες δύο δεξαμενές ορίστηκε αντίστοιχα (value_h2*100)*(63/100) και (value_h3*100)*(63/100).

Το διάγραμμα VR Signal Expander είναι υπεύθυνο για την επεξεργασία του σήματος που φτάνει στην είσοδο του VR Sink και είναι απαραίτητο για την περιστροφική κίνηση των βαλβίδων και για τις μεταβολές στο ύψος των κυλίνδρων για την στάθμη των δεξαμενών.

Το Subsystem είναι η συγχώνευση κάποιων δομικών τμημάτων (όπως Constant,Mux,Gain και VR Signal Expander) και χρησιμεύει στον έλεγχο των δεξαμενών. Δέχεται τρεις εισόδους, μία για κάθε δεξαμενή και έχει έξι εξόδους, δύο για κάθε μεταβολή του κυλίνδρου στάθμης (ύψος και θέση) τις κάθε δεξαμενής. Επιπλέον, εμπεριέχει τρία υποσυστήματα Subsystem1,Subsystem2 και Subsystem3), όπως φαίνεται και στο (Σχήμα 4.7.Β), κάθε ένα από αυτά δέχεται από μία είσοδο και έχει δύο εξόδους. Δηλαδή αποτελείται από δύο τμήματα, το ένα για την κίνηση του κυλίνδρου στο χώρο, ως προς τους άξονες x, y και z και το άλλο τμήμα που είναι υπεύθυνο για το ύψος του κυλίνδρου.



Σχήμα 4.8 Ανάλυση υποσυστήματος για τον έλεγχο της κάθε δεξαμενής

Για την κίνηση στο χώρο απαιτούνται τρία διαγράμματα Constant (για x, y και z αντίστοιχα) τα οποία συνδέονται με ένα διάγραμμα ενός Mux τριών εισόδων και οδηγεί αυτό το σήμα στην είσοδο WL1.translation του VRSink χωρίς να χρειάζεται το VR Signal Expander αφού πρόκειται για κίνηση στο χώρο.

Αφού ο κύλινδρος στάθμης κινείται κατά τον άξονα y ορίστηκαν τα Constant των x και z ίσα με το μηδέν. Ο λόγος που ο κύλινδρος στάθμης κινείται κατά τον άξονα y, δηλαδή είτε προς τα πάνω είτε προς τα κάτω, είναι γιατί όταν αυξάνεται ή μειώνεται αντίστοιχα, το ύψος του κυλίνδρου και γίνεται ταυτόχρονα και από τις δύο του πλευρές.

Το σήμα εισόδου y, προκύπτει από μια απλή εξίσωση (0.5*Input-31.5) που περιλαμβάνει τη μεταβλητή του σήματος εξόδου του Constant που έχει οριστεί ως είσοδος (Input) για την στάθμη της κάθε δεξαμενής και πολλαπλασιάζεται με το 0.5

μέσω του διαγράμματος Gain και από εκεί διαιρείται με την τιμή του ύψους της δεξαμενής με το δύο, δηλαδή 63/2=31.5, χρησιμοποιώντας τα διαγράμματα Sum και Constant (Σχήμα 4.8)

Το ύψος του κυλίνδρου της στάθμης των δεξαμενών, προκύπτει από το σήμα εξόδου του κάθε Constant Input3, Input4 και Input5 αντίστοιχα για κάθε δεξαμενή και οδηγείται σε ένα VR Signal Expander, που είναι απαραίτητο για την μεταβολή του ύψους του κυλίνδρου και έπειτα στην είσοδο WL1Cyl.height του VR Sink.

Αφού ανοιχθεί το simulink model, τρέχοντας το mfile του GUI, δίνεται η δυνατότητα μέσα από αυτό να μεταβληθούν τα ύψη των δεξαμενών και να μελετηθούν παρατηρώντας τις σχετικές γραφικές παραστάσεις και τα αποτελέσματα υπολογισμών που εμφανίζονται είτε στο command window είτε στο vrml viewer του Matlab.

4.4 ΠΑΡΑΣΤΑΣΗ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ ΔΥΝΑΜΙΚΗΣ ΣΥΣΤΗΜΑΤΩΝ ΣΤΟ ΠΕΔΙΟ ΤΟΥ ΧΡΟΝΟΥ

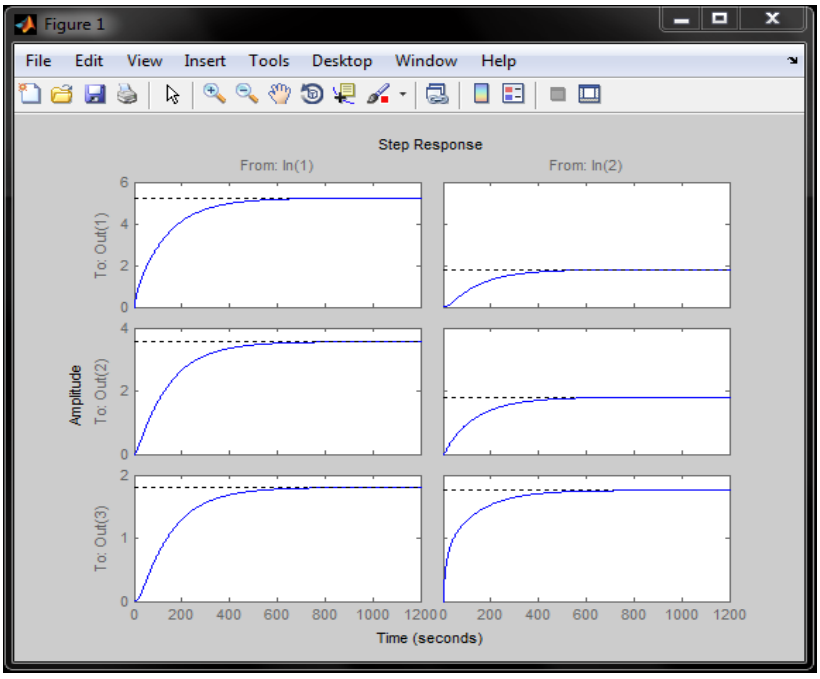
Στην παράγραφο αυτή γίνεται μοντελοποίηση και προσομοίωση του συστήματος τριών δεξαμενών. Η μοντελοποίηση γίνεται με τη βοήθεια των διαφορικών εξισώσεων και των εξισώσεων χρόνου κατάστασης. Επιπλέον, υπολογίζονται οι αποκρίσεις του συστήματος στο συνεχή χρόνο και παρουσιάζονται τα αντίστοιχα γραφήματα.

Συγκεκριμένα μελετάται η εξαναγκασμένης απόκρισης του (Σχήμα 4.9). Για το σκοπό αυτό πληκτρολογείται στο Command Window του Matlab αφού οριστούν οι πίνακες αναπαράστασης του χώρου κατάστασης (A, B, C και D) οι ακόλουθες εντολές.

```
A=[-0.039 0.039 0; 0.039 -0.077 0.039; 0 0.039 -0.077];
B=[0.065 0; 0 0; 0 0.065];
C=eye(3);
D=zeros(3,2);
sysc=ss(A,B,C,D);
```

Για τον υπολογισμό της εξαναγκασμένης απόκρισης σε εισόδους μοναδιαίας βαθμίδας (q_1=1lt/sec και q_2=1lt/sec) (Σχήμα 4.8) πληκτρολογείται η εντολή:

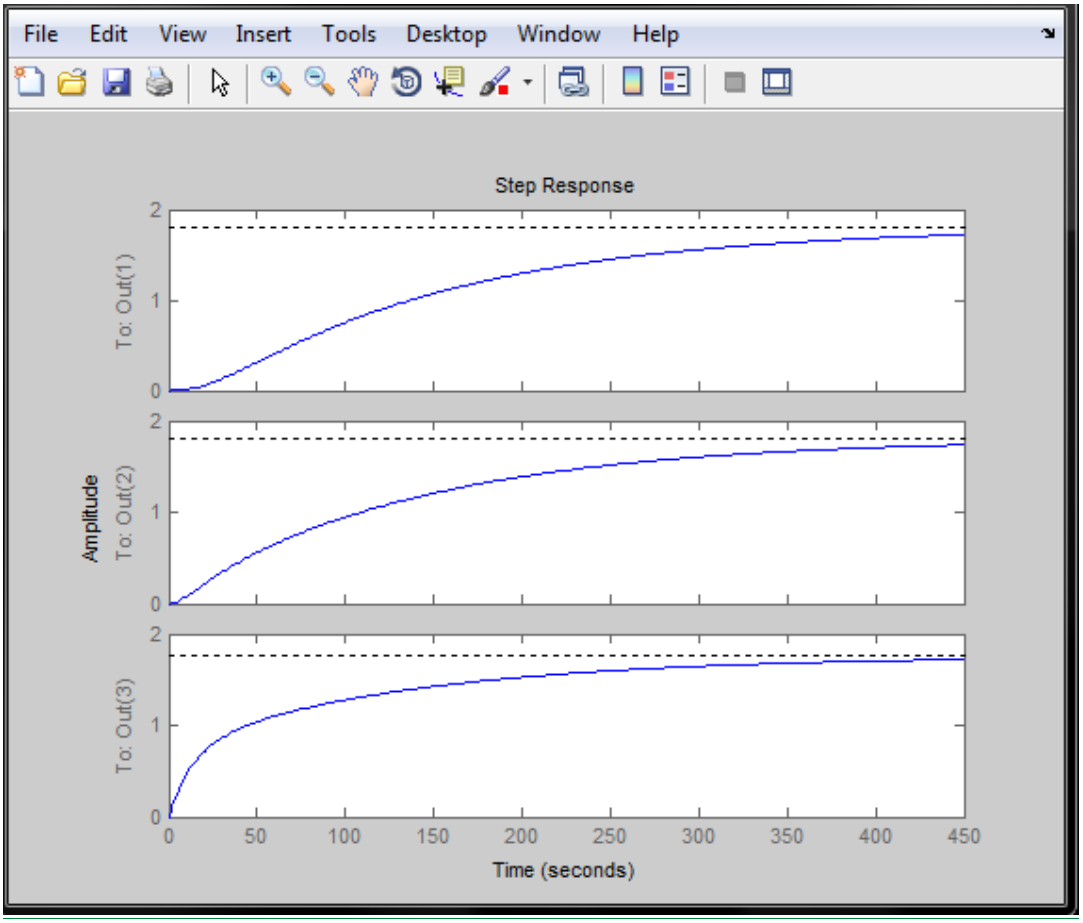
```
figure,step(A,B,C,D)
```



Σχήμα 4.9 Εξαναγκασμένη απόκριση σε εισόδους μοναδιαίας βαθμίδας (q_1=1lt/sec και q_2=1lt/sec) (συνεχής χρόνος)

Το γράφημα της εξαναγκασμένης απόκρισης σε είσοδο μοναδιαία βαθμίδα, q_2=1lt/sec για χρόνο 450 sec (Σχήμα 4.10) εμφανίζεται πληκτρολογώντας την εντολή:

