Thesis Title : 3D Schema Matching Tool

Author Name : George Tsamis

Thesis submitted in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science

University of Crete

School of Sciences and Engineering

Computer Science Department

Knossou Av., P.O. Box 2208, Heraklion, GR-71409, Greece

Thesis Advisors: Prof *First,* Dr. *Second*

Yannis Tzitzikas: Assistant Professor of Computer Science, University of Crete

Martin Doerr : Research Director at the Information Systems Laboratory (Foundation for Research and Technology - Hellas (FORTH))

This work has been performed at the fill in organization of Computer Science Department (University of Crete).

The work is partially supported by Foundation for Research & Technology – Hellas.

UNIVERSITY OF CRETE

COMPUTER SCIENCE DEPARTMENT

3D Schema Matching Tool

Thesis submitted by

George Tsamis

in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science

THESIS APPROVAL

Author : ……
Author Name

Committee approvals : ……
First member
Dr. Thesis Supervisor

……
Second member
Associate Professor Committee Member

……
Third member
Professor Committee Member

Departmental approval : ……
CSD Chair
Professor, Director of Graduate Studies

Heraklion, April 2012

Ευχαριστίες

Στο σημείο αυτό θα ήθελα να ευχαριστήσω θερμά τον επόπτη Καθηγητή μου, Κύριο Martin Doerr, για την άψογη συνεργασία, τον χρόνο που μου αφιέρωσε, την πολύτιμη καθοδήγησή του και που πίστεψε σε μένα. Τον ευχαριστώ θερμά για όλα όσα αποκόμισα κοντά του και είμαι σίγουρος ότι θα αποτελέσουν τις βάσεις για την μετέπειτα καριέρα μου στην επιστήμη της πληροφορικής.

Επίσης, θα ήθελα να ευχαριστήσω θερμά τον σύμβουλο – επόπτη Καθηγητή μου, Κύριο Ιωάννη Τζίτζικα για την καθοδήγηση, την συμβολή του και το ενδιαφέρον του στις μεταπτυχιακές μου σπουδές.

Φυσικά, θα ήθελα να ευχαριστήσω τρεις ανθρώπους που έχουν δώσει νόημα στην ζωή μου και είναι πάντα ακούραστοι υποστηρικτές σε οποιοδήποτε βήμα κάνω στην καριέρα μου. Αυτοί είναι ο πατέρας μου Βλάσης Τσάμης, η μητέρα μου Άννα Τσάμη και η αδελφή μου Χρύσα Τσάμη. Δεν θα μπορούσα να φανταστώ ποτέ καλύτερη οικογένεια από αυτήν που ήδη διαθέτω και γι' αυτό θα τους είμαι για πάντα ευγνώμων.

Θα ήθελα να ευχαριστήσω προσωπικά την δεσποινίς Μαρία Ρουσάκη για την αμέριστη συμπαράσταση, την υπομονή της, την κατανόηση και πολύ περισσότερο την συναισθηματική της υποστήριξη στις εύκολες και δύσκολες στιγμές που πέρασα κατά την διάρκεια των σπουδών μου.

Θα ήθελα να ευχαριστήσω πάρα πολύ τον καλύτερο φίλο που είχα ποτέ Χάρη Κουρδουνάκη για τις ξέγνοιαστες στιγμές που έχουμε περάσει μαζί, την ποιοτική παρέα του, την υποστήριξή του καθώς και τις συμβουλές του στην διάρκεια των σπουδών μου.

Τέλος, θα ήθελα να ευχαριστήσω τον Κύριο Δρυλλεράκη Ιωάννη για την πολύτιμη υποστήριξη του κατά την διάρκεια των προπτυχιακών και μεταπτυχιακών μου σπουδών. Χάρη σε αυτόν θυμήθηκα ότι ακόμα και στις μέρες μας υπάρχουν σωστοί, έντιμοι, και τίμιοι άνθρωποι που θα έκαναν τα πάντα για να βοηθήσουν.

Ευχαριστίες 3

Πίνακας Περιεχομένων 4

Λίστα Εικόνων 9

Λίστα Αναφορών Κώδικα 14

Λίστα Πινάκων 15

Λίστα Διαγραμμάτων 16

1 Περίληψη – Abstract 17

1.1 Περίληψη 17

1.2 Abstract 17

2 Εισαγωγή 18

3 Τεχνικές Εξαγωγής 3D 19

3.1 Τεχνολογία Κινηματογράφου 19

3.2 Εξαγωγή Τρισδιάστατων Γραφικών Υπολογιστών 21

4 Γιατί 3D? 21

4.1 Προτερήματα 21

4.2 Μειονεκτήματα 23

5 Οπτικοποίηση Δεδομένων 24

6 Αντιστοίχηση 28

7 RDF 29

7.1 RDFS 30

8 Στόχοι 30

8.1 Αντιστοίχηση 31

8.2 Ανθρώπινη Αντίληψη 32

8.3 Αντικείμενα 33

8.4 Σχεδίαση 33

8.5 Λειτουργίες Μεταφοράς στο Χώρο 33

8.6 Διαχείριση Δεδομένων 34

8.7 Αυτόματη και μη Κίνηση 35

8.8 Θέση Αντικειμένων 36

8.9 Μέγεθος Αντικειμένων 36
8.10 Πλήθος Αντικειμένων 37
8.11 Αυτόματο Layout 38
8.11.1 Ορθογώνιο (Orthogonal Layouts) 39
8.11.2 Ιεραρχικό (Hierarchical Layouts) 40
8.11.3 Συμμετρικό (Symmetric Layout) 40
8.11.4 Κυκλικό (Circular Layout) 41
8.11.5 Δενδρικό (Tree Layout) 41
8.11.6 Αξιολόγηση Αυτόματου Layout 42
8.12 Σημερινές Τεχνολογίες 43
8.13 Διεπαφή 2D 44
8.14 3D Σχήματα 44
8.15 3D Επικάλυψη 45
9 Θέματα προς Έλεγχο 46
10 Τεχνικές Έμφασης 47
11 3D Τεχνικές Οπτοκοποίησης Πληροφορίας 48
11.1 Στιλ σχεδίασης οντοτήτων-συνδέσεων (Node-Link Style Designs) 48
11.2 Επιφανειακή σχεδίαση (Surface Designs) 49
11.3 Τοπία δεδομένων (Information Landscapes) 50
12 Previous – Related works (Προγενέστερες-Σχετιζόμενες Εργασίες) 51
12.1 SemNet 52
12.2 Cone Tree 52
12.3 Hyperbolic Space 53
12.4 SeeNet3D 53
12.5 Perspective Wall 53
12.6 Document Lens 54
12.7 3DPS 54
12.8 FSN (file system visualizer) 55
12.9 SDM (Selective Dynamic Manipulation) 56
12.10 Bead 57

12.11 Web Forager 57
12.12 Information Cube 57
12.13 N-Vision 59
12.14 Αξιολόγηση 59
13 3D Τεχνικές ανίχνευσης επικάλυψης 60
13.1 Κατάτμηση Χώρου (Space Partitioning) 61
13.2 Περικλειόμενοι Κύβοι (Bounding Boxes) 62
13.3 Περικλειήζουσες Σφαίρες (Bounding Spheres) 63
13.4 Τεχνάσματα Ανίχνευσης Επικάλυψης 64
14 Συμπεράσματα 3D Οπτικοίησης 66
15 Υλοποίηση στην Java3D 67
15.1 Java3D API 67
15.2 Virtual Universe 67
15.3 Java3D Προγράμματα 70
15.4 Αντικείμενο SimpleUniverse 70
15.5 Live Objects 71
15.6 Compiled Objects 72
15.7 Εγκατάσταση Java3D 73
15.8 DirectX vs OpenGL 73
15.9 Πρώτο Πρόγραμμα Java3D 74
15.10 Vector3f Class 76
15.11 Transform3D Class 76
15.11.1 Συνδυασμός Transformations 76
15.12 BoundingSphere Class 77
15.13 Shape3D Class 77
15.14 Geometry Classes 78
15.14.1 Box Class 78
15.15 GeometryArray Class 78
15.16 Appearance Class 81
15.17 Text2D Class 81

15.18 Σύστημα συντεταγμένων 82
15.19 User Data 82
15.20 Αλληλεπίδραση 83
15.20.1 Behavior Class 83
15.20.2 MouseBehavior Classes 86
15.20.3 MouseRotate Class 87
15.20.4 MouseTranslate Class 88
15.20.5 MouseZoom Class 88
15.21 Επιλογή Αντικειμένων 88
15.22 Φωτισμός 90
15.23 Shading 93
15.24 Material 95
15.25 Σκίαση 95
15.26 Textures 97
16 Διαγράμματα ροής εργασιών 100
17 Εφαρμογή 107
17.1 Layout 108
17.2 Επίπεδα Παραθύρων (Layered Pane) 110
17.3 Εικονικό Δωμάτιο 111
17.4 Οντότητα Κύβος 115
17.5 Οντότητα Γραμμή 118
17.6 3D Ανίχνευση Επικάλυψης 122
17.7 Εφέ Σκίασης 125
17.8 Επιλογή Αντικειμένου 132
17.9 Εικονικό αόρατο τοίχος 134
17.10 Αρχείο Διαμόρφωσης 135
17.11 RDF - RDFS Parsing 136
17.11.1 Οντότητες (Entities) 136
17.11.2 Σχέσεις (Properties) 138
17.11.3 JTree 139

17.12 Διεπαφή Διαδικτύου 140
17.13 Λειτουργία αντιστοίχησης 141
17.13.1 Απευθείας Αντιστοίχηση (Direct - Instant Mapping) 142
17.13.2 Πολλαπλή Αντιστοίχηση (MultiMapping) 142
17.14 Συνθήκες Αντιστοίχησης (Mapping Conditions) 144
17.15 Εξαγωγή αντιστοίχησης σε XML 144
18 Έλεγχος Απόδοσης Συστήματος 146
19 Μελλοντικές Εξελίξεις (Future work) 152
20 Συμπεράσματα (Conclusion) 153
21 Αναφορές (References) 155

Figure 1. Αναγνώριση αντικειμένου και οπτικό βάθος 19
Figure 2. Τεχνολογία 3D κινηματογράφου.....20
Figure 3. Παράδειγμα 3D φωτογραφίας.....20
Figure 4. Οφθαλμαπάτη μεγέθους σφαίρας (οι σφαίρες έχουν το ίδιο μέγεθος) 24
Figure 5. Node – Link μοντέλο στα Social Networks.....26
Figure 6. Οπτικοποίηση πίνακα σε γράφο 26
Figure 7. Κατανομή 3D προγραμματισμού 31
Figure 8. Τετραγωνική ιεραρχία αντιστοίχησης 32
Figure 9. Ορθογώνια ομαδοποίηση (Orthogonal Cluster Layout) 39
Figure 10. Ιεραρχικό (Hierarchical Layout Technique).....40
Figure 11. Συμμετρικό (Symmetric Layout Technique).....41
Figure 12. Κυκλικό (Circular Layout Technique) 41
Figure 13. Δενδρικό (Tree Layout Technique) 42
Figure 14. Λάθος αυτόματο Layout 43
Figure 15. Αναγνώριση επικάλυψης.....45
Figure 16. Layout απλού γράφου 48
Figure 17. Layout κωνικού δέντρου 49
Figure 18. Οπτικοποίηση Internet Traffic.....50
Figure 19. 3D δέντρο σε επιφάνεια 50
Figure 20. Αναπαράσταση 2.5 διαστάσεων 50
Figure 21. SemNet System.....52
Figure 22. Κάθετο 3D ConeTree.....53
Figure 23. Αναπαράσταση αρχείων με Document Lens 54
Figure 24. Εκτύπωση 3DPS 55

Figure 25. Σύστημα αρχείων σε τρεις διαστάσεις 56
Figure 26. Τεχνικές οπτικοποίησης SDM 56
Figure 27. Δεδομένα βιβλιογραφίας στο BEAD 57
Figure 28. Photo Data Cube 59
Figure 29. 3D Space Partitioning 61
Figure 30. Εικονικά Bounding Boxes 62
Figure 31. 2D επικάλυψη αντικειμένων 63
Figure 32. Λάθος ανίχνευσης Bounding Spheres 63
Figure 33. Space Quadtree με τέσσερα επίπεδα 65
Figure 34. Quadtree όπου κανένα τετράγωνο δεν περιέχει παραπάνω από ένα αντικείμενα 65
Figure 35. Βασικές οντότητες γράφου σκηνής 68
Figure 36. Σωστός Scene Graph 69
Figure 37. Μη σωστός Scene Graph 69
Figure 38. SimpleUniverse Package 70
Figure 39. Κατασκευή Image Plate 71
Figure 40. Μεθόδοι ελέγχουν κατάστασης 72
Figure 41. Compile Method Optimization 73
Figure 42. Διεπαφή προγράμματος Java3D 74
Figure 43. GeometryArray Subclasses 79
Figure 44. Οπτικοποίηση GeometryArray Subclasses 79
Figure 45. GeometryArray γράφος σκηνής 81
Figure 46. Προσανατολισμός Java3D Axis 82
Figure 47. Μέθοδος User Data 83
Figure 48. Ιεραρχία Behavior Class 84

Figure 49. Προβολή PickRay 89
Figure 50. Ο φωτισμός βασίζεται σε διανύσματα επιφάνειας, θέση χρήστη, θέση φωτός 91
Figure 51. Φωτισμένη σφαίρα 91
Figure 52. Directional Light 92
Figure 53. Point Light 92
Figure 54. Spot Light 93
Figure 55. Flat vs Gouraud Shading 93
Figure 56. Different Shining Properties 95
Figure 57. Προβολή σκιάς 96
Figure 58. Δημιουργία Texture σκιάς 97
Figure 59. Coordinates And Image Set 99
Figure 60. Κυκλική διαδρομή στιγμιότυπου εικόνας 108
Figure 61. 3D Mapping Tool Window Layout 110
Figure 62. Επίπεδα παραθύρων 111
Figure 63. Σχεδίαση εικονικού δωματίου 112
Figure 64. Διανύσματα εικονικού χώρου 113
Figure 65. Διανύσματα εικονικού δωματίου 114
Figure 66. Αρχική εικόνα πατώματος 115
Figure 67. Αρχική εικόνα τοίχου 115
Figure 68. Μη σκιασμένο εικονικό δωμάτιο 115
Figure 69. Σκιασμένες άκρες πατώματος 115
Figure 70. Σκιασμένες άκρες τοίχου 115
Figure 71. Σκιασμένο εικονικό δωμάτιο 115
Figure 72. Χαρακτηριστικά οντότητας κύβου 116

Figure 73. Αυτόματη τοποθέτηση κύβου 117
Figure 74. Σχεδίαση οντότητας Κύβος 118
Figure 75. Σχεδίαση άκρων οντότητας γραμμής 119
Figure 76. Ειδική σχεδίαση άκρων για σχέσεις σε παραπλήσιο ύψος 119
Figure 77. Σχεδίαση χρωμάτων 120
Figure 78. Πολλαπλές γραμμές 121
Figure 79. Αρχικοποίηση σφαίρας επικάλυψης 123
Figure 80. Ανίχνευση επικάλυψης βάση υπολογισμού απόστασης 124
Figure 81. Λανθασμένη ανίχνευση επικάλυψης 125
Figure 82. Κίνητρο σκίασης 126
Figure 83. Χαρακτηριστικό διαφάνειας 127
Figure 84. Δημιουργία σκιάς κύβου 128
Figure 85. Φόρτιση Texture σκιάς 128
Figure 86. Δημιουργία σκιάς γραμμής 130
Figure 87. Υπολογισμός θέσης σκιάς 131
Figure 88. Παράδειγμα δημιουργίας σκιάς 132
Figure 89. Ιεραρχία 3D επιλογής αντικειμένου 133
Figure 90. Σχεδίαση εικονικού αόρατου τοίχους 135
Figure 91. Ψευδοκώδικας κατασκευής ιεραρχικού δέντρου για οντότητες 137
Figure 92. Δημιουργία Cidoc CRM δέντρου ιεραρχίας για οντότητες 138
Figure 93. Δημιουργία Cidoc CRM δέντρου ιεραρχίας για σχέσεις 139
Figure 94. HashMap σε JTree 139
Figure 95. Παράδειγμα απευθείας αντιστοίχησης 142
Figure 96. Πολλαπλή αντιστοίχηση για σχέσης 143

Figure 97. Παράδειγμα συνθήκης αντιστοίχησης 144

Figure 98. Παράδειγμα αντιστοίχησης Time Event 145

Figure 99. Παραγόμενο XML για Time Event Mapping..... 145

Code 1. Πρώτο Java3D πρόγραμμα 75

Code 2. Βελτιωμένο πρόγραμμα Java3D 76

Code 3. Συνδυασμός Transform3D 77

Code 4. Κατασκευή GeometryArray 80

Code 5. Αρχικοποίηση GeometryArray 80

Code 6. Αναφορές GeometryArray 80

Code 7. Δημιουργία Text2D 82

Code 8. Γράφος σκηνής Behavior 84

Code 9. Παράδειγμα περιστροφής MouseBehavior 87

Code 10. Παράδειγμα αναφοράς επιλογής 90

Code 11. Κατασκευή διανυσμάτων Σφαίρας 94

Code 12. Κατασκευή πολυγώνου σκιάς 96

Code 13. Κατασκευή Texture 98

Code 14. Εμφάνιση Texture 98

Code 15. Τοποθέτηση συντεταγμένων Texture 98

Code 16. Προεπιλεγμένη/Αντίστροφη κατεύθυνση διανυσμάτων 113

Code 17. Υπολογισμός ενδιάμεσου σημείου πολλαπλών γραμμών 122

Code 18. Ψευδοκώδικας δημιουργίας σκιάς γραμμής 130

Code 19. Παράδειγμα κώδικα 3D επιλογής αντικειμένου 134

Code 20. Παράδειγμα Configuration File 136

Array 1. Κριτήρια Wakeup On.....86

Array 2. MouseBehavior Classes.....86

Array 3. Τύποι αντιστοίχησης.....143

Chart 1. Χρόνος φόρτισης οντοτήτων 147

Chart 2. Επιβάρυνση CPU κατά την φόρτιση σε ποσοστό 148

Chart 3. Επιβάρυνση μνήμης σε MegaByte 149

Chart 4. Επιβάρυνση CPU σε ποσοστό σε κατάσταση αδράνειας 149

Chart 5. Επιβάρυνση CPU κατά την διάρκεια κίνησης σε ποσοστό 150

Chart 6. Ανάλυση απόδοσης – ποιότητας 151

1 Περίληψη – Abstract

1.1 Περίληψη

Στην παρούσα εργασία αναπτύξαμε ένα εργαλείο που βασίζετε πάνω σε τρισδιάστατα γραφικά για την οπτική παρουσίαση της πληροφορίας, την αντιστοίχηση οντοτήτων και σχέσεων ανάμεσα σε διαφορετικές πηγές αλλά και την υποβοήθηση του χρήστη για την γρήγορη και εύκολη αναγνώριση των δομών, της ιεραρχίας και των σχέσεων που αναπτύσσονται ανάμεσα σε οντότητες ή διαφορετικές πηγές.

Σημείο κλειδί στην οπτική παρουσίαση της πληροφορίας αποτελεί η χρήση της τρίτης διάστασης του χώρου, δηλαδή του βάθους, για μια πιο σύγχρονη, εύχρηστη και διαισθητική αναπαράσταση με στόχο την εύκολη, γρήγορη αντίληψη και ευέλικτη διαχείριση των οντοτήτων καθώς και των σχέσεων που τις διέπουν.

Στόχο μας αποτελεί η μελέτη, σχεδίαση και κατασκευή ενός εργαλείου που θα είναι σε θέση να βοηθάει τον χρήστη στην δημιουργία αντιστοιχήσεων οντοτήτων και σχέσεων ανάμεσα σε πληροφοριακά μοντέλα που βασίζονται στο πρότυπο RDFS, με κύριο παράδειγμα την συσχέτιση οποιουδήποτε πληροφοριακού μοντέλου «δηλωμένο σε RDFS», στο μοντέλο CIDOC CRM (Conceptual Reference Model) για να χρησιμοποιηθεί από άλλα εργαλεία για την μετατροπή δεδομένων από το ένα σχήμα στο άλλο.

1.2 Abstract

In this work we developed an information visualization tool based on three dimensional graphics. The core functionality provided by this specific tool enables the end user to map entities and properties between different sources aiding him to quickly and easily recognize the structures, the hierarchy and the relations which are developed between diverse entities or sources.

Key point in our information visualization tool is the usage of the third dimension of space, referring to depth, for a more advanced, user friendly and intuitive representation of information in order to manage entities and their relations easily, quickly and flexibly.

Our goal is to study, design and construct a tool which will be able to aid the end user in creating entities and relations mappings between information models written in RDFS language. Primary focus is on mapping any information model “declared in RDFS” to the CIDOC CRM (Conceptual Reference Model), so it can be used as input from other tools for data conversion among different schemas.

2 Εισαγωγή

Ο τομέας της οπτικοποίησης της πληροφορίας προσπαθεί να δώσει φυσική υπόσταση σε δεδομένα και σχέσεις που μπορεί να έχουν υλική και ποσοτική παρουσία καθώς επίσης και σε δεδομένα και σχέσεις που μπορεί να έχουν αποκλειστικά και μόνον νοητή υπόσταση. Για παράδειγμα ένα αμάξι είναι ένα αντικείμενο με φυσική υπόσταση διότι αποτελείται από ύλη, έχει συγκεκριμένο μέγεθος, βάρος ή χρώμα, και μπορούμε να το δούμε ή να το ακουμπήσουμε. Αντιθέτως, το τι αποτελεί τέχνη ή πολιτισμό δεν είναι κάτι που μπορούμε να το δούμε, να το ακουμπήσουμε, να το ποσοτικοποιήσουμε ή να το εγκλωβίσουμε σε ένα κουτί. Ο τομέας της οπτικοποίησης της πληροφορίας έρχεται να δώσει λύση σε αυτό ταυτίζοντας νοητές έννοιες με φυσικές λέξεις ή γεωμετρικά αντικείμενα ούτως ώστε να γίνει δυνατή η ψηφιοποίηση και αναπαράσταση οποιασδήποτε κατάστασης ή σεναρίου.

Η πληροφορία που προσπαθούμε να οπτικοποιήσουμε μπορεί να γίνει υπερβολικά πολύπλοκη , να περιέχει μεγάλο αριθμό υποπεδίων ή να διακατέχετε από ιεραρχική δομή και πολυάριθμες σχέσεις μεταξύ των οντοτήτων οι οποίες διασταυρώνονται.

Οι πρόσφατες τεχνολογικές εξελίξεις μέσα στην δεκαετία 2000-2010 στον τομέα των γραφικών συστημάτων των υπολογιστών έχουν κάνει πραγματικότητα την δυνατότητα ανάπτυξης και εξελίξεις 3D προγραμμάτων που είναι σε θέση να σχεδιάσουν και να χειριστούν τρισδιάστατο χώρο και αντικείμενα σε πραγματικό χρόνο αλλά και να δώσουν την δυνατότητα στον χρήστη να αλληλεπιδράσει με τον τρισδιάστατο αυτό κόσμο. Χάρη σε αυτά ο χρήστης έχει άμεση ανάδραση την οποία μπορεί να αξιολογήσει και να δράσει ανάλογα.

Επιπλέον, η ύπαρξη της RDF (Resource Description Framework) έχει δώσει το έναυσμα στην δημιουργία μίας γενικής μεθόδου για την περιγραφή διανοητικών περιγραφών ή μοντέλων πληροφορίας, η οποία υλοποιείται βάση διαδικτυακών πηγών. Οι περιγραφές αυτές όμως, δεν ακολουθούν όλες την ίδια μεθοδολογία ή νοοτροπία ακόμα και αν αναφέρονται στις ίδιες έννοιες ή στα ίδια δεδομένα.

Αυτά έχουν δώσει το έναυσμα στον χώρο των επιστημόνων των ηλεκτρονικών υπολογιστών για έρευνα σε ένα εντελώς καινούριο τομέα που αφορά την βέλτιστη και εύχρηστη αποτύπωση της πληροφορίας σε ένα τρισδιάστατο κόσμο καθώς επίσης και των καλύτερων δυνατών τρόπων διαχείρισης και επεξεργασίας της πληροφορίας μας, κάνοντας χρήση και της τρίτης διάστασης του τρισδιάστατου κόσμου (δηλαδή του βάθους). Χάρη στην τρίτη διάσταση μπορούμε να κάνουμε καλύτερη διαχείριση της θέσης που καταλαμβάνουν τα αντικείμενα εκμεταλλευόμενοι διαφορετικά επίπεδα βάθους φέρνοντας τα πιο κοντά και αποφεύγοντας διασταυρώσεις γραμμών.

Έχοντας λάβει όλα αυτά υπόψη, αντιληφθήκαμε την ανάγκη ύπαρξης και υλοποίησης ενός εργαλείου το οποίο θα είναι σε θέση να βοηθάει τον χρήστη στην δημιουργία αντιστοιχήσεων πληροφορίας ανάμεσα σε διαφορετικά μοντέλα αναπαράστασης πληροφορίας τα οποία θα έχουν οριστεί βάση του προτύπου RDF.

3 Τεχνικές Εξαγωγής 3D

Στην πραγματικότητα το ανθρώπινο μάτι είναι σε θέση να αναγνωρίσει τρισδιάστατα αντικείμενα χάρη στην απόσταση που υπάρχει ανάμεσα στα δυο μάτια μας καθώς οι εικόνες που φτάνουν σε αυτά από το ίδιο αντικείμενο παρουσιάζουν ελαφρά διαφορετική γωνία. Ο ανθρώπινος εγκέφαλος είναι σε θέση να συνδυάσει αυτές τις δυο εικόνες και να εξάγει πληροφορία για το σχήμα του αντικείμενου και την απόσταση του από το αντικείμενο.

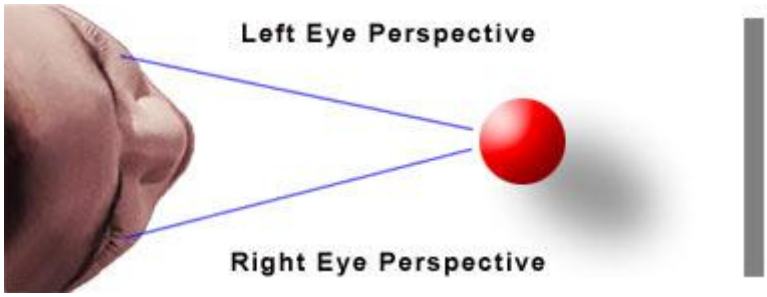


Figure 1. Αναγνώριση αντικειμένου και οπτικό βάθος

3.1 Τεχνολογία Κινηματογράφου

Στις σημερινές κινηματογραφικές αίθουσες είναι αρκετά διαδεδομένες πλέον οι προβολές τρισδιάστατων ταινιών οι οποίες γυρίζονται χάρη σε στερεοσκοπικές κάμερες που διαθέτουν δύο φακούς οι οποίοι προσομοιώνουν τα ανθρώπινα μάτια. Η τεχνολογία αυτή λειτουργεί χάρη στην ύπαρξη δυο προβολέων που αποτυπώνουν στο πανί δυο διαφορετικές εικόνες, η μια σε χρώμα κόκκινο και η άλλη σε χρώμα μπλε ή πράσινο. Τα «απαραίτητα» γυαλιά είναι σε θέση να επιτρέψουν στην μια εικόνα να οδηγηθεί στο ένα μάτι και την άλλη εικόνα να οδηγηθεί στο άλλο μάτι.

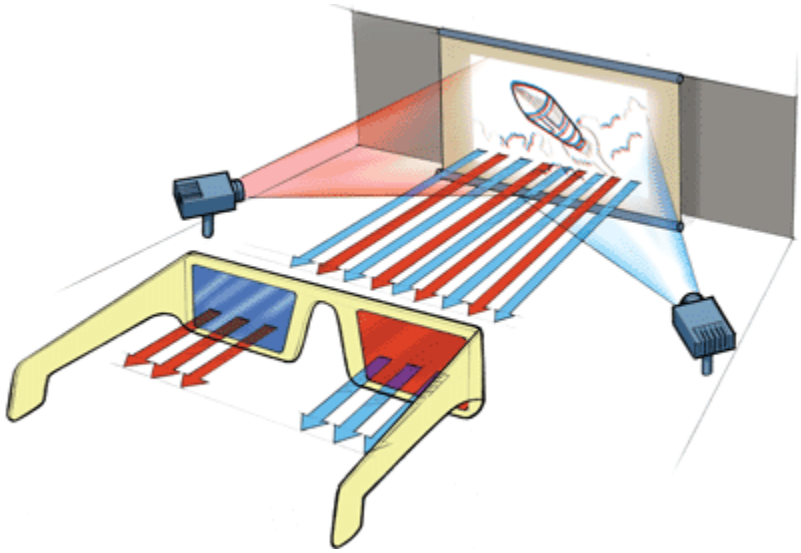


Figure 2. Τεχνολογία 3D κινηματογράφου

Χωρίς τα γυαλιά το μόνο πράγμα που θα ήμασταν σε θέση να δούμε θα ήταν μια θολή εικόνα. Για παράδειγμα οι παρακάτω φωτογραφίες Α και Β έχουν ληφθεί ταυτόχρονα από μια στερεοσκοπική κάμερα. Το αποτέλεσμα του συνδυασμού τους στην ιδία εικόνα παρουσιάζετε στην φωτογραφία C.

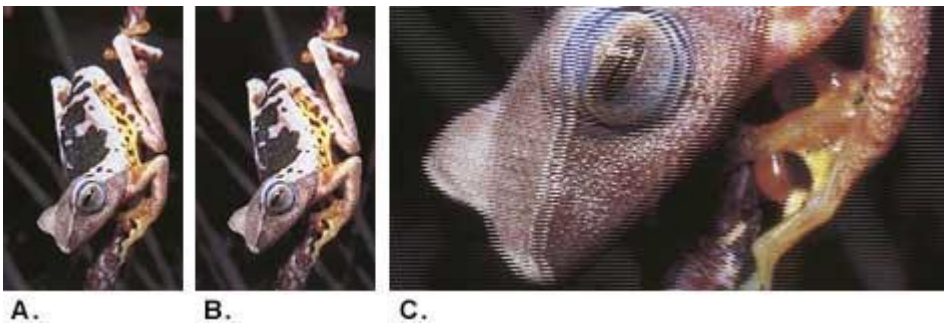


Figure 3. Παράδειγμα 3D φωτογραφίας

Η τεχνολογία αυτή δεν έχει γίνει ευρέως γνωστή εξαιτίας των ειδικών μηχανημάτων και γυαλιών που χρειάζονται για να λειτουργήσει. θα λέγαμε ότι θα ήταν υπερβολικό αν κάθε φορά που θα θέλαμε να αντιληφθούμε τρισδιάστατη αναπαράσταση θα έπρεπε να έχουμε μαζί μας τα απαραίτητα γυαλιά εικονικής πραγματικότητας.

Στην εφαρμογή μας δεν επιδιώκουμε την χρήση αυτής της τεχνολογίας αλλά δεν την απορρίπτουμε κιόλας στα πλαίσια μίας μελλοντικής αναβαθμισμένης έκδοσης που να υποστηρίζει την τεχνολογία αυτή.

3.2 Εξαγωγή Τρισδιάστατων Γραφικών Υπολογιστών

Αντίθετος πιο ευρέως διαδεδομένη τεχνολογία αποτελεί η αποτύπωση τρισδιάστατων γραφικών σε κοινές δυσδιάστατες οθόνες δίνοντας την ψευδαίσθηση στον χρήστη του τρισδιάστατου κόσμου κάνοντας χρήση σχημάτων, θέσης, μεγέθους, χρωμάτων, σκιάσεων και κίνησης. Αυτή είναι και η τεχνολογία που θα χρησιμοποιήσουμε στην εργασία μας. Ο βασικός λόγος χρήσης είναι το ότι δεν χρειάζεται ειδικός εξοπλισμός για την τρισδιάστατη αναπαράσταση των οντοτήτων μας.

4 Γιατί 3D?

4.1 Προτερήματα

Κάνοντας χρήση τρισδιάστατων γραφικών (δηλαδή αν προσθέσουμε την διάσταση z) μπορούμε να παρουσιάσουμε μεγαλύτερη πληθικότητα οντοτήτων στον χρήστη κάνοντας χρήση της ίδιας δυσδιάστατης οθόνης αν τα αντικείμενα τοποθετηθούν στην σωστή θέση στον τρισδιάστατο κόσμο. Καθώς στην ίδια απόσταση χωράει πολύ μεγαλύτερο πλήθος αντικειμένων. Αφού, αν ήμασταν στον δυσδιάστατο χώρο και λαμβάναμε όλα τα σημεία που έχουν απόσταση μικρότερη από d, θα καταλήγαμε σε μία επιφάνεια με το σχήμα κύκλου. Αν όμως, μεταφερόμασταν στον τρισδιάστατο χώρο και λαμβάναμε όλα τα σημεία που έχουν απόσταση μικρότερη από d, τότε θα καταλήγαμε σε ένα αντικείμενο τύπου σφαίρα που διαθέτει πολύ περισσότερο αξιοποιήσιμο χώρο από το κύκλο. Πέραν τούτου, μπορούμε να τοποθετήσουμε τα αντικείμενα στον εικονικό χώρο με τέτοιον τρόπο (εκμεταλλευόμενοι κατάλληλα την τρίτη διάσταση) ώστε να υπάρχει καλύτερη διασύνδεση των οντοτήτων μέσω των σχέσεων. Δηλαδή, δεδομένου ότι πλέον οι οντότητες εισάγονται στον τρισδιάστατο χώρο μπορούν πλέον όσες από αυτές σχετίζονται να εισαχθούν πλησιέστερα καθώς αν δεν χωράνε στο ίδιο επίπεδο μπορούν να εισαχθούν σε διαφορετικό αλλά διατηρώντας την ίδια απόσταση μεταξύ τους. Επιπλέον, οι σχέσεις που ενδέχεται να υπάρχουν ανάμεσα στις οντότητες έχουν μικρότερη πιθανότητα να συγκρούονται με άλλα αντικείμενα εφόσον αυτά θα είναι σε διαφορετικά επίπεδα.

Το προτέρημα αυτό που παρουσιάζει η σχεδίαση στον τρισδιάστατο χώρο είναι ιδιαίτερα χρήσιμο στα μοντέλα που βασίζονται στο πρότυπο RDF καθώς έχουμε πολλαπλές συνδέσεις μεταξύ κόμβων. Αφού, αν βασιζόμασταν σε μία σχεδίαση που θα έκανε χρήση ενός δυσδιάστατου επιπέδου, ενδέχεται , το μήκος των links ανάμεσα στις οντότητες να ήταν πολύ μεγάλο και ανάλογα με τις κατασκευαστικές αρχές θα

μπορούσε να ακολουθεί μη λογική πορεία σε περίπτωση αποφυγής σύγκρουσης με οντότητες.

Πέραν αυτού, η κατασκευή στις δύο διαστάσεις (σε περίπτωση πολλαπλών συνδέσεων) συνεπάγεται την ύπαρξη πολλών διασταυρώσεων σχέσεων ή απομακρυσμένων διασυνδέσεων, που με την σειρά του συνεπάγεται την ενδεχόμενη παραπλάνηση του χρήστη και την παρεμπόδιση του έργου του.

Επιπλέον, εξαιτίας τις οικονομίας χώρου που πρέπει να γίνει στις δύο διαστάσεις, είναι πιο δύσκολο να τοποθετηθούν οι οντότητες σε σωστή θέση ώστε να αναδυθούν σχέσεις και προτιμήσεις όπως για παράδειγμα ότι η οντότητα γονέας θα πρέπει να βρίσκεται πάνω από την οντότητα παιδί. Η επιλογή αυτή χαρακτηρίζεται ως απαραίτητη διότι στους διανοητικούς γράφους της RDF επιθυμούμε το πιο γενικό χαρακτηριστικό, η πιο γενική οντότητα, η οντότητα την οποία κληρονομούν άλλες να εμφανίζεται πιο ψηλά στην διάσταση του ύψους. Αυτό το πρόβλημα δεν εντοπίζεται στον τρισδιάστατο χώρο καθώς διαθέτουμε άπειρο βάθος για την τοποθέτηση της οντότητας στο επιθυμητό ύψος αλλά σε διαφορετικό επίπεδο βάθους.

Έτσι, υιοθετούμε μία αρχή που υποστηρίζει ότι οι σχέσεις τύπου Isa «επιθυμούμε» να έχουν αρχικό και τελικό σημείο που βρίσκεται σε διαφορετικό υψόμετρο και οι σχέσεις τύπου Property να έχουν αρχικό και τελικό σημείο που να βρίσκονται στο ίδιο υψόμετρο. Το επίπεδο βάθους του αρχικού και του τελικού σημείου μπορεί να ποικίλει ανάλογα με την περίπτωση ή τις επιλογές του χρήστη. Η αρχή αυτή δεν θα μπορούσε να επιτευχθεί χωρίς την χρήση της τρίτης διάστασης του βάθους.

Ο βασικός λόγος που επιθυμούμε αυτή την σχεδίαση των σχέσεων είναι ότι χάρη σε αυτήν, ο χρήστης μπορεί να αντιλαμβάνεται διαισθητικά την ιεραρχία και την δομή της πληροφορίας

Επίσης η τρίτη διάσταση μπορεί να χρησιμοποιηθεί με τέτοιο τρόπο ώστε να προσδώσει γνωρίσματα και χαρακτηριστικά στα αντικείμενα μόνο και μόνο από την θέση που καταλαμβάνουν και έτσι το σχήμα να είναι πιο ευέλικτο και να μην περιοριστεί στην αποτύπωση αντικειμένων σε ένα και μόνο επίπεδο. Για παράδειγμα στον 3D κόσμο τα σχήματα δεν έχουν όλα το ίδιο μέγεθος καθώς είναι μικρότερα όσο πιο βαθύτερα εμφανίζονται και επίσης αντικείμενα μπορούν να κρύβονται πίσω από άλλα ή και να εμφανίζονται μερικώς. Άρα με μια σωστή χρήση του χώρου ο χρήστης μπορεί να είναι σε θέση να βλέπει μεγαλύτερο πλήθος οντοτήτων που χωράνε στην ίδια απόσταση ή και καλυτέρα οργανωμένη και έτσι η τρισδιάστατη απεικόνιση χαρακτηρίζετε ως πιο αποδοτική και ικανοποιητική. Πράγμα που είναι ιδιαίτερα ικανοποιητικό όταν έχουμε να κάνουμε με μεγάλο πλήθος δεδομένων που ενδέχεται να συνδέονται μεταξύ τους με ένα ακόμα μεγαλύτερο αριθμό σχέσεων. Όπως έχουν

δείξει και αποτελέσματα ερευνών [1] η παρουσίαση πληροφορίας στον τρισδιάστατο κόσμο μπορεί να βοηθήσει περισσότερο τον χρήστη στην κατανόηση της πληροφορίας καθώς ο εγκέφαλος μας είναι συνηθισμένος στην επεξεργασία τρισδιάστατων εικόνων και αποκτούν νόημα οι κατευθύνσεις των γραμμών και η τοποθέτηση αντικειμένων σε διαφορετικά επίπεδα. Έτσι ο χρήστης θα είναι σε θέση να αντιληφθεί την πληροφορία, διαισθητικά, δίχως να καταβάλει μεγάλη προσπάθεια.

Θα πρέπει να τονίσουμε ότι καθώς στον τρισδιάστατο χώρο τι εστί πάνω και τι εστί κάτω είναι έννοιες υποκειμενικές, αφού τα χαρακτηριστικά αυτά μπορούν να αντιστραφούν με ένα απλό Rotation, το εικονικό δωμάτιο περιέχει μία πλευρά με διαφορετικό δάπεδο το οποίο προσομοιώνει το πάτωμα, ώστε ο χρήστης να έχει την αίσθηση του προσανατολισμού.

4.2 Μειονεκτήματα

Όπως βλέπουμε και στην εικόνα που ακολουθεί είναι πολύ εύκολο να παραπλανηθεί ο χρήστης παρουσιάζοντας αντικείμενα σε έναν εικονικό χώρο εφόσον η ανθρώπινη αντίληψη εξάγει συμπεράσματα από το σύνολο των γραφικών στοιχείων. Για παράδειγμα παρότι οι δύο σφαίρες που παρουσιάζονται στην εικόνα έχουν ακριβώς το ίδιο μέγεθος, το ανθρώπινο μάτι αντιλαμβάνεται τη σφαίρα που βρίσκεται στη δεξιά πλευρά ως μεγαλύτερου μεγέθους επειδή δίνεται αυτή η ψευδαίσθηση στον χρήστη χάρη στην σμίκρυνση που παρατηρείται στο πλάτος του εικονικού τοίχους. Τέτοιες περιπτώσεις παραπλάνησης της ανθρώπινης αντίληψης έχουν ληφθεί υπόψη και αποφεύγονται εξαρχής χάρη στη σωστή σχεδίαση του συστήματος.

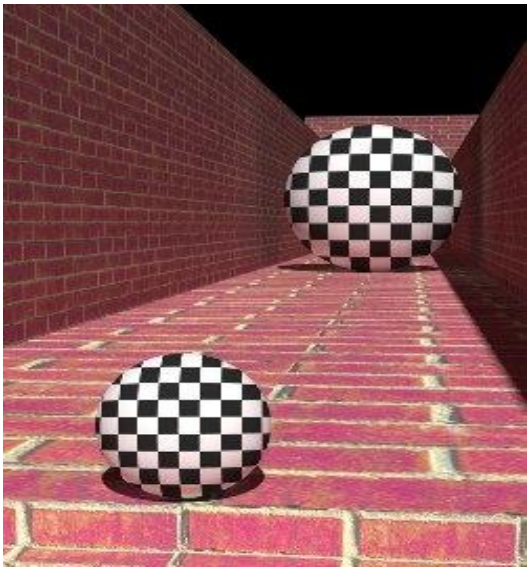


Figure 4. Οφθαλμαπάτη μεγέθους σφαίρας (οι σφαίρες έχουν το ίδιο μέγεθος)

Επιπλέον μειονέκτημα παρουσίασης πληροφορίας στον τρισδιάστατο κόσμο αποτελεί το γεγονός ότι καθώς επιτρέπεται να εισάγουμε οντότητες βαθύτερα στον εικονικό καμβά μπορεί ο χρήστης να αντιληφθεί το γεγονός ότι σε αυτό το σημείο υπάρχει κάποια οντότητα αλλά δυστυχώς χάνει τη δυνατότητα να αντιληφθεί το αναγνωριστικό της οντότητας αυτής ή την επιπλέον πληροφορία που αυτή φέρει.

Επίσης υπάρχει πιθανότητα η ύπαρξη κάποιας οντότητας να επισκιάζεται από την παρουσία κάποιας άλλης. Δηλαδή επάνω στην νοητή γραμμή που συνδέει την οντότητα με τον χρήστη ενδέχεται να υπάρχει άλλη οντότητα της οποίας η απόσταση από τον χρήστη να είναι μικρότερη από της πρώτης. Με πιο απλά λόγια υπάρχει πιθανότητα κάποια οντότητα να κρύβεται πίσω από κάποια άλλη. Γι’ αυτό δόθηκε η δυνατότητα στον χρήστη περιήγησης στον εικονικό χώρο και να μετακινεί χειροκίνητα εικονικά αντικείμενα ώστε να μπορεί να μεταφερθεί παντού ή να μετακινήσει αντικείμενα τα οποία θεωρεί ότι παρευρίσκονται σε σημείο που εμποδίζει την παρατήρησή του.

5 Οπτικοποίηση Δεδομένων

Ορίζεται ως η χρήση υπολογιστικών συστημάτων αλληλεπίδρασης με τον χρήστη για την οπτική αναπαράσταση δεδομένων με στόχο την ενίσχυση και την υποβοήθηση της ανθρώπινης αντίληψης [60].

Η οπτικοποιήση της πληροφορίας αποτελεί τομέα της επιστήμης των υπολογιστών που ασχολείται με την γραφική αναπαράσταση της πληροφορίας. Η οπτικοποιήση της

πληροφορίας γίνεται για δυο βασικούς λόγους. Ο πρώτος αφορά την απόδοση διαισθητικού νοήματος στην ιεραρχία του περιέχεται στην δομή της πληροφορίας και δεύτερον, η καλύτερη επικοινωνία ανάμεσα σε άνθρωπο και μηχανή (καλύτερη επίτευξη αλληλεπίδρασης) αλλά και καλύτερη επικοινωνία ανάμεσα σε επιστήμονες της τεχνολογίας των υπολογιστών και όχι μόνο. Όπως για παράδειγμα σχεδιαστές οπτικών συστημάτων, στατιστικολόγους κτλ.

Τα δεδομένα μπορούν να κουβαλάνε μέσα τους σημαντική ποσότητα πληροφορίας. Η οπτικοποίηση δεδομένων είναι ένα ισχυρό εργαλείο παρουσίασης, κατανόησης αλλά και αποκάλυψης-ανακάλυψης καινούριας πληροφορίας η οποία μπορεί να είναι σαφώς ορισμένη αλλά και να κρύβεται μέσα στα δεδομένα. Η πληροφορία που προσπαθούμε να παρουσιάσουμε με εικονικά σχήματα μπορεί να έχει φυσική και πραγματική υπόσταση στον πραγματικό κόσμο μας αλλά και να αποτελεί μια έννοια ιδεατή που υπάρχει αποκλείστηκα και μόνο μέσα στο μυαλό μας.

Η μεταφορά αυτή των ιδεατών εννοιών σε εικονικά σχήματα που έχουν χαρακτηριστικά όπως μέγεθος, χρώμα, σχήμα μπορεί να θεωρηθεί επιτυχής μόνο όταν κατανοήσουμε σωστά την οπτική αντίληψη και διαίσθηση που μπορεί να παραπέμπει στην έννοια προς απεικόνιση. Είναι εξάλλου γνωστό ότι από μια και μόνο εικόνα μπορεί να εξαχθεί μεγάλη ποσότητα πληροφορίας αν η εικόνα αυτή έχει αποτυπωθεί σωστά και με τον καλύτερο τρόπο. Γι’ αυτό και για να προσδώσουμε οπτική απεικόνιση σε δεδομένα θα πρέπει να ακολουθήσουμε ορισμένους κανόνες και αρχές που να συμβαδίζουν με τις δυνατότητες και την επίγνωση της ανθρώπινης αντίληψης.

Για παράδειγμα αν έχουμε έναν πίνακα με τις πωλήσεις, έσοδα και τα έξοδα μιας επιχείρησης ανά μήνα θα δυσκολευτούμε να εξάγουμε εύκολα και γρήγορα πληροφορία απλά και μόνο βλέποντας τον. Καθώς η ανθρώπινη αντίληψη δεν έχει εκπαιδευτεί σε αυτήν την μορφή αναπαράστασης. Αν όμως μετατρέψουμε αυτόν τον πίνακα σε μια γραφική αναπαράσταση που παρουσιάζει ακριβώς τα ίδια δεδομένα διαπιστώνουμε ότι μπορούμε να αντλήσουμε πολύ εύκολα και γρήγορα πληροφορία. Επίσης χάρη στην γραφική αναπαράσταση της πληροφορίας μπορεί να μειώνεται η ακρίβεια των δεδομένων αλλά αντίθετα μπορούμε να εξάγουμε μια γενική άποψη επί του συνόλου και να συγκρίνουμε στοιχεία μεταξύ τους. Η σύγκριση αυτή γίνεται εφικτή χάρη στην δυνατότητα σύγκρισης και αξιολόγησης της οπτικής πληροφορίας που παρουσιάζεται στον χρήστη. Καθώς η αντίληψη μας μπορεί εύκολα να συγκρίνει σχήματα, μεγέθη και χρώματα.

Έτσι, εφόσον τα δεδομένα μας οπτικοποιηθούν δεν μιλάμε πλέον για σύγκριση μεταξύ αριθμών αλλά μεταξύ σχημάτων. Επιπλέον, η οπτικοποίηση πληροφορίας προσφέρεται για την παρουσίαση σχέσεων μεταξύ οντοτήτων κάνοντας χρήση του μοντέλου κόμβων

– δεσμών, όπου οι οντότητες παρουσιάζονται ως κόμβοι και οι σχέσεις μεταξύ τους ως δεσμοί.

Χάρη στην οπτικοποίηση των οντοτήτων και των σχέσεων μπορούμε να εξάγουμε εύκολα και γρήγορα συμπεράσματα όπως για παράδειγμα τα Social Networks όπου ουσιαστικά γίνεται σύγκριση μεταξύ πυκνότητας σχέσεων ανά οντότητα.

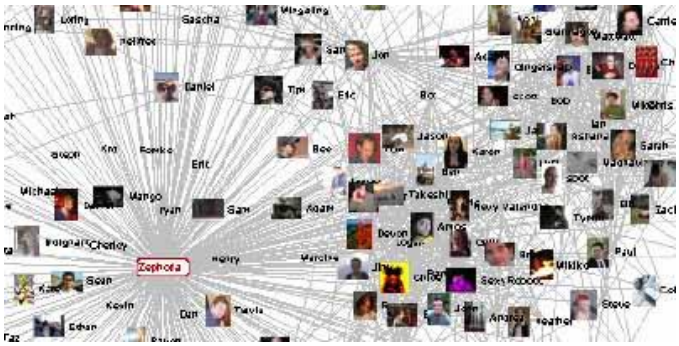


Figure 5. Node – Link μοντέλο στα Social Networks

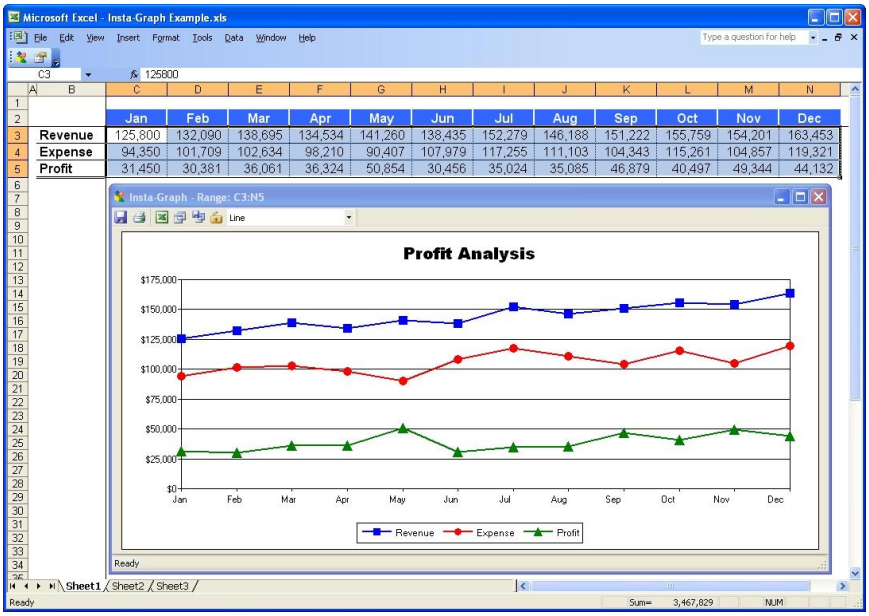


Figure 6. Οπτικοποίηση πίνακα σε γράφο

Για να παρουσιαστεί επιτυχώς η πληροφορία με οπτικό τρόπο στον χρήστη θα πρέπει να ικανοποιηθούν κάποιες προϋποθέσεις τις οποίες και παρουσιάζουμε παρακάτω.

Θα πρέπει να :

- δηλώνετε ξεκάθαρα ο τύπος των οντοτήτων και των σχέσεων.

- αναπαριστάτε σαφώς η πληθικότητα.
- γίνεται εύκολα σύγκριση μεταξύ όμοιων αντικειμένων
- γίνεται αντιληπτή η διαβάθμιση των αντικειμένων (πχ το αντικείμενο με την μεγαλύτερη πληθικότητα).
- είναι εμφανής ο τρόπος με τον οποίο μπορεί ο χρήστης να χειριστεί αυτήν την πληροφορία.
- βρίσκονται κοντά αντικείμενα που είναι όμοια ή που ανήκουν σε ομάδα ή που σχετίζονται.
- διαθέτουν όμοια χαρακτηριστικά (όπως χρώμα) αντικείμενα που σχετίζονται εννοιολογικά.
- παρέχεται δυνατότητα ενθυλάκωσης (οντότητες μέσα σε οντότητες) πληροφορίας αν αυτή είναι απαραίτητη.
- γίνεται πλήρη αναπαράσταση όλων των οντοτήτων και των σχέσεων (αν αυτό είναι απαραίτητο).
- γίνεται αναπαράσταση μόνον των αντικειμένων που επιθυμεί ο χρήστης (αν αυτό είναι επιθυμητό)

Στην εφαρμογή μας προσπαθήσαμε να ικανοποιήσουμε στο έπακρο τα παραπάνω σημεία. Για να επιτύχουμε το επιθυμητό αποτέλεσμα βασιστήκαμε στο σωστό φωτισμό, στον σωστό έλεγχο και στην σωστή εκτέλεση κίνησης, στην σωστή επιλογή σχημάτων και στην σωστή ολική προβολή για να αντιληφθούμε με τον καλύτερο τρόπο την τρισδιάστατη αναπαράσταση.

Επιπλέον θέλοντας να αναπαραστήσουμε γραφικά τους διανοητικούς γράφους της RDF επιθυμούμε η οντότητα την οποία κληρονομούν άλλες να εμφανίζεται πιο ψηλά στην διάσταση του ύψους. Οι σχέσεις τύπου Isa «επιθυμούμε» να έχουν αρχικό και τελικό σημείο που βρίσκεται σε διαφορετικό υψόμετρο και οι σχέσεις τύπου Property να έχουν αρχικό και τελικό σημείο που να βρίσκονται στο ίδιο υψόμετρο. Χάρη σε αυτήν την σχεδίαση, ο χρήστης μπορεί να αντιλαμβάνεται διαισθητικά την ιεραρχία και την δομή της πληροφορίας. Επιπλέον, δεδομένου ότι επιθυμούμε να έχουμε όσο το δυνατόν λιγότερα αντικείμενα στον εικονικό κόσμο και το τελικό αποτέλεσμα του σχήματος να είναι όσο το δυνατόν λιγότερο πολύπλοκο περιορίζουμε την αποτύπωση

των σχέσεων Isa μόνο στις οντότητες που σχετίζονται άμεσα με τα χαρακτηριστικά του προγόνου και του απογόνου, και όχι αναδρομικά.

6 Αντιστοίχηση

Η αντιστοίχηση ορίζεται ως μία διαδικασία δημιουργίας συσχέτισης μεταξύ δύο διαφορετικών μοντέλων δεδομένων [72].

