Ιόνιο Πανεπιστήμιο, Τμήμα Πληροφορικής

Γραφικά με Υπολογιστές

Εργαστήριο 4 – Υφές

Στο εργαστήριο αυτό θα δούμε το πώς εφαρμόζουμε και πως παραμετροποιούμε υφές (textures) σε αντικείμενα στην OpenGL.

Υφές στην OpenGL

Για να αυξήσει την ρεαλισμό κατά την απεικόνιση αντικειμένων, η OpenGL υποστηρίζει μια δημοφιλή τεχνική που ονομάζεται απεικόνιση με υφές (texture mapping). Μια υφή μπορεί να είναι μια οποιαδήποτε εικόνα της οποία το μέγιστο μέγεθος εξαρτάται από τις δυνατότητα του υλικού (κάρτα γραφικών) που υποστηρίζει την OpenGL.

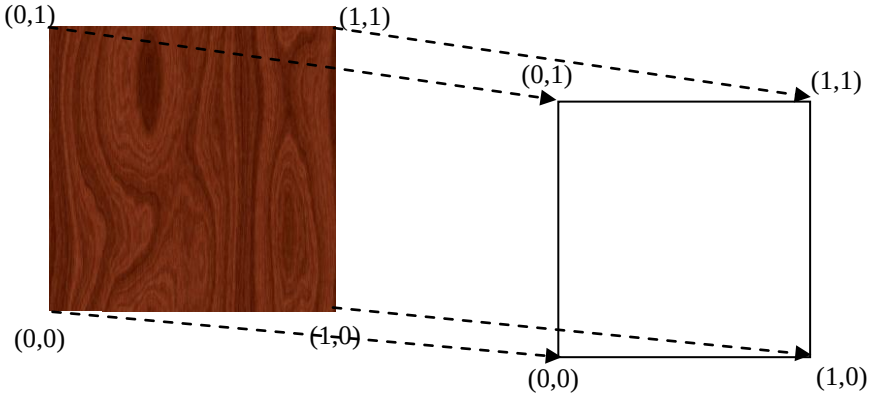
Η υφή μπορεί να εφαρμοστεί σε ένα αντικείμενο επιπλέον του φωτισμού και του χρωματισμού (Smooth ή Flat).

Εφαρμογή υφής σε ένα αντικείμενο

Η OpenGL μπορεί να εφαρμόσει μια υφή σε οποιοδήποτε αντικείμενο στην σκηνή αρκεί να της πούμε πώς να αντιστοιχήσει σημεία στο αντικείμενο με texels της υφής. Αυτό γίνεται μέσω ενός ζεύγους αριθμών που λέγονται συντεταγμένες υφής (texture coordinates) (s,t). Το ζεύγος s,t παίρνει τιμές από το διάστημα [0,1].

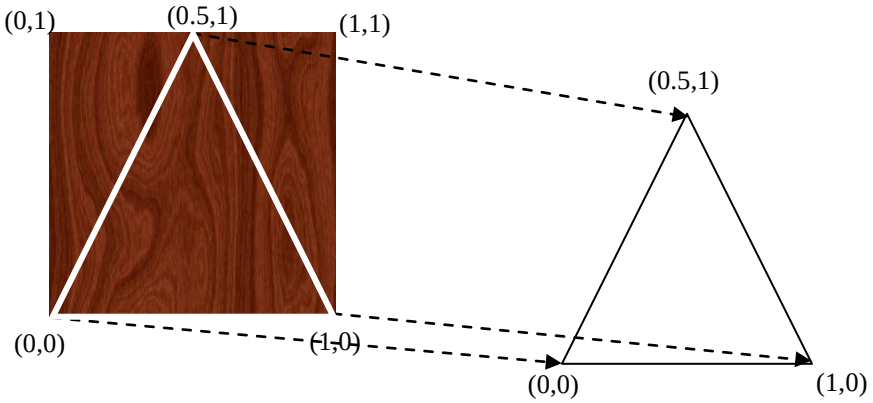
> **Το ζεύγος (s,t) δεν είναι άλλο από το γνωστό ζεύγος (u,v) που θα βρείτε στην βιβλιογραφία.**

Ένα ζεύγος συντεταγμένων (s,t) αντιστοιχεί σε κάθε κορυφή (vertex) του αντικειμένου πριν την εφαρμογή της υφής.



Στην γενική περίπτωση η υφή δεν θα ταιριάζει απόλυτα στην γεωμετρία μας, αλλά ένα μικρό τμήμα της θα εφαρμόζεται σ'αυτή.

