

Computergraphik II

Prüfungsvorleistung 01

Geometrie Daten

Abgabetermin: 17.04.2019 11:00 Uhr





Die Aufgaben sind von jedem Teilnehmer eigenständig zu bearbeiten, Gruppenarbeit ist nicht zulässig!
Möglichkeiten zur Abgabe:

- elektronisch über das OPAL-Portal <http://opal.sachsen.de/TUC>
Eine Einschreibung in die Lerngruppe der Lernressource „571110 Computergraphik II SS 2019“ ist zwingend erforderlich. Falls Sie mehrere Versionen hochladen, beachten wir nur die aktuellste Abgabe. Sollten Sie Fragen haben können Sie sich gerne per Mail (timon.zietlow@informatik.tu-chemnitz.de) an mich wenden.

Das Ziel dieser ersten PVL ist es, dass Sie sich selbstständig weiter mit dem gelernten befassen und dabei das Framework, welches in den kommenden Übungseinheiten verwendet wird, mit entwickeln.

Bisher haben wir die Geometrieinformationen in 'losen' Buffern und Vertex Array Objekten verwaltet. In komplexeren Anwendungen ist dies nicht wirklich praktikabel. In dieser PVL sollen Sie ein Werkzeug erstellen, mit dem Sie Vertexdaten aus dem Dateisystem in die entsprechenden Buffer laden können, um diese dann in einer Szene organisiert rendern zu können.

Tastaturbelegung:

-  : Programm beenden
-  /  +  : Wireframedarstellung an/aus

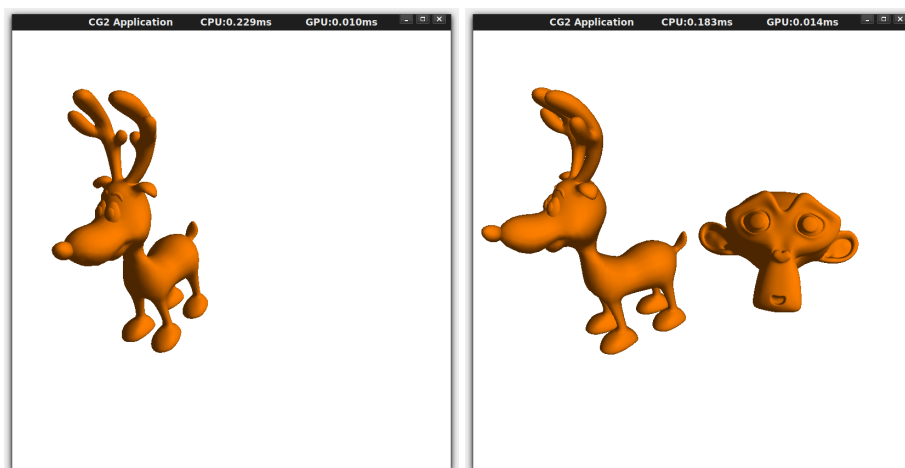


Abbildung 1: Links: Das erste sichtbare Zwischenergebnis. Rechts: Das Resultat, nachdem die ganze PVL bearbeitet wurde. 第一个可见的中间结果。 右：整个PVL处理完毕后的结果。

1. Die CG2Geometry Klasse (5 Punkte)

类CG2Geometry的对象表示顶点数组对象以及存储顶点和索引数据的相应缓冲区。它还存储drawcall所需的信息：基本类型，索引数量及其数据类型。

Ein Objekt der Klasse `CG2Geometry` repräsentiert ein Vertex Array Object zusammen mit den entsprechenden Buffern, in denen die Vertex- und Indexdaten gespeichert sind. Außerdem speichert es die, für den Drawcall benötigten, Informationen: Primitivtyp, Anzahl der Indices und deren Datentyp.

为了快速加载几何信息，我们使用专为计算机图形2开发的 .cg2vd 格式。此格式捕获练习期间我们将需要的所有数据并将其存储，以便可以将它们直接加载到适当的缓冲区中。为此，存储了三个数据块：

Um die Geometrieinformationen schnell laden zu können verwenden wir das, extra für die Computergraphik2 entwickelte, .cg2vd-Format. Dieses Format fasst alle Daten, die wir im Laufe der Übungen benötigen werden, und speichert diese so, dass sie direkt in entsprechende Buffer geladen werden können. Um dies zu erreichen werden drei Datenblöcke gespeichert:

包含渲染对象所需的信息和有关顶点数据结构的信息（属性的格式，偏移和跨参数等）

Metadaten beinhalten die Informationen, die benötigt werden um das Objekt zu rendern und die Informationen über die Struktur der Vertexdaten (Format der Attribute, offset und stride Parameter, etc.)

