Γραφικά & Οπτικοποίηση

Κεφάλαιο 9

Διαχείριση Σκηνής

Εισαγωγή

- Διαχείριση Σκηνής:
  - Τα στοιχειώδη αντικείμενα της σκηνής συγκεντρώνονται σε χωρικά συνεπείς συστάδες
  - Οι συστάδες μπορούν να ομαδοποιηθούν σε μεγαλύτερες χωρικές ομάδες
- Γιατί είναι χρήσιμη η διαχείριση σκηνής:
  - Όλα τα στοιχειώδη αντικείμενα τακτοποιούνται ιεραρχικά & προσπελαύνονται αποδοτικά
  - Περικόπτονται νωρίς (σε υψηλό ιεραρχικό επίπεδο) από διαδικασίες όπως η περικοπή στο οπτικό πεδίο
  - Διευκολύνεται η διαχείριση των δεδομένων (δυναμικό φόρτωμα, λειτουργίες κρυφής μνήμης, κ.τ.λ.)
- Κατά τη σχεδίαση εικονικών κόσμων σκεφτόμαστε με οντολογικούς όρους & ομαδοποιούμε τις οντότητες σύμφωνα με κάποιες λογικές σχέσεις αλλά:
  - Μπορούμε εναλλακτικά, να οργανώσουμε τα δεδομένα με χωρικά συνεπή τρόπο (*διαμέριση χώρου*)

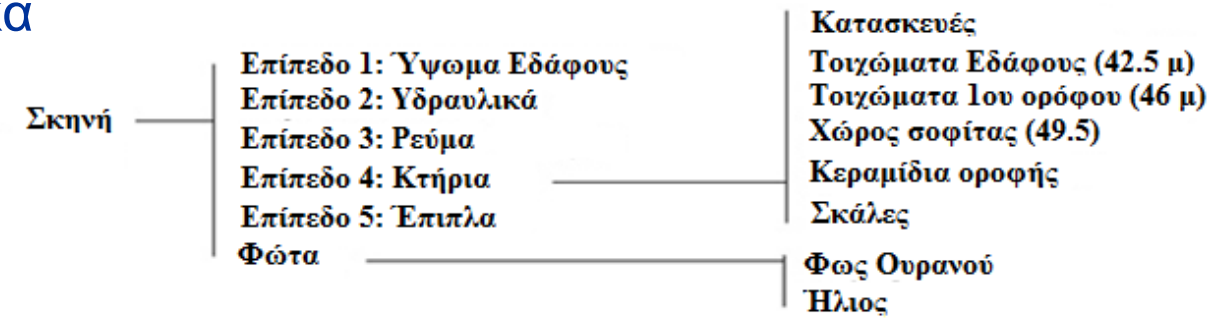
Εισαγωγή (2)

- Η διάσπαση της σκηνής σε χωρικά συνεχείς ιεραρχίες προσφέρει μεγάλη αποδοτικότητα
- Οι οντολογικές ιεραρχίες δε συμβαδίζουν απαραίτητα με τη χωρική διάταξη των αντικειμένων → δε βοηθάνε τη διαμέριση χώρου
- Η κατασκευή & ομαδοποίηση ενός ιεραρχικού κόσμου προσφέρει καλύτερη διαχείριση της σκηνής και λιγότερη σπατάλη μνήμης
- Η πολύ ενδελεχής αποσύνθεση της σκηνής σε ιεραρχικά τμήματα μπορεί να έχει αρνητικά αποτελέσματα:
  - Η εφαρμογή ξοδεύει χρόνο στην διάσχιση της ιεραρχίας της σκηνής
  - Στα γραφικά που επιταχύνονται από το υλικό του υπολογιστή, η διαμέριση μπορεί να οδηγήσει σε κακές γεωμετρίες & συχνές αλλαγές κατάστασης των χαρακτηριστικών σχεδίασης

