

idea

项目名称

BrainDance

项目目的

利用3D泼溅技术打造一个面向未来的移动端三维数字记忆库，用于扫描构建存储即时的实体或想法以留下三维记忆或三维灵感

项目功能

1. 使用手机进行记录（使用倾斜摄影），将记录的数据上传到云端进行处理生成3dgs模型再返回到本地
2. 在移动端输入图片或者文字，上传到云端进行文生或图生3dgs再返回本地
3. 使用对3dgs模型进行操作（如删改等）
4. 分享功能（将难以分享的.ply模型变成mp4等常见流媒体，使用云端渲染服务器/云端算力集群进行渲染）
5. 检索功能/为3dgs打标（RAG）

项目想法来源

本项目的灵感源于科幻概念“数字化记忆”（如《赛博朋克2077》中的“超梦（BrainDance）”），旨在探索人类记忆存储的终极形态。

现在有很多的知识库，也有很多的记录工具存储工具。但是3d的记录始终是受众面较小的。在当下元宇宙概念的兴起与vr/ar设备的逐渐普及，三维记录是面向未来的记录方式，提供更强的“临场感”。

而高斯泼溅算法即使相比于传统倾斜摄影构建3d模型计算量小了非常多，但是依旧难以在手机这种低功耗设备上进行迭代计算。所以本项目希望使用移动端交互，云端进行计算与构建。

由于3dgs模型的文件大小较大，难以在移动端大量存储模型，得益于5G技术的普及、LightGaussian技术、LOD技术，手机使用WebGL查看也可以有不错的交互体验

现在记录/创作2d图片（拍摄/文生图）以及2d视频（拍摄/文生视频/图生视频）在移动端应用十分广泛，但是对于3d内容的移动端记录与创作却应用很少，本项目意在填补本方向的空白

功能细节

1. 主界面设计
 1. 应用底部有导航栏，依次为
 1. recall
 2. record
 3. generate
 2. 默认进入软件时的界面为recall（类似于传统图库）
2. recall栏

1. 类似传统图库，显示3dgs缩略图，缩略图左上角有标签
(record/generate/record+generate)
2. 点开3dgs模型进行查看类似传统图库
 1. 滑动调整视角查看
 2. VR 模式（利用陀螺仪数据）
3. 有编辑功能（其中存在3dgs模型本地缓存机制，从本地打开或从云端打开）
3. record栏
 1. 类似于相机界面，可以录制1min的视频或拍摄多张图片
 2. 存在引导（例如如何拍摄，需要什么样子光源）
 3. 拍摄完毕进行设置
 1. 可以设置迭代次数，使用的照片数量等
 4. 进行上传
 1. 上传时调用多模态模型，判断拍摄的是否适合上传，给用户提示
 1. 检测场景，例如是否存在阴影等
 2. 检测图片质量（例如是否在室内，噪点是否多，清晰度是否足够，覆盖率检测），是否适合用于3dgs
 3. 检测视频中内容，是否连贯
 5. 上传完毕后经过云端的数分钟至几十分钟的计算在传回给用户（提交任务 -> 轮询状态 -> 下载结果）
4. generate栏
 1. 用户可以选择创作模式
 1. 选择已用的ply模型进行二次创作
 2. 使用文生ply进行二次创作
 3. 使用图生ply进行二次创作

技术创新点

1. 利用移动端优势（普及性以及集成了大量的传感器），可以在低成本低设备要求下生成高质量3dgs
传统的3dgs使用图片或视频需要去推算摄像机位置（COLMAP），耗时且不稳定，对于生成的3dgs质量有很大影响。但是使用移动端的ARCore（Android端），可以将位姿矩阵和内参记录并一同上传都云端辅助计算
2. 空间RAG
 1. LERF (Language Embedded Radiance Fields) 的技术落地（当然实现路径不太同，有简化）
3. 移动端数据与传统COLMAP结合的混合管线：ARCore 的位姿（SLAM Pose）通常存在漂移（Drift）和尺度问题（Scale Ambiguity）
使用 ARCore 的位姿作为初始化猜测 (Initial Guess)，然后传给服务器端的 SfM 算法（如 COLMAP 或 hloc）进行快速微调

