3D Reconstruction using COLMAP
Project Report

Team Darkstar

Ranaraja S.V. 200506K
Kavinda W.M.C. 200301D

GitHub link: https://github.com/sithija-vihanga/3D-Recontruction-using-colmap

1. Problem Outline

3D reconstruction from 2D images is a fundamental challenge in computer vision, involving the conversion of 2D images into a precise 3D representation of a scene. This project addresses the problem of enhancing the quality of 3D reconstructions, particularly in scenarios with complex scenes containing unwanted objects like people and lighting variations. Achieving high-quality reconstructions in such challenging environments requires a comprehensive approach to handling object removal, matching images with consistent focal lengths, and ensuring correct image order.

The uniqueness of this project lies in its integrated pipeline, which encompasses inpainting for object removal, selection and arrangement of images with matching focal lengths, and the utilization of COLMAP for 3D reconstruction. This report will provide insights into the methods used, the results achieved, and future prospects for further improvements. By addressing these complexities, this project contributes to the advancement of 3D reconstruction techniques, making them more accessible and effective for applications in fields such as virtual reality, robotics, and cultural heritage preservation.

2. Existing Alternatives (Related Work)

In the field of 3D reconstruction from 2D images, numerous approaches and software packages have been developed to address the challenges associated with capturing and modeling real-world scenes. One common alternative is the use of structure-from-motion (SfM) algorithms. SfM techniques, like VisualSFM and Bundler, are widely used for creating 3D models from image sequences. They rely on feature matching and camera pose estimation to reconstruct scene geometry. However, these methods often struggle when dealing with challenging scenarios, such as complex backgrounds, dynamic objects, or varying focal lengths.

Another alternative approach leverages deep learning and convolutional neural networks (CNNs). These methods, exemplified by techniques like COLMAP's deep model integration, aim to enhance the accuracy of feature matching and depth estimation. The use of neural networks allows for improved object recognition and depth map estimation. Nevertheless, these methods often require substantial computational resources and large training datasets.

3. Method Section

In this section, we detail our methodology for enhancing 3D reconstruction from 2D images with a focus on addressing challenging scenarios. The workflow consists of several key steps.

Image Preprocessing: We begin by capturing images of the scene. Also, we are using videos to sample the images that are required for 3D reconstruction. To improve the quality of the input data and to remove distortions due to noise, we employ a pre-trained inpainting model which consists of semantic segmentation and two-stage adversarial architecture to paint the removed regions. This has the capability to remove unwanted objects and focus on the required region. This step enhances the reliability of feature matching while reducing the required computational power.

Captured raw images for 3D reconstruction.

After preprocessing with the inpainting algorithm

The pre-trained model used for this inpainting algorithm is trained on output images of 256x256 pixel size. But for 3D reconstruction, we require higher-quality images. As a solution for that, we have used a Python code for slicing the high-quality images to 256x256 sized small images and saving them in a new directory to run the inpainting algorithm and another code for merging the outputs from the inpainting algorithm of the sliced original image. To ensure the proper order during merging, we use renaming for newly generated images with unique names to identify their original image, and the segment of the original image is that.

```python
def slice_image(input_image_path, output_folder):
    original_image = Image.open(input_image_path)
    width, height = original_image.size
    tile_size = 256

    if not os.path.exists(output_folder):
        os.makedirs(output_folder)

    for i in range(0, width, tile_size):
        for j in range(0, height, tile_size):
            left = i
            upper = j
            right = i + tile_size
            lower = j + tile_size

            small_image = original_image.crop((left, upper, right, lower))
            small_image.save(os.path.join(output_folder, f"{i}_{j}.png"))
```