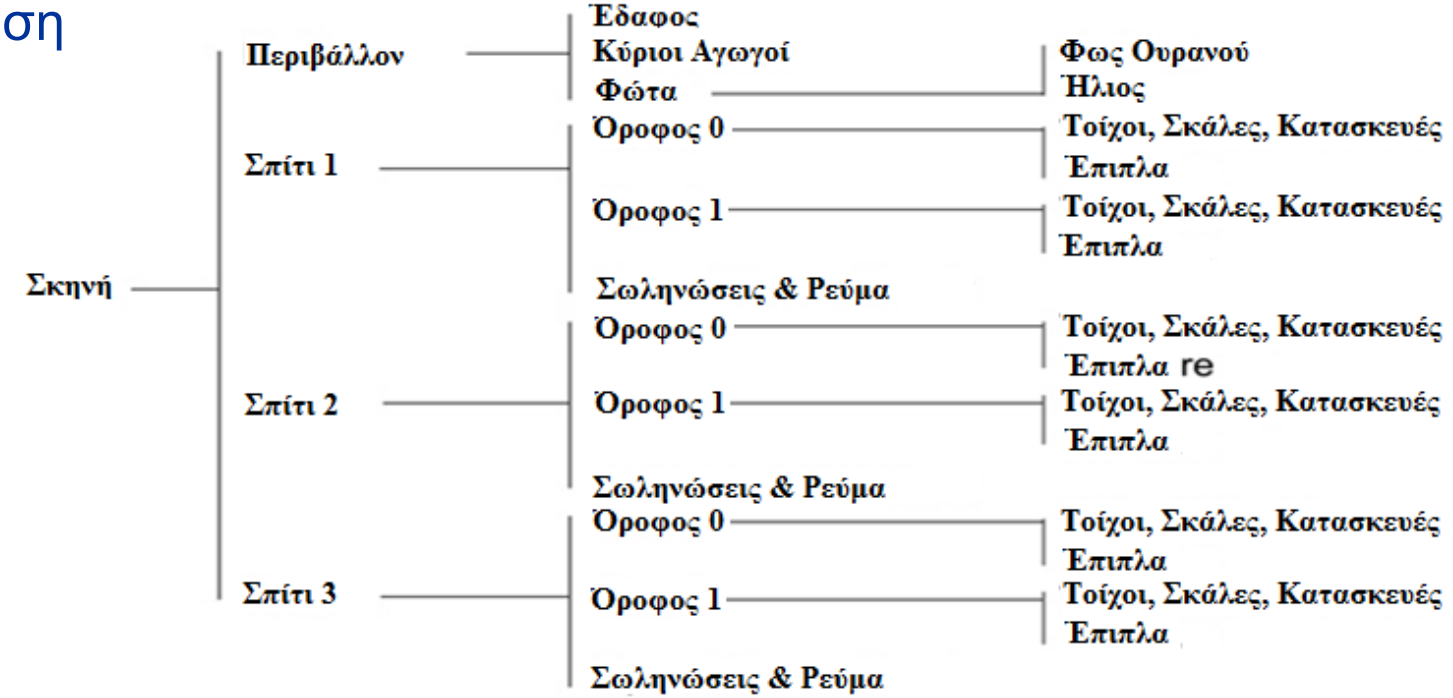
Παραδείγματα

Παραδείγματα (2)

Οντολογική διαμέριση με βάση τα υλικά



Χωρική διαμέριση



Παραδείγματα (3)

- Τα εσωτερικά περιβάλλοντα δημιουργούνται με περικοπή πυλών και δέντρα BSP
- Οι εξωτερικές σκηνές δημιουργούν ιεραρχίες με μεγάλο παράγοντα διακλάδωσης για αποδοτική περικοπή στο οπτικό πεδίο

Γράφοι Σκηνής

- *Γράφος Σκηνής (ΓΣ)*: Ιεραρχία γεωμετρικών στοιχείων συσχετιζόμενα με οντολογικό τρόπο και χωρική (γεωμετρική) εξάρτηση
- Αποτελείται από κόμβους που αναπαριστούν:
  - Γεωμετρικά στοιχεία (στατικά ή κινούμενα)
  - Συναθροίσεις κόμβων
  - Μετασχηματισμούς
  - Εξαρτώμενες επιλογές
  - Άλλες οντότητες προς αναπαραγωγή (π.χ. ήχους)
  - Κόμβους εργασιών προσομοίωσης
  - Γράφους άλλων σκηνών
- Οι ΓΣ είναι κατευθυνόμενοι, μη κυκλικοί γράφοι κόμβων, των οποίων οι ακμές ορίζουν γεωμετρικές ή λειτουργικές εξαρτήσεις ενός κόμβου παιδιού με τον πατρικό του
- Οι κόμβοι περιέχουν όλη την λειτουργικότητα που χρειάζεται για να ορισθεί μια συμπεριφορά → η λειτουργικότητα είναι ‘προσκολλημένη’ σε ένα αντικείμενο (βλ. αντικειμενοστραφή υλοποίηση)

Γράφοι Σκηνής (2)

- Ο κόμβος ρίζα είναι μια αφηρημένη αναπαράσταση της σκηνής:
  - Δίδει ένα σημείο έναρξης διάσχισης
  - Διαδίδει στην ιεραρχία όλες τις λειτουργίες που πρέπει να εφαρμοστούν στα στοιχεία της
- Μια λειτουργία σε ένα κόμβο επηρεάζει όλα τα παιδιά του
- Οι ομάδες (συναθροίσεις κόμβων) αντιμετωπίζονται σαν αυτοτελείς υπο-γράφοι σκηνής:
  - Κάνει εύκολη την μοντελοποίηση περίπλοκων περιβαλλόντων και των κινήσεων εντός αυτών
  - Επαναχρησιμοποιήσιμες οντότητες
- Περίπλοκες κινήσεις και συμπεριφορές χωρίζονται σε απλούστερες δίνοντας τοπική συμπεριφορά στα στοιχεία που οργανώνονται ιεραρχικά