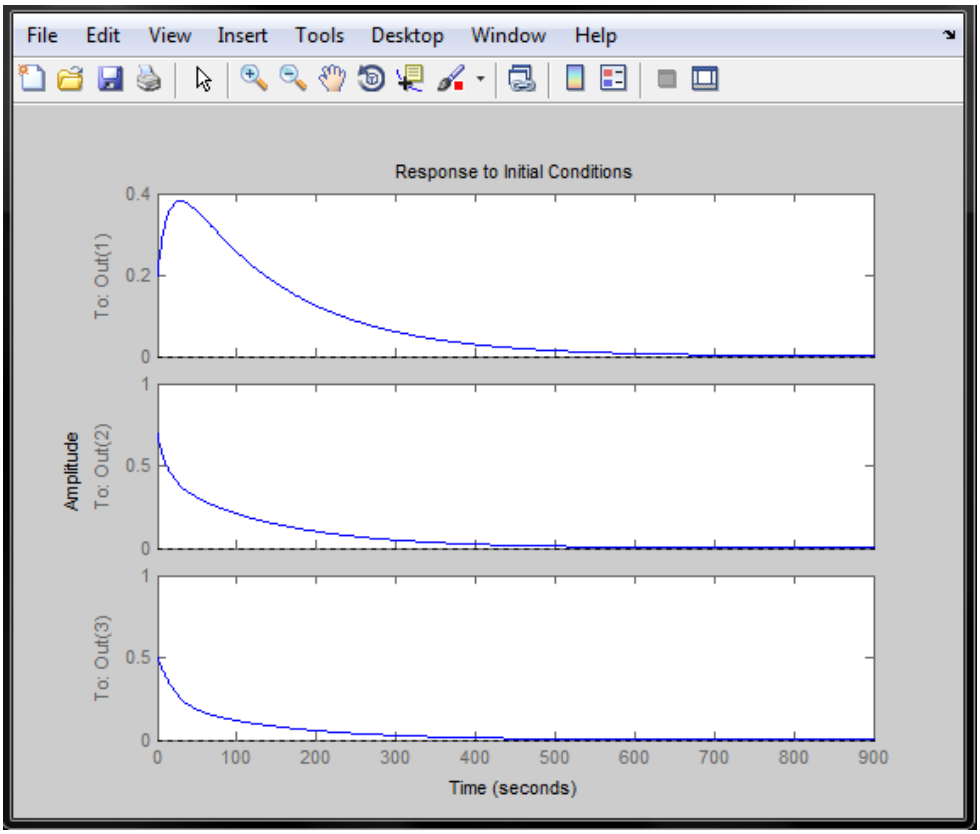
```
figure,step(A,B,C,D,2,450)
```



Σχήμα 4.10 Εξαναγκασμένη απόκριση σε είσοδο μοναδιαία βαθμίδα, q_2=1lt/sec

Η ελεύθερη απόκριση σε αρχικές συνθήκες h_1=0.2m, h_3=0.7m και h_2=0.5m (Σχήμα 4.11), υπολογίζεται με την εντολή:

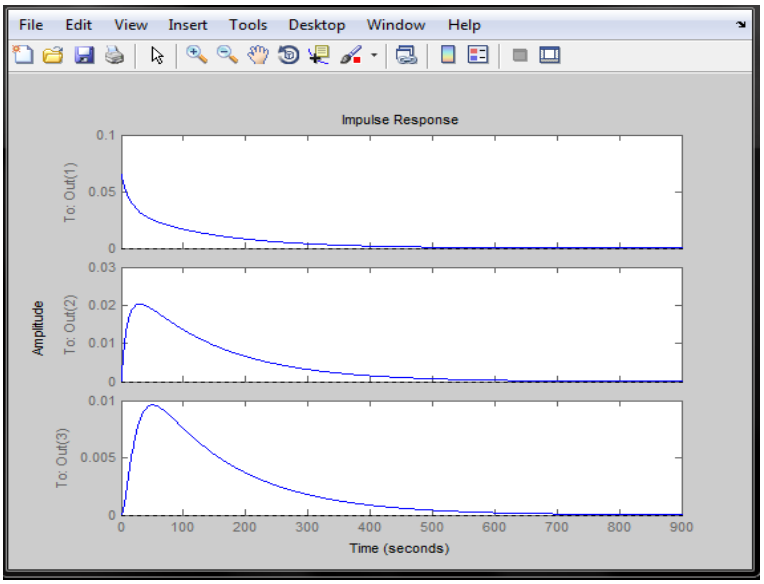
```
figure,initial(A,B,C,D,[0.2 0.7 0.5])
```



*Σχήμα 4.*11 Ελεύθερη απόκριση

Η εξαναγκασμένη απόκριση σε κρουστική είσοδο, παροχή q_1, (Σχήμα 4.12) υπολογίζεται με την εντολή:

```
figure,impulse(A,B,C,D,1)
```



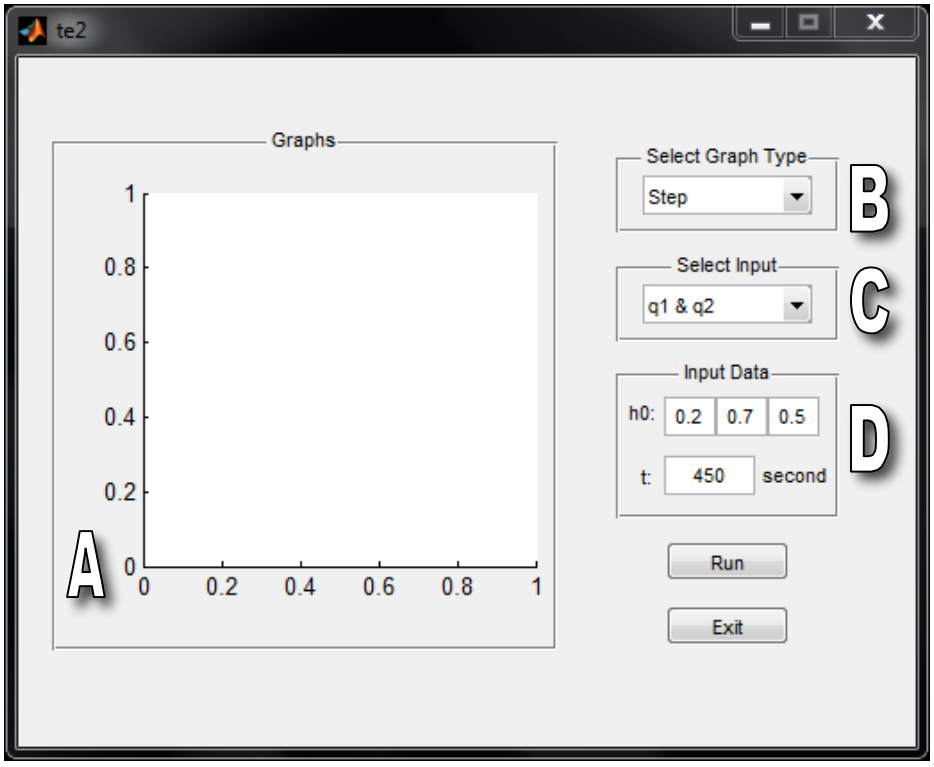
Σχήμα 4.12 Εξαναγκασμένη απόκριση σε κρουστική είσοδο q_1

4.4.1 ΑΝΑΠΤΥΞΗ ΓΡΑΦΙΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ ΔΙΕΠΑΦΗΣ ΓΙΑ ΤΗΝ ΠΑΡΑΣΤΑΣΗ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ ΔΥΝΑΜΙΚΗΣ ΣΥΣΤΗΜΑΤΩΝ ΣΤΟ ΠΕΔΙΟ ΤΟΥ ΧΡΟΝΟΥ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΣΟΜΟΙΩΣΗΣ

Για τη μελέτη της δυναμικής συστημάτων στο πεδίο του χρόνου σε περιβάλλον προσομοίωσης, αναπτύχθηκε το περιβάλλον διεπαφής που παρουσιάζεται στη συνέχεια (Σχήμα 4.13).

Το γραφικό περιβάλλον διεπαφής (Graphical User Interface - GUI), ο σχεδιασμός των εικονικών κόσμων απλοποιείται, δίνοντας στο χρήστη τη δυνατότητα να εισάγει τιμές του ύψους του συστήματος των δεξαμενών και να υπολογίσει τις μεταβλητές που επιθυμεί.

Σχεδιάζονται στο Matlab editor και δημιουργούνται αυτόματα mfiles στα οποία ο χρήστης μπορεί να παρέμβει προσθέτοντας εντολές και στη συνέχεια να τα συνδέσει με το Simulink model (Σχήμα 4.13).



Σχήμα 4.13 Γραφική διεπαφή χρήστη προσομοίωσης συστήματος τριών δεξαμενών

Στο συγκεκριμένο γραφικό περιβάλλον δίνεται η επιλογή στο χρήστη, να μελετήσει τις γραφικές αποκρίσεις που τον ενδιαφέρουν, έχοντας μάλιστα τη δυνατότητα να αλλάξει κάποιες μεταβλητές.

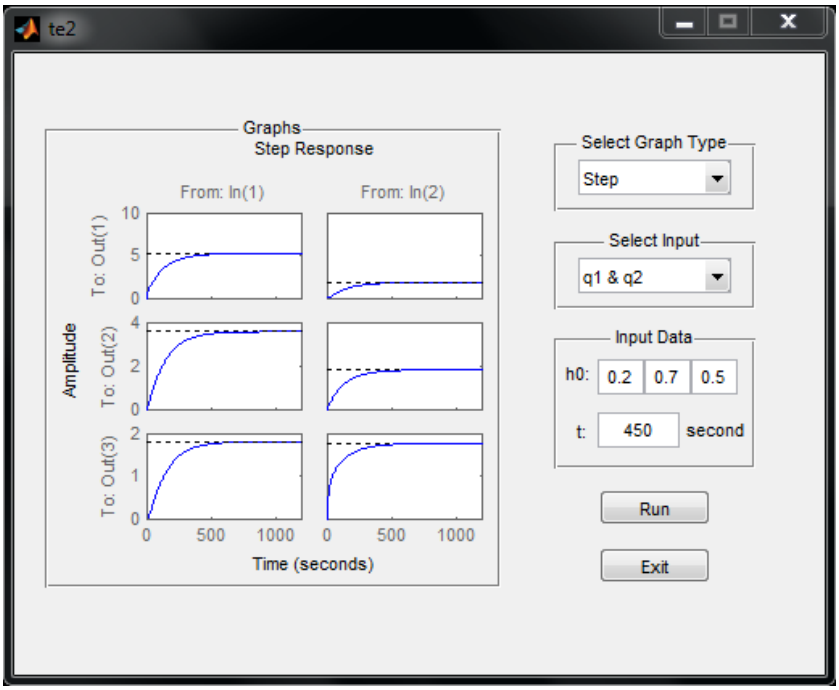
Με την επιλογή τύπου γραφήματος (Select Graph Type) ο χρήστης επιλέγει την είσοδο που επιθυμεί για το γράφημα που θέλει να παρουσιάσει. Ακόμα, μπορεί να επιλέξει μεταξύ ελεύθερης απόκρισης (Initial), εξαναγκασμένης απόκρισης μοναδιαίας βαθμίδας (Step) και εξαναγκασμένης απόκρισης σε κρουστική είσοδο (Impulse) (Σχήμα 4.13.B).

Ακόμη, μπορεί να επιλέξει την είσοδο του συστήματος (Select Input) που επιθυμεί μεταξύ q_1, q_2 και q_1&q_2 (Σχήμα 4.13.C).

Τέλος, ο χρήστης μπορεί να επιλέξει μεταξύ προκαθορισμένων αρχικών τιμών της στάθμης των δεξαμενών (h_0) και του χρονικού διαστήματος (t). Εναλλακτικά μπορεί να ορίσει διαφορετικές τιμές για τις ακόλουθες μεταβλητές (Σχήμα 4.13.D).

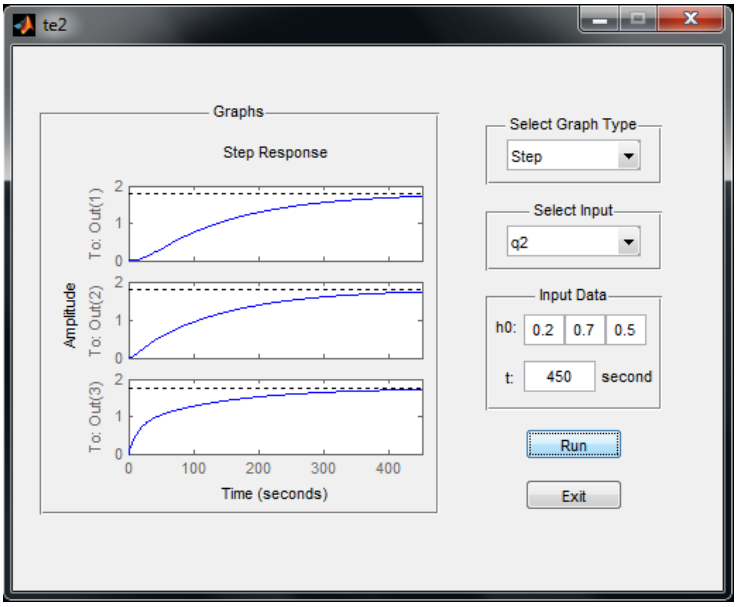
Σκοπός αυτής της ενότητας είναι η μοντελοποίηση και προσομοίωση του συστήματος τριών δεξαμενών, με τη βοήθεια των διαφορικών εξισώσεων και των εξισώσεων χρόνου κατάστασης και η μελέτη των αποτελεσμάτων που θα προκύψουν. Υπολογίζονται οι αποκρίσεις του συστήματος στο συνεχή και στο διακριτό χρόνο και παρουσιάζονται τα αντίστοιχα γραφήματα.

