BoelterWayfinder: Indoor Navigation System for UCLA Boelter Hall

Jiahui (Tiffany) Zhou
University of California, Los Angeles
Los Angeles, USA
jiahuitzhou@gmail.com

ABSTRACT

UCLA Boelter Hall is an academic building known for its inconsistent and complicated layout in the UCLA community. With nine floors and approximately 500 rooms in the building, navigation becomes especially challenging in the absence of maps. Based on the existing raster floor plan images, I propose BoelterWayfinder, a web-based navigation application that provides static indoor navigation. The application features autocomplete search, filtered destinations, and multi-floor route display. This report outlines the motivation, data collection process, technical implementation, and potential improvements of the system, providing insights into the challenges of indoor navigation without GPS infrastructure. The application is available at https://boelterwayfinder.onrender.com. The source code for this project is publicly available on GitHub at: https://github.com/tiffanyzhoujh/BoelterNavApp.

KEYWORDS

Navigation, Digitalized Maps, Graph, UI/UX

1 INTRODUCTION

Built in 1959, Boelter Hall is one of the oldest and arguably most intricate buildings in UCLA. The building spans nine floors, each with unique layouts, and connects to adjacent buildings like Engineering IV and Math Sciences. Students and faculty often find it hard to navigate the building due to a lack of maps, misleading signs, and inconsistent floor connectivity and layouts. In response to these issues, I developed *BoelterWayfinder*, a web-based application that enables users to search for rooms and visualize the shortest path between them.

BoelterWayfinder stores the layout of 9 floors and the location of 500 rooms in the building. It allows users to manually enter room numbers or select from suggested destinations. It also offers six filtering options for destination search: classrooms, offices, labs, restrooms, elevators, and exits. The route is statically rendered on floor diagrams, and each floor is shown separately in multi-floor routes. Due to the lack of GPS infrastructure in the building, the application provides static pathfinding.

2 MOTIVATION

Navigating Boelter Hall has long been a source of frustration for UCLA's community. In the first week of each quarter, it is not uncommon to see students wandering the halls, clearly lost in Boelter's maze-like layout. The primary cause is the fact that the architectural design of Boelter Hall is different by floors and includes irregular room shapes and unaligned vertical connectivity. For example, even though the Engineering Library and Room 8500 are both located on the 8th floor, to navigate from one to the other, the person needs to first go down to the 7th floor and take another elevator back to the 8th floor. In addition to the counterintuitive paths, lack of indoor signage and building maps makes navigating Boelter Hall even more challenging.

Given the scale and impact of this issue, there was a clear need for an accessible and intuitive navigation solution. Due to hardware and building infrastructure limitations, real-time indoor positioning via GPS was not feasible. As an alternative, I opted for a static navigation approach: computing paths based on a manually constructed graph that contains location information of all rooms and hallways in the building. This approach allows fast, accurate, and low-resource routing for indoor use.

3 DATA COLLECTION AND PREPARATION

3.1 UCLA Space Inventory System

The core of this project depends on acquiring accurate floor plan data. In collaboration with Anthony Redon, the Director of Building Services & Logistics of Samueli School of Engineering, I gained access to UCLA's Space Inventory System. However, as shown in Figure 1, the system exports only raster images of floor plans without embedded geospatial metadata.

To work around this limitation, I used Adobe Illustrator to redraw each floor plan based on these images (Figure 2). The redrawn maps formed the foundation for navigation rendering and were stored as static images in the backend. I also annotated each floor plan with icons representing key facilities: restrooms, elevators, staircases, exits, the Engineering Library, the Stack, and SEAS Café.

