# Yale GeoGuessr: A Custom Panoramic Reconstruction Pipeline for Campus-Scale Location Guessing

Andy Cheng and Jeet Parikh

December 2025

# Contents

**1 Introduction** **3**

**2 Related Work** **3**

**3 Methodology** **3**
3.1 Image Capture . . . . . 3
3.2 Cylindrical Projection . . . . . 4
3.3 Feature-Based Alignment . . . . . 5
3.4 Panorama Composition . . . . . 5
3.5 Drift Normalization . . . . . 5
3.6 Loop Closure Correction . . . . . 5
3.7 Depth-Aware Blending (Optional) . . . . . 5
3.8 Rendering and Gameplay . . . . . 5

**4 Data** **5**

**5 Implementation Details** **6**
5.1 Preprocessing . . . . . 6
5.2 Key Parameters . . . . . 6

**6 Results** **6**

**7 Discussion** **7**

**8 Play The Game** **8**

# 1 Introduction

Single images often lack enough spatial context to reliably identify a location, especially in visually similar environments such as university campuses. Although existing platforms like GeoGuessr address this limitation using street-level panoramic imagery, these systems rely on proprietary datasets and provide limited control over the data collection and reconstruction process. In addition, they do not provide any form of indoor viewing.

The goal of this project is to design and implement an end-to-end system that captures both indoor and outdoor real-world environments, reconstructs immersive 360-degree panoramas using classical computer vision techniques, and presents them in an interactive, game-based web interface. Specifically, we build a Yale-campus version of a GeoGuessr-style game where users guess their location based solely on a panoramic visual experience.

Our approach combines custom image capture using an RGB-D RealSense camera, cylindrical panorama stitching with feature-based alignment, optional depth-aware blending to reduce parallax artifacts, and a locally hosted web application for visualization and gameplay. Key challenges included camera drift over long rotations, feature matching robustness in architectural scenes, loop-closure errors in 360-degree panoramas, and bridging offline computer vision pipelines with real-time frontend rendering.

# 2 Related Work

Panoramic image stitching has been extensively studied in computer vision, with classical methods relying on feature detection, matching, and geometric alignment. In 2007, Brown and Lowe introduced automatic panoramic stitching using invariant local features, forming the foundation of many modern systems.

Feature detectors such as SIFT, SURF, and ORB have been widely used for image alignment due to their robustness to scale, rotation, and illumination changes. For cameras undergoing rotational motion, cylindrical and spherical projections are commonly employed to simplify alignment by reducing the transformation between views to near-translation.

Depth-aware panorama stitching has been explored to mitigate parallax errors by leveraging RGB-D sensors, often through full 3D reconstruction and reprojection. While such approaches yield high accuracy, they require precise calibration and pose estimation.

Unlike existing GeoGuessr-style platforms that depend on pre-existing street-view imagery, this project constructs panoramas from scratch using custom capture and classical computer vision techniques, emphasizing interpretability and full pipeline control.

# 3 Methodology

Our system follows a structured pipeline from data capture to interactive gameplay:

## 3.1 Image Capture

We capture sequential RGB and depth images using an Intel RealSense D435i camera. The camera is rotated manually around a fixed point, and frames are sampled at regular angular increments. Depth frames are aligned to the color stream using the RealSense SDK to ensure pixel-wise correspondence.

Figure 1: RealSense Camera Setup

### 3.2 Cylindrical Projection

Each RGB image is projected onto a cylindrical surface under the assumption of rotation-only camera motion. For each pixel $(x, y)$, the cylindrical mapping is defined as:

$$\theta = \frac{x - c_x}{f}, \quad x' = f \tan(\theta) + c_x, \quad y' = \frac{y - c_y}{\cos(\theta)} + c_y$$

where $f$ is the focal length estimated from the camera's horizontal field of view.

### 3.3 Feature-Based Alignment

We use SIFT feature detection and FLANN-based matching between consecutive cylindrically warped images. Lowe's ratio test filters unreliable matches. The median displacement of matched keypoints provides a robust estimate of horizontal and vertical translation.

### 3.4 Panorama Composition

Images are placed onto a global canvas using cumulative translations. A center-weighted pixel selection strategy chooses pixels from the image with the highest confidence weight, favoring regions near the image center to reduce distortion.

### 3.5 Drift Normalization

Vertical drift accumulates over long image sequences. We compute the average vertical displacement across all image pairs and subtract it incrementally to maintain a level horizon.