Συγκεκριμένα μελετάται η ελεύθερη απόκριση του συστήματος σε συνθήκες h_o=[0.2 0.7 0.5] (m) στο χρονικό διάστημα t= [0 450] (sec) (Σχήμα 4.14). Για το σκοπό αυτό επιλέγεται από το γραφικό περιβάλλον διεπαφής η απόκριση Step σε είσοδο μοναδιαία βαθμίδα συνεχούς χρόνου με q_1=1lt/sec και q_2=1lt/sec και στη συνέχεια επιλέγεται το κουμπί Run.



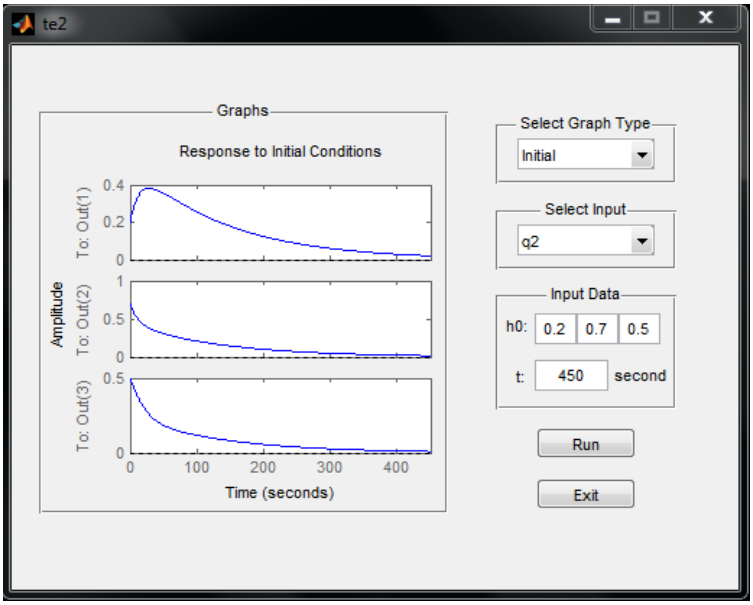
Σχήμα 4.14 Ελεύθερη απόκριση (συνεχής χρόνος)

Για τον υπολογισμό της εξαναγκασμένης απόκρισης σε είσοδο μοναδιαία βαθμίδα q_2, και μηδενικές αρχικές συνθήκες στο χρονικό διάστημα t=[0 450](sec) (Σχήμα 4.15), επιλέγεται το Step και στη συνέχεια το κουμπί Run.



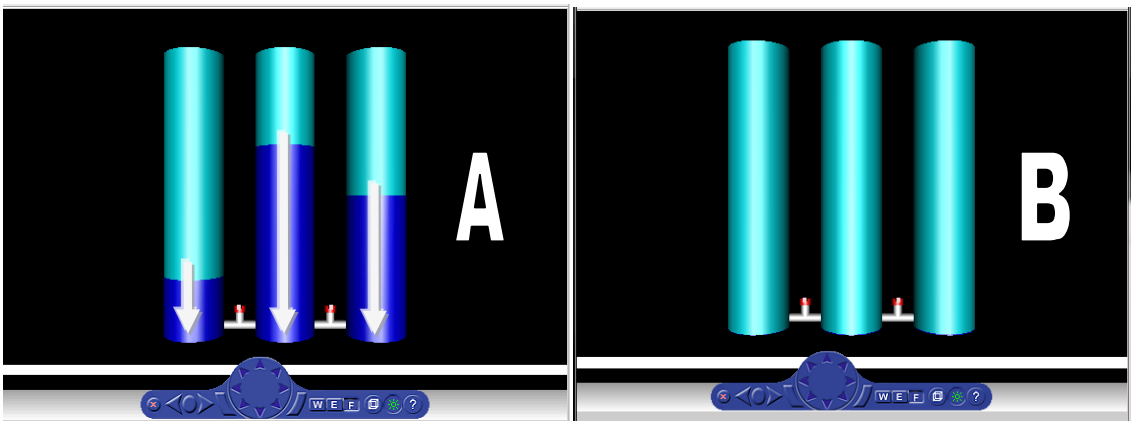
Σχήμα 4.15 Εξαναγκασμένη απόκριση σε είσοδο μοναδιαία βαθμίδα q_2

Το γράφημα της εξαναγκασμένης απόκρισης σε ελεύθερη απόκριση σε αρχικές συνθήκες h_1= 0.2m, h_2= 0.7m, h_3= 0.5m (Σχήμα 4.16) εμφανίζεται επιλέγοντας το Initial και στη συνέχεια το Run (Σχήμα 4.13).



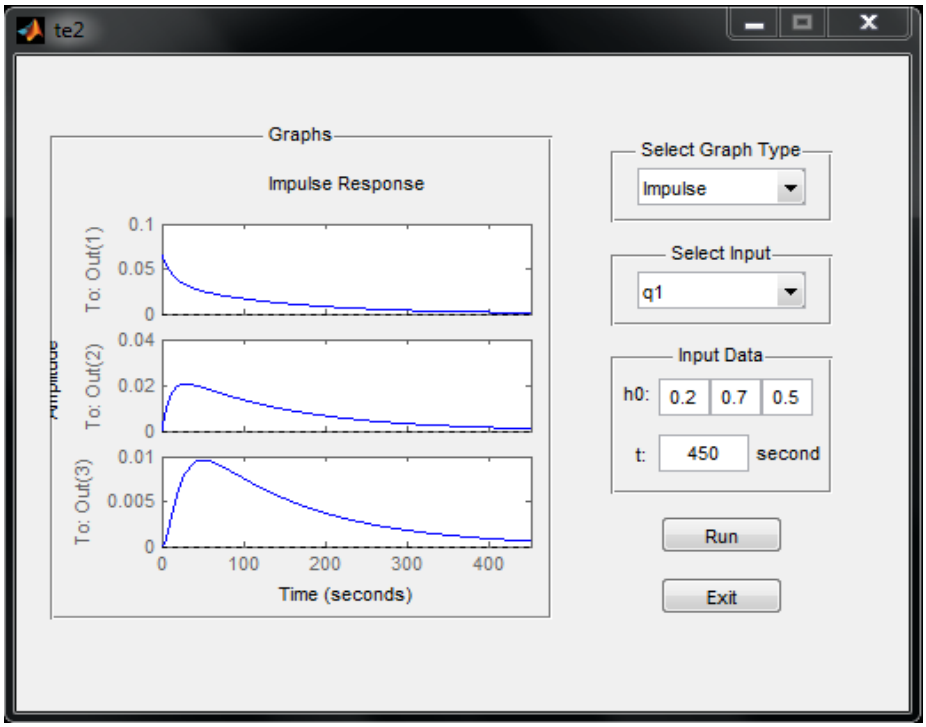
Σχήμα 4.16 Εξαναγκασμένη απόκριση σε κρουστικές εισόδους (q_1 και q_2) (συνεχής χρόνος)

Στον εικονικό κόσμο οι δεξαμενές αρχικά είναι γεμάτες (Σχήμα 4.17.Α) και σταδιακά η στάθμη τους μειώνεται έως να φτάσει στο μηδέν (Σχήμα 4.17.Β).



Σχήμα 4.17 Προβολή του εικονικού κόσμου των δεξαμενών από το vrml viewer του Matlab

Τέλος, υπολογίζεται η εξαναγκασμένη απόκριση σε κρουστική είσοδο παροχή q_1 (Σχήμα 4.18).



Σχήμα 4.18 Εξαναγκασμένη απόκριση

4.5 ΠΑΡΑΣΤΑΣΗ, ΠΡΟΣΟΜΟΙΩΣΗ ΚΑΙ ΑΝΑΛΥΣΗ ΔΥΝΑΜΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ ΣΤΟ MATLAB: ΠΕΔΙΟ LAPLACE ΚΑΙ ΖΗΤΑ

Στην παράγραφο αυτή γίνεται μαθηματική αναπαράσταση του χώρου κατάστασης, προσομοίωση και ανάλυση του συστήματος τριών δεξαμενών στα πεδία Laplace και Ζήτα. Συγκεκριμένα, υπολογίζονται οι συναρτήσεις μεταφοράς του υδραυλικού συστήματος στα πεδία Laplace (συνεχής χρόνος) και Ζήτα (διακριτός χρόνος).

Επίσης, υπολογίζονται οι πόλοι, οι μηδενιστές, οι συντελεστές κέρδους για την $1^{η}$ και τη $2^{η}$ είσοδο στο πεδίο Ζήτα, καθώς και οι χρονικές σταθερές και τα στατικά κέρδη. Επιπλέον οριστεί η συνάρτηση μεταφοράς $[H_1(s)/E(s)]$, υπολογίζονται οι αποκρίσεις του συστήματος σε είσοδο μοναδιαία βαθμίδα, η απόκριση συχνότητας του συστήματος (bode) συνεχούς χρόνου αλλά και διακριτού και οι φυσικές συχνότητες και οι λόγοι απόσβεσης.

Συγκερκιμένα γίνεται η μεταφορά από το χώρο κατάστασης (state space) σε συναρτήσεις μεταφοράς (transfer function) ως προς την $1^{η}$ είσοδο και εμφανίζεται στο Command window η συνάρτηση μεταφοράς του συστήματος στο συνεχή χρόνο με τις ακόλουθες εντολές;

```
A=[-0.039  0.039  0; 0.039  -0.077  0.039; 0  0.039  -0.077];
B=[0.065  0; 0  0; 0  0.065];
C=eye(3);
D=zeros(3,2);
[num1,den1]=ss2tf(A,B,C,D,1)
num1 =
       0    0.0650    0.0100    0.0003
       0   -0.0000    0.0025    0.0002
       0   -0.0000   -0.0000    0.0001

den1 =
   1.0000    0.1930    0.0089    0.0001

printsys(num1,den1,'s')
 num(1)/den =
      0.065 s^2 + 0.01001 s + 0.00028652
  ---------------------------------------
```

```
   s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005


num(2)/den =
    -2.7756e-017 s^2 + 0.002535 s + 0.00019519
   ------------------------------------------
   s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005


num(3)/den =
    -2.7756e-017 s^2 - 1.7347e-018 s + 9.8865e-005
   ---------------------------------------------
    s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005
```

Ακολουθείται ακριβώς η ίδια διαδικασία και για τη $2^{η}$ είσοδο.

```
[num2,den2]=ss2tf(A,B,C,D,2)
num2 =
        0   -0.0000   -0.0000    0.0001
        0    0.0000    0.0025    0.0001
        0    0.0650    0.0075    0.0001
den2 =
   1.0000    0.1930    0.0089    0.0001


printsys(num2,den2,'s')
num(1)/den =
   -2.7756e-017 s^2 - 1.7347e-018 s + 9.8865e-005
   ---------------------------------------------
    s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005
 num(2)/den =


   1.6653e-016 s^2 + 0.002535 s + 9.8865e-005
   ------------------------------------------
   s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005
```

```
num(3)/den =

     0.065 s^2 + 0.00754 s + 9.633e-005
  ----------------------------------------
  s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005
```

Για τις εισόδους του συστήματος (q_1 και q_2) υπολογίζονται οι πόλοι, οι μηδενιστές, οι συντελεστές κέρδους και τα στατικά κέρδη στο πεδίο Ζήτα, ως εξής:

```
[z1,p1,k1]=tf2zp(num1,den1)

z1 =
  1.0e+013 *
   -0.0000    9.1333   -0.0000
   -0.0000   -0.0000    0.0000
p1 =
   -0.1257
   -0.0600
   -0.0073
k1 =
    0.0650
   -0.0000
   -0.0000

[z2,p2,k2]=tf2zp(num2,den2)

z2 =
  1.0e+013 *
   -0.0000   -1.5222   -0.0000
    0.0000   -0.0000   -0.0000
```

```
p2 =
   -0.1257
   -0.0600
   -0.0073
k2 =
   -0.0000
    0.0000
    0.0650
```

Υπολογισμός χρονικών σταθερών του συστήματος για την 1η είσοδο.

```
taf=1./abs(real(p1))


taf =
    7.9529
   16.6679
  137.6751
```

Υπολογίζονται επίσης τα στατικά κέρδη του συστήματος.