Για παράδειγμα φανταστείτε ότι το αντικείμενο μας είναι τριγωνικό. Τότε η εφαρμογή της υφής στο τρίγωνο θα είναι κάπως έτσι:



Για να καθορίσουμε το ζεύγος συντεταγμένων (s,t) κάθε κορυφής χρησιμοποιούμε την εντολή:

```
void glTexCoord2f(Glfloat s, GLfloat t);
```

Την εντολή αυτή την καλούμε την στιγμή που δηλώνουμε τις κορυφές ενός αντικειμένου. Για παράδειγμα όταν κατασκευάζουμε ένα τετράγωνο:

```
glBegin(GL_QUADS);
      glTexCoord2f(0.0f, 1.0f);
      glVertex3f(-10, 10, -10);

      glTexCoord2f(1.0f, 1.0f);
      glVertex3f(10, 10, -10);

      glTexCoord2f(1.0f, 0.0f);
      glVertex3f(10, 10, 10);

      glTexCoord2f(0.0f, 0.0f);
      glVertex3f(-10, 10, 10);
glEnd();
```

Δημιουργώντας μια υφή

Στην γενική περίπτωση οι υφές είναι εικόνες (jpg, bmp, png, tga) που φορτώνουμε από το δίσκο σε κάποια περιοχή στην μνήμη. Από την στιγμή που έχουμε την περιοχή αυτή που κρατά τα bytes της εικόνας καθώς και το μέγεθος της μπορούμε να δημιουργήσουμε μια υφή στην OpenGL καλώντας την εντολή:

```
void glTexImage2D(GLenum target, GLint level, GLint internalformat,
                  GLsizei width, GLsizei height, GLint border,
                  GLenum format, GLenum type, void *data);
```

Ενδιαφέρον σε αυτή την εντολή έχουν οι παράμετροι width, height που είναι η ανάλυση της εικόνας, η internalformat που δηλώνει τι θέλουμε να αντιπροσωπεύει κάθε texel της υφής και φυσικά η data που παίρνει τα bytes της εικόνας.

Η παράμετρος internalformat μπορεί να πάρει τις εξής τιμές:

internalformat	Ερμηνεία
GL_RGB	Κάθε texel θα αντιπροσωπεύει ένα χρώμα RGB
GL_RGBA	Κάθε texel θα αντιπροσωπεύει ένα χρώμα RGB και διαφάνεια A
GL_LUMINANCE	Κάθε texel θα αντιπροσωπεύει φωτεινότητα
GL_LUMINANCE_ALPHA	Κάθε texel θα αντιπροσωπεύει φωτεινότητα και διαφάνεια A
GL_ALPHA	Κάθε texel θα αντιπροσωπεύει μόνο διαφάνεια A

> **Οι παράμετροι width και height μιας εικόνας που θέλουμε να μετατρέψουμε σε υφή πρέπει να είναι δυνάμεις του 2 για εκδόσεις της OpenGL μικρότερες τις 2.0**

Η υφή δημιουργήθηκε, αλλά πού; Πως θα την χρησιμοποιήσουμε, θα αλλάξουμε τις παραμέτρους της και πως θα την εφαρμόσουμε σε ένα αντικείμενο;

Η OpenGL για να μπορέσουμε να αποθηκεύσουμε και να διαχειριστούμε μια υφή μας δίνει την δυνατότητα να κατασκευάσουμε ένα **αντικείμενο υφής** (texture object). Ένα αντικείμενο υφής είναι μια περιοχή στην μνήμη που εκτός από τα bytes της εικόνας της υφής κρατά και πολλές άλλες πληροφορίες όπως μέγεθος, μέθοδος φιλτραρίσματος κατά την μεγέθυνση και σμίκρυνση, αν έχει mipmap κα.

Για να δημιουργήσουμε ένα αντικείμενο υφής αρκεί να καλέσουμε την εντολή:

```
glGenTextures(GLuint numtextures, GLuint* textureID );
```

Η εντολή θα δημιουργήσει το αντικείμενο και θα μας επιστρέψει μια αναφορά στην υφή (ένας μοναδικός ακέραιος αριθμός). Με βάση αυτή τον αριθμό αναφοράς μπορούμε να διαχειριστούμε πλήρως μια υφή. Η παράμετρος numtextures μας επιτρέπει να δημιουργήσουμε παραπάνω του ενός αντικείμενα υφής ταυτόχρονα. Σ'αυτή τη περίπτωση η εντολή θα επιστρέψει ένα πίνακα με τους αριθμούς αναφοράς.