技术栈

1. 前端
 1. flutter
2. 后端

1. 业务后端

使用go语言。通过数据库实现用户数据（模型，账号，操作记录等）的存储，与前端进行链接

2. 计算后端

使用高性能服务器进行3dgs生成和渲染

技术细节

1. 前端

1. 前端的3dgs模型渲染

1. 利用 Flutter 的 `webview_flutter` 组件，加载 Web 端 3DGS 查看器

2. 仓库：

1. `mkkello/gaussianSplats3D`

2. `antimatter15/splat`

2. 前端照片视频录制

选择借鉴项目 `ar_flutter_plugin`

或：

1. 录制视频 + 记录 CSV/JSON：

使用 `SharedCamera` 接口。ARCore 允许通过 `SharedCamera` 接口获取 `Surface`

1. 流式传输

1. 使用gRPC传输

2. 使用Protobuf协议

2. 整体传输

2. 连续拍照

3. 前端三维信息记录

1. 使用 Flutter 的 `Platform Channels` 或 `PlatformView`，在 Android 原生层（Kotlin/Java）调用 ARCore SDK，然后将数据回传

4. 本地存储

5. “图库”界面

1. 展示缩略图（由服务端传回）

6. 与服务器数据交互

1. 短交互 (REST API): 登录、获取列表、搜索、修改标题

大文件上传 (Multipart/Chunk): 上传视频文件

长时任务状态 (Polling/WebSocket): 查询 3DGS 训练进度

7. 把 ARCore 的 quaternion/translation 转换成 3DGS 训练所需的 view matrix

(这一步如果计算量过大也可以交给计算后端)

1. 在移动端将 ARCore 数据导出为 Nerfstudio 兼容的 `transforms.json` 格式

将 ARCore 的 `C2W` 矩阵乘以一个“翻转矩阵”

2. 注意点：后端记录缩放比例，防止模型忽大忽小

8. 流式加载（类似LOD）与动画/特效（利用3dgs特性）

1. 使用Vertex Shader 篡改

9. 更多idea:

1. 引入时间轴与地点信息
2. AR模式
3. 软件对于VR/AR头显的支持
4. 隐私与加密

2. 业务后端

1. 消息队列进行异步处理

1. channel
2. Redis

3. 计算后端

1. 相机位置修正

1. 使用 ARCore 的位姿作为初始化猜测 (Initial Guess), 然后传给服务器端的 SfM 算法 (如 COLMAP 或 hloc) 进行快速微调

2. 运行高斯泼溅 (机器学习) 算法, 将视频+位姿转化为 .ply 模型

1. 参考: [nerfstudio-project/gsplat](https://nerfstudio-project.github.io/gsplat)

3. 预处理: 运行多模态 LLM 进行标签识别、质量打分

1. 接入Gemini或者阿里云

4. 渲染: 把 .ply 渲染成 .mp4

1. Rasterizer

5. RAG

ChromaDB+LangChain+OpenAI GPT-4o (Vision) API / Google Gemini Flash API

1. 使用多模态大模型在传入视频时打标
2. 利用llm传回的描述与图片本身使用CLIP向量化
3. 空间绑定 (直接选择上传的照片所对应的空间坐标可以用来回看)

6. 传回缩略图

1. 使用多模态llm实现判断所拍摄物体的正面

7. 压缩算法 (可选)

1. 格式优化
 1. .splat格式
2. 算法剪枝
 1. LightGaussian

8. 进行3dgs模型修改

1. 静态物体修改
 1. [buaacyw/GaussianEditor](#)
2. 风格修改
 1. [ayaanzhaque/instruct-gs2gs](#)

9. 进行文生.ply

1. 使用nano banana等模型生成图片
2. 使用3DTopia/LGM图生ply

10. 进行图生.ply

1. 3DTopia/LGM

11. 使用动态物体移除，优化模型质量

1. Robust-Gaussian-Splatting

2. nerfstudio-project/nerfstudio

12. 更多idea

1. 引入状态机：使用 Temporal 或 Celery + Redis

2. 热存储与冷存储

3. 对于3DRAG：过期机制的引用，权重相关，根据 meta info 分类后再做回归