Αποτελεί ένα πρώτο βήμα για μία διαδικασία ενοποίησης δεδομένων που προέρχονται από μία μεγάλη ποικιλία πηγών και περιλαμβάνει λειτουργίες όπως:

- Μετατροπή δεδομένων μεταξύ σχήματος πηγής και σχήματος στόχου.
- Εντοπισμού και ανάλυση σχέσεων δεδομένων.
- Ανακάλυψης και ανάκτησης κρυμμένης πληροφορίας.
- Ενοποίησης διαφορετικών βάσεων δεδομένων σε μία.

Για παράδειγμα, αν δύο επιχειρήσεις που διαθέτουν διαφορετικά σχήματα βάσεων δεδομένων θελήσουν να ανταλλάζουν πληροφορία, θα πρέπει να χρησιμοποιήσουν κάποια λειτουργία αντιστοίχησης δεδομένων ώστε να δημιουργήσουν πρότυπα που θα μετατρέπουν αυτόματα την πληροφορία από το ένα μοντέλο στο άλλο και πίσω [73]. Εναλλακτικά, θα πρέπει και οι δύο επιχειρήσεις να υιοθετήσουν το ίδιο πρωτόκολλο επικοινωνίας [74].

Αυτό σημαίνει ότι και οι δύο επιχειρήσεις θα πρέπει να χρησιμοποιήσουν μία διαδικασία αντιστοίχησης δεδομένων ώστε να μετατρέπουν πληροφορία που στεγάζετε στο δικό τους σχήμα βάσης δεδομένων στο σχήμα που προσφέρεται από το κοινό πρωτόκολλο αυτόματα, και επίσης να μπορούν αυτόματα, να μετατρέπουν πληροφορία που ακολουθεί το πρότυπο του κοινού πρωτοκόλλου επικοινωνίας σε ένα στιγμιότυπο του δικού τους σχήματος βάσεων δεδομένων.

Στην σημερινή εποχή έχουν προταθεί και κατασκευαστεί πολλά εργαλεία που βοηθούν τον χρήστη στην κατασκευή μοντέλων αντιστοίχησης δεδομένων και το καθένα από αυτά απευθύνεται σε διαφορετικό πρότυπο καταχώρησης δεδομένων και προσφέρει διαφορετικά εργαλεία και διευκολύνσεις στους χρήστες του.

Ωστόσο, παρότι υπάρχουν στην αγορά πολλά εργαλεία που υποστηρίζουν την διαδικασία αντιστοίχησης, θα λέγαμε ότι δεν υπάρχει κάποιο που να μπορεί να υποστηρίζει όλα τα πρότυπα καταχώρησης δεδομένων και όλες τις πιθανές ανάγκες

των χρηστών. Γι' αυτό πολλές φορές, οι επιχειρήσεις προτιμούν να κατασκευάσουν εξ' αρχής ένα δικό τους εργαλείο που θα είναι σύμφωνο με τις ανάγκες τους από το να χρησιμοποιήσουν κάποιο έτοιμο εργαλείο, ακόμα και αν αυτό στοιχίζει πολύ περισσότερο.

Η αντιστοίχηση δεδομένων μπορεί να γίνει με πολλούς διαφορετικούς τρόπους. Ο πιο δημοφιλής και ο πιο εύκολος στον χειρισμό είναι αυτός που κάνει χρήση γραφικών εργαλείων. Είναι συνήθως εφαρμογές που κάνουν χρήση γεωμετρικών σχημάτων όπως γραμμές για να αναπαραστήσουν την αντιστοίχηση μεταξύ οντοτήτων πηγής και στόχου. Επίσης, πολλά εργαλεία υποστηρίζουν και μία μορφή αυτόματης αντιστοίχησης μεταξύ δύο μοντέλων βασιζόμενα στην ομοιότητα που εμφανίζεται ανάμεσα σε λέξεις. Όμως, η τεχνική αυτή έχει αποδειχτεί στην πράξη ότι παράγει αμφιβόλου ποιότητας και αξιοπιστίας αποτελέσματα.

Η εφαρμογή μας αναπτύχθηκε με στόχο την κατασκευή ενός εργαλείου που θα είναι σε θέση να δημιουργεί αντιστοιχήσεις ανάμεσα σε μοντέλα που βασίζονται στο πρότυπο RDF, με παράδειγμα το μοντέλο CIDOC CRM. Θα αποτελεί ένα γραφικό εργαλείο το οποίο θα κάνει χρήση τρισδιάστατων γραφικών για την αναπαράσταση των οντοτήτων, των σχέσεων μεταξύ τους καθώς επίσης και των αντιστοιχήσεων ανάμεσα στα δύο μοντέλα.

Η επιλογή της RDF δεν είναι τυχαία αφού στο μέλλον, η χρήση της RDF υπόσχετε περισσότερο αυτοματοποιημένες διαδικασίες αντιστοίχησης δεδομένων [75]. Καθώς, η ίδια η γλώσσα είναι προσπελάσιμη και κατανοητή από υπολογιστές ανοίγει τον δρόμο για την αυτόματη αντιστοίχηση μοντέλων δεδομένων ανάμεσα σε διαφορετικά πρότυπα.

7 RDF

Πρόκειται για ένα μοντέλο δεδομένων το οποίο είναι παρόμοιο με τη κλασική προσέγγιση διανοητικού μοντελισμού όπως τα μοντέλα ER (Entity Relationship) ή τα Class Diagrams, καθώς βασίζεται στην ιδέα της δημιουργίας δηλώσεων για πηγές του διαδικτύου [76], [77]. Οι δηλώσεις αυτές έχουν μορφή subject – predicate – object και είναι ευρέως γνωστές με την επονομασία τριπλέτες. Το subject δηλώνει την πηγή, το predicate δηλώνει την σχέση ανάμεσα στο subject και το object, και το object δηλώνει την τιμή του predicate. Οι επονομασίες αυτές είναι αλλιώς γνωστές και ως Resource, Property, and Property value και μία συλλογή από τριπλέτες αποτελεί έναν κατευθυνόμενο γράφο.

Για παράδειγμα, ένας τρόπος για να καταχωρηθεί η πρόταση «The author of http://www.w3schools.com/rdf is Jan Egil Refsnes» θα μπορούσε να είναι:

Subject = http://www.w3schools.com/rdf

Predicate = author

Object = Jan Egil Refsnes

Η RDF έχει σχεδιαστεί για είναι προσβάσιμη και κατανοητή από υπολογιστές. Αυτό αποτελεί βασικό προτέρημα καθώς χάρη σε αυτήν μπορούν οι υπολογιστές να αντιληφθούν και να χειρίζονται αυτόματα πληροφορία. Δεν έχει σχεδιαστεί για να παρουσιάζεται σε ανθρώπους και η πιο συνηθισμένη καταγραφή της γίνεται σε XML αν και δεν είναι ο μόνος τρόπος για να δημιουργηθούν RDF μοντέλα. Τέλος, αποτελεί κομμάτι και πρόταση του W3C's Semantic Web Activity.

7.1 RDFS

Αποτελεί μία επέκταση της RDF και είναι σε θέση να περιγράψει τις πηγές κάνοντας χρήση classes, properties και values. Οι κλάσεις αυτές θυμίζουν έντονα τις αντίστοιχες του οντοκεντρικού προγραμματισμού. Αυτό επιτρέπει σε πηγές να περιγραφούν ως στιγμιότυπα κλάσεων ή υποκλάσεις κλάσεων και επίσης διαθέτουμε την ιδιότητα της κληρονομιάς χάρη στις σχέσεις Isa που αναπτύσσονται.

Επίσης, αποτελεί κατάλληλο πρότυπο για την περιγραφή του μοντέλου Cidoc CRM.

Στην εφαρμογή μας είμαστε σε θέση να χειριστούμε αρχεία RDF/S. Έτσι θα διαθέτουμε προς αναπαράσταση πληροφορία για Entities τα οποία θα συνδέονται μεταξύ τους με σχέσεις και κληρονομικότητα Isa. Τα Entities θα συμβολίζονται με την εικονική οντότητα Κύβος και οι σχέσεις με Ευθύγραμμα τμήματα.

8 Στόχοι

Στην παρούσα εργασία βασικά θέματα που μας απασχολούν είναι η πληροφορία που θα παρουσιάζεται στο χρήστη, τα αντικείμενα που θα υπάρχουν στον τρισδιάστατο κόσμο, οι συμβολισμοί που θα χρησιμοποιηθούν, οι λειτουργίες που θα πραγματοποιούνται αυτόματα από το σύστημα και οι δυνατότητες που θα προσφέρονται στο χρήστη από την αλληλεπίδραση του με την δυσδιάστατη ή τρισδιάστατη διεπαφή που χρησιμοποιείται να προσφέρονται με τον καλύτερο δυνατό τρόπο.

Για να ικανοποιηθούν στον βέλτιστο βαθμό όλοι αυτοί οι στόχοι χρειάστηκε να συνδυαστούν γνώσεις, εμπειρίες και απόψεις από τέσσερις ομάδες ατόμων. Αυτές ήταν ο χρήστης, ο σχεδιαστής, ο μαθηματικός και ο προγραμματιστής. Βασικό μέλημα του χρήστη ήταν το τι τον βολεύει και τι θα ήθελε, του σχεδιαστή ήταν το τι θα ήταν ωραίο, του μαθηματικού ήταν οι συναρτήσεις που θα έπρεπε να χρησιμοποιηθούν και του προγραμματιστή η υλοποίηση των παραπάνω.

Οι στόχοι ήταν ιδιαίτερα δύσκολοι να επιτευχθούν από τις ομάδες του μαθηματικού και του προγραμματιστή. Αυτό συμβαίνει καθώς οι βασικές μαθηματικές γνώσεις δεν επαρκούν για υπολογισμούς και πράξεις που αφορούν τον τρισδιάστατο χώρο. Επίσης η προγραμματιστική υποδομή η οποία αναφέρεται στην σχεδίαση τρισδιάστατων γραφικών, που υπάρχει αυτή την στιγμή, είναι πολύ περιορισμένη, έχει συγκεκριμένες δυνατότητες και υπάρχει ελάχιστη ποσότητα πληροφορίας και ενημέρωσης διαθέσιμη σε πηγές. Επιπλέον, η περισσότερη έρευνα που αφορά την κατασκευή μηχανών παραγωγής τρισδιάστατών γραφικών, έχει γίνει από εταιρίες κατασκευής παιχνιδομηχανών και αποτελεί επτασφράγιστο μυστικό και πνευματική ιδιοκτησία των εταιριών.

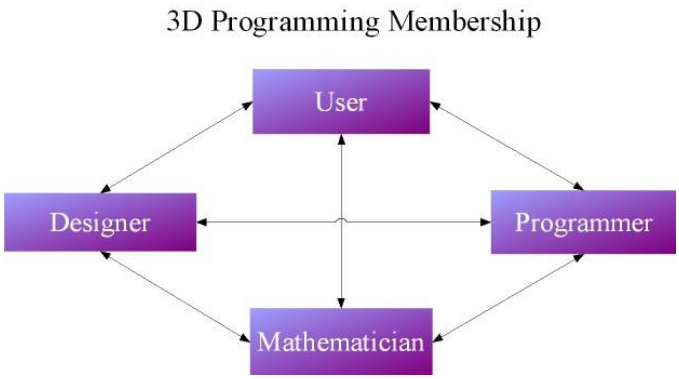


Figure 7. Κατανομή 3D προγραμματισμού

8.1 Αντιστοίχηση

Η ανθρώπινη αντίληψη θα βελτιωνόταν αισθητά εάν το σχήμα του Source και Target μοντέλου ήταν παραπλήσια και αν οντότητες που αντιστοιχούνται ανάμεσα σε διαφορετικά μοντέλα βρίσκονται σε παραπλήσιο ύψος ή παραπλήσια θέση. Δηλαδή οντότητες που σχετίζονται να βρίσκονται κοντά η μία στην άλλη.

Κατά την διάρκεια της αντιστοίχησης οντοτήτων θα βοηθούσε πολύ την ανθρώπινη αντίληψη αν η ιεραρχία των οντοτήτων διατηρούνταν ανάμεσα σε Source και Target Models. Με αυτόν τον τρόπο ο χρήστης θα ήταν σε θέση να αντιληφθεί πολύ πιο γρήγορα την μεταφορά από το ένα μοντέλο στο άλλο σχηματίζοντας έτσι ένα

εννοιολογικό σχήμα τετραγώνου το οποίο θα περιέχει τις οντότητες, τις σχέσεις και αντιστοιχίσεις μεταξύ τους.

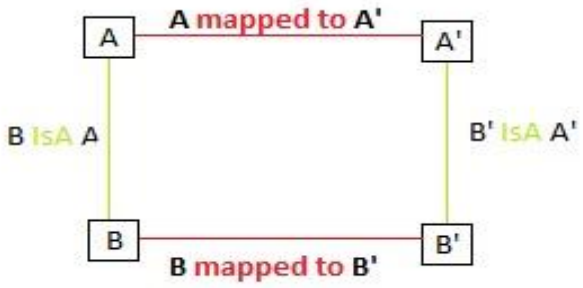


Figure 8. Τετραγωνική ιεραρχία αντιστοίχησης

8.2 Ανθρώπινη Αντίληψη

Ο απόλυτος στόχος που οποιοδήποτε συστήματος παρουσίασης και επεξεργασίας πληροφορίας είναι να εμφανίσει την πληροφορία γρήγορα, εύκολα και με σαφήνεια στον χρήστη γι' αυτό είναι σημαντικό να λάβουμε υπόψη μας τις ανθρώπινες δυνατότητες παρατήρησης και αναγνώρισης προτύπων και σχέσεων. Γι' αυτό και δεν περιοριζόμαστε στην χρήση δυσδιάστατου γραφικού περιβάλλοντος αλλά επεκτεινόμαστε στη σχεδίαση ενός τρισδιάστατου κόσμου οπού τα αντικείμενα μας θα έχουν γεωμετρικό σχήμα, χρώμα, μέγεθος, ύψος, πλάτος και βάθος. Έτσι σε περίπτωση που ένα αντικείμενο συγκρούεται ή περιέχεται σε ένα άλλο τότε μπορεί να γίνει χρήση της τρίτης διάστασης του βάθους και το ένα από τα δύο αντικείμενα να τηλεμεταφερθεί σε διαφορετικό σημείο λιγότερο ή περισσότερο βαθιά στον εικονικό κόσμο.

Η ανθρώπινη αντίληψη έχει εκπαιδευτεί να αντιδρά πιο εύκολα και γρηγορότερα σε ευμεγέθη αντικείμενα με έντονο χρωματισμό και να ξεχωρίζει σχήματα βασικής γεωμετρίας όπως τετράγωνο, κύκλο, ευθεία. Χάρις σε αυτό επιλέχθηκαν ως οντότητες του εικονικού κόσμου μόνο γεωμετρικά σχήματα που ικανοποιούν αυτές τις συνθήκες. Έτσι ο χρήστης είναι σε θέση να αντιληφθεί πιο γρήγορα και εύκολα τα δεδομένα που του παρουσιάζονται και η διεπαφή δεν επιβαρύνεται με πολύπλοκες δυσδιάκριτες οντότητες, και ο χρήστης θα είναι σε θέση να εστιάσει την προσοχή του στα αντικείμενα με όσο το δυνατόν λιγότερη προσπάθεια.

8.3 Αντικείμενα

Έμφαση δίνεται στο ότι τα αντικείμενα που χρησιμοποιούνται θα πρέπει να είναι ευδιάκριτα και να τραβάνε εύκολα την προσοχή του χρήστη δίχως αυτός να χρειάζεται να καταβάλει έντονη προσπάθεια για την αναγνώριση τους(εξοικονομώντας κόπο , χρόνο και χώρο στο 3D κόσμο).

8.4 Σχεδίαση

Επίσης η ίδια η σχεδίαση των αντικειμένων θα πρέπει να είναι σε θέση να παρουσιάσει και να εξυπηρετήσει την πραγματική εννοιολογική λειτουργία τους και να μην αποκρύψει πληροφορία από το χρήστη. Για παράδειγμα τα αντικείμενα που εισάγουμε στον εικονικό κόσμο και που συμβολίζουν οντότητες και σχέσεις θα μπορούσαν να προσομοιωθούν με το γεωμετρικό σχήμα του κύβου και του ευθύγραμμου τμήματος αντίστοιχα. Καθώς ο κύβος αποτελεί μία ολοκληρωμένη βασική γεωμετρική οντότητα και οι γραμμές υπάρχουν συνήθως για να συνδέουν αντικείμενα.

8.5 Λειτουργίες Μεταφοράς στο Χώρο

Επιπλέον οι λειτουργίες του προγράμματος θα πρέπει να είναι ευδιάκριτες και εύχρηστες για τον χρήστη διευκολύνοντας τον στις εργασίες που επιθυμεί να εκτελέσει, δίνοντας του πρόσβαση σε όλα τα αντικείμενα του τρισδιάστατου κόσμου καθώς και σε οποιοδήποτε σημείο του. Αυτό συμβαίνει προσφέροντας τις βασικές δυνατότητες στον χρήστη:

- της περιστροφής (rotation) της τάξεως των 360° σε οριζόντιο ή κάθετο άξονα.
- της μεταφοράς (translation) σε πλήρη κλίμακα σε όλες τις κατευθύνσεις (βοράς, νότος, ανατολή, δύση).
- του ζουμ (zoom) προσφέροντας τις δυνατότητες της σμίκρυνσης και της μεγέθυνσης μεταβάλλοντας την τιμή της τρίτης διάστασης του βάθους.

8.6 Διαχείριση Δεδομένων

Λάβαμε υπόψη μας ότι ο χρήστης θα πρέπει να έχει πλήρη πρόσβαση στην διαχείριση της πληροφορίας και στην αλληλεπίδραση του με τα αντικείμενα που υπάρχουν στον τρισδιάστατο κόσμο μας.

Γι’ αυτό δόθηκαν στον χρήστη οι δυνατότητες της :

- εισαγωγής νέας πληροφορίας (επιλογή οντότητας ή σχέσης και εισαγωγή της στο τρισδιάστατο κόσμο μέσω ποντικιού και click event)
- διαγραφής παλαιάς (μέσω επιλογής οντότητας ή σχέσης και επιλογής Menu Item από το αναδυόμενο Popup Menu)
- μεταφοράς αντικειμένων (επιλογή οντότητας, επιλογή Cut Menu Item από το αναδυόμενο Popup Menu και επιλογή Paste Menu Item από το αναδυόμενο Popup Menu στο σημείο προορισμού)
- αντιγραφής οντοτήτων (επιλογή οντότητας, επιλογή Copy Menu Item από το αναδυόμενο Popup Menu και επιλογή Paste Menu Item από το αναδυόμενο Popup Menu στο σημείο προορισμού)

Και στο σύστημα δόθηκαν οι δυνατότητες :

- επιλογής οντοτήτων (αριστερό ή δεξί κλικ πάνω σε ένα αντικείμενο και βασιζόμενοι στο User View και Depth Selection)
- ελέγχου λαθών χρήστη (γίνεται έλεγχος των εντολών του χρήστη για συμβατότητα και σφάλματα)
- απόκρυψης ολοκληρωμένης πληροφορίας (εμφάνιση στο χρήστη μόνο του ακρώνυμου της οντότητας και του ID)
- παρουσίασης πλήρους πληροφορίας (πλήρη εμφάνιση όλης της πληροφορίας που αφορά οντότητες και σχέσεις, ID & Label)
- παρουσίασης ιεραρχίας οντοτήτων (αυτόματη εισαγωγή IsA σχέσεων ανάμεσα στις οντότητες στο 3D Canva)
- παρουσίασης ιεραρχίας σχέσεων (αυτόματη εισαγωγή IsA σχέσεων ανάμεσα στις σχέσεις οντοτήτων στο 3D Canva)

- αυτόματης εισαγωγής οντοτήτων (μέσω αυτόματης εισαγωγής σχέσεων οντοτήτων όπου υπολείπετε το Domain ή Range Entity, το οποίο και τελικώς εισάγεται αυτόματα)
- αυτόματης εισαγωγής σχέσεων (μέσω δεξιού κλικ πάνω σε μία σχέση της οποίας υπάρχει έστω ένα από τα Domain ή Range Entities)
- αυτόματης διαγραφής σχέσεων (κατά την διαγραφή μίας οντότητας διαγράφονται αυτόματα και οι σχέσεις που την συνοδεύουν)

Το πρόγραμμα προσπαθεί και είναι σε θέση να προσαρμοστεί στις επιλογές του χρήστη προσφέροντας του την πληροφορία και τις λειτουργίες που χρειάζεται την εκάστοτε φορά, με τέτοιο τρόπο ώστε να διευκολύνει τον χρήστη στο να φέρει εις πέρας την αποστολή του αλλά και να τον αποτρέψει από τυχόν ανθρώπινα λάθη. Για παράδειγμα στο χρήστη εμφανίζονται την εκάστοτε στιγμή μόνο οι σχέσεις που θα μπορούσαν να εισαχθούν βάση της παρούσας συγκυρίας αντικειμένων που υπάρχουν στον τρισδιάστατο κόσμο μας και υπολείπονται αυτές που δεν θα μπορούσαν να εισαχθούν. Επιπλέον το πρόγραμμα είναι σε θέση να εισάγει αυτόματα αντικείμενα και πληροφορία αν αυτό είναι δυνατό διευκολύνοντας έτσι τον χρήστη στις εργασίες του αλλά και χωρίς να χάνει τον έλεγχο του μοντέλου που προσπαθεί να δημιουργήσει.

8.7 Αυτόματη και μη Κίνηση

Ένα στοιχείο που έχουμε λάβει υπόψη μας είναι ότι η ανθρώπινη αντίληψη αντιδρά πιο γρήγορα σε ένα σχήμα που βρίσκεται σε κίνηση παρά σε μια στατική οντότητα. Αυτό πιθανότατα είναι κάτι που έχει κληρονομηθεί από τους προγόνους μας οι οποίοι θα έπρεπε να κυνηγήσουν την λεία τους. Χάρη σε αυτό αντικείμενα που κινούνται ή που εμφανίζονται από το πουθενά τραβάνε πιο εύκολα την προσοχή μας και έτσι επικεντρωνόμαστε σε αυτά. Γι’ αυτό δώσαμε την δυνατότητα στο χρήστη της πλήρης μεταφοράς του στον τρισδιάστατο κόσμο καθώς επίσης και της μετακίνησης όλων των αντικειμένων που ανήκουν σε αυτό σε οποιοδήποτε επιτρεπτό σημείο. Η μετακίνηση αυτή, των εικονικών αντικειμένων θεωρείται απαραίτητη σε περίπτωση που κάποιο από αυτά έχει εγκλωβιστεί πίσω από κάποιο άλλο και δεν είναι παρατηρήσιμο από τον χρήστη.

Επίσης τυχόν επιλογές που αφορούν τις οντότητες του 3D canva εμφανίζονται ως αναδυόμενα παράθυρα (Menu), χάρη στα οποία ο χρήστης μπορεί να δηλώσει την επιθυμία του.

Τέλος, προσπαθώντας να επιτύχουμε βέλτιστη ποιότητα και ρεαλιστικότητα επιτύχαμε την αυτόματη κίνηση σε πραγματικό χρόνο όλων των οντοτήτων που σχετίζονται με το αντικείμενο που μετακινούμε, χωρίς την παρέμβαση του χρήστη.

8.8 Θέση Αντικειμένων

Παρά πέρα η ίδια η θέση που καταλαμβάνει μια οντότητα μπορεί να ορίσει την ποσότητα της προσοχής που θα λάβει αυτή η οντότητα από τον χρήστη. Συνήθως αντικείμενα που έχουν τοποθετηθεί κεντρικά είναι σε θέση να λάβουν περισσότερη προσοχή από το χρήστη σε σχέση με αντικείμενα που έχουν τοποθετηθεί περιφερειακά. Επίσης οντότητες που βρίσκονται κοντά σε μια οντότητα που έχουμε επικεντρώσει την προσοχή μας πιθανότατα θα λάβουν περισσότερη προσοχή από άλλες οντότητες που βρίσκονται σε μεγαλύτερη απόσταση.

Επίσης η θέση στην οποία έχει εισάγει ο χρήστης τα αντικείμενα λαμβάνεται υπόψη κατά την αυτόματη εισαγωγή Properties και IsA σχέσεων. Έτσι επιλέχθηκε να γίνει χρήση του κάθετου άξονα για να εισαχθούν οι σχέσεις IsA ούτως ώστε να αναπτυχθεί ένα δέντρο κληρονομικότητας στον άξονα αυτόν εισάγοντας τον γονέα επάνω και τον απόγονο κάτω.

Για την εισαγωγή των Properties προτιμήθηκε μια ανάπτυξη τους στον οριζόντιο άξονα. Ενδιαφέρον αποτελεί το γεγονός ότι ανάμεσα σε δύο οντότητες θα μπορούσαν να υπάρξουν πολλαπλά Properties και επίσης θα μπορούσαν να υπάρξουν Properties όπου το Domain και Range Entities θα μπορούσαν να είναι η ίδια οντότητα. Στην περίπτωση αυτή οι επιπλέον σχέσεις ανάμεσα στις οντότητες τοποθετούνται πιο βαθιά στον χώρο ή σε διαφορετικό ύψος με σεβασμό στην αποφυγή επικάλυψης ανάμεσα στα τρισδιάστατα αντικείμενα.

Με αυτόν τον τρόπο οδηγούμαστε στην δημιουργία μιας οπτικής γλώσσας παρουσίασης πληροφορίας η οποία είναι σε θέση να παρουσιάσει γρήγορα και εύκολα την ιεραρχία των οντοτήτων στον χρήστη.

8.9 Μέγεθος Αντικειμένων

Τέλος το ίδιο το σχήμα και το μέγεθος που χαρακτηρίζει μια οντότητα μπορεί να προσδώσει σε αυτήν γνωρίσματα και κύρος που θα διευκολύνουν τον χρήστη στην αναγνώριση της. Για παράδειγμα ένας κύβος με διπλάσιο όγκο από έναν άλλο πιθανότητα θα θεωρηθεί πιο σημαντικός και θα είναι πιο ευδιάκριτος από τον συγγενή του εξαιτίας αυτού του γνωρίσματος. Σε αυτό το σημείο θα πρέπει να διευκρινίσουμε

ότι το μέγεθος στον τρισδιάστατο κόσμο είναι κάτι το υποκειμενικό και εξαρτάται απ' την απόσταση του αντικειμένου από τον παρατηρητή και συχνά οδηγεί σε λάθος συμπεράσματα καθώς αντικείμενα που έχουν διαφορετικό μέγεθος μπορούν να παρουσιαστούν στον παρατηρητή ως ισομεγέθη αν τοποθετηθούν σε διαφορετικό βάθος.

8.10 Πλήθος Αντικειμένων

Η ίδια η ποσότητα των δεδομένων και το πλήθος των αντικειμένων μπορεί να παίξει ουσιαστικό ρόλο στην αναπαράσταση τους καθώς όταν το πλήθος τους αυξηθεί σημαντικά γίνεται πιο δύσκολη η αποτύπωση τους και η αναγνώριση τους από τον άνθρωπο. Αυτό το πρόβλημα είναι πολύ χαρακτηριστικό όταν χρησιμοποιούνται δυσδιάστατα γραφικά στοιχεία αφού όλα τα αντικείμενα έχουν το ίδιο μέγεθος και ο χώρος της οθόνης είναι συγκεκριμένος και περιορισμένος. Οι υπάρχουσες εφαρμογές προσπαθούν να υπερκεράσουν αυτές τις δυσκολίες κρύβοντας, διαγράφοντας ή ομαδοποιώντας πληροφορία.

Τα τρισδιάστατα γραφικά μπορούν να υπερβούν το συγκεκριμένο πρόβλημα καθώς το μέγεθος των αντικειμένων εξαρτάται από τη θέση τους στο χώρο και διατίθεται απεριόριστο βάθος. Επίσης στον εικονικό κόσμο μπορούν στην ίδια απόσταση να εισαχθεί μεγαλύτερο πλήθος αντικειμένων σε διαφορετικά επίπεδα βάθους. Πιο αναλυτικά αντικείμενα που δεν χαρακτηρίζονται ως σημαντικά , που περικλείονται σε άλλα ή που θα μπορούσαν να αγνοηθούν είναι δυνατόν να εισαχθούν σε θέσεις που βρίσκονται πιο βαθειά στον εικονικό κόσμο ή πίσω από άλλα πιο σημαντικά αντικείμενα προσδίδοντας έτσι μια μορφή ιεραρχίας στο όλο τρισδιάστατο σχήμα. Έρευνες έχουν δείξει ότι η αντίληψη του ανθρωπίνου εγκεφάλου συσχετίζει αντικείμενα που βρίσκονται κοντά, κάτω ή πίσω με κάποιο άλλο αντικείμενο σχηματίζοντας νοητές ομάδες ή ιεραρχίες. Για παράδειγμα αν ένα αντικείμενο είναι κάτω από κάποιο άλλο πιθανότατα σχετίζεται με αυτό με κάποια σχέση isA (γονέας, παιδί).

Αυτό όμως έχει και αρνητικές επιπτώσεις καθώς αντικείμενα που διαθέτουν μικρότερο μέγεθος αναγνωρίζονται πιο δύσκολα από τον χρήστη και επίσης είναι πιο δύσκολο να αντιληφθούμε την πληροφορία που αυτά φέρουν. Επίσης ενδέχεται να υπάρχει απόκρυψη οντοτήτων από άλλες. Γι' αυτό, όπως θα δούμε παρακάτω, αναπτύξαμε μια δυναμική λειτουργία από Info Labels τα οποία είναι σε θέση να βοηθήσουν τον χρήστη να αντιληφθεί πλήρως την πληροφορία που τα αντικείμενα φέρουν.

Επιπλέον, θέλοντας να συρρικνώσουμε το πλήθος των αντικειμένων που τοποθετούνται στον εικονικό κόσμο εισάγουμε αποκλειστικά τις οντότητες που

επιθυμεί ο χρήστης και μόνο τις σχέσεις Isa που συνδέουν άμεσα τον πρόγονο με τον απόγονο.

8.11 Αυτόματο Layout

Το θέμα της αποτελεσματικής διάταξης των οντοτήτων στην οθόνη είναι ένα από τα πιο σημαντικά στοιχεία που αφορούν την αναπαράσταση πληροφορίας με οπτικό τρόπο [29], [54]. Λέγοντας διάταξη οντοτήτων αναφερόμαστε στον υπολογισμό της θέσης και του μεγέθους κάθε αντικειμένου που παρουσιάζεται στο χρήστη καθώς επίσης και στο αποτέλεσμα αυτού, πράγμα που σημαίνει την παρουσίαση των οντοτήτων στον χρήστη. Η πληθώρα αλγορίθμων διάταξης οντοτήτων προς παρουσίαση πληροφορίας στις μέρες μας βασίζονται πάνω στις οδηγίες του ανθρώπου για τον ορισμό της θέσης που θα καταλάβουν τα αντικείμενα. Έτσι το περισσότερο βάρος της σχεδίασης πέφτει στους ώμους του σχεδιαστή και σε πολλά συστήματα αυτός είναι υπεύθυνος για την πλήρη διάταξη που θα χρησιμοποιηθεί, αφήνοντας έτσι το σύστημα απλά να εκτελεί τις εντολές που δίνει. Η διαδικασία αυτή χαρακτηρίζεται ως χρονοβόρα και πολλές φορές οι σχεδιαστές ξοδεύουν πολύ χρόνο μαθαίνοντας και εξελίσσοντας τις σχεδιακές τους ικανότητες πάνω στην δημιουργία εύχρηστων, ευέλικτων και ίσως βέλτιστων διατάξεων. Ενδέχεται πολλές φορές να χρειαστούν πολλές ώρες προκειμένου να παραχθούν πολύ απλές διατάξεις και δεδομένου της τεράστιας έκρηξης πληροφορίας που παρατηρούμε στη σύγχρονη εποχή αντιλαμβανόμαστε την ανάγκη ύπαρξης ενός μέσου αυτόματης διάταξης οντοτήτων.

Η αυτόματη διάταξη αποτελεί δύσκολο έργο για πολλούς λόγους. Τα περισσότερα συστήματα τείνουν να κάνουν βέλτιστη χρήση του χώρου της οθόνης και αφήνουν σε δεύτερη μοίρα την οπτική παρουσία και χρηστικότητα της σχεδίασης αυτής. Επίσης ο υπολογισμός βέλτιστης θέσης απορροφά πολλούς επεξεργαστικούς πόρους και παρ' ότι η ταχύτητα των επεξεργαστών έχει βελτιωθεί σημαντικά, παραμένει ένα σοβαρό μειονέκτημα της σχεδιαστικής αρχής αυτής. Γι' αυτό πιστεύουμε πως ο παράγοντας άνθρωπος δεν μπορεί να αγνοηθεί πλήρως και να οδηγηθούμε σε ένα καθ' όλα αυτόματο σύστημα διάταξης οντοτήτων στο χώρο.

Υπάρχουσες εφαρμογές βασίζονται πάνω και σε άλλες τεχνολογίες όπως γλώσσες προγραμματισμού για την περιγραφή στοίχισης, τεχνικές μηχανικής μάθησης [49], [50] κτλ για την εξέλιξη και την επιτυχής αυτόματη αποτύπωση οντοτήτων στην οθόνη. Για παράδειγμα αν το σύστημα θα μπορούσε να μαντέψει τις επιλογές του χρήστη από προηγούμενα παραδείγματα που είχε φέρει σε πέρας ο χρήστης αυτός, τότε δε θα χρειαζόταν κάθε φορά να οριστούν οι επιλογές αυτές από τον χρήστη.

Ορισμένοι αλγόριθμοι αυτόματης στοίχισης οντοτήτων είναι οι :

8.11.1 Ορθογώνιο (Orthogonal Layouts)

Στα διαγράμματα ορθογώνιας στοίχισης η οργάνωση της πληροφορίας βασίζεται σε οριζόντια ή κάθετα ορθογώνια τμήματα αυτό επιτρέπει στους χρήστες να δομήσουν την παρουσίαση με τρόπο κατάλληλο ώστε να γίνει αποφυγή επικαλύψεων και επίσης βελτιώνει την αναγνωσιμότητα και μειώνει την ανάγκη περεταίρω μελλοντικών αναγκών. Είναι κατάλληλο για μικρού ή μεσαίου μεγέθους διαγράμματα καθώς επίσης παράγει ολοκληρωμένο και ξεκάθαρο αποτέλεσμα στοίχισης. Μπορεί να οργανώσει τις οντότητες σε ομάδες πράγμα που είναι πολύ σημαντικό στην επεξεργασία και ανάκτηση πληροφορίας. Μια ομάδα δηλώνει ένα σύνολο από οντότητες οι οποίες θα πρέπει να τοποθετηθούν σε ένα συγκεκριμένο κομμάτι της επιφάνειας παρατήρησης του χρήστη, κάνοντας το έτσι εύκολα φανερό και προσβάσιμο.

Η τεχνική αυτή υιοθετείται και από τα διαγράμματα UML τα οποία μπορούν να παρουσιάσουν ιεραρχική και μη ιεραρχική πληροφορία και τα οποία επιπλέον μπορούν να μειώσουν τις αλληλοεπικαλύψεις, το μέγεθος του συνόλου του χώρου που θα χρειαστεί, να παρουσιάσουν έμμεσα πληροφορία βάση χρωματισμού κτλ.

Τα διαγράμματα αυτά είναι σε θέση να επιλύσουν πολύπλοκα μαθηματικά και γενικός αλγοριθμικά προβλήματα. Κρίνονται ιδανικά για την οπτικοποίηση βάσεων δεδομένων, διαγραμμάτων οντοτήτων-σχέσεων και πολλών άλλων τεχνικών αποθήκευσης πληροφορίας.

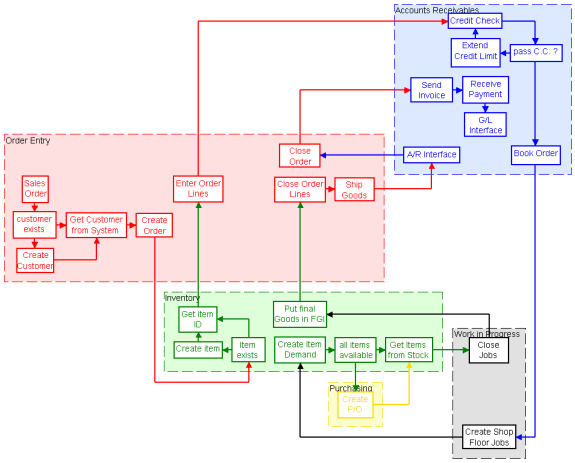


Figure 9. Ορθογώνια ομαδοποίηση (Orthogonal Cluster Layout)

8.11.2 Ιεραρχικό (Hierarchical Layouts)

Η τεχνική αυτή απεικόνισης εφαρμόζεται συνήθως όταν η δομή της αναπαράστασης της πληροφορίας θα πρέπει να αντανακλά την ύπαρξη ενός συστήματος διαβάθμισης ανάμεσα στις οντότητες. Θεωρείται κατάλληλη για συστήματα ροών δεδομένων καθώς επίσης για συστήματα επεξεργασίας γεγονότων και αναπαράσταση κληρονομικότητας και ιεραρχίας.

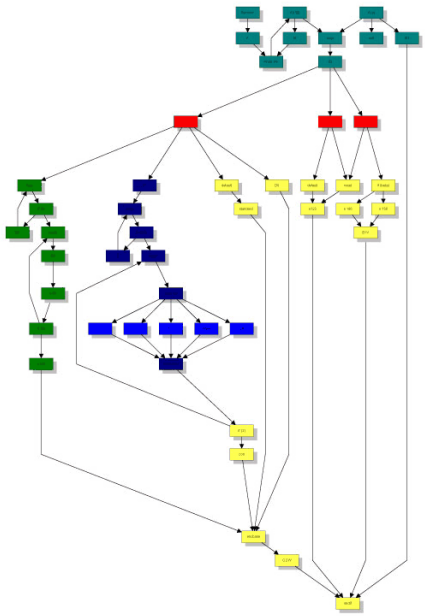


Figure 10. Ιεραρχικό (Hierarchical Layout Technique)

8.11.3 Συμμετρικό (Symmetric Layout)

Η τεχνική αυτή θεωρείται η κατάλληλη για την οπτικοποίηση μεγάλων και πολύπλοκων δικτύων και γράφων [56], [57]. Σημαντικό παράγοντα αποτελεί η πολυπλοκότητα της δομής που παρουσιάζεται στο χρήστη, η αναγνωσιμότητα του, η δυνατότητα παρουσίασης της πλήρους διαθέσιμης πληροφορίας και η κατανόηση του από το χρήστη.

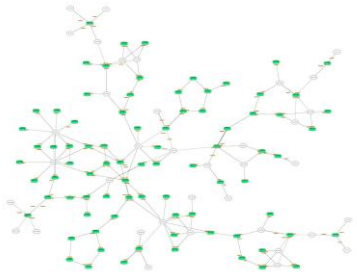


Figure 11. Συμμετρικό (Symmetric Layout Technique)

8.11.4 Κυκλικό (Circular Layout)

Ικανοποιεί τις ανάγκες και τις απαιτήσεις αποτύπωσης της δομής των τηλεπικοινωνιακών δικτύων. Η αναπαράσταση αυτή γίνεται γρήγορα και αποτελεσματικά παρέχοντας πληροφορία στο χρήστη που αφορά την ιεραρχία των δικτύων, την σύνδεση μεταξύ τους καθώς επίσης και την τοπολογία τους. Χάρις αυτά ο χρήστης είναι σε θέση να αντιλήφθη γρήγορα και εύκολα τυχόν προβλήματα αλλά επίσης και να ανακαλύψει και προτείνει βελτιστοποίησης που θα ανεβάσουν την γενικότερη απόδοση του δικτύου.

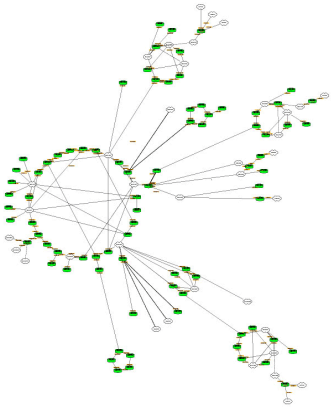


Figure 12. Κυκλικό (Circular Layout Technique)

8.11.5 Δενδρικό (Tree Layout)

Πρόκειται ίσως για τη συνηθέστερη μορφή αυτόματης απεικόνισης. Είναι ένας κατευθυνόμενος ά κύκλος γράφος όπου κάθε κόμβος έχει μηδέν, ένα ή περισσότερα παιδιά και το πολύ έναν γονέα [59]. Ο κόμβος που δεν έχει γονέα καλείται ρίζα και είναι μοναδικός. Κάθε δέντρο χαρακτηρίζετε από ένα συγκεκριμένο ύψος και από ένα πλάτος, όπου ύψος εστί μεγαλύτερη διαδρομή από τη ρίζα έως κάποιο φύλλο και πλάτος ο μέγιστος αριθμός των φύλλων. Παρότι υπάρχουν πολλοί τρόποι

αναπαράστασης δενδρικής μορφής συνηθίζεται να παρουσιάζονται οι κόμβοι ως οντότητες με συνδέσεις προς τα παιδιά τους.

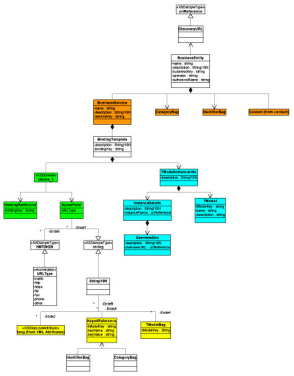


Figure 13. Δενδρικό (Tree Layout Technique)

8.11.6 Αξιολόγηση Αυτόματου Layout

Πολλοί αλγόριθμοι αυτόματης διάταξης οντοτήτων έχουν προταθεί [55]. Παρόλα αυτά δεν υπάρχει ακόμα κάποιος βέλτιστος αλγόριθμος που μπορεί να ανταπεξέλθει στις ανάγκες των χρηστών. Ο αλγόριθμος αυτός θα πρέπει να:

- Διατηρεί εμφανισιακά την ιεραρχία των οντοτήτων.
- Ομαδοποιεί οντότητες με κοινά χαρακτηριστικά.
- Τροποποιεί τη διάταξη ανάλογα με τις επιλογές και τις ανάγκες του χρήστη [52], [53].
- Μην αλλάζει δραματικά την όλη δομή του σχήματος σε ανθρώπινες μικροεπεμβάσεις.
- Επιλέγει σωστό μέγεθος, θέση και εμφανισιακά χαρακτηριστικά για την κάθε οντότητα.
- Υπολογίζει τη βέλτιστη απόσταση ανάμεσα σε αντικείμενα.
- Μπορεί να ανταποκριθεί σε μεγάλο πλήθος δεδομένων.
- Μην υπάρχουν αντικείμενα που επικαλύπτονται.
- Μαθαίνει από το χρήστη και συνεχώς βελτιώνεται [51].

- Έχει ως βασικό γνώμονα την δυνατότητα αντίληψης του ανθρώπου και όχι απλά την παρουσίαση πληροφορίας [58].

Δυστυχώς τα υπάρχοντα συστήματα αδυνατούν να ανταπεξέλθουν σε όλες αυτές τις ανάγκες και η παρέμβαση των χρηστών στην διάταξη των οντοτήτων κρίνεται συνήθως απαραίτητη. Επίσης η αυτόματη διάταξη μπορεί να οδηγήσει σε αποδιοργάνωση της οπτικής αντίληψης και έτσι να δυσκολέψει το έργο της κατανόησης της ιεραρχίας των δεδομένων. Πέραν τούτου ο αυτόματος υπολογισμός θέσης για όλες τις οντότητες ενός σχήματος βάση των περισσότερων υπαρχόντων αλγορίθμων απαιτεί μεγάλη επεξεργαστική ισχύ και πολλές φορές αυτή η ισχύς κρίνεται απαγορευτική για προσωπικούς υπολογιστές και ειδικά όταν απαιτείτε αυτόματος υπολογισμός θέσης σε πραγματικό χρόνο.

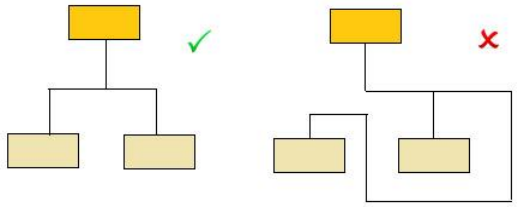


Figure 14. Λάθος αυτόματο Layout

8.12 Σημερινές Τεχνολογίες

Η τεχνολογία των υπολογιστών είναι σε θέση σήμερα όχι απλά να επιτρέψει την αποτύπωση ενός τρισδιάστατου κόσμου σε μια δυσδιάστατη οθόνη αλλά και να δώσει τη δυνατότητα στον χρήστη να αλληλεπιδράσει με αυτό το εικονικό περιβάλλον. Έρευνες για τρισδιάστατα γραφικά σε ηλεκτρονικούς υπολογιστές έχουν δείξει ότι η τεχνολογία αυτή βελτιώνει την αντίληψη των χρηστών όταν αυτοί μπορούν να διαχειριστούν-αλληλεπιδράσουν με το εικονικό περιβάλλον [2]. Η αλληλεπίδραση μπορεί να αρκεστεί στην πλοήγηση στο τρισδιάστατο περιβάλλον, δηλαδή ο χρήστης να μπορεί να μεταφερθεί σε οποιοδήποτε σημείο του τρισδιάστατου χώρου αλλά και να μπορεί να μεταβάλλει με εύκολο τρόπο τη θέση από την οποία βλέπει ο παρατηρητής. Επιπλέον η εφαρμογή θα μπορούσε να επιτρέπει στον χρήστη να αλληλεπιδράσει σε πραγματικό χρόνο με τα αντικείμενα που του παρουσιάζονται, δηλαδή θα μπορούσε να εισάγει, διαγράψει, μεταφέρει και να τροποποιήσει αντικείμενα του χώρου. Κάνοντας χρήση δυσδιάστατου γραφικού περιβάλλοντος χρήστη η εφαρμογή θα μπορούσε να δώσει τη δυνατότητα στο χρήστη να χειριστεί

ιδιότητες και λειτουργίες του τρισδιάστατου χώρου χρησιμοποιώντας κοινά δυσδιάστατα αντικείμενα ελέγχου της java. Για παράδειγμα οι εντολές περιστροφής του τρισδιάστατου κόσμου θα μπορούσαν να ελέγχονται από κλασικά δυσδιάστατα κουμπιά.

8.13 Διεπαφή 2D

Στην προσπάθεια μας να κάνουμε πιο εύχρηστο το σύστημα για τον χρήστη και να ανεξαρτητοποιήσουμε όσο το δυνατό περισσότερο τον τρισδιάστατο κόσμο από τον δυσδιάστατο δίνουμε την δυνατότητα στον χρήστη να διαχειριστεί τον 3D κόσμο απευθείας μέσω του ποντικιού [4]. Για παράδειγμα ο χρήστης μπορεί να κάνει μεγέθυνση (zoom in) και σμίκρυνση (zoom out) μέσω της ροδέλας του ποντικιού. Η απόκριση του συστήματος σε αλλαγές από τον χρήστη θεωρείται άμεση και ο χρόνος αναμονής του χρήστη αμελητέος, δίχως αυτό να συνεπάγεται χαμηλότερη ποιότητα των γραφικών στοιχείων ή δυνατοτήτων. Δηλαδή η ποιότητα της οπτικής παρουσίασης της πληροφορίας παραμένει υψηλή και σύγχρονος το σύστημα εγγυάται γρήγορη απόκριση.

Στην προσπάθεια μας δημιουργίας μιας διεπαφής όσο το δυνατό πιο εύχρηστης και στην οποία οι χρήστες θα έχουν ήδη εξοικειωθεί, επιλέξαμε να χρησιμοποιήσουμε μονό βασικά στοιχεία που εξυπηρετούν την αλληλεπίδραση του ανθρώπου με τη μηχανή [3]. Γι' αυτό επιλέχθηκαν ως στοιχεία αλληλεπίδρασης με τον χρήστη κουμπιά, αναδυόμενα μενού, εντολές ποντικιού κτλ. Οι ίδιες οι οντότητες του τρισδιάστατου κόσμου έχουν μέγεθος επαρκές ώστε να είναι ευδιάκριτα στο χρήστη αλλά και να μπορούν να επιλεγούν ή να μετατραπούν εύκολα και γρήγορα.

8.14 3D Σχήματα

Σχεδόν βέλτιστη αποτύπωση της πληροφορίας θα είχαμε αν το ίδιο το σχήμα των τρισδιάστατων αντικειμένων υπονοούσε και το νόημα τους, ώστε να γίνει χρήση μεταφορών και ο χρήστης παρατηρώντας τα απλά να μπορούσε να αντιληφθεί και το νόημα τους. Για παράδειγμα αν θέλαμε να εισάγουμε την οντότητα άνθρωπος στο σχήμα μας, εκεί να εμφανιζόταν ένα εικονίδιο που θα παρέπεμπε άμα τη εμφανίσει στην οντότητα μας χάρις το σχήμα του. Αυτό πρώτον θα ήταν πρακτικά αδύνατον αφού τα αντικείμενα και οι έννοιες που υπάρχουν στον κόσμο μας είναι άπειρα και οποιαδήποτε στιγμή γέννιουνται καινούργια και καταστρέφονται παλιά. Δεύτερον δεν είμαστε σε θέση να δώσουμε μορφή σε όλες τις έννοιες που να παραπέμπουν σε αυτές. Για παράδειγμα πια θα μπορούσε να είναι η μορφή που θα δίναμε στη χαρά, ευτυχία κτλ. Οι μεταφορές αυτές θα μπορούσαν να είναι δυσνόητες ή και να

παραπέμπουν σε λάθος συμπεράσματα. Για παράδειγμα το κόκκινο χρώμα συνήθως εκφράζει το απαγορευμένο και το μη επιτρεπτό, αλλά σε ορισμένες κουλτούρες συμβολίζει τη χαρά και το ευ ζην.

8.15 3D Επικάλυψη

Ο εντοπισμός συγκρούσεων και μάλιστα σε πραγματικό χρόνο είναι συχνά η πλέον εξαντλητική διαδικασία για τους επεξεργαστικούς πόρους του συστήματος στις περισσότερες εφαρμογές διαχείρισης και απεικόνισης γεωμετρικών σχημάτων [26]. Δεδομένου αυτού θα πρέπει να είμαστε προσεκτικοί στην εφαρμογή αλγορίθμων εντοπισμού συγκρούσεων καθώς και η παραμικρή βελτιστοποίηση μπορεί να προσφέρει σημαντικά στην απόδοση και ανταπόκριση της υπηρεσίας μας. Έχοντας δύο ή περισσότερα γεωμετρικά σχήματα σε συγκεκριμένα σημεία στο χώρο και σαφή διαστάσεις, στόχο αποτελεί ο έλεγχος αν αυτά τα αντικείμενα επικαλύπτονται σε κάποια χρονική στιγμή. Πρόκειται για μια υποπερίπτωση ελέγχου απόστασης ανάμεσα σε οντότητες του τρισδιάστατου κόσμου όπου η απόσταση μπορεί να ισούται με το μηδέν αλλά και να παίρνει αρνητικές τιμές αν οι οντότητες βρίσκονται εν μέρει ή εξολοκλήρου η μια μέσα στην άλλη [24],[25]. Σε μια εφαρμογή αντιστοίχησης πληροφορίας η επικάλυψη οντοτήτων θα πρέπει να αποφεύγεται αφού συνεπάγεται μερική ή και ολική απόκρυψη οντοτήτων καθώς επίσης δυσχεραίνει το έργο που έχει αναλάβει ο χρήστης να φέρει εις πέρας. Παραπέρα η δυνατότητα ρεαλιστικής αναπαράστασης τρισδιάστατων αντικειμένων δέχεται ισχυρό πλήγμα σε περίπτωση που μια οντότητα συνυπάρχει με μια άλλη στο ίδιο σημείο του τρισδιάστατου χώρου.

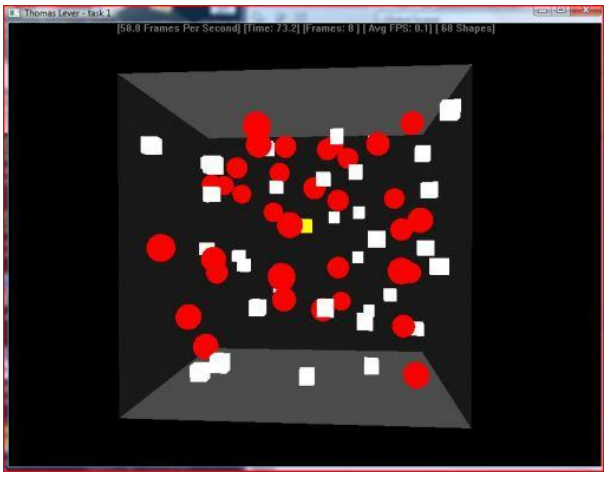


Figure 15. Αναγνώριση επικάλυψης

9 Θέματα προς Έλεγχο

Οι προβληματισμοί που μας απασχολούν στην τρισδιάστατη αναπαράσταση πληροφορίας είναι αρκετοί και ποικίλοι. Μπορούν όμως να καταταχτούν σε τέσσερις βασικές κατηγορίες με γνώμονα τον τρόπο αποτύπωσης πληροφορίας, το μέγεθος της πληροφορίας που παρουσιάζεται και οι ενέργειες που καλείται ο χρήστης να εκτελέσει [23].

- Πλήρη πληροφορία

  Σε αυτή την κατηγορία θα πρέπει να γίνει αποτύπωση όλης της ποσότητας της πληροφορίας. Αυτό πολλές φόρες κρίνεται από δύσκολο έως ακατόρθωτο έχοντας κάνει χρήση ακόμα και της τρίτης διάστασης του βάθους αφού η ποσότητα των δεδομένων μπορεί να είναι τέτοια ώστε να κριθεί ως υπερβολικά μεγάλη για τον ανθρώπινο χρήστη. Ας μην ξεχνάμε ότι στις τρεις διαστάσεις μπορεί να υπάρξει επικάλυψη οντότητας από μιας άλλης αν αυτή βρίσκεται ανάμεσα στην πρώτη οντότητα και το χρήστη, δηλαδή πάνω στην νοητή γραμμή που υπάρχει από την πρώτη οντότητα έως τον χρήστη .

