Graphics Export Manual

Μιχάλης Γεωργουλόπουλος Γιάννης Τσιομπίκας

9 Φεβρουαρίου 2007

1 Γενικά

Αυτό το manual περιγράφει τις λεπτομέρειες για την εξαγωγή σκηνών από το max στα format της Track7: *t7sce* και *t7anm*.

2 Βασικές έννοιες–Ορισμοί

2.1 Level

Στο context αυτού του manual ορίζουμε ως level το κομμάτι του game το οποίο φορτώνεται σαν μια αυτοτελής ενότητα. Μέριμνα πρέπει να λαμβάνεται έτσι ώστε όλα τα δεδομένα ενός level (*γεωμετρία, textures, animations*) να είναι καλά ζυγισμένα. Ούτε πολύ λίγα πράγματα σε ένα level (οπότε και γίνεται συχνό φαινόμενο το loading ενός νέου level), αλλά ούτε πάρα πολλά (οπότε τα δεδομένα είναι περισσότερα από όσα η μνήμη του συστήματος και της κάρτας γραφικών μπορούν να χωρέσουν). Ένας καλός εμπειρικός κανόνας είναι να μελετάμε το μέγεθος που έχουν τα levels σε τελευταία games και προσπαθούμε να κινούμαστε σε αυτά τα πλαίσια.

2.2 ObjectGroup

Ένα level περιλαμβάνει ένα πλήθος από αντικείμενα. Στο manual αυτό τα "αντικείμενα" αναφέρονται ως "ObjectGroups" ή "OG". Ένα ObjectGroup είναι μια συλλογή από meshes τα οποία μαζί συνθέτουν ένα αντικείμενο που θα κάνει μια συγκεκριμένη δουλειά στο level. Τα ObjectGroups μπορεί να είναι στατικά, ή να έχουν ένα ή περισσότερα animations. Μεγάλη προσοχή θα πρέπει να δοθεί στο σπάσιμο ενός level σε ObjectGroups. Στην ιδανική περίπτωση ένα level αποτελείται εξολοκλήρου από ένα OG. Βέβαια αυτό πρακτικά δεν γίνεται γιατί ένα level περιλαμβάνει πολλά αντικείμενα που κάνουν ανεξάρτητες κινήσεις. Το scripting system του game έχει ώς βασική μονάδα το OG. Για παράδειγμα το παρακάτω θα μπορούσε να είναι μέρος ενός script για ένα level:

```
IF andronikos pushed button THEN set porta2_animation = open.anm
```

Για να μπορέσει να συμβεί το παραπάνω θα πρέπει το porta2 να είναι ένα OG χωριστό από το υπόλοιπο level.

Ένας καλός κανόνας προκειμένου να αποφασίσουμε τί είναι ένα ObjectGroup είναι ο εξής: Ένα ObjectGroup είναι ένα ανεξάρτητα animated αντικείμενο μέσα στο level. Στην παραπάνω περίπτωση η πόρτα είναι ανεξάρτητα animated από το υπόλοιπο level γιατί το άνοιγμα και το κλείσιμό της δεν πρέπει να επηρεάζουν

το animation του υπόλοιπου level. Ας υποθέσουμε για παράδειγμα ότι η πόρτα βρίσκεται σε ένα δωμάτιο με γρανάζια που γυρνάνε και αποφασίσαμε ότι όλα μαζί πρέπει να γίνουν ένα OG. Σε αυτή την περίπτωση το OG θα έχει δύο animations:

1. Άνοιγμα πόρτας (μαζί με περιστροφή γραναζιών)
2. Κλείσιμο πόρτας (μαζί με περιστροφή γραναζιών)

Αυτό είναι ανεπιθύμητο για τους εξής λόγους:

1. Το animation "Περιστροφή γραναζιών" αντιγράφεται 2 φορές στα 2 παραπάνω animations χωρίς λόγο.
2. Στην περίπτωση που ο Ανδρόνικος πατήσει το κουμπί που αναφέραμε στο script παραπάνω, τότε όλο το OG θα πρέπει να μεταπηδήσει στο animation "Άνοιγμα πόρτας". Ναι μεν τα γρανάζια θα εξακολουθούν να γυρνάνε, αλλά θα αρχίσουν να περιστρέφονται από την αρχή του animation "Άνοιγμα πόρτας" και αυτό θα φανεί πολύ άσχημο.

