Title: Towards Automatic Vectorization of Architectural Floor Plans

Abstract: This research delves into the significant challenge of converting detailed architectural raster floor plan images into simplified, precise digital maps through the application of state-of-the-art computer vision and deep learning techniques, coupled with advanced image processing strategies. It specifically targets the shortcomings of prevalent wall extraction algorithms that often fail to deliver clear and accurate representations suitable for various practical uses. The core of the study is an innovative methodology that harnesses the capabilities of neural network models, enhanced by the implementation of Hough transforms, morphological operations, and feature extraction, to improve the transformation process markedly. A series of rigorous experiments were conducted to refine and validate the proposed methodology, focusing on the iterative enhancement of the clarity and precision of the resultant digital maps. The outcomes demonstrate a leap forward in the quality of the digital maps generated from architectural floor plans, with the integrated approach yielding significantly better-defined and more accurate maps than traditional methods. This advancement has profound implications for fields requiring precise spatial representations, such as urban planning, emergency response planning, real estate, and interior design, as well as for creating navigational aids that enhance the autonomy and mobility of the visually impaired. By improving the fidelity of digital map translations, this research contributes to the body of knowledge in computer vision, providing a scalable solution for automated map generation. The findings also open avenues for subsequent research to further refine the conversion process and explore new applications in various industries that depend on detailed spatial data.

I. Introduction

A. Background and Significance

The advent of digital technology has ushered in a new era of architectural design and representation. With the growing complexity of architectural plans, there is an increasing need to translate these intricate designs into accessible and manageable formats. Raster images of architectural floor plans, while rich in detail, present challenges in terms of clarity and usability, particularly when they need to be integrated into various digital applications. These raster images, which represent visual data as a grid of individually colored pixels, are ideal for detailed and complex imagery but can be problematic due to their fixed resolution, which limits scaling and editing capabilities without loss of quality.

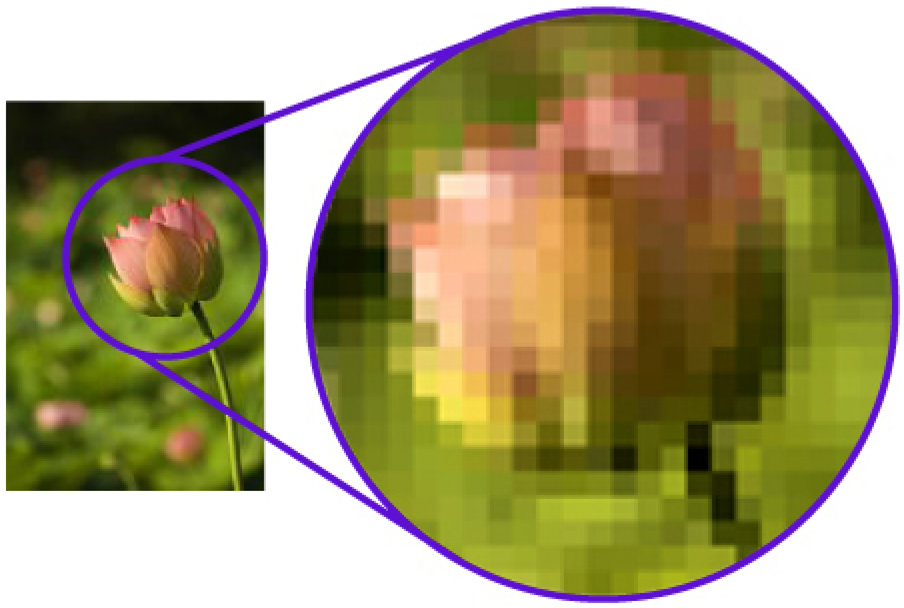
Figure 1: Sample of a raster image which represents visual data as a grid of individually colored pixels [22]