Παράδειγμα για να φορτώσουμε την εικόνα της υφής ενεργοποιούμε το αντικείμενο υφής και καλούμε την glTexImage2D() όπως αναφέραμε παραπάνω:

```
glGenTextures(1, &textureID );
glBindTexture(GL_TEXTURE_2D, textureID);
glTexImage2D(GL_TEXTURE_2D, ……);
```

Η ενεργοποίηση του αντικειμένου υφής γίνεται με την εντολή glBindTexture() που παίρνει ως παράμετρο τον αριθμό αναφοράς του αντικειμένου όπως εξηγήσαμε παραπάνω. Η παράμετρος GL_TEXTURE_2D που θα δείτε και σε παρακάτω εντολές διαχείρισης υφών λέει ότι οι υφές μας είναι διδιάστατες εικόνες. Υπάρχει δυνατότητα στην OpenGL να χρησιμοποιήσουμε επίσης και μονοδιάστατες ή τριδιάστατες εικόνες ως υφές.

> **Οποτεδήποτε χρειαστεί να κάνουμε μια αλλαγή στις παραμέτρους μια υφής πρέπει προηγουμένως να ενεργοποιήσουμε το αντικείμενο υφής με την χρήση της εντολής glBindTexture();**

Παραμετροποίηση μιας υφής

Κάθε υφή μπορεί να παραμετροποιηθεί ως προς το πώς θα συμπεριφερθεί κατά την μεγέθυνση/σμίκρυνση, και τον τρόπο με τον οποίο θα εφαρμοστεί σε ένα αντικείμενο. Για να θέσουμε μια παράμετρο της υφής πρέπει πρώτα να ενεργοποιήσουμε το αντικείμενο υφής με την εντολή glBindTexture() και έπειτα καλώντας κάποια από τις εντολές:

```
void glTexParameterf(GLenum target, GLenum pname, GLfloat param);
void glTexParameteri(GLenum target, GLenum pname, GLint param);
void glTexParameterfv(GLenum target, GLenum pname, GLfloat *params);
void glTexParameteriv(GLenum target, GLenum pname, GLint *params);
```

Η διαφορά μεταξύ τους έγκειται στο τι τύπο παραμέτρου δέχονται (float ή integer) και αν δέχονται ένα αριθμό ή ένα διάνυσμα αριθμών (vector).

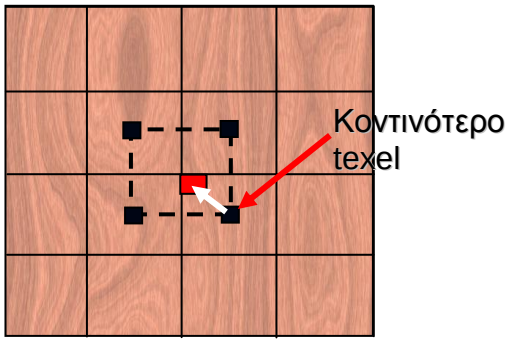
Η παράμετρος target είναι πάντα GL_TEXTURE_2D στην συγκεκριμένη εφαρμογή (υφές 2 διαστάσεων).

Με την χρήση αυτής της εντολής μπορούμε να θέσουμε για παράδειγμα τον τύπο φιλτραρίσματος της υφής όταν αυτή μεγεθύνεται:

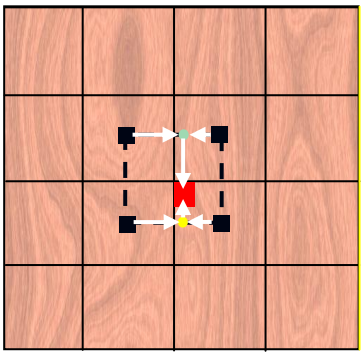
```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
```

Η GL_NEAREST θα εφαρμόσει την μέθοδο «του κοντινότερου γείτονα (nearest neighbour)» κατά την δειγματοληψία της υφής που οδηγεί σε aliasing στην εικόνα του αντικειμένου (είναι ορατά τα pixel).

Αντ' αυτής μπορούμε να χρησιμοποιήσουμε την τιμή GL_LINEAR που οδηγεί σε διγραμμική παρεμβολή (bilinear interpolation) μεταξύ των texel της υφής και ομαλότερα αποτελέσματα στην εικόνα του αντικειμένου.



Η μέθοδος του κοντινότερου γείτονα



Διγραμμική παρεμβολή texel

Αντιστοίχως μπορούμε να θέσουμε και το είδος φιλτραρίσματος της υφής κατά την σμίκρυνση της με την παράμετρο GL_TEXTURE_MIN_FILTER. Και αυτή δέχεται τις τιμές GL_NEAREST και GL_LINEAR.