Indexdaten sind im wesentlichen ein Array vom Typ `GLushort` oder `GLuint`. 本质上是GLushort或GLuint类型的数组。

Vertexdaten sind die eigentlichen Vertexdaten, deren Struktur in den Metadaten beschrieben wurde. 是实际的顶点数据，其结构在元数据中描述。

要加载和编写 .cg2vd 格式，可以在 `vd / vd.h` 中使用类 `CG2VertexData`。

Zum Laden und Schreiben des .cg2vd-Formats steht in `vd/vd.h` die Klasse `CG2VertexData` bereit.

- Öffnen Sie die Datei `vd/vd.h` und machen Sie sich mit den Klassen `CG2VertexData`, `CG2VDMeta` und `CG2VDAAttribute` vertraut. 打开 `vd / vd.h` 文件，熟悉 `CG2VertexData`, `CG2VDMeta` 和 `CG2VDAAttribute` 类。

- Implementieren Sie die Methode `void CG2Geometry::load(const std::string &path)` in `geometry.cpp`. Sorgen Sie dafür, dass ...

... die Daten aus der Datei `parameterpath` in ein `CG2VertexData` Objekt gelesen werden. 参数路径文件中的数据被读入CG2VertexData对象。

... das Vertex Array Object `vao` erstellt und gebunden wird. 创建并绑定顶点数组Objectvao。

... die entsprechenden Buffer `vbo` (für die Vertexdaten) und `ibo` (für die Indexdaten) erstellt werden und die Daten übertragen werden. 创建相应的缓冲区vbo（用于顶点数据）和undibo（用于索引数据）并传输数据。

... die entsprechenden Vertex Attribute Pointer gesetzt und eingeschaltet werden. 设置并打开相应的顶点属性指针。

确保缓冲区 `ibo` 必须绑定到 `GL_ELEMENT_ARRAY_BUFFER`！最后，从 `CG2Geometry` 对象的相应属性中的 `CG2VertexData` 对象中保存渲染所需的信息（索引的数量，基元类型和索引类型）。

Achten Sie darauf, dass der Buffer `ibo` an `GL_ELEMENT_ARRAY_BUFFER` gebunden werden muss! Speichern Sie zuletzt die, für das Rendering benötigten Informationen (Anzahl der Indices, Primitivtyp und Indextyp) aus dem `CG2VertexData`-Objekt in den entsprechenden Attributen des `CG2Geometry` Objekts.

- Implementieren sie die Methode `void CG2Geometry::render()` so, dass das `vao` gebunden wird, und der entsprechende Drawcall (mittels `glDrawElements`) ausgeführt wird. 实现 `void CG2Geometry::render()` 方法，以便绑定 `vao` 并执行相应的 drawcall（使用 `glDrawElements`）。

- In dem Verzeichnis `data/models/` befindet sich noch ein zweites Modell: `suzanne.cg2vd`. 在目录 `data / models /` 中有第二个模型： `suzanne.cg2vd`。

- Erweitern Sie die Klasse `CG2App` so, dass ein weiteres `CG2Geometry` Objekt gespeichert werden kann. 扩展 `CG2App` 类，以便可以保存另一个 `CG2Geometry` 对象。

- Laden Sie `data/models/suzanne.cg2vd` in dieses Objekt. (Orientieren Sie sich dabei an dem Code, der zum Laden des Rentiers in `void CG2App::init_gl_state()` bereits gegeben ist.) 将 `data / models / suzanne.cg2vd` 加载到此对象中。（使用已经提供的代码在 `void CG2App::init_gl_state()` 中加载驯鹿。）

- Erweitern Sie die Methode `void CG2App::render_one_frame()` so, dass neben dem Rentier auch das Suzanne-Objekt gerendert wird. 扩展 `void CG2App::render_one_frame()` 方法以呈现驯鹿和Suzanne对象。

- Da die `CG2Geometry`-Objekte OpenGL-Objekte verwalten, kommt es bei dem Beenden des Programms zum Absturz, da zu dem Zeitpunkt des Aufrufs der entsprechenden Destruktoren bereits kein OpenGL-Context mehr verfügbar ist. Um die OpenGL-Objekte dennoch löschen zu können verfügt die `CG2Geometry` über die Methode `CG2Geometry::destroyGLObjects()`, welche die OpenGL-Objekte zerstört. Rufen Sie diese Methode an geeigneter Stelle (`void CG2App::clean_up_gl_state()`) für ihr Suzanne-Objekt auf.

由于 `CG2Geometry` 对象管理 OpenGL 对象，程序崩溃是因为在调用相应的析构函数时 OpenGL 上下文不再可用。但是为了能够删除 OpenGL 对象，`CG2Geometry` 的方法是 `CG2Geometry::destroyGLObjects()`，它会破坏 OpenGL 对象。在 `Suzanne` 对象的适当位置 (`void CG2App::clean_up_gl_state()`) 调用此方法。