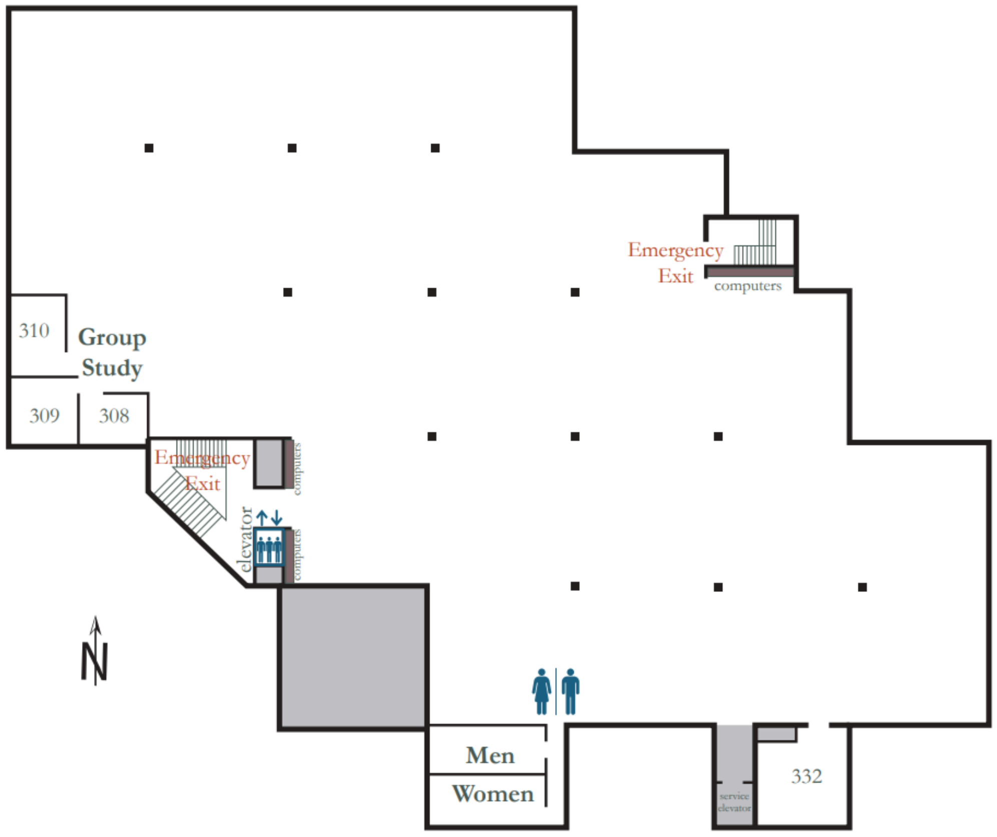

Figure 2: This image shows the UCSC Science and Engineering Library floor plan, a raster input image [24]

The ability to convert these raster images into vectorized formats is of paramount importance as it allows for scalable, precise, and easily manipulable representations of architectural spaces. Vector graphics, which represent images using paths rather than pixels, are ubiquitous in industrial designs [5], including graphic designs [11], 2D interfaces [12], and floor plans [13].

Traditional methods of vectorization, while foundational, have proven inadequate in dealing with the diversity and complexity found in modern floor plans. Wall extraction algorithms, a crucial step in this process, have been prone to inaccuracies, leading to a gap in the efficiency and reliability of the vectorized output. These gaps not only hinder the practical applications of floor plans but also limit their use in advanced simulations and analytical models that professionals in urban planning, real estate, and interior design rely upon.

The significance of this research lies in its potential to revolutionize the way architectural floor plans are processed and interpreted. By leveraging the latest advancements in computer vision and deep learning, the study proposes a sophisticated approach to address these longstanding issues. The development of an integrated system that employs noise removal, optical character recognition (OCR), and a series of advanced vectorization techniques represents a leap forward in the field. It stands to significantly enhance the digital manipulation of floor plans, providing a level of detail and accuracy that was previously unattainable.

B. Purpose of the Study

The primary purpose of this study is to develop an effective and reliable method for converting raster images of architectural floor plans into vectorized digital maps. This conversion is crucial for several reasons: it enhances the clarity and utility of the plans, facilitates their integration into various digital platforms and applications, and enables precise scaling and manipulation of the spatial data they contain.