Μπορούμε επίσης να καθορίσουμε το πώς θα συμπεριφερθεί η υφή όταν οι συντεταγμένες υφής πέφτουν έξω από το διάστημα [0,1]. Μπορούμε να ορίσουμε την συμπεριφορά για κάθε συντεταγμένη υφής ανεξάρτητα με τις παραμέτρους GL_TEXTURE_WRAP_S και GL_TEXTURE_WRAP_T. Οι ποιο συνήθεις τιμές που χρησιμοποιούμε με αυτές τις παραμέτρους είναι οι GL_REPEAT και GL_CLAMP.

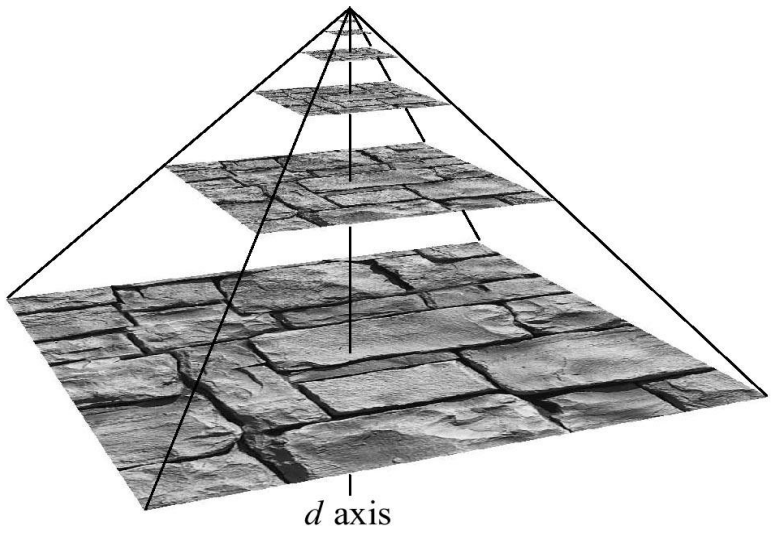
Η GL_REPEAT ορίζει ότι όταν μια συντεταγμένη s,t γίνει μεγαλύτερη του 1, τότε ως τιμή του s,t θα χρησιμοποιηθεί το δεκαδικό μέρος του αριθμού. Για παράδειγμα αν (s,t)=(2.3 , 4.5) τότε σαν συντεταγμένη υφής θα χρησιμοποιηθεί το (0.3, 0.5). Μπορείτε να φανταστείτε ότι αν συντεταγμένες συνεχίσουν να αυξάνονται γραμμικά αυτό θα οδηγήσει σε μια επανάληψη της εικόνας (REPEAT) της υφής.

Η GL_CLAMP ορίζει ότι όταν μια συντεταγμένη s,t γίνει μεγαλύτερη του 1 η τιμή της θα μένει στο 1. Για παράδειγμα αν (s,t)=(2.3 , 4.5) τότε σαν συντεταγμένη υφής θα χρησιμοποιηθεί το (1, 1).

Mipmapping

Η τεχνική mipmapping δημιουργεί πολλές διαφορετικές εκδόσεις της αρχικής υφής σε μικρότερες αναλύσεις. Κατά την εφαρμογή της υφής στο αντικείμενο, η OpenGL επιλέγει μια ανάλυση της υφής που ταιριάζει καλύτερα στο μέγεθος του αντικειμένου στην υφή και την εφαρμόζει, αντί του να εφαρμόζει πάντα την ίδια μεγάλη υφή.

Ένα mipmap μιας υφής έχει αυτή την μορφή:



Κάθε επίπεδο περιέχει μια έκδοση της υφής του προηγούμενου επιπέδου 4 φορές μικρότερη (4 texel σε ένα επίπεδο συνδυάζονται για να δημιουργήσουν 1 texel στο επόμενο επίπεδο). Αυτό δημιουργεί μια συνολική αύξηση του μεγέθους της υφής κατά 33% περίπου.

Η τεχνική mipmapping λύνει το πρόβλημα της σωστής σμίκρυνσης μιας υφής όταν πρέπει να εφαρμοστεί σε μακρινά αντικείμενα καθώς και επιταχύνει την εφαρμογή της υφής σ'αυτά τα αντικείμενα λόγω του έχει να δειγματοληπτίσει υφές μικρής ανάλυσης.

Η OpenGL μπορεί να δημιουργήσει ένα mipmap μιας υφής αυτόματα με την εντολή

```
int gluBuild2DMipmaps(GLenum target, GLint internalFormat,
                      GLint width, GLint height,
                      GLenum format, GLenum type, const void
                      *data);
```

Οι παράμετροι της εντολής αυτής είναι ίδιες με της εντολής glTexImage2D() που είδαμε πιο πριν. Προϋποθέτει ότι έχουμε φορτώσει την εικόνα της υφής σε μια

περιοχή μνήμης και ότι πριν έχουμε δημιουργήσει και ενεργοποιήσει ένα αντικείμενο υφής.

