15-418 Project Proposal - 3D Image Reconstruction

Matthew Yu (myu2), Emily Ho (emilyho)

November 2025

1 URL

https://github.com/EH67/3D-Image-Reconstruction-15618-Project

2 Summary

We are going to implement an optimized version of the 3D image reconstruction algorithm on NVIDIA GPUs that takes in multiple 2D images as input, and use the image correspondence pipeline + triangulation technique to transform the image into 3D.

3 Background

Simple 2D images lack depth, which is a key problem to solve when creating agents that aim to interact with the real world. Using multiple 2D images to generate a 3D environment solves this problem. One of the main issues in implementing this approach is the large amount of compute time it takes. Our approach to this project is to parallelize the reconstruction pipeline so that the construction of the 3D space can happen in realtime.

4 The Challenge

While the CV pipeline for 3D reconstruction is well researched and many implementations exist online, optimizing the program for GPUs presets some challenges. In a realtime video, there are 60 frames per second, and if there are thousands of features per frame, this leads to a large number of features that need to be processed in total. From there, feature matching requires matching features from one frame to the next, which requires a lot of compute time unless it is optimized. The number of features per frame can also differ, with one frame having 50 features while another having 2000, which can cause many threads to sit idle if a frame has very little features.

5 Resources

We will be running our CUDA implementation on the GHC machines. We plan to read the paper https://arxiv.org/pdf/2104.00681 to determine how they render the reconstructed image (but we are not implementing the same algorithm as them). In terms of the algorithm, we plan to reference slides and assignments from the Computer Vision course here at CMU.

6 Goals And Deliverables

6.1 Plan to Achieve

1. Have a working, correct implementation (one for serial, one for GPU) of 3D reconstruction from a set of images of the same scene (but different angles)

2. Achieve significant speedup in the GPU version compared to the serial version, and utilize data parallelism (and potentially task parallelism)

3. Add the ability to render the 3D image (either as a plot or in a 3D environment) upon reconstruction

6.2 Hope to Achieve

1. Reconstruct & render from a video rather than a set of images - this is achievable if we take snapshots of frames within the video and run the pipeline we have already implemented to reconstruct.

2. Add the ability to connect to a camera and reconstruct in real time as the user adjusts the camera angle - this would take a lot more effort than the first one, but would be possible if we could add frames into the pipeline as we perform the reconstruction instead of having a list of frames/images that we use for reconstruction prior to the pipeline.

7 Platform

We chose to use the GHC machines as this project relies on many embarrassingly parallel operations, which GPUs are designed to be efficient at doing. There is not much communication between threads that would nessecitate CPU like processers.

8 Schedule

1. 11/17-11/23: Design and implement a serial version of the pipeline that can take in sets of images and output a 3D reconstruction of the image. Additionnally, we will implement a program to take the reconstruction and render it.

2. 11/24-11/30: Implement a working GPU implementation that successfully reconstructs the 3D points, comparing output to the serial version for correctness

3. 12/1-12/8: Optimize the GPU implementation to compute 3D points faster. Work on implementing ”hope to achieve” goals like computing points from a video or computing points in real time.