数据结构期末 Project

2022 年 11 月 28 日

摘要

截止时间 2022 年 12 月 25 日 23：59 PJ 的目标是写一个简单的 BVH 文件解析器，BVH 文件一般是用来描述一个虚拟角色动作的文本数据，其中包含了角色每个关节的位置和朝向，同时通过大括号来表明关节之间的树状关系，具体结构请参见链接。可以忽略链接关于运动学的介绍，只关注关于 BVH 文件结构的描述即可。

1 Project 要求

编写的解析器需要读入一个 bvh 文件，并输出一个 json 格式的文件，json 以树形结构递归描述骨骼结构，包含以下信息

- `frame`: motion 数据的总帧数，整数
- `frame_time`: 单帧时间，浮点数
- `joint`: 从根关节开始的递归的关节信息，每个关节信息包含以下内容
  - `type`: 关节类型，字符串，可能是 `ROOT`,`JOINT` 和 `End`
  - `name`: 关节名，字符串，如果这个关节的类型为 End，则其 name 字段应为父关节名 +"_End"
  - `offset`: 一个长为 3 的浮点数数组，表示该关节相对父关节的偏移
  - `channels`: 表示在 motion 数据中该关节包含的通道，字符串数组
  - `motion`: 一个二维嵌套数组，数组长度为总帧数，每个元素是一个长度和通道数一致的数组，为这个关节在相应帧每个通道的值，若关节通道数为 0，则每个元素都是空数组，实际上在检查时，不会访问 End 关节的 motion 数据，故 End 关节该字段为空不会影响结果。
  - `children`: 数组，数组元素嵌套一个 joint，递归表示这个关节的子关节信息，End 关节该字段应为一个空数组。

示例 bvh 文件见 `sample.bvh` 和 `output.json`

2 代码结构

需要编写两个文件，分别为解析器文件 `bvh_parser.cpp` 和输出 json 的文件 `json.cpp`，具体需要使用的结构可参考 `bvh_parser.h` 中的定义。可以仅完成 bvh_parser 部分以获得基本分数，即只需要读取 bvh 文件的数据并以 `bvh_parser.h` 中定义的 `joint` 结构存储，助教提供了已经编译好的 `json.o` 文件，可以在编译时直接通过链接该文件以使程序具有完整的功能。

3 编译运行方式

助教提供了一个简易的代码框架，仅完成 base 部分的同学需要在该框架下编写代码，仅可通过修改 `bvh_parser.cpp` 并保持其他文件不变来完成 PJ；完成完整 bonus 内容的同学可以自行修改各个文件内容或添加文件，具体要求参见提交要求。

接下来会简单描述如何在 Windows 下通过 WSL 搭建一个可以顺利运行代码框架的环境。在评分时也会在这个环境下编译大家提交的代码。

1. 通过 Windows Store 安装 WSL，建议使用 `wsl --install -d Ubuntu` 安装乌班图发行版(参考文档)。如果安装过程出现网络问题，可以尝试更换网络环境。
2. 安装完成后设置好用户名密码，请记住此时设置的密码，并请不要使用中文。
3. 执行以下命令，在需要输入密码的时候输入上面设置好的密码，输入时不会有明文字符或者星号显示，不必担心。
    1. `sudo apt update`
    2. `sudo apt install build-essential`
4. 安装完成后，输入 `gcc -v`，会有一段输出，结尾类似

    `gcc version 9.4.0 (Ubuntu 9.4.0-1ubuntu1~20.04.1)`,

    此时环境搭建完成。
5. WSL 已经对硬盘做好挂载，比如代码文件在 D 盘的 work 文件夹下，直接执行 `cd /mnt/d/work` 即可切换到对应文件夹，可以直接使用 VSCode 等文本编辑器直接在 Windows 下编程，在 WSL 的命令行编译运行即可。也可了解一下 VSCode 的 remote 套件（可以自行在搜索引擎搜索 `vscode remote wsl -csdn` 来查询教程）。