Η OpenGL στην γενική περίπτωση συνδυάζει δυο texel από γειτονικές υφές στο mipmap για να παράγει ένα χρώμα. Μπορούμε να καθορίσουμε επακριβώς το θα γίνεται αυτός ο συνδιασμός. Αυτό εξαρτάται από το είδος φιλτραρίσματος που ορίζουμε για το mipmap με την γνωστή εντολή glTexParameteri(), μέσω των παραμέτρων GL_TEXTURE_MAG_FILTER και GL_TEXTURE_MIN_FILTER. Για ένα mipmap έχουμε τις εξής επιλογές φιλτραρίσματος:

Τιμή	**Ερμηνεία**
GL_NEAREST	Θα εφαρμοστεί η μέθοδος nearest neighbour στην αρχική (πλήρης ανάλυσης) υφή μόνο
GL_LINEAR	Θα εφαρμοστεί η μέθοδος bilinear interpolation στην αρχική (πλήρης ανάλυσης) υφή μόνο
GL_NEAREST_MIPMAP_NEAREST	Επέλεξε την κοντινότερη υφή στο mipmap και εφάρμοσε την μέθοδο nearest neighbour
GL_NEAREST_MIPMAP_LINEAR	Εφάρμοσε γραμμική παρεμβολή μεταξύ δυο επιπέδων στο mipmap και έπειτα εφάρμοσε τη μέθοδο nearest neighbour στο αποτέλεσμα
GL_LINEAR_MIPMAP_NEAREST	Επέλεξε την κοντινότερη υφή στο mipmap και εφάρμοσε την μέθοδο γραμμικής παρεμβολής
GL_LINEAR_MIPMAP_LINEAR	Εφάρμοσε γραμμική παρεμβολή μεταξύ δυο επιπέδων στο mipmap και έπειτα εφάρμοσε τη μέθοδο γραμμικής παρεμβολής στο αποτέλεσμα

Η μέθοδος GL_NEAREST_MIPMAP_NEAREST είναι η γρηγορότερη μέθοδος φιλτραρίσματος ενός mipmap αλλά παράγει μέτρια αποτελέσματα, η GL_LINEAR_MIPMAP_LINEAR είναι πιο αργή μέθοδος, αλλά παράγει τα καλύτερα αποτελέσματα.

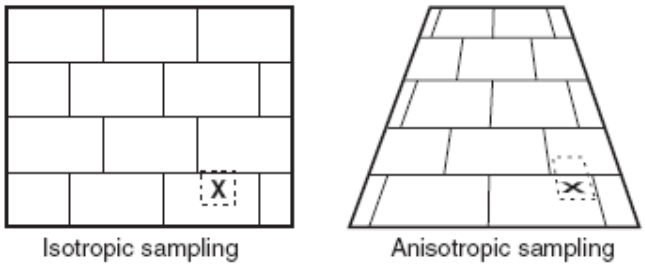
Μη ισοτροπικό φιλτράρισμα υφών (Anisotropic Filtering)

Οι συνήθεις μέθοδοι για το φιλτράρισμα μια υφής είναι όπως είπαμε η μέθοδος του κοντινότερου γείτονα (nearest neighbour) και της διγραμμικής παρεμβολής (bilinear interpolation). Και οι δύο μέθοδοι λαμβάνουν υπόψη την τετράδα των texel μέσα στο οποίο ένα pixel αντιστοιχεί και με συνδυασμό (ή μη) αυτή παράγουν το χρώμα του pixel.

Οι δυο μέθοδοι αποδίδουν καλά όταν βλέπουμε την επιφάνεια σχεδόν κάθετα. Όταν βλέπουμε το αντικείμενο με μικρή γωνία όμως οι μέθοδοι φιλτραρίσματος αυτοί κάνουν την υφή να δείχνει θολή. Αυτό συμβαίνει γιατί καθώς η γωνία της επιφάνειας γίνεται μικρή ένα pixel «καλύπτει» περισσότερα texel στην υφή, όμως ακόμα χρησιμοποιούμε 4 texel για να παράγουμε το χρώμα ενός pixel (λίγα δείγματα για μεγάλη επιφάνεια υφής). Αυτός ο τύπος φιλτραρίσματος λέγεται ισοτροπικός γιατί γίνεται πάντα με τον ίδιο τρόπο ανεξάρτητα της γωνίας με την οποία βλέπουμε μια επιφάνεια.

Η μη ισοτροπική μέθοδος φιλτραρίσματος (anisotropic filtering), αλλάζει τον αριθμό των texel που χρησιμοποιεί για να παράγει το χρώμα ενός pixel με βάση την γωνία

θέασης. Έτσι για μικρές γωνίες χρησιμοποιεί περισσότερα texel και οδηγεί σε λιγότερο θόλωμα της εικόνας.



