Offline 3D Augmentation with COLMAP

Date of current version December 15th 2024

JASMINE KHALIL
Pennsylvania State University, State College PA (e-mail: jkk5987@psu.edu)

This work was carried out as part of my Honors project for the Fundamentals of Computer Vision course at Penn State, instructed by Dr. Robert Collins.

ABSTRACT This project describes the development of an **Augmented Reality (AR) Viewer** capable of overlaying virtual objects onto images of real-world scenes. Our offline implementation integrates concepts from computer vision, computer graphics, and geometry. Using **COLMAP software** for 3D reconstruction, we generated a sparse 3D point cloud from overlapping camera image views. The key tasks completed include extracting the dominant plane using a custom **RANSAC algorithm**, transforming coordinate systems, and projecting a virtual 3D object onto 2D images using camera parameters. This project emphasizes foundational mathematical techniques, such as camera projection and geometric transformations, with a focus on implementing core algorithms without relying on external computer vision libraries. Our work provides a solid basis for future real-time AR applications.

The code outlined in this work can be found **here**.

INDEX TERMS 3D Augmentation, Augmented Reality, Camera Projection, COLMAP, Computer Vision, RANSAC.

INTRODUCTION

Augmented reality (AR) combines virtual and real-world elements by superimposing digital objects onto real physical environments. Modern AR systems, such as ARKit, achieve real-time interactions through advanced computational techniques. This project lays the groundwork for an AR viewer by building an offline prototype that overlays virtual objects onto images of 3D scenes. Our work encompasses essential steps for augmented reality, including 3D reconstruction, object placement, and 3D-to-2D projection.

This project uses COLMAP software for 3D point cloud reconstruction, which provides the foundational geometry of a scene. However, we independently developed much of the implementation, such as dominant plane detection using a custom RANSAC algorithm and 3D-to-2D projection and visualization. Using Matlab, this project bridges the gap between theoretical computer vision and practical AR viewers.

We first captured a set of overlapping images of a 3D scene and used COLMAP to generate a sparse 3D point cloud. We then developed and implemented a RANSAC algorithm to identify the dominant plane within the scene, followed by defining a local coordinate system for placing a virtual object. Afterward, we created a virtual 3D object, transformed it to fit the scene's geometry, and projected it onto the original images using the pinhole camera model based on internal and external camera parameters. Finally, we rendered the image views with the virtual object accurately overlaid, achieving an offline AR viewer.

I. COLLECTING A SET OF IMAGES

We started by collecting a set of images that show different views of the same 3D scene. We used a single camera for each view and did not zoom in or out between shots so that the internal camera parameters remained the same. We chose a scene with a dominant planar surface to ensure we could accurately place an object in the scene. The desk space we chose contains three detailed paintings on the wall with many features for matching, which became the dominant plane in the reconstructed scene. Below are the seven views we obtained.

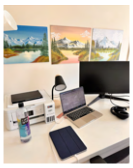
FIGURE 1: View 1

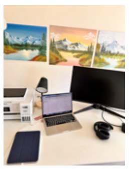
FIGURE 2: View 2

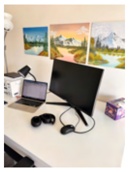
FIGURE 3: View 3

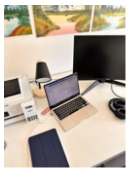
FIGURE 4: View 4

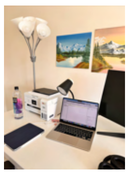
FIGURE 5: View 5

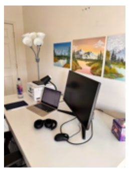
FIGURE 6: View 6

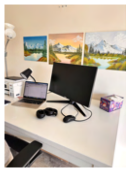
FIGURE 7: View 7

II. COLMAP

COLMAP creates a 3D reconstruction of a scene from overlapping camera views. It produced three text files for the sparse 3D point model, which we used to do post-processing to find the dominant plane of our scene, place a virtual 3D object on that plane, and then project it back into the original images.

After importing our views, we ran feature extraction, matching, incremental reconstruction, and bundle adjustment to obtain a sparse reconstruction. Since we took all the pictures using the same camera, we also checked the 'shared for all images' box to tell COLMAP to fit a single set of internal camera parameters.

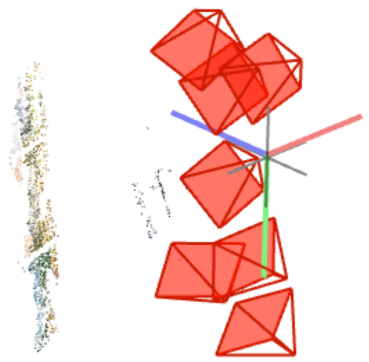

FIGURE 8: 3D reconstruction of the scene in COLMAP from 7 views.

We then exported the sparse 3D point model produced by COLMAP as a set of three text files: cameras.txt, images.txt, and points3D.txt.

III. RANSAC

RANdom **SA**mple **C**onsensus is an iterative method that estimates model parameters in Machine Learning and Computer Vision.

