作业 0: 虚拟机的使用
GAMES101, 2020 年春季

教授: 闫令琪

计算机图形学与混合现实研讨会

GAMES: Graphics And Mixed Environment Seminar

截止时间为北京时间 2020 年 2 月 25 日 (星期二) 上午 10:00

注意:

- 任何更新或更正都将发布在论坛上，因此请偶尔检查一下。
- 论坛链接`http://games-cn.org/forums/forum/graphics-intro/`
- 你必须独立完成自己的作业。
- 你可以在论坛上发布帖子求助，但是发布问题之前，请仔细阅读本文档。
- 在截止时间之前将你的作业提交到 SmartChair 上。

---

1 虚拟机的使用

为了免去配置作业所需环境的麻烦，本次课程使用虚拟机，学生在虚拟机内编写，编译和运行代码。我们提供的文件为虚拟硬盘文件，使用虚拟机挂载该文件后，就可以保证所有人的环境是统一并且完善的，不需要再手动配置环境。在安装完虚拟机后，我们需要手动安装 Guest Additions 来增强虚拟机的功能。

1.1 安装虚拟机

这里我们使用 **Oracle VM VirtualBox** 虚拟机。

如果你使用 Windows 系统，你可以直接下载`https://download.virtualbox.org/virtualbox/6.1.4/VirtualBox-6.1.4-136177-Win.exe`，下载完成后按照指示完成安装。

如果你使用 Mac OS 系统，你可以直接下载`https://download.virtualbox.org/virtualbox/6.1.4/VirtualBox-6.1.4-136177-OSX.dmg`，下载完成后按照指示完成安装。

如果你使用 Linux 内核的系统，你可以查看`https://www.virtualbox.org/wiki/Linux_Downloads`，找到你使用的系统，按照对应的指示完成安装。

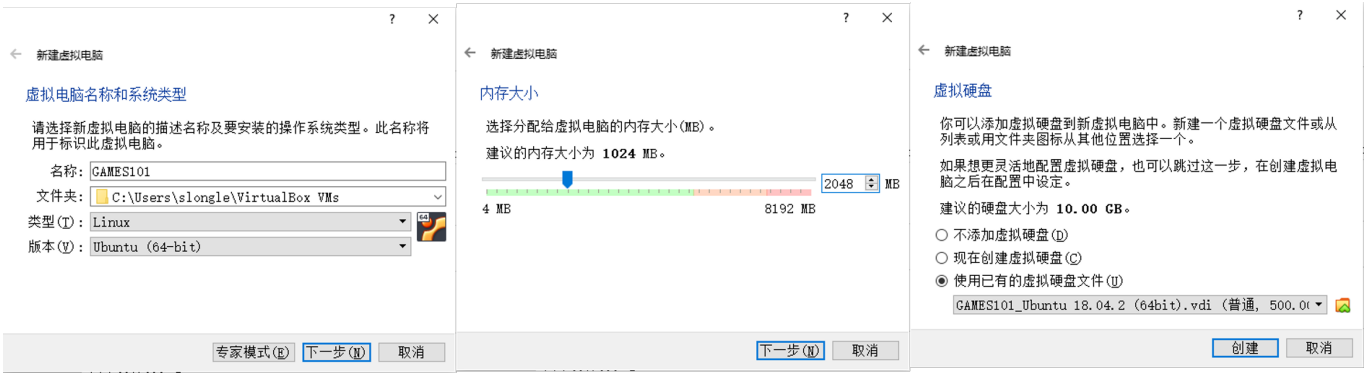
1.2 下载虚拟硬盘

虚拟硬盘文件的下载地址为 `https://cloud.tsinghua.edu.cn/f/103133da1bf8451b8ba6`，密码为 games101。下载完成后得到 **GAMES101_Ubuntu 18.04.2 (64bit).rar**，将其解压后得到虚拟硬盘文件 **GAMES101_Ubuntu 18.04.2 (64bit).vdi**。

1.3 配置虚拟机

打开 **Virtual Box**，点击新建，设置类型为 **Linux**，版本为 **Ubuntu-64 bit**，建议设置虚拟机的内存大小为 **2GB**，然后选择**使用已有的虚拟硬盘文件**，设置为

之前解压得到的 **GAMES101_Ubuntu 18.04.2 (64bit).vdi**，最后点击创建就完成了虚拟机的配置工作。



之后就可以使用创建好的虚拟机了，选中刚刚创建好的虚拟机，点击右侧上方的启动按钮就可以打开虚拟机了，Ubuntu 系统的密码为 **Ilovegraphics**。