Η μέθοδος αυτή είναι ακριβή υπολογιστικά, αλλά παράγει το καλύτερο αποτέλεσμα από κάθε είδος φιλτραρίσματος υφής που είδαμε μέχρι τώρα.

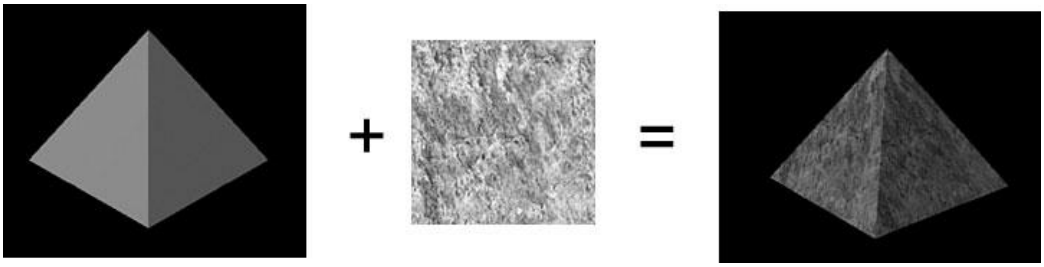
Για να ενεργοποιήσουμε anisotropic filtering στην OpenGL αρκεί να θέσουμε την παράμετρο GL_TEXTURE_MAX_ANISOTROPY_EXT στο αντικείμενο υφής.

```
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAX_ANISOTROPY_EXT,
fLargest);
```

Η τιμή της παραμέτρου αυτής είναι το μέγιστο ποσό φιλτραρίσματος που θα χρησιμοποιηθεί και εξαρτάται από την κάρτα γραφικών. Τιμή 1.0 απενεργοποιεί το anisotropic filtering.

Συνδυασμός χρώματος υφής και χρώματος αντικειμένου

Είναι δυνατόν να εφαρμόσουμε μια υφή σε ένα αντικείμενο για το οποίο έχουμε προηγουμένως ορίσει υλικό και φωτίσει. Η OpenGL μας παρέχει μερικές επιλογές για τον πως μπορούμε να συνδυάσουμε το χρώμα που έχει προκύψει από τον φωτισμό και το χρώμα που μας δίνει η υφή.



Οι παράμετροι που ορίζουν τον συνδυασμό χρωμάτων δίνονται μέσω των εντολών:

```
void glTexEnvi(GLenum target, GLenum pname, GLint param);
void glTexEnvf(GLenum target, GLenum pname, GLfloat param);
void glTexEnviv(GLenum target, GLenum pname, GLint *param);
void glTexEnvfv(GLenum target, GLenum pname, GLfloat *param);
```

Η διαφορά μεταξύ τους έγκειται στο τι τύπο παραμέτρου δέχονται (float ή integer) και αν δέχονται ένα αριθμό ή ένα διάνυσμα αριθμών (vector).

Η παράμετρος που ορίζει τον συνδυασμό χρωμάτων ονομάζεται GL_TEXTURE_ENV_MODE.

Παράδειγμα αν θέλουμε το χρώμα που προκύπτει από τον φωτισμό να πολλαπλασιάζεται(modulated) με το χρώμα της υφής και το αποτέλεσμα να απεικονίζεται στην οθόνη τότε:

```
glTexEnvi(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);
```

Άλλες τιμές για την παράμετρο αυτή είναι η GL_REPLACE που θα αντικαταστήσει το χρώμα που προέκυψε από το φωτισμό με το χρώμα της υφής (δηλαδή ο φωτισμός δεν θα έχει επίδραση στο αντικείμενο) και οι GL_ADD που θα προσθέσει τα δυο αυτά χρώματα.

Μετασχηματισμοί συντεταγμένων υφής

Οι συντεταγμένες μια υφής μπορούν να μετασχηματιστούν σαν να ήταν η υφή κανονικό αντικείμενο, μέσω των εντολών glTranslate(), glRotate() και glScale().

Η OpenGL διατηρεί έναν ξεχωριστό πίνακα μετασχηματισμών για τις υφές που συσσωρεύει ότι μετασχηματισμό δημιουργήσουμε με τις άνω εντολές (ανάλογος του GL_MODELVIEW). Ο πίνακας αυτός ονομάζεται GL_TEXTURE και ενεργοποιείται με την εντολή:

```
glMatrixMode(GL_TEXTURE);
```

Παράδειγμα αν θέλουμε να μετακινήσουμε μια υφή κατά 0.5 σε κάθε διάσταση S, T αρκεί να ορίσουμε τον μετασχηματισμό:

```
glMatrixMode(GL_TEXTURE);
glLoadIdentity();
glTranslatef(0.5,0.5,0);
```

πριν ζωγραφίσουμε το αντικείμενο με την υφή. Το πώς θα φανεί η υφή μετά τον μετασχηματισμό εξαρτάται από την τιμή που έχουμε θέσει στις παραμέτρους GL_TEXTURE_WRAP_S και GL_TEXTURE_WRAP_T.