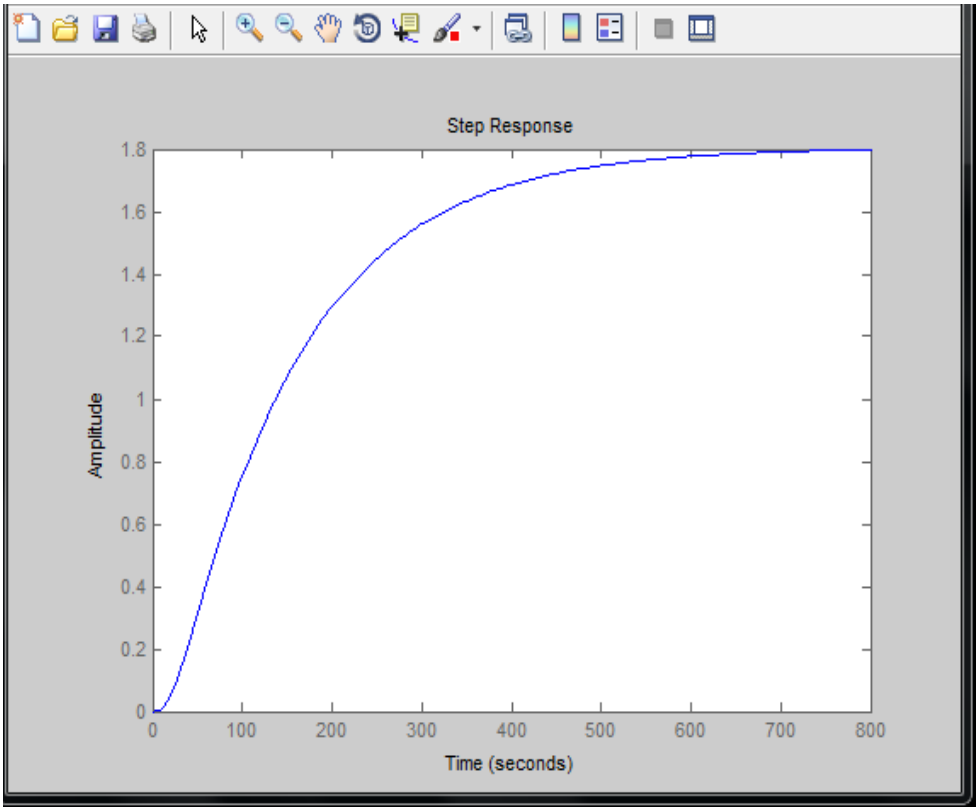
```
dcgain(A,B,C,D)


ans =
    5.2289    1.8043
    3.5623    1.8043
    1.8043    1.7580
```

Οι χρονικές σταθερές και τα στατικά κέρδη του συστήματος για την 1η κα τη 2η είσοδο είναι ίδια.
Ακόμη υπολογίζεται η συνάρτηση μεταφοράς [$H1(s)/E(s)$].

```
sys=tf(9.89e-5,[1 0.193 0.0089 5.48e-5])
Transfer function:
          9.89e-005
-------------------------------------
s^3 + 0.193 s^2 + 0.0089 s + 5.48e-005
```

Απόκριση του συστήματος σε είσοδο μοναδιαία βαθμίδα (Σχήμα 4.19).

```
step(sys),hold
```



Σχήμα 4.19 Απόκριση του συστήματος σε είσοδο μοναδιαία βαθμίδα

Υπολογισμός της μέγιστης ιδιοσυχνότητας του συστήματος.

```
fsmax=max(abs(p1))/2/pi


fsmax =
   0.0200
```

Επιλογή περιόδου δειγματοληψίας για τη διακριτοποίηση του συστήματος.

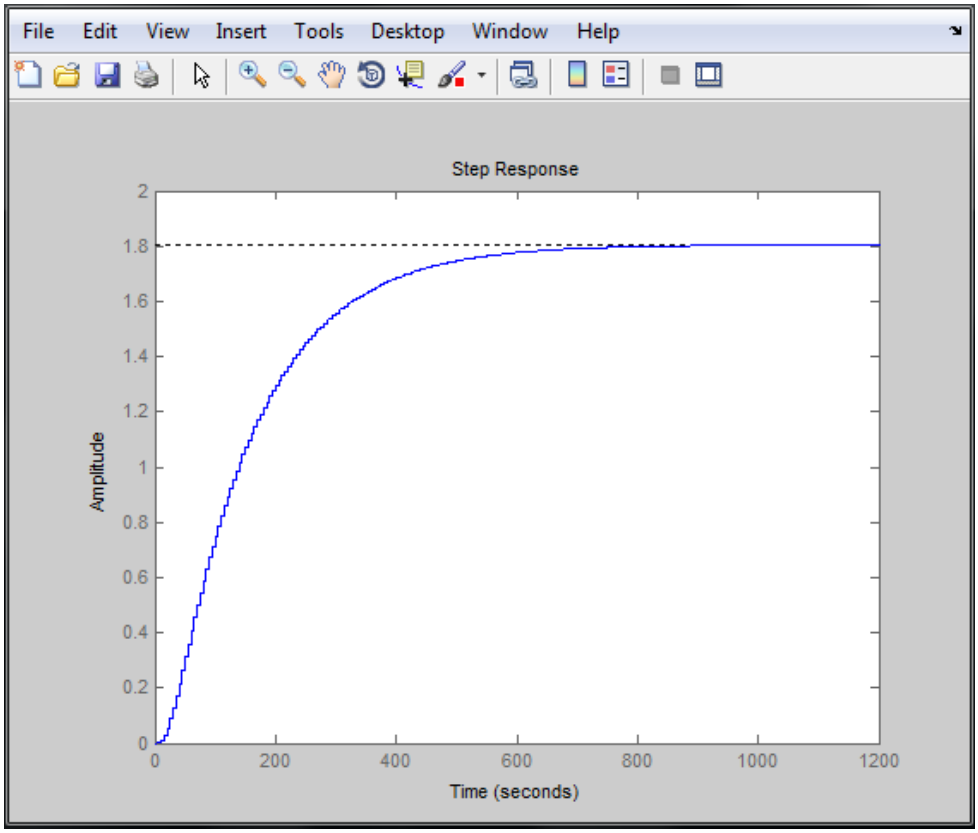
```
Ts=0.2*(1/(2*fsmax))


Ts =
   4.9969
```

Διακριτοποίηση συστήματος συνεχούς χρόνου.

```
sysd=c2d(sys,Ts)

Transfer function:
0.001626 z^2 + 0.005144 z + 0.001004
-------------------------------------
 z^3 - 2.239 z^2 + 1.624 z - 0.3812

Sampling time (seconds): 4.9969
```

Απόκριση του ψηφιακού συστήματος σε είσοδο μοναδιαία βαθμίδα(Σχήμα 4.20).

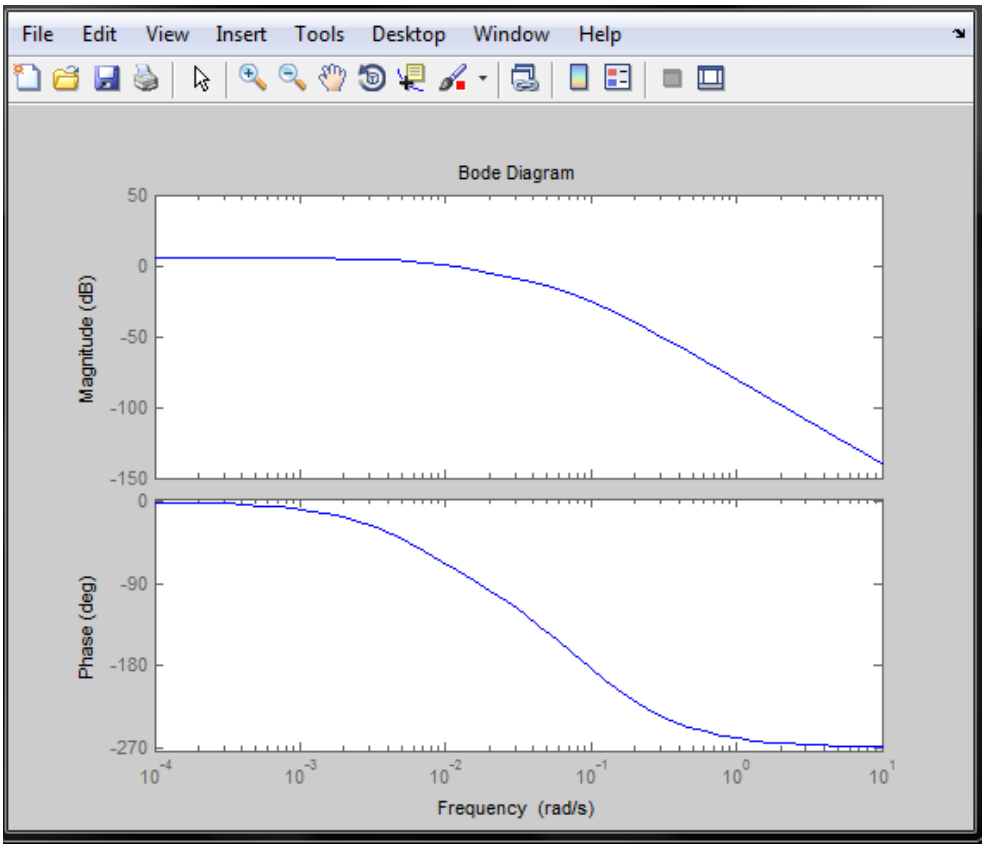
```
step(sysd)
```



Σχήμα 4.20 Απόκριση του ψηφιακού συστήματος σε είσοδο μοναδιαία βαθμίδα.

Απόκριση συχνότητας του συστήματος συνεχούς χρόνου (Σχήμα 4.21).

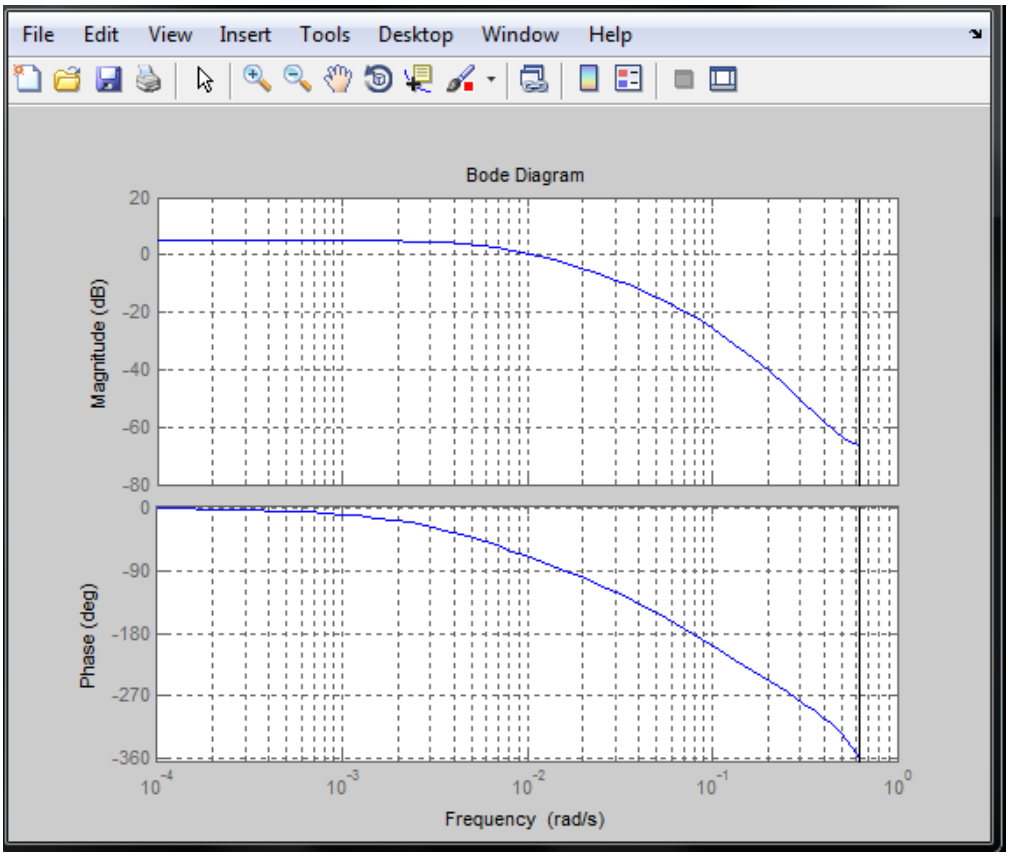
```
figure, bode(sys), hold
Current plot held
```



Σχήμα 4.21 Απόκριση συχνότητας του συστήματος συνεχούς χρόνου.

