Manual de instalación y uso de gsplat (splatfacto-3DGS) y compatibilidad con VolingaVR y UE5

Seguimos estos manuales, pero con ligeras modificaciones:
https://docs.nerf.studio/quickstart/installation.html

Crear un entorno conda para gsplat con python>=3.9

conda create --name gsplat -y python=3.9
conda activate gsplat
python -m pip install --upgrade pip

Instalar dependencias

pip uninstall torch torchvision functorch tinycudann

pip install torch==2.1.2+cu118 torchvision==0.16.2+cu118 --extra-index-url https://download.pytorch.org/whl/cu118

conda install -c "nvidia/label/cuda-11.8.0" cuda-toolkit

pip install ninja git+https://github.com/NVlabs/tiny-cuda-nn/#subdirectory=bindings/torch

Instalar RUST:

https://rustup.rs/ :
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh

https://www.rust-lang.org/tools/install :
sudo apt install rustup

Abrir nueva consola y chequear que rust está instalado correctamente:

rustc --version

Instalación de gsplat y NerfStudio

```
pip install gsplat
```

pip install nerfstudio

Reboot para aplicar correctamente todos los cambios.

Uso de gsplat y NerfStudio

Recordamos activar primero el entorno de conda:

conda activate gsplat

Para entrenar la red con nuestro input procedente de COLMAP lanzar el siguiente comando donde {processed_data} es la ruta a la carpeta donde se ha generado la fotogrametría de COLMAP:

ns-train splatfacto –data {processed_data}

Por si no hemos obtenido aún las poses de la cámara a partir de la fotogrametría de COLMAP recordamos como se genera: https://docs.nerf.studio/quickstart/custom_dataset.html

ns-process-data {video,images,polycam,record3d} --data {DATA_PATH} --output-dir {PROCESSED_DATA_DIR} (--num-frames-target {3*second_of_video})

--num-frames-target es opcional, pero si se deja sin especificar el máximo de frames que se obtendrá será en torno a 300. Si el valor de 3 fotogramas por cada segundo de vídeo es mayor que 300 se recomienda especificar el valor aproximado a COLMAP para obtener mejores resutlados, aunque llevará más tiempo su procesado.

También se pueden especificar otras herramientas diferentes a COLMAP. Se recomienda hacer “ns-process-data" seguido del tipo de input y “-h” para que se muestre la explicación completa de todos los comandos disponibles. Por ejemplo:

ns-process-data video –h

Nota: para implementar otros métodos es necesaria su instalación. Por ejemplo, hloc se instala siguiendo la guía disponible en su repositorio de GitHub: https://github.com/cvg/Hierarchical-Localization

Después ya se puede entrenar la red con Gaussian Splatting para generar el modelo 3D:

ns-train splatfacto –data {output_dir_de_ns-process-data}

Y después para exportar el resultado del entrenamiento:

ns-export gaussian-splat --load-config {splatfacto_output_dir/config.yml} --output-dir {my_ply_output_dir_para_el_.ply_file}

Compatibilidad con Unreal Engine

El archivo .ply que se ha obtenido de la exportación de Gaussian Splatting debe procesarse usando bien la aplicación de escritorio de Volinga Suite o bien haciendo uso de su aplicación web (https://volinga.ai/). No es necesario el uso de la extensión volinga-model ya que la compatibilidad entre el archivo .ply de Gaussian Splatting y Volinga es directa. Una vez procesado, el archivo .nvol es compatible con el plugin de volinga tanto en su versión estándar como en su versión compatible con VR.

Trubleshooting

Si se detecta cualquier problema de incompatibilidad o fallos en la configuración, se recomienda hacer una instalación completa de CUDA 11.8 detallada en el manual: “Manual instalación y uso de NVIDIA instant-ngp en Ubuntu 18.04”