Εφαρμογή υφών στην OpenGL

Στην σημερινή εργαστηριακή άσκηση θα εφαρμόσουμε διάφορες υφές σε αντικείμενα της σκηνής και θα τις παραμετροποιήσουμε.

Φορτώστε το project lab4.dev στο περιβάλλον ανάπτυξης εφαρμογών DevC++.

Ο κώδικας της εφαρμογής βρίσκεται στο αρχείο main.cpp. Ο κώδικας περιέχει σχόλια για όλα τα σημεία του προγράμματος που μας ενδιαφέρουν. Κομμάτια του κώδικα και άλλες παραμετροποιήσεις δεν μας αφορούν σε αυτή την άσκηση και μένουν ασχολίαστα.

Η εφαρμογή ακολουθεί το γνωστό σκελετό που χρησιμοποιούμε στις εργαστηριακές ασκήσεις. Οι αλλαγές στην σημερινή άσκηση εστιάζονται στην συνάρτηση αρχικοποίσης init() και στην συνάρτηση που ζωγραφίζει την σκηνή μας renderScene():

init()

Σ'αυτή την άσκηση φορτώνουμε επιπλέον τις υφές που θα χρησιμοποιήσουμε:

```
textures[TEXTURE_WALL] = LoadBitmap("textures/brick.bmp");
textures[TEXTURE_SPHERE] = LoadBitmap("textures/stone1.bmp");
```

Η LoadBitmap() είναι μια βοηθητική συνάρτηση που χρησιμοποιούμε για να φορτώσουμε ένα αρχείο bmp από το δίσκο και που ορίζεται στο αρχείο texture.cpp. Η συνάρτηση LoadBitmap() αφού φορτώσει την εικόνα, δημιουργεί ένα αντικείμενο υφής, δημιουργεί το mipmap για την υφή και ορίζει την μέθοδος φιλτραρίσματος:

```
//Δημιούργησε ένα αντικείμενο υφής
glGenTextures(1, &textureID );
//ενεργοποίησε το αντικείμενο υφής
glBindTexture(GL_TEXTURE_2D, textureID);

//Δημιούργησε το mipmap για αυτή την υφή
gluBuild2DMipmaps(GL_TEXTURE_2D, 4, infoheader.biWidth,
infoheader.biHeight, GL_RGBA, GL_UNSIGNED_BYTE, l_texture);

//Όρισε την συμπεριφορά της υφής όταν οι s,t συντεταγμένες είναι έξω
//από το διάστημα [0,1]
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
//Όρισε την μεθοδο φιλτραρίσματος της υφής
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_NEAREST_MIPMAP_NEAREST);

//Όρισε πώς θα χρησιμοποιούμε το χρώμα της υφής σε σχέση με το
//υπάρχον χρώμα της επιφάνειας.
glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);
```

renderScene()

Στην renderScene() ενεργοποιούμε τις υφές με την εντολή glEnable(GL_TEXTURE_2D) και ζωγραφίζουμε κάθε αντικείμενο (την σφαίρα και το δωμάτιο) ενεργοποιώντας τις αντίστοιχες υφές τους.

Για να ζωγραφίσω κάθε τοίχο του δωματίου χρησιμοποιώ τις εξής εντολές:

```
//ζωγράφισε τον τοίχο
glBindTexture(GL_TEXTURE_2D, textures[TEXTURE_WALL2]);
glBegin(GL_QUADS);
      vNormal[0] = 0; vNormal[1] = 0; vNormal[2] = -1;
      glNormal3fv(vNormal);
      //θέσε τις συντεταγμένες υφής για αυτή την κορυφή
      glTexCoord2f(0.0f, 1.0f);
      glVertex3f(-width, height, length);

      glNormal3fv(vNormal);
      glTexCoord2f(1.0f, 1.0f);
      glVertex3f(width, height, length);

      glNormal3fv(vNormal);
      glTexCoord2f(1.0f, 0.0f);
      glVertex3f(width, -height, length);

      glNormal3fv(vNormal);
      glTexCoord2f(0.0f, 0.0f);
      glVertex3f(-width, -height, length);
glEnd();
```

Αρχικά ενεργοποιώ την υφή του τοίχου και στην συνέχεια τον ζωγραφίζω σαν GL_QUAD (τετράπλευρο), ορίζοντας για κάθε κορυφή την θέση της glVertex3f(), τις συντεταγμένες της υφής glTexCoord2f() καθώς και ένα κανονικό διάνυσμα που θα χρησιμοποιηθεί κατά τον φωτισμό glNormal3fv().