Παράδειγμα

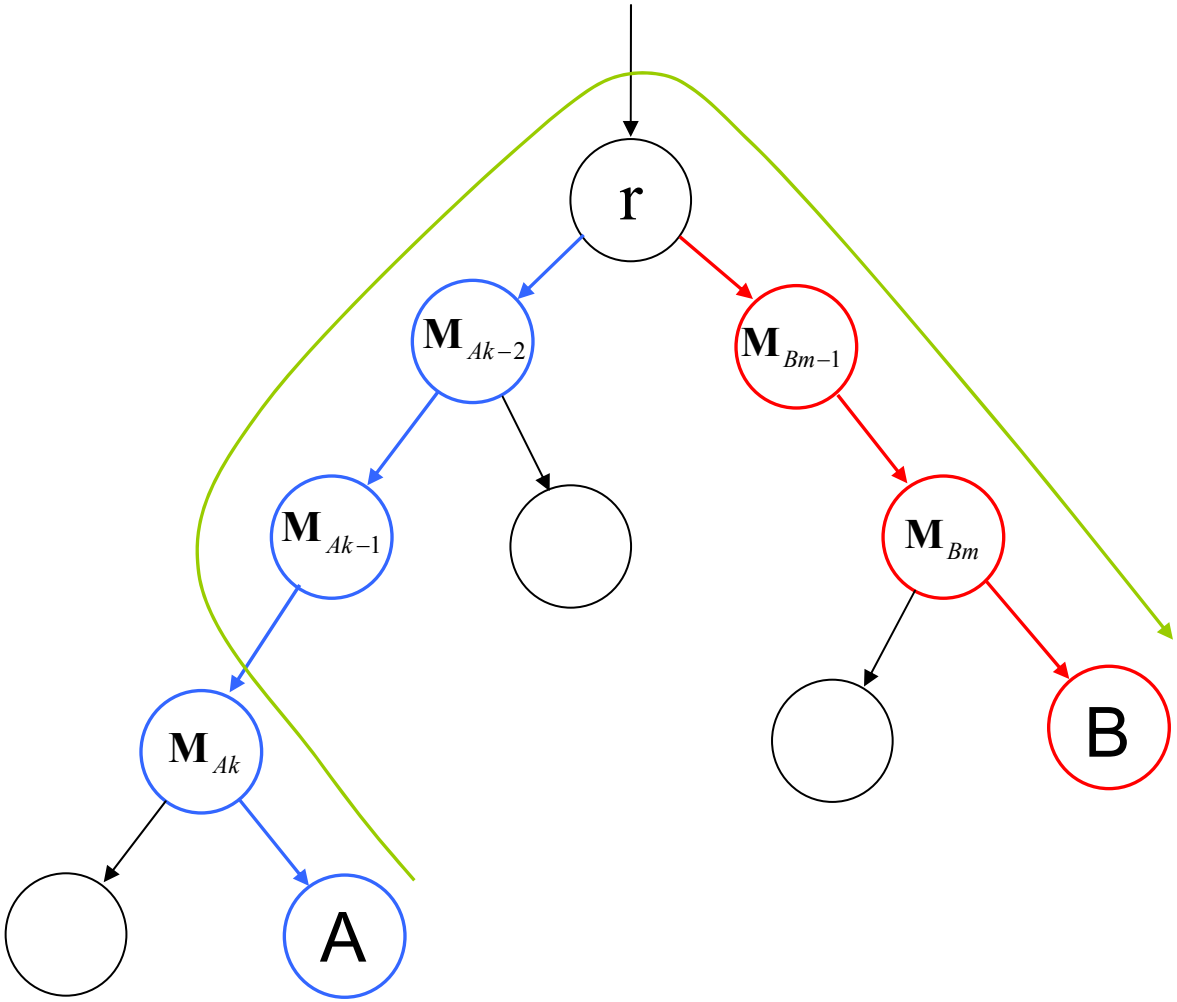
- Η κίνηση κάθε μηχανικού μέρους ενός σκάφους που σχετίζεται με το ΠΣΣ είναι δύσκολο να παραχθεί άμεσα
- Πχ, Ποια είναι η φαινομενική κίνηση της προπέλας του σκάφους σε σχέση με το άτομο στο λιμάνι και ποια σε σχέση με τον οδηγό?

Σχέσεις Κόμβων

- Εν γένει, εκφράζουμε ένα κόμβο γράφου σκηνής (στόχος) που σχετίζεται με έναν άλλο (κόμβος αναφοράς) με μια αλλαγή του συστήματος συντεταγμένων αναφοράς σύμφωνα με τους ενδιάμεσους μετασχηματισμούς:
  - Εφαρμογή μιας προς τα πάνω διάσχισης του δέντρου από τον στόχο προς τον κοινό γονικό κόμβο του στόχου και του κόμβου αναφοράς
  - Διάσχιση προς τα κάτω στον κόμβο αναφοράς με εφαρμογή των αντίστροφων μετασχηματισμών του μονοπατιού
- Ο μετασχηματισμός ενός κόμβου στο επίπεδο *k* σε ένα κλαδί *A* που σχετίζεται με έναν άλλο κόμβο που βρίσκεται στο επίπεδο *m* σε ένα κλαδί *B* με κοινή ρίζα στο επίπεδο *r* είναι:

$$\mathbf{M}_{A\to B} = \prod_{j=m}^{r+1} \mathbf{M}_{Bj}^{-1} \prod_{i=r+1}^{k} \mathbf{M}_{Ai}$$

Σχέσεις Κόμβων (2)



Οργάνωση Δεδομένων

- Οι πλειοψηφία των γράφων σκηνής περιγράφει σχέσεις μεταξύ:
  - Οντοτήτων
  - Συναθροίσεων κόμβων
- Οι κόμβοι γεωμετρίας μπορούν να περιέχουν:
  - Δεδομένα
  - Πληροφορίες δεδομένων στοιχειωδών αντικειμένων (Επιφάνειες NURBS, δεδομένα όγκου)
- Τα δεδομένα των κόμβων μπορεί να είναι:
  - Αταξινόμητα (ακατέργαστα)
  - Ευρετηριοποιημένα