Δεν είναι όμως μόνο εικαστικοί οι λόγοι του διαχωρισμού σε OG, αλλά και πρρακτικοί. Τί γίνεται πχ στην περίπτωση που το δωμάτιο έχει 2 πόρτες· Καθεμιά θα πρέπει να ανοιγοκλείνει ανεξάρτητα από την άλλη, οπότε και θα πρέπει να αποτελούν χωριστα OG.

Φυσικά πρέπει να προσέχουμε να μην κόβουμε το level σε περισσότερα OG από όσα χρειάζονται. Ιδιαίτερα τα animated OG βαραίνουν ένα level, αφού γίνεται δύσκολος ο προσδιορισμός του αν είναι ορατά από την camera ή όχι. Πρακτικά αυτό που θα κάνουμε είναι το εξής:

1. Φτιάχνουμε ένα μεγάλο OG με όλα τα στατικά αντικείμενα του level (πατώματα, τοίχοι, καναπέδες κλπ), αντικείμενα τα οποία από την αρχή ώς το τέλος του level θα παραμείνουν στην ίδια θέση.
2. Για κάθε ανεξάρτητα animated αντικείμενο προσθέτουμε ένα OG.

Κι εδώ τονίζεται το ανεξάρτητα. Ένα δωμάτιο πχ μπορεί να έχει τα γρανάζια που αναφέραμε, αλλά και άλλα πράγματα πχ φίδια που κρέμονται και κινούνται δεξιά αριστερά. Αν δούμε ότι όλα αυτά μαζί χωράνε σε μια λούπα, τότε καλό είναι να τα κάνουμε όλα ένα OG. Από την άλλη μπορεί αυτό να μας φανεί πολύ επαναλαμβανόμενο και κουραστικό, οπότε το σπάμε σε περισσότερα OG (ένα για τα γρανάζια και ένα για τα φίδια), ώστε να φαίνεται περισσότερο πειστική η κίνηση. Πάντα χρησιμοποιούμε κοινή λογική και ζυγίζουμε τους 2 παράγοντες: πόσο πιο ωραίο θα φανεί και πόσα resources θα ξοδέψει.

Πολύ σημαντικό είναι να περιορίζουμε τα animated ObjectGroups σε ένα (ή όσο το δυνατόν λιγότερα) δωμάτια. Αυτό γιατί ο εντοπισμός του αν είναι visible ή όχι δυσκολεύεται κατά πολύ και φτάνουμε στο σημείο να κάνουμε render πράγματα που τελικά δεν φαίνονται στην οθόνη. Πχ αποφασίζουμε να κάνουμε ένα OG τα γρανάζια του δωματίου Α και τα γρανάζια του δωματίου Β. Αφού μπορούν να γίνουν loop, αυτό φαίνεται σαν καλή ιδέα, δεδομένου ότι έχουμε και ένα OG λιγότερο στο level. Αυτό δεν πρέπει να συμβέι σε καμία περίπτωση, γιατι έτσι ενώ βρισκόμαστε στο δωμάτιο Α, σπαταλάμε resources για να υπολογίσουμε το animation στο δωμάτιο Β, που δεν θα είναι ορατό.

Ένα OG δεν είναι τίποτε άλλο από μια σκηνή στο Max. Περιέχει meshes, materials, lights, cameras, ιεραρχία από nodes και animation. Καλό είναι φώτα

και κάμερες να μπαίνουν όλα σε ένα OG, κατα προτίμηση σε αυτό που περιέχει όλα τα στατικά αντικείμενα του level, για διευκόλυνση του level designer. Βέβαια μερικά από αυτά δεν μπορούν να μπούν όλα στο ίδιο OG. Πχ μια λάμπα που πηγαίνει πέρα δώθε θα πρέπει να έχει και το αντίσοτοιχο light στο ίδιο OG, ώστε να ακολουθεί και αυτό το animation.