助教已经提前在该环境下将编写完成的 `json.cpp` 编译为 `json.o` 文件，用于给完成 base 任务的同学直接进行链接。在编写完毕后可以直接使用命令行在当前目录下运行 `make base`，如果成功编译则可以获得一个文件名为 `base` 的可执行文件，然后执行命令`./base sample.bvh` 即可获得输出。如果不慎使用 `make clean` 清除了助教提供的 `json.o`，请自行复制一个 `json.o` 到当前文件夹。对于完成 bonus 完整任务的同学不做硬性要求，可以只修改 `bvh_parser.cpp` 和 `json.cpp` 两个文件，然后执行 `make bonus` 后获得可执行文件 `bonus`，然后执行命令`./bonus sample.bvh` 即可获得输出。也可以自行决定代码结构，但同样需要修改 `Makefile` 文件并留好 bonus 入口以说明自己代码的编译顺序。

可以在互联网上搜索其他 bvh 文件以检验自己代码的鲁棒性。

如果需要使用其他环境（如苹果系统）完成作业，请自行查询资料搭建编译套件并保证能按要求正确提交相应文件，建议使用虚拟机安装 Ubuntu 的形式，否则可能无法使用助教下发的 `json.o` 文件，不建议使用 M1 芯片的 MacBook 来完成作业。

4 提交要求

作业通过 elearning 进行提交，提交文件应为一个后缀名为 `rar` 或者 `zip` 的压缩包，文件名为学号，压缩包直接解压后应是以下结构

- student-id.zip
  - report.pdf/md
  - src
    - Makefile
    - *.cpp
    - *.h/hpp

请在报告开头注明完成的是 base 还是 bonus。

报告若只包含文本，可以提交 markdown 文件，也请不要通过 base64 编码等方式将图片嵌入件中，若需要包含图片，请直接渲染成 pdf 文件提交。报告长度不宜超过 10 页，正文字号行距等以清晰易读为佳，解释清楚完成 PJ 的基本思路即可，可以包含拿到 PJ 到形成最终代码的构思、字符串处理方案等；但搭建环境等方面的内容无需撰写相应报告，可以在报告最后简单描述一下自己在完成 PJ 过程中的收获（非必须）。另外，在完成 PJ 过程中若参考了互联网资料或获得了同学的帮助，请在报告的末尾注明，如**未注明**但在评分时发现有直接抄袭互联网资料或其他同学提交的代码等情况，**零分**处理。

如完成的是 pj 的 base 部分，则在 src 文件夹中只需要提交 `bvh_parser.cpp` 文件，评分时会将其放入包含 `bvh_parser.h`，`json.o` 和相应 `Makefile` 的文件夹中编译测试。如果完成的是 pj 的完整部分即 bonus 部分，则在 src 文件夹中需要包含完整的源码和 Makefile，并保证能在上述构建的环境中通过 `make bonus` 命令直接构建出可执行文件。在保证能直接构建，能通过命令行参数输入 bvh 文件名，并输出正确的 `output.json` 的前提下，头文件、cpp 源文件以及 `Makefile` 均可修改，但注意 `Makefile` 需要保留 bonus 入口且可执行文件名须为 bonus，无后缀。

5 评分标准

完成 parser 部分，即 base 任务，可以获得基础分数；完成 parser 和 json 输出的功能，即 bonus 任务，即可获得全部分数。仅提交代码或仅提交报告，以及提交的代码无法通过编译/不符合提交要求的，分数会有较大折扣。除此以外，考虑一些错误处理以及输出 json 的合理缩进会得到酌情加分，但在已经完成上述所有功能获得全部分数的情况下，不会有额外加分。

助教会提供一个在线的可视化工具，需要在校园网环境访问，各位可以访问网站上传输出的 json 文件查看火柴人运动结果。网站在上传文件后会立刻开始渲染动画，如两次上传同名文件，请在中间上传一次其他名字的文件以刷新缓存。另外，相机位置可能不合适导致可视化结果极小，请

自行尝试拖动鼠标、滑动滚轮来获得最佳观测角度，如果已经把摄像机拖到一个奇怪的位置，可以按 F12 后在控制台输出 `camera.position.z=1000` 来把摄像机复位到一个相对合理的位置。实际上给出的示例就是一个相对相机位置较小的小人，请同学们先上滚滚轮以获得一个拉近的视角。更新动画的逻辑是上传文件名发生改变，故如需要重新上传一个内容不同的同名文件，请先上传另一个任意不同名文件后再重新上传需要查看的文件。

对于提交代码存在雷同或大部分雷同的，双方均直接零分