Making 256x256 images from high-quality image

```python
def merge_images(input_folder, output_image_path, original_width):
    images = []
    for filename in os.listdir(input_folder):
        if filename.endswith(".png"):
            x, y = map(int, filename[:-4].split("_"))  # Extract coordinates from filename
            img_path = os.path.join(input_folder, filename)
            img = Image.open(img_path)
            images.append((x, y, img))

    if not images:
        print("No images found in the input folder.")
        return

    # Sort images based on their coordinates
    images.sort(key=lambda x: (x[1], x[0]))

    # Calculate total width and height of the merged image
    total_width = images[0][2].width * (original_width // images[0][2].width)
    total_height = images[0][2].height * ((len(images) // (original_width // images[0][2].width)) + 1)

    # Create a new image with the calculated dimensions
    merged_image = Image.new( mode: "RGB",  size: (total_width, total_height))
```

Merging small images to make the high quality image

Image Selection and Arrangement: Next, we select images with consistent focal lengths to ensure accurate reconstruction. This reduces distortions and improves the quality of the final 3D model. Based on the COLMAP documentation, COLMAP algorithm functions best when images are provided in a matching sequence. For capturing all the details, we use manual image capturing and sampling images through videos from obtaining a larger dataset. After obtaining the data we use sift property matching to detect the similarity of images using number of sift properties and rename the images to make the matching image sequence.

```python
# Detect keypoints and descriptors for both images
keypoints1, descriptors1 = sift.detectAndCompute(image1, None)
keypoints2, descriptors2 = sift.detectAndCompute(image2, None)

# Initialize a BFMatcher (Brute Force Matcher) with default params
bf = cv2.BFMatcher()

# Match descriptors using KNN (K-Nearest Neighbors)
matches = bf.knnMatch(descriptors1, descriptors2, k=2)
```

Calculating sift features for given 2 images

```python
def find_best_reference_image(images):
    max_keypoints = -1
    best_reference_image = None
    for filename in images:
        image_path = os.path.join(image_directory, filename)
        num_common_keypoints = compare_sift_properties(reference_image_path, image_path)
        if num_common_keypoints > max_keypoints:
            max_keypoints = num_common_keypoints
            best_reference_image = image_path
    return best_reference_image


# List to store the order of images
ordered_images = []

# Initial reference image
reference_image_path = os.path.join(image_directory, "001.jpg")

# Get a list of all image files in the directory
image_files = [filename for filename in os.listdir(image_directory) if filename.endswith(".jpg")]
```

Select a random image at first (as 001.jpg) and compute sift properties to find the best matching image and rename it as 002.jpg. Then take 002.jpg as the reference and repeat the process to get the ordered images set.

Feature Matching and Reconstruction: The selected images are passed through COLMAP, for 3D reconstruction. COLMAP uses feature matching and camera pose estimation to reconstruct the 3D scene geometry. By building upon COLMAP's capabilities, we improve its ability to handle challenging scenarios using preprocessing the provided images to remove distracting objects using machine vision.

4. Techniques Used in COLMAP

COLMAP is a powerful and versatile tool for structure-from-motion (SfM) and 3D reconstruction, known for its use of various computer vision techniques. At its core, COLMAP employs feature-based matching and bundle adjustment, which are fundamental in the field of SfM. Feature-based matching relies on detecting key points in images and establishing correspondence between them. COLMAP utilizes a variety of feature detectors and descriptors, such as SIFT, SURF, and ORB, to find these key points and match them across multiple images. This technique provides the foundation for building the sparse 3D structure.

In addition to feature matching, COLMAP also incorporates bundle adjustment, which is essential for refining camera poses and 3D points. Bundle adjustment is an optimization process that minimizes the reprojection error between the 3D points and their projections in multiple images. COLMAP employs a robust and efficient bundle adjustment algorithm, enabling the accurate alignment of the reconstructed model. Moreover, COLMAP supports various camera models, from simple pinhole models to more complex ones, allowing users to choose the most appropriate camera model based on the specifics of their dataset. The flexibility in camera modeling is a significant advantage, particularly for users with prior knowledge of the camera parameters. Finally, COLMAP integrates tools for image-based and point cloud-based dense reconstruction, enabling users to create detailed and textured 3D models.