Απόκριση συχνότητας του ψηφιακού συστήματος (Σχήμα 4.22).

```
bode(sysd), grid
```



Σχήμα 4.22 Απόκριση συχνότητας του ψηφιακού συστήματος.

Υπολογισμός φυσικών συχνοτήτων και λόγων απόσβεσης.

```
[wn,zita]=damp(sys)


wn =
   0.0073
   0.0601
   0.1256
zita =
    1
    1
    1
```

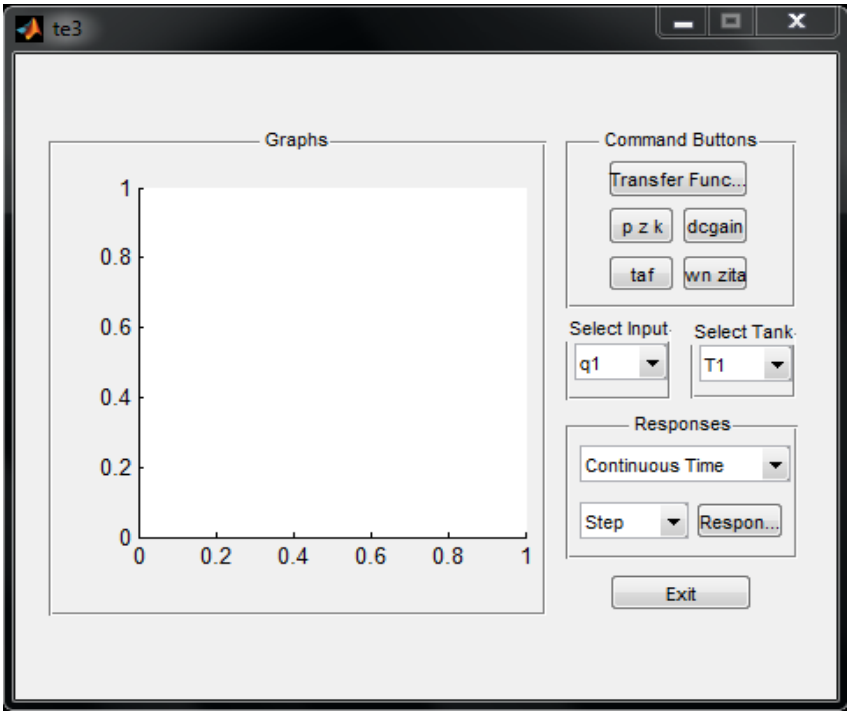
4.5.1 ΠΑΡΑΣΤΑΣΗ, ΠΡΟΣΟΜΟΙΩΣΗ ΚΑΙ ΑΝΑΛΥΣΗ ΔΥΝΑΜΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ ΣΤΟ MATLAB: ΠΕΔΙΟ LAPLACE ΚΑΙ ΖΗΤΑ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΣΟΜΟΙΩΣΗΣ

Σε αυτό το γραφικό περιβάλλον γίνεται μαθηματική αναπαράσταση του χώρου κατάστασης, προσομοίωση και ανάλυση του συστήματος τριών δεξαμενών στα πεδία Laplace (συνεχής χρόνος) και Ζήτα (διακριτός χρόνος). Υπολογίζονται οι συναρτήσεις μεταφοράς [G_{11},G_{21},G_{31}] της πρώτης, της δεύτερης και της τρίτης εξόδου (h_1, h_2, h_3), ως προς την πρώτη παροχή (q_1), χρησιμοποιώντας τους πίνακες αναπαράστασης του χώρου κατάστασης.

Επιλέγεται λοιπόν, ο συνεχής χρόνος για την πρώτη παροχή (q_1) και στη συνέχεια το κουμπί Transfer Function. Αυτόματα εμφανίζεται στο Command Window του Matlab η συνάρτηση μεταφοράς του συστήματος. Ομοίως, και για τη δεύτερη παροχή (q_2). Ακολουθούν τα αποτελέσματα αυτών των επιλογών:

Η ίδια ακριβώς διαδικασία ακολουθείται και για το πεδίο Ζήτα (διακριτός χρόνος), αφού βέβαια επιλεγεί η περίοδος δειγματοληψίας (T_S).

Στη συνέχεια υπολογίζονται οι πόλοι, οι μηδενιστές, οι συντελεστές κέρδους και τα στατικά κέρδη στο πεδίο Ζήτα (διακριτός χρόνος), ως προς την πρώτη είσοδο επιλέγοντας το κουμπί zpk, πρώτα για την πρώτη και μετά για τη δεύτερη είσοδο (q_1 και q_2). Παρουσιάζεται η γραφική διεπαφή χρήστη προσομοίωσης του συστήματος (Σχήμα 4.23).



Σχήμα 4.23 Γραφική διεπαφή χρήστη προσομοίωσης συστήματος τριών δεξαμενών

Για να εμφανιστεί η συνάρτηση μεταφοράς του συστήματος επιλέγεται το κουμπί Transfer function και εμφανίζεται στην οθόνη τα παρακάτω αποτελέσματα:

```
ans =


q1
 num(1)/den =
       0.065 s^2 + 0.01001 s + 0.00028652
   ------------------------------------------
   s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005


num(2)/den =
    -2.7756e-017 s^2 + 0.002535 s + 0.00019519
   ------------------------------------------
   s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005


num(3)/den =
    -2.7756e-017 s^2 - 1.7347e-018 s + 9.8865e-005
   ------------------------------------------
    s^3 + 0.193 s^2 + 0.008893 s + 5.4795e-005
```

Για τον υπολογισμό των πόλων, μηδενιστών και στατικών κερδών για την $1^{η}$ παροχή επιλέγεται από το Select input το q_1 και στη συνέχεια το κουμπί “p z k”. Αντίστοιχα και για τη $2^{η}$ παροχή. Εμφανίζονται τα ακόλουθα αποτελέσματα:

```
ans =
q1
z1 =
  1.0e+013 *


   -0.0000    9.1333   -0.0000
   -0.0000   -0.0000    0.0000
p1 =
```

```
   -0.1257
   -0.0600
   -0.0073
k1 =
    0.0650
   -0.0000
   -0.0000

ans =
q2
z2 =
  1.0e+013 *
   -0.0000   -1.5222   -0.0000
    0.0000   -0.0000   -0.0000
p2 =
   -0.1257
   -0.0600
   -0.0073
k2 =
   -0.0000
    0.0000
    0.0650
```

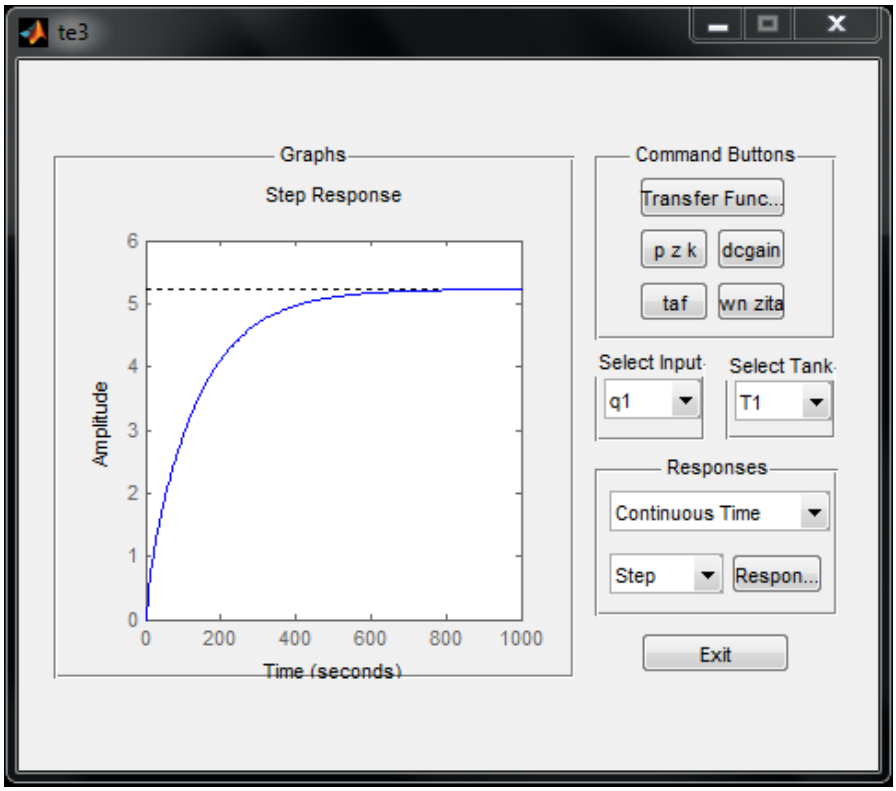
Επίσης επιλέγοντας taf εμφανίζονται οι χρονικές σταθερές του συστήματος.

```
taf =
    7.9529
   16.6679
  137.6751
```

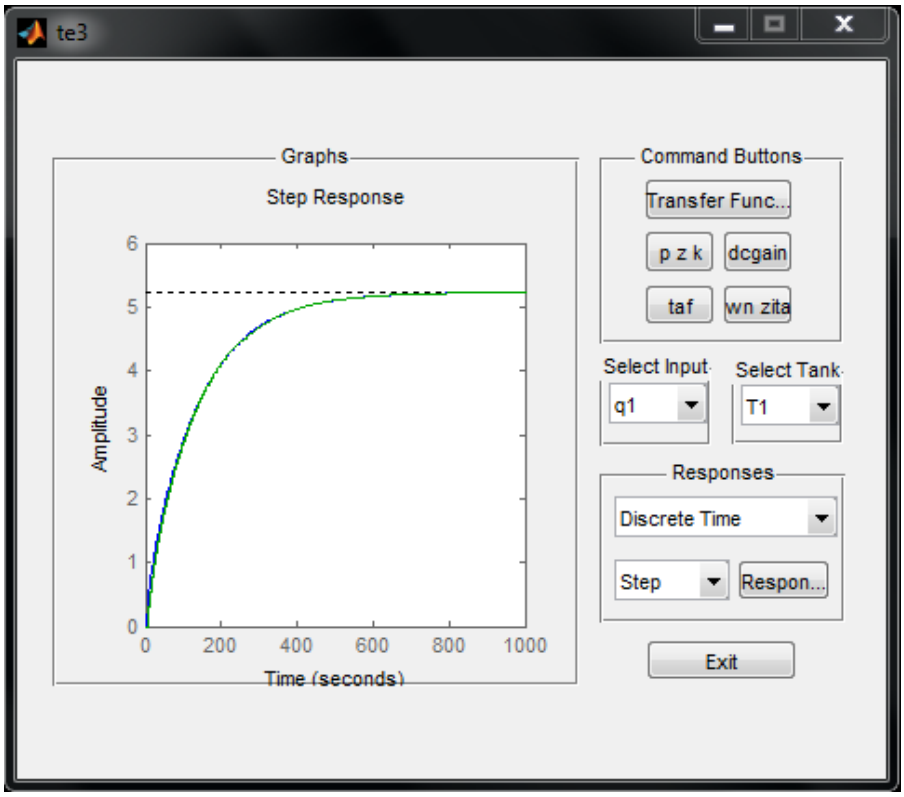
Τα στατικά κέρδη των συναρτήσεων μεταφοράς, υπολογίζονται από την επιλογή dcgain και έχουν την ίδια τιμή και για τις δύο εισόδους (q_1 και q_2).

```
ans =


   5.2289    1.8043
   3.5623    1.8043
   1.8043    1.7580
```

Επιλέγοντας την απόκριση step συνεχή και στη συνέχεια για διακριτό χρόνο εμφανίζεται η απόκριση του συστήματος σε είσοδο μοναδιαία βαθμίδα για την $1^{η}$ παροχή και στον εικονικό κόσμο το ύψος της στάθμης της $1^{ης}$ δεξαμενής, όπως φαίνεται στα Σχήματα 4.24 και 4.25). Με τον ίδιο τρόπο μπορεί να επιλεγεί και η $2^{η}$ παροχή ως προς τις άλλες δύο δεξαμενές από το 'Select Tank'.