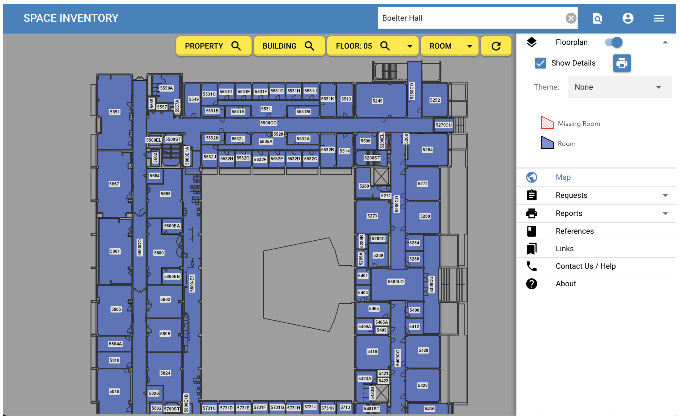


Figure 1: Information provided by UCLA Space Inventory

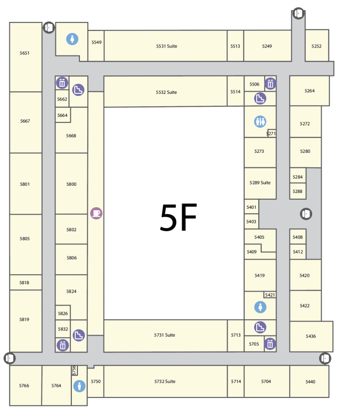
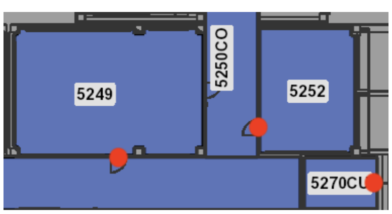
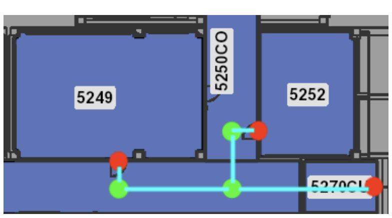


Figure 2: Maps manually redrawn with Adobe Illustrator



(a) Mark the dots for all destinations, including exits. Save coordinates in CSV files.

(b) Insert intermediate dots in the walkway to form paths. Save coordinates in CSV files.



(c) Connect the dots to form paths that cover all possible routes. Save the adjacency list in JSON files.

Figure 3: Map data digitalization

3.2 Data Digitalization and Storage

Due to the irregularity of Boelter Hall's layout, a traditional grid-based 2D spatial model was highly memory-inefficient and inflexible. Instead, I deployed a sparse graph model, where each room is represented by a single door node, and door-to-door routing is simplified as a shortest-path problem within the graph. The data preparation process included the following steps:

- Auditing in-person to verify floor layout and room placement. Most mismatches between Space Inventory floor plans and actual layout fall into these categories:
  - Room number mismatch. E.g. Room 6528 is labeled as 6532, Room 6731L is labeled as 6723 in real life.
  - Door mismatch. E.g. Two doors appear to be entrances for SEAS Cafe, but one of them is exit only in reality.
  - Restricted access areas. E.g. On Floor 9, access of Room 9404, 9406, and 9410 is restricted by fences. The Space Inventory map does not show the fact that they are accessible through one specific door.
- Manually placing doors on the maps and assigning coordinates. To simplify path finding, each room is assigned one door, and the coordinates of the doors represent the location of the rooms (Figure 3a).
- Adding intermediate dots (as shown in Figure 3b) along the walkways and recording coordinates, ensuring paths make only 90°turns for clean route rendering.
- As in Figure 3c, constructing edges by connecting walkway vertices, connecting door vertices to walkway ones, and connecting elevator vertices to each other. Thereby, the entire building could function as multi-level graph, and the weight of all edges can be calculated by Euclidean distances between vertices. Weights of edges that connect elevator vertices to each other were assigned manually .