- Μερική πληροφορία

  Στην περίπτωση αυτή γίνεται αποτύπωση μόνο της απαραίτητης πληροφορίας για τον χρήστη. Σε αυτή την περίπτωση θα πρέπει να παρουσιαστεί η πληροφορία με τρόπο κατάλληλο ώστε να είναι όσο το δυνατό πιο ευδιάκριτη και εύκολη στο χειρισμό. Πληροφορία που κρίνεται ως σημαντική θα πρέπει να εμφανίζεται κοντά στον χρήστη και δίχως εμπόδια μπροστά της. Η δεύτερη αυτή κατηγορία θα πρέπει να συνεργάζεται με την πρώτη καθώς για παράδειγμα ακόμα και όταν γίνεται πλήρης αποτύπωση πληροφορίας τα αντικείμενα που επεξεργάζεται ο χρήστης θα πρέπει να είναι όσο το δυνατό πιο ευδιάκριτα σε αυτόν.

- Παρουσίαση πληροφορίας

  Ο χρήστης διαισθητικά θα πρέπει να αντιλαμβάνεται τα αντικείμενα που του παρουσιάζονται καθώς και τις σχέσεις ανάμεσα τους. Τεχνικές προσήλωσης της προσοχής του χρήστη θα πρέπει να χρησιμοποιηθούν. Οποιαδήποτε στιγμή θα πρέπει να παρουσιάζεται η μεγαλύτερη δυνατή ποσότητα πληροφορίας και να αυτό είναι εφικτό να μην υπάρξει κανένα κρυμμένο αντικείμενο στον τρισδιάστατο χώρο. Ο χρήστης θα πρέπει να μπορεί να μεταβεί σε οποιοδήποτε σημείο του χώρου και η γεωμετρία

των οντοτήτων που χρησιμοποιούνται να έχει επιλεχθεί με σωστά κριτήρια.

- Λειτουργίες και τεχνικές

  Οι βασικές λειτουργίες και τεχνικές του συστήματος πρέπει να είναι εμφανής και προσιτές στον χρήστη. Αυτός θα πρέπει να είναι σε θέση να φέρει εις πέρας εύκολα και γρήγορα την αποστολή του. Το σύστημα θα πρέπει να είναι σε θέση να εκτελεί αυτόματα λειτουργίες, να βοηθάει τον χρήστη όταν είναι εφικτό και να αποτρέπει τυχόν ανθρώπινα λάθη.

10 Τεχνικές Έμφασης

Για να προσελκύσουμε το ενδιαφέρον του χρήστη σε συγκεκριμένες οντότητες του τρισδιάστατου χώρου μπορούμε να εξασκήσουμε ορισμένες τεχνικές τις οποίες και παρουσιάζουμε.

- Αποτύπωση σημαντικών οντοτήτων στο κέντρο της οθόνης καθώς έρευνες έχουν δείξει ότι η προσοχή των χρηστών επικεντρώνεται περισσότερο σε αυτό το σημείο. Οι υπόλοιπες οντότητες μπορούν να αποτυπωθούν περιφερειακά των σημαντικών.

- Αποτύπωση μόνο των οντοτήτων που θέλουμε να επικεντρωθεί η προσοχή του χρήστη πάνω τους. Οι υπόλοιπες οντότητες μπορούν είτε να μην παρουσιαστούν είτε να αποτυπωθούν μερικώς ως πολύ μικρές ή ημιδιαφανείς οντότητες.

- Οντότητες ιδιαίτερης σημασίας μπορούν να τονιστούν δίνοντας έμφαση στο μέγεθος τους αποκτώντας μεγαλύτερο ή αλλάζοντας το σχήμα τους όπως και τη θέση τους.

- Πιο απλές τεχνικές αλλά όχι και αμελητέες θα μπορούσαν να διαμορφώσουν κατάλληλα το χρώμα, την σκίαση κτλ.

- Απόδοση κίνησης στις οντότητες που θέλουμε να τονίσουμε καθώς το ανθρώπινο μάτι έχει εκπαιδευτεί να εντοπίζει αντικείμενα που κινούνται.

11 3D Τεχνικές Οπτοκοποίησης Πληροφορίας

Πολλές συζητήσεις έχουν γίνει γύρω από το θέμα της βέλτιστης αποτύπωσης πληροφορίας σε έναν τρισδιάστατο εικονικό κόσμο δείχνοντας τον απαραίτητο σεβασμό στις ανάγκες των χρηστών και πολλοί ερευνητές έχουν ασχοληθεί με τον τομέα αυτό [5],[6]. Η αναζήτηση αυτή έχει καταλήξει σε ορισμένες βασικές μορφές σχηματισμού τρισδιάστατων δομών, τις οποίες θα αναφέρουμε πιο κάτω. Όλες αυτές οι τεχνικές δείχνουν να ανταπεξέρχονται στις απαιτήσεις των χρηστών και έχουν χωριστοί με γνώμονα τον τρόπο παρουσίασης της πληροφορίας, τα αντικείμενα που χρησιμοποιούνται, τις τεχνικές που υιοθετούν και τις λειτουργίες που προσφέρουν.

11.1 Στιλ σχεδίασης οντοτήτων-συνδέσεων (Node-Link Style Designs)

Τα διαγράμματα οντοτήτων σχέσεων έχουν χρησιμοποιηθεί κυρίως στην αναπαράσταση γράφων [7],[8]. Ασχολούνται ως επί των πλείστων με θέματα που αφορούν τον τρόπο παρουσίασης της πληροφορίας, της ελαχιστοποίησης συχνότητας συγκρούσεων, της ελαχιστοποίησης της μέγιστης απόστασης μεταξύ οντοτήτων και συμμετρίας σχημάτων. Το βασικό πρόβλημα που αντιμετωπίζει αυτό το μοντέλο εφάπτεται στο ότι από τη στιγμή που το πλήθος των οντοτήτων υπερβεί μια ποσότητα ο γράφος γίνεται δυσνόητος και δύσκολος στον χειρισμό του καθώς οντότητες και γραμμές τείνουν να επικαλύπτονται.

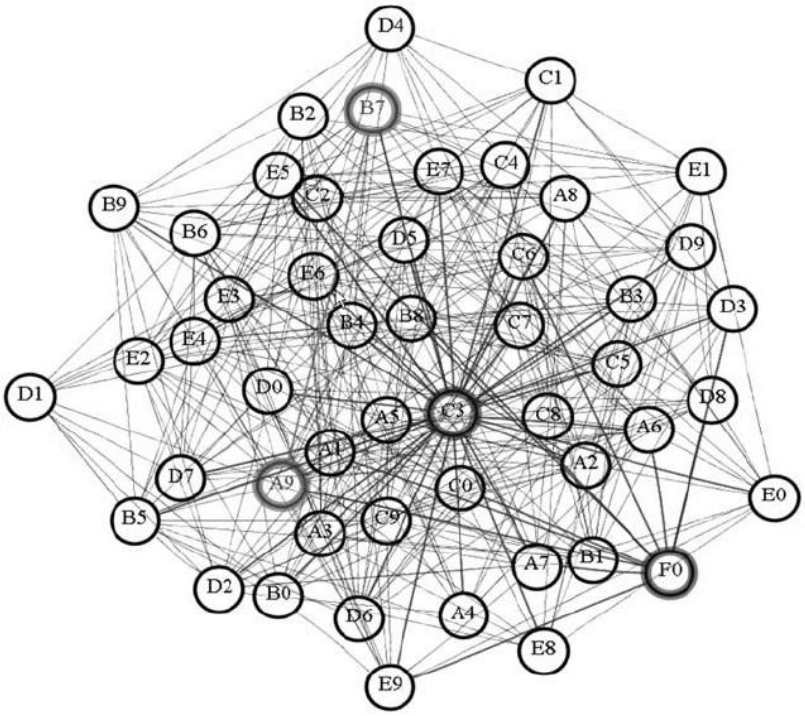


Figure 16. Layout απλού γράφου

Σε αυτή την περίπτωση η επικάλυψη τείνει να γίνει κανόνας και όχι υποπερίπτωση. Έτσι ο χρήστης δυσκολεύεται να αντιληφθεί την πληροφορία και χάνεται στην πολυπλοκότητα του σχήματος. Πλεονέκτημα του τρισδιάστατου μοντέλου αποτελεί το γεγονός του ότι προσφέρεται η τρίτη διάσταση του βάθους και έτσι είμαστε σε θέση να εισάγουμε περισσότερες οντότητες και σχέσεις μεταξύ τους καθώς αυτές μπορούν να τοποθετηθούν σε διαφορετικό μέγεθος βάθους έτσι η πιθανότητα επικάλυψης μικραίνει σε σχέση με την αναπαράσταση οντοτήτων σε δυσδιάστατα γραφικά.

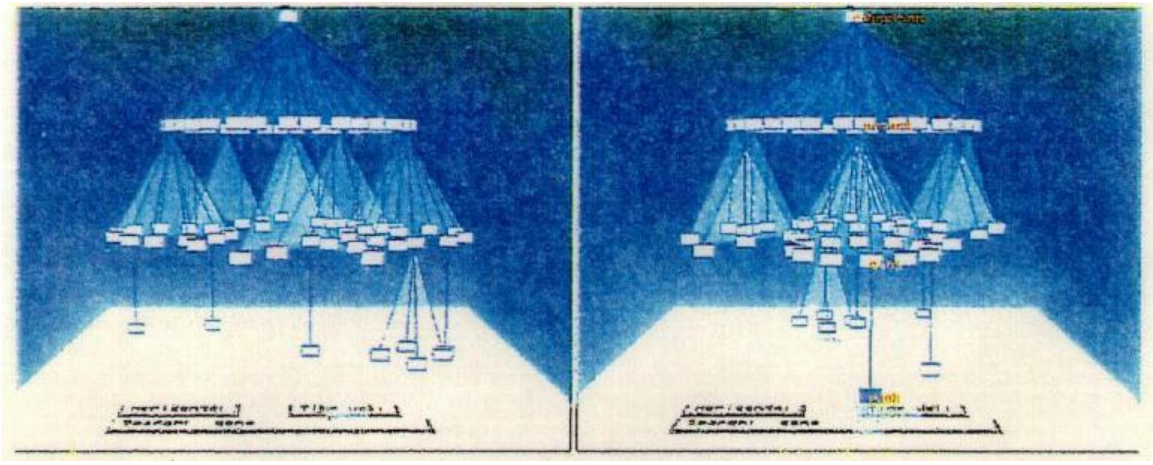


Figure 17. Layout κωνικού δέντρου

11.2 Επιφανειακή σχεδίαση (Surface Designs)

Η βασική σχεδίαση αναπτύσσεται σε επιφάνειες σε οποιαδήποτε κατεύθυνση. Γίνεται χρήση ευθειών ή καμπύλων και επίσης το μέγεθος των αντικειμένων τονίζει τη σημασία τους καθώς σημαντικά αντικείμενα εμφανίζονται μεγαλύτερα σε μέγεθος ή πλησιέστερα στον χρήστη χάρις στα τρισδιάστατα γραφικά.

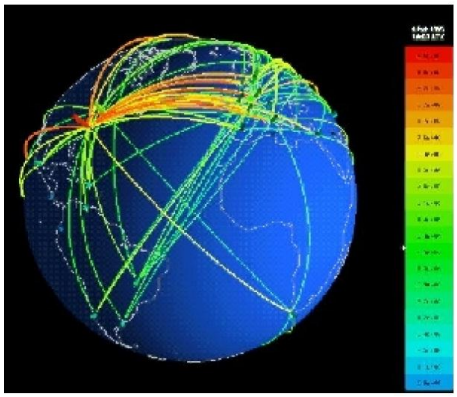


Figure 18. Οπτικοποίηση Internet Traffic

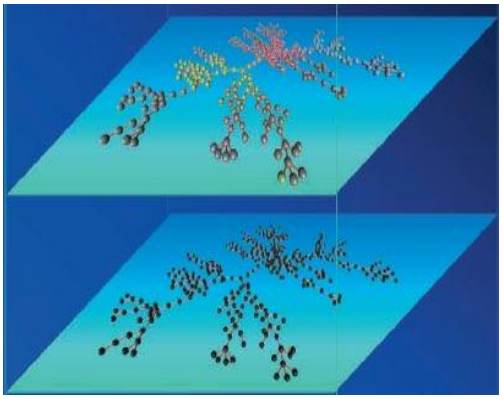


Figure 19. 3D δέντρο σε επιφάνεια

11.3 Τοπία δεδομένων (Information Landscapes)

Στην κατηγορία αυτή υπάγονται οι εφαρμογές που κάνουν χρήση των λεγόμενων 2.5D γραφικών διαστάσεων. Δηλαδή όλες οι οντότητες αποτυπώνονται ως επιφάνειες και απλά δίνεται η ψευδαίσθηση στον χρήστη της τρισδιάστατης αναπαράστασης.

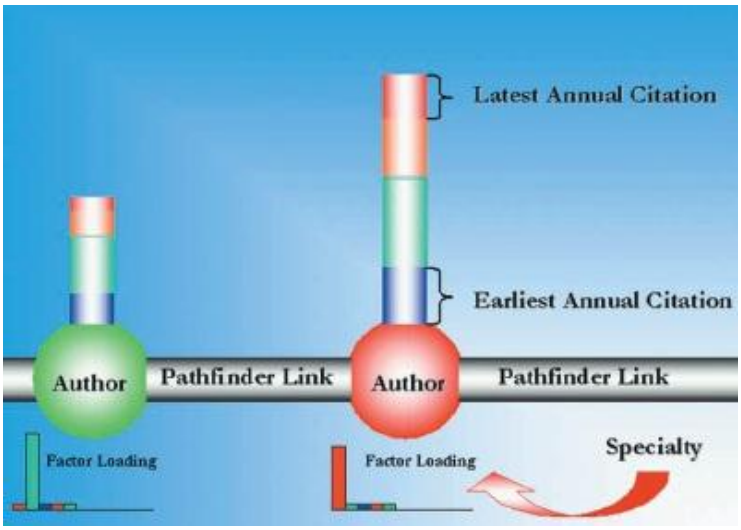


Figure 20. Αναπαράσταση 2.5 διαστάσεων

12 Previous – Related works (Προγενέστερες-Σχετιζόμενες Εργασίες)

Σημείο κλειδί στην εργασία μας αποτέλεσε μια προγενέστερη εργασία την οποία αναφέρουμε στο [78]. Θέμα αυτής αποτελεί η οπτικοποίηση στο τρισδιάστατο χώρο μοντέλων οντολογιών με αυτόματο τρόπο. Ο αλγόριθμος τους χρησιμοποιεί την τρίτη διάσταση του χώρου (δηλαδή το βάθος) σχεδόν αποκλειστικά για την παρουσίαση των σχέσεων isa (κληρονομικότητα). Τα properties (ιδιότητες), τοποθετούνται σε κάθετα επίπεδα. Έτσι, σεβόμενοι αυτές τις αρχές οδηγούμαστε στην κατασκευή ενός ιεραρχικού δέντρου επιπέδων, όπου τα επίπεδα θα συνδέονται με σχέσεις isa και σε κάθε επίπεδο θα υπάρχουν οντότητες που θα συνδέονται με properties.

Έμφαση δίνεται στο γεγονός ότι θα πρέπει να δημιουργηθούν ψευδό κόμβοι (ψευδό οντότητες) όταν προκύπτουν σχέσεις isa ανάμεσα σε επίπεδα τα οποία δεν θεωρούνται γειτονικά αφού επιθυμείται οι σχέσεις isa να συνδέουν οντότητες που βρίσκονται σε γειτονικά επίπεδα. Έτσι, οδηγούμαστε στην κατασκευή αυτών των ψευδό κόμβων οι οποίοι τοποθετούνται στα ενδιάμεσα επίπεδα και συνδέονται μεταξύ τους καθώς και με τις δύο αρχικές οντότητες που θέλαμε να συνδέσουμε με τις σχέσεις isa. Με αυτόν τον τρόπο καταφέρνουμε να διασπάσουμε την αρχική σχέση isa σε επιμέρους σχέσεις που συνδέουν τις αρχικές οντότητες και τις ψευδό οντότητες.

Επίσης, σε περίπτωση που δύο οντότητες που βρίσκονται σε διαφορετικά επίπεδα συνδέονται με κάποιο property τότε τοποθετούμε μία ψευδό οντότητα σε ένα από τα δύο επίπεδα και συνδέουμε την ψευδό οντότητα αυτή με σχέση τύπου property με την οντότητα που βρίσκονται στο ίδιο επίπεδο, και με σχέση τύπου isa με την οντότητα που βρίσκεται σε διαφορετικό επίπεδο.

Τέλος, λόγος γίνεται για την βέλτιστη τοποθέτηση των αντικειμένων στα επίπεδα καθώς θα πρέπει να υπολογίσουμε κατάλληλες x, y μεταβλητές διαστάσεων ώστε οι θέση των οντοτήτων να θεωρείται ικανοποιητική οπτικά καθώς θέλουμε οντότητες που συνδέονται με properties να βρίσκονται κοντά το ένα στο άλλο, και οντότητες που συνδέονται με σχέσεις isa να βρίσκονται σχετικά η μία πάνω στην άλλη στην διάσταση Z.

Στην εφαρμογή μας, επιθυμούμε παραπλήσια διάταξη οντοτήτων κάνοντας χρήση ενός τρισδιάστατου εικονικού χώρου. Πιο αναλυτικά, επιθυμούμε οι σχέσεις isa να βρίσκονται σε διαφορετικά επίπεδα ύψους (διάσταση Y) και τα properties να απλώνονται στο επίπεδο που σχηματίζουν οι διαστάσεις X, Z. Επίσης, ο αλγόριθμος από κατασκευής του προσπαθεί να τοποθετήσει οντότητες που σχετίζονται με properties κοντά το ένα στο άλλο και να μην επιβαρύνει το εικονικό χώρο με περιττά

αντικείμενα λαμβάνοντας υπόψη όχι απλά την παρουσίαση της πληροφορίας αλλά την εύκολή και γρήγορη αντίληψη της από τον χρήστη.

12.1 SemNet

Το σύστημα αυτό [9] απευθύνεται στην οπτικοποίηση σε τρισδιάστατο χώρο κατευθυνόμενων γράφων όπου και οι οντότητες καθώς και οι συνδέσεις έχουν ιδιότητες. Το εργαλείο χρησιμοποιεί ορθογώνια σχήματα που φέρουν πάνω τους κείμενο και χρωματιζόμενες συνδέσεις μεταξύ τους. Προσπαθεί να ομαδοποιήσει τις οντότητες στο χώρο βάση κοινών ιδιοτήτων παρέχοντας στο χρήστη πιο οργανωμένη πληροφορία. Σε περίπτωση που ο χρήστης επεξεργάζεται μια οντότητα η ομάδα αυτή τονίζεται.

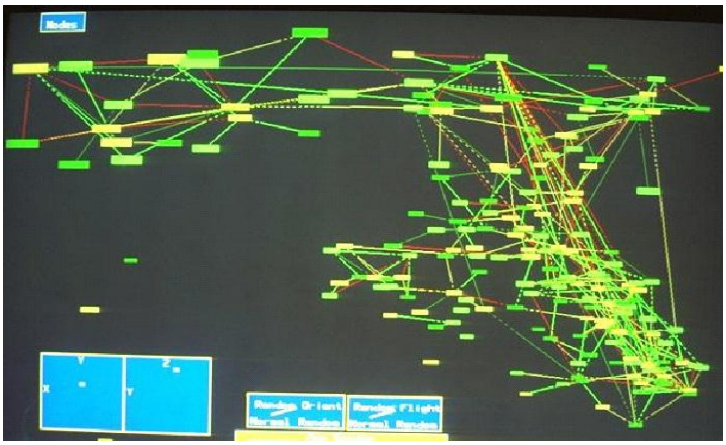


Figure 21. SemNet System

12.2 Cone Tree

Το σύστημα είναι σε θέση να παρουσιάσει οπτικοποίηση πληροφορίας κάνοντας χρήση οντοτήτων και συνδέσεων [10]. Το σχήμα παραπέμπει σε κώνο και η ιεραρχία τους σε δέντρο. Βασική οντότητα αποτελεί η ρίζα του δέντρου το οποίο εμφανίζεται στην κορυφή του κώνου και έπειτα κάτω από αυτό εμφανίζονται όλα τα παιδιά του. Η πλοήγηση περιορίζεται στη μεταφορά από πάνω έως κάτω και αριστερά, δεξιά. Οι οντότητες εμφανίζονται ως ημιδιαφανείς. Η ίδια η δομή του δέντρου προσφέρει ένα βαθμό οργάνωσης και ιεραρχίας των οντοτήτων. Για να γίνει πιο εύκολη η αλληλεπίδραση με το χρήστη όταν μια οντότητα επιλεγεί τότε αυτή μεταφέρεται αυτόματα κοντά στο χρήστη ώστε να είναι πιο εύκολος ο χειρισμός της. Με κατάλληλη διαδικασία κομμάτι του δέντρου που αποτελεί υπόδεντρο μιας οντότητας μπορεί να εξαφανιστεί ή να εμφανιστεί ανάλογα με την επιθυμία του χρήστη.

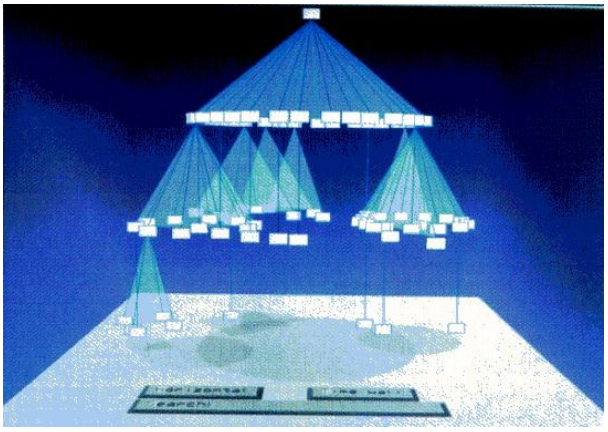


Figure 22. Κάθετο 3D ConeTree

12.3 Hyperbolic Space

Η εφαρμογή αυτή είναι σε θέση να παρουσιάσει οπτικοποίηση σε τρισδιάστατο χώρο κυκλικών γράφων όπως αυτών που παρουσιάζονται στον παγκόσμιο ιστό [11]. Τα τρισδιάστατα γραφικά επιλέγονται ώστε να γίνει καλύτερη διαχείριση του χώρου της οθόνης. Ο τρόπος που εμφανίζεται ο γράφος θυμίζει αυτόν των κωνικών δέντρων που αναφέραμε παραπάνω και αντικείμενα που βρίσκονται στο ίδιο ιεραρχικό επίπεδο φέρουν το ίδιο χρώμα. Το εργαλείο κάνει χρήση του μεγέθους του αντικειμένου που χρησιμοποιείται για να μπορέσει να τονίσει τη σημασία τους στο χρήστη. Αντικείμενα που βρίσκονται στις άκρες τις οθόνης είναι μικρότερα από αυτά που βρίσκονται στο κέντρο επικεντρώνοντας έτσι την προσοχή του χρήστη στα δεύτερα.

12.4 SeeNet3D

Κατάλληλο για αναπαράσταση γεωγραφικών δικτύων αφού είναι σε θέση να αναπαραστήσει το σημείο που βρίσκεται το δίκτυο [12]. Οι οντότητες αποτυπόνωνται πάνω σε μια εικονική σφαίρα. Οι οντότητες πάνω στη σφαίρα συμβολίζουν τις πόλεις ανά τον κόσμο και οι συνδέσεις μεταξύ τους συμβολίζουν τις πραγματικές συνδέσεις του διαδικτύου. Το χρώμα που φέρουν οι συνδέσεις συμβολίζει την κίνηση δεδομένων που γίνεται μέσω του διαδικτύου ανάμεσα στις συγκεκριμένες πόλεις. Ο χρήστης είναι σε θέση να κάνει περιστροφή της τρισδιάστατης σφαίρας.

12.5 Perspective Wall

Αποτελεί σύστημα για τρισδιάστατη αναπαράσταση συριακής πληροφορίας [13]. Όλη η πληροφορία αποτυπώνεται σε εικονικά επίπεδα, πράγμα που μπορεί να λειτουργήσει

θετικά και ως προς την ομαδοποίηση της πληροφορίας. Ο κάθετος άξονας μπορεί να λειτουργήσει ως σύνδεση ανάμεσα στα επίπεδα. Όταν ο χρήστης επιλέξει ένα αντικείμενο, τότε αυτό τονίζεται αυτόματα για να είναι πιο ευδιάκριτο στον χρήστη. Το σύστημα υστερεί σε σχέση με τις δυνατότητες που δίνει στον χρήστη για αλληλεπίδραση με τον τρισδιάστατο κόσμο αφού δεν δίνονται δυνατότητες πλοήγησης σε αυτόν.

12.6 Document Lens

Το εργαλείο αυτό χρησιμοποιείται ως επί το πλείστον για τρισδιάστατη απεικόνιση των σελίδων που περιέχονται μέσα σε ένα αρχείο κειμένου [14]. Το βασικό σχήμα πάνω στο οποίο αποτυπώνονται οι σελίδες παραπέμπει σε γεωμετρία ορθογωνίου. Ο χρήστης είναι σε θέση να αντλήσει πληροφορία από την σελίδα που έχει αποτυπωθεί στην επιφάνεια που παρουσιάζετε μπροστά του και η οποία εμφανίζεται ως πιο έντονη. Όταν ο χρήστης επικεντρωθεί σε μια σελίδα τότε το γενικό σχήμα της δομής μετατρέπεται σε γεωμετρία παραπλήσια εκείνης της πυραμίδας. Έτσι ο χρήστης μπορεί να αντλήσει πληροφορία από την σελίδα που παρουσιάζεται μπροστά του αλλά και διαισθητικά να αντλήσει πληροφορία και για τις άλλες σελίδες που εμφανίζονται γύρω από την κύρια. Η βασική δυνατότητα αλληλεπίδρασης με τον χρήστη εντοπίζεται στην ύπαρξη επιλογών περιστροφής της πυραμίδας – ορθογωνίου και στον τονισμό των λέξεων που συμφωνούν με την αναζήτηση που γίνεται.

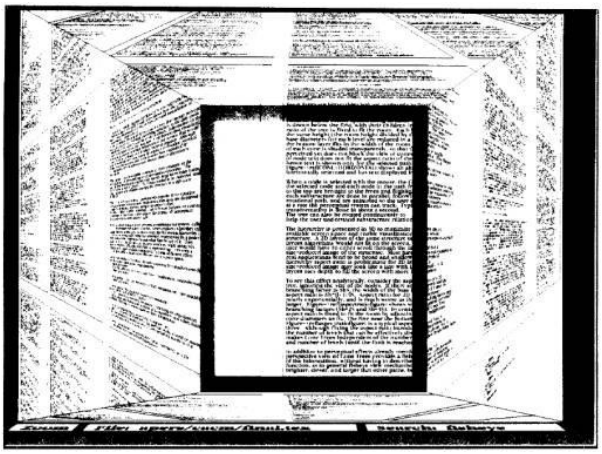


Figure 23. Αναπαράσταση αρχείων με Document Lens

12.7 3DPS

Κατάλληλο για αναπαράσταση γράφων και χαρτών [15]. Το εργαλείο αυτό απευθύνεται σε εργοστασιακή χρήση και λειτουργεί βάση εισαγωγής στον τρισδιάστατο χώρο

πολλαπλών σημείων τα οποία μετέπειτα συνδέονται σχηματίζοντας έναν πολύπλοκο γράφο. Ανάμεσα στα σημεία αυτού του γράφου σχηματίζονται επιφάνειες πάνω στις οποίες αποτυπώνονται εικόνες. Το πλήθος των σημείων μπορεί να διαμορφωθεί ανάλογα με τις ανάγκες του χρήστη και έτσι να τονιστεί ή όχι κάποιο σημείο της γενικότερης δομής.

Figure 24. Εκτύπωση 3DPS

12.8 FSN (file system visualizer)

Πρόκειται για μια πειραματική εφαρμογή που στόχο έχει την παρουσίαση όλου του συστήματος αρχείων σε τρισδιάστατα γραφικά κατασκευασμένο από την SGI [16]. Παρότι δεν κατάφερε ποτέ να εξελιχθεί σε ένα πλήρως λειτουργικό διαχειριστή αρχείων κέρδισε μεγάλη αναγνωρισημότητα χάρη στην προβολή του από το Hollywood σε ταινία. Το μέγεθος των φακέλων και των αρχείων επιδεικνύεται βάση του ύψους και το χρώμα αποτυπώνει την ηλικία τους. Όταν ο κέρσορας βρεθεί πάνω σε ένα αντικείμενο τότε εμφανίζονται τα στοιχεία του. Όταν ένα αντικείμενο επιλεγεί τότε γίνεται αυτόματο zoomπάνω σε αυτό. Όλες αυτές οι τεχνικές χρησιμοποιήθηκαν για να προσελκύσουν το ενδιαφέρων των χρηστών. Το βασικό εργαλείο που χρησιμοποιείται για την αλληλεπίδραση είναι το ποντίκι όπως επίσης και τα menu, buttons κτλ. Τα τρισδιάστατα αντικείμενα μπορούν να επιλεγούν, μετακινηθούν, εκτελεστούν.

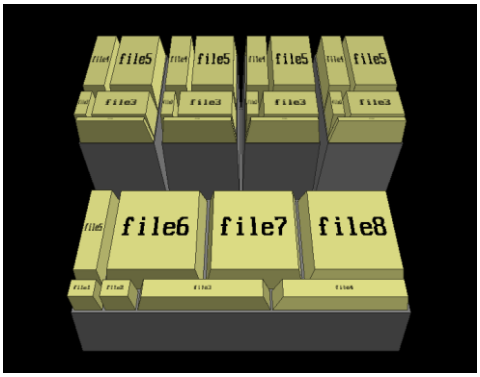


Figure 25. Σύστημα αρχείων σε τρεις διαστάσεις

12.9 SDM (Selective Dynamic Manipulation)

Οι οντότητες σε αυτό το μοντέλο αναπαριστώντας ως κύλινδροι πάνω σε μια επιφάνεια [17]. Προσφέρονται δυνατότητες συνάθροισης σε ομάδες. Τα επιλεγμένα αντικείμενα μπορούν να χαρακτηριστούν από διαφορετικό χρώμα, μέγεθος ή θέσης στον χώρο. Και όλο αυτό γίνεται για να επικεντρωθεί η προσοχή του χρήστη σε αυτά.

Προτέρημα αυτού του μοντέλου αποτελεί το γεγονός ότι είναι ιδιαίτερα εύκολο να γίνει σύγκριση μεταξύ οντοτήτων χάρη στο μέγεθος τους, στο χρώμα τους ή στην θέση τους στον εικονικό χώρο. Τα προτερήματα αυτά αναγνωρίζει, επιδιώκει και επιτυγχάνει επίσης και η εφαρμογή μας δίνοντας βάση στο μέγεθος, το χρώμα και την θέση που καταλαμβάνουν τα τρισδιάστατα αντικείμενα όπως επίσης και στην γρήγορη αντίληψη των σχέσεων μεταξύ των οντοτήτων.

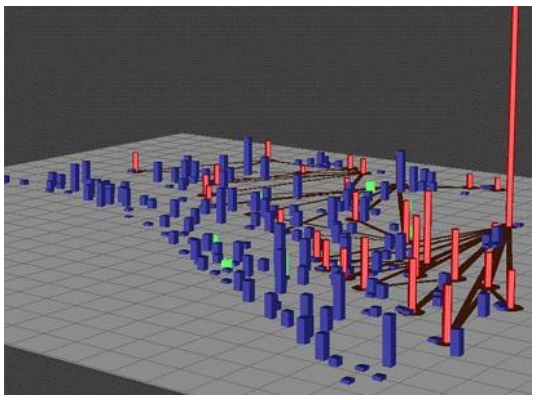


Figure 26. Τεχνικές οπτικοποίησης SDM

12.10 Bead

Στο σύστημα αυτό γίνεται αποτύπωση δεδομένων που αφορούν βιβλιογραφία [18]. Τα αρχεία συμβολίζονται ως κύβοι πάνω σε ένα επίπεδο το οποίο χωρίζεται σε θεματολογίες. Τα ονόματα των αρχείων εμφανίζονται ως οντότητες κειμένου πάνω από τους κύβους. Το κάθε αρχείο τοποθετείται στο επίπεδο πάνω στο κομμάτι που αφορά την κατηγορία του. Έτσι επιτυγχάνεται ομαδοποίηση της πληροφορίας. Στις οντότητες που βρίσκονται κοντά στον χρήστη εμφανίζονται περισσότερες πληροφορίες, όπως π.χ. ο τίτλος τους. Το πλεονέκτημα αυτού του συστήματος είναι το ότι μπορεί να χρησιμοποιηθεί ταυτόχρονα από περισσότερους από έναν χρήστες.

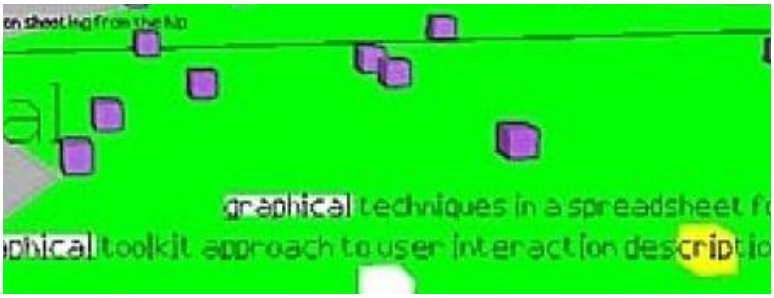


Figure 27. Δεδομένα βιβλιογραφίας στο BEAD

12.11 Web Forager

Ο λόγος δημιουργίας του αφορά την οργάνωση των αρχείων του διαδικτύου κατηγοριοποιώντας τα ως βιβλία [19]. Η διεπαφή είναι παρόμοια με αυτήν ενός γραφείου και οι χρήστες καλούνται να τοποθετήσουν τα αρχεία στους κατάλληλους καταλόγους. Τα αρχεία ανήκουν σε τρεις βασικές κατηγορίες οι οποίες είναι: η τοποθεσία συγκέντρωσης όπου βρίσκονται τα έγγραφα που χρειάζονται άμεση επεξεργασία, τοποθεσία αποθήκευσης όπου αποθηκεύονται έγγραφα για βραχυχρόνια μελλοντική χρήση, τοποθεσία μακροχρόνιας αποθήκευσης όπου στοιβάζονται αντικείμενα που δεν πρόκειται να χρησιμοποιηθούν στο έγγειης μέλλον. Η μεταφορά από την μια κατηγορία στην άλλη μπορεί να γίνει από το χρήστη άμεσα ή έμμεσα καθώς το σύστημα μπορεί να αλλάξει κατηγορία αυτόματα σε έγγραφα αν διαπιστώσει ότι ο χρήστης τα επιλέγει πολλές φορές ή λίγες.

12.12 Information Cube

Οι χρήστες των συστημάτων υποβοήθησης λήψης αποφάσεων συχνά βλέπουν τα δεδομένα ως μια μορφή κύβων δεδομένων [20], [21]. Ο κύβος χρησιμοποιείται για να

αναπαραστήσει καθώς ταυτόχρονα και ορισμένα χαρακτηριστικά που μας ενδιαφέρουν. Παρότι αποκαλείται κύβος μπορεί να είναι δυσδιάστατος, τρισδιάστατος ή και παραπάνω διαστάσεων. Κάθε διάσταση είναι σε θέση να αναπαραστήσει κάποιο χαρακτηριστικό και το κελί που περικλείεται στον κύβο δεδομένων και που αντιστοιχεί στις διαστάσεις αυτές περιέχει την πληροφορία που θέλουμε να αποθηκεύσουμε.

Η ίδια η δομή του κύβου μας διευκολύνει στην αποτίμηση επερωτήσεων. Για παράδειγμα αν σε ένα κύβο δεδομένων έχουν αποθηκευτεί πληροφορίες που αφορούν θερμοκρασίες που καταγραφήκαν στην Ελλάδα ανά πόλη αυτές θα είχαν εισαχθεί με ταξινομημένο τρόπο πρώτον ανά πόλη και δεύτερων ανά αύξουσα αριθμητική τιμή. Έτσι η αποτίμηση οποιασδήποτε επερώτησης θα αντιστοιχούσε στην επιλογή ενός χωρίου του κύβου εφόσον η ίδια η δομή του επιτρέπει την ύπαρξη μόνο οργανωμένης πληροφορίας και η αυτόματη ταξινόμηση που γίνεται στους υποκύβους της βασικής οντότητας προσφέρει λειτουργία αντίστοιχη αυτή του group-by των βάσεων δεδομένων.

Πλεονέκτημα ή μειονέκτημα αυτού του μοντέλου αποτελεί το ότι ανάλογα με το είδος της πληροφορίας που αποθηκεύεται ο κύβος μπορεί να χωριστεί σε λίγους ή πάρα πολλούς υποκύβους. Σε περίπτωση που οι υποκύβοι είναι λίγοι στον αριθμό τότε αυτό μπορεί να σημαίνει ότι ο καθένας τους περιέχει μεγάλη ποσότητα πληροφορίας. Στην αντίθετη περίπτωση όπου ο κύβος έχει υπερβολικά πάρα πολλούς υποκύβους τότε η ποσότητα αυτή γίνεται πολύ δύσκολη στο χειρισμό της ακόμα και για τους σημερινούς υπολογιστές.

Το βασικότερο μειονέκτημα αυτού του μοντέλου αποτελεί το γεγονός ότι μπορεί να δεσμεύει μεγάλη ποσότητα αχρησιμοποίητου αποθηκευτικού χώρου. Για παράδειγμα αν κατασκευάζαμε τον προηγούμενο κύβο δύο διαστάσεων όπου η μια διάσταση ήταν οι θερμοκρασίες και στην άλλη οι πόλεις τότε θα υπήρχε τεράστια κατασπατάληση χώρου εφόσον σε μια πόλη θα αντιστοιχούσε μοναχά μια θερμοκρασία. Γι’ αυτό θα πρέπει να γίνει χρήση ειδικών τεχνικών ώστε να αποφευχθεί η σπατάλη αποθηκευτικού χώρου.

Figure 28. Photo Data Cube

12.13 N-Vision

Το εργαλείο αυτό απευθύνεται στην οπτικοποίηση δεδομένων πολλών διαστάσεων [22]. Το σύστημα οπτικοποιεί τρισδιάστατα αντικείμενα τα οποία περικλείουν άλλα τρισδιάστατα αντικείμενα. Η δυνατότητα εισαγωγής αντικειμένων μέσα σε αντικείμενα προσφέρει ένα μηχανισμό παρουσίασης σχέσεων ανάμεσα στα αντικείμενα. Οι οντότητες που έχουν επιλεγεί παρουσιάζονται πλησιέστερα στον χρήστη από τις υπόλοιπες. Η διαχείριση του συστήματος μπορεί να γίνει μέσω του ποντικιού ή δυσδιάστατου γραφικού περιβάλλοντος. Επίσης υποστηρίζει περιστροφή, μεταφορά, αλλαγή μεγέθους επιλεγμένων αντικειμένων.

12.14 Αξιολόγηση

Τα συστήματα αυτά αν και αξιόλογα, στην πλειοψηφία τους προσφέρουν απλά πληροφορία στον χρήστη δίχως αυτός να είναι σε θέση να την μεταβάλει. Έτσι ο χρήστης δεν είναι σε θέση να αλληλεπιδράσει με το σύστημα και να εισάγει ή να διαγράψει πληροφορία σε πραγματικό χρόνο μεταβάλλοντας με αυτό τον τρόπο την πληροφορία που απεικονίζεται στον εικονικό χώρο.

Επίσης, συνήθως τα συστήματα αυτά παρουσιάζουν μία στατική άποψη – θέαση των αντικειμένων και ο χρήστης δεν είναι σε θέση να πλοηγηθεί στον εικονικό κόσμο. Δηλαδή, δεν είναι σε θέση να μεταφερθεί σε οποιοδήποτε σημείο εφαρμόζοντας τεχνικές όπως μεταφορά, περιστροφή, μεγέθυνση, σμίκρυνση στον εικονικό χώρο.

Σε πολλά συστήματα (όπως τα DataCube) υπάρχει τεράστια απόκρυψη πληροφορίας από τον χρήστη, αφού αντιλαμβάνεται πολύ μικρό ποσοστό της, καθώς η μεγαλύτερη ποσότητα βρίσκεται εσωτερικά στον κύβο.

Περαιτέρω, συνήθως ο χρήστης δεν είναι σε θέση να αναφερθεί συγκεκριμένα σε ένα αντικείμενο και να το χειριστεί ως μοναδική οντότητα. Για παράδειγμα, να επιλέξει ένα αντικείμενο και να το μεταφέρει σε μία άλλη θέση ή να αλλάξει τα εμφανισιακά χαρακτηριστικά αυτού του αντικειμένου.

Όσα συστήματα αναπτύσσονται πάνω σε μία συγκεκριμένη επιφάνεια επιτελούν μεγάλη σπατάλη εικονικού χώρου καθώς έτσι δεν μπορούν να εισάγουν μεγάλο πλήθος αντικειμένων (επάνω σε μία μόνο επιφάνεια) και υπάρχει μεγαλύτερη πιθανότητα επικάλυψης αντικειμένων μεταξύ τους.

Εξαιτίας αυτών των αρνητικών παραγόντων οδηγηθήκαμε στην κατασκευή του εργαλείου μας στο οποίο αντιμετωπίζουμε όλα τα παραπάνω προβλήματα με τον βέλτιστο πιστεύουμε τρόπο.

13 3D Τεχνικές ανίχνευσης επικάλυψης

Στην προσπάθεια μας να είμαστε σίγουροι ότι κανένα σημείο του χώρου δε θα φιλοξενεί παραπάνω από μια οντότητες χρειάστηκε αν κάνουμε χρήση τεχνικών εντοπισμού επικάλυψης βάση της γεωμετρίας των σχημάτων που χρησιμοποιούνται [33]. Προς απλοποίηση και βελτιστοποίηση της διαδικασίας ελέγχου το σχήμα των οντοτήτων που χρησιμοποιούνται παραπέμπει σε βασικά γεωμετρικές δομές όπως κύβοι, γραμμές, επιφάνειες.

Το μεγάλο πρόβλημα που καλούμαστε να αντιμετωπίσουμε είναι ο αριθμός των ελέγχων που θα πρέπει να γίνουν ώστε να ανιχνευθεί επιτυχώς η επικάλυψη και που με τη σειρά του μας οδηγεί στη βέλτιστη χρήση των επεξεργαστικών πόρων που διαθέτει το σύστημα [40], [41], [46].

Για παράδειγμα σε περίπτωση που ο τρισδιάστατος κόσμος μας περιείχε n οντότητες τότε οι έλεγχοι που θα έπρεπε να γίνουν για το πρώτο αντικείμενο (αν επικαλύπτεται μα κάποιο από τα άλλα) θα ήταν n-1 σε αριθμό. Το δεύτερο αντικείμενο θα μπορούσε να επικαλύπτεται με n-2 άλλα αντικείμενα και ως εκ τούτου θα έπρεπε να γίνουν n-2 έλεγχοι έχοντας λάβει υπόψην μας το ότι έχει γίνει ήδη ο έλεγχος επικάλυψης για το πρώτο αντικείμενο [35]. Συνεχίζοντας αναδρομικά την διαδικασία αυτή καταλήγουμε στο συμπέρασμα ότι στο σύνολο για ελέγξουμε το ενδεχόμενο επικάλυψης n οντοτήτων μεταξύ τους σε ένα τρισδιάστατο εικονικό κόσμο θα πρέπει ο αριθμός των ελέγχων να ισούται με :

(n-1) * (n-2) * (n-3) ... 1

η οποία ποσότητα ισούται με => n! / 2!*(n-2)!

Σε συστήματα απεικόνισης κινούμενων οντοτήτων θα χρειαζόταν να κάνουμε έλεγχο επικάλυψης σε κάθε στιγμιότυπο της εικόνας. Γι’ αυτό και θεωρείται τόσο σημαντική η διαδικασία ελέγχου επικάλυψης να είναι αποδοτική, να ανταποκρίνεται σε πραγματικό χρόνο και να μην επιβαρύνει υπερβολικά την επεξεργαστική ισχύ του συστήματος. Έχοντας λάβει υπόψην τα παραπάνω προτείνονται οι μετέπειτα τεχνικές ελέγχου επικάλυψης.

13.1 Κατάτμηση Χώρου (Space Partitioning)

Στην προσπάθεια μείωσης του αριθμού των ελέγχων που θα πρέπει να γίνουν για να εντοπιστεί επικάλυψη ένας τρόπος να επιτευχθεί αυτό θα ήταν η αντιστροφή της λογικής χάρη στην οποία γίνονται οι έλεγχοι αυτοί.

Δηλαδή αντί να ψάξουμε αν δύο οντότητες συνυπάρχουν στο ίδιο σημείο του τρισδιάστατου χώρου εναλλακτικά θα μπορούσαμε να ψάξουμε αν σε ένα σημείο του τρισδιάστατου χώρου συνυπάρχουν δύο οντότητες [44]. Αυτό συνεπάγεται ότι ο ίδιος ο τρισδιάστατος χώρος θα πρέπει να χωριστεί κατάλληλα σε επιμέρους υποχώρους με τέτοιο τρόπο ούτος ώστε να μπορεί να ανιχνευθεί επιτυχώς επικάλυψη οντοτήτων αλλά και να μην επιβαρυνθεί το σύστημα με μεγάλο σε πλήθος αριθμό ελέγχων.

Ο συνήθης τρόπος διαχωρισμού του χώρου σε υποχώρους γίνεται τεμαχίζοντας το βασικό σχήμα σε μικρότερα κομμάτια τα οποία διαθέτουν τη μορφή κύβου, ορθογωνίου, πυραμίδας, δέντρων. Οι τεχνικές αυτές ενδέχεται να γίνουν παραλήπτες βελτιστοποιήσεων που θα έχουν ως αποτέλεσμα την καλύτερη και πιο γρήγορη λειτουργία τους. Το σημαντικό πλεονέκτημα της τεχνικής αυτής είναι ότι ο αριθμός των ελέγχων που θα πρέπει να γίνουν βάση του πλήθους των οντοτήτων που υπάρχουν ακολουθεί γραμμική πολυπλοκότητα, σε αντίθεση με άλλες τεχνικές.

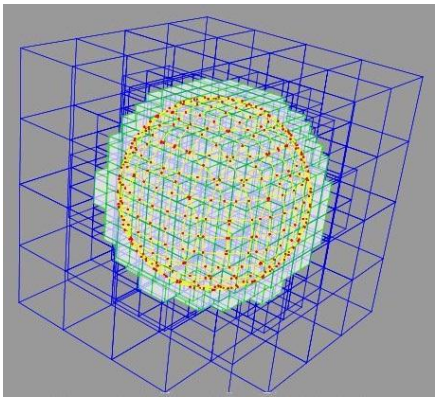


Figure 29. 3D Space Partitioning

13.2 Περικλειόμενοι Κύβοι (Bounding Boxes)

Στην τεχνική αυτή η κάθε τρισδιάστατη οντότητα περιβάλλεται από μια εικονική τρισδιάστατη ορθογώνια δομή η οποία και θέτει τα όρια της οντότητας αυτής στο χώρο [27], [43]. Σε περίπτωση που οποιοδήποτε σημείο των ορθογώνιων αυτών δομών επικαλύψει κάποιο άλλο τότε οι οντότητες που περιέχονται μπορεί να επικαλύπτονται ή και όχι. Δηλαδή ενδέχεται να επικαλύπτονται οι εικονικές ορθογώνιες δομές που τοποθετούνται περιφερειακά των σχημάτων αλλά να μην επικαλύπτονται στην πραγματικότητα τα γεωμετρικά αυτά σχήματα. Σε περίπτωση που θελήσουμε να είμαστε σίγουροι για την ύπαρξη ή μη επικάλυψης θα πρέπει να κάνουμε περισσότερους ελέγχους. Στην αντίθετη περίπτωση όπου οι ορθογώνιες δομές δεν ακουμπάνε η μια την άλλη, μπορούμε να είμαστε σίγουροι ότι και τα γεωμετρικά σχήματα που περιέχονται σε αυτά δεν ακουμπάνε το ένα το άλλο.

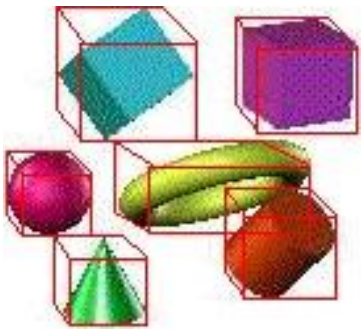


Figure 30. Εικονικά Bounding Boxes

Για να ελέγξουμε βάση των ορθογώνιων αυτών δομών αν κάποιο σχήμα ακουμπάει κάποιο άλλο μπορούμε πολύ εύκολα να συγκρίνουμε τη μικρότερη και τη μεγαλύτερη τιμή που λαμβάνουν οι μεταβλητές x, y, z των κορυφών.

Για παράδειγμα αν διαθέτουμε δύο αντικείμενα A και B και οι συντεταγμένες τους είναι AxMin, AxMax, AyMin, AyMax, AzMin, AzMax, BxMin, BxMax, ByMin, ByMax, BzMin, and BzMax τότε τα αντικείμενα αυτά επικαλύπτονται αν ικανοποιηθούν οι παρακάτω προϋποθέσεις :

AxMin < BxMax and AxMax > BxMin

AyMin < ByMax and AyMax > ByMin

AzMin < BzMax and AzMax > BzMin

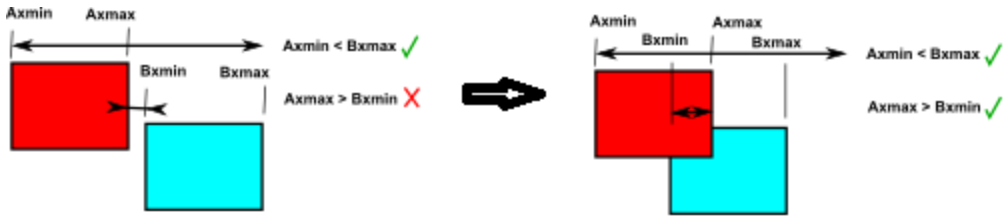


Figure 31. 2D επικάλυψη αντικειμένων

13.3 Περικλειήζουσες Σφαίρες (Bounding Spheres)

Αποτελεί τεχνική παρόμοια με την προηγούμενη με τη διαφορά ότι πλέον μιλάμε για περικλείον σχήμα σφαίρας και όχι ορθογωνίου [31], [45]. Για παράδειγμα έστω αντικείμενα A και B με κέντρο ax, ay, az, bx, by, bz και διάμετρο ar, br. Σε αυτή την περίπτωση οι περικλείουσες σφαίρες επικαλύπτονται αν ισχύει ότι:

$(ax-bx)^2+(ay-by)^2+(az-bz)^2 < (ar+br)^2$

Το πλεονέκτημα αυτής της μεθόδου είναι το ότι η τεχνική αυτή είναι ανεξάρτητη κατεύθυνσης των αντικειμένων καθώς όλα τα σημεία της επιφάνειας της σφαίρας απέχουν ίσα από το κέντρο της. Μειονέκτημα της μεθόδου αυτής αποτελεί το γεγονός ότι ενδέχεται η γεωμετρία του σχήματος που θέλουμε να ελέγξουμε αν επικαλύπτεται με κάποιο άλλο, να μην συμβαδίζει με αυτήν της σφαίρας και έτσι να οδηγηθούμε σε λανθασμένα αποτελέσματα ελέγχου επικάλυψης. Για παράδειγμα αν θέλουμε να κάνουμε έλεγχο για μακρόστενα αντικείμενα όπως είναι οι ευθείες, τότε η τεχνική αυτή αποτυγχάνει στις περισσότερες των περιπτώσεων καθώς ο χώρος που καταλαμβάνεται από τις ευθείες είναι ελάχιστος σε σχέση με τον χώρο που καταλαμβάνεται από τις σφαίρες [42].

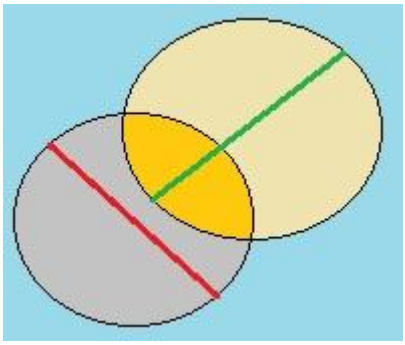


Figure 32. Λάθος ανίχνευσης Bounding Spheres

13.4 Τεχνάσματα Ανίχνευσης Επικάλυψης

Έχοντας στόχο την βέλτιστη κατανομή και διαχείριση της επεξεργαστικής ισχύς του συστήματος καθώς και την αποφυγή περιττών υπολογισμών και ελέγχων προτείνονται οι παρακάτω τεχνικές βελτιστοποιήσεις ελέγχου επικάλυψης [28], [32], [34], [36], [47], [48].

- Moving Object: Στην τεχνική αυτή γίνεται έλεγχος τυχόν επικάλυψης μόνο για τα αντικείμενα που έχουν κινηθεί και αγνοούνται τα αντικείμενα που έχουν παραμείνει ακίνητα. Αυτό συνεπάγεται ότι για «n» αντικείμενα όπου έχει κινηθεί μόνο ένα αντικείμενο τότε το πλήθος των ελέγχων που θα πρέπει να γίνει ισούται με n-1. Άρα αποφεύγονται (n-2) * (n-3)* ... *1 έλεγχοι [38].

- Frame Counter: Εφόσον η ανθρώπινη αντίληψη δεν είναι σε θέση να επεξεργαστεί όλα τα στιγμιότυπα που προβάλλονται από τον ηλεκτρονικό υπολογιστή θα μπορούσαμε να διενεργήσουμε ελέγχους σε συγκεκριμένα frames και όχι σε όλα. Για παράδειγμα αν εκτελούσαμε ελέγχους κάθε 5 frames αυτό αυτόματα θα σήμαινε ότι το πλήθος των ελέγχων που θα έπρεπε να γίνουν θα μειωνόταν στο 1/5 ή 20%.

- Set of Interest: Δεδομένου ότι δεν αποτελούν όλα τα αντικείμενα που βρίσκονται στον εικονικό κόσμο οντότητες ενδιαφέροντος, θα μπορούσαμε να αγνοήσουμε τα αντικείμενα αυτά και έτσι να μειώσουμε το πλήθος των οντοτήτων που ελέγχουμε ως προς την ύπαρξη επικάλυψης ή μη.

- Movement Direction: Σε περίπτωση που ένα αντικείμενο κινηθεί προς κάποια κατεύθυνση θα μπορούσαμε να ελέγξουμε την ύπαρξη επικάλυψης μόνο ανάμεσα στην οντότητα που μετακινήσαμε και σε οντότητες που βρίσκονται προς την κατεύθυνση κίνησης [37].