Υβριδική Οργάνωση

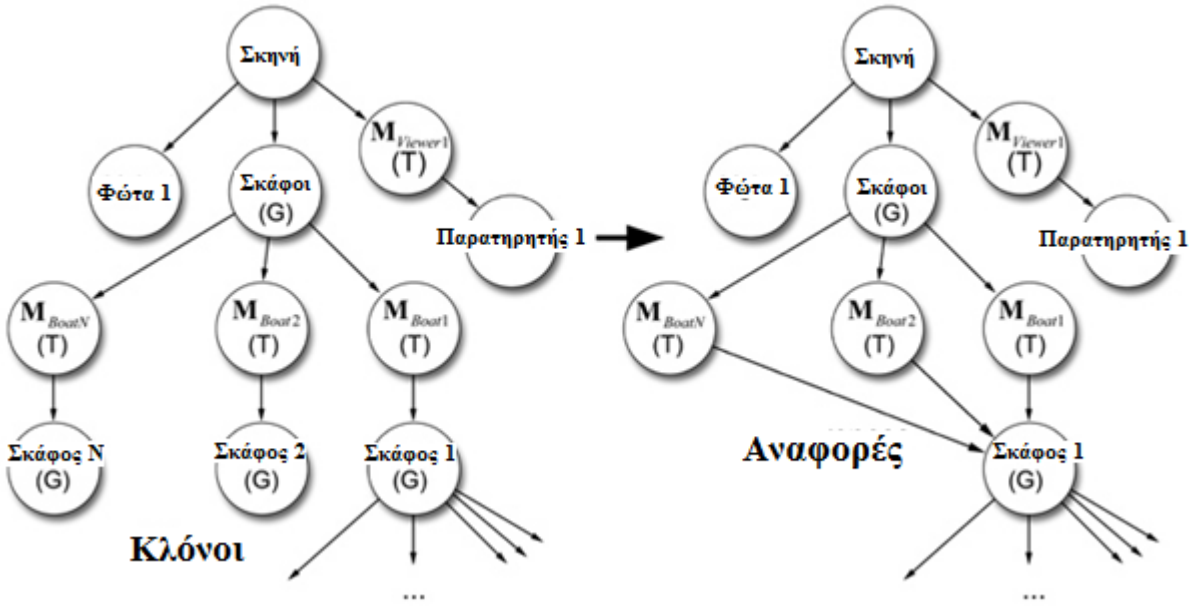
- Οι κόμβοι γεωμετρίας μπορεί να είναι κομμάτι δομών χωρικής διαμέρισης (Δέντρα AABB)
- Αυτή η δομή χρησιμοποιείται συχνά σε εφαρμογές με μεγάλα σύνολα δεδομένων (σχεδίαση σε μη πραγματικό χρόνο)
- Γράφος Σκηνής:
  - Αναπαριστά την οντολογική οργάνωση των δεδομένων, αλλά:
  - Η γεωμετρία συνυπάρχει δηλωμένη και στα φύλλα του γράφου και σε ένα μηχανισμό χωρικής διαμέρισης
- Για την ενημέρωση των χαρακτηριστικών των αντικειμένων και τους μετασχηματισμούς τους χρησιμοποιείται ο γράφος σκηνής
- Για τη σχεδίαση της γεωμετρίας χρησιμοποιείται η προσπέλαση μέσω του συστήματος χωρικής διαμέρισης (βλ. κεφάλαιο Παρακολούθησης ακτινών)

Υβριδική Οργάνωση – Πραγμ. Χρόνος

- Ένας γράφος σκηνής χρησιμοποιείται μόνο για κινούμενα αντικείμενα
- Όλη η πληροφορία του στατικού περιβάλλοντος απομονώνεται σε ένα σύστημα χωρικής διαμέρισης υψηλής απόδοσης:
  - Πχ. Δέντρο BSP

Δημιουργία Στιγμιότυπου Κόμβου

- Γιατί πρέπει να διπλασιαστεί ένας κόμβος για την δημιουργία αντιγράφων της ίδιας οντότητας εφόσον αυτά είναι ταυτόσημα με την αρχική;
- Δημιουργείται μια αναφορά στον αρχικό κόμβο, όποτε πρέπει να εισαχθεί στον γράφο ένας ταυτόσημος κόμβος
- Στο νέο κόμβο δεν επισυνάπτεται αντίγραφο των δεδομένων
- Όταν υπάρχει δημιουργία στιγμιότυπου, η δενδρική δομή του γράφου σκηνής μετασχηματίζεται σε ένα κατευθυνόμενο κυκλικό γράφο



Δημιουργία Στιγμιότυπου Κόμβου (2)

- Η δημιουργία στιγμιότυπων κόμβων αντί αντιγράφων:
  - Εξοικονομεί χώρο αποθήκευσης
  - Επιταχύνει τους υπολογισμούς για συγκεκριμένα βήματα προσομοίωσης
  - Η επεξεργασία γίνεται μια φορά για τον αρχικό κόμβο & έπειτα όλα τα στιγμιότυπα επανά-χρησιμοποιούν τα νέα δεδομένα
- Η σειρά προσπέλασης δεν είναι σημαντική:
  - Όταν τα δεδομένα ενός κόμβο προσπελαύνονται πρώτη φορά στο καρέ k+1, ο κόμβος σημειώνεται σαν "επεξεργασμένος" για το τρέχον χρονικό διάστημα
  - Μεταγενέστερες επισκέψεις στον κόμβο (απευθείας ή μέσω αναφοράς) ελέγχουν τον μετρητή καρέ και παραλείπουν τους τοπικούς υπολογισμούς

Διάσχιση Γράφου Σκηνής

- Κύριο πλεονέκτημα αναπαράστασης γράφου σκηνής ενός 3Δ κόσμου:
  - Κάθε διαδικασία εφαρμόζεται στον κόμβο ρίζα της ιεραρχίας & διαδίδεται στις υπόλοιπες οντότητες μέσω της διάσχισης του γράφου σκηνής
- 4 κύριες διαδικασίες εφαρμόζονται σε ένα γράφο σκηνής:
  - ***Αρχικοποίηση***
  - ***Προσομοίωση***
  - ***Περικοπή***
  - ***Σχεδίαση***
- Κάθε διαδικασία εφαρμόζεται ιεραρχικά στους κόμβους

Διαδικασία Προσομοίωσης (app)

- Καθορίζει όλες τις εσωτερικές παραμέτρους των κόμβων για το k καρέ
- Ενημερώνει τις μεταβλητές σύμφωνα με τη συμπεριφορά του κόμβου
- Κάνει οποιεσδήποτε αλλαγές στα δεδομένα του κόμβου είναι απαραίτητες για τη μετάβαση από την κατάσταση του καρέ k-1 σε αυτή του καρέ k

Περικοπή (cull)