1.4 安装 Guest Additions

进入系统后，点击上方菜单的**设备**，点击**安装增强功能**，如下图所示。安装完成后，重启虚拟机系统就完成了 Guest Additions 的安装。

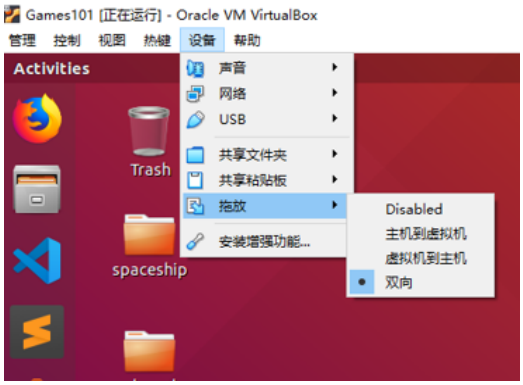
如果上面的方法安装失败了，可以使用 ctrl+alt+t 调出终端，使用如下命令安装 Guest Additions 功能。

```
sudo mkdir –p /media/cdrom
sudo mount -t auto /dev/cdrom /media/cdrom/
cd /media/cdrom/
sudo sh VBoxLinuxAdditions.run
```

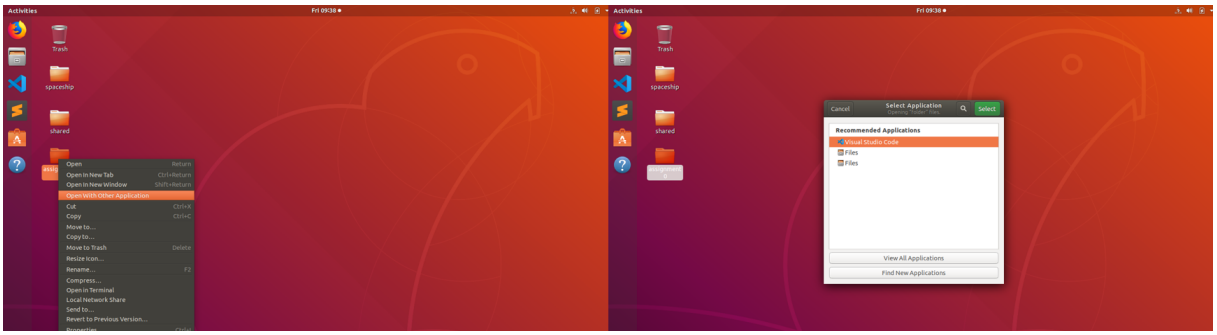
执行完毕后，重启虚拟机系统就完成了 Guest Additions 的安装。

1.5 作业框架的传输及编辑

作业框架的导入和导出有很多种方式，这里只提一种。当你在你的主机上下载好作业框架后，直接将其拖进虚拟机系统里。这里需要开启 Virtual Box 的拖放功能：进入虚拟机系统后，点击上方菜单的**设备**，将**拖放**功能设置为**双向**即可。



导入作业框架后，可以使用 Visual Studio Code 来查看和编辑。右键作业框架的文件夹，选择使用其他应用来打开，选择 Visual Studio Code，具体如下图所示



2 作业框架说明

本部分将体现在样例程序 main.cpp 中。

2.1 开发工具说明

虚拟机中已经自带 Visual Studio Code 与 Sublime 作为文本编辑器，**课程推荐使用 VSCode 编辑代码，并且在命令行中编译、运行程序**。

将框架拷贝到虚拟机中，打开 VSCode，选择 File->Open Folder 找到目标文件夹，选择打开即可。

2.2 C++ 语言使用注意事项

本部分只会简单介绍一些作业涉及的 C++ 语法内容，更多的 C++ 知识请通过 `https://devdocs.io/cpp/` 或者 Stack Overflow 查阅。

2.2.1 头文件

在 C 语言和 C++ 中，头文件或包含文件是一个文件，通常是源代码的形式，由编译器在处理另一个源文件的时候自动包含进来。一般来说，程序员通过编译器指令将头文件包含进其他源文件的开始（或头部）。

实践中，一般通过头文件来引入是现在其他文件中的函数模块。

```
#include<cmath>
#include<iostream>
```

样例程序中的上述头文件引入了 C++ 中输入、输出、数学计算所需要的必需模块。

2.2.2 函数体

函数体是程序进行指定目标的运算的模块，其中以 main 命名的函数被称为主函数，是程序运行的入口。

```
int main(){
    float a = 1.0, b = 2.0;
    std::cout<< a <<std::endl;
    std::cout<< a/b <<std::endl;
    std::cout<< std::sqrt(a) <<std::endl;
    std::cout << std::acos(-1) << std::endl;
    std::cout << std::sin(30.0/180.0*acos(-1)) << std::endl;
    return 0;
```

```
}
```