- Objects Distance: Η ίδια η απόσταση ανάμεσα στα αντικείμενα μπορεί να παίξει ζωτικό ρόλο στο πλήθος των ελέγχων που θα χρειαστεί να εκτελεστούν αφού οντότητες που βρίσκονται σε μεγάλη απόσταση μεταξύ τους δεν χρειάζεται να ελεγχθούν για την ύπαρξη επικάλυψης καθώς το μέγεθος τους είναι μικρότερο από την μεταξύ τους απόσταση [30], [37], [39].

- Mass Center: Το κέντρο μάζας των οντοτήτων μπορεί να ληφθεί υπόψη δεδομένου ότι χάρη σε αυτό μπορεί να υπολογιστεί πιο αποτελεσματικά η απόσταση ανάμεσα στις οντότητες και έτσι να γίνει χρήση τεχνικής που αναφέρθηκε πιο πάνω για βελτιστοποίηση ελέγχων βάση απόστασης.

- Space Trees: Χάρη στα δέντρα αυτά μπορεί να γίνει διχοτόμηση του χώρου σε υποχώρους με κριτήρια κατάλληλα ώστε να μπορούν στο μέλλον να αποφευχθούν συγκρίσεις ανάμεσα σε οντότητες που βρίσκονται σε διαφορετικά υπόδεντρα. Το βάθος του δέντρου αλλά και το πλήθος των παιδιών κάθε κόμβου μπορούν να αντιστοιχούν σε μια στατική ποσότητα αλλά μπορεί να είναι και δυναμικά ανάλογα με το μέγεθος του χώρου ή το πλήθος των οντοτήτων που θα προστεθούν σε αυτό.

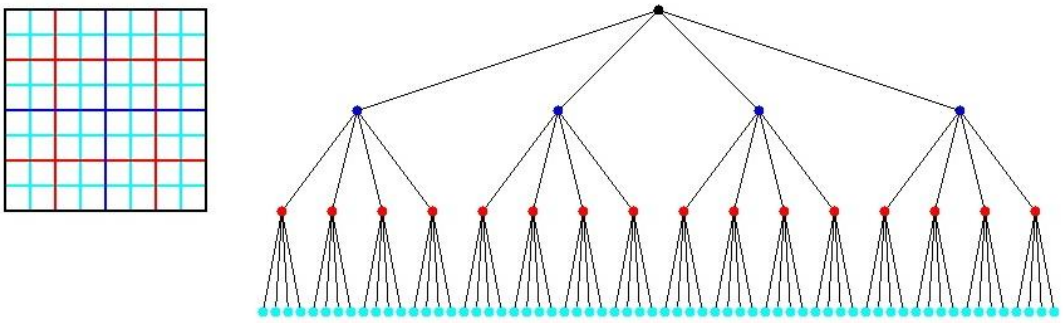


Figure 33. Space Quadtree με τέσσερα επίπεδα

Στην υλοποίηση αυτή ενδέχεται να υπάρχουν και κομμάτια του χώρου που δεν λαμβάνονται υπόψη. Για παράδειγμα υπόδεντρο κομμάτι που αντιστοιχεί σε χώρο που βρίσκεται μέσα σε ένα γεωμετρικό σχήμα δεν χρειάζεται να ελεγχθεί αν στο περίβλημα του δεν υπάρχει σύγκρουση με κάποιο άλλο σχήμα.

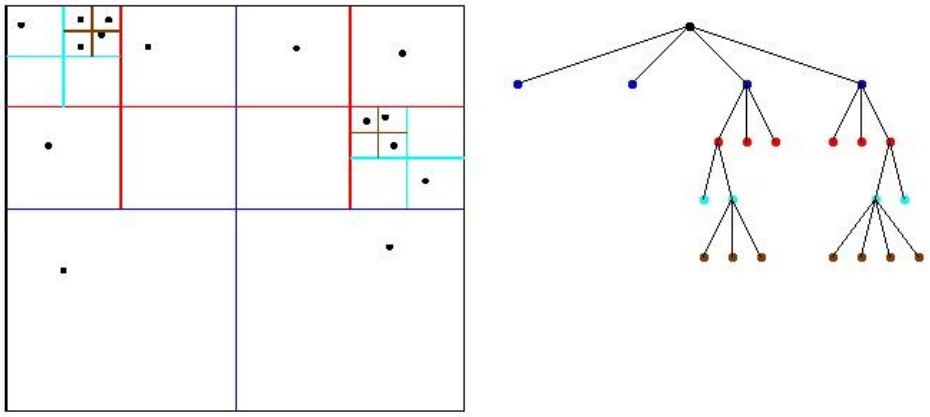


Figure 34. Quadtree όπου κανένα τετράγωνο δεν περιέχει παραπάνω από ένα αντικείμενα

14 Συμπεράσματα 3D Οπτικοίησης

Ο κύριος λόγος που χρησιμοποιήθηκε τρισδιάστατη αναπαράσταση είναι ότι χάρη σε αυτήν επιτυγχάνεται πολύ καλύτερη διαχείριση του χώρου της οθόνης και της ανθρώπινης αντίληψης. Στον ίδιο χώρο οθόνης μπορεί να παρουσιασθεί πολλαπλάσια ποσότητα πληροφορίας καθώς πλέον διαθέτουμε την δυνατότητα να εισάγουμε αντικείμενα σε διαφορετικό μέγεθος βάθους. Ο ανθρώπινος εγκέφαλος έχει εκπαιδευτεί να επεξεργάζεται τρισδιάστατες εικόνες και έρευνες έχουν δείξει ότι μπορεί πολύ πιο εύκολα να αντιληφθεί και να επεξεργαστεί δεδομένα από τον υπολογιστή όταν αυτά παρουσιάζονται ως οντότητες ενός τρισδιάστατου χώρου.

Σε σύγκριση με τα περισσότερα συστήματα αναπαράστασης τρισδιάστατων γραφικών η δική μας εφαρμογή δίνει στο χρήστη την δυνατότητα πλήρους πλοήγησης στον τρισδιάστατο χώρο, της εισαγωγής, διαγραφής, μεταφοράς, επεξεργασίας πληροφορίας σε πραγματικό χρόνο και της ρεαλιστικής αναπαράστασης τρισδιάστατων γραφικών. Επίσης δίνει δυνατότητες τονισμού και διαχωρισμού οντοτήτων μέσω του μεγέθους τους, του σχήματος τους καθώς και του χρώματος τους.

Πειραματικά βλέπουμε ότι η απόκριση του συστήματος χαρακτηρίζεται σχεδόν άμεση και η επιβάρυνση μνήμης-επεξεργαστή είναι αμελητέα για μικρό αριθμό οντοτήτων και κυμαίνεται σε φυσιολογικά πλαίσια για μεγάλο αριθμό τους.

15 Υλοποίηση στην Java3D

Πρόκειται για μια υποενότητα της γλώσσας προγραμματισμού Java που προσφέρει μια διεπαφή για τη δημιουργία προγραμμάτων με τρισδιάστατα γραφικά στοιχεία. Το API που παρέχεται από τη γλώσσα αυτή μας δίνει μια συλλογή από κλάσεις και λειτουργίες για τη δημιουργία καθώς και τη διαχείριση αντικειμένων που προκύπτουν ως άθροισμα βασικών τρισδιάστατων γεωμετρικών σχημάτων [61], [62], [63], [64], [65], [66].

15.1 Java3D API

Είναι η ιεραρχία των Java κλάσεων που μπορεί να χειριστεί ο χρήστης για τη δημιουργία τρισδιάστατων γραφικών παρέχοντας του υψηλού επιπέδου δομές όπως γεωμετρικά αντικείμενα τα οποία έχουν ιδιότητες και χαρακτηριστικά και που μπορούν να εισαχθούν σε οποιοδήποτε σημείο του εικονικού κόσμου [67]. Η τελευταία έκδοση του API είναι η 1.3 και η γλώσσα σχεδιάστηκε με στόχο την δημιουργία και προσομοίωση του πραγματικού κόσμου στην εικονική πραγματικότητα [69]. Η ποιότητα και η ποσότητα της πληροφορίας που απεικονίζεται στην οθόνη του χρήστη ενδέχεται να ελέγχεται αυτόματα και στόχο έχει τη βέλτιστη κατανομή των πόρων του συστήματος καθώς επίσης και τη δυνατότητα αναπαράστασης τρισδιάστατων στοιχείων σε κοινές δυσδιάστατες οθόνες υπολογιστών. Βασικό στοιχείο της Java 3D είναι ο γράφος σκηνής όπου προστίθενται οι οντότητες με μια μορφή δενδρικής ιεραρχίας που ξεκάθαρα δηλώνει το περιεχόμενο του εικονικού κόσμου και τα χαρακτηριστικά του.

Σημαντικό πλεονέκτημα προγραμμάτων που έχουν υλοποιηθεί στη γλώσσα προγραμματισμού Java 3D αποτελεί το γεγονός ότι μπορούν να λειτουργούν αυτόνομα ή μέσο φυλλομετρητών που έχουν ρυθμιστεί κατάλληλα για να τα υποστηρίξουν. Το API περιέχει εκατοντάδες κλάσεις και πολλές περισσότερες μεθόδους. Παρόλα αυτά απλά παραδείγματα εικονικών αντικειμένων μπορούν να υλοποιηθούν κάνοντας χρήση μερικών κλάσεων.

15.2 Virtual Universe

Για τη δημιουργία τρισδιάστατου σύμπαντος απαραίτητη κρίνεται η ύπαρξη ενός γράφου σκηνής ο οποίος γίνεται πραγματικότητα χάρη σε στιγμιότυπα κλάσεων της Java. Περιέχει αντικείμενα που είναι υπεύθυνα για τη γεωμετρία, τη θέση, του προσανατολισμού, της εμφάνισης κτλ. Είναι ουσιαστικά μια δενδρική αναπαράσταση της τεχνικής απεικόνισης node-links όπου κόμβοι είναι οι οντότητες της Java 3D και σχέσεις οι οποίες μπορεί να είναι δύο ειδών. Το πρώτο είδος αφορά την σχέση γονέα

παιδιού η οποία λαμβάνει μέρος κυρίως στις οντότητες Group, και η δεύτερη που χρησιμοποιείται για τις αναφορές. Για παράδειγμα την αναφορά που υπάρχει από ένα γεωμετρικό αντικείμενο όπως η κλάση “Cube” προς ένα αντικείμενο τύπου εμφάνισης όπως η κλάση “Appearance” κάνοντας χρήση της μεθόδου “setappearance”. Πρέπει να τονίσουμε ότι στο δέντρο αυτό κάθε κόμβος μπορεί να έχει το πολύ έναν γονέα και γι’ αυτό μια οντότητα μπορεί να προστεθεί ως παιδί σε το πολύ έναν κόμβο. Η αρχή αυτή τηρείτε για τον αποκλεισμό λειτουργιών που έρχονται σε αντίθεση μεταξύ τους.

Όλη η πληροφορία που αφορά την τελική κατάσταση μιας οντότητας (θέση, μέγεθος, εμφάνιση, προσανατολισμός, φωτισμός, είδος υλικού κτλ.) κρύβεται στην διαδρομή από την ρίζα του δέντρου έως το φύλο που ανήκει στην οντότητα αυτή. Αυτό αυτόματα σημαίνει την αποφυγή λαθών και διφορούμενων καταστάσεων καθώς επίσης και βελτιστοποίηση του συστήματος αφού η τεχνική αυτή υπόσχεται γρηγορότερη λειτουργία του συστήματος.

Για να κατανοήσουμε περισσότερο τον γράφο σκηνής θα κάνουμε χρήση μιας οπτικής αναπαράστασής του.

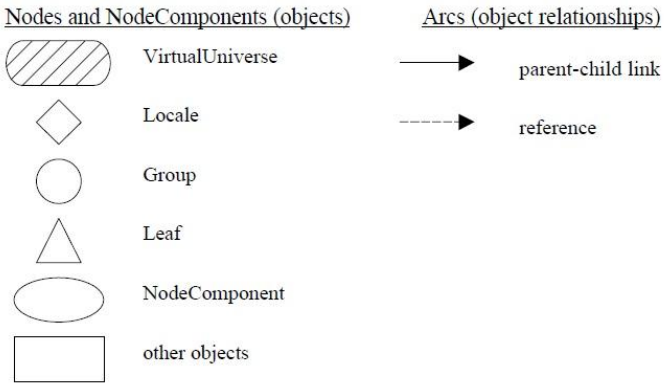


Figure 35. Βασικές οντότητες γράφου σκηνής

Ένα παράδειγμα αποτύπωσης γράφου σκηνής παρουσιάζεται στην αμέσως επόμενη εικόνα όπου παρουσιάζεται ο γράφος ενός τρισδιάστατου κόσμου που περιέχει τρείς οντότητες. Κάθε γράφος σκηνής θα πρέπει να περιέχει μια μοναδική οντότητα VirtualUniverse η οποία θα περιέχει μια λίστα από τοπικές μεταβλητές τύπου Locale. Τεχνικός είναι δυνατή η ύπαρξη παραπάνω από μιας οντότητας τύπου VirtualUniverse αλλά αυτές οι οντότητες δε θα μπορούσαν να επικοινωνήσουν μεταξύ τους και δε θα μπορούσε να υπάρξει ένα τρισδιάστατο αντικείμενο που να ανήκει και στους δύο αυτούς εικονικούς κόσμους. Όπως τα αντικείμενα VirtualUniverse έτσι και τα αντικείμενα Locale συνήθως είναι ένα ανά γράφο σκηνής.

Το αντικείμενο BranchGroup αποτελεί την οντότητα στην οποία προστίθεται όλες οι οντότητες που μπορεί να χειριστεί ο χρήστης όπως γεωμετρικό σχήμα, θέση αντικειμένου, θέση θεατή, κατεύθυνση αντικειμένου, κατεύθυνση θεατή, εμφανισιακά χαρακτηριστικά καθώς και συμπεριφορά. Οι οντότητες TransferGroup είναι υπεύθυνες για τη μεταφορά οντοτήτων στον εικονικό κόσμο και οι οντότητες Shape3D απεικονίζουν εικονικά ένα γεωμετρικό σχήμα. Οι οντότητες Shape3D μπορούν να έχουν χαρακτηριστικά όπως εμφάνιση και γεωμετρία.

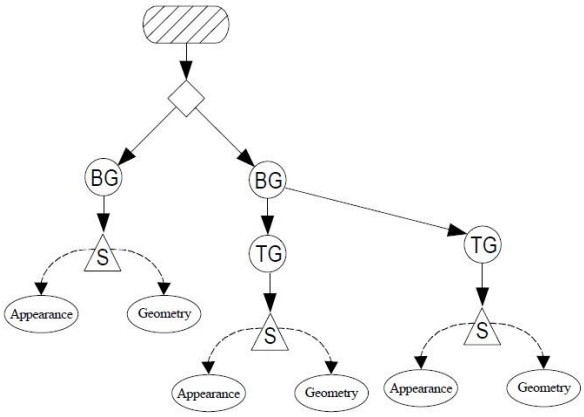


Figure 36. Σωστός Scene Graph

Στο παρακάτω σχήμα φαίνεται μια λανθασμένη αποτύπωση γράφου σκηνής. Το πρόβλημα εφάπτεται στο γεγονός ότι ένα αντικείμενο Shape3D αποτελεί απόγονο δύο οντοτήτων τύπου TransferGroup.

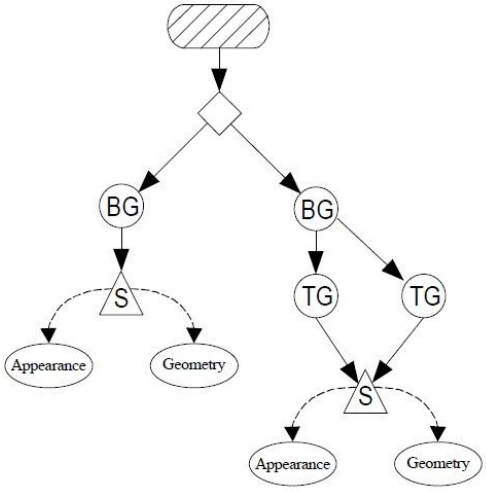


Figure 37. Μη σωστός Scene Graph

15.3 Java3D Προγράμματα

Για την επιτυχή δημιουργία προγραμμάτων στη γλώσσα αυτή μπορούν να ακολουθηθούν ορισμένοι βασικοί κανόνες που μπορούμε να εκτελέσουμε ανά βήματα.

1) Δημιουργία οντότητας Canvas3D
2) Δημιουργία οντότητας VirtualUniverse
3) Δημιουργία οντότητας Locale
4) Δημιουργία οντότητας View BranchGraph
5) Δημιουργία οντότητας Content BranchGraph
6) Μεταγλώττιση BranchGraph
7) Σύνδεση Graphs με Locale

15.4 Αντικείμενο SimpleUniverse

Η οντότητα αυτή είναι σε θέση να δημιουργήσει ένα γράφο σκηνής που περιλαμβάνει τις οντότητες VirtualUniverse και Locale καθώς επίσης και ένα κατάλληλα αρχικοποιημένο View BranchGraph. Η κλάση αυτή είναι σε θέση να χειριστεί όλες τις λειτουργίες που βρίσκονται μέσα στο μπλέ πλαίσιο στην παρακάτω εικόνα και έτσι ο χρήστης δε θα χρειαστεί να έρθει σε άμεση επαφή με αυτές.

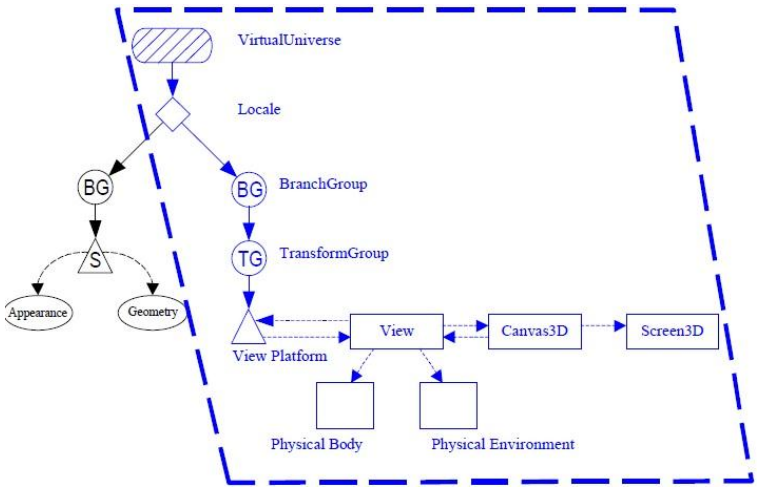


Figure 38. SimpleUniverse Package

Λαμβάνοντας υπόψη μας την παραπάνω εικόνα και τα όσα ειπώθηκαν για την κλάση SimpleUniverse ο πίνακας με τις βασικές εντολές για την επιτυχημένη δημιουργία προγραμμάτων αλλάζει στον ακόλουθο συγχωνεύοντας τα βήματα 2,3 και 4 σε ένα.

1) Δημιουργία οντότητας Canvas3D

2) Δημιουργία οντότητας SimpleUniverse

3) Δημιουργία οντότητας Content BranchGraph

4) Μεταγλώττιση BranchGraph

5) Σύνδεση Graphs με Locale

Επίσης, η κλάση SimpleUniverse δημιουργεί ένα πλήρες View BranchGraph το οποίο περιέχει ένα image plate. Η οντότητα image plate παίζει το ρόλο ενός εικονικού επιπέδου όπου οι τρισδιάστατες οντότητες αποτυπώνουν την προβολή τους ώστε να σχηματιστεί η τελική εικόνα που θα παρουσιαστεί στο χρήστη. Το έργο διαχείρισης της οντότητας αυτής αναλαμβάνει η κλάση Canvas3D. Ένα χαρακτηριστικό παράδειγμα λειτουργίας των οντοτήτων που μόλις προαναφέραμε παρουσιάζεται στην επόμενη εικόνα όπου αναφέρονται τα εικονικά αντικείμενα, ο χρήστης και ανάμεσα τους η ιδεατή επιφάνεια του Image plate.

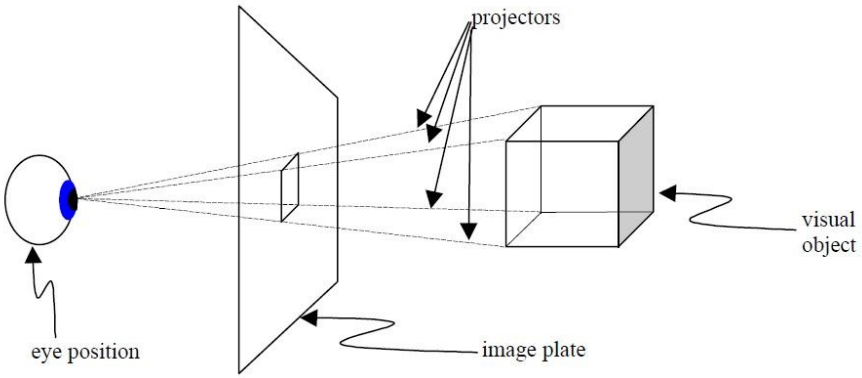


Figure 39. Κατασκευή Image Plate

Μια επιτυχημένη αρχικοποίηση της κλάσης SimpleUniverse κάνει χρήση της μεθόδου setNominalViewingTransform() η οποία θέτει κατάλληλα τη θέση παρατήρησης του χρήστη, τις διαστάσεις , την αρχική θέση που βρισκόμαστε μέσα στον εικονικό κόσμο.

15.5 Live Objects

Σε προγράμματα που είναι γραμμένα σε γλώσσα Java3D, η δημιουργία τρισδιάστατων αντικειμένων δε σημαίνει ταυτόχρονα και την ύπαρξη τους στον εικονικό κόσμο. Ο μόνος τρόπος για να συμβεί αυτό είναι η πρόσθεση αυτών ως απογόνους στις οντότητες BranchGraph. Η πρόσθεση αύτη λέμε ότι μετατρέπει τις οντότητες αυτές σε ζωντανές. Από τη στιγμή αυτή και έπειτα η οντότητα αυτή είναι παρατηρησιμη από το

χρήστη και ενδέχεται να υπάρξει αλληλεπίδραση μεταξύ τους. Πριν τα αντικείμενα γίνουν ζωντανά (Live) θα πρέπει να ρυθμιστούν κατάλληλα ορισμένες μεταβλητές ώστε ο χρήστης να μπορεί να τα μετατρέπει μετέπειτα, δηλαδή σε πραγματικό χρόνο και όταν αυτά θα αποτελούν μέρος του εικονικού κόσμου, μέσω της συνάρτησης setCapability.

15.6 Compiled Objects

Πριν οι οντότητες BranchGraph προστεθούν στον εικονικό κόσμο και γίνουν με αυτό τον τρόπο αυτό που αποκαλούμε ζωντανές οντότητες, αποτελεί καλή τεχνική το κάλεσμα της συνάρτησης Compile. Το κάλεσμα αυτό υπόσχεται μετατροπή της δομής σε μια πιο βελτιωμένη μορφή της, και έτσι η διαδικασία αποτύπωσης στην οθόνη γίνεται πιο αποδοτική. Συνιστάται η χρήση της συνάρτησης αυτής πριν από κάθε πρόσθεση οντότητας BranchGraph στην οντότητα Locale.

Οι μέθοδοι υπεύθυνοι για τον έλεγχο της παρούσας κατάστασης Live ή Compile είναι οι παρακάτω:

boolean isCompiled()
Returns a flag indicating whether the node is part of a scene graph that has been compiled.

boolean isLive()
Returns a flag indicating whether the node is part of a live scene graph.

Figure 40. Μεθόδοι ελέγχουν κατάστασης

Το API της Java3D δεν ορίζει σαφώς ποιες βελτιστοποιήσεις μπορούν να γίνουν χάρη στην μέθοδο compile. Παρ' όλα αυτά βελτιστοποιήσεις μπορούν να γίνουν με πολλούς και διαφορετικούς τρόπους. Για παράδειγμα, αν κάποια οντότητα δεν είναι παρατηρήσημη (δεν είναι μέσα στο οπτικό πεδίο του χρήστη) τότε μπορεί να αγνοηθεί πλήρως από το σύστημα ή αν υπάρχουν δύο οντότητες τύπου TransformGroup και αυτά σχετίζονται μεταξύ τους με σχέση γονέα – παιδιού, τότε η συνολική κίνηση μπορεί να γίνει και χάρη σε μία οντότητα TransformGroup.

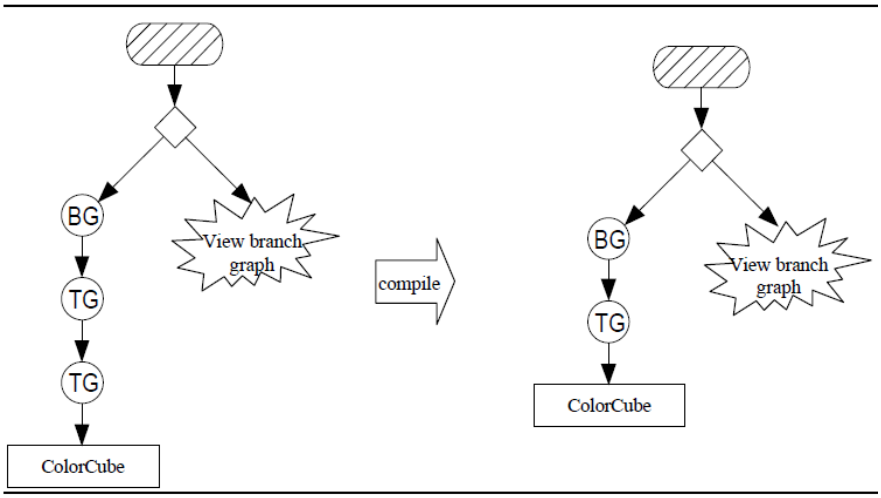


Figure 41. Compile Method Optimization

15.7 Εγκατάσταση Java3D

Η Java3D αποτελεί ένα πακέτο από κλάσης που είναι σε θέση να δημιουργήσουν και να χειριστούν τρισδιάστατα γραφικά. Αποτελεί επέκταση του Java2 JDK και ως αυτού είναι απόλυτα συμβατό με τις μετέπειτα εκδόσεις της Java. Η τελευταία έκδοση είναι η 1.5.1 και στεγάζεται στην επίσημη σελίδα της Oracle – Java[1]. Εντοπίζουμε, αποθηκεύουμε και εγκαθιστούμε το αρχείο "java3d-1_5_1-windows-i586.exe" στον υπολογιστή μας το οποίο είναι κατάλληλο για λειτουργικό σύστημα Windows. Περιβάλλον κατάλληλο, συμβατό και προτεινόμενο για ανάπτυξη προγραμμάτων σε Java3D αποτελεί το εργαλείο Netbeans.

15.8 DirectX vs OpenGL

Βασική διαφορά ανάμεσα στις δύο βιβλιοθήκες γραφικών αποτελεί το γεγονός ότι το DirectX περιέχει εργαλεία που σχετίζονται ήχο, διαδίκτυο και πολυμέσα. Από την άλλη, το OpenGL αποτελεί ένα API που απευθύνεται αποκλειστικά και μόνο στα γραφικά. Και οι δύο βασίζονται πάνω στην ίδια πλατφόρμα αρχιτεκτονικής διασωλήνωσης γραφικών. Επίσης και οι δύο περιγράφουν τις κορυφές ως ένα σετ από δεδομένα που περιέχουν τις συντεταγμένες του σημείου στο χώρο. Παρόλα αυτά χειρίζονται διαφορετικά τον συνδυασμό των κορυφών για το σχηματισμό του συνολικού γεωμετρικού σχήματος.

[1] http://www.oracle.com/technetwork/java/javase/tech/index-jsp-138252.html

Βασικό προτέρημα του OpenGL αποτελεί το γεγονός ότι καλύπτει ένα μεγάλο εύρος από πλατφόρμες και λειτουργικά συστήματα ενώ αντίθετα το DirectX περιορίζεται σε υπολογιστές Windows και στην πλατφόρμα Xbox.

Πειραματικά αντιληφθήκαμε ότι υπολογιστές βασισμένοι σε λειτουργικό σύστημα Windows έκδοση 7 ανταποκρίνονται καλύτερα κάνοντας χρήση βιβλιοθήκης DirectX παρά της OpenGL. Πιο αναλυτικά, η βιβλιοθήκη OpenGL αρχίζει να δυσκολεύεται στην ανταπόκριση έχοντας εισάγει 800 ή περισσότερα αντικείμενα στον εικονικό κόσμο. Σε αντίθεση με αυτό η βιβλιοθήκη DirectX αρχίζει να καθυστερεί στην ανταπόκριση της έχοντας εισάγει περισσότερες από 2000 οντότητες στον εικονικό κόσμο.

15.9 Πρώτο Πρόγραμμα Java3D

Στο πρώτο αυτό παράδειγμα θα προσπαθήσουμε να δούμε τα βασικά απαραίτητα βήματα για την αναπαράσταση ενός τρισδιάστατου αντικειμένου στον εικονικό κόσμο. Το πρόγραμμα αυτό δημιουργεί ένα αντικείμενο με γεωμετρικό σχήμα κύβου και επιφάνεια με διαφορετικό χρώμα σε κάθε έδρα του. Η επιφάνεια που παρουσιάζεται στον χρήστη διαθέτει κόκκινο χρώμα, καταλαμβάνει σχεδόν την μισή επιφάνεια της οθόνης και παρουσιάζεται στην εικόνα που ακολουθεί.

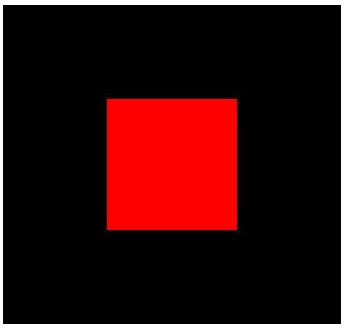


Figure 42. Διεπαφή προγράμματος Java3D

Τα βασικά βήματα που γίνονται για την δημιουργία του πρώτου αυτού του παραδείγματος είναι τα παρακάτω:

1) Δημιουργία εικονικού κόσμου που θα περιέχει τις οντότητες μας
2) Δημιουργία δομής δεδομένων που θα περιέχει τις οντότητες μας
3) Πρόσθεση του αντικειμένου μας στη δομή δεδομένων
4) Αρχικοποίησε κατάλληλα το εικονικό κόσμο και οπτική γωνία του χρήστη
5) Πρόσθεσε τη δομή δεδομένων στο εικονικό κόσμο

```
1  import com.sun.j3d.utils.universe.SimpleUniverse;
2  import com.sun.j3d.utils.geometry.ColorCube;
3  import javax.media.j3d.BranchGroup;
4  import javax.swing.JApplet;
5
6  public class Hello3d extends JApplet{
7
8      public Hello3d()
9      {
10
11         SimpleUniverse universe = new SimpleUniverse();
12
13         BranchGroup group = new BranchGroup();
14
15         group.addChild(new ColorCube(0.3));
16
17         universe.getViewingPlatform().setNominalViewingTransform();
18
19         universe.addBranchGraph(group);
20
21     }
22
23     public static void main( String[] args ) {
24         new Hello3d();
25     }
26 }
```

Code 1. Πρώτο Java3D πρόγραμμα

Το παράδειγμα αυτό μπορεί να βελτιωθεί και να εξελιχθεί σε ένα πρόγραμμα που ακολουθεί την προγραμματιστική νοοτροπία της Java3D και το οποίο φαίνεται στην επόμενη εικόνα. Η εξέλιξη οδηγεί στην χρήση του Layout Manager της Java και την πρόσθεση του τρισδιάστατου κόσμου στο κέντρο του αναδυόμενου Applet. Η χρήση Applet δεν είναι τυχαία καθώς προγράμματα που έχουν γραφτεί σε Java3D είναι συμβατά με τους περισσότερους Web Browsers [68].

Γίνεται επίσης χρήση της οντότητας Canvas3D η οποία παρέχει έναν καμβά πάνω στον οποία μπορούν να ζωγραφιστούν τρισδιάστατα αντικείμενα και το οποίο είναι σε θέση να μετατρέπει το κάθε pixel σε εικονικό ή πραγματικό.

Στο αντικείμενο BranchGroup, πριν αυτό προστεθεί στον εικονικό κόσμο καλείται η μέθοδος compile() η οποία δίνει την δυνατότητα στην Java3D να πραγματοποιήσει βελτιστοποιήσεις ώστε να είναι πιο εύκολη και γρήγορη η αποτύπωση σε πραγματικό χρόνο.

Τέλος, το πρόγραμμα αυτό ακολουθεί την πρότυπη αρχιτεκτονική της Java3D και γι' αυτό γίνεται χρήση της συνάρτησης createSceneGraph() όπου γίνεται η αρχικοποίηση των οντοτήτων που θα υπάρχουν στον εικονικό κόσμο.

```
8   public class Hello3d extends JApplet{
9
10      public Hello3d()
11      {
12          setLayout(new BorderLayout());
13          Canvas3D canvas3D = new Canvas3D(null);
14          add("Center", canvas3D);
15
16          BranchGroup scene = createSceneGraph();
17          scene.compile();
18
19          SimpleUniverse simpleU = new SimpleUniverse(canvas3D);
20          simpleU.getViewingPlatform().setNominalViewingTransform();
21          simpleU.addBranchGraph(scene);
22      }
23
24      public BranchGroup createSceneGraph()
25      {
26          BranchGroup objRoot = new BranchGroup();
27          objRoot.addChild(new ColorCube(0.4));
28          return objRoot;
29      }
30
31      public static void main( String[] args ) {
32          new Hello3d();
33      }
34  }
```

Code 2. Βελτιωμένο πρόγραμμα Java3D

15.10 Vector3f Class

Αποτελεί μια μαθηματική κλάση που στεγάζεται στο πακέτο javax.vecmath και χρησιμοποιείται για τον ορισμό μίας δομής που περιέχει τρεις αριθμούς τύπου float. Γίνεται χρήση του ως επί το πλείστον για τον ορισμό σημείου στον εικονικό κόσμο, όπου κάθε διάσταση αντιστοιχεί σε έναν από τους τρεις αριθμούς τύπου float.

15.11 Transform3D Class

Η κλάση αυτή είναι υπεύθυνη για την μεταφορά και περιστροφή εικονικού κόσμου. Η μεταφορά ορίζεται βάση float μεταβλητών και η περιστροφή βάση radians. Στιγμιότυπα αυτής της κλάσης δίνονται συνήθως ως όρισμα σε αντικείμενα τύπου TransformGroup. Κάθε φορά που γίνεται μία αλλαγή στο αντικείμενο Transform3D, αυτό θα πρέπει να ξαναφορτωθεί στο αντικείμενο TransformGroup.

15.11.1 Συνδυασμός Transformations

Υπάρχουν πολλά σενάρια όπου θα χρειαστεί να γίνει μια κίνηση που αποτελεί συνδυασμό δύο ή περισσότερων κινήσεων. Σε αυτήν την περίπτωση θα χρειαστεί να

γίνει χρήση δύο διαφορετικών μεταφορών και στο τέλος θα χρειαστεί συνδυασμός τους και πρόσθεσή τους σε ένα αντικείμενο τύπου TransformGroup.

Όπως δείχνουμε και στο παράδειγμα που ακολουθεί, ο συνδυασμός δύο αντικειμένων τύπου Transform3D σε έναν μπορεί να γίνει χάρη στην χρήση της μεθόδου mul(). Η συνάρτηση αυτή δέχεται ως όρισμα ένα αντικείμενο τύπου Transform3D και εφαρμόζεται σε ένα υπάρχον αντικείμενο του ίδιου τύπου. Έτσι, αρχικά αναλύουμε την κίνηση μας σε επιμέρους βασικές κινήσεις, έπειτα αναθέτουμε την εκάστοτε κίνηση σε μια οντότητα Transform3D, μετά συνδυάζουμε τις κινήσεις χάρη στην συνάρτηση mul() και τέλος αναθέτουμε την οντότητα Transform3D σε μία οντότητα TransformGroup που περιέχει τις τρισδιάστατες οντότητες που θέλουμε να μετακινήσουμε.

```
1. public BranchGroup createSceneGraph() {
2.         // Create the root of the branch graph
3.         BranchGroup objRoot = new BranchGroup();
4.
5.         // rotate object has composite transformation matrix
6.         Transform3D rotate = new Transform3D();
7.         Transform3D tempRotate = new Transform3D();
8.
9.         rotate.rotX(Math.PI/4.0d);
10.        tempRotate.rotY(Math.PI/5.0d);
11.        rotate.mul(tempRotate);
12.        TransformGroup objRotate = new TransformGroup(rotate);
13.
14.        objRotate.addChild(new ColorCube(0.4));
15.        objRoot.addChild(objRotate);
16.        return objRoot;
```

Code 3. Συνδυασμός Transform3D

15.12 BoundingSphere Class

Η σφαίρα οριοθέτησης αυτή χαρακτηρίζεται από το κέντρο της και από την ακτίνα της. Η συνήθης χρήση της γίνεται με κέντρο το σημείο (0, 0, 0). Η ακτίνα επιλέγεται με κριτήρια κατάλληλα ώστε να περιέχει το τρισδιάστατο αντικείμενο και όλα σημεία όπου αυτό μπορεί να μεταφερθεί.

15.13 Shape3D Class

Τα αντικείμενα αυτής της κλάσης ορίζουν σαφώς μια τρισδιάστατη εικονική οντότητα. Αποτελούν πάντα φύλλα του γράφου σκηνής. Επίσης τα στιγμιότυπα αυτής της κλάσης δεν περιέχουν πληροφορία σχετική με το σχήμα ή το χρώμα των εικονικών αντικειμένων, καθώς αυτή περιέχεται σε δομές με της οποίες συνεργάζονται τα

στιγμιότυπα αυτά. Αυτό συμβαίνει αφού η κλάση Shape3D μπορεί να αναφερθεί σε οντότητες τύπου Geometry και Appearance που επιτελούν λειτουργία σύμφωνη με το όνομα τους. Για αυτό κάθε στιγμιότυπο κλάσης Shape3D θα πρέπει να συνοδεύεται από στιγμιότυπο κλάσης Geometry και Appearance.

15.14 Geometry Classes

Η κλάση αυτή είναι υπεύθυνη για τη δημιουργία βασικών γεωμετρικών σχημάτων όπως κύβος, σφαίρα, κύλινδρος κτλ. Οι οντότητες αυτές δίνουν στο χρήστη τη δυνατότητα να της προσαρμόσει κατάλληλα βάσει των προτιμήσεων του. Ο χρήστης είναι σε θέση να διαμορφώσει κατάλληλα το σχήμα και την εμφάνιση τους. Επίσης δίνουν τη δυνατότητα στον προγραμματιστή να έχει πολλαπλά στιγμιότυπα της ίδιας κλάσης Geometry όπου το καθένα τους έχει διαφορετική εμφάνιση.

15.14.1 Box Class

Το γεωμετρικό αυτό πρωτεύων αντικείμενο είναι σε θέση να δημιουργήσει τρισδιάστατα εικονικά αντικείμενα τύπου κύβος. Το μήκος, πλάτος, ύψος είναι παραμετροποιημένα καθώς και ο τύπος της εμφάνισης τους. Τέλος κάθε επιφάνεια του κύβου διαθέτει δικά της χαρακτηριστικά τα οποία μπορεί να διαφέρουν από των άλλων επιφανειών.

15.15 GeometryArray Class

Αποτελεί υποκλάση της βασικής Geometry και χρησιμοποιείται για να δηλωθούν σημεία, γραμμές, πολύγωνα. Για παράδειγμα αν χρειαζόμασταν μια δομή δεδομένων που να περιέχει την απαραίτητη πληροφορία για τη δήλωση ενός τριγώνου θα μπορούσαμε να δημιουργήσουμε ένα στιγμιότυπο της GeometryArray με μέγεθος Array που να ισούται με τρία.

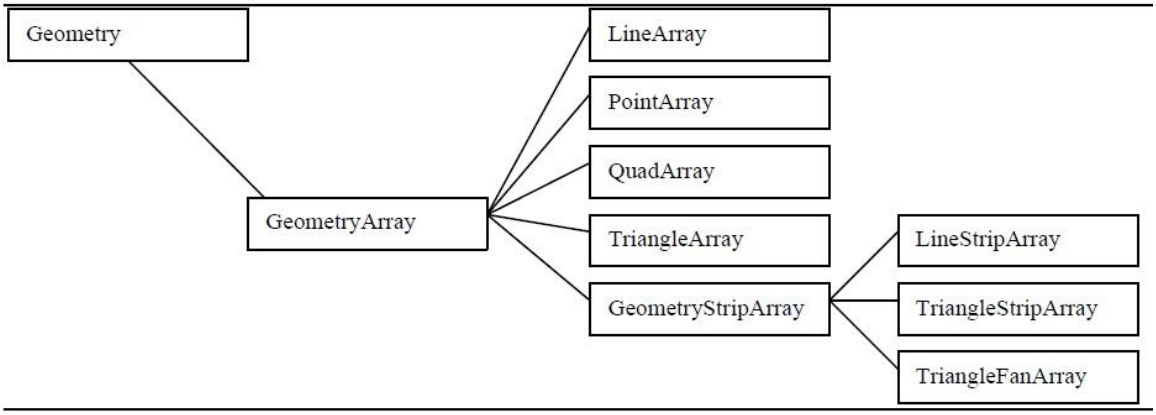


Figure 43. GeometryArray Subclasses

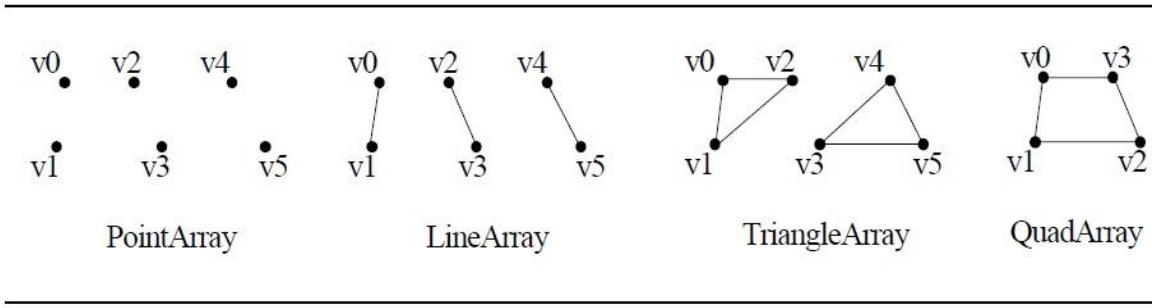


Figure 44. Οπτικοποίηση GeometryArray Subclasses

Για να είναι λειτουργικό ένα στιγμιότυπο αυτής της κλάσης θα πρέπει να εκτελεσθούν τρία βήματα:

1) Δημιουργία και αρχικοποίηση ενός κενού στιγμιότυπου (αριθμός απαραίτητων κορυφών και είδος τύπου δεδομένων που αυτές φέρουν)

2) Ανάθεση τιμών στις θέσεις του πίνακα σημείων (ορισμός σημείου στο τρισδιάστατου χώρου μέσω ανάθεση τιμών στις διαστάσεις ή χρώματος βάση RGB μοντέλου)

3) Συσχετισμός στιγμιότυπου με ένα αντικείμενο τύπου Shape3D

```
1. // construct object to represent the X axis
2. LineArray axisXLines= new LineArray (2, LineArray.COORDINATES);
3.
4. // construct object to represent the Y axis
5. LineArray axisYLines = new LineArray(2, LineArray.COORDINATES
6.                                       | LineArray.COLOR_3);
7.
8. // construct object to represent the Z axis
9. LineArray axisZLines = new LineArray(10, LineArray.COORDINATES
10.                                        | LineArray.COLOR_3);
```

Code 4. Κατασκευή GeometryArray

```
7. axisYLines.setCoordinate(0, new Point3f( 0.0f,-1.0f, 0.0f));
8. axisYLines.setCoordinate(1, new Point3f( 0.0f, 1.0f, 0.0f));
9. axisYLines.setColor(0, green);
10.axisYLines.setColor(1, green);
```

Code 5. Αρχικοποίηση GeometryArray

```
1. axisBG = new BranchGroup();
2.
3. axisBG.addChild(new Shape3D(axisYLines));
4. axisBG.addChild(new Shape3D(axisZLines));
```

Code 6. Αναφορές GeometryArray

Έτσι οδηγούμαστε τελικός στο παρακάτω μοντέλο γράφου σκηνής το οποίο περιέχει τρία αντικείμενα τύπου γραφής όπου το καθένα έχει δικιά του γεωμετρία και εμφάνιση.

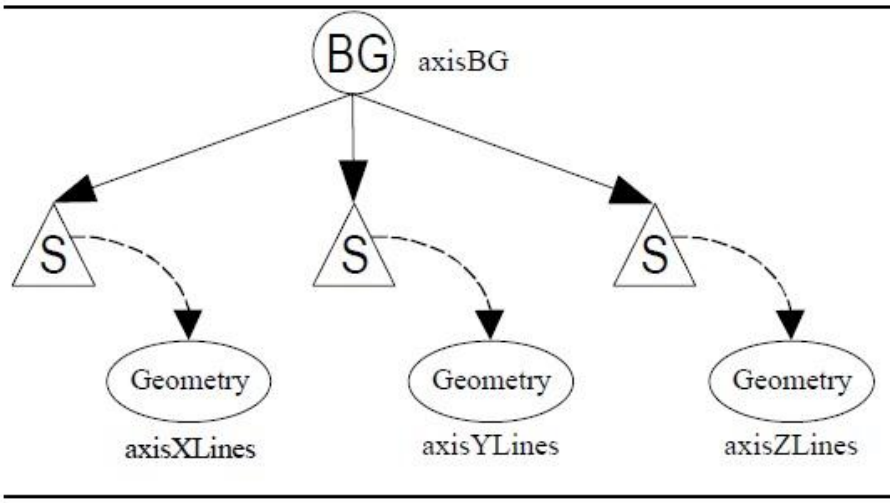


Figure 45. GeometryArray γράφος σκηνής

15.16 Appearance Class

Στιγμιότυπα αυτής της κλάσης μπορούν να συνδεθούν με οντότητες τύπου Shape3D. Οι οντότητες τύπου Geometry δεν μπορούν να περιγράψουν πλήρως την εμφάνιση ορισμένων οντοτήτων. Έτσι η χρήση στιγμιότυπων τύπου Appearance κρίνεται απαραίτητη και φέρει πληροφορία σχετική με το μέγεθος γραμμής, χρόνο, το υλικό κτλ..

15.17 Text2D Class

Ένας από τους δυνατούς τρόπους εισαγωγής κειμένου στον τρισδιάστατο εικονικό κόσμο είναι μέσω στιγμιότυπων της παρούσας κλάσης η οποία είναι σε θέση να κατασκευάσει αναπαράσταση κειμένου η οποία μπορεί να προστεθεί σε ένα Canvas3D. Η αναπαράσταση αυτή θα είναι δυσδιάστατη και έτσι το κείμενο θα φαίνεται ως ζωγραφισμένο σε ένα επίπεδο όπου δεν υπάρχει τρίτη διάσταση. Καθώς αποτελεί υποκλάση της Shape3D στιγμιότυπα της μπορούν κατευθείαν να προστεθούν σε οντότητες τύπου Branch. Οπτικό εφέ αποτελεί η δυνατότητα ύπαρξης διαφανούς φόντου έτσι ώστε γίνεται εμφανής μόνο η γραπτή πληροφορία. Προσοχή πρέπει να δείξουμε στο ότι αντικείμενα αυτού του τύπου είναι διαφανή όταν το οπτικό πεδίο του χρήστη βρίσκεται από πίσω τους.

```
18.       BranchGroup objRoot = new BranchGroup();
19.
20.       // Create a Text2D leaf node, add it to the scene graph.
21.       Text2D text2D = new Text2D("2D text is a textured polygon",
22.                                  new Color3f(0.9f, 1.0f, 1.0f),
23.                                  "Helvetica", 18, Font.ITALIC));
24.       objRoot.addChild(text2D);
```

Code 7. Δημιουργία Text2D

15.18 Σύστημα συντεταγμένων

Όταν κάνουμε χρήση του όρου "εικονικός κόσμος", αναφερόμαστε εμμέσως πλην σαφώς στην ύπαρξη τριών διαστάσεων που σχηματίζουν ένα καρτεσιανό χώρο. Το σύστημα συντεταγμένων που χρησιμοποιεί η Java3D κάνει χρήση τριών αξόνων οι οποίοι είναι κάθετοι μεταξύ τους με την ονομασία x, y, z. Ο άξονας x λαμβάνει μεγαλύτερες τιμές προς τα δεξιά, ο άξονας y προς τα πάνω και ο άξονας z προς τον χρήστη.

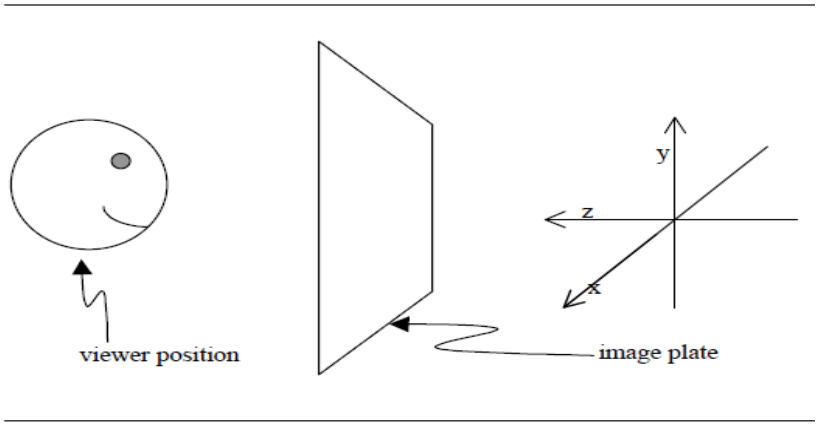


Figure 46. Προσανατολισμός Java3D Axis

15.19 User Data

Σημαντικό πλεονέκτημα της Java3D αποτελεί το γεγονός ότι όλα τα οπτικά αντικείμενα της γλώσσας φέρουν ένα αντικείμενο τύπου Object στο οποίο ο προγραμματιστής μπορεί να αποθηκεύσει απαραίτητη κατά την κρίση του πληροφορία. Για παράδειγμα όταν ο χρήστης επιλέξει ένα εικονικό αντικείμενο τότε θα μπορούσε να του παρουσιαστεί πληροφορία την οποία έχει αποθηκεύσει ο προγραμματιστής στην οντότητα αυτή με την μορφή π.χ. κειμένου. Πρόσβαση στην πληροφορία αυτή μπορεί να αποκτήσει κάποιος χάρις τις παρακάτω μεθόδους:

```
java.lang.Object getUserData()
```
Retrieves the userData field from this scene graph object.

```
void setUserData(java.lang.Object userData)
```
Sets the userData field associated with this scene graph object.

Figure 47. Μέθοδος User Data

15.20 Αλληλεπίδραση

Ορίζεται ως η ανταπόκριση του συστήματος σε εντολές του χρήστη και η ανταπόκριση του χρήστη στην πληροφορία που του παρουσιάζεται. Μέχρι στιγμής τα προγράμματα που θα μπορούσαν να φτιαχτούν χάρη στην Java3D βάσει των όσων έχουν ειπωθεί, θα ήταν όλα στατικά. Στην πλειονότητα των εφαρμογών η αλληλεπίδραση με το χρήστη κρίνεται απαραίτητη. Πέραν τούτου η διεπαφή γίνεται πιο ελκυστική και κατανοητή όταν υπάρχουν κινήσεις και ανταπόκριση σε πραγματικό χρόνο.

15.20.1 Behavior Class

Πρόκειται για μια αφηρημένη κλάση η οποία παρέχει τους μηχανισμούς για την μετατροπή του γράφου σκηνής και συνδέονται με τον κώδικα του προγραμματιστή ώστε να μεταβάλλουν κατάλληλα τον τρισδιάστατο εικονικό κόσμο. Η κλάση αυτή αντιδρά σε ερεθίσματα όπως πάτημα πλήκτρου του πληκτρολόγιου ή του ποντικιού, σύγκρουση αντικειμένων κτλ. . Τα ερεθίσματα αυτά μπορούν να χωριστούν σε δύο κατηγορίες:

- Δημιουργούμενες από τον άνθρωπο
- Δημιουργούμενες αυτόματα

Η πρώτη κατηγορία ονομάζεται Interaction όπου ο άνθρωπος δίνει εντολές στον υπολογιστή και αυτός ανταποκρίνεται, και η δεύτερη καλείται Animation όπου εδώ ο υπολογιστής ανταποκρίνεται σε ερεθίσματα που ο ίδιος δημιουργεί.

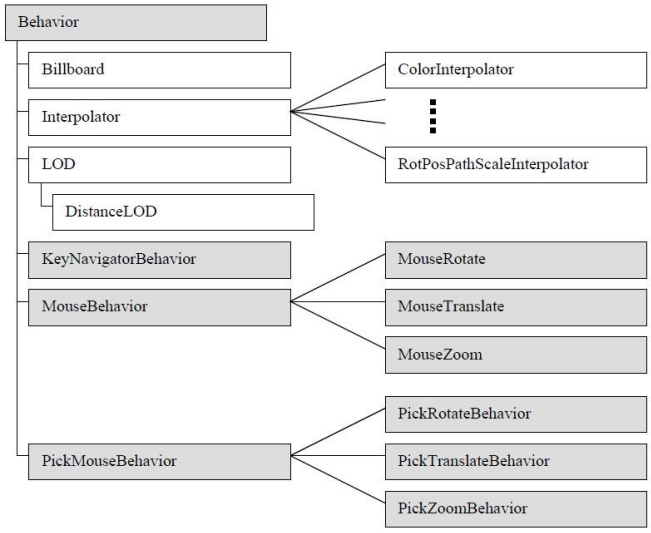


Figure 48. Ιεραρχία Behavior Class

Για να λειτουργήσει επιτυχώς η κλάση αυτή θα πρέπει να γίνουν τα παρακάτω:

1) Προετοιμασία γράφου σκηνής (προσθέτοντας οντότητα TransformGroup ή άλλης απαραίτητης οντότητας)

2) Εισαγωγή οντότητας τύπου Behavior στο γράφο σκηνής και συσχετισμός με την οντότητα στην οποία θα γίνει η αλλαγή.