### 3.6 Loop Closure Correction

To ensure seamless 360-degree panoramas, we detect and match features between the left and right edges of the panorama. A homography is estimated using RANSAC and applied to correct accumulated loop-closure error.

### 3.7 Depth-Aware Blending (Optional)

When depth data is available, we apply depth-weighted blending in overlapping regions. Pixels corresponding to closer surfaces receive higher weights, reducing ghosting caused by parallax.

### 3.8 Rendering and Gameplay

The final panorama is rendered inside a cylindrical mesh using Three.js and React Three Fiber. Users interactively rotate their view and place guesses on a map-based interface within a locally hosted web application.

## 4 Data

All data used in this project was personally collected. RGB and depth images were captured on Yale University's campus using an Intel RealSense D435i camera. No external or public datasets were used. Each location consists of approximately 20–50 RGB-D image pairs and we collected data from 12 locations.

Example: Rosenkranz Hall

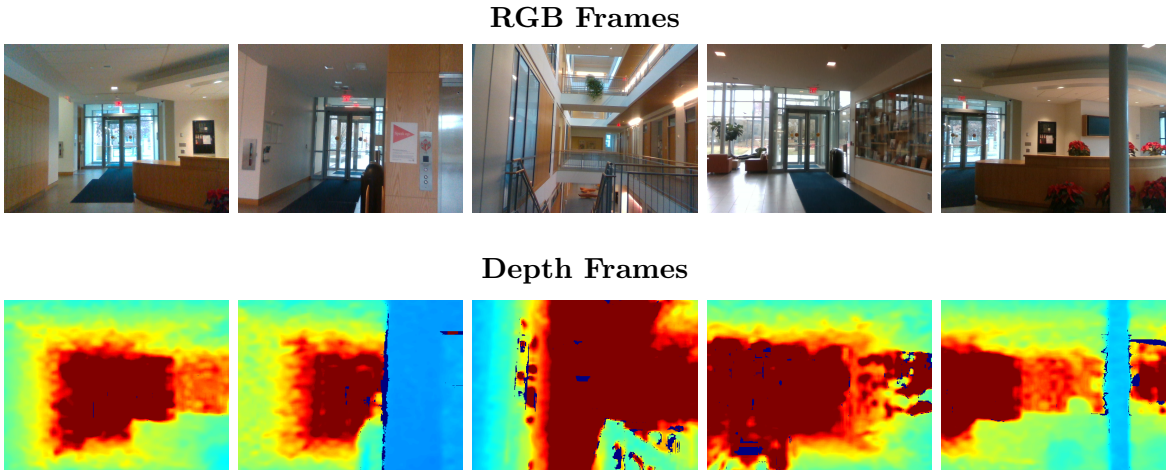


Figure 2: Sample RGB and depth frames captured during data collection.

Our 12 locations were Becton Center, Berkeley Courtyard, Franklin Courtyard, Humanities Quadrangle, Jonathan Edwards Courtyard, Marsh Lecture Hall, St. Mary's Church, Morse/Stiles Underground, Payne Whitney Gymnasium, Rosenkranz Hall, Schwarzman Center, and Watson Center.

# 5 Implementation Details

## 5.1 Preprocessing

Images are resized to a consistent resolution (640×480). Depth frames are saved in raw NumPy format. Focal length is estimated assuming a 69-degree horizontal field of view.

## 5.2 Key Parameters

- Feature detector: SIFT (2000 features)
- Matcher: FLANN with KD-tree
- Lowe's ratio threshold: 0.7
- Cylindrical projection focal length: $f = \frac{W}{2\tan(\mathrm{FOV}/2)}$
- Depth weighting: inverse depth with clipping

No machine learning models were trained in this project; all methods are classical and deterministic.

# 6 Results

The system successfully reconstructs visually coherent 360-degree panoramas for multiple campus locations. Vertical structures remain straight, seams are minimal, and loop closure ensures seamless wrap-around. Depth-aware blending reduces ghosting in scenes with nearby objects.

Evaluation is qualitative, based on:

- Visual seam continuity
- Alignment of architectural features
- Gameplay usability and immersion

The resulting panoramas are effectively used in the location-guessing game, with users able to infer location based solely on visual cues.