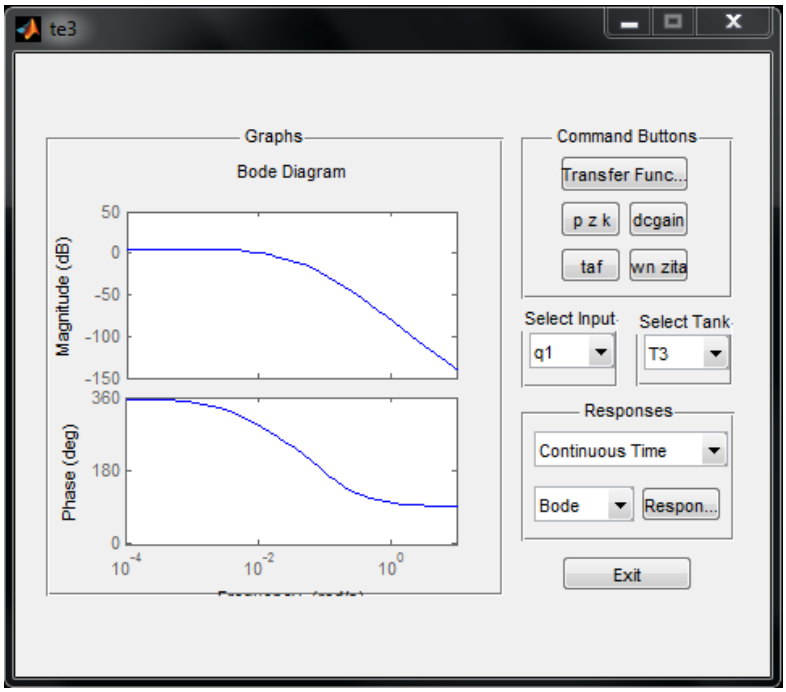


Σχήμα 4.24 Απόκριση σε είσοδο μοναδιαία βαθμίδα (συνεχής χρόνος)

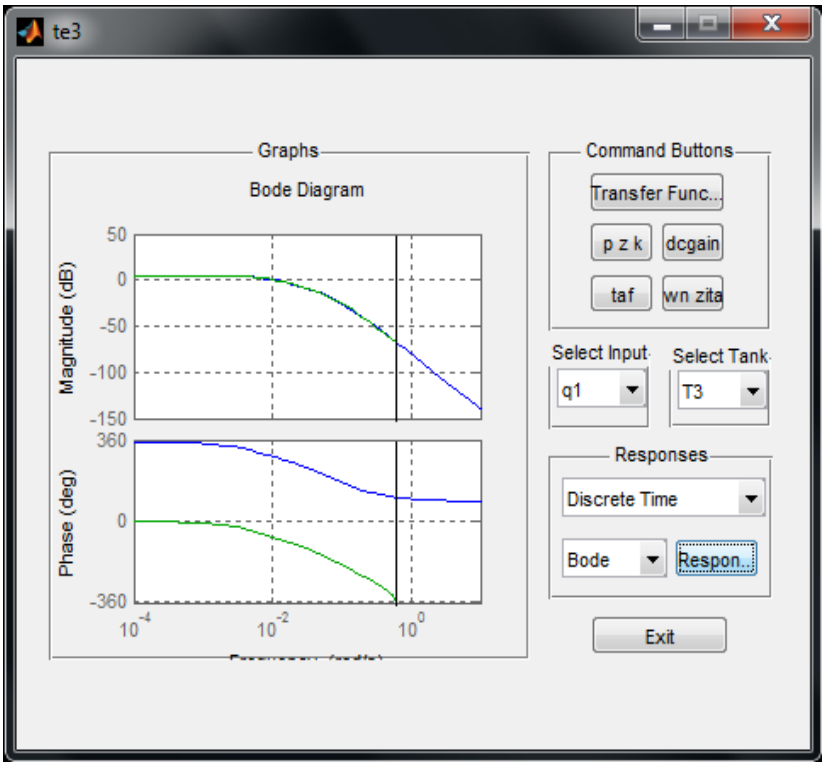


Σχήμα 4.25 Απόκριση σε είσοδο μοναδιαία βαθμίδα (διακριτός χρόνος)

Τέλος, εμφανίζονται οι αποκρίσεις συχνότητας του συστήματος αρχικά στο συνεχή και έπειτα στο διακριτό χρόνο για την δεύτερη δεξαμενή (T_3) (Σχήμα 4.26 και 4.27 αντίστοιχα).



Σχήμα 4.26 Απόκριση συχνότητας συνεχούς χρόνου



Σχήμα 4.27 Απόκριση συχνότητας διακριτού χρόνου

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Remote and Virtual Control Lab at DEI-FCTUC: Systems and Control Experiments for Engineering Courses,

(url:http://www.sefi.be/wp-content/papers2011/T12/169.pdf)

[2] PendCon Basic: Level Control for a Three-Tank-System,

(url:http://www.pendcon.de/level-control-for-a-three-tank-system.html)

[3] Συστήματα Ελέγχου δεξαμενών: Τεχνικά χαρακτηριστικά συστήματος,

(url:http://www.diodebell.com/el/products-el/tank-level-controller-and-alarm-el)

[4] Σύστημα παρακολούθησης δεξαμενών

(url: http://www.logicom.gr/html/page.asp?Lang=1&PageID=15)

[5] Virtual Reality Modelling Language (VRML),

(url:http://www.it.uom.gr/project/MultimediaTechnologyNotes/extra/append9_3.htm)

[6] Φοίβος-Απόστολος Μυλωνάς, "Σχεδιασμός & Υλοποίηση VRML Browser σε Java", Διπλωματική εργασία, Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών, 2003,

[7] Εισαγωγή στην Virtual Reality Modelling Language (VRML),

(url:http://alexandra.di.uoa.gr/mmtech/VirtualReality/eisagvgh.pdf)

[8] VRML, από Wikipedia,

(url:http://en.wikipedia.org/wiki/VRML)

[9] 3DMLW, από Wikipedia,

(url:http://en.wikipedia.org/wiki/3DMLW)

[10] O3D, από Wikipedia,

(url:http://en.wikipedia.org/wiki/O3D)

[11] COLLADA, από Wikipedia,

(url:http://en.wikipedia.org/wiki/COLLADA)

[12] Universal 3D, από Wikipedia,

(url:http://en.wikipedia.org/wiki/Universal_3D)

[13] X3D, από Wikipedia,

(url:http://en.wikipedia.org/wiki/X3D)

[14] Μαυραντζάς Νικόλαος, "Οι τεχνολογίες 3D στην τάξη και παραδείγματα ενσωμάτωσης στη διδασκαλία χρησιμοποιώντας την γλώσσα VRML", 4ο Συνέδριο στη Σύρο - ΤΠΕ στην Εκπαίδευση, 2007

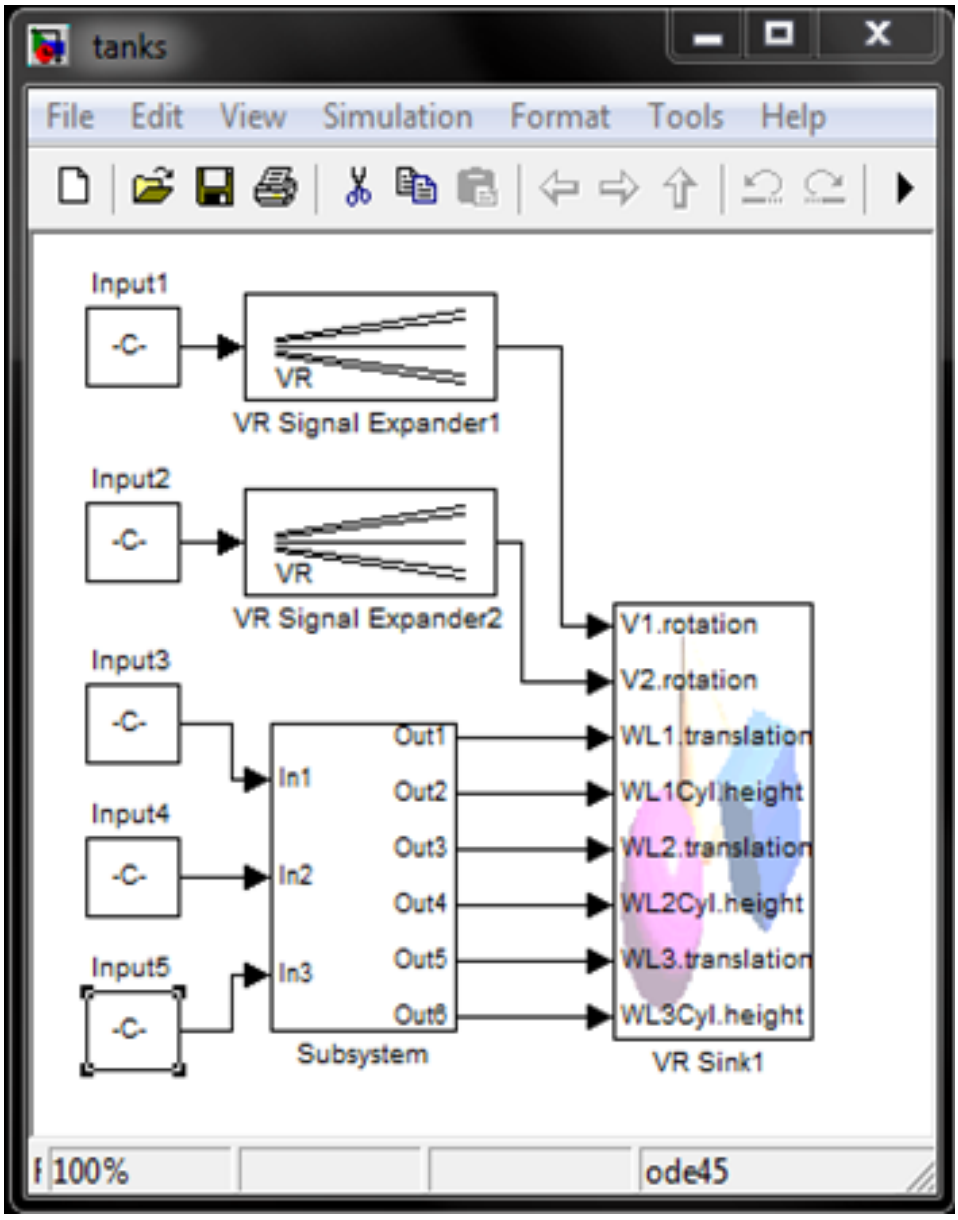
[15] Introduction to V·Realm Builder Concepts, από το μενού help του V-Realm Builder 2.0

[16] Installing the VRML Editor on the Host Computer: Installation (Virtual Reality Toolbox™), από το μενού help του λογισμικού MATLAB

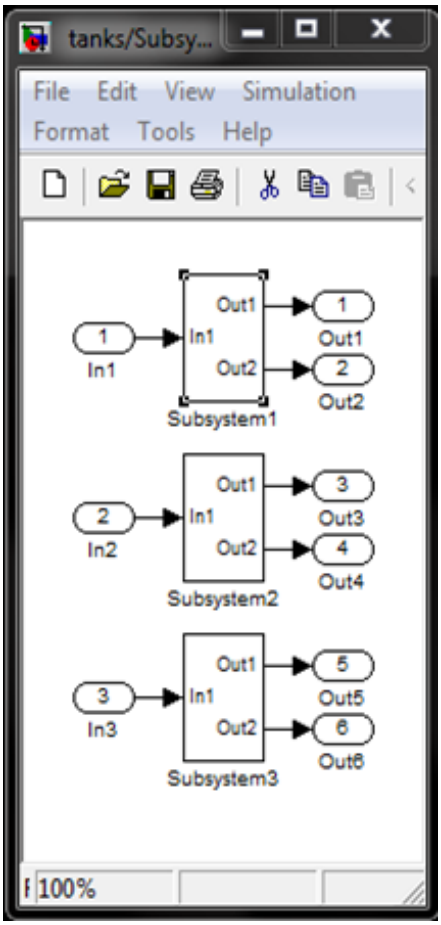
[17] Γεώργιος Ν. Φουσκιτάκης, Παράσταση και προσομοίωση δυναμικών συστημάτων στο Matlab: Πεδίο Χρόνου, Ψηφιακά Συστήματα Ελέγχου, Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης, Τμήμα Ηλεκτρονικής