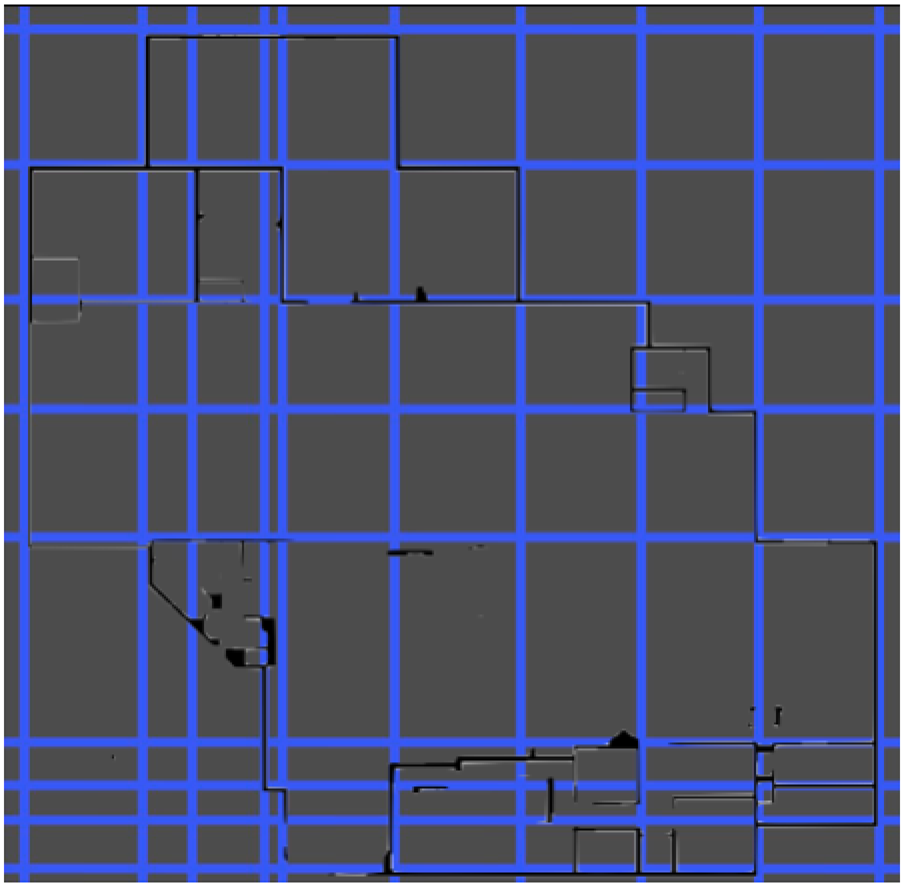
Figure 3: This image shows the input raster image (in black) overlaid with the extracted walls (in blue).

Despite the advancements in computer-aided design (CAD) and geographic information systems (GIS), the transition from raster to vector formats remains a challenge due to the noise inherent in scanned images, the complexity of architectural drawings, and the presence of non-geometric elements such as textual annotations. To overcome these hurdles, the study aims to create a sophisticated toolchain that incorporates a series of image processing techniques for noise removal, an innovative convolutional neural network (CNN) based algorithm [1] for accurate feature recognition, and advanced vectorization procedures.

The objective of the research is to streamline the process by automating the vectorization of floor plans, thereby reducing the time and effort traditionally required for manual conversion. This automation is expected to lead to a more standardized and efficient production of digital maps, ultimately facilitating a wide array of applications in real estate, urban planning, and beyond.

Additionally, the study intends to advance the understanding and capabilities of OCR in the context of architectural floor plans by customizing its application to recognize and remove text without compromising the integrity of the underlying spatial data. This aspect is critical as textual noise can significantly hinder the accuracy of feature detection and vectorization.

Finally, the study aspires to explore the efficacy of multiple techniques—including bitwise masks, corner detection, contour finding, skeletonization, and Hough transforms—in conjunction with each other. By systematically analyzing and combining these techniques, the research aims to propose a composite methodology that outperforms existing algorithms in both precision and efficiency.

Through achieving these objectives, the study will not only contribute a novel approach to the field of architectural image processing but will also serve as a foundation for future research, potentially leading to further advancements and new applications of computer vision and machine learning in the analysis and interpretation of architectural and spatial data.

C. Thesis Statement

This thesis posits that the integration of advanced image processing techniques, coupled with the application of a specialized convolutional neural network algorithm and a comprehensive vectorization process, can significantly enhance the conversion of architectural raster floor plans into vectorized digital maps. By addressing the limitations inherent in current vectorization methodologies—specifically, the noise interference and the imprecision of geometric feature extractions—the proposed approach aims to deliver a substantial improvement in the clarity, accuracy, and usability of the digitized floor plans. The research contends that such an integrated system can transform the landscape of architectural design, urban planning, and related fields by providing a reliable, efficient, and scalable solution for digital map generation, thereby expanding the scope and depth of spatial analysis and its subsequent applications.

D. Scope and Delimitation of Research

The scope of this research is deliberately focused on the development and validation of an automated system for the vectorization of architectural floor plans. It encompasses the investigation and application of noise removal techniques tailored to the context of raster images, the adaptation and optimization of a convolutional neural network algorithm known as DeepFloorPlan, and the implementation of a comprehensive suite of image processing and vectorization techniques.