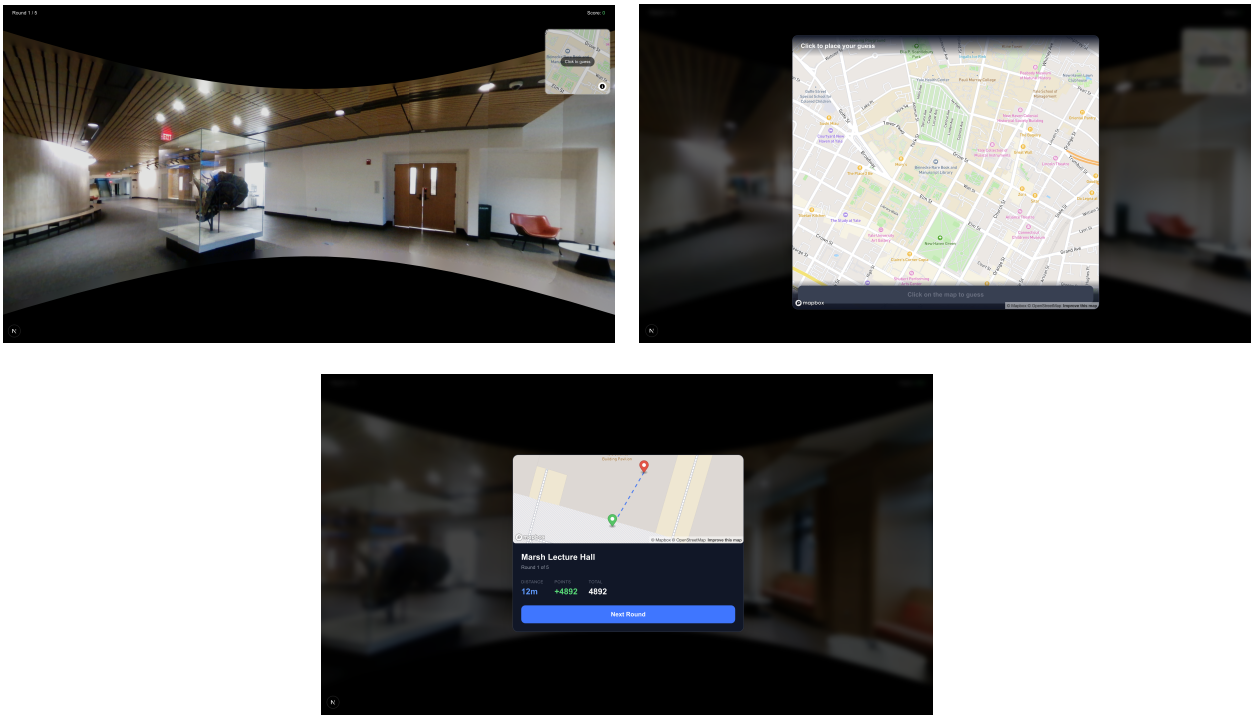


Figure 3: Gameplay screenshots demonstrating user interaction within the panoramic environment constructed from the captured RGB-D data.

## 7 Discussion

This project demonstrates how classical computer vision techniques can be combined with modern web technologies to create an immersive, end-to-end interactive system. Key challenges included managing drift over long image sequences and handling parallax without full 3D reconstruction. These were partially addressed through drift normalization and depth-aware blending.

Limitations of this work include reliance on approximate focal length estimation rather than explicit camera calibration, which can introduce subtle distortions in the cylindrical projection. Image capture was performed via manual camera rotation, making the system sensitive to uneven angular spacing and small deviations from ideal rotation-only motion. Additionally, depth information is used heuristically during blending rather than for full geometric reprojection, limiting its effectiveness in correcting parallax in scenes with significant depth variation.

A potential improvement involves capturing images at multiple vertical angles during data collection. In the current pipeline, images are captured at a single horizontal elevation, which results in limited vertical coverage and unused regions in the panoramic view. Capturing additional views at upward and downward angles would increase scene coverage, reduce blank regions, and enable

the construction of more immersive panoramas with improved vertical field of view. In addition, the pipeline would benefit from stronger seam handling to blend the panorama endpoints more smoothly (via improved overlap selection, exposure/color compensation, and multi-band (pyramid) blending to reduce visible discontinuities at the wrap-around boundary).

In the future, we aim to automate the data collection process by using servos to control both horizontal and vertical rotation, which would allow us to incorporate accurate camera pose information to project RGB-D pixels into a unified 3D point cloud and then reproject them into a panoramic representation. Doing so would drastically improve geometric consistency across views and significantly reduce parallax-induced artifacts.

Overall, this project reinforced the importance of geometric assumptions, robust feature matching, and careful system integration when deploying computer vision methods in real-world applications.

## 8 Play The Game

Use this link to play the game yourself: https://yale-geoguessr.vercel.app/!