ΠΑΡΑΡΤΗΜΑ

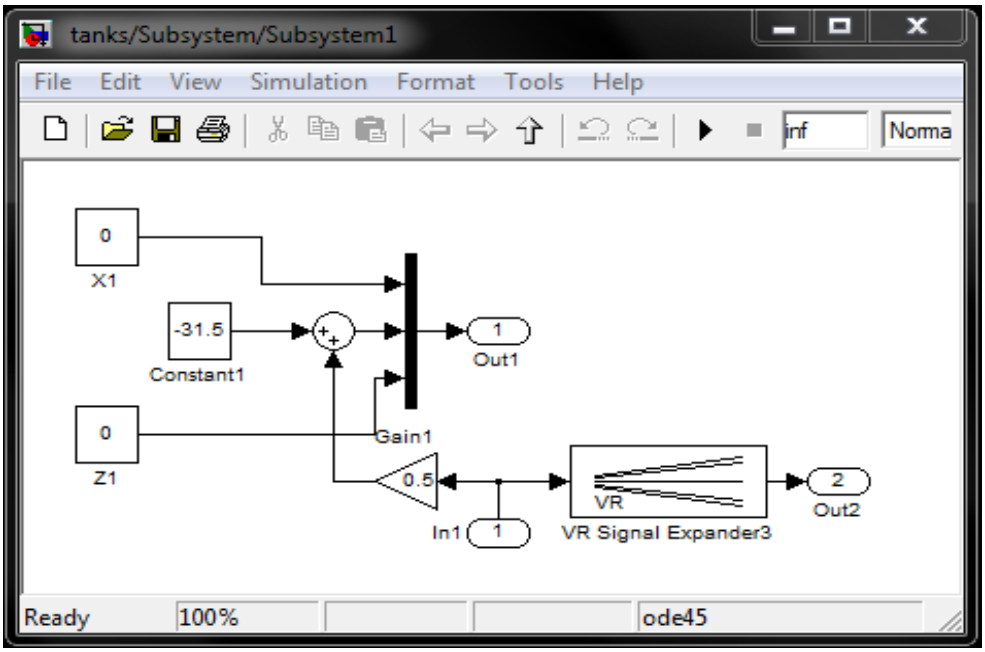
ΜΟΝΤΕΛΟ ΠΡΟΣΟΜΟΙΩΣΗΣ (SIMULINK MODEL)



Κεντρικό μοντέλο προσομοίωσης στο λογισμικό Matlab



Ανάλυση του κεντρικού υποσυστήματος του μοντέλου προσομοίωσης



Ανάλυση υποσυστήματος για τον έλεγχο της κάθε δεξαμενής

ΚΩΔΙΚΕΣ

te2.m

Το te2.m δημιουργεί τη γραφική διεπαφή για παράσταση και προσομοίωση δυναμικών συστημάτων στο πεδίο χρόνου.

```
function varargout = te2(varargin)
% TE2 MATLAB code for te2.fig
%      TE2, by itself, creates a new TE2 or raises the existing
%      singleton*.
%
%      H = TE2 returns the handle to a new TE2 or the handle to
%      the existing singleton*.
%
%      TE2('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in TE2.M with the given input arguments.
%
%      TE2('Property','Value',...) creates a new TE2 or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before te2_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to te2_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help te2

% Last Modified by GUIDE v2.5 03-Dec-2012 02:02:56

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @te2_OpeningFcn, ...
                   'gui_OutputFcn',  @te2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before te2 is made visible.
function te2_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to te2 (see VARARGIN)

% Choose default command line output for te2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes te2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = te2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu1


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu2


% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit1a_Callback(hObject, eventdata, handles)
% hObject    handle to edit1a (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1a as text
%        str2double(get(hObject,'String')) returns contents of edit1a
as a double


% --- Executes during object creation, after setting all properties.
function edit1a_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1a (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit1b_Callback(hObject, eventdata, handles)
% hObject    handle to edit1b (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1b as text
%        str2double(get(hObject,'String')) returns contents of edit1b
as a double


% --- Executes during object creation, after setting all properties.
function edit1b_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1b (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit1c_Callback(hObject, eventdata, handles)
% hObject    handle to edit1c (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1c as text
%        str2double(get(hObject,'String')) returns contents of edit1c
as a double


% --- Executes during object creation, after setting all properties.
function edit1c_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1c (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
```

```
%        str2double(get(hObject,'String')) returns contents of edit2 as
a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=[-0.039 0.039 0;0.039 -0.077 0.039;0 0.039 -0.077];
B=[0.065 0;0 0;0 0.065];
C=eye(3);
D=zeros(3,2);

h1=str2double(get(handles.edit1a,'String'));
h3=str2double(get(handles.edit1b,'String'));
h2=str2double(get(handles.edit1c,'String'));
h0=[h1 h3 h2]
t=str2double(get(handles.edit2,'String'))

graph=(get(handles.popupmenu1,'Value'));
q_input=(get(handles.popupmenu2,'Value'));

%Sample Sensitivity
S=10;

if graph==1
    'Step'
    if q_input==1
        'q1'
        step(A,B,C,D,1,t)
        [Y]=step(A,B,C,D,1,t);
        L=length(Y);
        n=1;
        for I=0:S:L
            temp1=Y(n,1);
            temp2=Y(n,2);
            temp3=Y(n,3);
            n=n+S;
```

```
            opts = simset('SrcWorkspace','current');
            value_h1=temp2*0.1;%tank2
            value_h2=temp3*0.1;%tank3
            value_h3=temp1*0.1;%tank1
            value_p1=100;%pipe1
            value_p2=100;%pipe2
            sim('tanks',1,opts);
        end
    else if q_input==2
            'q2'
            step(A,B,C,D,2,t)
            [Y]=step(A,B,C,D,1,t);
            L=length(Y);
            n=1;
            for I=0:S:L
                temp1=Y(n,1);
                temp2=Y(n,2);
                temp3=Y(n,3);
                n=n+S;
                opts = simset('SrcWorkspace','current');
                value_h1=temp2*0.1;%tank2
                value_h2=temp3*0.1;%tank3
                value_h3=temp1*0.1;%tank1
                value_p1=100;%pipe1
                value_p2=100;%pipe2
                sim('tanks',1,opts);
            end
        else if q_input==3
                'q1 & q2'
                step(A,B,C,D)
            end
        end
    end
end
if graph==2
        'Initial'
        initial(A,B,C,D,[h1 h2 h3],t)

        [Y]=initial(A,B,C,D,[h1 h2 h3],t);
        L=length(Y);
        n=1;
        for I=0:S:L
            temp1=Y(n,1);
            temp2=Y(n,2);
            temp3=Y(n,3);
            n=n+S;
            opts = simset('SrcWorkspace','current');
            value_h1=temp2;%tank2
            value_h2=temp3;%tank3
            value_h3=temp1;%tank1
            value_p1=100;%pipe1
            value_p2=100;%pipe2
            sim('tanks',1,opts);
        end
end
if graph==3
        'Impulse'
```

```
        if q_input==1
            'q1'
            impulse(A,B,C,D,1,t)
            [Y]=impulse(A,B,C,D,1,t);
            L=length(Y);
            n=1;
            for I=1:S:L
                temp1=Y(n,1);
                temp2=Y(n,2);
                temp3=Y(n,3);
                n=n+S;
                opts = simset('SrcWorkspace','current');
                value_h1=temp2*10;%tank2
                value_h2=temp3*10;%tank3
                value_h3=temp1*10;%tank1
                value_p1=100;%pipe1
                value_p2=100;%pipe2
                sim('tanks',1,opts);
            end
        else if q_input==2
                'q2'
                impulse(A,B,C,D,2,t)
                [Y]=impulse(A,B,C,D,2,t);
                L=length(Y);
                n=1;
                for I=1:S:L
                    temp1=Y(n,1);
                    temp2=Y(n,2);
                    temp3=Y(n,3);
                    n=n+S;
                    opts = simset('SrcWorkspace','current');
                    value_h1=temp2*10;%tank2
                    value_h2=temp3*10;%tank3
                    value_h3=temp1*10;%tank1
                    value_p1=100;%pipe1
                    value_p2=100;%pipe2
                    sim('tanks',1,opts);
                end
            else if q_input==3
                    'q1 & q2'
                    impulse(A,B,C,D)
                end
            end
        end
end

% --- Executes on button press in pushbutton1.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

delete(te2)
```