Delimitations of the study are established to maintain a tight research focus and feasible project boundaries. The research will concentrate specifically on the vectorization of 2D floor plans and will not extend to 3D models or elevation drawings. The types of floor plans to be considered will be restricted to residential and commercial buildings, excluding specialized structures like industrial complexes, which often require a different set of parameters and considerations.

In terms of image processing, while the study will address common forms of noise found in raster images, such as speckles and smudges due to scanning, it will not cover the correction of distortions due to perspective or warping of paper plans. The OCR aspect will aim to eliminate textual noise from images but will not delve into the recognition and extraction of text for digital documentation purposes.

The convolutional neural network will be tailored to identify and process architectural features within floor plans, and the research will not include the development of new neural network architectures but will rather focus on optimizing existing models for the task at hand.

Vectorization will involve several advanced techniques; however, the research will delimit its scope to those that are well-suited to the 2D representation of architectural plans. Techniques that are not commonly applied in the floor plan vectorization domain, or those that require an excessive computational load disproportionate to their added value, will be excluded from this study.

The final output of the research will be a set of digital maps in a vector format, and the study will not extend to the integration of these maps into other systems or the exploration of their use in real-time applications. The research aims to build a robust foundation upon which such integrations and applications can be developed in future studies.

II. Literature Review

A. Current Methods in Architectural Floor Plan Vectorization

The vectorization of architectural floor plans is a complex domain that has seen considerable transformation over the years. Historically, the process relied heavily on heuristic-based methods that utilized low-level image processing techniques. These traditional methods often incorporated thresholding, edge detection, and morphological operations to extract key features from raster images of floor plans. Typically, these approaches aimed to identify essential architectural primitives like walls, doors, and windows by detecting lines and shapes and applying rules grounded in standard architectural drawing conventions [2].

However, the traditional techniques frequently struggled with handling complex or noisy images, proving to be less effective with varying styles and qualities of architectural drawings.

In response to these challenges, the field has seen a paradigm shift towards incorporating more sophisticated algorithms that leverage the advancements in artificial intelligence, particularly through the application of deep learning technologies. A notable method by De introduces a sophisticated approach that differentiates between thick and thin lines representing boundary and interior walls, respectively [3]. This method improves upon traditional techniques by incorporating morphological operations and geometric analysis, thereby enhancing the efficiency and accuracy of vectorizing complex architectural drawings.

Further advancements in the field are exemplified by the work of Song et al., who developed a specialized algorithm for vectorizing high-definition blueprints that contain intricate architectural details. This innovative approach transcends simple line detection by integrating advanced techniques, including semantic segmentation using deep learning, refinement through generative adversarial networks (GANs), and heuristic-based simplifications. This methodology not only captures finer details but also enhances the scalability of vectorization processes for digital applications, making significant strides over traditional methods [4].

Moreover, researchers such as Liu et al. have made significant contributions by improving vectorization accuracy using convolutional neural networks (CNNs) and integer programming to meticulously connect detected architectural features [2]. This development marks a significant departure from earlier methods and offers improved precision and recall in the extraction of floor plan features. Such advancements open new avenues for automating the processing of architectural drawings, thereby facilitating more efficient digital archiving and editing.

Recent innovations also include the application of style transfer techniques for vectorization. Isola et al. introduced the use of Conditional GANs for style transfer [14], effectively altering the style of floorplans while preserving their structural information. This approach is particularly beneficial for unifying the style of diverse floorplan drawings, which simplifies subsequent vectorization and analysis tasks.

Another significant advancement is the introduction of the VectorFloorSeg system by Yang et al., which employs a two-stream Graph Attention Network to improve the segmentation and vectorization of roughcast floorplans. This method not only enhances the precision of the vectorization process but also contributes to more structured and analyzable vector outputs [5].

The transition to more intelligent and adaptable vectorization systems signifies a major advancement in the ability to handle a broader spectrum of complexities found in floor plans. The integration of machine learning not only mitigates the limitations of traditional image processing techniques but also significantly enhances the automation and refinement of the vectorization process. This progress expands the practical applications of vectorized outputs in architectural, engineering, and construction industries, promising more robust tools for professionals in these fields.

B. Image Processing Techniques and Their Limitations

Image processing techniques play a crucial role in the vectorization of architectural floor plans, but they are not without limitations. Traditional methods, which have been heavily reliant on heuristic approaches, often struggle with the diversity and complexity of floor plan designs. These traditional techniques typically employ a variety of strategies such as edge detection, thresholding, and morphological operations to extract meaningful features from raster images. While effective in certain scenarios, these methods can falter with images that contain noisy data or intricate details, often leading to incomplete or inaccurate vectorization.