上述代码的作用是在三行分别输出 a, $\frac{a}{b}$, $\sqrt{a}$, $\arccos(-1)$, $\sin(30)$ 的计算结果，并安全退出程序。**请观察输出，并尝试解释这些结果（提示：C++ 三角函数运算使用弧度制）**。

2.2.3 C++ 常见错误指南

1. Compile Error 编译错误: 认真阅读编译器给出的报错信息，找到报错位置修改代码；如果无法自己解决，建议将报错信息拷贝到 Stack Overflow 查找类似情况。
2. undefined reference to xxx: 一般是链接错误，检查 CMakeLists.txt 中是否包括了需要引入的模块。
3. Segmentation Fault: 段错误，一般是数组越界、栈空间开销过大等问题导致。
4. Bus Error: 总线错误，成因一般与段错误相似。
5. Math Error: 一般是除数为 0 导致。

2.3 Eigen 库使用注意事项

Eigen 是本课程使用的线性代数运算库，官方文档为 `http://eigen.tuxfamily.org`.

2.3.1 头文件

如样例程序 main.cpp 所示，eigen 需要额外引入头文件 <eigen3/Eigen/Core>.

```
#include<eigen3/Eigen/Core>
```

2.3.2 向量、矩阵

关于本部分内容，请详细阅读官方文档的矩阵部分`https://eigen.tuxfamily.org/dox/group__TutorialMatrixArithmetic.html`以获得更全面、清晰的理解。

```cpp
// Example of vector
std::cout << "Example of vector \n";
// vector definition
Eigen::Vector3f v(1.0f,2.0f,3.0f);
Eigen::Vector3f w(1.0f,0.0f,0.0f);
// vector output
std::cout << "Example of output \n";
std::cout << v << std::endl;
// vector add
std::cout << "Example of add \n";
std::cout << v + w << std::endl;
// vector scalar multiply
std::cout << "Example of scalar multiply \n";
std::cout << v * 3.0f << std::endl;
std::cout << 2.0f * v << std::endl;
```

上述关于 Vector 的使用样例展示了如何定义一个三维浮点向量并且进行输出、加减、数乘，**请自行根据数乘的形式与向量点积的形式探索点积的用法**。

```cpp
// Example of matrix
std::cout << "Example of matrix \n";
// matrix definition
Eigen::Matrix3f i,j;
i << 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0;
j << 2.0, 3.0, 1.0, 4.0, 6.0, 5.0, 9.0, 7.0, 8.0;
// matrix output
std::cout << "Example of output \n";
std::cout << i << std::endl;
// matrix add i + j
// matrix scalar multiply i * 2.0
// matrix multiply i * j
// matrix multiply vector i * v
```

上述关于 Matrix 的使用样例展示了如何定一个三维浮点矩阵进行输出，**请自行根据注释与 Vector 部分的经验探索矩阵加减、数乘、矩阵乘法、矩阵乘向量的用法**。

3 作业描述与提交

3.1 作业描述

给定一个点 P=**(2,1)**, 将该点绕原点先逆时针旋转 45°, 再平移 (1,2), 计算出变换后点的坐标（要求用齐次坐标进行计算）。

3.2 编译

为方便之后的作业编写，本次作业要求使用 **cmake** 进行编译。
首先，编写好本次作业的程序 **main.cpp**。
然后, 在 **main.cpp** 所在目录下，打开终端 (命令行)，依次输入：

- **mkdir build**: 创建名为 build 的文件夹。
- **cd build**: 移动到 build 文件夹下。
- **cmake ..**: 注意其中'..' 表示上一级目录，若为'.' 则表示当前目录。
- **make**: 编译程序，错误提示会显示在终端中。
- **./Transformation**: 若上一步无错误, 则可运行程序 (这里的 Transformation 为可执行文件名，可参照 CMakeLists.txt 中修改)。

3.3 提交

作业提交使用的平台为 **Smartchair** 平台，地址为`http://www.smartchair.org/GAMES2020Course-YLQ`, 平台的具体操作说明请在`http://games-cn.org/submit_homework/`下载。

3.4 评分

由于本次作业主要目的是让同学们熟悉虚拟机的使用、使用 C++ 与 **Eigen** 库编写简单的程序和用 **cmake** 进行编译, 所以本次作业不进行评分。同学们只需将编写好的程序打包提交即可, 顺便可以熟悉 **Smartchair** 平台提交作业的流程。