3) Δήλωση σφαίρας εμβέλειας

4) Ορισμός bits ιδιοτήτων – δυνατοτήτων

Το ακόλουθο παράδειγμα δείχνει μια εφαρμογή των όσων είπαμε και σημειώνουμε με αριθμούς τα σημεία ενδιαφέροντος που αντιστοιχούν στις τέσσερις παραπάνω κατηγορίες.

```
41.    TransformGroup objRotate = new TransformGroup();
42. ❹  objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
43. ❶
44.    objRoot.addChild(objRotate);
45.    objRotate.addChild(new ColorCube(0.4));
46.
47.    SimpleBehavior myRotationBehavior = new SimpleBehavior(objRotate);
48. ❸  myRotationBehavior.setSchedulingBounds(new BoundingSphere());
49. ❷  objRoot.addChild(myRotationBehavior);
```

Code 8. Γράφος σκηνής Behavior

Προσοχή πρέπει να δειχτεί κατά τη χρήση ή κατά την επανεγγραφή της κλάσης Behavior καθώς η παραμικρή αλλαγή μπορεί να αλλάξει δραματικά την απόδοση του

συστήματος. Έτσι πρέπει να δείξουμε προσοχή καθώς παρατηρούνται δύο προβλήματα. Αυτά είναι του Memory Burn και του Unnecessary Triggering.

Memory Burn: Φαινόμενο που συμβαίνει όταν υπάρχει αδικαιολόγητη δημιουργία αντικειμένων σε προγράμματα Java. Το σύνηθες πρόβλημα που δημιουργείται είναι το καταναλώνεται πολύ μνήμη και το ότι παρατηρούνται φαινόμενα όπου σταματάει η ανανέωση στιγμιότυπων της οθόνης καθώς το σύστημα απασχολείται με λειτουργίες που επιτελεί ο garbage collector. Για να αποφευχθεί αυτό το πρόβλημα θα πρέπει να γίνει σωστή χρήση της εντολής “new”.

Unnecessary Triggering: φαινόμενο που παρατηρείται όταν ενεργοποιείται ένα event πολύ συχνά δίχως αυτό να είναι αναγκαίο σε κάθε στιγμιότυπο εικόνας που αποτυπώνεται στην οθόνη. Για παράδειγμα σε περίπτωση που θέλαμε να επιτελείται μια λειτουργία όταν το ποντίκι βρίσκεται πάνω από ένα αντικείμενο θα μπορούσαμε να ενεργοποιήσουμε το συμβάν όταν το ποντίκι θα εισέρχονταν στο χώρο που καταλαμβάνει το αντικείμενο αυτό στην οθόνη και θα απενεργοποιούνταν όταν το ποντίκι θα εξέρχονταν. Στην αντίθετη περίπτωση κακή χρήση των πόρων θα γινόταν εφόσον ενεργοποιούσαμε το συμβάν κάθε φορά που κινούνταν το ποντίκι ακόμα και αν παρέμενε εντός του αντικειμένου.

Τα κριτήρια που θα μπορούσαν να ενεργοποιήσουν ένα συμβάν στην Java3D είναι ποικίλα και περιγράφονται στον παρακάτω πίνακα:

Wakeup Criterion	Trigger
WakeupOnActivation	on first detection of a ViewPlatform's activation volume intersecting with this object's scheduling region.
WakeupOnAWTEvent	when a specific AWT event occurs
WakeupOnBehaviorPost	when a specific behavior object posts a specific event
WakeupOnCollisionEntry	on the first detection of the specified object colliding with any other object in the scene graph
WakeupOnCollisionExit	when the specified object no longer collides with any other object in the scene graph
WakeupOnCollisionMovement	when the specified object moves while in collision with any other object in the scene graph
WakeupOnDeactivation	when a ViewPlatform's activation volume no longer intersects with this object's scheduling region
WakeupOnElapsedFrames	when a specific number of frames have elapsed
WakeupOnElapsedTime	when a specific number of milliseconds have elapsed
WakeupOnSensorEntry	on first detection of any sensor intersecting the specified boundary
WakeupOnSensorExit	when a sensor previously intersecting the specified boundary no longer intersects the specified boundary
WakeupOnTransformChange	when the transform within a specified TransformGroup changes
WakeupOnViewPlatformEntry	on first detection of a ViewPlatform activation volume intersecting with the specified boundary
WakeupOnViewPlatformExit	when a View activation volume no longer intersects the specified boundary

Array 1. Κριτήρια Wakeup On

15.20.2 MouseBehavior Classes

Πρόκειται για τη βασική κλάση για όλες τις κλάσεις διαχείρισης συμβάντων του ποντικιού. Σε αυτήν την περίπτωση, το ποντίκι χρησιμοποιείται ως μέσω για την αλληλεπίδραση με οπτικά αντικείμενα. Οι λειτουργίες που περιέχονται είναι σε θέση να μετακινήσουν, να κάνουν ζουμ και να περιστρέψουν τρισδιάστατα αντικείμενα βάση των επιλογών και της κίνησης του ποντικιού. Οι υπεύθυνες κλάσεις για τις λειτουργείες αυτές παρουσιάζονται στον παρακάτω πίνακα.

MouseBehavior class	Action in Response to Mouse Action	Mouse Action
MouseRotate	rotate visual object in place	left-button held with mouse movement
MouseTranslate	translate the visual object in a plane parallel to the image plate	right-button held with mouse movement
MouseZoom	translate the visual object in a plane orthogonal to the image plate	middle-button held with mouse movement

Array 2. MouseBehavior Classes

Για να γίνει σωστή χρήση των λειτουργιών της κλάσης MouseBehavior θα πρέπει να γίνουν ορισμένες ενέργειες από την πλευρά του προγραμματιστή. Οι ενέργειες αυτές φαίνονται στα παρακάτω βήματα και δεν είναι απαραίτητο να εκτελεστούν με την σειρά που δίνονται.

1) Ενεργοποίηση δυνατοτήτων ανάγνωσης και εγγραφής του TransformGroup που στοχεύουμε να χειριστούμε.

2) Δημιουργία οντότητας MouseBehavior

3) Ανάθεση και συσχέτιση οντότητας TransformGroup σε οντότητα MouseBehavior

4) Δημιουργία σφαίρας επίδρασης μέσα στην οποία η οντότητα MouseBehavior μπορεί ασκεί δράση

5) Πρόσθεση οντότητας MouseBehavior στον γράφο σκηνής

Μία εφαρμογή των πέντε λειτουργιών που τονίσαμε παρουσιάζεται στο παρακάτω πρόγραμμα. Στο παράδειγμα αυτό εισάγεται δυνατότητα περιστροφής του τρισδιάστατου κόσμου χάρη στην λειτουργία της κλάσης MouseRotate. Βεβαίως ότι ισχύει για το παράδειγμα λειτουργίας της κλάσης MouseRotate, ισχύει και για τις κλάσεις MouseTranslate, MouseZoom:

```
    TransformGroup objRotate = new TransformGroup();
❶   objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
❶   objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);

    objRoot.addChild(objRotate);
    objRotate.addChild(new ColorCube(0.4));

❷   MouseRotate myMouseRotate = new MouseRotate();
❸   myMouseRotate.setTransformGroup(objRotate);
❹   myMouseRotate.setSchedulingBounds(new BoundingSphere());
❺   objRoot.addChild(myMouseRotate);
```

Code 9. Παράδειγμα περιστροφής MouseBehavior

15.20.3 MouseRotate Class

Χάρη στην λειτουργικότητα αυτής της κλάσης ο χρήστης είναι σε θέση να περιστρέψει τρισδιάστατα αντικείμενα που υπάρχουν στον εικονικό κόσμο. Πέραν τούτου του δίνεται η δυνατότητα να περιστρέψει ακόμα και ολόκληρο το τρισδιάστατο κόσμο μαζί με τα αντικείμενα που περικλείονται μέσα του. Βάση των προεπιλεγμένων ρυθμίσεων

η οντότητα αυτή υπακούει σε εντολές του αριστερού mouse button σε συνεργασία με κίνηση του ποντικιού.

15.20.4 MouseTranslate Class

Βάση των λειτουργιών που προσφέρει αυτή η κλάση ο χρήστης είναι σε θέση να μετακινήσει τρισδιάστατα εικονικά αντικείμενα σε ένα επίπεδο το οποίο είναι κάθετο στην ιδεατή ευθεία που υπάρχει ανάμεσα στον χρήστη και την οθόνη.

15.20.5 MouseZoom Class

Η κλάση αυτή είναι σε θέση να μετακινήσει αντικείμενα με κατεύθυνση το εικονικό βάθος της οθόνης ή προς την οθόνη σύμφωνα με τις οδηγίες που δέχεται από τον χρήστη μέσω του ποντικιού. Η μετακίνηση αυτή γίνεται πάνω στην ιδεατή ευθεία που σχηματίζεται από τον χρήστη, στην οθόνη.

15.21 Επιλογή Αντικειμένων

Σε όλα τα παραδείγματα που έχουν αναφερθεί μέχρι στιγμής είναι απλά σε θέση να παρακολουθεί τα γινόμενα στον εικονικό κόσμο και να δίνει εντολές που εφαρμόζονται σε όλο τον τρισδιάστατο κόσμο. Όμως δεν είναι σε θέση να επιλέγει και να χειριστεί ξεχωριστά και μοναδικά το εκάστοτε αντικείμενο. Γι΄ αυτό η δυνατότητα επιλογής αντικειμένων μας δίνει έναν τρόπο αλληλεπίδρασης με τα εικονικά αντικείμενα.

Στην επιλογή αυτή ο χρήστης θα πρέπει να τοποθετεί τον κέρσορα άνω του εικονικού αντικειμένου και να το επιλέγει μέσω mouse click. Τότε το πρόγραμμα θα πρέπει να είναι σε θέση να ψάχνει αυτόματα τον 3D canvas και να επιλέγει το αντικείμενο που βρίσκεται πρώτο σε επίπεδο βάθους σε σχέση με την οπτική γωνία από την οποία παρατηρεί ο χρήστης. Έτσι σχηματίζεται μία εικονική ευθεία από την θεωρητική θέση που βρίσκεται ο χρήστης, περνάει από το σημείο του έγινε κλικ πάνω στην οθόνη και καταλήγει στο τρισδιάστατο αντικείμενο. Η παρακάτω εικόνα παρουσιάζει μια μικρογραφία της διαδικασίας αυτής.

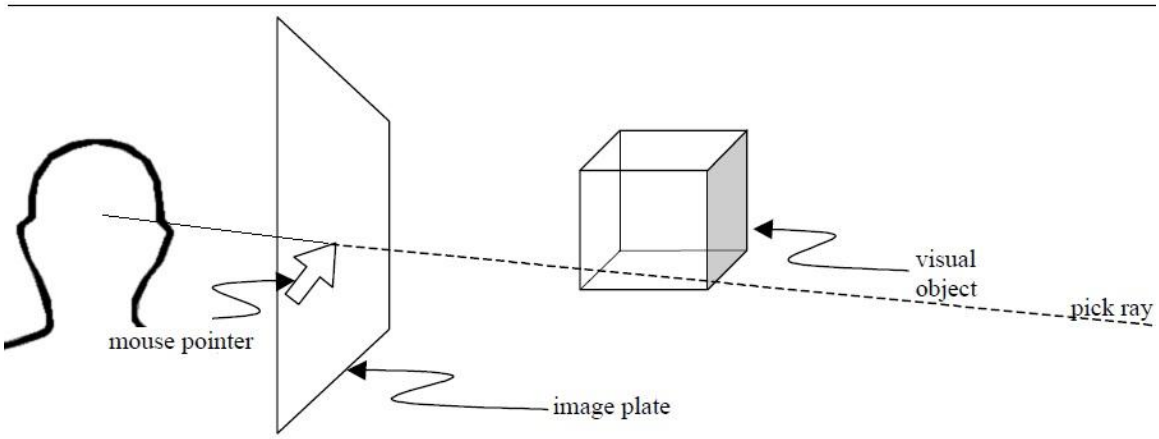


Figure 49. Προβολή PickRay

Η εύρεση του σωστού αντικειμένου που έχει επιλέξει ο χρήστης απαιτεί σημαντική επεξεργαστική ισχύ. Όσο πιο πολύπλοκος είναι ο γράφος σκηνής και όσο περισσότερα αντικείμενα έχουμε τόσο πιο δύσκολο είναι για τον υπολογιστή να βρει αυτόματα το επιλεγμένο αντικείμενο έτσι η Java3D προσφέρει ορισμένες βελτιστοποιήσεις και έξυπνες λύσεις για την αντιμετώπιση του προβλήματος. Ένα παράδειγμα αποτελεί η ύπαρξη δύο ειδών αντικειμένων. Ο χωρισμός αυτός εφάπτεται στην ύπαρξη επιλέξιμων ή μη οντοτήτων χάρη στην ύπαρξη της συνάρτησης setPickable(). Άρα αντιλαμβανόμαστε ότι τα αντικείμενα που δεν έχουν οριστεί ως επιλέξιμα δε θα ληφθούν υπόψη κατά τη διάρκεια των ελέγχων. Ένας άλλος τρόπος μείωσης της πολυπλοκότητας της διαδικασίας ελέγχου αποτελεί ο έλεγχος βάσει του ιδεατού κύβου που περιέχει την οντότητα και όχι της γεωμετρίας της. Ενδέχεται επίσης να υπάρχουν και άλλα είδη βελτιστοποιήσεων όπως ο έλεγχος των σφαιρών εμβέλειας.

Στην προσπάθεια μας να χειριστούμε επιτυχώς λειτουργίες επιλογής οντοτήτων στην Java3D θα πρέπει να προσέξουμε τα παρακάτω:

1) Δημιουργία γράφου σκηνής
2) Δημιουργία οντότητας επιλογής με τις κατάλληλες προδιαγραφές οντοτήτων αρχικοποίησης
3) Πρόσθεση της οντότητας επιλογής στο γράφο σκηνής
4) Ορισμός των κατάλληλων bits μέσω της συνάρτησης setPickable()

Ένα παράδειγμα εφαρμογής των όσων είπαμε αποτελεί ο παρακάτω κώδικας:

```
3.          BranchGroup objRoot = new BranchGroup();
4.
5.          TransformGroup objRotate = null;
6.          PickRotateBehavior pickRotate = null;
7.          Transform3D transform = new Transform3D();
8.          BoundingSphere behaveBounds = new BoundingSphere();
9.
10.         // create ColorCube and PickRotateBehavior objects
11.         transform.setTranslation(new Vector3f(-0.6f, 0.0f, -0.6f));
12.         objRotate = new TransformGroup(transform);
13.         objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
14. ❹       objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
15.         objRotate.setCapability(TransformGroup.ENABLE_PICK_REPORTING);
16.
17.         objRoot.addChild(objRotate);
18.         objRotate.addChild(new ColorCube(0.4));
19.
20. ❷       pickRotate = new PickRotateBehavior(objRoot,canvas, behaveBounds);
21. ❸       objRoot.addChild(pickRotate);
22.
23.         // add a second ColorCube object to the scene graph
24.         transform.setTranslation(new Vector3f( 0.6f, 0.0f, -0.6f));
25.         objRotate = new TransformGroup(transform);
26.         objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
27. ❹       objRotate.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
28.         objRotate.setCapability(TransformGroup.ENABLE_PICK_REPORTING);
29.
```

Code 10. Παράδειγμα αναφοράς επιλογής

15.22 Φωτισμός

Σε αυτό το σημείο θα αφιερώσουμε το χρόνο μας στην παρουσίαση τεχνικών απόχρωσης, επισκίασης, διαμόρφωσης και διαφώτισης οπτικών εικονικών αντικειμένων. Κάνοντας χρήση σωστού διαφωτισμού αντικειμένων μπορούμε να επιτύχουμε σωστή επισκίαση. Επιτυχημένη διαμόρφωση μπορούμε να έχουμε κλάση αποτύπωσης επάνω σε εικονικά επίπεδα του τρισδιάστατου κόσμου. Ρόλο επίσης παίζει ο ορισμός του σωστού είδος υλικού, φωτός καθώς και της θέσης στην οποία βρίσκεται.

Τα χρώματα που αντιλαμβανόμαστε στον πραγματικό κόσμο αποτελούν ένα συνδυασμό των φυσικών ιδιοτήτων των αντικειμένων, των χαρακτηριστικών της πηγής φωτός της σχετικής θέσης μεταξύ τους και της οπτικής γωνίας από την οποία βλέπει ο παρατηρητής. Η Java3D προσπαθεί και επιτυγχάνει σε ένα αρκετά μεγάλο βαθμό εξομοίωση των χαρακτηριστικών αυτών.

Το αποτέλεσμα διαφωτισμού βασίζεται σε τρείς παράγοντες. Πρώτων το διάνυσμα επιφάνειας (N) , η κατεύθυνση του φωτός (L) και η κατεύθυνση της γωνίας παρατήρησης του θεατή (E). επιπλέον λαμβάνονται υπόψη τα χαρακτηριστικά του φωτός και του υλικού ένα παράδειγμα εφαρμογής των όσων είπαμε αποτελεί η

παρακάτω εικόνα όπου η διαδικασία αυτή επαναλαμβάνεται για όλη την επιφάνεια της σφαίρας.

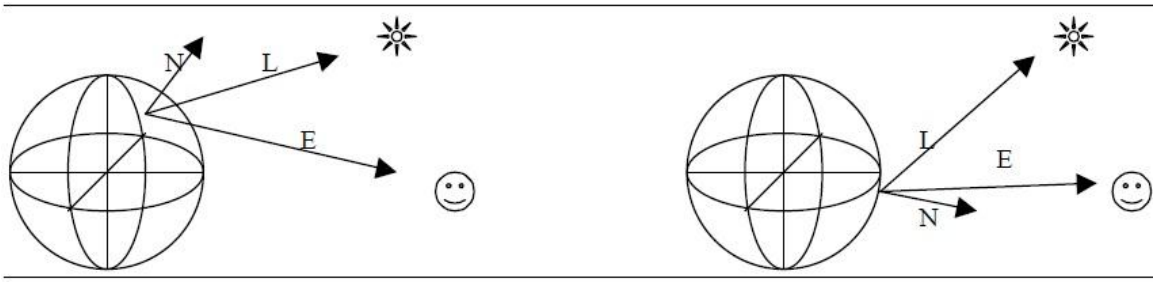


Figure 50. Ο φωτισμός βασίζεται σε διανύσματα επιφάνειας, θέση χρήστη, θέση φωτός

Η νοοτροπία του παραδείγματος αυτού αν εφαρμοστεί μπορεί να οδηγήσει στο παρακάτω αποτέλεσμα:

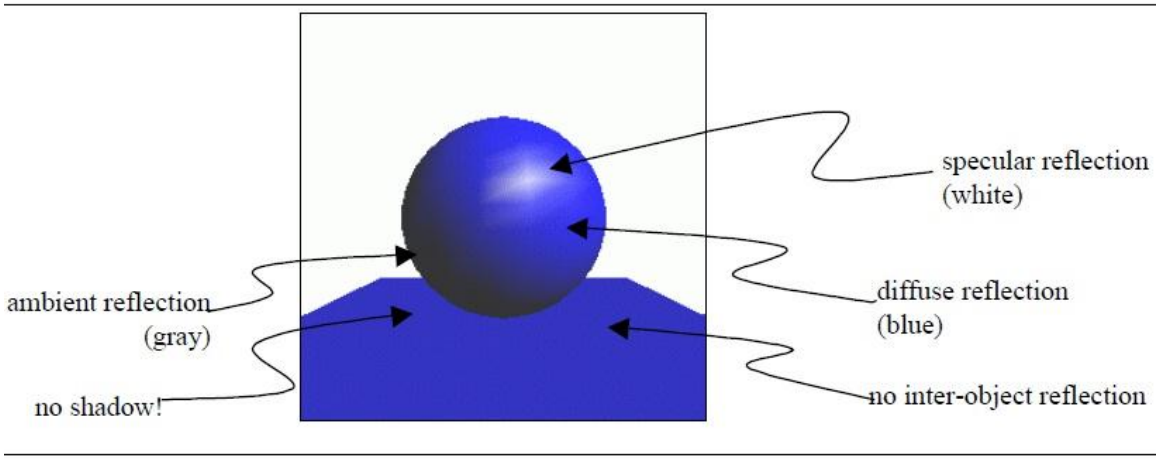


Figure 51. Φωτισμένη σφαίρα

Παρότι εξομοιώνονται πολλά χαρακτηριστικά φωτισμού που επικρατούν στον πραγματικό κόσμο, η Java3D αδυνατεί να ανταπεξέλθει σε περιπτώσεις όπου επικρατούν πολύπλοκα μοντέλα. Έτσι, αντανακλάσεις, σκιές και άλλα πολύπλοκα φαινόμενα αποκλείονται από τις δυνατότητες απεικόνισης οντοτήτων και ακόμα περισσότερο σε προγράμματα που θα πρέπει να ανταποκριθούν στις εντολές του χρήστη σε πραγματικό χρόνο.

Ο υπολογισμός των παραπάνω χαρακτηρίζεται ως εξαιρετικά πολύπλοκος αφού για παράδειγμα ένα αντικείμενο θα μπορούσε να αντανακλάσει το φώς που πέφτει πάνω του σε ν αντικείμενα και αυτά με τη σειρά τους σε ν-1 αντικείμενα. Γι' αυτό η Java3D (όπως και σχεδόν όλα τα συστήματα απεικόνισης γραφικών σε πραγματικό χρόνο) δεν λαμβάνουν υπόψη τη σχετική τους θέση των οντοτήτων.

Τα είδη φωτισμού που παρέχει η Java3D είναι:

- AmbientLight: Το φώς αυτό έχει την ίδια ένταση σε όλα τα σημεία και σε όλες τις κατευθύνσεις.
- DirectionalLight: Εξομοιώνει πολύ μακρινές πηγές φωτός όπως ο ήλιος και περιέχει το χαρακτηριστικό της κατεύθυνσης όπου όλα τα διανύσματα του φωτός είναι παράλληλα. Πληροφορία απαραίτητη για την ύπαρξη Shading.

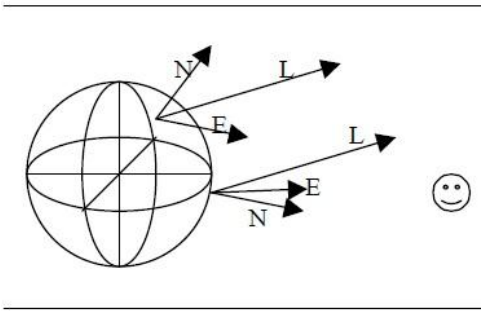


Figure 52. Directional Light

1) PointLight: πρόκειται για μια εξελιγμένη έκδοση του Directional Light. Και σε αυτή την περίπτωση το φώς είναι κατευθυνόμενο με τη διαφορά ότι έχει μια μοναδική πηγή.

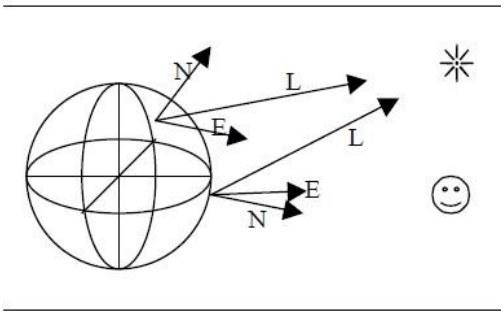


Figure 53. Point Light

2) Spot Light: Αποτελεί υποπερίπτωση της κλάσης Point Light και εξομοιώνει την επίδραση που θα είχε ένας φακός. Το φως αυτό δρα προς συγκεκριμένη κατεύθυνση και σε ένα ορισμένο εύρος γωνίας.

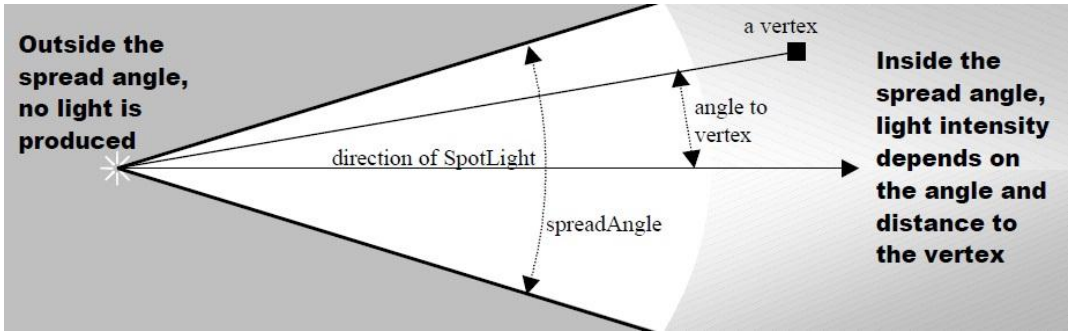


Figure 54. Spot Light

15.23 Shading

Υπάρχουν τέσσερις βασικές μορφές αυτές είναι SHADE_GOURAUD, SHADE_FLAT, FASTEST, NICEST και η καθεμία έχει τα δικά της βασικά χαρακτηριστικά. Το επιθυμητό αποτέλεσμα συνεπάγεται την εύρεση της χρυσής τομής ανάμεσα στην βέλτιστη αποτύπωση και στην ελάχιστη επεξεργαστική καταπόνηση. Ο κανόνας λέει ότι όσο πιο απλό είναι το αποτέλεσμα τόσο λιγότερη επεξεργαστική ισχύ θα χρειαστεί. Στη βέλτιστη δυνατή ποιότητα το αποτέλεσμα του Shading υπολογίζεται βάσει

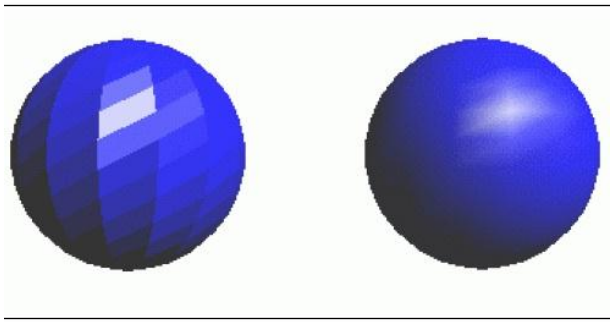


Figure 55. Flat vs Gouraud Shading

Για την επιτυχή λειτουργία του Lighting και του Shading θα πρέπει να εκτελεσθούν ορισμένα βασικά στοιχεία:

1) Φωτισμός
    a. Ορισμός σφαίρας εμβέλειας
    b. Πρόσθεση στο γράφο σκηνής
2) Εικονικά αντικείμενα
    a. Ορισμός διανυσμάτων

b. Ορισμός ιδιοτήτων υλικών

Θα πρέπει να επισημάνουμε ότι στα τρισδιάστατα εικονικά αντικείμενα της Java3D χρειάζεται να επικαλεσθούμε προγραμματιστικά τη δημιουργία των διανυσμάτων. Εναλλακτικά θα παρουσιαστούν στην οθόνη αποκλειστικά και μόνο μαύρα στο χρώμα αντικείμενα.

Το παρακάτω πρόγραμμα δείχνει τη δημιουργία μιας εικονικής σφαίρας με προεπιλεγμένες ρυθμίσεις υλικού και εμφάνισης καθώς και φωτιζόμενο από μια πηγή φωτός τύπου AmbientLight. Στο παράδειγμα αυτό κατά τη δημιουργία της σφαιρικής οντότητας ζητούμε ταυτόχρονα και τη δημιουργία διανυσμάτων επιφάνειας.

```
1.   Appearance createAppearance() {
2.     Appearance appear = new Appearance();
3.     Material material = new Material();
4.     appear.setMaterial(material);
5.
6.     return appear;
7.   }
8.
9.   BranchGroup createScene (){
10.    BranchGroup scene = new BranchGroup();
11.
12.    scene.addChild(new Sphere(0.5f, Sphere.GENERATE_NORMALS,
13.                              createAppearance()));
14.
15.    AmbientLight lightA = new AmbientLight();
16.    lightA.setInfluencingBounds(new BoundingSphere());
17.    scene.addChild(lightA);
18.
19.    return scene;
20.  }
```

Code 11. Κατασκευή διανυσμάτων Σφαίρας

15.24 Material

Η ιδιότητα αυτή δηλώνεται μέσω της οντότητας Appearance και περιέχει πληροφορία για το χρώμα, τη σκίαση των οντοτήτων, το πόσο λεία είναι η επιφάνεια και την ποσότητα του φωτός που αντανακλά. Στο παρακάτω παράδειγμα φαίνεται μια εφαρμογή των παραπάνω όπου η κάθε σφαίρα έχει διαφορετικό βαθμό αντανάκλασης.

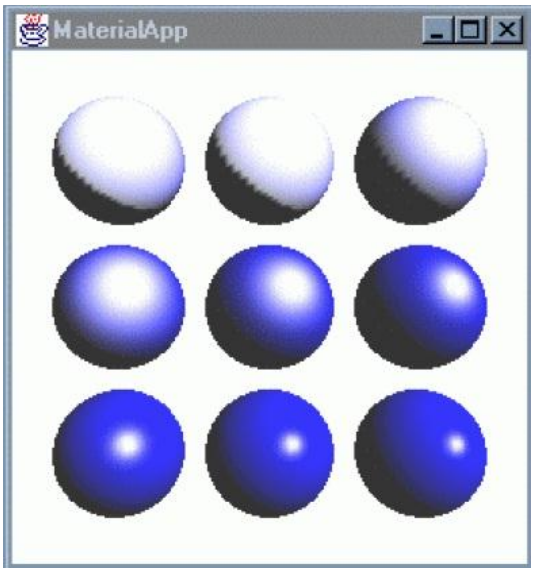


Figure 56. Different Shining Properties

15.25 Σκίαση

Χάρη στην πολυπλοκότητα δημιουργίας εφέ σκίασης και μάλιστα σε πραγματικό χρόνο η Java3D όπως και πολλά άλλα συστήματα γραφικών δεν είναι σε θέση να το υποστηρίξει. Η δημιουργία σκίασης αποτελεί πολύ πιο πολύπλοκη διαδικασία από ότι θα πιστεύαμε. Οι πηγές φωτός δεν είναι απόλυτα κατευθυντικές και επίσης οι σκιές δεν έχουν ευθύγραμμα τμήματα και απόλυτες γωνίες στις άκρες. Το πιο σημαντικό όμως είναι ότι ο αλγόριθμος αυτόματης διαμόρφωσης σκίασης θα πρέπει να ανταποκρίνεται στο οποιοδήποτε σχήμα με οποιαδήποτε γεωμετρία.

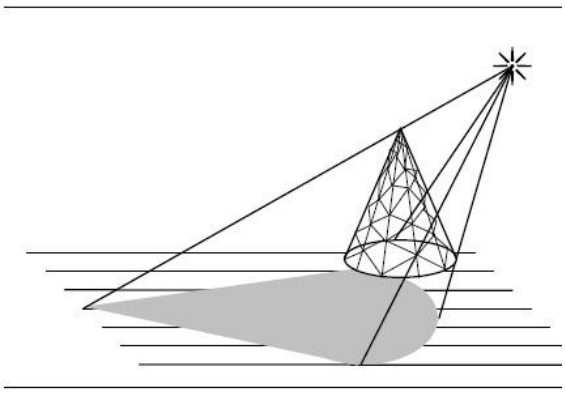


Figure 57. Προβολή σκιάς

Στην προσπάθεια δημιουργία σκιάς τα δύο πράγματα που πρέπει να λάβουμε υπόψη είναι ο υπολογισμός της θέσης που θα βρίσκεται καθώς επίσης και ο υπολογισμός του σχήματος που θα έχει. Πέραν τούτου θα πρέπει να έχουμε υπόψη μας ότι όταν κινείται ένα αντικείμενο θα πρέπει αυτόματα να κινηθεί και να μετατραπεί κατάλληλα η σκιά αυτής της οντότητας. Η μόνη λύση που προτείνεται είναι η δημιουργία των λεγόμενων ψευδό σκιών για πολύ απλά γεωμετρικά σχήματα τα οποία μένουν ακίνητα. Οι σκιές αυτές θα είναι απλές γκρίζες επιφάνειες με βασικό γεωμετρικό σχήμα όπως φαίνεται και στον παρακάτω κώδικα.

```
1. public class SimpleShadow extends Shape3D {
2.     SimpleShadow(GeometryArray geom, Vector3f direction,
3.                  Color3f col, float height) {
4.
5.         int vCount = geom.getVertexCount();
6.         QuadArray poly = new QuadArray(vCount, GeometryArray.COORDINATES
7.                                              | GeometryArray.COLOR_3
8.                                              );
9.
10.        int v;
11.        Point3f vertex = new Point3f();
12.        Point3f shadow = new Point3f();
13.        for (v = 0; v < vCount; v++) {
14.            geom.getCoordinate(v, vertex);
15.            shadow.set( vertex.x + (vertex.y-height) * direction.x,
16.                        height + 0.0001f,
17.                        vertex.z + (vertex.y-height) * direction.y);
18.            poly.setCoordinate(v, shadow);
19.            poly.setColor(v, col);
20.        }
21.
22.        this.setGeometry(poly);
23.    }
```

Code 12. Κατασκευή πολυγώνου σκιάς

Το αποτέλεσμα εκτέλεσης αυτού του κώδικα παρουσιάζεται στην παρακάτω εικόνα.

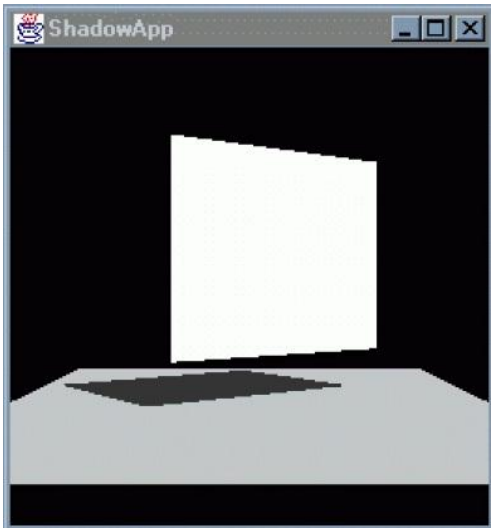


Figure 58. Δημιουργία Texture σκιάς

15.26 Textures

Αποτελεί ένα πολύγωνο που σχηματίζει μια επιφάνεια πάνω στην οποία αποτυπώνεται μια φωτογραφία. Η Java3D δίνει μεγάλη βάση στην τεχνολογία αυτή γι' αυτό και δίνονται στον προγραμματιστή πολλές επιλογές και μεγάλη ελαστικότητα λειτουργιών. Πιο κάτω αναφέρουμε τα βασικά βήματα που πρέπει να εκτελεσθούν για μια επιτυχή δημιουργία ενός Texture (όχι απαραίτητα με αυτή τη σειρά). Η γεωμετρία και η εμφάνιση του αντικειμένου θα δοθεί κατά τη δημιουργία της οντότητας τύπου Shape3D.

1) Κατάλληλη επεξεργασία εικόνας για μετέπειτα πρόσθεση στην οντότητα Texture

2) Δημιουργία και φόρτωση οντότητας τύπου Texture

3) Συσχετισμός Texture με οντότητα τύπου Appearance

4) Δήλωση συντεταγμένων και γεωμετρίας της οντότητας Texture

Σχετικά με το πρώτο βήμα, θα πρέπει να γίνουν δύο βασικές λειτουργίες. Πρώτον η εικόνα θα πρέπει να έχει τις κατάλληλες διαστάσεις πράγμα που σημαίνει ότι το μέγεθος της στην κάθε διάσταση θα πρέπει να είναι πολλαπλάσιο του δύο. Δεύτερον το είδος αρχείου στο οποίο έχει αποθηκευτεί η εικόνα θα πρέπει να είναι κατάλληλο και αναγνωρίσιμο από την Java3D όπως JPEG, GIF κτλ.

Αναφορικά με το δεύτερο βήμα, προτείνεται η χρήση οντοτήτων TextureLoader και ImageComponent.

```
1. TextureLoader loader = new TextureLoader("stripe.gif", this);
2. ImageComponent2D image = loader.getImage();
```

Code 13. Κατασκευή Texture

Στο βήμα τρία, πρέπει να αναθέσουμε στην οντότητα μας χαρακτηριστικά εμφάνισης. Η πιο απλή μορφή εμφάνισης που μπορεί να υπάρξει δίνεται στο παρακάτω παράδειγμα.

```
1.    TextureLoader loader = new TextureLoader("stripe.jpg", this);
2.    ImageComponent2D image = loader.getImage();
3.    Texture2D texture = new Texture2D();
4.    texture.setImage(0, image);
5.    Appearance appear = new Appearance();
6.    appear.setTexture(texture);
```

Code 14. Εμφάνιση Texture

Στο τέταρτο βήμα γίνεται ανάπτυξη θέσης και γεωμετρίας. Κάθε κορυφή Texture ανατίθεται σε ένα σημείο του εικονικού κόσμου χάρη στη συνάρτηση setTextureCoordinate. Στο παρακάτω παράδειγμα θα δούμε τη δημιουργία ενός Texture με τέσσερις κορυφές όπου η κάθε μια τοποθετείται σε ένα σημείο του εικονικού κόσμου και όπου στην κάθε μια ανατίθεται μια κορυφή της εικόνας προς αναπαράσταση.

```
1.  QuadArray plane = new QuadArray(4, GeometryArray.COORDINATES
2.                                   | GeometryArray.TEXTURE_COORDINATE_2);
3.  Point3f p = new Point3f();
4.  p.set(-1.0f,  1.0f,  0.0f);
5.  plane.setCoordinate(0, p);
6.  p.set(-1.0f, -1.0f,  0.0f);
7.  plane.setCoordinate(1, p);
8.  p.set( 1.0f, -1.0f,  0.0f);
9.  plane.setCoordinate(2, p);
10. p.set( 1.0f,  1.0f,  0.0f);
11. plane.setCoordinate(3, p);
12.
13. Point2f q = new Point2f();
14. q.set(0.0f, 1.0f);
15. plane.setTextureCoordinate(0, q);
16. q.set(0.0f, 0.0f);
17. plane.setTextureCoordinate(1, q);
18. q.set(1.0f, 0.0f);
19. plane.setTextureCoordinate(2, q);
20. q.set(1.0f, 1.0f);
21. plane.setTextureCoordinate(3, q);
```

Code 15. Τοποθέτηση συντεταγμένων Texture

Μια αναπαράσταση σε εικόνα της σημασιολογίας του παραπάνω κώδικα φαίνεται στην εικόνα που ακολουθεί.

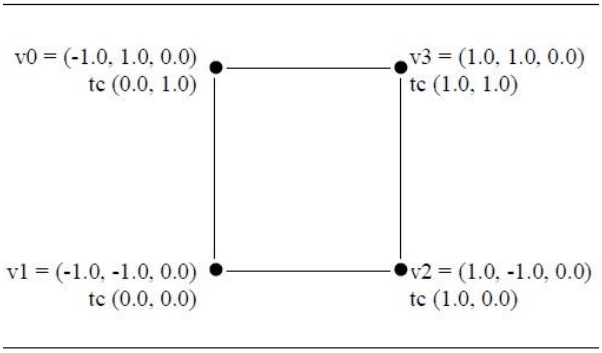
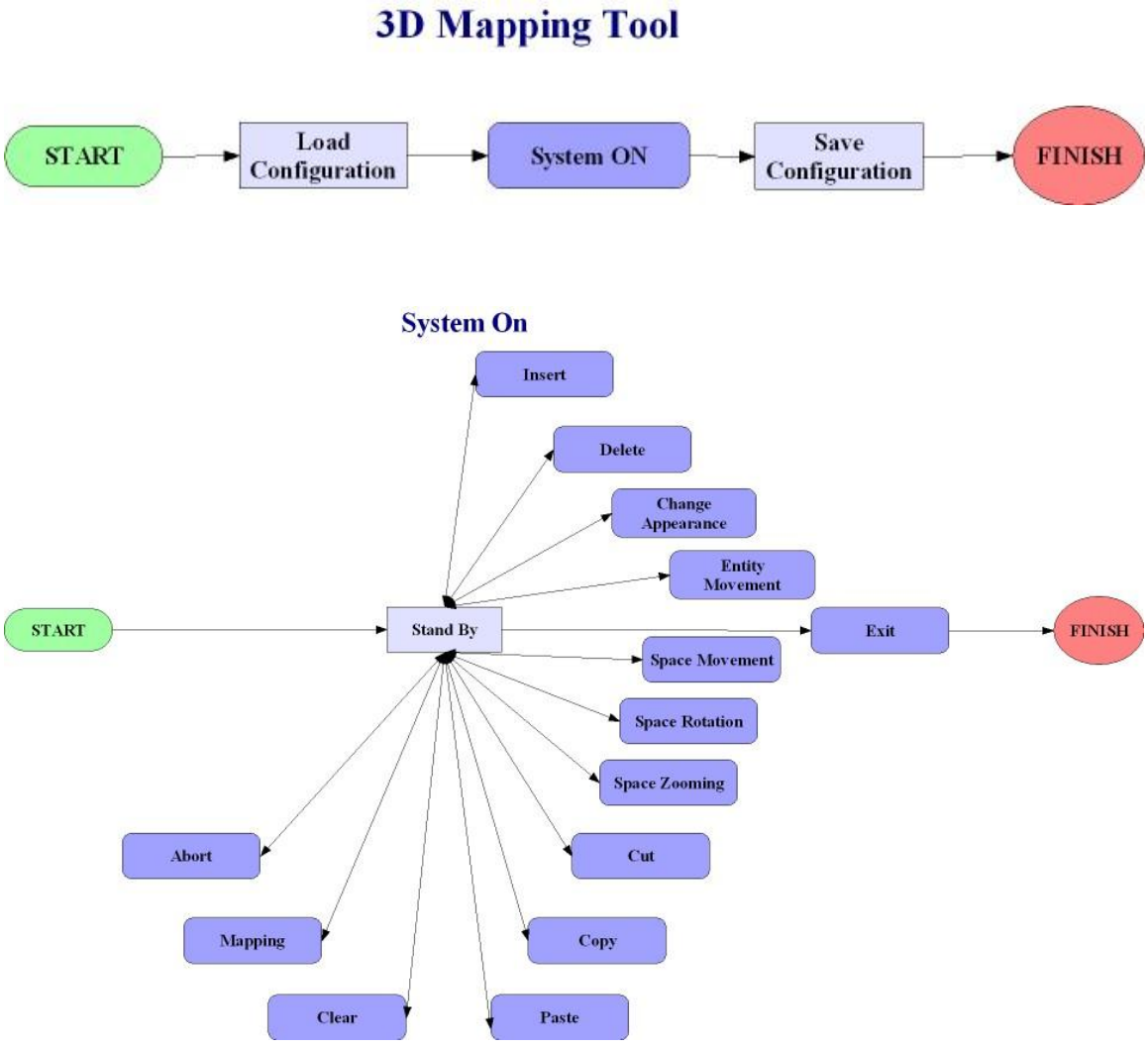
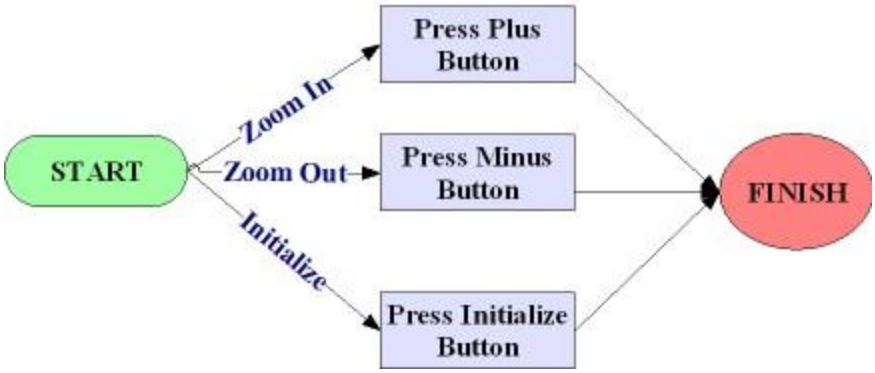


Figure 59. Coordinates And Image Set

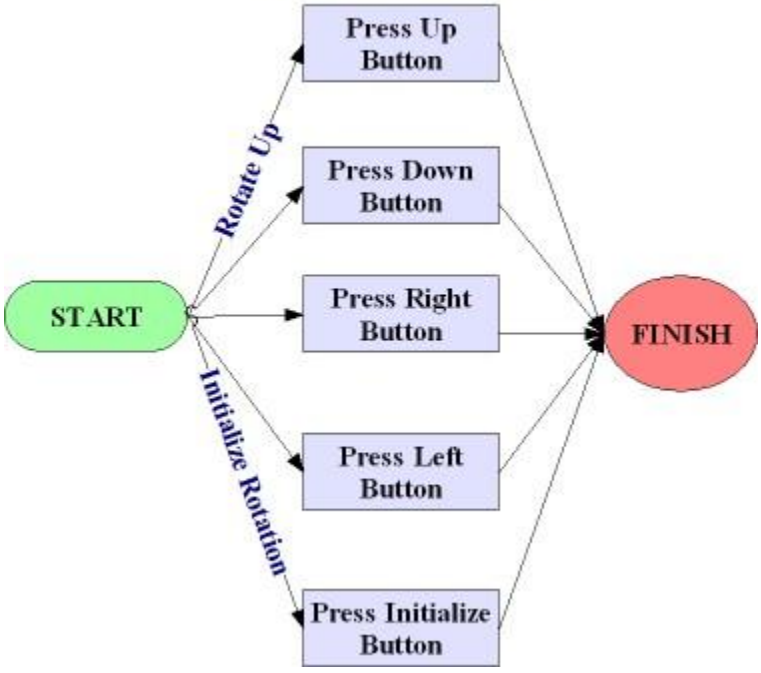
16 Διαγράμματα ροής εργασιών



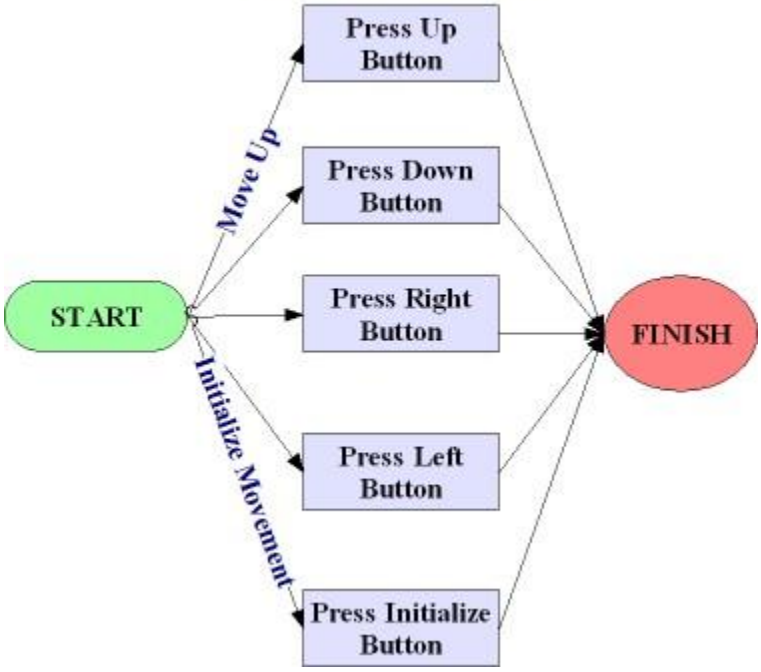
Space Zooming



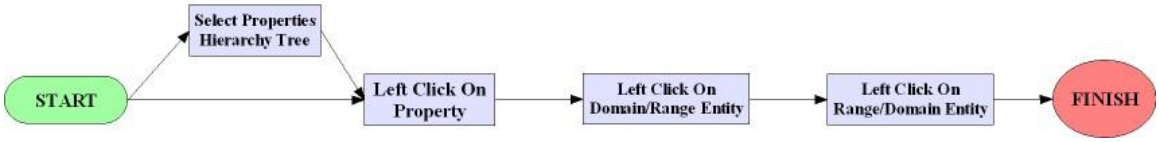
Space Rotation



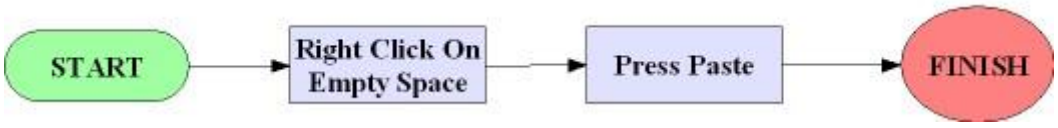
Space Movement



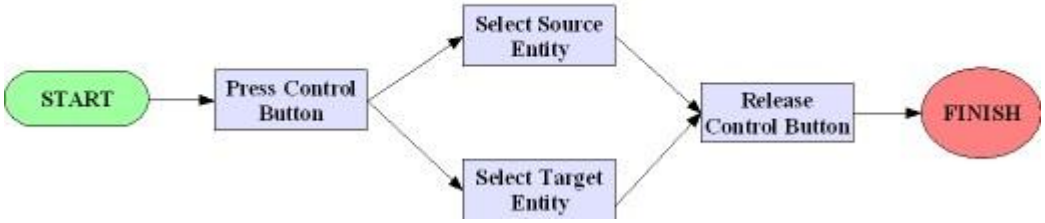
Property Manual Insert



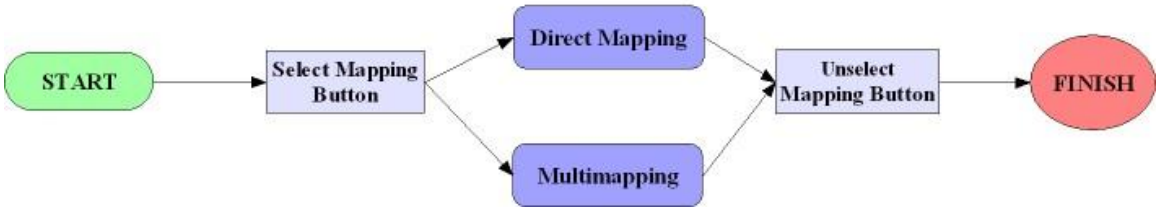
Paste



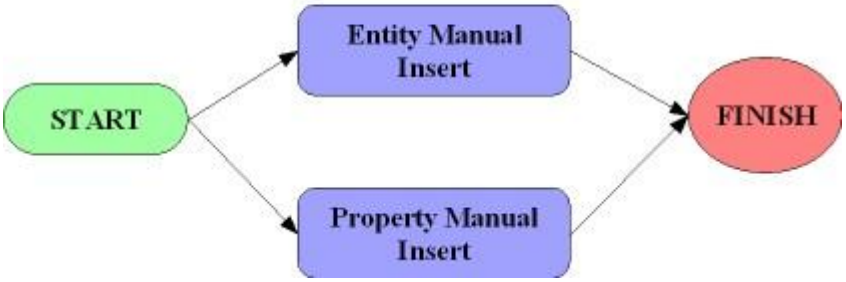
Multimapping



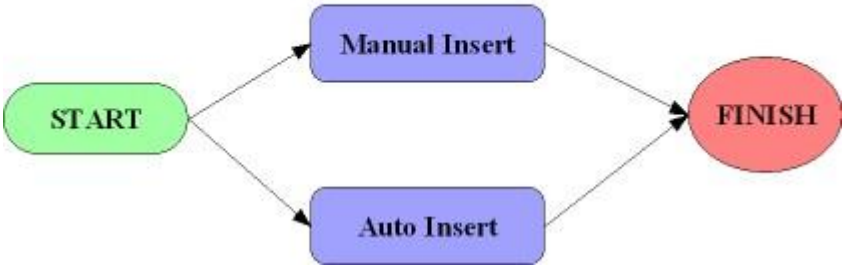
Mapping



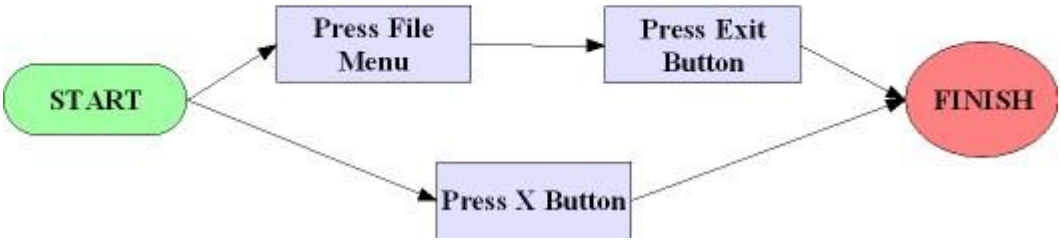
Manual Insert



Insert

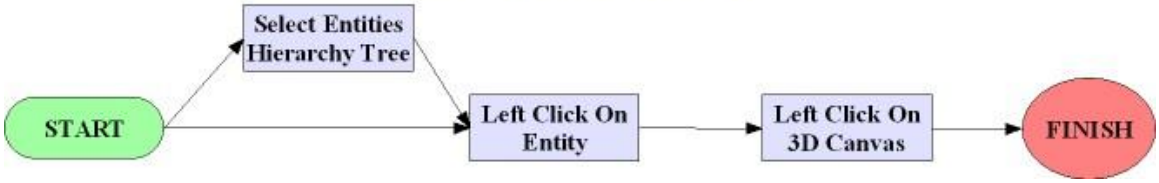


Exit

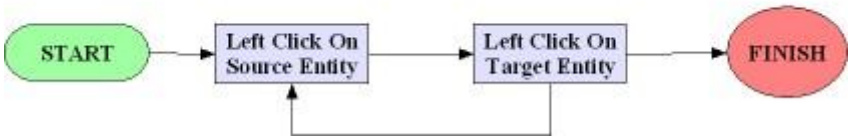


Entity Movement

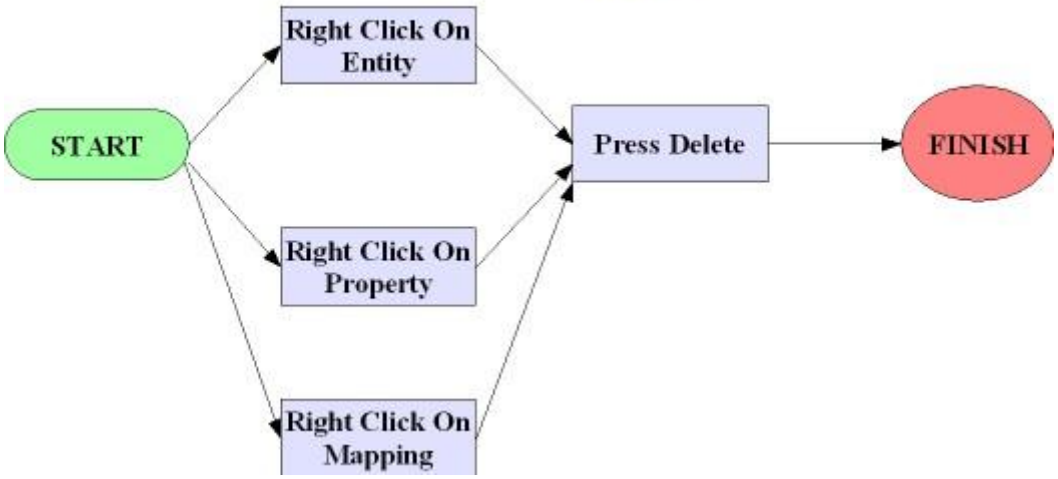
Entity Manual Insert



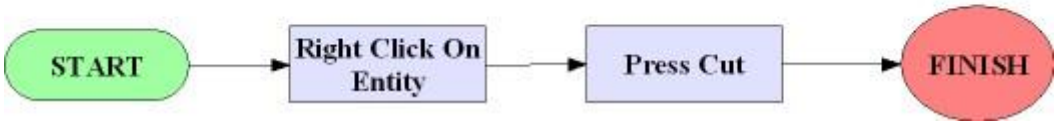
Direct Mapping



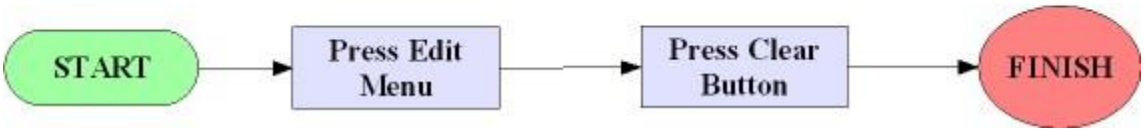
Delete



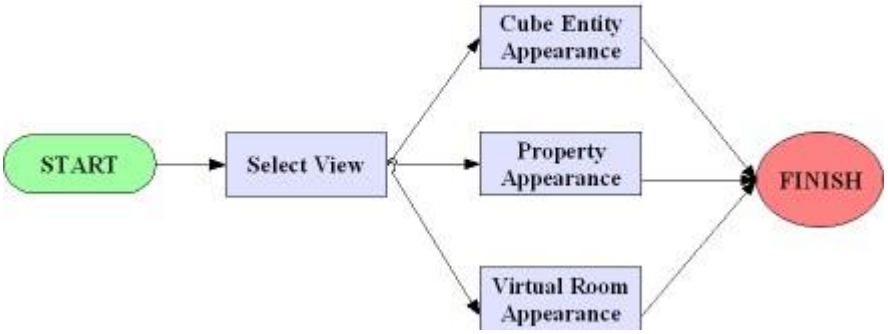
Cut



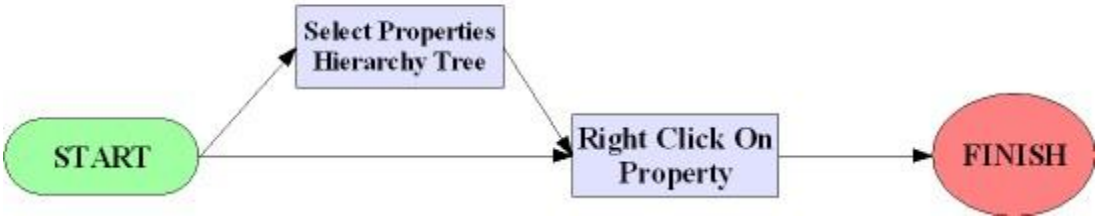
Clear



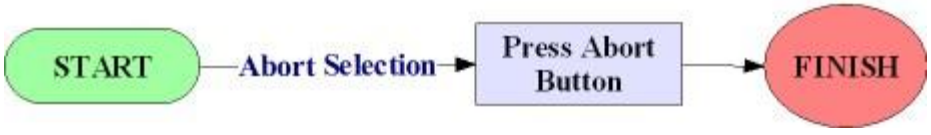
Change Appearance



Auto Insert



Abort



17 Εφαρμογή

Στην εφαρμογή μας δίνονται στον χρήστη όλες οι δυνατότητες και λειτουργίες που πιστεύουμε ότι ενδέχεται να χρειαστεί – εκτελέσει. Αυτές είναι:

- της εισαγωγής οντοτήτων και σχέσεων
- αυτόματης εισαγωγής οντοτήτων και σχέσεων
- αυτόματης αναπροσαρμογής των σχέσεων που μπορούν να εισαχθούν ανάλογα με τις οντότητες που υπάρχουν ήδη στον εικονικό κόσμο
- έλεγχος λαθών χρήστη κατά την διαδικασία της εισαγωγής
- επιλογής εικονικού αντικειμένου
- διαγραφής οντότητας, σχέσης ή αντιστοίχησης
- μεταφοράς οντότητας και αυτόματης μεταφοράς όλων των σχέσεων που συνδέονται με αυτήν
- ταχείας μεταφοράς αντικειμένου
- μετατροπής εμφανισιακών χαρακτηριστικών οντοτήτων και σχέσεων
- μεταφοράς – περιστροφής – αυξομείωσης απόστασης του κέντρου του εικονικού σύμπαντος
- αντιστοίχησης ανάμεσα σε οντότητες και σχέσεις που ανήκουν σε διαφορετικά μοντέλα
- πολλαπλής αντιστοίχησης ανάμεσα σε οντότητες και σχέσεις που ανήκουν σε διαφορετικά μοντέλα
- φόρτωσης εναλλακτικών μοντέλων σχημάτων
- εισαγωγής περιορισμών για την μετέπειτα αντιστοίχηση στιγμιοτύπων των σχημάτων
- εξαγωγής σε XML του αποτελέσματος της αντιστοίχησης ανάμεσα σε Source και Target Model.