te3.m

Το te3.m δημιουργεί τη γραφική διεπαφή για παράσταση, προσομοίωση και ανάλυση δυναμικών συστημάτων στα πεδία Laplace και Ζήτα.

```
function varargout = te3(varargin)
% TE3 MATLAB code for te3.fig
%      TE3, by itself, creates a new TE3 or raises the existing
%      singleton*.
%
%      H = TE3 returns the handle to a new TE3 or the handle to
%      the existing singleton*.
%
%      TE3('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in TE3.M with the given input arguments.
%
%      TE3('Property','Value',...) creates a new TE3 or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before te3_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to te3_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help te3

% Last Modified by GUIDE v2.5 05-Dec-2012 18:12:58

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @te3_OpeningFcn, ...
                   'gui_OutputFcn',  @te3_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before te3 is made visible.
function te3_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to te3 (see VARARGIN)

% Choose default command line output for te3
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes te3 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = te3_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on selection change in popupmenu2.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu2


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu2


% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu3


% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on selection change in popupmenu4.
function popupmenu4_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu4
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu4


% --- Executes during object creation, after setting all properties.
function popupmenu4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=[-0.039 0.039 0;0.039 -0.077 0.039;0 0.039 -0.077];
B=[0.065 0;0 0;0 0.065];
C=eye(3);
D=zeros(3,2);

q_input=(get(handles.popupmenu1,'Value'));

if q_input==1
    'q1'
    [num1,den1]=ss2tf(A,B,C,D,1);
    printsys(num1,den1,'s')
else if q_input==2
        'q2'
        [num2,den2]=ss2tf(A,B,C,D,2)
        printsys(num2,den2,'s')
    else
        'Error!'
    end
end
```

```
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=[-0.039 0.039 0;0.039 -0.077 0.039;0 0.039 -0.077];
B=[0.065 0;0 0;0 0.065];
C=eye(3);
D=zeros(3,2);

q_input=(get(handles.popupmenu1,'Value'));

if q_input==1
    'q1'
    [num1,den1]=ss2tf(A,B,C,D,1);
    [z1,p1,k1]=tf2zp(num1,den1)
else if q_input==2
        'q2'
        [num2,den2]=ss2tf(A,B,C,D,2);
        [z2,p2,k2]=tf2zp(num2,den2)
    else
        'Error!'
    end
end


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=[-0.039  0.039  0; 0.039  -0.077  0.039; 0  0.039  -0.077];
B=[0.065  0; 0  0; 0  0.065];
C=eye(3);
D=zeros(3,2);
dcgain(A,B,C,D)


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=[-0.039  0.039  0; 0.039  -0.077  0.039; 0  0.039  -0.077];
B=[0.065  0; 0  0; 0  0.065];
C=eye(3);
D=zeros(3,2);

q_input=(get(handles.popupmenu1,'Value'));
```

```
if q_input==1
    [num1,den1]=ss2tf(A,B,C,D,1);
    [z1,p1,k1]=tf2zp(num1,den1);
    taf=1./abs(real(p1))
else if q_input==2
        [num2,den2]=ss2tf(A,B,C,D,2);
        [z2,p2,k2]=tf2zp(num2,den2);
        taf=1./abs(real(p2))
    end
end


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=[-0.039 0.039 0;0.039 -0.077 0.039;0 0.039 -0.077];
B=[0.065 0;0 0;0 0.065];
C=eye(3);
D=zeros(3,2);

sys_T1_q1=tf([0.065 0.01001 0.00028652],[1 0.193 0.008893 5.4795e-05]);
sys_T2_q1=tf([-2.7756e-17 0.002535 0.00019519],[1 0.193 0.008893
5.4795e-05]);
sys_T3_q1=tf([-2.7756e-17 -1.7347e-18 9.8865e-05],[1 0.193 0.008893
5.4795e-05]);

sys_T1_q2=tf([-2.7756e-17 -1.7347e-18 9.8865e-05],[1 0.193 0.008893
5.4795e-05]);
sys_T2_q2=tf([1.6653e-16 0.002535 9.8865e-05],[1 0.193 0.008893
5.4795e-05]);
sys_T3_q2=tf([0.065 0.00754 9.633e-05],[1 0.193 0.008893 5.4795e-05]);

q_input=(get(handles.popupmenu1,'Value'));
T_input=(get(handles.popupmenu4,'Value'));

if q_input==1
    'q1'
    if T_input==1
        'T1'
        sys=sys_T1_q1
    else if T_input==2
            'T2'
            sys=sys_T2_q1
        else if T_input==3
                'T3'
                sys=sys_T3_q1
            end
        end
    end
else if q_input==2
        'q2'
```

```
        if T_input==1
            'T1'
            sys=sys_T1_q2
        else if T_input==2
                'T2'
                sys=sys_T2_q2
            else if T_input==3
                    'T3'
                    sys=sys_T3_q2
                end
            end
        end
    end
end

[wn,zita]=damp(sys)


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=[-0.039 0.039 0;0.039 -0.077 0.039;0 0.039 -0.077];
B=[0.065 0;0 0;0 0.065];
C=eye(3);
D=zeros(3,2);

sys_T1_q1=tf([0.065 0.01001 0.00028652],[1 0.193 0.008893 5.4795e-05]);
sys_T2_q1=tf([-2.7756e-17 0.002535 0.00019519],[1 0.193 0.008893
5.4795e-05]);
sys_T3_q1=tf([-2.7756e-17 -1.7347e-18 9.8865e-05],[1 0.193 0.008893
5.4795e-05]);

sys_T1_q2=tf([-2.7756e-17 -1.7347e-18 9.8865e-05],[1 0.193 0.008893
5.4795e-05]);
sys_T2_q2=tf([1.6653e-16 0.002535 9.8865e-05],[1 0.193 0.008893
5.4795e-05]);
sys_T3_q2=tf([0.065 0.00754 9.633e-05],[1 0.193 0.008893 5.4795e-05]);

q_input=(get(handles.popupmenu1,'Value'));
time=(get(handles.popupmenu2,'Value'));
graph=(get(handles.popupmenu3,'Value'));
T_input=(get(handles.popupmenu4,'Value'));

%Sample Sensitivity
S=5;

if graph==1
    'Step'
    if time==1
        'Continuous'
        g1=subplot(1,1,1,'Parent',handles.uipanel1); plot(g1,[00]);
%Axes Reset
```

```
if q_input==1
    if T_input==1
        'T1'
        sys=sys_T1_q1
        step(sys),hold

        [Y]=step(sys);
        L=length(Y);
        n=1;
        for I=0:S:L
            temp1=Y(n,1);
            n=n+S;
            opts = simset('SrcWorkspace','current');
            value_h1=0; %tank2
            value_h2=0; %tank3
            value_h3=temp1*0.1; %tank1
            value_p1=100;%pipe1
            value_p2=100;%pipe2
            sim('tanks',1,opts);
        end
    else if T_input==2
            'T2'
            sys=sys_T2_q1
            step(sys),hold

            [Y]=step(sys);
            L=length(Y);
            n=1;
            for I=1:S:L
                temp2=Y(n,1);
                n=n+S;
                opts = simset('SrcWorkspace','current');
                value_h1=temp2*0.1; %tank2
                value_h2=0; %tank3
                value_h3=0; %tank1
                value_p1=100;%pipe1
                value_p2=100;%pipe2
                sim('tanks',1,opts);
            end
        else if T_input==3
                'T3'
                sys=sys_T3_q1
                step(sys),hold

                [Y]=step(sys);
                L=length(Y);
                n=1;
                for I=0:S:L
                    temp3=Y(n,1);
                    n=n+S;
                    opts = simset('SrcWorkspace','current');
                    value_h1=0; %tank2
                    value_h2=temp3*0.1; %tank3
                    value_h3=0; %tank1
                    value_p1=100;%pipe1
                    value_p2=100;%pipe2
```

```
                    sim('tanks',1,opts);
                end
            end
        end
    end
else if q_input==2
        'q2'
        if T_input==1
            'T1'
            sys=sys_T1_q2
            step(sys),hold

            [Y]=step(sys);
            L=length(Y);
            n=1;
            for I=0:S:L
                temp1=Y(n,1);
                n=n+S;
                opts = simset('SrcWorkspace','current');
                value_h1=0; %tank2
                value_h2=0; %tank3
                value_h3=temp1*0.1; %tank1
                value_p1=100;%pipe1
                value_p2=100;%pipe2
                sim('tanks',1,opts);
            end
    else if T_input==2
            'T2'
            sys=sys_T2_q2
            step(sys),hold

            [Y]=step(sys);
            L=length(Y);
            n=1;
            for I=0:S:L
                temp2=Y(n,1);
                n=n+S;
                opts = simset('SrcWorkspace','current');
                value_h1=temp2*0.1; %tank2
                value_h2=0; %tank3
                value_h3=0; %tank1
                value_p1=100;%pipe1
                value_p2=100;%pipe2
                sim('tanks',1,opts);
            end
        else if T_input==3
                'T3'
                sys=sys_T3_q2
                step(sys),hold

                [Y]=step(sys);
                L=length(Y);
                n=1;
                for I=0:S:L
                    temp3=Y(n,1);
                    n=n+S;
```

```
                        opts = simset('SrcWorkspace','current');
                        value_h1=0; %tank2
                        value_h2=temp3*0.1; %tank3
                        value_h3=0; %tank1
                        value_p1=100;%pipe1
                        value_p2=100;%pipe2
                        sim('tanks',1,opts);
                    end
                end
            end
            end
        end
    end
else if time==2
        'Discrete'
        if q_input==1
            'q1'
            [num1,den1]=ss2tf(A,B,C,D,1);
            [z1,p1,k1]=tf2zp(num1,den1);
            taf=1./abs(real(p1));
            if T_input==1
                'T1'
                sys=sys_T1_q1

                fsmax=max(abs(p1))/2/pi;
                Ts=0.2*(1/(2*fsmax));
                sysd=c2d(sys,Ts);
                step(sysd)

                [Y]=step(sysd);
                L=length(Y);
                n=1;
                for I=0:S:L
                    temp1=Y(n,1);
                    n=n+S;
                    opts = simset('SrcWorkspace','current');
                    value_h1=0; %tank2
                    value_h2=0; %tank3
                    value_h3=temp1*0.1; %tank1
                    value_p1=100;%pipe1
                    value_p2=100;%pipe2
                    sim('tanks',1,opts);
                end
            else if T_input==2
                    'T2'
                    sys=sys_T2_q1
                    fsmax=max(abs(p1))/2/pi;
                    Ts=0.2*(1/(2*fsmax));
                    sysd=c2d(sys,Ts);
                    step(sysd)

                    [Y]=step(sysd);
                    L=length(Y);
                    n=1;
                    for I=1:S:L
                        temp2=Y(n,1);
```

```
                            n=n+S;
                            opts = simset('SrcWorkspace','current');
                            value_h1=temp2*0.1; %tank2
                            value_h2=0; %tank3
                            value_h3=0; %tank1
                            value_p1=100;%pipe1
                            value_p2=100;%pipe2
                            sim('tanks',1,opts);
                        end
                    else if T_input==3
                            'T3'
                            sys=sys_T3_q1
                            fsmax=max(abs(p1))/2/pi;
                            Ts=0.2*(1/(2*fsmax));
                            sysd=c2d(sys,Ts);
                            step(sysd)

                            [Y]=step(sysd);
                            L=length(Y);
                            n=1;
                            for I=1:S:L
                                temp3=Y(n,1);
                                n=n+S;
                                opts =
simset('SrcWorkspace','current');
                                value_h1=0; %tank2
                                value_h2=temp3*0.1; %tank3
                                value_h3=0; %tank1
                                value_p1=100;%pipe1
                                value_p2=100;%pipe2
                                sim('tanks',1,opts);
                            end
                        end
                    end
                end
            else if q_input==2
                    'q2'
                    [num2,den2]=ss2tf(A,B,C,D,2);
                    [z2,p2,k2]=tf2zp(num2,den2);
                    taf=1./abs(real(p2));
                    if T_input==1
                        'T1'
                        sys=sys_T1_q2

                        fsmax=max(abs(p2))/2/pi;
                        Ts=0.2*(1/(2*fsmax));
                        sysd=c2d(sys,Ts);
                        step(sysd)

                        [Y]=step(sysd);
                        L=length(Y);
                        n=1;
                        for I=1:S:L
                            temp1=Y(n,1);
                            n=n+S;
                            opts = simset('SrcWorkspace','current');
```

```
                            value_h1=0; %tank2
                            value_h2=0; %tank3
                            value_h3=temp1*0.1; %tank1
                            value_p1=100;%pipe1
                            value_p2=100;%pipe2
                            sim('tanks',1,opts);
                        end
                    else if T_input==2
                            'T2'
                            sys=sys_T2_q2
                            fsmax=max(abs(p2))/2/pi;
                            Ts=0.2*(1/(2*fsmax));
                            sysd=c2d(sys,Ts);
                            step(sysd)

                            [Y]=step(sysd);
                            L=length(Y);
                            n=1;
                            for I=1:S:L
                                temp2=Y(n,1);
                                n=n+S;
                                opts =
simset('SrcWorkspace','current');
                                value_h1=temp2*0.1; %tank2
                                value_h2=0; %tank3
                                value_h3=0; %tank1
                                value_p1=100;%pipe1
                                value_p2=100;%pipe2
                                sim('tanks',1,opts);
                            end
                        else if T_input==3
                                'T3'
                                sys=sys_T3_q2
                                fsmax=max(abs(p2))/2/pi;
                                Ts=0.2*(1/(2*fsmax));
                                sysd=c2d(sys,Ts);
                                step(sysd)

                                [Y]=step(sysd);
                                L=length(Y);
                                n=1;
                                for I=1:S:L
                                    temp3=Y(n,1);
                                    n=n+S;
                                    opts =
simset('SrcWorkspace','current');
                                    value_h1=0; %tank2
                                    value_h2=temp3*0.1; %tank3
                                    value_h3=0; %tank1
                                    value_p1=100;%pipe1
                                    value_p2=100;%pipe2
                                    sim('tanks',1,opts);
                                end
                            end
                        end
```

```
                end
            end
        end
    end
else if graph==2
        'Bode'
        if time==1
            'Continuous'
            g1=subplot(1,1,1,'Parent',handles.uipanel1); plot(g1,[00]);
%Axes Reset
            if q_input==1
                'q1'
                if T_input==1
                    'T1'
                    sys=sys_T1_q1
                else if T_input==2
                        'T2'
                        sys=sys_T2_q1
                    else if T_input==3
                            'T3'
                            sys=sys_T3_q1
                        end
                    end
                end
            else if q_input==2
                    'q2'
                    if T_input==1
                        'T1'
                        sys=sys_T1_q2
                    else if T_input==2
                            'T2'
                            sys=sys_T2_q2
                        else if T_input==3
                                'T3'
                                sys=sys_T3_q2
                            end
                        end
                    end
                end
            end
            bode(sys),hold
        else if time==2
                'Discrete'
                if q_input==1
                    'q1'
                    [num1,den1]=ss2tf(A,B,C,D,1);
                    [z1,p1,k1]=tf2zp(num1,den1);
                    taf=1./abs(real(p1));
                    if T_input==1
                        'T1'
                        sys=sys_T1_q1
                    else if T_input==2
                            'T2'
                            sys=sys_T2_q1
                        else if T_input==3
                                'T3'
                                sys=sys_T3_q1
```

```
                        end
                    end
                end
                fsmax=max(abs(p1))/2/pi;
                Ts=0.2*(1/(2*fsmax));
                sysd=c2d(sys,Ts);
                bode(sysd),grid
            else if q_input==2
                    'q2'
                    [num2,den2]=ss2tf(A,B,C,D,2);
                    [z2,p2,k2]=tf2zp(num2,den2);
                    taf=1./abs(real(p2));
                    if T_input==1
                        'T1'
                        sys=sys_T1_q2
                    else if T_input==2
                            'T2'
                            sys=sys_T2_q2
                        else if T_input==3
                                'T3'
                                sys=sys_T3_q2
                            end
                        end
                    end
                    fsmax=max(abs(p2))/2/pi;
                    Ts=0.2*(1/(2*fsmax));
                    sysd=c2d(sys,Ts);
                    bode(sysd),grid
                end
            end
        end
    end
else
    'Error!'
end
end


% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

delete(te3)
```