2.3 Batch

Ώς batch ορίζουμε ένα κομμάτι γεωμετρίας που θα τυπωθεί αυτούσιο στην οθόνη. Λόγω του τρόπου που γίνεται το rendering στην engine, ένα batch μπορεί να έχει μόνο ένα material. Το ίδιο συμβαίνει και με τα smoothing groups προς το παρόν. Λόγο του τρόπου λειτουργίας του *IGame* του MAX, κάθε smoothing group γίνεται ένα batch. Για να υπολογίσουμε πόσα batches έχει κάθε mesh χρήσιμοποιούμε τον εξής κανόνα:

$$B = M \times S$$

Όπου B ο αριθμός των batches, M ο αριθμός των materials, και S ο αριθμός των smoothing groups. Έτσι όσα meshes φτιάχνουμε ξεκινώντας από box, αυτόματα γίνονται εξαπλάσια batches, γιατι το box έχει 6 smoothing groups.

Όσο παράξενο και αν φαίνεται, ο αριθμός των batches είναι από τους σημαντικότερους περιοριστικούς παράγοντες για την ταχύτητα του game. Γι αυτό πρέπει να τον κρατάμε χαμηλά. Έτσι χρειάζεται να ομαδοποιούμε αρκετά meshes σε ένα, πράγμα που περιλαμβάνει και ομαδοποίηση πολλών textures σε ένα. Εδώ δεν συζητούνται τεχνικές για το πώς θα γίνει αυτό, αλλά γιατί και πότε πρέπει να γίνει.

Πουθενά δεν πρέπει να υπάρχουν batches με λίγα τρίγωνα (λιγότερα των 100 - 200). Είναι άσκοπη απώλεια να χρησιμοποιούμε ένα rendering call στην κάρτα γραφικών για να τυπωθεί μια αφίσα που αποτελείται από 2 τρίγωνα. Κάνουμε batch πάντα ομοειδή και γειτονικά πράγματα. Ομοειδή μπορεί να είναι μια σειρά αφίσες σε ένα δωμάτιο, που όπως είπαμε είναι σπατάλη να γίνουν ένα batch η κάθε μία. Όταν λέμε γειτονικά εννοούμε πράγματα που βρίσκονται εξορισμού μαζί, πχ ένα γραφείο με τα διάφορα ψιλοπράγματα που βρίσκονται πάνω του, ή ένα σαλόνι με καναπέδες, που και το ίδιο material έχουν και κοντά βρίσκονται. Βέβαια και εδώ πρέπει να δοθεί προσοχή και να ζυγίσουμε το κατά πόσο μας συμφέρει να batcharουμε πράγματα, με παρόμοιο τρόπο με αυτόν που αποφασίσαμε την ομαδοποίηση σε OG παραπάνω. Ποτέ πχ δεν batchάρουμε πράγματα που βρίσκονται σε χωριστά δωμάτια, αλλά μπορούμε να το σκεφτούμε και λίγο περισσότερο μέσα στο ίδιο το δωμάτιο. Για παράδειγμα, αν έχουμε ένα μακρόστενο δωμάτιο με 2 σαλόνια, ένα στη μια άκρη και ένα στην άλλη, τότε μπορεί να θέλουμε να κάνουμε τους καναπέδες 2 batches, ένα σε κάθε άκρη, αφού αναγνωρίζουμε ότι είναι πολύ δύσκολο να τα δει και τα 2 η camera ταυτόχρονα. Βέβαια μπορεί να αποφασίσουμε και το αντίθετο, στην περίπτωση που δεν μας ικανοποιεί το μέγεθος του batch. Τα πάντα εξαρτώνται από την περίσταση και δεν υπάρχει συγκεκριμένος κανόνας που πρέπει να ακολουθηθεί. Ποτέ δεν πρέπει να γίνονται υπερβολές στο batching. Ποτέ πχ δεν πρέπει να batchαριστεί ένα ολόκληρο δωμάτιο, γιατί αυτό θα προκαλέσει προβλήματα. Καταρχήν είναι πολύ δύσκολη δουλειά από μόνη της, και θα φάει πολύ χρόνο. Έπειτα θα προκύψουν πιθανώς προβλήματα με τα mipmaps, με την ανανέωση τπυ χώρου, αν κάτι δεν μας αρέσει τελικά, ή είναι προβληματικό για το gameplay κλπ. Τέλος αυτό θα έχει αρνητικό