17.1 Layout

Η διεπαφή επικοινωνίας με τον χρήστη κάνει χρήση τεσσάρων βασικών πλαισίων τα οποία είναι Virtual Reality Panel, Info Label, Entity/Property Selection Panel, Functions Panel.

Στην οντότητα Virtual Reality Panel προστίθεται η οντότητα Canvas3D που φέρει την οντότητα SimpleUniverse. Έτσι το Virtual Reality Panel περιέχει όλο το εικονικό σύμπαν (ή για να είμαστε ακριβείς την προβολή του) και εμείς μπορούμε πλέον να στέλνουμε μηνύματα όπως πρόσθεση εικονικού αντικειμένου, μετακίνηση, περιστροφή κτλ.

Θα πρέπει να επισημάνουμε ότι η οντότητα Canvas3D ρυθμίστηκε με τρόπο κατάλληλο ώστε ο ελάχιστος χρόνος παραμονής – εμφάνισης – υπολογισμού δεδομένων για το κάθε frame να οριστεί στα 20 ms. Αυτό σημαίνει ότι το εκάστοτε frame θα παραμένει το λιγότερο 20 millisecond στην οθόνη του χρήστη και όλο το σύστημα συγχρονίζει σε το πολύ 50 Hertzs. Αυτό κρίνεται ως επιθυμητό αποτέλεσμα καθώς υπερβαίνει τα 25 frames το δευτερόλεπτο που είναι το κατώτατο όριο το οποίο αντιλαμβάνεται το ανθρώπινο μάτι ως συνεχόμενη κίνηση. Επίσης πλησιάζει αρκετά την συνήθη χρήση των τεχνολογιών συγχρονισμού οθόνης σημερινών ηλεκτρονικών υπολογιστών όπου ο συνήθης χρονισμός αναφέρεται στα 60 Hertzs.

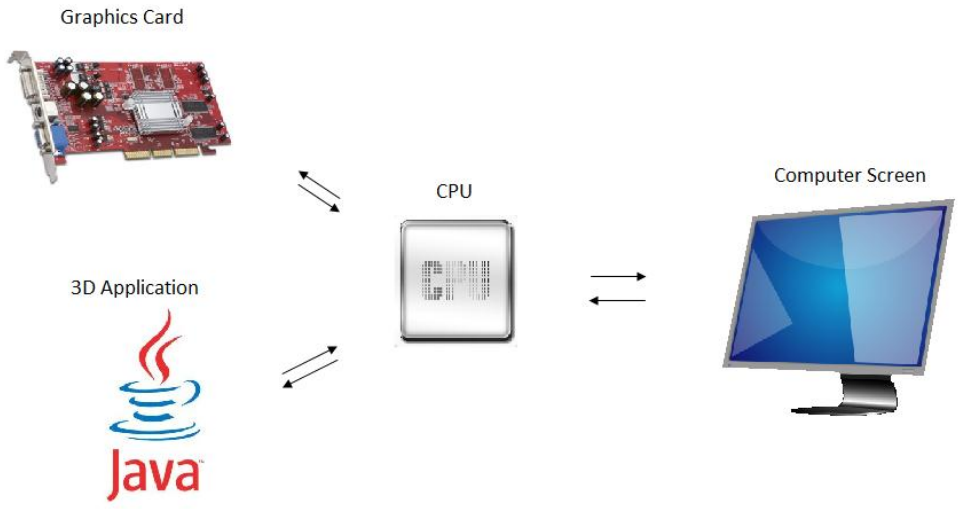


Figure 60. Κυκλική διαδρομή στιγμιότυπου εικόνας

Ο λόγος που χρειάστηκε να επέμβουμε στο σύστημα και να περιορίσουμε τον αριθμό των frames που προβάλλονται ανά δευτερόλεπτο είναι ότι παρατηρήθηκε μεγάλη επιβάρυνση επεξεργαστή και μνήμης. Ένα πολύ απλό παράδειγμα όπου προβαλλόταν στον εικονικό κόσμο ένα και μοναδικό αντικείμενο ήταν σε θέση να επιβαρύνει τον επεξεργαστή κατά 60% - 70% και να παρατηρείτε χρονοκαθυστέρηση στη ανταπόκριση του συστήματος. Αιτία αυτού του προβλήματος αποτελεί το γεγονός ότι το σύστημα

υπολόγιζε την αναπαράσταση περίπου τετρακοσίων «400» frames το δευτερόλεπτο δίχως αυτό να είναι απαραίτητο.

Δεύτερη οντότητα αποτελεί η Info Label όπου παρουσιάζονται μηνύματα στον χρήστη όταν αυτός πρέπει να ειδοποιηθεί για κάποιο event.

Τρίτη οντότητα αποτελεί η Entity/Property Selection Panel όπου παρουσιάζονται με κληρονομική ιεραρχική δομή δέντρου οι οντότητες και οι σχέσεις ανάμεσα στις οντότητες. Θα πρέπει να τονίσουμε ότι επειδή υποστηρίζετε πολλαπλή κληρονομικότητα μία οντότητα ενδέχεται να παρουσιαστεί ως απόγονος περισσότερων από ένα οντοτήτων. Δηλαδή μία οντότητα ενδέχεται να παρουσιαστεί παραπάνω από μία φορές στο ιεραρχικό δέντρο. Επίσης το δέντρο ιεραρχίας των σχέσεων αναδημιουργείτε σε πραγματικό χρόνο ανάλογα με τις οντότητες που υπάρχουν στον εικονικό κόσμο. Δηλαδή στο δέντρο ιεραρχίας των σχέσεων υπάρχουν μόνο οι σχέσεις που έχουν domain/range entity ή resource/value οντότητες που υπάρχουν στον εικονικό κόσμο. Κάθε φορά που εισάγεται ή διαγράφεται οντότητα στον εικονικό κόσμο το δέντρο σχέσεων δημιουργείται εξαρχής.

Τέλος υπάρχει το panel Functions Panel όπου φιλοξενούνται οι λειτουργίες που αναφέρονται στην μεταφορά του εικονικού σύμπαντος, την περιστροφή του, την μεγέθυνση και την απομάκρυνση. Επιπλέον, υπάρχει το κουμπί Abort το οποίο ματαιώνει την επιλογή κάποιου entity ή property προς εισαγωγή στο τρισδιάστατο καμβά. Τέλος, υπάρχει η επιλογή Mapping η οποία θέτει το πρόγραμμα σε κατάσταση αντιστοίχησης οντοτήτων.

Η αντιστοίχηση οντοτήτων εργάζεται σε ομάδες των δύο. Η μία οντότητα θα πρέπει να ανήκει στην πηγή και η άλλη στο μοντέλο προορισμού. Ο χρήστης καλείται να επιλέξει αντιστοίχηση ανάμεσα σε οντότητες πηγής και προορισμού επαναληπτικά.

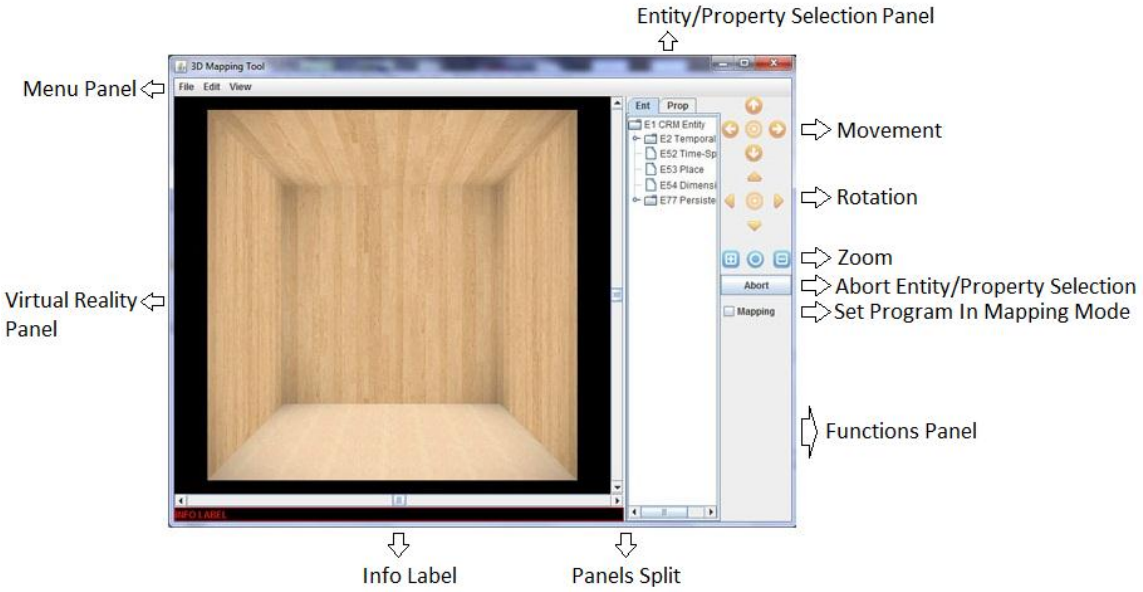


Figure 61. 3D Mapping Tool Window Layout

17.2 Επίπεδα Παραθύρων (Layered Pane)

Κατά την διάρκεια εκτέλεσης η εφαρμογή βασίζεται στην ύπαρξη, λειτουργία και συνεργασία τριών επιπέδων. Ο λόγος ύπαρξης παραπάνω του ενός επιπέδου είναι ο διαχωρισμός γραφικών οντοτήτων της Java σε Heavy Weight και Light Weight με τις πρώτες να είναι σε θέση να επισκιάσουν οντότητες τις δεύτερης κατηγορίας. Δεδομένου αυτού, δεν θα μπορούσαμε να εμφανίσουμε οντότητες Light Weight όπως JPanel μπροστά από οντότητες Heavy Weight όπως Canvas3D.

Ο λόγος που τοποθετήθηκε οντότητα μπροστά από το Canvas3D είναι ότι επιθυμούμε να εμφανίζεται στον χρήστη ένα label με όλη την διαθέσιμη πληροφορία για την οντότητα. Όταν το ποντίκι βρεθεί επάνω από μια οντότητα του τρισδιάστατου κόσμου τότε ενεργοποιείτε ένα thread που αναθέτει την εμφάνιση του label στο χρόνο και την παρουσίαση του έπειτα από 500 ms.

Το label προστίθεται στο επίπεδο layer μηδέν «0» καθώς θα πρέπει να εμφανιστεί πρώτο από όλα τα υπόλοιπα αντικείμενα. Πέραν τούτου θα πρέπει να υπολογιστεί και η κατάλληλη θέση για το αντικείμενο καθώς θα πρέπει να βρίσκεται πλησίον του εικονικού αντικειμένου. Γι’ αυτό η θέση του label υπολογίζεται βάση του σημείου που βρισκόταν ο κέρσορας κατά την στιγμή όπου βρέθηκε μπροστά από το εικονικό αντικείμενο.

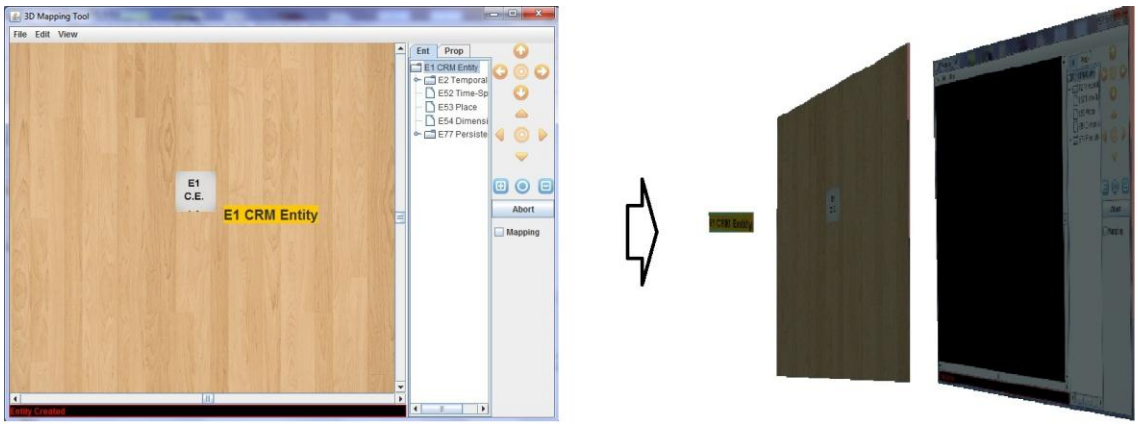


Figure 62. Επίπεδα παραθύρων

17.3 Εικονικό Δωμάτιο

Η εφαρμογή μας βασίστηκε στην δημιουργία ενός εικονικού δωματίου το οποίο θα περιέχει τις οντότητες που εισάγονται στον τρισδιάστατο κόσμο. Το εικονικό δωμάτιο δημιουργήθηκε ώστε να βελτιωθεί η ρεαλιστικότητα, η επίγνωση, η διαισθητικότητα και η κατανόηση του χρήστη καθώς και η οργάνωση θέσης των οντοτήτων.

Το εικονικό δωμάτιο αποτελείται από έναν κύβο με πέντε έδρες, οκτώ κορυφές, πέντε textures και δύο φωτογραφίες που εξομοιώνουν το πάτωμα και τους τοίχους. Τέσσερα textures τα οποία είναι τα ceiling, right wall, center wall, left wall φέρουν εμφάνιση κατάλληλη για τοίχο, ενώ το texture floor φέρει την φωτογραφία του δαπέδου.

Καθώς στον τρισδιάστατο χώρο τι εστί πάνω και τι εστί κάτω είναι έννοιες υποκειμενικές, αφού τα χαρακτηριστικά αυτά μπορούν να αντιστραφούν με ένα απλό Rotation, το δάπεδο φέρει διαφορετική σχεδίαση ώστε ο χρήστης να έχει την αίσθηση του προσανατολισμού.

Πιο αναλυτικά, στο κέντρο του δωματίου υπάρχει το κέντρο του εικονικού σύμπαντος, δηλαδή το σημείο (0 , 0 , 0). Επίσης η εφαρμογή φέρει μια μεταβλητή που αρχικοποιείτε κατά την έναρξη του προγράμματος και βάση αυτής δημιουργείτε το εικονικό δωμάτιο. Η μεταβλητή αυτή ονομάζεται limit και φέρει το επιθυμητό μέγεθος του δωματίου. Έτσι βάση αυτής αρχικοποιούνται οι οκτώ κορυφές όπως δείχνουμε παρακάτω.

```
Point3f point_0 = new Point3f (-limit, -limit, limit);

Point3f point_1 = new Point3f (limit, -limit, limit);

Point3f point_2 = new Point3f (limit, limit, limit);

Point3f point_3 = new Point3f (-limit, limit, limit);

Point3f point_4 = new Point3f (-limit, -limit, -limit);

Point3f point_5 = new Point3f (limit, -limit, -limit);

Point3f point_6 = new Point3f (limit, limit, -limit);

Point3f point_7 = new Point3f (-limit, limit, -limit);
```

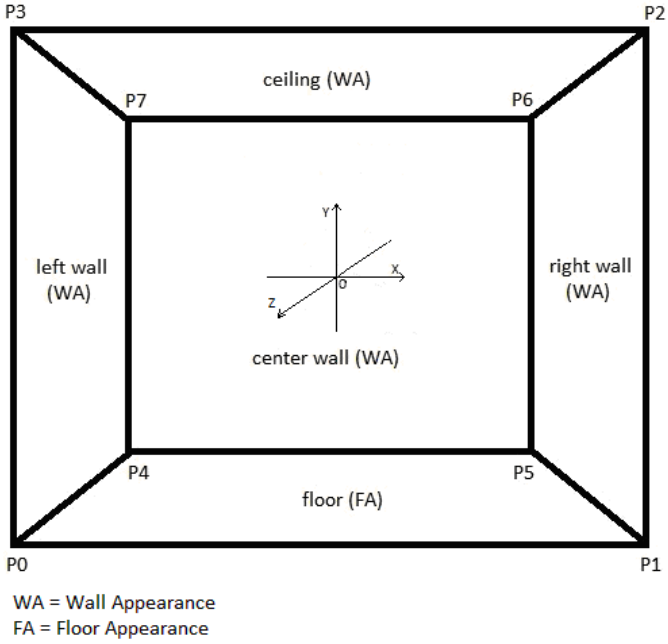


Figure 63. Σχεδίαση εικονικού δωματίου

Για μία επιτυχή αποτύπωση ενός εικονικού δωματίου θα πρέπει να λάβουμε υπόψη μας την σωστή διαχείριση των οντοτήτων τύπου texture που προσομοιώνουν τους τοίχους και το πάτωμα. Γι' αυτό και το διάνυσμα χειρισμού κατεύθυνσης πρόσοψης του αντικειμένου τύπου texture Θα πρέπει να συμπεριφέρεται κατάλληλα και να προβλέπει σωστή κατεύθυνση. Ως προεπιλογή, η κατεύθυνση διανύσματος ορίζεται στον άξονα X από αριστερά προς δεξιά κατά αύξουσα κατεύθυνση τιμών, στον άξονα Y από κάτω προς τα πάνω κατά αύξουσα κατεύθυνση τιμών και στον άξονα Z από μέσα προς τα έξω όπως η κατεύθυνση αύξοντα αριθμού.

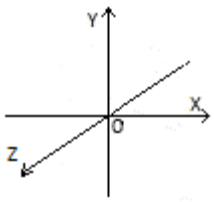


Figure 64. Διανύσματα εικονικού χώρου

Σε περίπτωση που επιθυμούμε την χρήση της προεπιλεγμένης κατεύθυνσης διανύσματος texture τότε ο ορισμός κορυφών μπορεί να διαθέτει την μορφή που δίνεται στο παρακάτο παράδειγμα κώδικα στην υποπερίπτωση Α. Εναλλακτικά, εάν αυτό δεν μας εξυπηρετεί μπορούμε να αντιστρέψουμε την φορά κατεύθυνσης του διανύσματος. Αυτό μπορεί να επιτευχθεί αντιστρέφοντας την σειρά ανάθεσης κορυφών όπως στην υποπερίπτωση Β που ακολουθεί.

```
center_wall.setCoordinate (0, point_4);
center_wall.setCoordinate (1, point_5);
center_wall.setCoordinate (2, point_6);
center_wall.setCoordinate (3, point_7);
```

A

```
ceiling.setCoordinate (3, point_3);
ceiling.setCoordinate (2, point_2);
ceiling.setCoordinate (1, point_6);
ceiling.setCoordinate (0, point_7);
```

B

Code 16. Προεπιλεγμένη/Αντίστροφη κατεύθυνση διανυσμάτων

Κάνοντας πράξη τους κανόνες περί διανυσμάτων textures και βάση του αποτελέσματος που επιθυμούμε για την δημιουργία του εικονικού δωματίου καταλήγουμε στο ότι οι έδρες floor, center wall και left wall ακολουθούν την προεπιλεγμένη κατεύθυνση διανυσμάτων, ενώ οι έδρες right wall και ceiling αντιστρέφουν την κατεύθυνση του διανύσματος κατεύθυνσης επιφάνειας.

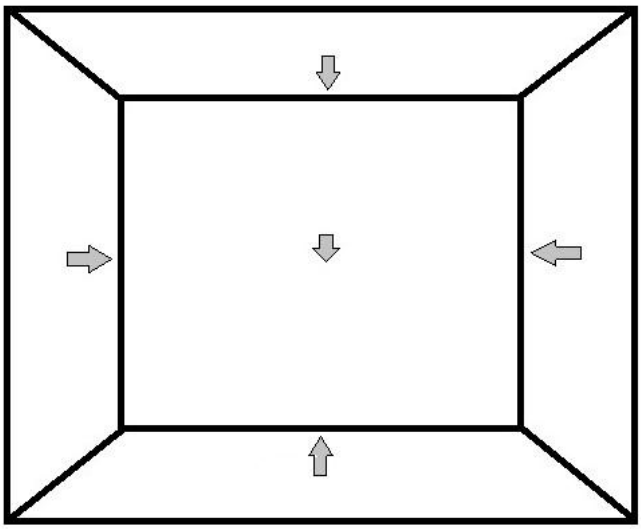


Figure 65. Διανύσματα εικονικού δωματίου

Βασιζόμενοι πάνω σε τεχνικές αύξησης ρεαλιστικότητας αναπαράστασης οντοτήτων στον τρισδιάστατο χώρο χρειάστηκε να επεξεργαστούμε ψηφιακά τις εικόνες που χρησιμοποιούνται για ταπετσαρίες επιφανειών του εικονικού κόσμου. Πειράματα έχουν δείξει ότι σκίαση των ακρών βελτιώνει την κατανόηση και την φυσικότητα της ύπαρξης των αντικειμένων στον εικονικό κόσμο αφού ο χρήστης αντιλαμβάνεται πιο εύκολα τα όρια τους.

Στις εικόνες που ακολουθούν περιγράφετε η εμφάνιση του εικονικού δωματίου όταν σε αυτό έχει και όταν δεν έχει εφαρμοστεί σκίαση κορυφών. Στις τρεις πρώτες εικόνες δεν έχει εφαρμοστεί σκίαση και στην τρίτη εικόνα (που εμφανίζει το εικονικό δωμάτιο) παρατηρούμε ότι το αποτέλεσμα υστερεί σε ρεαλισμό καθώς όλοι οι τοίχοι και η οροφή φαίνονται ενιαίοι.

Το αποτέλεσμα βελτιώνεται σημαντικά στις τρεις επόμενες εικόνες όπου έχει εφαρμοστεί σκίαση στις άκρες. Το τελικό αποτέλεσμα εμφανίζεται στην εικόνα «Shadowed Virtual Room» όπου οι γωνίες και οι τοίχοι διακρίνονται καθώς επίσης η αίσθηση του βάθους έχει βελτιωθεί σημαντικά.

Figure 66. Αρχική εικόνα πατώματος

Figure 67. Αρχική εικόνα τοίχου

Figure 68. Μη σκιασμένο εικονικό δωμάτιο

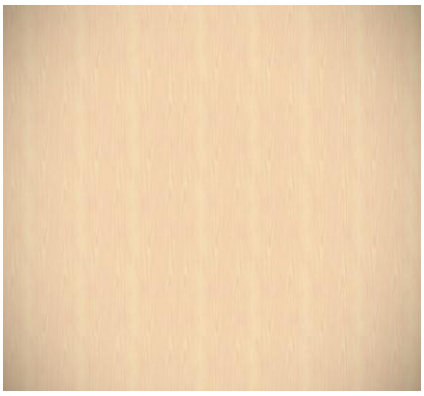


Figure 69. Σκιασμένες άκρες πατώματος

Figure 70. Σκιασμένες άκρες τοίχου

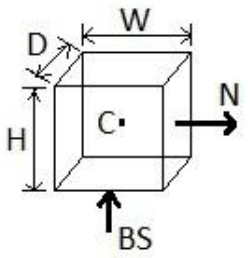
Figure 71. Σκιασμένο εικονικό δωμάτιο

17.4 Οντότητα Κύβος

Βασική σχεδιαστική οντότητα στην εφαρμογή μας αποτελεί το γραφικό αντικείμενο τύπου Cube. Το αντικείμενο αυτό συνοδεύουν ορισμένα χαρακτηριστικά και λειτουργίες που δίδονται ως επί το πλείστον κατά την διάρκεια δημιουργίας του. Αυτά είναι τα χαρακτηριστικά W, H, D, N. Η παραγωγή διανυσμάτων κρίνεται απαραίτητη εφόσον θελήσουμε το αντικείμενο μας να αλληλεπιδράσει επιτυχώς με οντότητες που βασίζουν την λειτουργίας στη χρήση διανυσμάτων όπως το φως. Το οκτάγωνο που παράγουμε διαθέτει έξη στον αριθμό επιφάνειες «όπως το BS» στις οποίες μπορεί να τοποθετηθεί ένα αντικείμενο τύπου texture που θα φέρει επάνω του την επιθυμητή πληροφορία και εμφάνιση.

Τέλος όλα τα αντικείμενα τύπου Cube διαθέτουν ένα κέντρο C αλλά παρόλα αυτά για να τοποθετηθούν στο τρισδιάστατο κόσμο σε σημείο διαφορετικό του κέντρου του

«δηλαδή το σημείο (0,0,0)» θα πρέπει να βασιστούν στην βοήθεια αντικειμένων τύπου TransformGroup και Transform3D. Έτσι το προσθέτουμε ως απόγονο σε αντικείμενο τύπου TransformGroup και δίδουμε την επιθυμητή θέση στο αντικείμενο τύπου Transform3D.



C = Center = Cube Entity Coordinates
W = Width = Cube Entity Width
H = Height = Cube Entity Height
D = Depth = Cube Entity Depth
N = Normal
BS = Cube Entity Bottom Side

Figure 72. Χαρακτηριστικά οντότητας κύβου

Θα πρέπει να επισημάνουμε ότι παρότι η Java3D επιτρέπει την επικάλυψη οντοτήτων η εφαρμογή μας προσπαθεί να το αποτρέψει αυτό κάνοντας ελέγχους επικάλυψης κατά την μεταφορά οντοτήτων από τον χρήστη και κατά την αυτόματη εισαγωγή οντοτήτων από την ίδια την εφαρμογή. Σε περίπτωση που υπάρχει επικάλυψη, το νέο αντικείμενο που εισάγεται στον χώρο τον οποίο καταλαμβάνει κάποιο άλλο, τοποθετείτε πιο βαθιά στον άξονα Z του τρισδιάστατου χώρου.

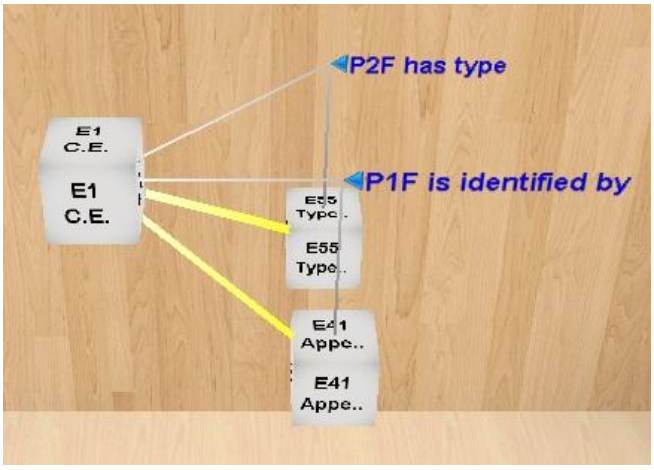


Figure 73. Αυτόματη τοποθέτηση κύβου

Η δημιουργία του texture αποτελεί μια διαδικασία που αποτελείται από τέσσερα βήματα τα οποία απεικονίζονται στην εικόνα που ακολουθεί. Πρώτον, βάση του επιθυμητού κειμένου που επιθυμούμε να αναγραφεί στο αντικείμενο θα πρέπει να υπολογίσουμε το μέγιστο επιτρεπτό μέγεθος γραμματοσειράς. Αυτό θα πρέπει να μας επιτρέπει να εκτυπώσουμε την πλήρη επιθυμητή πληροφορία καθώς επίσης να αποτελεί το βέλτιστο εμφανισιακά αποτέλεσμα για το χρήστη. Θα πρέπει να λάβουμε υπόψη κατά τη διάρκεια των υπολογισμών ότι :

1) Όλη η επιφάνεια μίας έδρας του κύβου αποτελεί χώρο προσβάσιμο για το κείμενο (δηλαδή τα γράμματα μπορούν να εμφανιστούν σε οποιοδήποτε σημείο).

2) Η πληροφορία θα μπορεί να διασπάτε σε παραπάνω από μια γραμμές.

3) Το αποτέλεσμα που πρέπει να είναι όσο το δυνατόν πιο κεντραρισμένο (δηλαδή να έχει ως σημείο αναφοράς στοίχισης το κέντρο της έδρας).

4) Το αποτέλεσμα θα πρέπει να είναι όσο το δυνατόν ομογενοποιημένο δίχως κενά ή επικαλύψεις.

Μετά το σημείο που έχουμε υπολογίσει τη μέγιστη γραμματοσειρά που θα μπορούσε να χρησιμοποιηθεί αποτυπώνουμε το κείμενο σε μια διαφανή επιφάνεια που αποτελεί το Text Layer. Έπειτα το επόμενο στάδιο αφορά τη φόρτωση του επιθυμητού background για το αντικείμενο τύπου κύβος που θα χρησιμοποιήσουμε. Αυτό αποτελεί μια εικόνα που φιλοξενείται στο πακέτο δεδομένων της εφαρμογής. Τρίτη φάση αποτελεί η κατασκευή του Texture Layer. Αυτό περιλαμβάνει την δημιουργία μιας οντότητας τύπου Texture. Ως περιεχόμενο της επιφάνειας που διαθέτει το αντικείμενο

texture θέτουμε την εικόνα που έχει κατασκευαστεί στο επίπεδο Image Layer. Τέλος βασική οντότητα αποτελεί το στιγμιότυπο της οντότητας Box που δημιουργεί ένα τρισδιάστατο γεωμετρικό εξάεδρο σχήμα συμβατό με την Java3D. Ως περιεχόμενο και των έξι επιφανειών που διαθέτει το αντικείμενο αυτό ορίζεται το texture που έχει κατασκευαστεί στο Texture Layer.

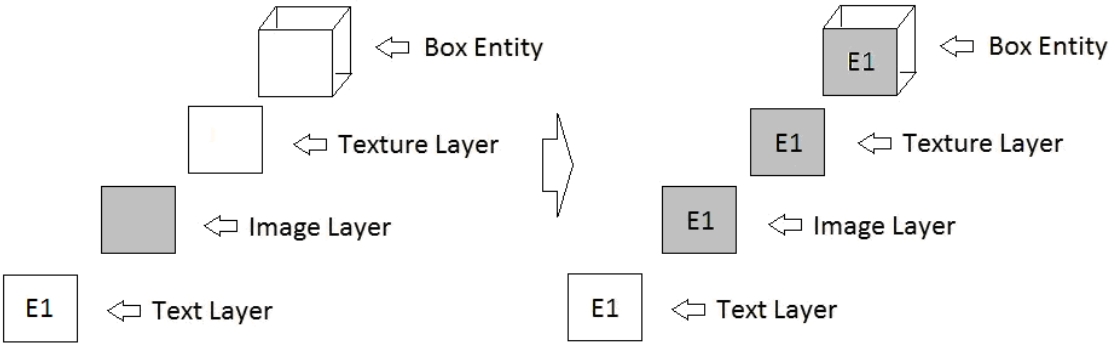


Figure 74. Σχεδίαση οντότητας Κύβος

17.5 Οντότητα Γραμμή

Αποτελείται από δυο βασικά συστατικά στοιχεία. Αυτά είναι οι κορυφές που υφίστανται ως σημεία στο τρισδιάστατο χώρο καθώς και τα ευθύγραμμα τμήματα που ενώνουν τις κορυφές. Το πλήθος των κορυφών είναι πάντα ζυγός αριθμός και διπλάσιος του πλήθους των ευθύγραμμων τμημάτων. Οποιοδήποτε γεωμετρικό σχήμα πιο πολύπλοκο αυτού της ευθείας θα πρέπει να κατασκευαστεί βάση συνδυασμού ευθύγραμμων τμημάτων.

Για παράδειγμα, το γεωμετρικό σχήμα που αναφέρεται στην παρακάτω εικόνα με την επονομασία Angle έχει κατασκευαστεί βάση δύο ευθύγραμμων τμημάτων και τεσσάρων κορυφών. Αυτά είναι το ευθύγραμμο τμήμα (SP=>MP) με κορυφές (SP,MP) και το ευθύγραμμο τμήμα (MP=>FP) με κορυφές (MP,FP). Τα ενδιάμεσα σημεία από τα οποία προελαύνει η εκάστοτε γραμμή υπολογίζονται βάση του σημείου εκκίνησης και τερματισμού. Για παράδειγμα, στον τύπο γραμμής “Angle” το ενδιάμεσο σημείο MP ορίζεται ως (X=FP.x , Y=SP.y , Z=FP.z) όπου (SP=σημείο εκκίνησης) και (FP=σημείο τερματισμού).

Στο γεωμετρικό σχήμα τύπου Circular, χάρη στην ευελιξία που προσφέρει ο τρισδιάστατος κόσμος, κάνουμε χρήση της διάστασης Z για να το μορφοποιήσουμε κατάλληλα. Έτσι το σχήμα γίνεται πιο ευέλικτο και δεν επιβαρύνουμε πολύ την εικόνα

που παρουσιάζεται στον χρήστη. Γι' αυτό, το σημείο MP2, MP3 η διάσταση Z ισούται με SP.z+space , όπου space είναι μια σταθερά που έχει προκύψει μετά από πειράματα.

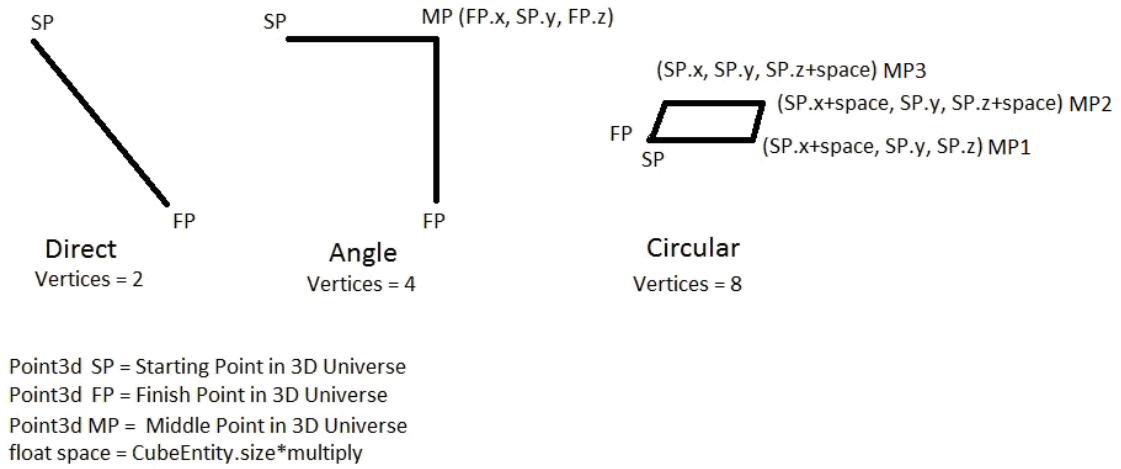


Figure 75. Σχεδίαση άκρων οντότητας γραμμής

Σε περίπτωση που τα σημεία SP και FP βρίσκονται σε παραπλήσιο ύψος και η σχέση ανάμεσα τους είναι τύπου property τότε αποφασίστηκε να διατηρούμε μια σχεδιαστική αρχή που μεταφέρει το σημείο MP στην μέση της απόστασής τους με τρόπο κατάλληλο ώστε να μην συμπέφτει με γραμμές σχέσεων τύπου Isa. Η σχεδίαση αυτή επιλέχτηκε ώστε να αποφεύγουμε όσο το δυνατόν περισσότερο το φαινόμενο επικάλυψης Property Label από Cube Entities. Με αυτόν τον τρόπο αποφεύγουμε την επεξεργαστική επιβάρυνση χάρη στο Collision Detection Look Ahead αφού αποφεύγουμε έμμεσα την επικάλυψη πριν καν αυτή προκύψει. Δυστυχώς με αυτόν τον τρόπο δεν μπορούμε να αποφύγουμε την επικάλυψη όταν τα αντικείμενα βρίσκονται αρκετά κοντά.

Figure 76. Ειδική σχεδίαση άκρων για σχέσεις σε παραπλήσιο ύψος

Πέραν της διαδρομής που ακολουθεί και των οντοτήτων που συνδέει, μια γραμμή διαθέτει και άλλα χαρακτηριστικά όπως το χρωματισμό του και το πάχος του. Το χαρακτηριστικό του χρώματος εξελίχθηκε και χάρη σε αυτό τα αντικείμενα τύπου γραμμής διαθέτουν διαφορετικά χρώματα στις άκρες και ομαλή μετάβαση από το ένα χρώμα στο άλλο κατά την μεταφορά από το σημείο SP στο FP.

Το σημείο SP χαρακτηρίζεται από λευκό χρώμα και το σημείο FP χαρακτηρίζεται από χρώμα που προκύπτει ανάλογα με την κατηγορία της και την λειτουργία της. Στα ενδιάμεσα σημεία γίνεται μεταφορά από το λευκό χρώμα στο χρώμα του σημείου τερματισμού.

Ο λόγος που υιοθετήθηκε και εξελίχθηκε αυτό το εφέ είναι το ότι με αυτόν τον τρόπο ο χρήστης λαμβάνει διαισθητικά πληροφορία για το αντικείμενο που αποτελεί το Domain Entity και για το αντικείμενο που αποτελεί το Range Entity στην σχέση που αναπαριστούμε, δίχως το σχήμα που παρουσιάζουμε στον χρήστη να επιβαρύνετε με επιπλέον οντότητες ή χαρακτηριστικά.

Όταν το σχήμα αποτελείται από παραπάνω από ένα ευθύγραμμα τμήματα, τότε θα πρέπει να αναλογιστούμε κατάλληλο διαμερισμό του παράγοντα χρώμα που θα εντοπίζεται σε κάθε κορυφή. Αυτό θα προκύψει εάν αφαιρεθεί από το χρώμα του σημείου εκκίνησης το χρώμα του σημείου τερματισμού και το αποτέλεσμα διαιρεθεί δια του πλήθους των ευθύγραμμων τμημάτων. Όπως δείχνουμε στην παρακάτω εικόνα, η μετάβαση του χρώματος γίνεται σταδιακά βάση μαθηματικών υπολογισμών στο μοντέλο RGB με value [0,1].

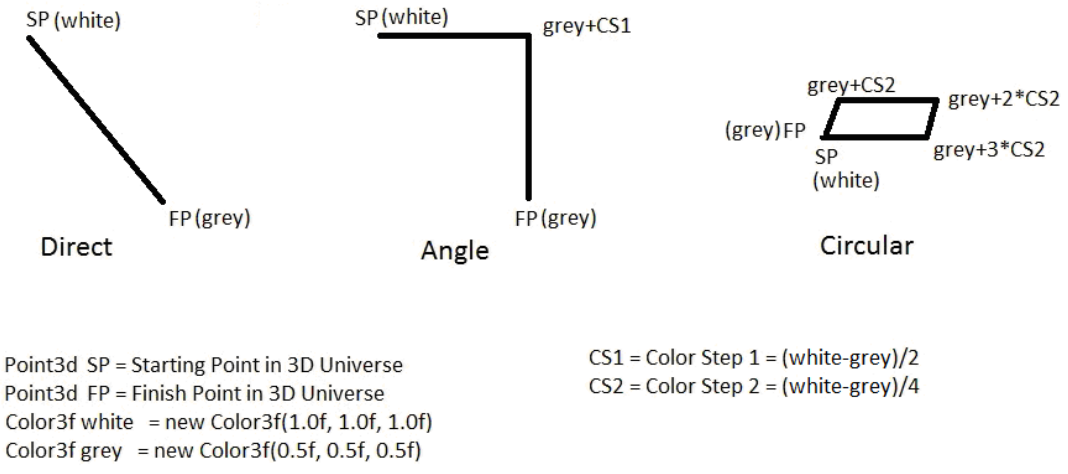


Figure 77. Σχεδίαση χρωμάτων

Ένα πρόβλημα που χρειάστηκε να λύσουμε είναι ότι ανάμεσα σε δύο οντότητες του εικονικού κόσμου ενδέχεται να υπάρξουν παραπάνω από μία σχέσεις που τις συνδέουν.

Στην περίπτωση αυτή αν η γραμμή είναι τύπου Angle τότε η πρώτη γραμμή εισάγεται φυσιολογικά στις συντεταγμένες που έχουμε αναφέρει πιο πάνω. Οι επιπλέον γραμμές όμως που αναπαριστούν τις επιπλέον σχέσεις κατέχουν το ίδιο σημείο εκκίνησης και τερματισμού αλλά διαφορετικό ενδιάμεσο σημείο MP. Στην περίπτωση Angle το ενδιάμεσο σημείο MP τοποθετείτε βαθύτερα στο τρισδιάστατο χώρο. Ενώ στην περίπτωση Circular οι επιπλέον σχέσεις τοποθετούνται ψηλότερα στο άξονα Y.

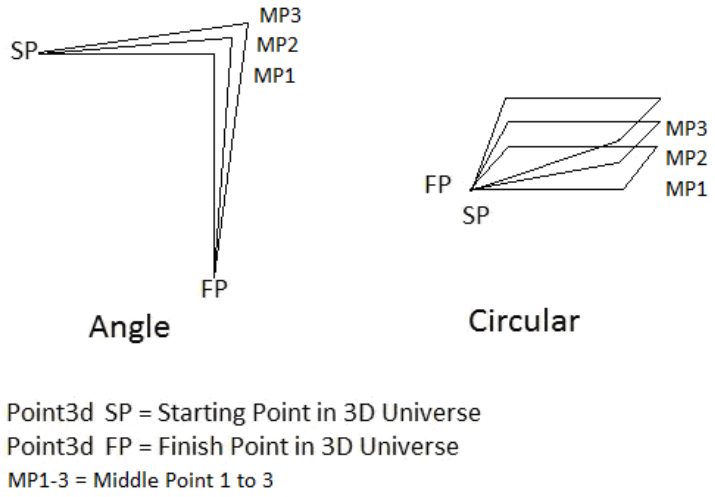


Figure 78. Πολλαπλές γραμμές

Για τον υπολογισμό του κατάλληλου βάθους γραμμής σε περίπτωση που υφίστανται πολλαπλές σχέσεις ανάμεσα σε δύο οντότητες θα πρέπει να ελέγξουμε εάν η θέση που πρόκειται να εισάγουμε το νέο αντικείμενο είναι διαθέσιμη ή όχι. Για να ικανοποιηθεί το κριτήριο αυτό θα πρέπει να ελέγξουμε τέσσερις παράγοντες:

- το σημείο έναρξης (start)
- το σημείο τερματισμού (finish)
- το ενδιάμεσο σημείο (angle_point)
- το προϋπολογισμένο ενδιάμεσο σημείο (precalculated_angle_point)

Έτσι, εάν υπάρχει ήδη γραμμή που διαθέτει τα ίδια στοιχεία στα προαναφερθέντα χαρακτηριστικά, τότε θα πρέπει να υπολογίσουμε καινούριες συντεταγμένες για την γραμμή αυτή.

Πιο αναλυτικά, ελέγχουμε εάν τα χαρακτηριστικά της γραμμής που θα θέλαμε να εισάγουμε συμπίπτουν με κάποιας άλλης που έχουμε εισάγει νωρίτερα. Τα χαρακτηριστικά αυτά είναι το σημείο έναρξης της ευθείας, το σημείο τερματισμού, το προϋπολογισμένο ενδιάμεσο σημείο που ισούται με το σημείο όπου θα έπρεπε να βρίσκετε το angle_point και τέλος το ενδιάμεσο σημείο angle_point που αλλάζει σε περίπτωση που υπάρχει κάποια άλλη γραμμή με το ίδιο angle_point. Σε περίπτωση που αλλάξουμε το σημείο angle_point, τότε θα πρέπει να ξανακάνουμε τους πιο πάνω ελέγχους.

```
Vector3d  angle_point = new Vector3d( precalculated_angle_point );

while( exist_line_with_same_coordinates(start, finish, angle_point) )

{

depth=depth+step;

angle_point = new Vector3d( finish.x , start.y , finish.z-depth );

}
```

Code 17. Υπολογισμός ενδιάμεσου σημείου πολλαπλών γραμμών

17.6 3D Ανίχνευση Επικάλυψης

Για την εξυπηρέτηση των αναγκών ανίχνευσης επικάλυψης οντοτήτων στο τρισδιάστατο χώρο επιλέχτηκε η μέθοδος χρήσης των λεγόμενων περικλείουσων σφαιρών. Βασικό λόγω επιλογής αυτής της τεχνικής αποτελεί το γεγονός ότι είναι σε θέση να λειτουργήσει αποτελεσματικά διαθέτοντας μικρό περιθώριο λάθους. Επίσης σημαντικό πλεονέκτημα αποτελεί το γεγονός ότι είναι σε θέση να ανταπεξέλθει σε πραγματικό χρόνο με ελάχιστη επεξεργαστική επιβάρυνση. Η μέθοδος αυτή δεν ελέγχει την επικάλυψη των ιδίων των τρισδιάστατων οντοτήτων αλλά των ιδεατών σφαιρών που είναι σε θέση να επικαλύψουν τις οντότητες αυτές.

Οι σφαίρες αυτές έχουν ως κέντρο τους το ίδιο σημείο που έχει ως κέντρο και το τρισδιάστατο αντικείμενο που θέλουμε να παρατηρήσουμε ως προς την επικάλυψη. Το σημείο αυτό εφόσον αποτελεί σημείο του τρισδιάστατου χώρου περιέχει τρεις μεταβλητές (x, y, z) τα οποία το χαρακτηρίζουν. Επίσης, η ακτίνα τους συνήθως ισούται με την απόσταση που υπάρχει ανάμεσα στο κέντρο βάρους της εικονικής οντότητας και στο σημείο της που απέχει περισσότερο από αυτό. Πρέπει να τονίσουμε ότι οι σφαίρες ελέγχου επικάλυψης αποτελούν αυτόνομα αντικείμενα και δεν ταυτίζονται με τις οντότητες που περιέχουν. Έτσι μπορούν να υπάρξουν και να λειτουργήσουν ανεξάρτητα από τα αντικείμενα που μπορεί ή όχι να περικλείονται μέσα τους.

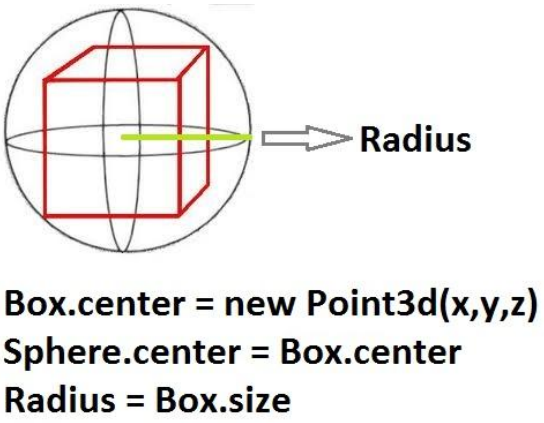


Figure 79. Αρχικοποίηση σφαίρας επικάλυψης

Η απόσταση μεταξύ δύο σημείων αποτελεί την διαδρομή που τα συνδέει. Η μικρότερη διαδρομή είναι η ευθεία ανάμεσα τους και στον τρισδιάστατο χώρο αυτή η ευθεία που θα περιέχεται ανάμεσα στα σημεία (X_1, Y_1, Z_1) και (X_2, Y_2, Z_2) δίνεται από τον τύπο:

$$d = \text{sqrt}[(x_1-x_2)^2 + (y_1-y_2)^2 + (z_1-z_2)^2]$$

Όπως βλέπουμε και στην παρακάτω εικόνα αρχικά διαθέτουμε δύο οντότητες, οι οποίες είναι οι Σφαίρα Ανίχνευσης Επικάλυψης και η οντότητα Κύβος, δίχως αυτά να σχετίζονται με κάποιο τρόπο. Έπειτα αυτά εισάγονται στον εικονικό κόσμο και από την στιγμή αυτή και μετά θεωρούνται “live” οντότητες, άρα είναι οπτικά παρουσιάσιμες στον χρήστη. Κατά την διάρκεια της εισαγωγής προσέχουμε ώστε το κέντρο της σφαίρας να συμπίπτει με αυτή του κύβου, διαφορετικά η εφαρμογή δεν θα είναι σε

θέση να εντοπίσει επικάλυψη οντοτήτων. Από την στιγμή εισαγωγής της ιδεατής σφαίρας και του κύβου στον εικονικό κόσμο η εφαρμογή είναι σε θέση να εντοπίζει επικαλύψεις οντοτήτων. Επικάλυψη προκύπτει όταν μέσα στον χώρο που περικλείει η σφαίρα βρεθεί η σφαίρα επικάλυψης κάποιου άλλου εικονικού αντικειμένου.

Προγραμματιστικά αποτελεί ένα Thread το οποίο ενεργοποιείται κατά την εισαγωγή και εξαγωγή μίας εικονικής οντότητας μέσα στην σφαίρα ελέγχου επικάλυψης κάποιας άλλης εικονικής οντότητας. Για την αποφυγή επεξεργαστικής επιβάρυνσης του συστήματος επιλέχτηκε η ενεργοποίηση του Thread μόνο στα Events CollisionEntry και CollisionExit .

Ο έλεγχος αυτός , όπως είπαμε και πιο πριν, προκύπτει βάση της απόστασης των κέντρων των σφαιρών ελέγχου επικάλυψης και της ακτίνας τους. Αξίζει να τονιστεί ότι εφόσον η εφαρμογή μας είναι σε θέση να αλλάξει σε πραγματικό χρόνο το σημείο στο οποίο βρίσκονται οι εικονικές οντότητες, έτσι και ο έλεγχος ύπαρξης επικαλύψεις σχεδιάστηκε ούτως ώστε να μπορεί να ελέγχει σε πραγματικό χρόνο την ιδιότητα αυτή.