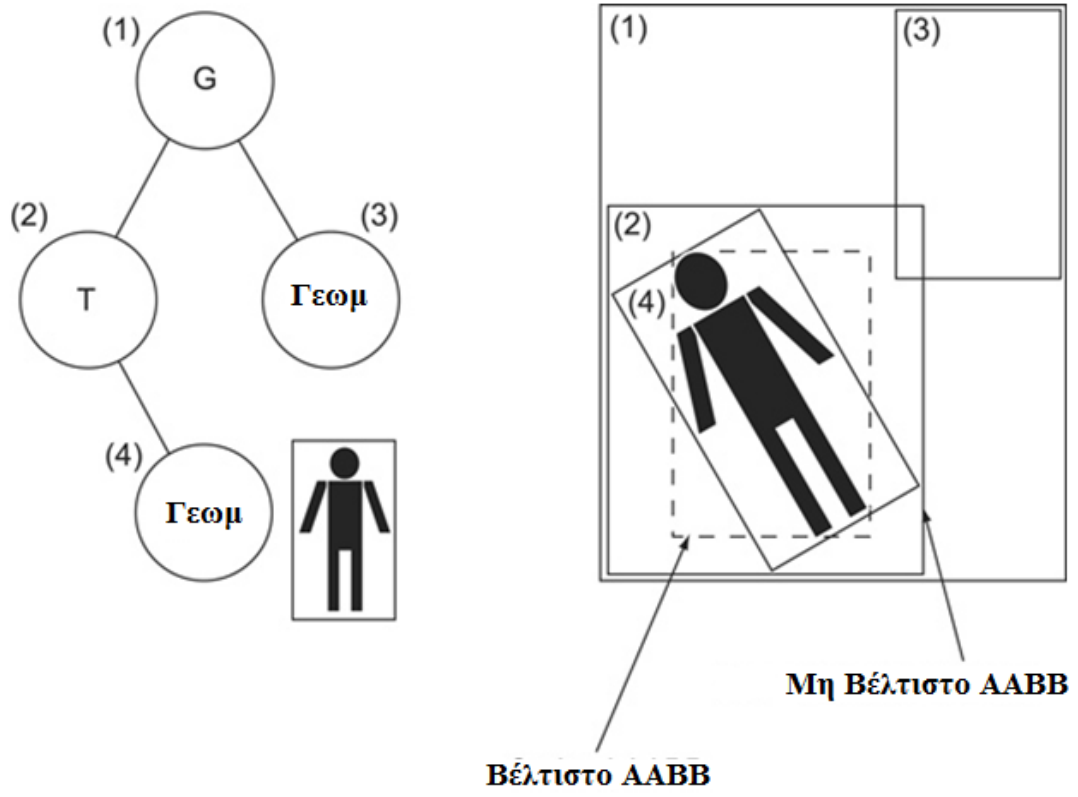
- Η **περικοπή** σε ένα γράφο σκηνής συνδέεται με τις εξαρτήσεις κόμβων
  - Κάθε κόμβος "περιέχει" όλα τα παιδιά του:
    - Αν ο κόμβος ρίζα ενός υποδέντρου θεωρηθεί "αφανής" → κάθε κόμβος παιδί του είναι επίσης αφανής → ολόκληρο το υποδέντρο περικόπτεται
  - Αν ένας συναθροιστικός κόμβος είναι ορατός → τα παιδιά του ελέγχονται ξεχωριστά → σε περίπτωση μερικής παρεμπόδισης του κόμβου γονέα, κάποιοι κόμβοι παιδιά μπορεί να είναι ορατοί
  - Το αποτέλεσμα τις διαδικασίας περικοπής καθορίζεται από τον έλεγχο του περιβάλλοντα όγκου του κόμβου που έχει επιλεχθεί
  - Για έναν συναθροιστικό κόμβο, ο περιβάλλον όγκος αντικατοπτρίζει τις συγκεντρωτικές επεκτάσεις των παιδιών του → **Ιεραρχίες περιβαλλόντων όγκων**

Ιεραρχίες Περιβαλλόντων Όγκων

- Αν ένας κόμβος περιέχει ένα ιεραρχικό παρακλάδι κόμβων που εκφράζουν κίνηση, τα όρια του περιβάλλοντος όγκου του πρέπει να διευθετούνται δυναμικά κάθε φορά που αλλάζουν τα όρια κάποιου από τα παιδιά του
- Ο επανυπολογισμός των ορίων των κόμβων γεωμετρίας απαιτεί τουλάχιστο την επαναληπτική καταγραφή της μέγιστης και ελάχιστης συντεταγμένης
- Για τα υποδέντρα κόμβων κίνησης που πρέπει να υπάρξει επαναληπτική επεξεργασία στα ακατέργαστα δεδομένα τους ούτως ή άλλως (πχ. Σκελετική Κίνηση), αυτό δεν επιβάλλει πρόσθετη δουλειά
- Για την κίνηση στερεών σωμάτων, ο επανυπολογισμός σπαταλά αρκετό χρόνο και μπορεί να είναι απαγορευτικός για μεγάλα μοντέλα

Ιεραρχίες Περιβαλλόντων Όγκων (2)

- Μια τεχνική που υιοθετείται όταν η ταχύτητα είναι πιο σημαντικός παράγοντας από την ακριβή ορατότητα: Προσαρμογή του περιβάλλοντα όγκου ενός συναθροιστικού κόμβου με βάση:
  - Τα μετασχηματισμένα όρια των περιβαλλώντων όγκων των παιδιών του
  - Αντί για τα όρια των μετασχηματισμένων παιδιών του



Ιεραρχίες Περιβαλλόντων Όγκων (3)

