Texture & 3D Models

عملي مشترك
محتوى مجاني غير مخصص للبيع التجاري

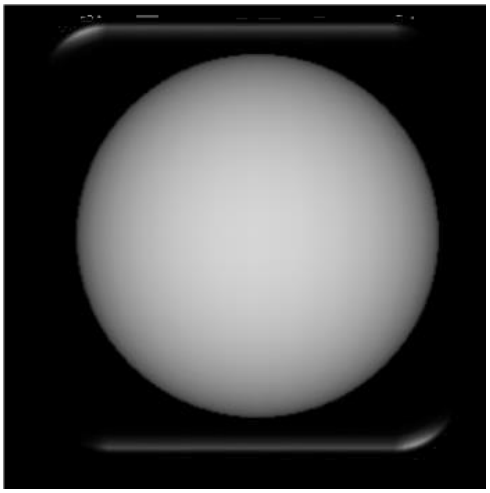
RB Informatics; 24/11/2021 البيانيات والرسم بمعونة الحاسب

Texture

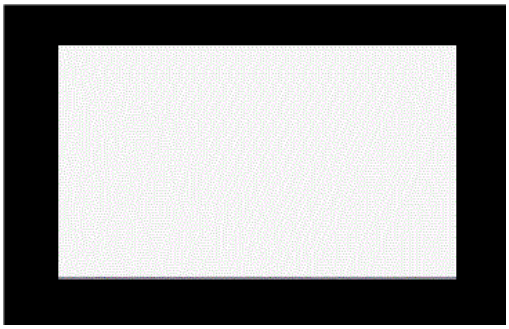
- Texture mapping:

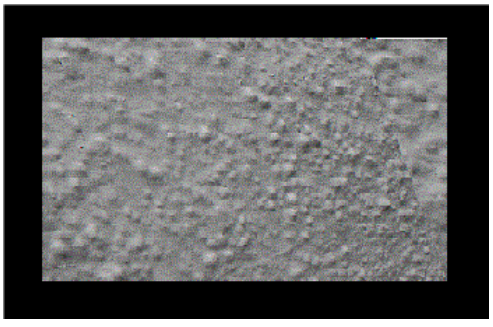
-عند رسم أشكال هندسية وتلوينها نحصل على نتيجة بعيدة عن الواقع، ولتحقيق درجة أعلى من الواقعية نستخدم تقنية الإكساء texture، تأخذ هذه التقنية صورة لسطح حقيقي ثم تطبق هذه الصورة على سطح مضلع، على سبيل المثال:
إذا رسمنا كرة بدون إكساء فستبدو كرة عادية غير واقعية أما إذا كسيناها بخريطة الكرة الأرضية فسنحصل على محاكاة للكرة الأرضية، لاحظ الأمثلة التالية:

قبل تطبيق الإكساء:



بعد تطبيق الإكساء:





خطوات تطبيق الإكساء:

- تهيئة البيئة بإضافة الملفات اللازمة.
- تهيئة OpenGL لتنفيذ الإكساء عبر تفعيل Texture ضمن التابع ( )InitGL مع تحديد نوع الـ texture من خلال التابع glEnable(Glenum parameter):

حيث الـ parameter يمكن أن يكون:

GL_TEXTURE_1D
GL_TEXTURE_2D
GL_TEXTURE_3D

- تحميل الصورة من الـ Hard disk إلى ال RAM عبر التابع التالي:

```
int LoadTexture (char*filename, int alpha=255);
```

الخطوات:

أولاً: نعرف متغير لتخزين الصورة قبل تابع ()InitGL:

```
int Image;
```

ثانيًا: نستدعي تابع التحميل ضمن تابع ()InitGL:

```
Image = LoadTexture ();
```

يعيد هذا التابع عنوان الصورة في الذاكرة ووسائطه هي اسم الصورة مع مسار الوصول إليها.