Χειρισμός προγράμματος

Με τα **βελάκια** (cursor keys) μπορούμε να περιστρέψουμε πάνω-κάτω και αριστερά-δεξιά την κάμερα. Με τα πλήκτρα **z,c** μετακινούμε την κάμερα αριστερά-δεξιά και με τα **s,x** την μετακινούμε πίσω-μπρος.
Επίσης με το πλήκτρο **'a'** μπορούμε να εμφανίσουμε και να κρύψουμε το σύστημα αξόνων του κόσμου μας. Με το **spacebar** μπορούμε να κάνουμε το φως να περιστρέφεται γύρω από το αντικείμενο μας.

Πράγματα να δοκιμάσετε

Α) Εφαρμόστε την υφή TEXTURE_WALL στον «τοίχο»

Β) Παρομοίως εφαρμόστε την υφή TEXTURE_SPHERE στο μοντέλο της σφαίρας.

Γ) Κοιτάξτε την σφαίρα με την υφή πέτρας. Παρατηρείστε την ανακλάσεις του φωτός (γυαλάδες) πάνω στη σφαίρα. Με το spacebar περιστρέψτε το φως ώστε να έρθει πίσω από τι σφαίρα και σταματήστε το πάλι με το spacebar. Τι συμβαίνει με τις γυαλάδες τώρα;

Δ) Στρέψτε την κάμερα προς τον πέτρινο τοίχο (στα αριστερά όπως κοιτάει αρχικά) με τα βελάκια. Μετακινήστε την κάμερα μπρος-πίσω (πλήτρα s-x). Τι παρατηρείτε;

Ε) Στην συνάρτηση LoadBitmap() (αρχείο texture.cpp), αλλάξτε τις τιμές των παραμέτρων φιλτραρίσματος

```
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
```

διαδοχικά σε GL_LINEAR (κάντε το πρώτα για το `GL_TEXTURE_MAG_FILTER`). Γυρίστε την κάμερα προς τον πέτρινο τοίχο όπως πριν και μετακινήστε την κάμερα κοντά στον τοίχο (και μετά τραβήξτε την προς τα πίσω). Τι παρατηρείτε τώρα σχετικά με την επίδραση του bilinear interpolation κατά την μεγέθυνση/σμίκρυνση μιας υφής;

Ζ) Αλλάξτε την τιμή της παραμέτρου `GL_TEXTURE_MIN_FILTER` διαδοχικά σε GL_NEAREST_MIPMAP_NEAREST, GL_LINEAR_MIPMAP_NEAREST και GL_LINEAR_MIPMAP_ LINEAR δοκιμάζοντας το βήμα (Β) για κάθε περίπτωση.

Τι συμπεράσματα βγάζετε για τις τεχνικές φιλτραρίσματος του mipmap;

Η) Θέστε τις μεθόδους φιλτραρίσματος τις υφής σε

```
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_LINEAR_MIPMAP_LINEAR);
```

στην συνάρτηση LoadBitmap()

Φέρτε την κάμερα στην μία άκρη του τοίχου με τα κόκκινα τούβλα και στρέψτε την ώστε να κοιτά σχεδόν παράλληλα με αυτόν. Τι παρατηρείτε σχετικά με την ποιότητα της υφής στο βάθος του τοίχου;

Πατήστε το πλήκτρο **f** που ενεργοποιεί/απενεργοποιεί το anisotropic filtering. Τι παρατηρείτε σχετικά με την ποιότητα της υφής τώρα; Περιηγηθείτε στο δωμάτιο και κοιτάξτε διάφορες υφές με ή χωρίς anisotropic filtering.

Θ) Δοκιμάστε να εφαρμόσετε τον μετασχηματισμό μετακίνησης κατά τον άξονα των Χ στην υφή του πατώματος. Ο πίνακας μετασχηματισμού υφής GL_TEXTURE είναι ήδη ενεργοποιημένος (με την glMatrixMode(GL_TEXTURE);) . Μπορείτε απλά να εφαρμόσετε μια μετακίνηση με τον glTranslatef(). Έχετε στην διάθεση σας την μεταβλητή **a** που αλλάζει με το χρόνο.

Προσοχή: να σώσετε τον GL_TEXTURE στη στοίβα με glPushMatrix()/glPopMatrix() πριν εφαρμόσετε τον μετασχηματισμό.