- Η παραπάνω λύση δεν παράγει βέλτιστους περιβάλλοντες όγκους
  - Τα όρια των μετασχηματισμένων περιβαλλόντων όγκων είναι, εν γένει, μεγαλύτερα από τα όρια της περικλειόμενης γεωμετρίας
- Η χωρική υποδιαίρεση (αν υπάρχει), μπορεί σε μεγάλο βαθμό να χειριστεί την περικοπή

Σχεδίαση Γράφου Σκηνής (draw)

- Η διαδικασία **σχεδίασης** :
  - Αναδρομικά κατέρχεται την ιεραρχία και εφαρμόζει αλγορίθμους σχεδίασης εικόνας σε κάθε ορατό κόμβου
  - Στην απευθείας σχεδίαση εικόνας και τον αλγόριθμο εκπομπής ακτινών $1^{ου}$ επιπέδου τα υποδέντρα που έχουν περικοπεί αγνοούνται

Προγραμματίζοντας με Γράφους Σκηνής

- Σε ένα γράφο σκηνής, οι περισσότεροι κόμβοι είναι αυτό-διαχειριζόμενοι → παρέχουν οι ίδιοι τις λειτουργίες σχεδίασης και τη συμπεριφορά τους
- Ένας συναθροιστικός κόμβος (που δε σχεδιάζεται) πρέπει να μεταβιβάζει στους απογόνους του λειτουργίες όπως η σχεδίαση και η περικοπή
- Η κατανεμημένη λειτουργία ενός γράφου σκηνής:
  - Διευκολύνει μια αντικειμενοστραφή σχεδίαση των κόμβων
  - Επωφελείται από τον πολυμορφισμό και την αφαιρετικότητα

Προγραμματίζοντας με Γράφους Σκηνής (2)

- Όλοι οι κόμβοι ενός γράφου σκηνής προέρχονται από μια κλάση κόμβων & μεταβάλλουν τη συμπεριφορά τους καθώς ένας κόμβος κληρονομεί κοινά χαρακτηριστικά και εξειδικεύει τη συμπεριφορά του:

```
class Node
{
    protected:
        bool active;
        bool culled;
    public:
        Node();
        virtual void init();
        virtual void simulate();
        virtual void cull();
        virtual void draw();
        virtual void reset();
};
```

Προγραμματίζοντας με Γράφους Σκηνής (3)

```
class Group : Node
{
    protected:
        vector<Node*> children;
        Bvolume extents;
    public:
        Group();
        void add(Node *n);
        void remove(int i);
        Node * getChild(int i);
        int  getNumChildren();
        virtual void init();
        virtual void simulate();
       virtual void cull();
       virtual void draw();
};
```

Προγραμματίζοντας με Γράφους Σκηνής (4)

- Οι κόμβοι που είναι επέκταση της κλάσης `Node` κληρονομούν τις 4 βασικές λειτουργίες
- Οι συναθροιστικοί κόμβοι παράγονται από την κλάση `Group` & μοιράζονται ένα κοινό τρόπο λειτουργίας:
  - Διατηρούν μια λίστα απογόνων και διαθέτουν βασικές λειτουργίες μιας συλλογής
- Πιο περίπλοκες υποκλάσεις `Group` επεκτείνουν τη συμπεριφορά με την πρόσθεση ειδικών μεθόδων
- Κοινό χαρακτηριστικό όλων των κόμβων `Group` είναι η διάσχιση των απογόνων τους:
  - Εκδηλώνεται ως ένα αναδρομικό κάλεσμα όλων των αντικειμένων `Node` στην εσωτερική τους λίστα
- Λόγω του πολυμορφισμού, ένα αντικείμενο `Group` μπορεί καλέσει τις μεθόδους `init()`, `draw()`, `cull()`, `simulate()` χωρίς να ξέρει από ποιες υποκλάσεις προέρχεται το αντικείμενο

Προγραμματίζοντας με Γράφους Σκηνής (5)

- Όλοι οι κόμβοι του γράφου σκηνής μοιράζονται ένα κοινό interface:

```
void Group::draw()
{
    vector <Node>::size_type i, sz;
    sz = children.size();
    for (i=0; i<sz; i++)
        children[i] -> draw();
}
void Geometry::draw() // Geometry: υποκλάση του Node
{
    if (!enabled || culled)
        return;
    // ... Σχεδίαση της γεωμετρίας
}
```

- Η περικοπή, αρχικοποίηση και προσομοίωση ορίζονται με όμοιο τρόπο

Προγραμματίζοντας με Γράφους Σκηνής (6)

- Ο τρόπος διάσχισης ενός γράφου σκηνής ορίζει τη σειρά εκτέλεσης κάθε μεθόδου
- Η αρχικοποίηση και τα 3 επαναλαμβανόμενα βήματα εκτελούνται ιεραρχικά το ένα μετά το άλλο ξεκινώντας από τη ρίζα της σκηνής
- Μετά την αρχικοποίηση, οι αρμοδιότητες ενός γράφου σκηνής περιλαμβάνουν:
  - Την εκτέλεση της προσομοίωσης
  - Την περικοπή
  - Τις μεθόδους σχεδίασης (draw) του κόμβου ρίζας
- Οπότε, γίνονται 3 διασχίσεις πριν οπτικοποιηθεί ο γράφος σκηνής

Προγραμματίζοντας με Γράφους Σκηνής (7)

```
//Η Scene είναι μια υποκλάση της Group
Scene *myScene = new Scene();
myScene -> load("village.scn");
myScene -> init();
while (notTerminating)
{
    //... Άλλες λειτουργίες
    myScene -> simulate() ;
    myScene -> cull() ;
    myScene -> draw() ;
}
```