Generally, RANSAC follows these steps:

1) Repeatedly divides data into the minimum number of points required to determine the model's parameters.
2) Solves for the parameters of the model.
3) Determines how many points from the entire dataset fit within a predetermined threshold, X.
4) If the current number of inliers exceeds the maximum number of inliers, the current estimated model parameters are the best.
5) This process repeats for N iterations.

In our case, fitting a plane requires at least three 3D points. Below is the algorithm we used to find the dominant plane in the scene (the plane containing the largest subset of 3D points).

Algorithm 1 RANSAC for Dominant Plane

```
1:  Threshold $\rightarrow$ X
2:  Iterations $\rightarrow$ N
3:  Global Inlier Count $\rightarrow$ 0
4:  Max Inliers $\rightarrow$ 0
5:  Best Inliers $\rightarrow$ []
6:  Best Plane $\rightarrow$ [0, 0, 0, 0]
7:  for Steps in range $(1, N)$ do
8:      Select 3 random point correspondences
9:      Calculate plane equation Ax+By+Cz+D = 0
10:     Calculate distance of all points to plane
11:     Inliers $\rightarrow$ Distances < X
12:     if Current Inliers > Max Inliers then
13:         Max Inliers $\rightarrow$ Current Inliers
14:         Best Inliers $\rightarrow$ Inliers
15:         Best Plane $\rightarrow$ Current Plane
16:     else
17:         Max Inliers $\rightarrow$ Max Inliers
18:     end if
19: end for
```

After reading in the 3D point cloud stored in points3D.txt to obtain X, Y, and Z coordinates for each point, we ran our algorithm in Matlab. We obtained the following result, successfully identifying the scene's dominant plane.

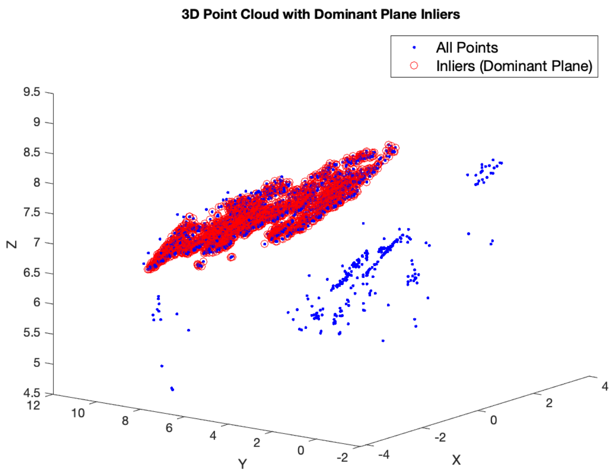

FIGURE 9: Plot of all points and inliers accurately identifying our dominant plane.

IV. 3D EUCLIDEAN TRANSFORMATION OF A VIRTUAL OBJECT

Our next goal was forming a local x-y-z coordinate system where the dominant plane we found became the z = 0 plane. Additionally, we chose the local origin of our coordinate system so that (x, y) = (0, 0) laid roughly in the middle of the set of inlier points on the dominant plane. To do this Euclidean transformation, we completed the following steps:

1) Computed the dominant plane's normal vector.
2) Computed two orthogonal vectors on the plane (both orthogonal to the normal).

3) Computed the local coordinate system origin.
4) Constructed a rotation matrix using the three orthogonal vectors previously calculated.

With our rotation matrix, we constructed a 3D virtual box and transformed its corners into scene coordinates. To do this, we followed these general steps:

1) Defined the size of a 3D virtual box and its corners.
2) Transformed the corners of the box using the rotation matrix and center of the dominant plane:

$$\text{Transformed Corners} = R * \text{Box Corners}^T + \text{Centroid}.$$

3) Defined the box edges from its corners.
4) Extracted the 3D coordinates of each corner pair.
5) Drew lines between each corner pair in 3D space.

After completing these steps, we successfully plotted the following 3D virtual box centered on the dominant plane:

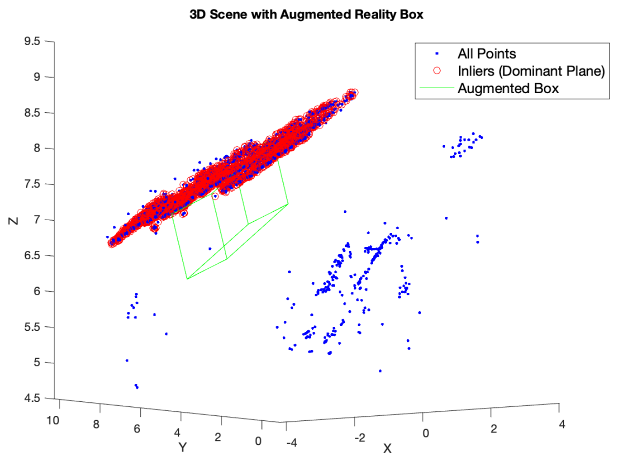

FIGURE 10: Plot of 3D box accurately placed at the center of the dominant plane.