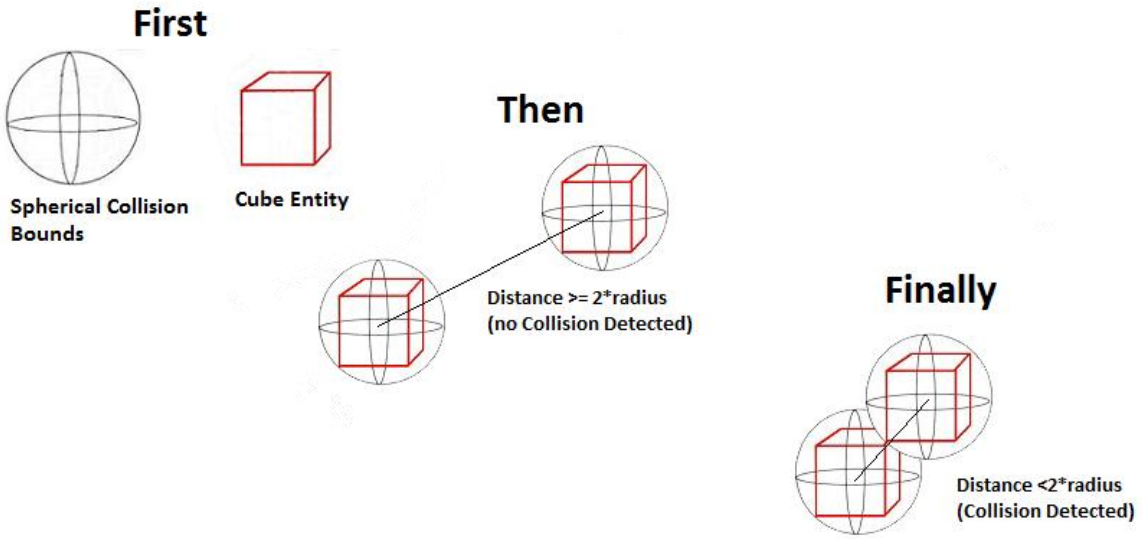


Figure 80. Ανίχνευση επικάλυψης βάση υπολογισμού απόστασης

Επιθυμητό αποτέλεσμα αποτελεί η εξολοκλήρου αποφυγή επικάλυψης των οντοτήτων. Έτσι κάνοντας χρήση του σωστού μεγέθους σφαίρας επικάλυψης μπορούμε να προλάβουμε και να αποφύγουμε το φαινόμενο αυτό σε ικανοποιητικό βαθμό. Για παράδειγμα, αν η σφαίρα είναι μικρότερη από το μέγεθος που θα έπρεπε να είναι τότε θα υπάρχει επικάλυψη μεταξύ οντοτήτων η όποια δεν θα ανιχνεύεται. Στην αντίθετη

περίπτωση, αν η σφαίρα είναι μεγαλύτερη από το πρέπον μέγεθος τότε η εφαρμογή θα αναφέρει (λανθασμένα) ύπαρξη επικάλυψης δίχως αυτή να υφίσταται. Γι' αυτό το μέγεθος που προτιμήθηκε στην εφαρμογή μας ισούται με την απόσταση του κέντρου του κύβου από μία εκ των επιφανειών του.

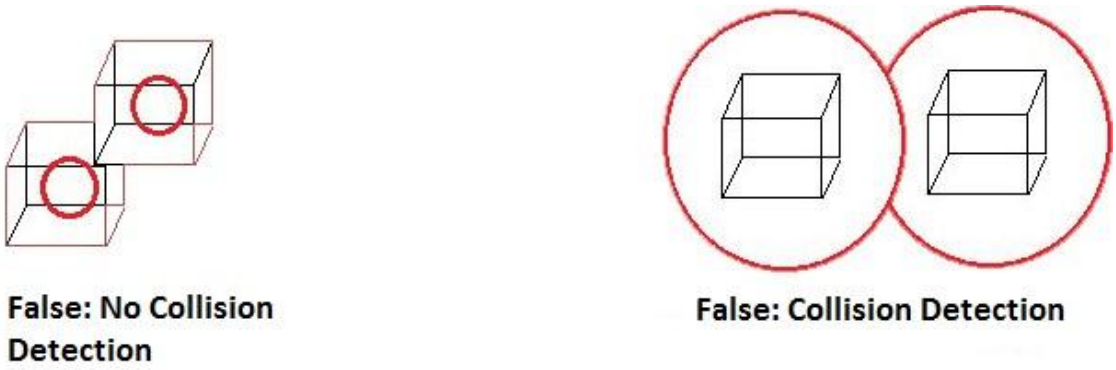


Figure 81. Λανθασμένη ανίχνευση επικάλυψης

17.7 Εφέ Σκίασης

Τα εφέ σκίασης θεωρείται ότι βελτιώνουν την κατανόηση και τον ρεαλισμό αναπαράστασης οντοτήτων σε υπολογιστικά συστήματα απεικόνισης εικόνας. Το πλέον σημαντικό πλεονέκτημα που παρέχουν πολύτιμη διαισθητική πληροφορία για την θέση που βρίσκονται τα αντικείμενα στο χώρο αλλά και την σχετική μεταξύ τους θέση, αποφεύγοντας τυχόν αμφιλεγόμενες καταστάσεις και βελτιώνοντας έτσι την κατανόηση του θεατή. Στην παρακάτω εικόνα βλέπουμε ένα χαρακτηριστικό παράδειγμα οφθαλμαπάτης όπου οι σκιές παίζουν βασικό ρόλο. Παρότι τα δύο αντικείμενα (κύβος, σφαίρα) βρίσκονται στην ίδια ακριβός θέση και τις τρεις φορές, ο θεατής έχει την αίσθηση ότι αυτό δεν ισχύει. Στην εικόνα «B» σε σχέση με την πρώτη ο θεατής έχει την αίσθηση (χάρη στην σκίαση) ότι η απόσταση μεταξύ των αντικειμένων έχει αυξηθεί. Ενώ στην εικόνα «C» δίνεται η αίσθηση στον χρήστη ότι η σφαίρα βρίσκεται μπροστά από τον κύβο.

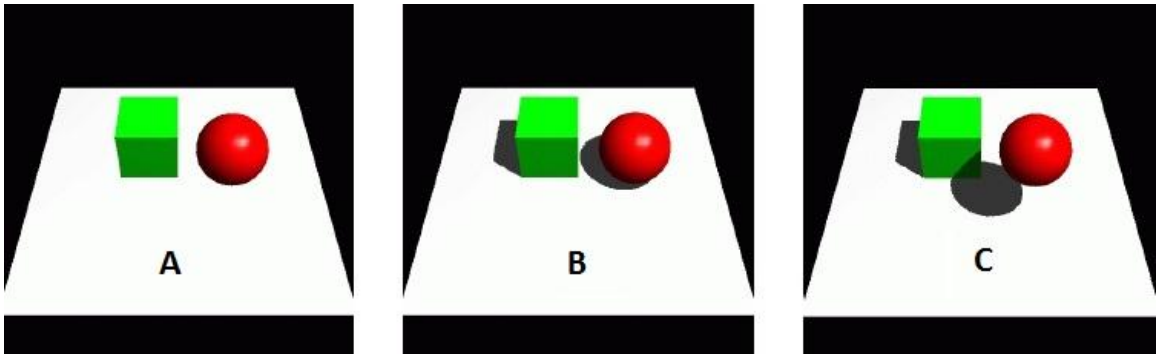


Figure 82. Κίνητρο σκίασης

Η υλοποίηση των εφέ σκίασης στην εφαρμογή μας βασίστηκαν στην ύπαρξη τεσσάρων παραγόντων. Αυτοί οι παράγοντες ήταν οι QuadArray, Image, Floor και Transparency και τους περιγράφουμε παρακάτω.

- QuadArray: Αποτελεί ένα εικονικό τετράεδρο που τοποθετείτε στον τρισδιάστατο χώρο και στο επίπεδο που σχηματίζει μπορεί να εφαρμοστεί μια επιθυμητή εικόνα.

- Image: Η επιθυμητή εικόνα που θα θέλαμε να παρουσιαστεί στον χρήστη και που προσομοιώνει την ύπαρξη σκιάς.

- Floor: Το εικονικό δάπεδο του τρισδιάστατου κόσμου το οποίο εισάγουμε για να δώσουμε μια αίσθηση προσανατολισμού στον χρήστη και πάνω στο οποίο τοποθετείται το texture της σκίασης. Όπως η σκιά, έτσι και το πάτωμα αποτελεί ένα texture το οποίο φέρει άλλη εικόνα και έχει μεγαλύτερο μέγεθος. Για να επιτευχθεί βέλτιστη απόδοση ρεαλισμού και βέλτιστη ποιότητα αποικώνισης το texture που φέρει την εικόνα της σκίασης θα πρέπει να τοποθετηθεί στο ίδιο επίπεδο με το πάτωμα όσον αφορά τις διαστάσεις X, Z και ελάχιστα πιο ψηλά στην διάσταση Y που συμβολίζει και το ύψος. Η υψομετρική αυτή διαφορά ισούται με 0.0001 μέτρα και υφίσταται λόγω του ότι, αν και τα δύο textures είχαν το ίδιο ύψος, τότε η εφαρμογή δεν θα γνώριζε πιο από τα δύο θα έπρεπε να παρουσιαστεί στον χρήστη καθώς δυνατότητες «color blending» δεν προσφέρονται από την Java3D.

- Transparency: Το texture που φέρει την εικόνα της σκιάς θα πρέπει να είναι σε θέση να προσομοιώσει την υψομετρική διαφορά που υπάρχει

ανάμεσα στα αντικείμενα. Γι' αυτό αντικείμενα που βρίσκονται ψηλότερα θα πρέπει να διαθέτουν αμυδρή έως ανύπαρκτη σκιά και αντικείμενα που βρίσκονται χαμηλά θα πρέπει να διαθέτουν έντονη σκιά. Έτσι το αποτέλεσμα σκίασης θα είναι περισσότερο ρεαλιστικό. Το Transparency παίρνει τιμές ανάμεσα στο [0,1] με την μέγιστη τιμή 1 να θεωρείται ως απόλυτα διαφανείς. Η τιμή αυτή προκύπτει βάση της διαίρεσης του ύψους του αντικειμένου δια του ύψους του δωματίου «EntityHeight/RoomSize». Για να γίνουν σωστά οι υπολογισμοί θα πρέπει να λάβουμε υπόψη μας ότι το ύψος λαμβάνει θετικές και αρνητικές τιμές καθώς το κέντρο του συστήματος συντεταγμένων βρίσκεται στο κέντρο του εικονικού δωματίου.

```
public static void setTransparency( Appearance polygonAppearance , float trns )
{
    TransparencyAttributes ta = new TransparencyAttributes();
    ta.setTransparencyMode( ta.BLENDED );
    ta.setTransparency( trns );
    polygonAppearance.setTransparencyAttributes( ta );
}
```

Figure 83. Χαρακτηριστικό διαφάνειας

Έτσι στο αντικείμενο QuadArray προστίθεται μία γκρίζα φωτογραφία που εξομοιώνει την σκιά. Έπειτα, γνωρίζοντας το ύψος του αντικειμένου του οποίου την σκιά κατασκευάζουμε, υπολογίζουμε το μέγεθος του Transparency που θα χρησιμοποιηθεί. Εφαρμόζοντας την διαπερατότητα αυτή στο texture προκύπτει η τελική εικόνα προσομοίωσης της σκίασης. Έπειτα, αυτή προστίθεται στον εικονικό κόσμο σε θέση η οποία διαθέτει τις ίδιες τιμές με το αντικείμενο στις διαστάσεις X, Z αλλά ελαφρός ψηλότερα από το δάπεδο κατά 1/10 του χιλιοστού ή 0.0001 μέτρα.

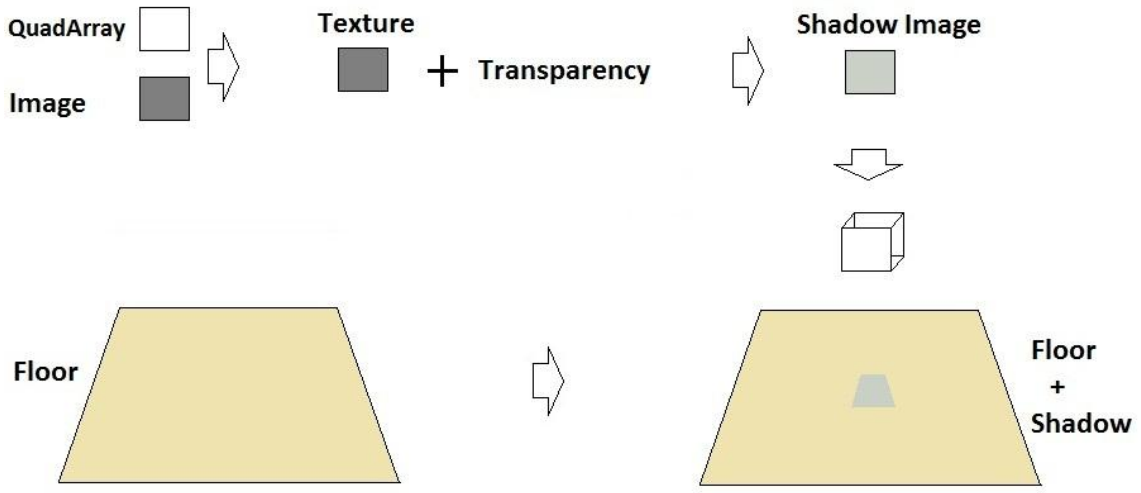


Figure 84. Δημιουργία σκιάς κύβου

Παρόμοια διαδικασία ακολουθείται και κατά την διάρκεια δημιουργίας σκίασης για οντότητες τύπου γραμμής. Η διαφορά σε αυτήν την περίπτωση προκύπτει βάση του γεγονότος ότι η αρχή και το τέλος μιας γραμμής μπορεί να παρουσιάζουν υψομετρική διαφορά. Λόγω αυτού το σημείο της ευθείας που βρίσκεται ψηλότερα θα πρέπει να διαθέτει λιγότερο έντονη σκίαση σε σχέση με το σημείο της ευθείας που βρίσκεται χαμηλότερα. Επίσης αυτή η μεταβολή θα πρέπει να συμβαίνει σταδιακά ώστε να μην διαταραχθεί η ρεαλιστικότητα του σχήματος. Έτσι η επιλογή της εικόνας που προσομοιώνει την σκιά της γραμμής δεν ήταν τυχαία αλλά επιλέχτηκε επειδή παρουσιάζει σταδιακή μετάβαση από το χρώμα γκρι σε διαφανές. Αυτό μπορεί να υποστηριχθεί χάρη σε αρχεία εικόνας τύπου PNG. Επίσης για να γίνει σωστή αναγνώριση της εικόνας από την Java3D θα πρέπει να εκτελέσουμε ορισμένα βήματα σωστής αρχικοποίησης που φαίνονται στην παρακάτω εικόνα.

```
TextureLoader loader = new TextureLoader(img,"RGBA",TextureLoader.GENERATE_MIPMAP,observer);

Texture texImage = loader.getTexture();
```

Figure 85. Φόρτιση Texture σκιάς

Ο αλγόριθμος υπολογισμού σκίασης για οντότητες τύπου γραμμής περιγράφετε στο παρακάτω τμήμα ψευδοκώδικα.

```
//creates a shadow effect for a line which is between quordinates start and finish
BranchGroup create_line_shadow( Vector3d start , Vector3d finish )
{
   BranchGroup BG = new BranchGroup();
   if( start.x <= finish.x )   //if start point is after finish point
   swap = (start,finish);

   QuadArray shadow = new QuadArray (4,QuadArray.COORDINATES | GeometryArray.TEXTURE_COORDINATE_2);
   shadow.setCoordinate (0, new Point3d (start.x , -limit+0.00001f , start.z-width ));
   shadow.setCoordinate (1, new Point3d (finish.x , -limit+0.00001f , finish.z-width ));
   shadow.setCoordinate (2, new Point3d (finish.x , -limit+0.00001f , finish.z ));
   shadow.setCoordinate (3, new Point3d (start.x , -limit+0.00001f , start.z ));

   if( start.y >= finish.y )
   {
      shadow.setTextureCoordinate (1, new Point2f(0.0f,1.0f));
      shadow.setTextureCoordinate (2, new Point2f(0.0f,0.0f));
      shadow.setTextureCoordinate (3, new Point2f(1.0f,0.0f));
      shadow.setTextureCoordinate (0, new Point2f(1.0f,1.0f));
   }
   else
   {
      shadow.setTextureCoordinate (3, new Point2f(0.0f,1.0f));
      shadow.setTextureCoordinate (0, new Point2f(0.0f,0.0f));
      shadow.setTextureCoordinate (1, new Point2f(1.0f,0.0f));
      shadow.setTextureCoordinate (2, new Point2f(1.0f,1.0f));
   }

   BufferedImage img = null;
   try {
      img = ImageIO.read(new File("./images/wall/line_shadow.png"));
   } catch (IOException e) {
```

```
    }
    TextureLoader loader = new TextureLoader(img,"RGBA",TextureLoader.GENERATE_MIPMAP,observer);

    Texture texImage = loader.getTexture();
    Appearance app = new Appearance();

    app.setTexture (texImage);
    setTransparency(app,  transparency);
    BG.setUserData("Line Shadow");
    BG.addChild (new Shape3D (shadow,app));
    return BG;
}
```

Code 18. Ψευδοκώδικας δημιουργίας σκιάς γραμμής

Έχοντας λάβει αυτά υπόψη τοποθετούμε την πλευρά της εικόνας σκίασης που περιέχει έντονο γκρι χρώμα στο σημείο όπου η γραμμή απέχει λιγότερο από το δάπεδο. Αντίστοιχα τοποθετούμε την πλευρά της εικόνα που τείνει να είναι διαφανής στο σημείο που η γραμμή απέχει περισσότερο από το δάπεδο.

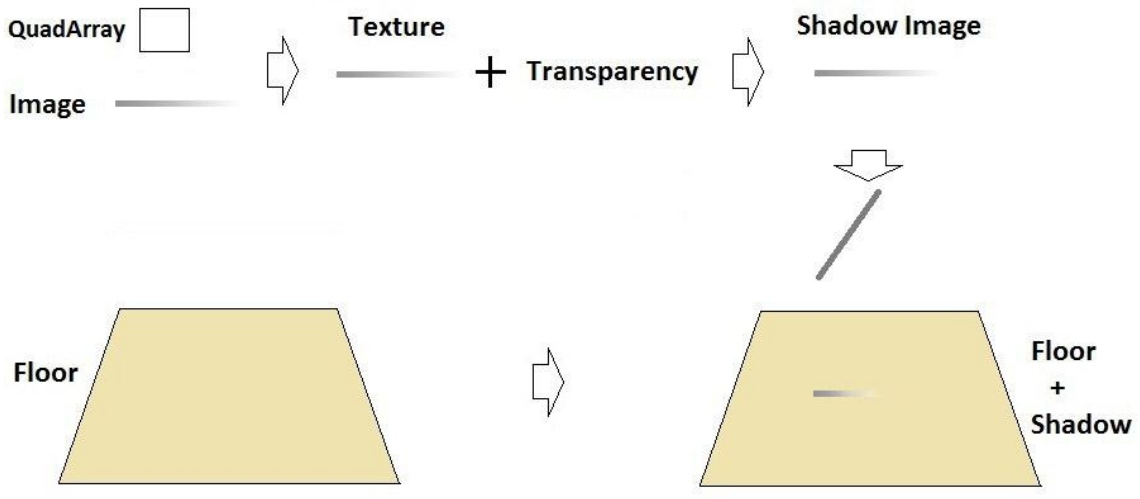


Figure 86. Δημιουργία σκιάς γραμμής

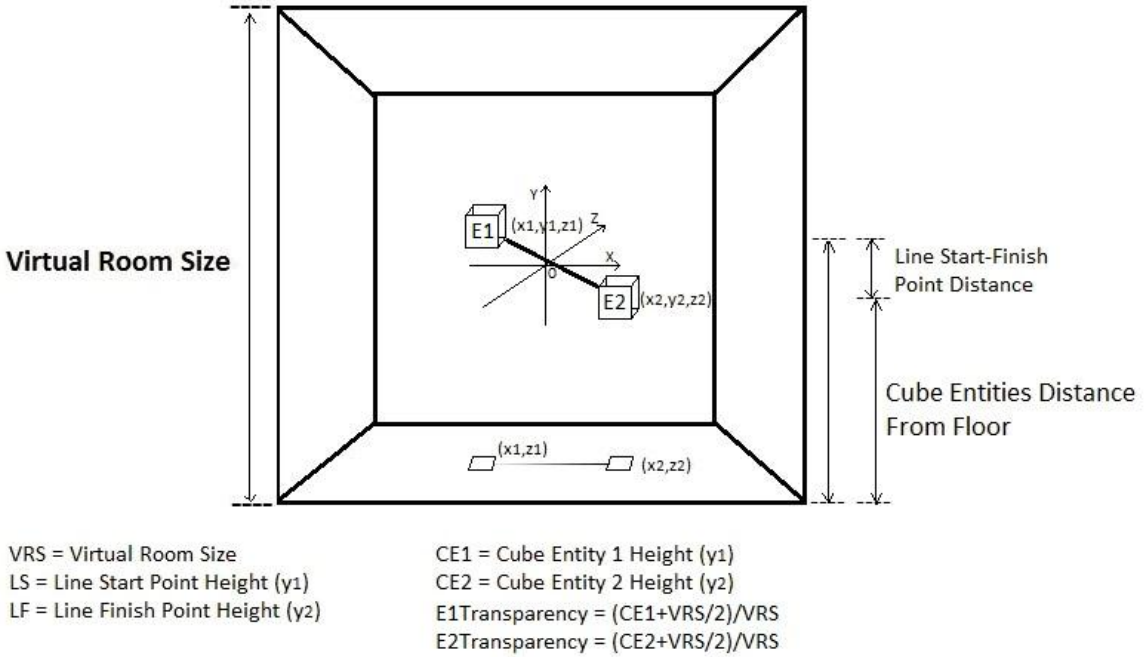


Figure 87. Υπολογισμός θέσης σκιάς

Ένα χαρακτηριστικό παράδειγμα εφαρμογής των πιο πάνω παρουσιάζουμε στην εικόνα που ακολουθεί. Η οντότητα E1 βρίσκεται ψηλότερα από την οντότητα E53 και γι' αυτό η σκιά της οντότητας E1 είναι λιγότερο έντονη σε σχέση με αυτή της E53. Εξαιτίας της ιεραρχίας οντοτήτων του μοντέλου Cidoc CRM, ανάμεσα στις οντότητες αυτές υπάρχει σχέση isa καθώς η οντότητα E53 αποτελεί παιδί της E1. Έτσι εισάγεται στο τρισδιάστατο χώρο μία γραμμή που συμβολίζει την σχέση αυτή και η οποία έχει ως αφετηρία την οντότητα E1 και ως σημείο προορισμού την οντότητα E53.

Εφόσον η αφετηρία της γραμμής που συμβολίζει την σχέση ανάμεσα σε δύο οντότητες βρίσκεται ψηλότερα από το σημείο προορισμού, θα πρέπει αυτό να αντικατοπτρίζεται και στην σκιά της γραμμής. Γι' αυτό το τμήμα της γραμμής που βρίσκεται χαμηλότερα έχει εντονότερη σκιά και όσο πιο ψηλά ανεβαίνουμε τόσο πιο αδρά θα φαίνεται το σχήμα της σκιάς.

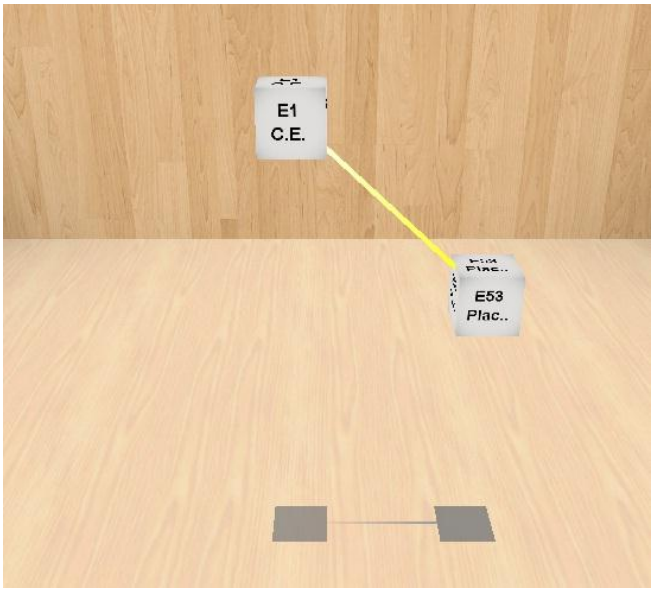


Figure 88. Παράδειγμα δημιουργίας σκιάς

17.8 Επιλογή Αντικειμένου

Για να μπορέσουμε να επιλέξουμε ένα από τα τρισδιάστατα αντικείμενα αλλά και για να αντλήσουμε πληροφορία από αυτό ή να μπορέσουμε να το χειριστούμε θα πρέπει να σεβαστούμε τις αρχές της Java3D.

Η ιεραρχία που λαμβάνει μέρος κατά την επιλογή ενός αντικειμένου και τον χειρισμό αυτού του αντικειμένου από τον χρήστη περιγράφεται στην παρακάτω εικόνα.

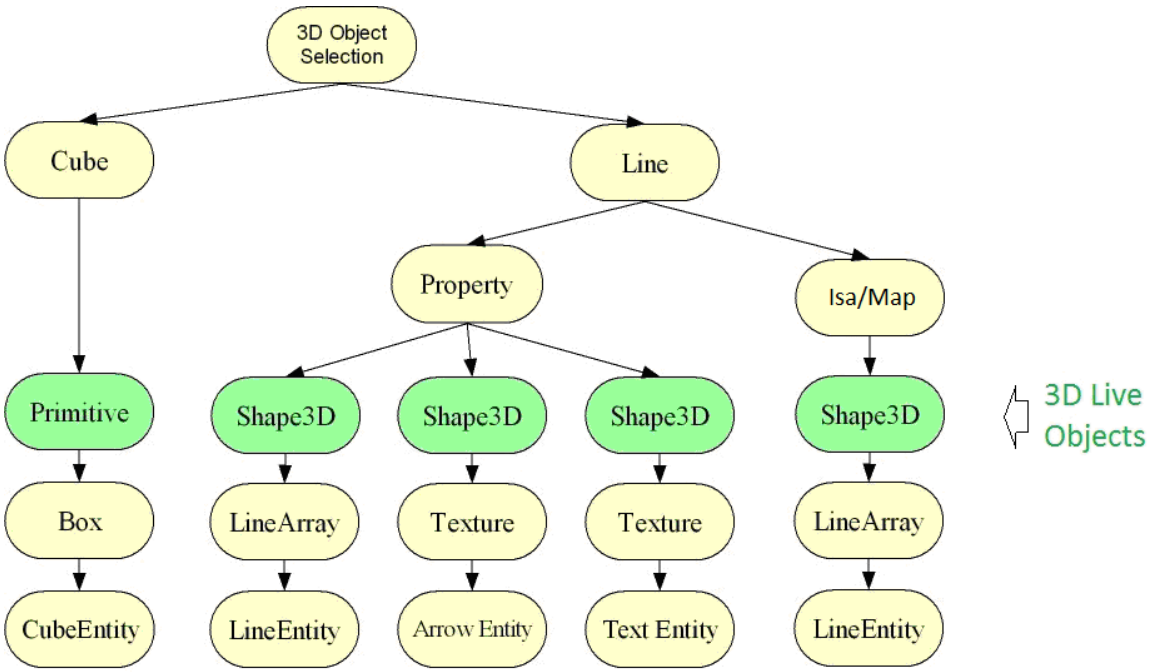


Figure 89. Ιεραρχία 3D επιλογής αντικειμένου

Τα τρισδιάστατα αντικείμενα που μπορεί να χειριστεί ο χρήστης ανήκουν σε δυο βασικές κατηγορίες, οι οποίες είναι οι οντότητες τύπου κύβος και γραμμές. Οι οντότητες τύπου γραμμής χωρίζονται σε τύπου Property και Isa/Map. Ο διαχωρισμός αυτός γίνεται δεδομένου του ότι οι γραμμές τύπου Property συνοδεύονται από γραφικές οντότητες τύπου βελάκι και κείμενο.

Ένα σημαντικό πλεονέκτημα των αντικειμένων τύπου Property αποτελεί το χαρακτηριστικό τους ότι παρόλο που αποτελούνται από τρεις εντελώς διαφορετικές και ανεξάρτητές οντότητες τύπου Shape3D, συμπεριφέρονται και αντιδρούν ως μία ενιαία οντότητα. Για παράδειγμα, εάν ο χρήστης επιθυμεί να διαγράψει ένα Property που συνδέει δύο οντότητες θα μπορούσε να το κάνει αυτό επιλέγοντας οποιαδήποτε από τις τρεις οντότητες τύπου Line, Arrow, Text και στην συνέχεια μέσω δεξί κλικ και Delete θα διαγραφεί ολόκληρη η οντότητα Property διαγράφοντας αυτόματα και τις υπόλοιπες δύο οντότητες.

Με πράσινο χρώμα περιγράφονται οι οντότητες που αποτελούν Live αντικείμενα του 3D Canva. Αυτά τα αντικείμενα είναι προσβάσιμα από τον χρήστη, επιλέγονται μέσω Mouse Events και χωρίζονται σε δύο κατηγορίες οι οποίες είναι η Primitive και Shape3D.

```
pickCanvas.setShapeLocation(e);
PickResult result = pickCanvas.pickClosest();

if (result != null)
{
      Primitive p = (Primitive)result.getNode(PickResult.PRIMITIVE);
      Shape3D s = (Shape3D)result.getNode(PickResult.SHAPE3D);
}
```

Code 19. Παράδειγμα κώδικα 3D επιλογής αντικειμένου

Τα αντικείμενα τύπου Primitive μπορούν να φέρουν πληροφορία για βασικά γεωμετρικά σχήματα όπως είναι οι τρισδιάστατοι κύβοι. Τα αντικείμενα τύπου Shape3D έχουν μια πολύ μεγαλύτερη και ευέλικτη γκάμα σχημάτων και βασίζονται πάνω στο Vertices Array που παράγει η γεωμετρία των σχημάτων καθώς και στην πληροφορία που δίνεται σε αυτά μέσω αντικειμένων τύπου Appearance. Συγκεκριμένα, χειρίζονται αντικείμενα τύπου LineArray και Texture όπου τα πρώτα είναι υπεύθυνα για την δημιουργία ευθύγραμμων τμημάτων και τα δεύτερα παράγουν τρισδιάστατες επιφάνειες πάνω στις οποίες μπορεί να φιλοξενηθεί οπτική πληροφορία. Η οπτική αυτή πληροφορία θα μπορούσε να είναι αντικείμενο τύπου Image ή αντικείμενο τύπου Text2D.

17.9 Εικονικό αόρατο τοίχος

Για να εκτελεστεί επιτυχώς η διαδικασία εισαγωγής οντοτήτων στον εικονικό κόσμο δόθηκε στον χρήστη η δυνατότητα επιλογής του σημείου χάρη σε εντολή τύπου αριστερού κλικ του ποντικιού στο επιθυμητό σημείο της οθόνης όπου αντιστοιχεί το εικονικό σημείο.

Ο προβληματισμός που προκύπτει από την μέθοδο αυτή είναι ότι σε κάθε σημείο της οθόνης «θεωρητικά» αντιστοιχούν Ν στον αριθμό σημεία του εικονικού κόσμου. Άρα καλούμαστε να υιοθετήσουμε κάποια σχεδιαστική αρχή ώστε ο χρήστης να είναι σε θέση να επιλέξει μόνο ένα από αυτά τα σημεία.

Η λύση που επικράτησε αναφέρεται στην δημιουργία μιας εικονικής διάφανης επιφάνειας η οποία θα είναι σε θέση να αντισταθεί στις μεταβολές κίνησης, περιστροφής και αυξομείωσης απόστασης από το χρήστη του ολικού εικονικού σύμπαντος και η οποία επιφάνεια θα βρίσκεται πάντα σε σταθερή απόσταση, μπροστά από την οπτική γωνία που αντιλαμβάνεται ο χρήστης.

Όπως βλέπουμε και στην εικόνα που ακολουθεί η επιφάνεια αυτή σχηματίζει κάθετη γωνία με την ευθεία που διέρχεται ανάμεσα στο μάτι του χρήστη και το σημείο πάνω στο οποίο παρευρίσκεται ο κέρσορας με αποτέλεσμα ο πολλαπλασιασμός των συντελεστών διεύθυνσης επιφάνειας και ευθείας να ισούται με μείον ένα.

Χάρη σε αυτά η επιλογή του επιθυμητού σημείου μεταφράζεται ως εύρεση του σημείου όπου τέμνεται η ευθεία με το εικονικό επίπεδο.

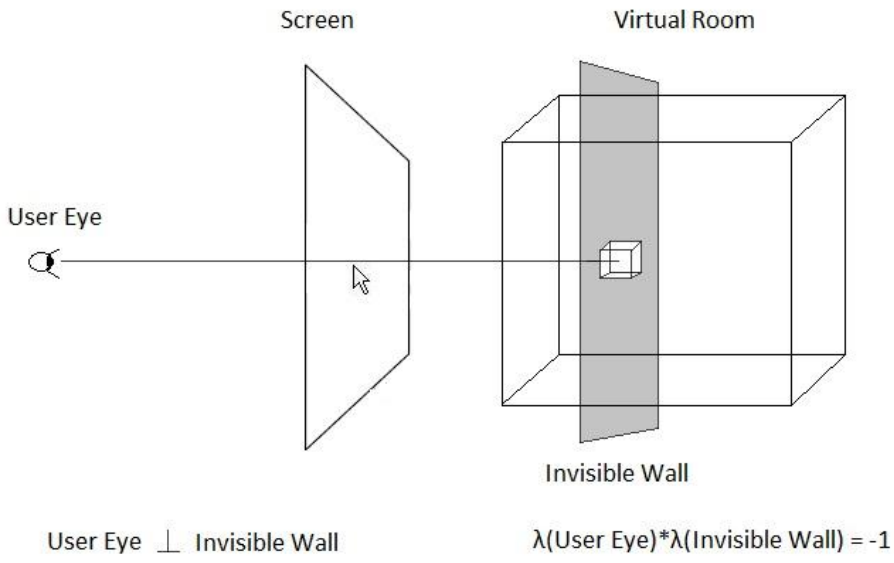


Figure 90. Σχεδίαση εικονικού αόρατου τοίχους

17.10 Αρχείο Διαμόρφωσης

Για την επιτυχή αρχικοποίηση του συστήματος και για την επιτυχή αποθήκευση ρυθμίσεων απαιτήθηκε η χρήση ειδικού αρχείου τύπου XML όπου αποθηκεύονται όλες οι απαραίτητες ρυθμίσεις για την σωστή έναρξη λειτουργίας του προγράμματος αλλά και την αποθήκευση των προσωπικών ρυθμίσεων συστήματος του χρήστη.

Η δομή του αρχείου έχει την μορφή <category> value </category> και ένα παράδειγμα περιγράφουμε στον παρακάτω κώδικα όπου δίνετε το όνομα του αρχείου που περιέχει όλη την απαραίτητη πληροφορία για τις οντότητες και τις σχέσεις του μοντέλου Cidoc CRM. Επίσης φέρει πληροφορία για την εικόνα που θα αναπαριστά τους τοίχους, το πάτωμα και το μέγεθος του εικονικού δωματίου.

Το αρχείο αυτό διαβάζεται κάθε φορά κατά την έναρξη του προγράμματος και επανεγγράφεται κάθε φορά κατά τον τερματισμό του.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Mapping_Tool>
3         <Cidoc_RDF_File>cidoc_crm_v5.0.2.rdfs</Cidoc_RDF_File>
4         <Wall>wood.jpg</Wall>
5         <Floor>wooden.jpg</Floor>
6         <Room_Size>3.0</Room_Size>
7 </Mapping_Tool>
```

Code 20. Παράδειγμα Configuration File

17.11 RDF - RDFS Parsing

Για να περιγράψουμε την διαδικασία ανάγνωσης θα χρησιμοποιήσουμε ως παράδειγμα το μοντέλο Cidoc CRM το οποίο έχει περιγραφεί στο πρότυπο RDFS. Η διαδικασία όμως παραμένει ίδια για οποιοδήποτε αρχείο RDF. Η μόνη διαφορά υφίσταται στο γεγονός ότι στο πρότυπο RDF βασιζόμαστε στις τριπλέτες για να εξάγουμε πληροφορία σχετική με ιεραρχίες.

17.11.1 Οντότητες (Entities)

Κατά την διάρκεια έναρξης της εφαρμογής το πρόγραμμα αρχικοποιεί το ιεραρχικό μοντέλο οντοτήτων και σχέσεων μέσω του κατάλληλου RDFS αρχείου. Στην εφαρμογή περιέχετε η έκδοση Cidoc 5.0.2 μέσω του αρχείου «cidoc_crm_v5.0.2.rdfs» το οποίο διαβάζετε κάθε φορά κατά την διάρκεια εκκίνησης της εφαρμογής. Επίσης, δίνεται η δυνατότητα σε πραγματικό χρόνο να φορτωθεί μία διαφορετική έκδοση του μοντέλου Cidoc CRM μέσω γραφικού περιβάλλοντος όπου ο χρήστης επιλέγει το κατάλληλο αρχείο ύστερα από περιήγηση στο αποθηκευτικό μέσο.

Η δομή δεδομένων που χρησιμοποιείται είναι η HashMap λόγω της γρήγορης αναζήτησης που προσφέρει. Εξαρχής τοποθετείτε η οντότητα E1. Έπειτα βρίσκουμε όλες τις οντότητες που έχουν ως γονέα αυτή την οντότητα. Για κάθε μία από τις οντότητες που αποτελούν παιδιά του E1, αν η οντότητα αυτή υπάρχει στο HashMap τότε συνδέουμε την οντότητα E1 με την οντότητα παιδί. Διαφορετικά αν η οντότητα παιδί δεν υπάρχει στο HashMap τότε την εισάγουμε, δημιουργούμε το Link ανάμεσα στην οντότητα E1 και στο παιδί, θέτουμε την οντότητα παιδί ως τρέχουσα οντότητα και ξανακαλούμε όλη την διαδικασία. Με αυτόν τον τρόπο φτιάχνεται όλη η ιεραρχία του μοντέλου Cidoc αναδρομικά περιέχοντας όλες τις οντότητες καθώς και τις σχέσεις κληρονομικότητας μεταξύ τους.

```
        Insert E1 Entity
        Current_entity = E1;
Start   Find_all_Childs
        For ( all_Childs )
        {
                If( entity_exist_in_hashmap)
                Link(Current_entity,child);
                Else
                {
                        Insert_child_in_map(child);
                        Link(Current_entity,child);
                        Current_entity = child;
                        Go to Start;
                }
        }
```

Figure 91. Ψευδοκώδικας κατασκευής ιεραρχικού δέντρου για οντότητες

Στο παρακάτω παράδειγμα φαίνεται ένα στιγμιότυπο εκτέλεσης του αλγορίθμου που αναφέραμε καθώς και τα βήματα που εκτελούνται. Τα αντικείμενα δημιουργούνται βάση χρόνου με την ακόλουθη σειρά η οποία είναι E1, E2, E3, E4, … , E52, E53, E54, E77 και έπειτα συνεχίζει με τους απογόνους του E77. Ο αλγόριθμος όντας αναδρομικός ακολουθεί λογική depth first. Το αποτέλεσμα του αποτελεί μία προβολή ενός δέντρου σε πίνακα και ο λόγος που επιλέχτηκε αυτή η δομή ήταν ότι προσφέρει πολύ γρήγορη αναζήτηση καθώς και οργάνωση πληροφορίας χάρη στην συνάρτηση Hash.

Οι οντότητες έχουν χρωματιστεί ανά επίπεδο βάθους αναδρομής έχοντας κόκκινο χρώμα οι οντότητες του πρώτου επιπέδου, πράσινο χρώμα του δεύτερου επιπέδου «που αποτελούν παιδιά της οντότητας E1» και μπλε χρώμα του τρίτου επιπέδου «που αποτελούν παιδιά του E2».

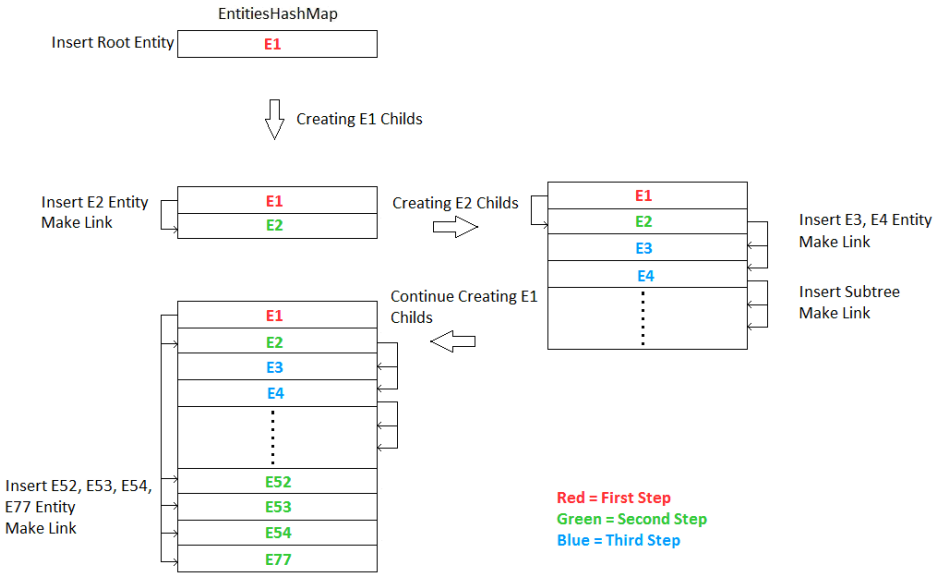


Figure 92. Δημιουργία Cidoc CRM δέντρου ιεραρχίας για οντότητες

17.11.2 Σχέσεις (Properties)

Ανάλογα με τα Entities έτσι και τα Properties ακολουθούν την ίδια νοοτροπία δημιουργίας ιεραρχικού δέντρού κληρονομικότητας. Σε αντίθεση με τις οντότητες, οι σχέσεις δεν διαθέτουν μία σχέση που να λειτουργεί ως ρίζα του δέντρου και την οποία σχέση κληρονομούν τα παιδιά του. Γι' αυτό και για λόγους προγραμματιστικών αναγκών ορίστηκε μια νέα σχέση με αναγνωριστικό "SUPERVISOR" η οποία διαθέτει ως παιδιά της όλες τις σχέσεις οι οποίες δεν διαθέτουν πρόγονο και που κληρονομούν από αυτό. Οι προγραμματιστικοί λόγοι αναφέρονται στο γεγονός ότι οι Java οντότητες τύπου JTree (που αναπαριστούν γραφικά τις οντότητες και τις σχέσεις) απαιτούν την υποχρεωτική ύπαρξη δενδρικής μορφής.

Για να είναι η εφαρμογή πιο φιλική προς τον χρήστη και για να βοηθήσουμε τον χρήστη στο έργο που προσπαθεί να φέρει εις πέρας το ιεραρχικό δέντρο σχέσεων αναδημιουργείτε εκ νέου κάθε φορά που εισάγεται ή διαγράφεται οντότητα στον εικονικό κόσμο. Στην νέα έκδοση του δέντρου θα υπάρχουν αποκλειστικά και μόνο σχέσεις οι οποίες είναι σε θέση να εισαχθούν, δηλαδή που θα διαθέτουν Domain ή Range Entity οντότητα που θα υπάρχει στον εικονικό κόσμο.

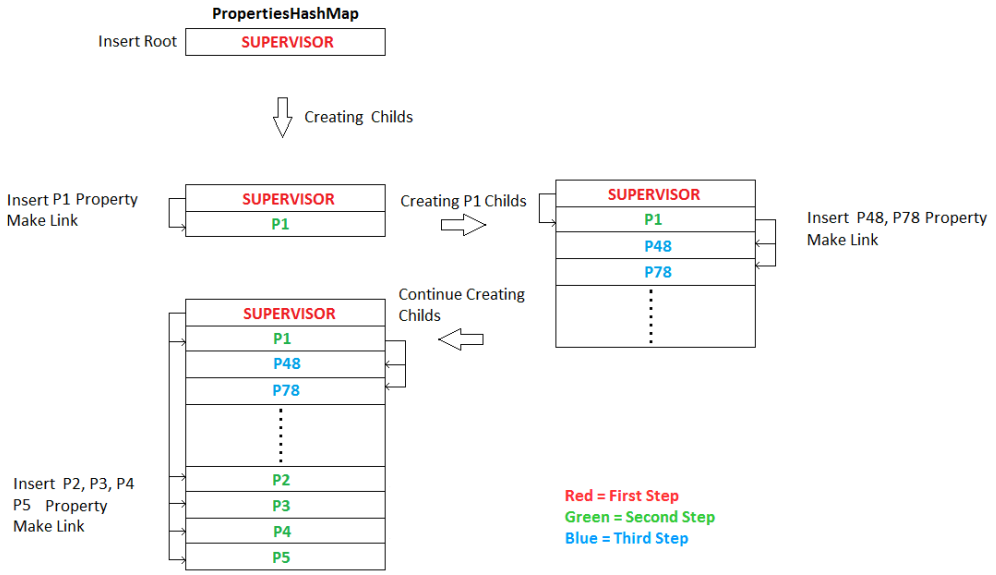


Figure 93. Δημιουργία Cidoc CRM δέντρου ιεραρχίας για σχέσεις

17.11.3 JTree

Το HashMap που φέρει την ιεραρχία οντοτήτων του μοντέλου Cidoc θα πρέπει να μετατραπεί σε δενδρική μορφή που να αποτελείται από αντικείμενα τύπου DefaultMutableTreeNode και η δενδρική αυτή μορφή δίνεται ως όρισμα στην οντότητα τύπου JTree η οποία είναι σε θέση να παράγει την γραφική δενδρική αναπαράσταση.

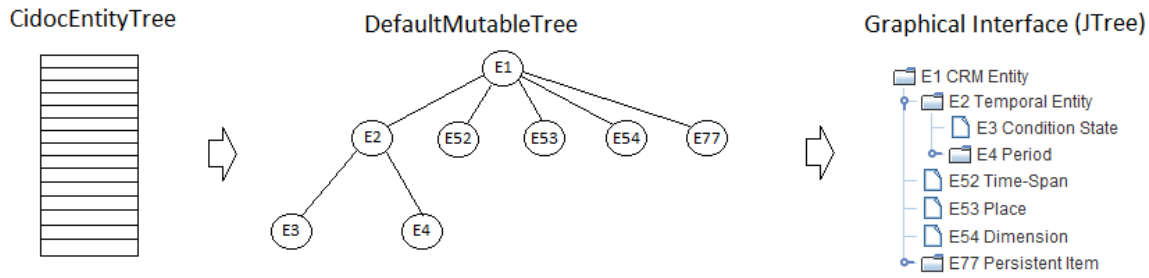


Figure 94. HashMap σε JTree

Σε ενδεχόμενη μελλοντική επέκταση της εφαρμογής το πρόγραμμα θα μπορούσε να προσαρμοστεί στο καινούριο πρότυπο εύκολα και γρήγορα, εξελίσσοντας απλά και μόνο το parsing του νέου προτύπου καθώς το υπόλοιπο προγραμματιστικό κομμάτι της εφαρμογής σχεδιάστηκε με γνώμονα την προσαρμογή και την συμβατότητα με οποιοδήποτε τύπο εισόδου δεδομένων.

Έτσι καθώς θα έχει δημιουργηθεί επιτυχώς ένα parsing της πληροφορίας, δηλαδή θα έχουν δημιουργηθεί σωστά τα EntityMap και τα PropertyMap, τότε με την σειρά τους θα δημιουργηθούν κατάλληλα και τα DefaultMutableTree καθώς και μετέπειτα η γραφική αναπαράσταση των JTree.

Συμπερασματικά μπορούμε να πούμε ότι η εφαρμογή θα μπορούσε να επεκταθεί και εκτός του προτύπου της RDF και να εφαρμοστεί σε διαφορετικές μορφές αναπαράστασης πληροφορίας.

17.12 Διεπαφή Διαδικτύου

Η εφαρμογή σχεδιάστηκε εξαρχής με την προοπτική να είναι όσο το δυνατόν πιο ευέλικτη και προσαρμόσιμη σε αλλαγές, καθώς και σε διαφορετικά περιβάλλοντα. Γι' αυτό αναπτύχθηκε με τρόπο κατάλληλο ούτως ώστε να μπορεί πολύ εύκολα να μετατραπεί σε μια απόλυτα λειτουργική Web εφαρμογή.

1) Αλλάζουμε την δήλωση της βασικής κλάσης που περιέχει την μέθοδο main ώστε να κληρονομεί την κλάση JApplet και όχι την JFrame.

2) Μετατρέπουμε το αναγνωριστικό της μεθόδου initComponents() σε init().

3) Θέτουμε σε σχόλια την υλοποίηση της μεθόδου addWindowListener().

4) Οι διευθύνσεις αρχείων που στεγάζονταν στον δίσκο πρέπει να μετατραπούν σε διευθύνσεις URL που θα παραπέμπουν στον ιστότοπο που φιλοξενεί την εφαρμογή μας.

5) Το IDE ανάπτυξης της εφαρμογής θα πρέπει να ρυθμιστεί ώστε να παραχθεί κώδικας συμβατός με Web εφαρμογές. Στο εργαλείο ανάπτυξης Java εφαρμογών Netbeans, αυτό μπορεί γίνει μέσω επιλογής Properties του Project, έπειτα Web Start, Enable Web Start + Applet Descriptor.

6) Τοποθέτηση κώδικα και δεδομένων στον επιθυμητό ιστότοπο.

Έπειτα από την εκτέλεση αυτών των έξη βημάτων η εφαρμογή θα είναι σε θέση να εξυπηρετεί οποιονδήποτε χρήστη ανά τον κόσμο, οποιανδήποτε στιγμή.

17.13 Λειτουργία αντιστοίχησης

Τα λογισμικά αντιστοίχησης οντοτήτων και εννοιών χρησιμοποιούνται για την δημιουργία δομημένων μοντέλων διαγραμμάτων και σύμφωνα με έρευνες μπορούν να βελτιώσουν σημαντικά τις ικανότητες μάθησης και αντίληψης των χρηστών σε σχέση με τον συμβατικό τρόπο δημιουργίας σημειώσεων που όλοι έχουμε χρησιμοποιήσει.

Τα διαγράμματα αυτά μπορούν να δείξουν τις σχέσεις που υπάρχουν μεταξύ οντοτήτων και εννοιών βασιζόμενοι σε ένα γραφικό εργαλείο που χρησιμοποιείται για την οργάνωση και την αναπαράσταση γνώσης. Όπως ένας χάρτης μας βοηθάει να κατανοήσουμε πιο γρήγορα και εύκολα μία τοποθεσία έτσι και τα διαγράμματα αυτά μας βοηθούν να κατανοήσουμε πιο εύκολα και γρήγορα την πληροφορία που μας παρουσιάζεται.

Για την εικονική αναπαράσταση της πληροφορίας συνήθως χρησιμοποιούνται βασικά γεωμετρικά σχήματα τα οποία αντιλαμβάνεται πιο εύκολα ο ανθρώπινος εγκέφαλος. Οι οντότητες – έννοιες συνήθως αναπαριστούνται με τετράγωνα ή κύκλους και οι σχέσεις μεταξύ τους με ευθύγραμμα τμήματα που φέρουν κάποια πληροφορία κειμένου. Ο συνδυασμός αυτών των δύο σχηματίζει μία δομημένη μορφή ιεραρχικού δέντρου όπου οι πιο γενικές οντότητες τοποθετούνται συνήθως πιο ψηλά.

Τα διαγράμματα αντιστοίχησης είναι ένας τρόπος ανάπτυξης λογικών και μαθησιακών ικανοτήτων καθώς αποκαλύπτουν σχέσεις και την γενικότερη ιδέα που κρύβεται στην ιεραρχία των προς αναπαράσταση μοντέλων. Επίσης ο τρόπος αναπαράστασης της πληροφορίας με οντότητες , γεγονότα, σχέσεις είναι ιδιαίτερα κατανοητός από τον ανθρώπινο εγκέφαλο καθώς έρευνες έχουν δείξει ότι και αυτός χειρίζεται με παρόμοιο τρόπο που αντιλαμβάνεται στην καθημερινότητά του.

Συμπερασματικά θα λέγαμε ότι οι δυνατότητες του εφάπτονται στην δημιουργία, οπτικοποίηση, δόμηση και κατηγοριοποίηση πληροφορίας και λειτουργεί ως υποβοήθημα στην κατανόηση και οργάνωση της πληροφορίας [70],[71] , στην λήψη αποφάσεων και λύση προβλημάτων.

Η θέση που καταλαμβάνει κάθε οντότητα δεν είναι τυχαία αλλά έχει ως στόχο την εύκολη και γρήγορη διαισθητική αντίληψη των σχέσεων και των ιεραρχιών από τον χρήστη. Γι’ αυτό στην εφαρμογή μας ίθυσται οι σχέσεις μεταξύ οντοτήτων να τοποθετούνται οριζόντια και οι σχέσεις Isa να τοποθετούνται σε διαφορετικό ύψος.

Στον παρακάτω πίνακα περιγράφουμε την δομή της διαδικασίας δημιουργίας αντιστοιχήσεων ανάμεσα σε οντότητες και σχέσεις του Source Model και του Target Model. Η δημιουργία αντιστοιχήσεων χωρίζεται σε δύο βασικές κατηγορίες οι οποίες

είναι πρώτων η Direct (άμμεση) και η MultiMapping. Τονίζουμε ότι για να μπορέσουμε να εκτελέσουμε αντιστοιχήσεις θα πρέπει να έχουμε μπει σε Mapping Mode.

17.13.1 Απευθείας Αντιστοίχηση (Direct - Instant Mapping)

Στην πρώτη κατηγορία επιλέγονται διπλέτες από αντικείμενα που το ένα ανήκει στο Source Model και το άλλο στο Target Model. Η επιλογή αυτή μπορεί να γίνει με τυχαία σειρά και από την στιγμή που έχουν επιλεγεί δύο διαφορετικά αντικείμενα από διαφορετικά μοντέλα το πρόγραμμα εισάγει την γραμμή στον τρισδιάστατο χώρο που υποδηλώνει την αντιστοίχηση μεταξύ τους. Καθώς το πρόγραμμα θα έχει τεθεί σε Mapping Mode δεν είναι ανάγκη η ακολουθιακή δήλωση του χρήστη για εισαγωγή νέας συσχέτισης ανάμεσα σε αντικείμενα του Source και Target μοντέλου. Χάρη σε αυτό ο χρήστης θα μπορεί επαναληπτικά να εκτελεί αντιστοιχήσεις έως ότου δηλώσει την επιθυμία του για επαναφορά του συστήματος στην κανονική κατάσταση λειτουργίας. Παράδειγμα εφαρμογής όσων αναφέραμε παρουσιάζουμε στην παρακάτω εικόνα όπου η οντότητα Temporal Event του Cidoc CRM μοντέλου έχει αντιστοιχηθεί με μία οντότητα τύπου Event κάποιου άλλου μοντέλου.

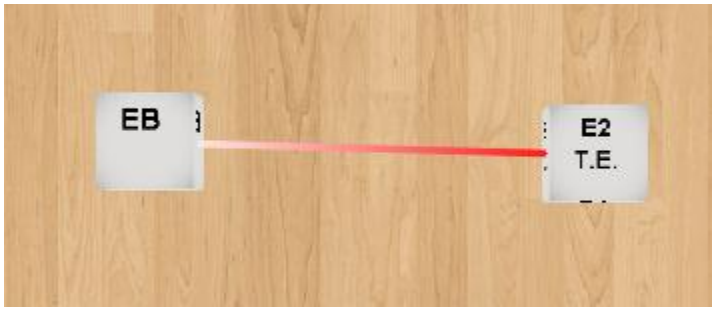


Figure 95. Παράδειγμα απευθείας αντιστοίχησης

17.13.2 Πολλαπλή Αντιστοίχηση (MultiMapping)

Στην δεύτερη κατηγορία μπορεί να γίνει αντιστοίχηση ανάμεσα σε περισσότερα από ένα αντικείμενα που να ανήκουν σε διαφορετικά μοντέλα. Για παράδειγμα θα μπορούσαν να αντιστοιχηθούν ταυτόχρονα δύο οντότητες που ανήκουν στο Source Model σε τρεις οντότητες που ανήκουν στο Target Model. Στην κατηγορία αυτή η εφαρμογή δύναται να υποστηρίξει αντιστοίχησης με σχέση 1-N, N-1, N-N ανάμεσα στα δύο μοντέλα.

Για να μπορέσει η εφαρμογή να αναγνωρίσει την πρόθεση του χρήστη, να εξασκήσει την δυνατότητα της πολλαπλής αντιστοίχησης, θα πρέπει ο χρήστης να έχει ενεργοποιήσει καθ’ όλη την διάρκεια της διαδικασίας το πλήκτρο «Control». Σε

αντίθετη περίπτωση όπου το πλήκτρο «Control» δεν έχει ενεργοποιηθεί η εφαρμογή θα προσπαθήσει να εκτελέσει Direct Mapping.

Source Object	Target Object	Mapping Type	Control Key Pressed
1	1	Direct (Instant Mapping)	False
1	N	MultiMapping (Design On The Fly)	True
N	1	MultiMapping (Design On The Fly)	True
N	N	MultiMapping (Design On The Fly)	True

Array 3. Τύποι αντιστοίχησης

Κατά την διάρκεια εκτέλεσης πολλαπλής αντιστοίχησης η σχεδίαση των γραμμών αντιστοίχησης ολοκληρώνετε σε πραγματικό χρόνο και ενδέχεται να μετατραπεί έχοντας ως γνώμονα την αυτόματη προσαρμογή του Layout στην εκάστοτε κατάσταση και στις εντολές του χρήστη. Το σύστημα καθ' όλη την χρονική διάρκεια κατά την οποία ο χρήστης έχει πατημένο το πλήκτρο Control παρατηρεί τις νέες επιλογές εικονικών αντικειμένων από τον χρήστη και καταχωρεί αυτές τις επιλογές σε λίστες αναμονής ανάλογα με το μοντέλο στο οποίο ανήκουν τα αντικείμενα. Αργότερα το σύστημα προελαύνει αυτές τις λίστες για να εντοπίσει όλους τους πιθανούς συνδυασμούς που μπορούν να προκύψουν.

Για παράδειγμα, σε περίπτωση Properties MultiMapping όπου όλα τα Mapping Lines ενώνονται σε ένα κεντρικό σημείο και έπειτα οδηγούνται στο Target Model, οι γραμμές που εξομοιώνουν την αντιστοίχηση δημιουργούνται σταδιακά σύμφωνα με τις οδηγίες του χρήστη.

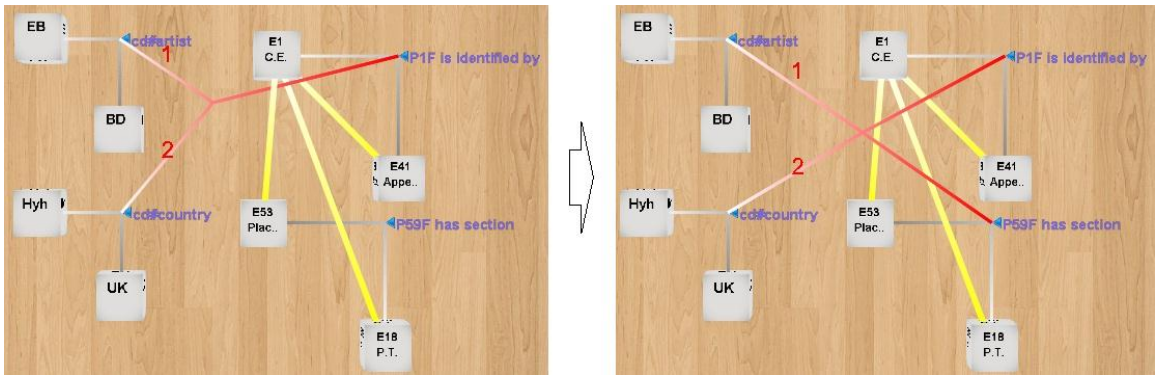


Figure 96. Πολλαπλή αντιστοίχηση για σχέσης

17.14 Συνθήκες Αντιστοίχησης (Mapping Conditions)

Η αντιστοίχηση σχημάτων είναι μια μετατροπή πληροφορίας χάρη στην οποία δεδομένου ενός στιγμιότυπου σχήματος Source, μπορεί να παράγει ένα στιγμιότυπο σχήματος Target διατηρώντας το περιεχόμενο και την σημασιολογία της πληροφορίας κατά την μετατροπή. Επίσης, η αντιστοίχηση σχημάτων είναι πλέον μία πολύ διαδεδομένη πρακτική που χρησιμοποιείται κατά την ανταλλαγή πληροφορίας ανάμεσα σε πηγές που χρησιμοποιούν διαφορετικά σχήματα.

Όμως, κατά την διάρκεια αντιστοίχησης αντικειμένων ανάμεσα σε δύο μοντέλα παρατηρείται η ανάγκη χειροκίνητου περιορισμού του πλήθους τιμών που μπορεί να λάβει η οντότητα τύπου Target σε σχέση με την οντότητα τύπου Source.

Εξαιτίας αυτού του γεγονότος δίνουμε την δυνατότητα στον χρήστη μετά την εισαγωγή μίας αντιστοίχησης ανάμεσα σε μοντέλα να εισάγει περιορισμούς που να αφορούν τις αντιστοιχήσεις στιγμιότυπων που θα προκύψουν μελλοντικά. Οι περιορισμοί αυτοί δίνονται υπό την μορφή ελεύθερου κειμένου και γίνεται εξαγωγή τους στο XML αρχείο που παράγεται και το οποίο φέρει όλη την διαθέσιμη πληροφορία για την αντιστοίχηση σχημάτων.

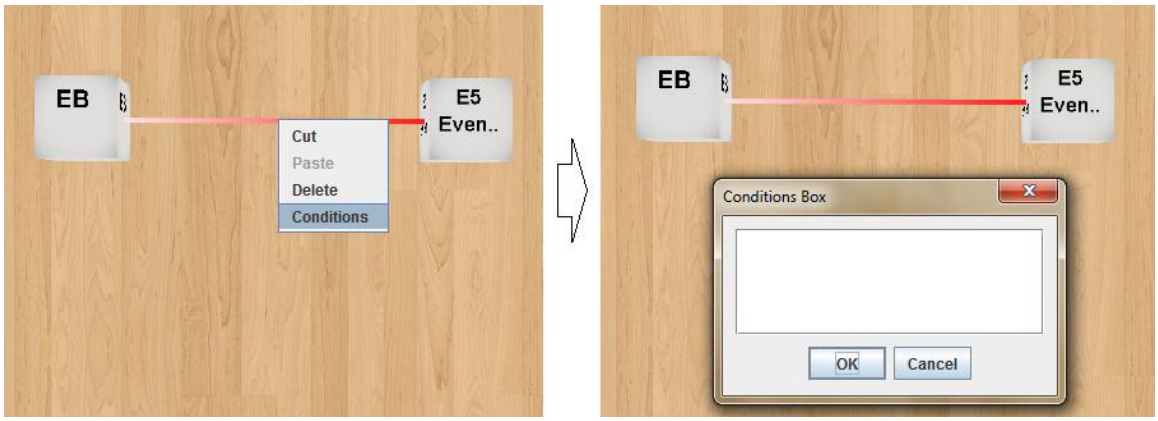


Figure 97. Παράδειγμα συνθήκης αντιστοίχησης

Η πληροφορία που εισάγουμε στο πλαίσιο περιορισμών «Conditions Box» συνοδεύει την γραμμή αντιστοίχησης καθ' όλες τις μεταβολές που επιδρούν επάνω της έως ότου αυτή διαγραφεί.

17.15 Εξαγωγή αντιστοίχησης σε XML

Παράγωγο της διαδικασίας αντιστοίχησης αποτελεί η δημιουργία ενός αρχείου XML που θα φέρει όλη την απαραίτητη πληροφορία που έχει συγκεντρωθεί κατά την διάρκεια αντιστοίχησης αντικείμενων από το Source Model στο Target Model και η

οποία θα χρησιμοποιηθεί για την μετατροπή στιγμιοτύπων από το πρώτο μοντέλο στο δεύτερο. Για να γίνει εύκολα αντιληπτή η διαδικασία περιγράφουμε παρακάτω ένα χαρακτηριστικό παράδειγμα.

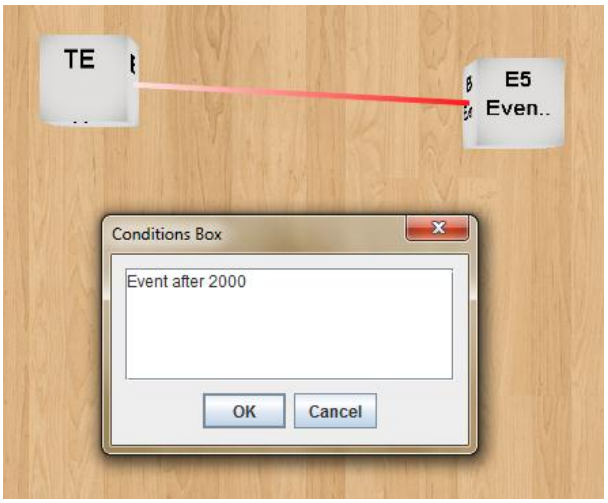


Figure 98. Παράδειγμα αντιστοίχησης Time Event

Στο παράδειγμα αυτό γίνεται αντιστοιχία ανάμεσα σε μία οντότητα του Source μοντέλου με την επονομασία «Time Event» στην οντότητα E5 Event του μοντέλου Cidoc. Επίσης τίθεται ως περιορισμός η απαίτηση του ότι το γεγονός θα πρέπει να έχει συμβεί μετά το έτος 2000. Αυτή η αντιστοιχία έχει ως αποτέλεσμα την παραγωγή του παρακάτω αρχείου XML.

Στο αρχείο αυτό περιγράφεται η αντιστοιχία ανάμεσα σε δύο οντότητες. Από το Source μοντέλο η οντότητα που περιγράφεται είναι η Time Event και από το Target μοντέλο περιγράφεται η οντότητα E5. Στο αρχείο αυτό βλέπουμε επίσης ότι για να μετατραπεί μία οντότητα τύπου Time Event σε μία τύπου E5 θα πρέπει να συμμορφώνεται με τον περιορισμό που απαιτεί το γεγονός να έχει συμβεί μετά το 2000.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Mapper>
3          <Entities>
4                  <Entity>
5                          <Source>http://www.sourcemodel.com/schema/Time Event</Source>
6                          <Target>E5</Target>
7                          <Conditions>Event after 2000</Conditions>
8                  </Entity>
9          </Entities>
10         <Properties />
11 </Mapper>
```