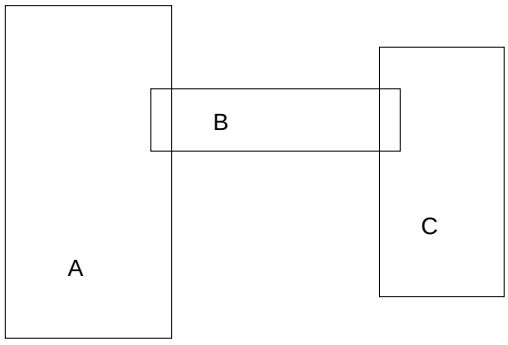
αποτέλεσμα στο framerate, γιατί αν έστω και μια μικρή γωνία του δωματίου είναι ορατή από την camera, τότε όλο το δωμάτιο θα τυπωθεί. Όταν ένα batch φαίνεται πολύ δύσκολο να δημιουργηθεί, τότε μάλλον υπάρχει πολύ καλός λόγος για να μην το κάνουμε.

2.4 PVS

Στα παραπάνω μιλάμε συχνά για visibility determination και για περιορισμό OG ή batch σε ένα δωμάτιο. Πώς όμως η engine ξέρει ποια είναι τα "δωμάτια"; Η απάντηση σε αυτό είναι τα PVS (Potential Visible Sets) meshes. Ένα PVS είναι ένα mesh που ικανοποιεί τα εξής κριτίρια:

- Είναι κλειστό, δηλαδή ορίζει έναν κλειστό όγκο και δεν αφήνει τρύπες.
- Είναι convex, δηλαδή κανένα ζεύγος γειτονικών τριγώνων δεν σχηματίζει γωνία μεγαλύτερη των 180 μοιρών.
- Περικλείει μια περιοχή περιορισμένης ορατότητας μέσα στο level. Τέτοια περιοχή μπορεί να είναι ένα δωμάτιο με ένα παράθυρο. Υπάρχει περιορισμένη ορατότητα, αφού το δωμάτιο φαίνεται μόνο από ένα παράθυρο.

Χρησιμοποιούμε PVS έτσι ώστε να περικλείουν τη γεωμετρία η οποία θέλουμε να αποτελέσει ένα δωμάτιο. Για να ορίσουμε σημεία οπτικής επαφής μεταξύ των PVS (παράθυρα, τρύπες στον τοίχο κλπ), Τα τέμνουμε μεταξύ τους στα συγκεκριμένα σημεία. Η παρακάτω εικόνα απεικονίζει 3 PVS και τις μεταξύ τους σχέσεις. Υπάρχουν 2 δωμάτια A και C, τα οποία επικοινωνούν με τον διάδρομο B. Δείχνουμε αυτή τη σχέση τέμνοντας λίγο τα δωμάτια με τον διάδρομο. Με αυτόν τον τρόπο είναι σαν να λέμε στην engnine "Το δωμάτιο A επικοινωνεί με τον διάδρομο B σε αυτό το σημείο και ο διάδρομος B επικοινωνεί με το δωμάτιο C σε αυτό το σημείο".



Με αυτό τον τρόπο, όταν η camera βρίσκεται στο δωμάτιο A και κοιτάει προς τα αριστερά, τότε η engine ξέρει ότι μόνο αντικείμενα του δωματίου A πρέπει να τυπωθούν. Έτσι γρήγορα γλυτώνουμε το τύπωμα των B και C, με δραματική αύξηση του frame rate.