وتعبر alpha عن عتامة الصورة كلما نقصت زادت شفافية الصورة، وللحفاظ على الوضع الطبيعي نضع alpha = 255

- يكون العمل الآن داخل تابع () DrawGLScene، نهيئ الصورة لتصبح مناسبة للمشهد عبر التابع:

```
glBindTexture (Glenum Target, Gluint Texture);
```

حيث نحدد فيه عنوان الصورة والبعد المطلوب:

Target:
GL_TEXTURE_1D
GL_TEXTURE_2D
GL_TEXTURE_3D

Texture: الـ int الذي يرجعه التابع LoadTexture في الخطوة السابقة

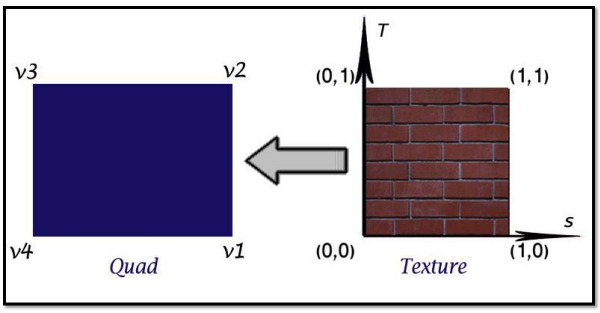
- مقابلة كل نقطة من الشكل مع نقاط الصورة (تكون بشكل افتراضي مربع طول ضلعه 1 عبر التابع:

```
glTexcoord2f (s, t)
```

حيث يأخذ إحداثيات النقطة كوسيط، يُستفاد من هذه الإحداثيات في تحديد محاذاة التركيب بالنسبة للعنصر المطبق الإكساء عليه. على سبيل المثال: من أجل التراكيب ثنائية البعد يكون مجال إحداثيات التراكيب من 0.0 إلى 1.0 في كلا الاتجاهين، لكن إحداثيات العنصر المراد إكساءه يمكن أن يكون أي شيء، لنفرض مثلًا أنه لدينا جدار شكله مربع وأردنا إكساءه بنسخة تركيب واحدة، فستكون إحداثيات التركيب (0.0) للزاوية اليسارية السفلية و (1,1) للزاوية اليمينية العلوية و (0,1) للزاوية اليسارية العلوية.

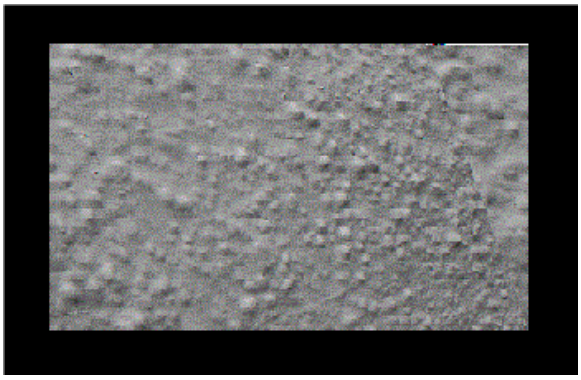
لاحظ المثال التالي:

```
glBegin (GL_QUADS);
glVertex3fv(V1);        glTexCoord2f(1,0);
glVertex3fv(V2)         glTexCoord2f(1,1);
glVertex3fv(V3)         glTexCoord2f(0,1);
glVertex3fv(V4)         glTexCoord2f(0,0);
glEnd ();
```



مثال: رسم شكل رباعي وإكساؤه بصورة تم تحميلها مسبقًا:

```
glBegin(GL_QUADS);
glBindTexture( GL_TEXTURE_2D,Image);
glTexCoord2f(0.0f, 0.0f);
glVertex3f(-5.0f,0.0f,0.0f);
glTexCoord2f(0.0f, 1.0f);
glVertex3f(-5.0f,6.0f,0.0f);
glTexCoord2f(1.0f, 1.0f);
glVertex3f(5.0f,6.0f,0.0f);
glTexCoord2f(1.0f, 0.0f);
glVertex3f(5.0f,0.0f,0.0f);
glEnd();
```



■ Repeating Texture:

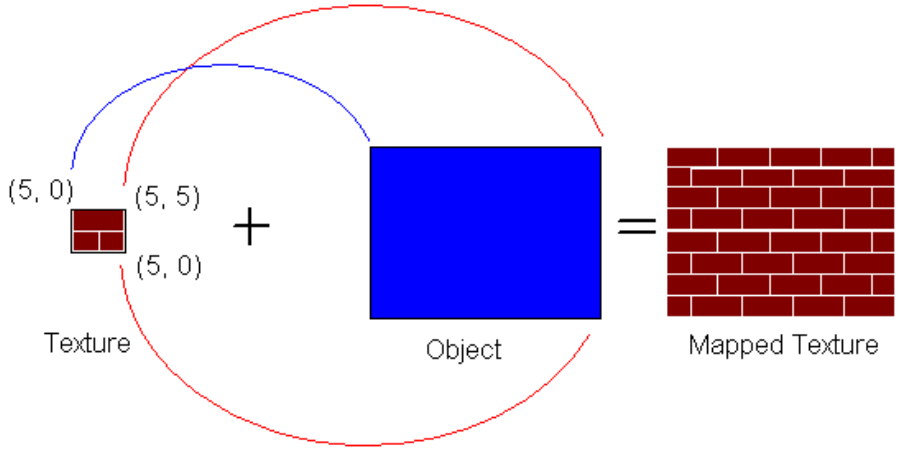
في حال كان الشكل أكبر من الصورة يمكن أن نقوم بتكرار الصورة على الشكل، وذلك من خلال التابع الذي يقوم بالتكرار على محور معين في حال كانت قيمة s او t ضمن التابع glTexCoord2f () أكبر من الواحد وتكون هذه القيمة هي مقدار التكرار.