V. 3D TO 2D PROJECTION OF VIRTUAL OBJECT IN REAL SCENE

To project the 3D augmented box into the real 2D image views, we used the pinhole camera model. We converted the box's corners from 3D points to 2D coordinates by extracting the intrinsic and extrinsic parameters from the text files produced by COLMAP, and constructed the following components.

- A rotation matrix for each image view from its quaternion.
- The intrinsic matrix, Kmat, from the focal length, principal point in x, and principal point in y.
- An extrinsic matrix for each image view from their rotation matrix and translation vector.

Substituting these variables into the pinhole camera model:

$$\underbrace{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}_{\substack{\text{Pixel}\\\text{Location}}} \sim \underbrace{\begin{bmatrix} f & 0 & 0 & o_x \\ 0 & f & 0 & o_y \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\substack{\text{Perspective}\\\text{Projection}}} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\substack{\text{World}\\\text{to Camera}}} \underbrace{\begin{bmatrix} U \\ V \\ W \\ 1 \end{bmatrix}}_{\substack{\text{World}\\\text{Point}}},$$

we obtained a set of projected 2D points for the box's corners in each image view.

To overlay the box in each image, we defined its six faces using its corners. From the four corners of each face, we found the face's center and rotated and translated it into camera coordinates. We then found the depth of each face as the third component of their new rotated and translated coordinates and sorted the faces from farthest to closest. We repeated this process for each image view.

Lastly, to match the painting in the background, we defined a color gradient from pink to yellow and filled the faces in order of depth.

The following images are the views with the 3D-augmented box placed on the dominant plane of the scene, the wall with the paintings.

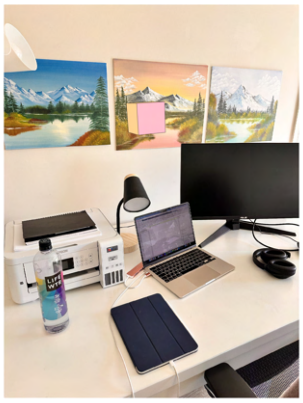

FIGURE 11: View 1 with 3D augmented box.

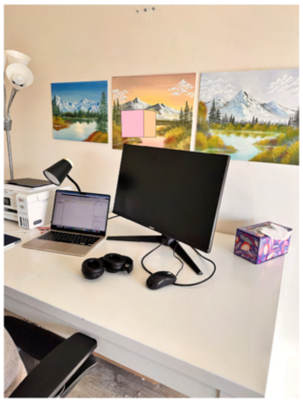

FIGURE 12: View 2 with 3D augmented box.

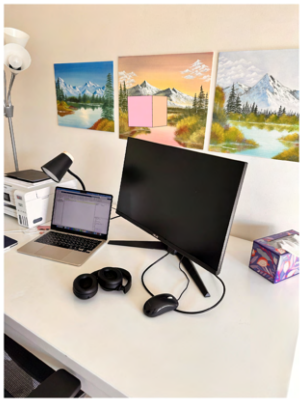

FIGURE 13: View 3 with 3D augmented box.

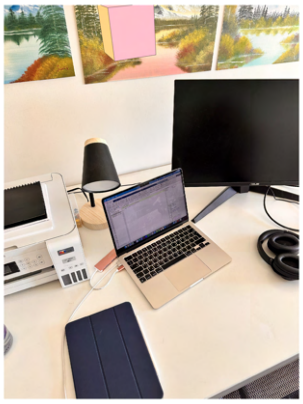

FIGURE 14: View 4 with 3D augmented box.

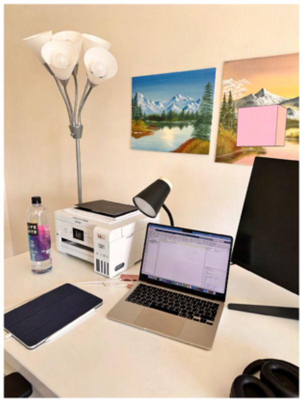
FIGURE 15: View 5 with 3D augmented box.

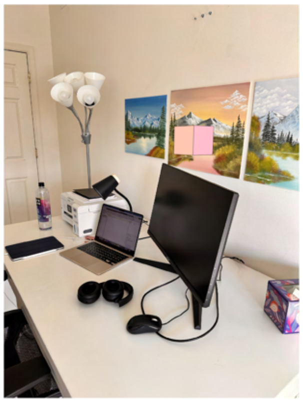
FIGURE 16: View 6 with 3D augmented box.

CONCLUSION

This project successfully demonstrates the creation of an augmented reality (AR) viewer by integrating concepts from computer vision, geometry, and graphics. Using COLMAP, we generated a sparse 3D reconstruction of a scene and identified a dominant plane using a custom RANSAC algorithm. We then implemented 3D-to-2D camera projection and coordinate transformations to project an augmented box onto the views of the 2D image scene. This project provides a robust foundation for extending these techniques toward future advancements in AR, including interactive and real-time solutions for various applications.

ACKNOWLEDGMENT

I would like to acknowledge Dr. Robert Collins for his guidance and insightful comments throughout the duration of the project.