Figure 99. Παραγόμενο XML για Time Event Mapping

18 Έλεγχος Απόδοσης Συστήματος

Για την αξιολόγηση της εφαρμογής έγιναν πειράματα με συγκεκριμένο αριθμό οντοτήτων στον εικονικό κόσμο. Το αποτέλεσμα προκύπτει ως ο μέσος όρος των αποτελεσμάτων δέκα επαναλήψεων του ιδίου πειράματος με τα ίδια δεδομένα και παρόμοιες συνθήκες περιβάλλοντος εκτέλεσης. Όλα τα πειράματα έγιναν σε δυπήρινο επεξεργαστή με συχνότητα στα 2 GHertzs, μνήμη 2 GByte, κάρτα γραφικών με αυτόνομη μνήμη 256 MB και κοινόχρηστη 1024MB και DirectX 11.

Τα πειράματα αφορούν σε πρώτη φάση τον χρόνο εισαγωγής των οντοτήτων στον εικονικό κόσμο, έπειτα την καταπόνηση του επεξεργαστή κατά την διάρκεια εισαγωγής των οντοτήτων στον εικονικό κόσμο, την μνήμη που καταναλώνει η εφαρμογή από την στιγμή που έχουν εισαχθεί οι οντότητες, την κατανάλωση επεξεργαστικής ισχύς καθώς οι οντότητες υπάρχουν στον εικονικό κόσμο αλλά δεν βρίσκονται σε κίνηση, την κατανάλωση επεξεργαστικής ισχύς όταν οι τρισδιάστατες οντότητες βρίσκονται σε κίνηση και τέλος την αξιολόγηση της συμπεριφοράς του συστήματος από τον χρήστη.

Ο αριθμός των οντοτήτων στα πειράματα είναι 10, 20, 50, 100, 200, 400, 600, 800, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 5000 και ο αριθμός των σχέσεων κυμαίνεται σε περίπου τρεις ανά οντότητα.

Στο παρακάτω παράδειγμα μετρήσαμε τον χρόνο που είναι απαραίτητος για την εισαγωγή των οντοτήτων στον εικονικό κόσμο και η μονάδα μέτρησης που χρησιμοποιείται είναι το «millisecond». Παρατηρούμε ότι η σχέση ανάμεσα στις οντότητες και στον χρόνο που χρειάζονται για να εισαχθούν είναι σχεδόν γραμμική και η γραμμή σύνδεση των σημείων πλησιάζει σχεδόν την ευθεία. Η απόδοση αυτή κρίνεται ως ιδιαίτερα ευνοϊκή καθώς υπερτερεί τυχόν εκθετικής πολυπλοκότητας και είναι ανάλογη των οντοτήτων με μία αναλογία της τάξης 1/100.

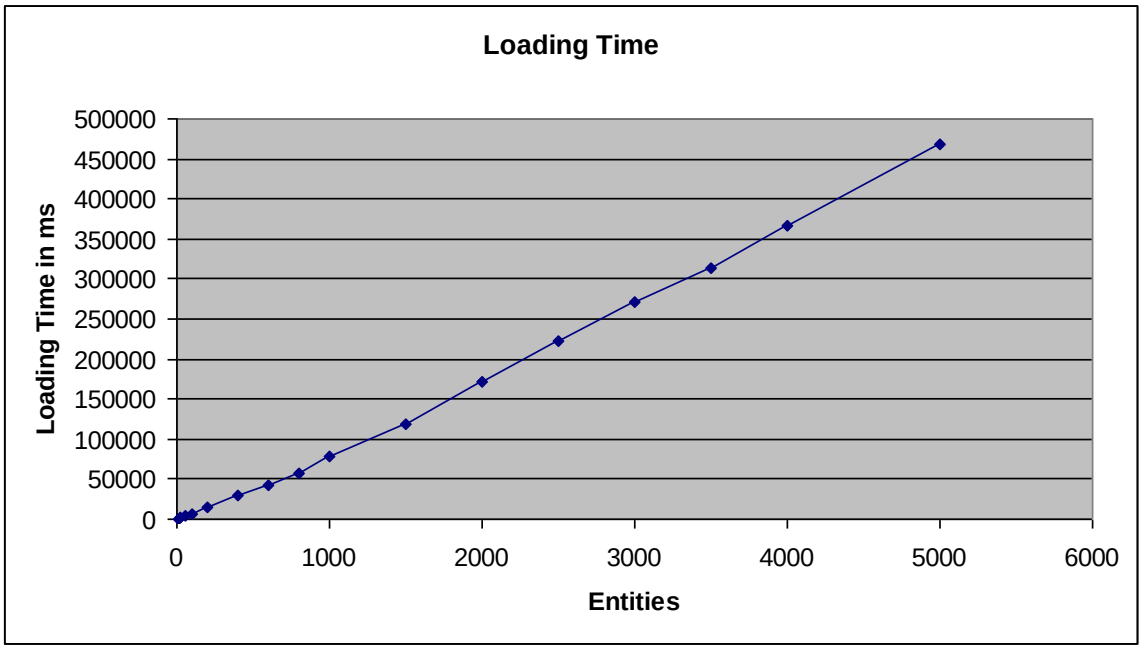


Chart 1. Χρόνος φόρτισης οντοτήτων

Στην εικόνα που ακολουθεί παρουσιάζουμε την χρήση επεξεργαστικής ισχύς κατά την διάρκεια εισαγωγής οντοτήτων στον εικονικό κόσμο ανά πλήθος οντοτήτων. Η γραφική παράσταση θυμίζει έντονα εκείνη της γραφικής απεικόνισης λογαριθμικής συνάρτησης με βάση μεγαλύτερη της μονάδας $g(x) = \log_{\alpha} x$, $\alpha>1$. Οι μετρήσεις αναφέρονται στην μέγιστη καταπόνηση του επεξεργαστή κατά την διάρκεια εισαγωγής οντοτήτων. Η χρήση αυτής της επεξεργαστικής ισχύς εφάπτεται σε εσωτερικούς μηχανισμούς της Java3D και είναι σε θέση να κάνει πλήρη χρήση του επεξεργαστή από 2000 οντότητες και άνω.

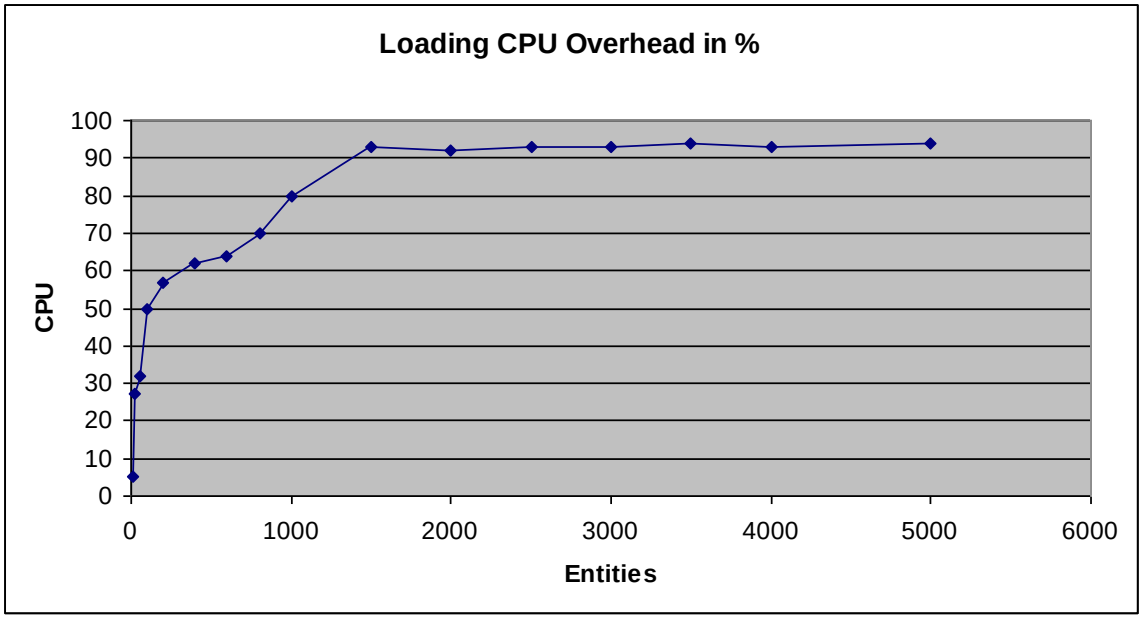


Chart 2. Επιβάρυνση CPU κατά την φόρτιση σε ποσοστό

Η διαχείριση μνήμης τείνει να έχει γραμμική σχέση με το πλήθος των οντοτήτων που έχουν εισαχθεί στον εικονικό κόσμο με αναλογία η οποία τείνει να ισούται με το συντελεστή 0.15 . Η χρήση αυτή της μνήμης υφίσταται χάρη στις δομές δεδομένων που χρησιμοποιεί εσωτερικά ο μηχανισμός της Java3D για την απεικόνιση αντικειμένων καθώς οι δομές δεδομένων και η πληροφορία που διαχειρίζεται η εφαρμογή μας έχουν μέγεθος αμελητέο σε σχέση με το μέγεθος των αποτελεσμάτων των μετρήσεων κατανάλωσης μνήμης.

Αξίζει να σημειωθεί ότι η όλη σχεδίαση και υλοποίηση του συστήματος έγινε σύμφωνα με τις αρχές και τους κανόνες της Java3D που δίνουν και το βέλτιστο αποτέλεσμα με την μικρότερη δυνατή κατανάλωση μνήμης και επεξεργαστικής ισχύς.

Οι μετρήσεις έγιναν με μονάδα μέτρησης το MegaByte και από το λειτουργικό σύστημα έχουν δοθεί διαθέσιμα στην εφαρμογή μας τουλάχιστον 1000 MB μνήμης.

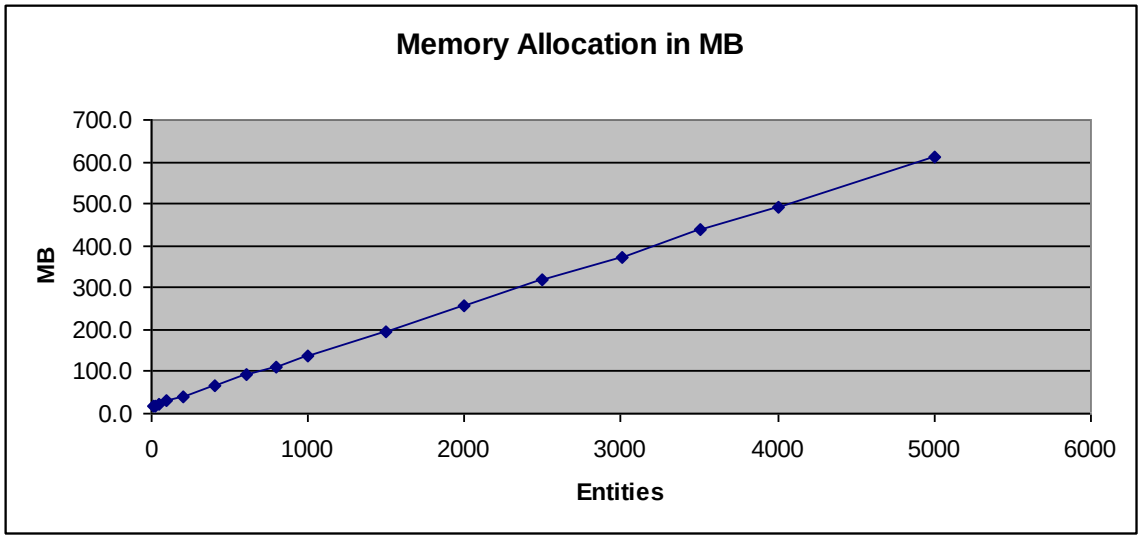


Chart 3. Επιβάρυνση μνήμης σε MegaByte

Κατά την διάρκεια αδρανούς εκτέλεσης της εφαρμογής η χρήση επεξεργαστικής ισχύς κρίνεται ως αμελητέα και κυμαίνεται ανάμεσα στο 1 και 3 της εκατό.

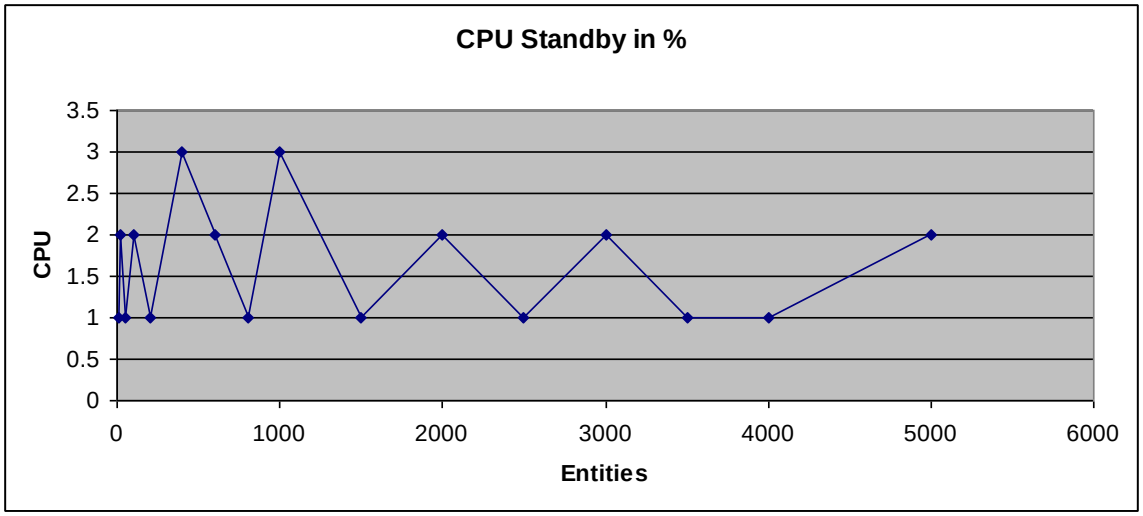


Chart 4. Επιβάρυνση CPU σε ποσοστό σε κατάσταση αδράνειας

Η κίνηση στον τρισδιάστατο χώρο απαιτεί την πραγματοποίηση μεγάλου πλήθους υπολογισμών σε πραγματικό χρόνο καθώς επίσης και την διατήρηση της σχετικής θέσης μεταξύ τους. Το αποτέλεσμα των πειραμάτων κρίνεται ως λογικό και αναμενόμενο καθώς η εφαρμογή μπορεί να χειριστεί μαζικά όλες τις οντότητες ώστε να μετακινηθούν στον εικονικό κόσμο στους άξονες X, Y, Z ή να περιστρεφούν.

Παρά την όλη απαιτούμενη επεξεργαστική ισχύ, η εφαρμογή είναι σε θέση να ανταπεξέλθει σε πραγματικό χρόνο στις εντολές και στις απαιτήσεις του χρήστη. Παρατηρείτε ραγδαία αύξηση χρήσης επεξεργαστή όταν στον εικονικό κόσμο έχουν εισαχθεί περισσότερες από 400 οντότητες και καθυστέρηση στην ανταπόκριση όταν στον εικονικό κόσμο έχουν εισαχθεί περισσότερες από 3000 οντότητες.

Μετά τις περίπου 3500 οντότητες παρατηρείτε μια σταδιακή επιβράδυνση της απόδοσης και ανταπόκρισης της εφαρμογής.

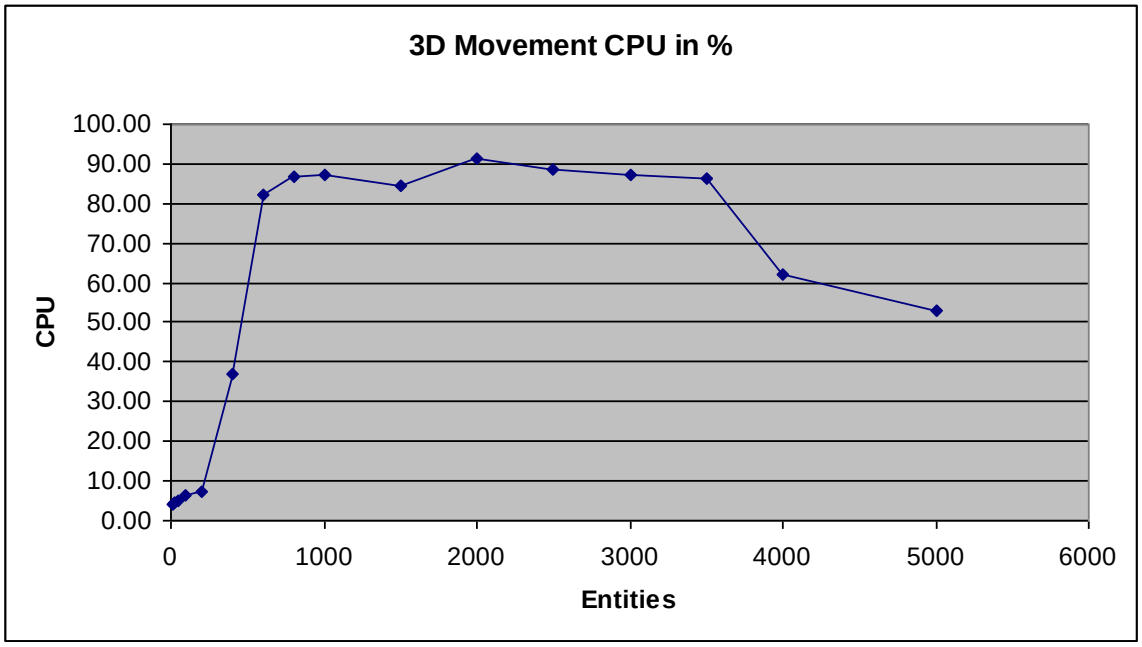


Chart 5. Επιβάρυνση CPU κατά την διάρκεια κίνησης σε ποσοστό

Η ποιότητα χαρακτηρίζεται ως εξαιρετική έως τις 1500 οντότητες, πολύ καλή ή καλή έως τις 2500 οντότητες, σχετικά καλή έως τις 3500 οντότητες και από αυτό το σημείο και πέρα θεωρείται ως κακή ή ελλιπής. Θα πρέπει να τονίσουμε ότι οι υπολογισμοί αυτοί έχουν γίνει έχοντας εισάγει το συγκεκριμένο πλήθος οντοτήτων στον εικονικό κόσμο, κατά μέσο όρο τρεις σχέσεις ανά οντότητα καθώς επίσης και τις σκιές που τις ακολουθούν.

Χάρη σε αυτό, το συνολικό πλήθος αντικειμένων που υπάρχουν στον εικονικό κόσμο και των οποίων την μετακίνηση υπολογίζουμε είναι πολλαπλάσιο του πλήθους των οντοτήτων που αναφέρουμε στις μετρήσεις και ισούται με οκτώ φορές το πλήθος τους. Αυτό συμβαίνει καθώς η κάθε οντότητα έχει τρεις σχέσεις και το καθένα από αυτά τα αντικείμενα έχει μία σκιά. Έτσι το πλήθος των 3D Objects ισούται με οκτώ φορές το πλήθος των Entities.

$$|3D_Objects| = 8*|Entities|$$

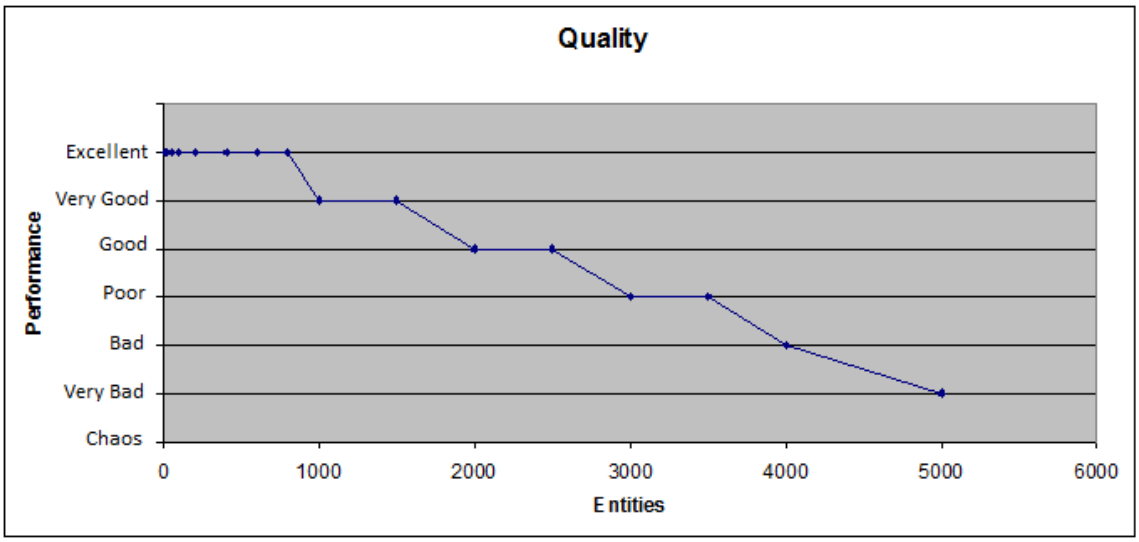


Chart 6. Ανάλυση απόδοσης – ποιότητας

19 Μελλοντικές Εξελίξεις (Future work)

Το πρόγραμμα που αναπτύξαμε ανοίγει ένα νέο παράθυρο για πολλές μελλοντικές επεκτάσεις και εφαρμογές. Στην παρούσα κατάσταση η εφαρμογή μας απευθύνεται σε αντιστοίχησης μοντέλων που βασίζονται στο πρότυπο RDF. Όμως παρότι το πρότυπο RDF είναι πολύ διαδεδομένο, δεν αποτελεί το μοναδικό πρότυπο που υπάρχει στην επιστήμη των υπολογιστών. Επιπλέον, δεν μπορούμε να αποκλείσουμε την μελλοντική εξέλιξη της RDF ή την δημιουργία ενός καινούριου προτύπου το οποίο θα είναι πιο εξελιγμένο από την αυτήν.

Έχοντας λάβει αυτά υπόψη, οδηγηθήκαμε στο συμπέρασμα ότι η εξαρχής σχεδίαση του συστήματος θα έπρεπε να είναι σύμφωνη με μελλοντικές επεκτάσεις ή εξελίξεις, οι οποίες αλλαγές που θα πρέπει να γίνουν, θα απευθύνονται σε ένα μικρό προγραμματιστικό κομμάτι της εφαρμογής. Εφαρμόζοντας λοιπόν αυτές τις αλλαγές, το σύστημα θα έπρεπε να είναι σε θέση να προσαρμοστεί αυτόματα στον νέο τύπο δεδομένων δίχως την ύπαρξη ανάγκης περισσότερων επεμβάσεων σε πιο δομικά και λειτουργικά στοιχεία όπως για παράδειγμα η σχεδίαση των τρισδιάστατων αντικείμενων ή την μεταφορά τους στο τρισδιάστατο χώρο.

Σύμφωνα λοιπόν με τους παραπάνω όρους δεσμευόμαστε ότι σε ενδεχόμενη μελλοντική επέκταση της εφαρμογής σε ένα καινούριο είδος προτύπου σχήματος, το πρόγραμμα θα μπορούσε να προσαρμοστεί στο καινούριο πρότυπο εύκολα και γρήγορα, εξελίσσοντας απλά και μόνο την κλάση που θα είναι υπεύθυνη για το parsing του νέου προτύπου, καθώς το υπόλοιπο προγραμματιστικό κομμάτι της εφαρμογής σχεδιάστηκε με γνώμονα την αυτόματη προσαρμογή και την συμβατότητα με οποιοδήποτε τύπο εισόδου δεδομένων.

Έτσι καθώς θα έχει δημιουργηθεί επιτυχώς ένα parsing της πληροφορίας, δηλαδή θα έχουν δημιουργηθεί σωστά τα EntityMap και τα PropertyMap (τα οποία είναι προγραμματιστικές οντότητες που φέρουν όλη την πληροφορία για τα Entities και τα Properties ενός μοντέλου), τότε με την σειρά τους θα δημιουργηθούν κατάλληλα και τα DefaultMutableTree καθώς και μετέπειτα η γραφική αναπαράσταση που προσφέρουν τα αντικείμενα τύπου JTree.

Συμπερασματικά μπορούμε να πούμε ότι η εφαρμογή δεν δεσμεύεται μόνο σε πρότυπα που έχουν οριστεί στην RDF αλλά θα μπορούσε να επεκταθεί και εκτός της RDF και να εφαρμοστεί σε οποιαδήποτε διαφορετική μορφή δομημένης αναπαράστασης πληροφορίας.

20 Συμπεράσματα (Conclusion)

Στην παρούσα εργασία προσπαθήσαμε να αποτυπώσουμε στον τρισδιάστατο κόσμο πληροφορία που περιέχεται σε αρχεία τύπου RDF με τον καλύτερο δυνατό τρόπο ώστε ο χρήστης να μπορεί διαισθητικά να αντιλαμβάνεται οντότητες και ιεραρχίες. Επιλέξαμε τεχνικές που αποτρέπουν την απόκρυψη και την επικάλυψη πληροφορίας για να είναι ο χρήστης σε θέση να την αντιλαμβάνεται πλήρως.

Κατασκευάσαμε έναν τρισδιάστατο κόσμο ο οποίος είναι παντού προσβάσιμος και προσφέρει στο χρήστη όλες τις βασικές και απαραίτητες λειτουργίες περιαγωγής όπως Movement, Rotation, Zooming.

Θελήσαμε να βοηθήσουμε τον χρήστη στο έργο του απλοποιώντας και κάνοντας χρήση αυτοματισμών σε διαδικασίες που χειρίζεται πολύ συχνά όπως είναι η εισαγωγή, διαγραφή πληροφορίας και η αντιστοίχηση ανάμεσα σε δύο μοντέλα.

Δώσαμε στον χρήστη την δυνατότητα να αλληλεπιδρά με τον εικονικό κόσμο εισάγοντας, διαγράφοντας, αλλάζοντας εμφανισιακά χαρακτηριστικά και μετακινώντας εικονικά αντικείμενα.

Προσπαθήσαμε εξαρχής να σχεδιάσουμε το σύστημα με χαρακτηριστικά που να επιτρέπουν την μετέπειτα εξέλιξη του ή προσαρμογή του σε ένα διαφορετικό από το RDF πρότυπο αναπαράστασης γνώσεις.

Για να τα καταφέρουμε όλα αυτά παλέψαμε με τις δυνατότητες και τους περιορισμούς που θέτει η Java3D καθώς επίσης και με τις δυνατότητες και τους περιορισμούς που θέτουν τα σημερινά συστήματα χειρισμού και αποτύπωσης γραφικών. Καθώς, επιθυμούμε να παρέχουμε την καλύτερη δυνατή ποιότητα ρεαλισμού σε γραφικά στοιχεία που υπολογίζονται και αποτυπώνονται σε πραγματικό χρόνο.

Παρόλα αυτά δεν μπορέσαμε να αποφύγουμε την απόκρυψη οντοτήτων – πληροφορίας χάρη σε άλλα αντικείμενα που μπορεί να βρίσκονται μπροστά τους. Δεν μπορέσαμε να αποφύγουμε πλήρως αστοχίες της Java3D αλλά περιορίσαμε τις επιπτώσεις τους σε ακίνδυνο βαθμό. Τα πειράματα έδειξαν ότι ένας αυτόματος αλγόριθμος τοποθέτησης αντικειμένων στον εικονικό χώρο δεν είχε επιθυμητά αποτελέσματα για τον χρήστη καθώς τον αποπροσανατόλιζε και χάρη την πολυπλοκότητα των υπολογισμών δεν ήταν δυνατών να ανταποκριθεί σε πραγματικό χρόνο. Επιπλέον, δεν μπορέσαμε να αποφύγουμε την σύγκρουση των γραμμών με αντικείμενα ή μεταξύ τους και τα conditions αποτελούν απλό ελεύθερο κείμενο δίχως δομή ή έλεγχο.

Ωστόσο, το μέλλον είναι πολλά υποσχόμενο καθώς έχει πρόσφατα αρχίσει μία καινούρια εκστρατεία για ανάπτυξη και εξέλιξη γλωσσών προγραμματισμού και hardware που προσφέρουν περισσότερες λειτουργίες και δυνατότητες στην σχεδίαση τρισδιάστατων γραφικών.

21 Αναφορές (References)

[1] ColinWare and Glenn Franck. Viewing a graph in a virtual reality display is three times as good as a 2d diagram. In *Proceedings of 1994 IEEE Visual Languages*, pages 182–183. IEEE, 1994.

[2] Geoffrey S Hubona, Gregory W Shirah, and David G Fout. 3d object recognition with motion. In *Extended Abstracts of CHI'97*, pages 345–346. ACM, 1997.

[3] Ben Shneiderman. *Designing the User Interface*. Addison-Wesley, 3 edition, 1997.

[4] M Sheelagh, T Carpendale, David J Cowperthwaite, and F David Fracchia. 3-dimensional pliable surfaces: For the effective presentation of visual information. In *Proceedings of UIST'95*, pages 217–226. ACM, 1995.

[5] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of 1996 IEEE Visual Languages*, pages 336–343. IEEE, 1996.

[6] G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: Animated 3d visualizations of hierarchical information. In Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, pages 189--194. ACM, April 1991.

[7] GhoniemM., FeketeJ.-D., and CastagliolaP., On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. Information Visualization, 4 (2). 114–135.

[8] S. Zhao, M. J. McGuffin, and M. H. Chignell. Elastic Hierarchies: Combining Treemaps and Node-Link-Diagrams. In *Proc. of InfoVis'05*, pages 57–64, 2005.

[9] Kim M Fairchild, Steven E Poltrock, and George W Furnas. *Cognitive Science and its Applications for Human Computer Interaction*, chapter 5, pages 201–233. Lawrence Erlbaum Associates, 1988. SemNet: Three-dimensional graphic representations of large knowlege bases.

[10] George G Robertson, Jock D Mackinlay, and Stuart K Card. Cone trees: Animated 3d visualizations of hierarchical information. In *Proceedings of SIGCHI'91*, pages 189–194. ACM, 1991.

[11] Tamara Munzner and Paul Burchard. Visualizing the structure of the world wide web in 3d hyperbolic space. In *Proceedings of VRML '95*, pages 33–38. ACM, 1995.

[12] Kenneth C Cox, Stephen G Eick, and Taosong He. 3d geographic network displays. *Sigmod Record*, 25(4):50–54, December 1996.

[13] Jock DMackinlay, George G Robertson, and Stuart K Card. The perspective wall: Detail and context smoothly integrated. In *Proceedings of SIGCHI'91*, pages 173–179. ACM, 1991.

[14] George G Robertson and Jock D Mackinlay. The document lens. In *Proceedings of UIST'93*, pages 101–108. ACM, 1993.

[15] M Sheelagh, T Carpendale, David J Cowperthwaite, and F David Fracchia. 3-dimensional pliable surfaces: For the effective presentation of visual information. In *Proceedings of UIST'95*, pages 217–226. ACM, 1995.

[16] J Tesler and S Strasnick. Fsn: 3d information landscapes, 1992. Man page entry for an unsupported but publically released system from Silicon Graphics, Inc.

[17] Mei C Chuah, Steven F Roth, JoeMattis, and John Kolojejchik. Sdm: Selective dynamic manuipulation of visualizations. In *Proceedings of UIST'95*, pages 61–70. ACM, 1995.

[18] Matthew Chalmers, Robert Ingram, and Christoph Pfranger. Adding imageability features to information displays. In *Proceedings of UIST'96*, pages 33–39. ACM, 1996.

[19] Stuart K Card. Visualizing retrieved information: A survey. *IEEE Computer Graphics and Applications CG&A*, 16(2):63–67, March 1996.

[20] Jun Rekimoto and Mark Green. The information cube: Using transparency in 3d information visualization. In *Proceedings of the Third Annual Workshop on Information Technologies & Systems (WITS'93)*, pages 125–132, 1993.

[21] Bottom-up computation of sparse and Iceberg CUBE, Kevin Beyer, Raghu Ramakrishnan, Proceedings of the 1999 ACM SIGMOD international conference on Management of data.

[22] Steven Feiner and Clifford Beshers. Worlds within worlds: Metaphors for exploring ndimensional virtual worlds. In *Proceedings of UIST'90*, pages 76–83. ACM, 1990.

[23] Stuart K Card. Visualizing retrieved information: A survey. *IEEE Computer Graphics and Applications CG&A*, 16(2):63–67, March 1996.

[24] S. Cameron, "Enhancing gjk: Computing Minimum and Penetration Distance between Convex Polyhedra," *Proceedings of International Conference on Robotics and Automation*, 1997.

[25] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A Fast Procedure for Computing the Distance between Objects in Three-Dimensional Space," IEEE Journal of Robotics and Automation, 4(2), pp193-203, April 1988.

[26] P. M. Hubbard, "Interactive Collision Detection," In *Proceedings of IEEE Symposium on Research Frontiers in Virtual Reality*, October 1993.

[27] J. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, and K. Zikan, "Efficient Collision Detection Using Bounding Volume Hierarchies of K-DOP's," *Siggraph' 96 Visual Proceedings*, 1996.

[28] M. C. Lin, "Efficient Collision Detection for Animation and Robotics," Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, December 1993.

[29] Tamassia, R., and Eades, P. Algorithms for drawing graphs : an annotated bibliography. Tech. Rep. CS-89-09, Brown University, Department of Computer Science, October 1989.

[30] S. Quinlan, "Efficient Distance Computation between Non-Convex Objects," *Proceedings of International Conference on Robotics and Automation*, pp3324-3329, San Diego, CA, 1994.

[31] I. J. Palmer and R. L. Grimsdale, "Collision Detection for Animation Using Sphere-Tree," *Computer Graphics Forum*, 14(2), pp105-116, June 1995.

[32] G. Zachmann, "Rapid Collision Detection by Dynamically Aligned DOP-Trees," *Proceedings of IEEE Virtual Reality Annual International Symposium (VRAIS '98),* Atlanta, Georgia, March 1998.

[33] P. Jiménez, F. Thomas, C. Torras. 3D Collision Detection: a Survey. *Computers & Graphics*, 25(2):269–285, 2001.

[34] Lin MC, Gottschalk S. Collision detection between geometric models: a survey. IMA Conference on Mathematics of Surfaces, San Diego, CA, vol. 1, May 1998. p. 602}8.

[35] Held M, Klosowski JT, Mitchell J. Evaluation of collision detection methods for virtual reality fly-throughs. Proceedings of the Seventh Canadian Conference on Computer Geometry, vol. 3, 1995. p. 205-10.

[36] Herman M. Fast, three-dimensional, collision-free motion planning. Proceedings of the IEEE International Conference on Robotics and Automation, vol. 2, April 1986. p. 1056-63.

[37] Culley RK, Kempf KG. A collision detection algorithm based on velocity and distance bounds. Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, vol. 2, April 1986. p. 1064-9.

[38] Gilbert EG, Hong SM. A new algorithm for detecting the collision of moving objects. Proceedings of the IEEE International Conference on Robotics and Automation, Scottsdale, AR, vol. 1, May 1989. p. 8-14.

[39] Gilbert EG, Foo C-P. Computing the distance between general convex objects in three-dimensional space. IEEE Transactions on Robotics and Automation 1990;6(1):53-61.

[40] Moore M, Wilhelms J. Collision detection and response for computer animation. ACM Computer Graphics 1988;22(4):289-98.

[41] SchoK mer E, Thiel C. E$cient collision detection for moving polyhedra. Proceedings of the Eleventh Annual Symposium on Computational Geometry, 1995. p. 51-60.

[42] Hamlin GJ, Kelley RB, Tornero J. E$cient distance calculation using the spherically-extended polytope (s-tope) model. Proceedings of the IEEE International Conference on Robotics and Automation, Nice, France, vol. 3, May 1992. p. 2502-7.

[43] Meyer W. Distance between boxes: applications to collision detection and clipping. In Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, vol. 1, April 1986. p. 597-602.

[44] Bandi S, Thalmann D. An adaptive spatial subdivision of the object space for fast collision detection of animating rigid bodies. Eurographics'95, Maastricht, August 1995. p. 259-70.

[45] Palmer IJ, Grimsdale RL. Collision detection for animation using sphere-trees. Computer Graphics Forum 1995;14(2):105-16.

[46] Hubbard PM. Real-time collision detection and time-critical computing. Proceedings of the First ACM Workshop on Simulation and Interaction in Virtual Environments, vol. 1, 1995. p. 92-6.

[47] Ponamgi MK, Manocha D, Lin MC. Incremental algorithms for collision detection between solid models. Proceedings of ACM/Siggraph Symposium on Solid Modelling, vol. 1, 1995. p. 293-304.

[48] Gregory A, Lin MC, Gottschalk S, Taylor R. Fast and accurate collision detection for haptic interaction using a three degree-of-freedom force feedback device. Computational Geometry: Theory and Applications, to appear.

[49] Browne, D., Totterdell, P., and Norman, M., Eds. *Adaptive User Interfaces*. Academic Press, London, 1990.

[50] Holland, J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[51] Maes, P. Learning interface agents. In Proceedings of the 1994 Friend21 International Symposium on Next Generation Human Interface (February 1994).

[52] Miyashita, K., Matsuoka, S., Takahashi, S., and Yonezawa, A. Interactive generation of graphical user interfaces by multiple visual examples. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '94)* (November 1994), ACM Press.

[53] Schneider-Hufschmidt, M., Ed. *Adaptive User Interface*. North-Holland, Amsterdam, 1993.

[54] R. F. Cohen, P. Eades, T. Lin, and F. Ruskey. Three–dimensional graph drawing. In *Proceedings of Graph Drawing'94*, volume 894 of *LNCS*, pages 1–11. Springer, 1994.

[55] Arne K. Frick, Heiko Mehldau, and Andreas Ludwig. A fast adaptive layout algorithm for undirected graphs. In *Proceedings of Graph Drawing'94*, volume 894 of *LNCS*, pages 388–403. Springer, 1994.

[56] J.G. Hollands, T.T. Carey, M.L. Matthews, and C.A. McCann. Presenting a graphical network: A comparison of performance using fisheye and scrolling views. In *Proceedings of the 3rd International Conference on Human-Computer Interaction*, pages 313–320, September 1989.

[57] Cathleen McGrath, Jim Blythe, and David Krackhardt. The effect of graph layout on inference from social network data. In *Proceedings of GD'95*, 1995.

[58] Helen C. Purchase, Robert F. Cohen, and Murray I. James. Validating graph drawing æsthetics. In *Proceedings of GD'95*, 1995.

[59] George G. Robertson, J.D. Mackinlay, and S.K. Card. Cone trees: Animated 3-D visualizations of hierarchical information. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1991.

[60] B. Shneiderman, S.-K. Card, J.-D. MacKinlay., Readings in Information Visualization, Using Vision to Think, Morgan Kaufmann Publishers, New York, 1999.

[61] Sowizral H, Rushforth K, Deering M. The Java 3D API Specification. Addison-Wesley 2000.

[62] Java 3D API Specification, Version 1.1.2, June 1999, http://java.sun.com/products/javamedia/3D/forDevelopers/j3dguide/j3dTOC.doc.html

[63] "Star Trek Technology for Java3D - An Extendable Particle System for Java3D", Mike Jacobs, JDJ 2005 http://jdj.sys-con.com/read/99792.htm

[64] "A Java 3D visualization application for research use in astronomy", Teresa Stritch, Jason Best, Journal of Computing Sciences in Colleges, Volume 21, Issue 2 (December 2005) http://portal.acm.org/citation.cfm?id=1089053.1089061#references

[65] "Jabiru: harnessing Java 3D behaviors for device and display portability", Stromer, J.N. Quon, G.T. Gordon, P.M.K. Turinsky, A.L. Sensen, C.W. Calgary Univ., Alta., Canada; Computer Graphics and Applications, IEEE Publication Date: March-April 2005 Volume: 25, Issue: 2. On page(s): 70- 80

[66] "Building virtual anatomic models using Java3D (2004)", Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry http://portal.acm.org/citation.cfm?id=1044588.1044676

[67] "Brain Visualization in Java3D", Best Posters - Medinfo 2004 - Gold Medal Eider Moore, A. Poliakov, and J.F. Brinkley, University of Washington, Seattle, WA, USA http://sig.biostr.washington.edu/projects/brain/http://sigpubs.biostr.washington.edu/archive/00000151/

[68] "Web-Based Visualization Of Three-Dimensional Geospatial Data Using Java3D", Gobe Hobona, Phil James, David Fairbairn, IEEE Computer Graphics and Applications, 26(4): 28 - 33, 2006

[69] "Teaching Introductory Computer Graphics Using Java 3D, Games, and Customized Software: A Brazilian Experience", Romero Tori, Joao Luiz Bernardes, Jr., Ricardo Nakamura, ACM SIGGRAPH 2006 Educators program.

[70] Farrand, P.; Hussain, F.; Hennessy, E. (2002). "The efficacy of the mind map study technique". Medical Education 36 (5): 426–431. doi:10.1046/j.1365-2923.2002.01205.x. PMID 12028392. Retrieved 2009-02-16.

[71] Beel, Jöran; Gipp, Bela; Stiller, Jan-Olaf (2009). "Information Retrieval On Mind Maps - What Could It Be Good For?". Proceedings of the 5th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom'09). Washington: IEEE

[72] Bogdan Alexe, Laura Chiticariu, Renée J. Miller, Wang Chiew Tan: Muse: Mapping Understanding and deSign by Example. ICDE 2008: 10-19

[73] Khalid Belhajjame, Norman W. Paton, Suzanne M. Embury, Alvaro A. A. Fernandes, Cornelia Hedeler: Feedback-Based Annotation, Selection and Refinement of Schema Mappings for Dataspaces. EDBT 2010: 573-584

[74] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, Lucian Popa: Data exchange: semantics and query answering. Theor. Comput. Sci. 336(1): 89-124 (2005)

[75] Maurizio Lenzerini: Data Integration: A Theoretical Perspective. PODS 2002: 233-246

[76] Lassila, O., Swick, R.R.: Resource Description Framework (RDF) Model and Syntax Specification. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/ (1999)

[77] Klyne, G., Carroll, J.: Resource Description Framework (RDF) Concepts and Abstract Syntax. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/ (2004)

[78] Papamanthou, Ch., Tollis, I.G., & Doerr, M. (2005). 3D Visualization of Semantic Metadata Models and Ontologies. ***Graph Drawing. 12 International Symposium, GD 2004*****. 29 September 2004 through 2 October 2004, New York, USA. (pp. 377-388). Springer**