2.5 Proxy

Εκτός από τα PVS, η engine πρέπει να ξέρει και διάφορα άλλα πράγματα για το level. Αυτά έχουν να κάνουν με τη λειτουργία του κάθε αντικειμένου μέσα

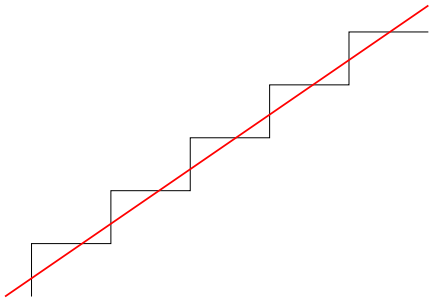
στο level. Πχ "αυτό είναι ένα πάτωμα". Ο χαρακτήρας πρέπει να πατάει πάνω του και να κάνει διάφορες κινήσεις (περπάτημα, τρέξιμο, άλμα ...). Ή "αυτό είναι ένας τοίχος". Ο χαρακτήρας πρέπει να σταματήσει όταν φτάσει εδώ. Πολλά από αυτά είναι εύκολο να υπολογιστούν. Πχ ένα πάτωμα είναι ένα mesh που έχει το normal προς τα πάνω. Αλλά για λόγους πρακτικούς χρειάζεται να τα ορίζουμε εμείς. Ο κυριότερος λόγος είναι ότι το collision detection (διαδικασία εύρεσης επαφών μεταξύ αντικειμένων) είναι πολύ βαριά εργασία για τη CPU και πρέπει να εφαρμόζεται σε όσο το δυνατόν λιγότερα τρίγωνα. Άλλοι λόγοι είναι ότι δεν θέλουμε όλα τα meshes σε μια σκηνή να κάνουν κάτι συγκεκριμένο. Πχ ο χαρακτήρας πρέπει να περνάει μέσα από ιστούς αράχνης, ή μπορεί κάποια στιγμή να θελήσουμε ο χαρακτήρας να πέφτει μέσα από ένα πάτωμα-οφθαλμαπάτη.

Για αυτούς τους λόγους χρησιμοποιούμε proxy meshes. Τα proxy πρέπει να ικανοποιούν τις εξής συνθήκες:

- Να έχουν όσο το δυνατόν λιγότερα και μεγάλα τρίγωνα. Λίγα γιατί όπως είπαμε βαραίνουν πολύ την όλη διαδικασία και μεγάλα, γιατί τα πολύ μικρά τρίγωνα στο proxy προκαλούν προβλήματα. Το collision dettection γίνεται με meshes της τάξης του χαρακτήρα (χέρι χαρακτήρα, κορμός χαρακτήρα κλπ), οπότε μικρά τρίγωνα στο proxy μπορει να έχουν ως αποτέλεσμα περίεργη συμπεριφορά.
- Να προσεγγίζουν την αληθινή γεωμετρία όσο το δυνατόν καλύτερα.
- Τα normals πρέπει να είναι προς τη μεριά του χαρακτήρα. Πχ ένα proxy πάτωμα πρέπει να έχει τα normals προς τα πάνω, από τη μεριά δηλαδή που το πατάει ο χαρακτήρας.
- Να ακολουθούν το animation του χώρου που προσεγγίζουν, αν αυτό υπάρχει.
- Να είναι όσο το δυνατόν convex. Το proxy χρησιμοποιείται και για να μην περνάει η camera μέσα από τοίχους. Σε περιπτώσεις μη-convex proxy, η camera συναντάει δυσκολίες.

Στο παράδειγμα της παρακάτω εικόνας, έχουμε να προσεγγίσουμε μία σκάλα. Έχουμε δύο επιλογές:

1. Να αναπαραστήσουμε τη σκάλα ως έχει στο proxy (άσπρη γραμμή)
2. Να την προσεγγίσουμε σαν ένα κεκλιμένο επίπεδο (κόκκινη γραμμή)



Το τί θα κάνουμε εξαρτάται από τις περιστάσεις. Αν τα σκαλοπάτια είναι σχετικά μεγάλα, τότε θα θέλουμε να δώσουμε την εντύπωση ανεβάσματος σε σκάλα, όπου ο χαρακτήρας ανεβαίνει σταδιακά. Αν όμως τα σκαλοπάτια είναι μικρά (πχ της τάξης του ποδιού του χαρακτήρα), τότε θα βλέπουμε "απρόβλεπτες" αναπηδήσεις από σκαλοπάτι σε σκαλοπάτι, σύμφωνα με το animation που έχει ο χαρακτήρας. Οπότε σε αυτή την περίπτωση καλύτερη είναι η προσέγγιση του κεκλιμένου επιπέδου. Σε μια σκάλα με μικρά σκαλοπάτια, περιμένουμε να δούμε μια πιο smooth ανάβαση.