5. How our solution is building on top of it

Our solution builds upon the capabilities of COLMAP by enhancing the preprocessing and optimization stages of 3D reconstruction, making it particularly suitable for applications in urban and architectural environments. While COLMAP provides a robust framework for reconstructing 3D scenes from unordered image sets, our approach focuses on addressing challenges specific to building reconstruction. We have extended COLMAP's functionality to incorporate a specialized preprocessing step that uses inpainting models to remove unwanted objects and individuals from images, ensuring that only the core structure of the building is captured. This not only streamlines the reconstruction process but also minimizes errors caused by occlusions, thus leading to more accurate 3D models. Our modifications also include an improved feature point matching algorithm that leverages SIFT to automatically arrange images in chronological order, enhancing the efficiency of COLMAP in handling large datasets of building images.

What sets our solution apart is its adaptability and optimization for architectural applications. We have tailored the preprocessing techniques to the unique challenges of building reconstruction, such as handling scenes with varying lighting conditions and mitigating the impact of dynamically moving objects. Our solution also offers 3D reconstruction of buildings or famous places where we can’t obtain clear isolated images due to crowds or any other distractions. Furthermore, we have an integrated image ordering system

instead of using unordered images to obtain better results quickly to produce highly detailed and precise 3D models. These optimizations make our solution not only more efficient but also more accurate, providing a specialized tool for those in the fields of architecture, urban planning, and heritage preservation. It paves the way for expedited, high-fidelity 3D reconstructions of complex structures while reducing the need for time-consuming manual data cleaning and ordering.

This integrated approach aims to address limitations found in existing alternatives and enhance the quality of 3D reconstruction in complex scenes. This method capitalizes on the strengths of inpainting, feature matching, and COLMAP to create accurate and reliable 3D models from 2D images. The following section will present point cloud representations and MeshLab results of buildings in complex environments.

6. Results

6.1. ENTC Building

We captured approximately 250 images of the ENTC building, but the presence of substantial tree cover and surrounding buildings presents a significant obstacle to achieving an accurate 3D reconstruction of the structure.

Limitations - Lack of images from the top view of the building

Unable to remove objects which cover the building more than 75% from the image.