4 IMPLEMENTATION

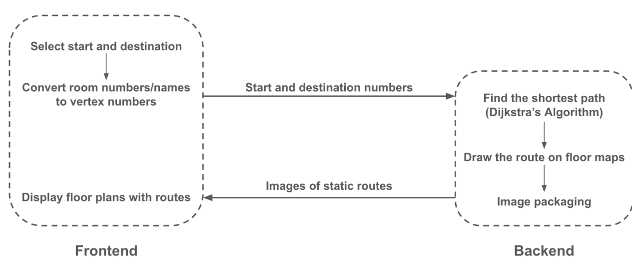
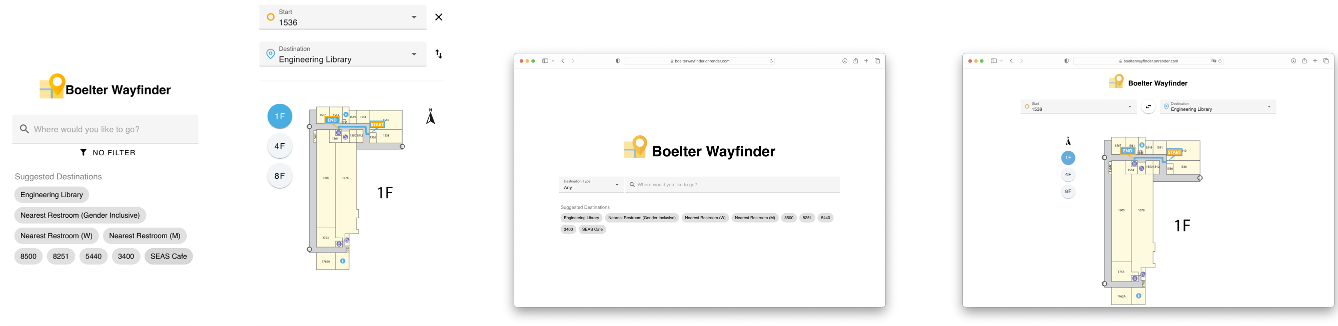


Figure 4: System design



(a) Homepage mobile view **(b) Navigation mobile view** **(c) Homepage desktop view** **(d) Navigation desktop view**

Figure 5: Mobile and desktop views of frontend

4.1 Frontend

The frontend of BoelterWayfinder is built using Vue.js, a JavaScript framework known for its flexibility and reactive data binding. Major components include:

- Homepage (Figure 5a, Figure 5c)
  - Filter-based destination search, enabling users to narrow down the destination out of 500 rooms with six categories - classrooms, offices, labs, restrooms, elevators, and exits.
  - Suggested destinations, such as Engineering Library, SEAS Cafe, and Nearest Restrooms which are automatically determined from the user's starting location.

  After the user specifies a destination or takes the suggested destination shortcut, they will be redirected to the navigation page.
- Navigation page (Figure 5b, Figure 5d)
  - Two autocomplete input fields for start and destination room selection. To keep the user interface concise, search fields on the navigation pagedo not include filtering options. Start and end points can be swapped by clicking on the swap button.
  - The best route is displayed in the form of static PNG files. When the route involves multiple floors, users can navigate between each map via the sidebar.

  When both the start and the destination are specified, a route will be displayed instantly. If the user updates any of the input fields, the route will also be updated in real time.

As shown in Figure 5, the user interface is optimized for both mobile and desktop devices.

4.2 Backend

The backend is written in Python using the Flask framework. It performs the following core functions:

- Special destination preprocessing (nearest restrooms), finding the optimal destination.
- Path computation using Dijkstra's algorithm over the preconstructed adjacency graph.
- Route rendering, overlaying the selected path on floor plan images using dot coordinates.
- Image packaging, returning per-floor route images to the frontend for display.

All room and dot data are stored as JSON and CSV files in the backend, which enables lightweight and fast lookup operations. The raw floor plans are stored as high resolution PNG files.

5 FUTURE WORK

While BoelterWayfinder provides a complete static routing experience, there are still potential improvements for better user experience.

- Even though GPS signal inside Boelter Hall is very limited, it can still be helpful to provide the real-time position of the user optionally as reference. The aid of GPS location can potentially improve user experience in higher floors.
- Real-life views, similar to Google Maps' Street View, could further simplify the user experience.
- User feedback integration. Since the room assignment of Boelter Hall can be updated each quarter, it can be useful to add a user feedback intake to keep information up-to-date.

6 CONCLUSION

BoelterWayfinder provides a lightweight solution to the long-standing navigation problem faced by the UCLA community by combining digitized maps, graph theory, and user-centric

UI/UX design. While static in nature, it lays the foundation for future enhancements that could eventually support real-time navigation. This project not only improves daily experience for students and faculty but also demonstrates how thoughtful technical design can solve localized but impactful usability issues.