للتكرار على محور X:

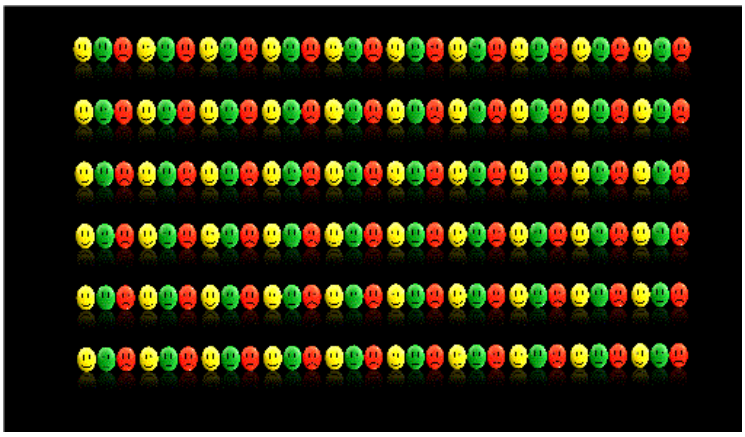
```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
```

للتكرار على محور Y:

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
```

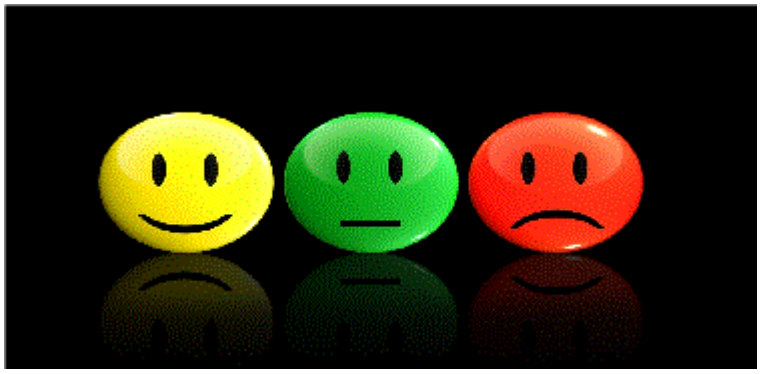


```
glBegin(GL_QUADS);
glColor3f(1,1,1);
glNormal3f(0,0,1);
glTexCoord2f(0.0f, 0.0f);
glVertex3f(-5.0f,0.0f,0.0f);
glTexCoord2f(0.0f, 6.0f);
glVertex3f(-5.0f,6.0f,0.0f);
glTexCoord2f(10.0f, 6.0f);
glVertex3f(5.0f,6.0f,0.0f);
glTexCoord2f(10.0f, 0.0f);
glVertex3f(5.0f,0.0f,0.0f);
glEnd();
```



- من دون استخدام glTexParameteri داخل تابع ال InitGL:

```
glBegin(GL_QUADS);
glColor3f(1,1,1);
glNormal3f(0,0,1);
glTexCoord2f(0.0f, 0.0f);
glVertex3f(-5.0f,0.0f,0.0f);
glTexCoord2f(0.0f, 1.0f);
glVertex3f(-5.0f,6.0f,0.0f);
glTexCoord2f(1.0f, 1.0f);
glVertex3f(5.0f,6.0f,0.0f);
glTexCoord2f(1.0f, 0.0f);
glVertex3f(5.0f,0.0f,0.0f);
glEnd();
```



- ملاحظات :

- في حال وجود مشاكل في تضمين ملفات Texture اكتب هذه التعليمة في main.cpp :

#define_CRT_SECURE_NO_DEPRECATE

واضبط إعدادات ال project:

Project ⟶ properties ⟶ c/c++ ⟶ preprocessor ⟶ preprocessor definition

ضع ضمنه:CRT_SECURE_NO_WARNINGS

- لرسم شكل مع Texture فمن المستحسن تفعيل الـ Texture قبل رسم الشكل وتعطيلها بعد رسمه فورًا حتى لا تتأثر الأشياء التي ترسم بعده.

الصورة المستخدمة يجب أن تكون (RGB) وبلاحقة BMP ويجب أن تكون أبعاد الصورة من قوى العدد 2 ومجموعها مثال: 256*384 , 128*256 , 64*64 , 128*128

لاحظ أن 348 ليس من قوى العدد 2 لكنه مجموع 128+256.

3D Models

الـModel : هو شكل ثلاثي الأبعاد يمكن تضمينه ضمن البرنامج عوضاً عن الرسم يدوياً ومن نقطة البداية، وتكون هذه الـModels مبينة باستخدام برامج مختصة مثل Autodesk.

سنتعلم كيفية استخدام model من النمط 3ds وسنحتاج لتضمين "Model_3DS.h" في بداية البرنامج.

يمكننا تحميل Models واستخدامها في أي مشهد كما يمكن التعديل عليها باستخدام البرامج المخصصة لذلك.

خطوات إضافة واستخدام model ما:

1. إضافة مكتبات ال Models إلى Project ومع إضافة ملفات ذات اللاحقة .h إلى:

C:\Program Files (x86) \Microsoft Visual Studio 14.0\VC\include

2. في ملف الـ Source الخاص بالـ project نكتب:

```
#include "Model_3DS.h"
#include "GLTexture.h"
```

3. لتخزين الـModel نحتاج إلى تعريف نوع معطيات يسمى *Model_3DS وهو مؤشر لـModel ثلاثي الأبعاد ونقوم بتعريفه قبل التابع () InitGL حتى نتمكن من استخدامه ضمنه، فـ على سبيل المثال:

```
Model_3DS *tree;
```

4. ضمن التابع () InitGL نضمّن الـmodel الذي يجب أن يكون موجوداً ضمن ملفات المشروع على الشكل الآتي:

```
tree = new Model_3DS ();
tree->Load("Tree.3DS");
```

حيث أن "Tree.3DS" هو Model ضمن ملفات المشروع.

5. نقوم برسم الشكل ضمن التابع () DrawGLScene كالآتي:

```
tree->draw ();
```

ويمكن استخدام مجموعة من خواص الـmodel للتحكم بحجمه وموقعه وباقي خصائصه مثل:

```
tree->pos.x=0;
tree->pos.y=0;
tree->pos.z=0;
tree->scale=0.1;
```

6. في حال حاولنا تشغيل البرنامج الآن سيظهر لنا الشكل في حال كان مكون من قطعة واحدة أما في حال كان مكوناً من أكثر من قسم فسيظهر باللون الأبيض وذلك لأننا لم نقم بربطه بالـtexture المناسبة له والتي تكون جاهزة وموجودة مع ملفات الـmodel عند تحميله من المواقع المختصة به، وللقيام بذلك يجب أن نعرف ضمن () InitGL ما يلي:

```
GLTEXTURE Bark, Leaf
Leaf.LoadBMP("leaf.bmp);
Bark.LoadBMP("Bark.bmp);
```

حيث أن كلاً من الـTextures السابقة هي ملف بصيغة bmp ويخضع لنفس قواعد الـtexture المذكورة سابقاً وموجود ضمن ملفات المشروع.

7. بعد أن قمنا بتحميل الـtextures يجب أن نربط كل منها بالـmodel ضمن تابع () InitGL على الشكل الآتي:

```
tree->Materials[0].tex = Bark;
tree->Materials[1].tex=Leaf;
```

حيث أن Materials هي مصفوفة عناصر الـModel .

بعد تطبيق كافة الخطوات السابقة أصبح الـmodel جاهزاً وسيظهر لنا الشكل عند تشغيل البرنامج

ملاحظات:

- في حال واجهنا مشاكل أثناء التشغيل متعلقة بالCRT نضيف
  CRT_NONSTDC_NO_DEPRECATE CRT_SECURE_NO_WARNINGS
  إلى الخانة preprocessor Definition والتي يمكن الوصول إليها باتباع المسار:
  Project -> Properties -> C/C++ -> preprocessor -> preprocessor Definition
- يوجد العديد من المواقع لتحميل Models جاهزة أهمها:
  blender.com free3d.com/3d-models sketchfab.com archive3d.net

انتهت المحاضرة