- Έτσι οι κόμβοι συγχρονίζονται σε σχέση με την κατάστασή τους και τις τιμές των παραμέτρων

Ενημέρωση Δεδομένων με Υστέρηση

- Σε αυτή τη μέθοδο, οι αλλαγές στην κατάσταση μιας οντότητας δεν παράγουν άμεσο αποτέλεσμα:
  - Μια φορά γίνεται ο υπολογισμός τιμής για όλα τα χαρακτηριστικά για την παραγωγή μιας προσομοίωσης, περικοπής, ή οπτικού αποτελέσματος
- Οι ενημερώσεις με υστέρηση είναι χρήσιμες όταν πρέπει να γίνεται μια υπολογιστικά ακριβή διαδικασία κάθε φορά που αλλάζει μια μεταβλητή
- Παραδείγματα σε ένα γράφο σκηνής:
  - Συνθετική κίνηση βασισμένη σε μηχανική προσομοίωση
  - Υπολογισμός πολυέδρου σκιάς

Μεταφορά Μηνυμάτων

- Σύνθετοι γράφοι σκηνής→ δεν περιοριζόμαστε στον ιεραρχικό έλεγχο
- Άμεση επικοινωνία: οι κόμβοι μπορούν να επικοινωνήσουν μεταξύ τους
  - Με άμεσο κάλεσμα μεθόδων άλλων κόμβων
  - Με μηνύματα
- Στη δεύτερη περίπτωση (πιο εύκολη στο συγχρονισμό), κάθε κόμβος πρέπει να επεκταθεί για να υποστηρίζει μια ουρά μηνυμάτων και ένα χάρτη γεγονότων:

```
class NodeMessage
{
    Node *from, *to;
    int ID;
    void * params;
};
typedef EventID int;
```

Μεταφορά Μηνυμάτων (2)

```
class Node
{
    protected:
        ...
        vector<NodeMessage *> msgQueue;
        multimap<EventID,NodeMessage *> eventMap;
        // Επεξεργασία εισερχομένων μηνυμάτων
        void processMessages();
        // Αποστολή εξερχομένων μηνυμάτων
        void dispatchMessages();
    public:
        ...
        // Εισαγωγή μηνύματος στην ουρά του κόμβου
        message( NodeMessage *msg );
    registerEvent( EventID evt, Node* target,
                   int msgID, void* params );
};
```

Μεταφορά Μηνυμάτων (3)

- Απαραίτητη η ουρά μηνυμάτων→ ένας κόμβος μπορεί να λάβει πολλά μηνύματα εντολών από άγνωστο πλήθος κόμβων
- Χάρτης γεγονότων:
  - Δημιουργεί μια διεπαφή για αντιδράσεις που ορίζονται από το χρήστη στις αλλαγές κατάστασης του κόμβου

Μεταφορά Μηνυμάτων: Παράδειγμα

- Θεωρείστε ένα δωμάτιο γεμάτο έπιπλα.
  - Αρχικά το φως είναι σβηστό → δε χρειάζεται να έχουν τα έπιπλα σκιές → αρχικά απενεργοποιημένο για αυτούς του κόμβους γεωμετρίας
  - Όταν ανάβει το φως → τα έπιπλα πρέπει να αποκτήσουν σκιές
- Θέτουμε ένα αντικείμενο με όνομα halo ορατό γύρω από τη λάμπα για να είναι ρεαλιστική η σκηνή

```
Light * bulb; // Το Light επεκτείνει το Node
Geometry *furniture, *halo;
...
bulb -> registerEvent( EVENT_ON, furniture, MSG_SHADOWS_ON,
 NULL);
bulb -> registerEvent( EVENT_ON, halo, MSG_ENABLE, NULL);
```

Μεταφορά Μηνυμάτων (4)

- Στην επεκταμένη κλάση `Node` προστίθενται 2 νέες συναρτήσεις protected :
  - `processMessages()`
  - `dispatchMessages()`
- Αυτές οι λειτουργίες εκτελούνται πριν & μετά από το βήμα προσομοίωσης

Προ/Μετά Βήματα Επεξεργασίας Κόμβων

- Πρέπει να προσθέσουμε μεθόδους *προ/μετα*-προσομοίωσης → θα εκτελούνται μέσω ενός βήματος προ/μετά προσομοίωσης για όλη τη σκηνή :

```
void Scene::simulate()
{
    preSimulate();
    Group::simulate();
    postSimulate();
}
...
void Group::preSimulate()
{
    vector <Node>::size_type i,sz;     sz = children.size();
    for (i=0; i<sz; i++)
        children[i]->preSimulate();
}
```

Προ/Μετά Βήματα Επεξεργασίας Κόμβων (2)

- Όμοιες προ/μετα συναρτήσεις υλοποιούνται για τα στάδια της σχεδίασης και της περικοπής:
  - Είτε τοπικά για κάθε κόμβο (εκτελείται πριν & μετά από την αντίστοιχη λειτουργία κάθε κόμβου)
  - Ή καθολικά (σαν στάδια προ- και μετα-προσομοίωσης)

Παραδείγματα:

- Όταν σχεδιάζουμε με OpenGL, πρέπει να υλοποιηθεί ένας κόμβος *Transformation* :
  - Μια συνάρτηση προ-σχεδίασης για να τοποθετήσει το τρέχοντα πίνακα κατάστασης στη στοίβα
  - Μια συνάρτηση μετα-σχεδίασης για να αφαιρέσει το τρέχοντα πίνακα κατάστασης από τη στοίβα
- Μια καθολική συνάρτηση μετα-σχεδίασης μπορεί να προκαλέσει εναλλαγή καταχωρητών εικόνας