3 Προδιαγραφές

Όταν εξάγουμε μια σκηνή σε t7sce/t7anm, πρέπει να έχουμε υπόψη τα ακόλουθα:

- Όλα τα textures πρέπει να είναι σε format Targa ή JPEG.
- Η σκηνή πρέπει να περιέχει και τα όποια proxy / PVS της έχουμε ορίσει.
- Τα proxy meshes ονομάζονται με πρόθεμα `proxy_`. Δηλαδή το proxy ενός mesh με όνομα `patoma2`, θα ονομάζεται `proxy_patoma2`. Αντίστοιχα κανένα mesh που δεν είναι proxy δεεν πρεπει να έχει όνομα που ξεκινάει από `proxy_`.
- Αντίστοιχα τα PVS meshes ονομάζονται με πρόθεμα `pvs_`. Πχ `pvs_dwmatio1`.
- Τα proxy meshes πρέπει να ακολουθούν το animation της γεωμετρίας. Πχ στην περίπτωση του κεκλιμένου επιπέδου παραπάνω, αν σε κάποιο animation τα σκαλιά έρχονται στο ίδιο επίπεδο (η σκάλα γίνεται επίπεδη), το κεκλιμένο επίπεδο θα πρέπει να περιστραφεί ανάλογα για να ακολουθήσει το animation. Σημειωτέο το ότι τα proxy υποστηρίζουν μόνο rigid animation (PRS), και όχι skinning.
- Τα PVS δεν πρέπει ποτέ να έχουν animation. Περιγράφουν στατικούς χώρους.

Ειδικά για την εξαγωγή χαρακτήρων:

- Το SCE αρχείο εξάγεται σε DaVinci pose.
- Τα διάφορα bones πρέπει να ονομάζονται όπως στο biped.
- Στην περίπτωση που ο χαρακτήρας δεν βασίζεται σε biped ή δεν έχει skin , πχ ένα ρομπότ με rigid μέλη, τότε πρέπει όλη η ιεραρχία από nodes να ονομάζεται όπως και στο biped.

4 Εγκατάσταση του plugin

Αντιγράφουμε το αρχείο `sce.dle` στο plugin directory του max. Συνήθως "`C:\Program Files\Autodesk\3dsMax8\plugins`".

5 Χρήση του plugin

Το plugin εξάγει δεδομένα σκηνής και animation σε αρχεία t7sce και t7anm αντίστοιχα. Για στατικές σκηνές εξάγουμε μόνο το t7sce αρχείο. Για σκηνές με ένα ή περισσότερα animations εξάγουμε επιπλέον και ένα ή περισσότερα t7anm αρχεία αντίστοιχα. Για να κάνουμε export επιέγουμε File->Export και σαν τύπο αρχείου επιλέγουμε t7sce ή t7anm.

Στην περίπτωση του t7sce αρχείου, μας εμφανίζεται και ένα dialog box με ορισμένες επιλογές. Το mesh export το αφήνουμε ως έχει. Αλλαγές θα κάνουμε μόνο αν το t7sce αρχείο είναι υπερβολικά μεγάλο. Στο texture action μπορούμε να επιλέξουμε ένα από τα ακόλουθα:

- No action.
- Copy in “data” folder. Αυτή η επιλογή θα κάνει και ένα texture collection στο φάκελο data, εκεί όπου επιλέξαμε να σώσουμε το t7sce.
- Pack in t7sce. Όλα τα textures θα αποθηκευτούν μέσα στο ίδιο το t7sce.

6 Συχνές ερωτήσεις

Ρωτήστε πρώτα και εδώ θα μπουν οι απαντήσεις.