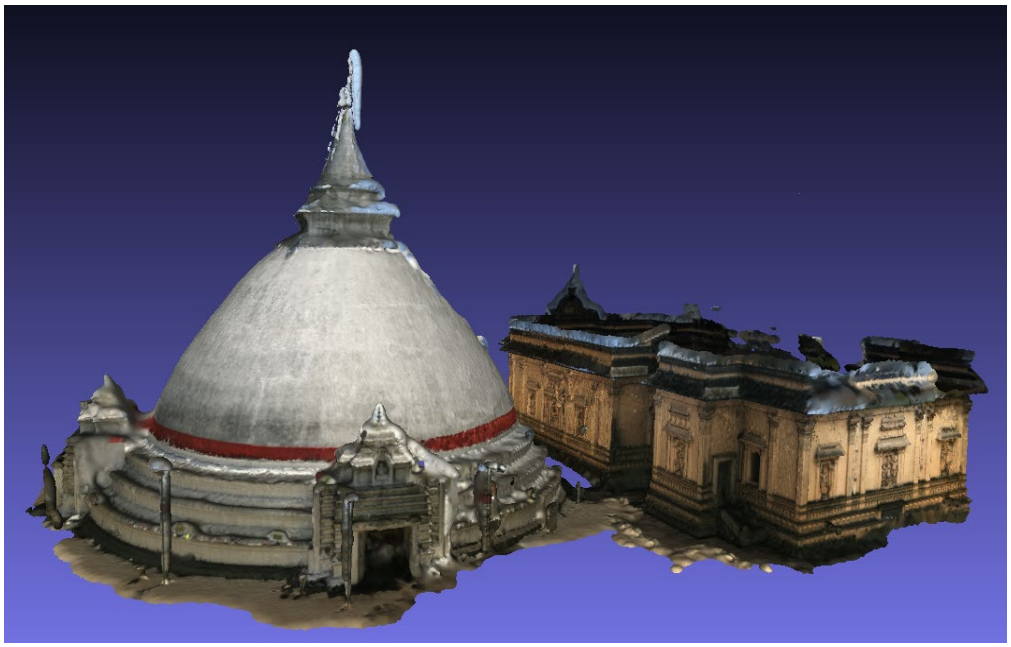
Point cloud representation

6.2. Kelaniya Raja Maha Viharaya

We collected 325 images of Kelaniya Raja Maha Viharaya, a crowded place. These images were preprocessed to obtain the desired results swiftly, and the removal of people and unwanted elements was facilitated through the inpainting model.

Limitations - Lack of images from the top view of the building
Crowded environments affect image quality, and its incompatibility with COLMAP software for advanced 3D reconstruction.

Results using MeshLab software



7. Future Challenges and Possible Improvements

While COLMAP represents a robust and comprehensive tool for 3D reconstruction, there are several potential areas for further development and improvement.

Real-time Reconstruction: One of the ongoing challenges is to enhance the speed of reconstruction. While COLMAP is capable of processing large datasets, further optimization for real-time or near real-time reconstruction would be invaluable, especially in applications like augmented reality and autonomous navigation systems. Our approach mainly addresses reducing the computational power and time required for 3D reconstruction by applying pre-trained models for removing distracting objects. This method reaches one step toward real-time 3D reconstructions.

Large-Scale Scene Reconstruction: Expanding the capabilities of COLMAP to handle even larger-scale scenes and datasets is a crucial area for development. This includes the ability to handle scenes with thousands of images, making it more suitable for applications like urban planning and large-scale environment mapping. Image preprocessing and ordering based on feature matching can be improved further so that COLMAP can easily process a large number of image datasets at once.

Improved User Interfaces: While COLMAP offers both command-line and GUI interfaces, there is room for more user-friendly GUI enhancements. Streamlining the user interface and simplifying the parameter tuning process could make it more accessible to a wider user base.

Integration with Sensor Data: Many modern applications involve the fusion of 3D reconstruction with sensor data such as LiDAR or GPS. Enhancing COLMAP's capabilities to integrate seamlessly with other sensor data sources would be beneficial, especially for robotics and autonomous systems.

More Robust Handling of Challenging Data: Improving COLMAP's robustness to challenging scenarios, such as low-light conditions, dynamic scenes, or severe occlusions, remains a significant challenge. This could involve more advanced feature detection and tracking techniques.

8. Acknowledgment

For this research and implementation, we acknowledge the following resources and tools.

We used COLMAP extensively for the 3D reconstruction and analysis tasks. COLMAP served as the foundation for our work, enabling us to leverage its capabilities for structure-from-motion and dense reconstruction. During the preprocessing stage, we utilized inpainting models and techniques for the removal of unwanted objects and people from the images. This helped us create cleaner input data for the reconstruction process. We relied on the SIFT algorithm for feature matching and image ordering. SIFT played a crucial role in ensuring that our images were correctly aligned and matched, forming the basis for the reconstruction process.

We referred to a variety of research papers, technical documentation, and online resources related to computer vision, 3D reconstruction, and inpainting techniques. These resources provided valuable insights and guidance throughout our research. We would like to express our gratitude to the developers of COLMAP and the various open source inpainting models that made our work possible. Additionally, we appreciate the availability of GPU resources that facilitated the efficient execution of our tasks.

9. References

- COLMAP: https://colmap.github.io/
- CVPR 2017 Tutorial: https://demuc.de/tutorials/cvpr2017/
- Inpainting Algorithm: https://github.com/sujaykhandekar/Automated-objects-removal-inpainter