Κατανεμημένη Δημιουργία Σκηνής

- Απαιτητικές εφαρμογές ή πολλαπλών-χρηστών/πολλαπλών-οθονών χρειάζονται να κατανέμουν τα δεδομένα του γράφου σκηνής και να τα σχεδιάζουν σε πολλαπλούς επεξεργαστές
- Ένας επεξεργαστής μπορεί να είναι:
  - Ένας ηλεκτρονικό υπολογιστής
  - Ένας ειδικός συνεπεξεργαστής
  - Ένα ή παραπάνω συστήματα παράλληλων επεξεργαστών
- Η σκηνή δεν είναι απαραίτητο να βρίσκεται σε κοινό χώρο στη μνήμη

Κατανεμημένη Δημιουργία Σκηνής (2)

- Μια διαδικασία κατανεμημένης σχεδίασης μιας σκηνής χωρίζεται σε 3 κατηγορίες
  - Κατανομή στο πεδίο του τριδιάστατου χώρου
  - Κατανομή στο πεδίο του χρόνου
  - Κατανομή στο πεδίο της εικόνας
- Η διαδικασία σχεδίασης ενός καρέ μιας συνθετικής αναπαράστασης αποτελείται από 4 στάδια:
  - Διάσπαση (Splitting)
  - Κατανομή (Distribution)
  - Σχεδίαση (Rendering)
  - Σύνθεση (Compositing)

Κατανεμημένη Σχεδίαση με Μετα-Ταξινόμηση

- Κατανομή της σχεδίασης μιας σκηνής σε επεξεργαστές με χωρικό κατακερματισμό:
  - ένα μέρος της σκηνής μεταφέρεται σε κάθε επεξεργαστή
  - το κάθε μέρος σχεδιάζεται ανεξάρτητα
  - μετά συντίθεται ένα ενιαίο τελικό αποτέλεσμα
- Η σκηνή διαιρείται σύμφωνα με την ιεραρχία του γράφου σκηνής ή το μηχανισμό υποδιαίρεσης χώρου
  - τα τμήματα κατανέμονται στους διαθέσιμους επεξεργαστές
  - κάθε επεξεργαστής σχεδιάζει ένα τμηματικό αποτέλεσμα
  - μετά το αποτέλεσμα πρέπει να συνδυαστεί με τα υπόλοιπα αποτελέσματα
- Στην περίπτωση της άμεσης σχεδίασης:
  - Οι ενδιάμεσες εικόνες-αποτελέσματα δεν είναι ταξινομημένες ως προς το βάθος
  - Επικαλύπτονται στο χώρο της εικόνας
  - Οι προκύπτοντες καταχωρητές εικόνας δεν μπορούν να συνδυαστούν άμεσα
  - Συνήθης πρακτική είναι η διατήρηση και μετάδοση των καταχωρητών βάθους κάθε μέρους και η σχεδίαση με βάση το συγκριτικό βάθος

Κατανεμημένη Σχεδίαση με Μετα-Ταξινόμηση (2)

- Οι διαδικασίες κατανεμημένης σχεδίασης, ονομάζονται *μετα-ταξινόμηση* γιατί η απόφαση για το ποιο μέρος της τελικής εικόνας προκύπτει από ποιο κόμβο συμβαίνει στο τέλος της παραγωγής του καρέ

Κατανεμημένη Σχεδίαση με Προ-Ταξινόμηση

- Διαδικασίες *προ-ταξινόμησης*:
  - Εκτελούν μια προ-διαμέριση του πεδίου εξόδου (εικόνα)
  - Σε κάθε επεξεργαστής ανατίθεται ένα ή περισσότερα κομμάτια
- Η σύνθεση των κομματιών προς σχεδίαση είναι τετριμμένη σε αυτούς τους αλγορίθμους → τα κομμάτια εικόνας που μαζεύονται δεν έχουν επικαλύψεις
- Στην περίπτωση της έμμεσης (offline) σχεδίασης συνθετικής κίνησης :
  - Κάθε επεξεργαστής διατηρεί ένα πλήρες αντίγραφο της βάσης δεδομένων της σκηνής καθώς και εξωτερικά χαρακτηριστικά (υφές), και σχεδιάζει ανεξάρτητα μια ολοκληρωμένη εικόνα καρέ ή τμήμα της.
  - Τετριμμένα παράλληλα: όχι επικοινωνία εκτός από την αρχικοποίηση και τη συγκομιδή των τελικών τμημάτων εικόνας.

Κατανεμημένη Σχεδίαση με Προ-Ταξινόμηση(2)

- Οι στρατηγικές προ-ταξινόμησης στο χώρο εικόνας είναι πιο κοινές:
  - Η βάση δεδομένων της σκηνής αντιγράφεται στους επεξεργαστές, ή μοιράζεται από πολλαπλές διεργασίες που εκτελούν τη σχεδίαση
  - Σε κάθε επεξεργαστή ανατίθενται ένα ή περισσότερα "παράθυρα" της τελικής εικόνας
  - Τα αποτελέσματα συντίθενται με την αντιγραφή των τμημάτων εικόνας σε έναν κοινό καταχωρητή
  - Η άμεση κατανεμημένη σχεδίαση σε πολλαπλά συστήματα γραφικών στην ίδια μηχανή διαχειρίζεται από το υλικό των οθονών

Διαμέριση με Προ-Ταξινόμηση

- Οι στρατηγικές διαμέρισης στο χώρο εικόνας είναι σημαντικές και αποτελεσματικές ως προς το φόρτο εργασίας:
  - Συνήθεις μέθοδοι διαμέρισης:
    - Εναλλαγή γραμμών σάρωσης (interlaced scan-line)
    - Σε Ψηφίδες (Tiled)
    - Ολίσθηση ολόκληρης εικόνας (offset full-image – για παράλληλο antialiasing)