The challenges are further compounded by the diverse nature of architectural drawings. For instance, floor plans can vary significantly in style, notation, and quality depending on the source, which may include scanned paper plans or digitally created documents. Each type presents unique challenges, such as variations in line thickness, superimposed text and symbols, or faded lines in older documents. These factors often necessitate tailored approaches for different types of images, limiting the scalability of traditional processing techniques.

Recent advancements in deep learning have provided new avenues for addressing these limitations. For example, the integration of convolutional neural networks (CNNs) has been shown to enhance feature recognition capabilities significantly. However, while these models offer improved accuracy and adaptability, they require substantial computational resources and extensive training data, which can be a barrier to implementation. Additionally, deep learning models may still struggle with highly irregular patterns and non-standard layout designs often found in customized or non-traditional architectural drawings.

Moreover, a novel approach proposed by Kim et al. explores the use of style transfer techniques to preprocess floor plans into a unified style before vectorization [6]. This method shows promise in handling complicated drawings characterized by overlapping graphics and irregular notation, which are typical issues that confound standard image processing techniques. By converting various formats of floor plans into a unified style using conditional generative adversarial networks (cGANs), the vectorization process becomes more straightforward and less susceptible to the common errors of traditional methods. However, this technique, while innovative, still faces challenges in consistency across diverse architectural styles and must be finely tuned to manage the subtleties of different building element representations.

In summary, while image processing techniques have evolved considerably, their effectiveness is often curtailed by the inherent limitations of dealing with complex and diverse architectural data. The ongoing development of machine learning models presents potential solutions but also introduces new challenges in terms of data requirements and computational demands. As these technologies continue to advance, further research is needed to fully harness their capabilities and mitigate their limitations.

C. Advances in Convolutional Neural Networks for Image Analysis

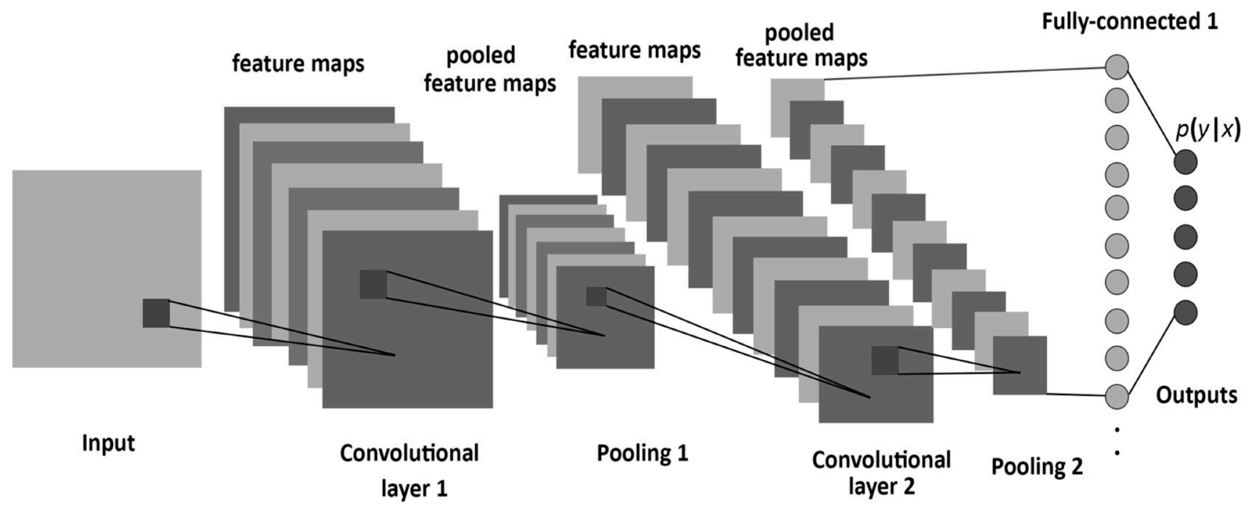

Figure 4: Convolutional Neural Network Architecture [9]

Convolutional Neural Networks (CNNs) have emerged as a cornerstone in the field of deep learning, particularly revolutionizing tasks in image analysis due to their unique architectural features. These networks leverage layered convolutions, which are mathematical operations that filter input data to extract increasingly complex features in hierarchical layers. This process is pivotal in handling high-dimensional data like images, where traditional neural networks falter due to the sheer volume of connections and parameters involved.

CNNs are distinguished from traditional neural networks by their ability to enforce a local connectivity pattern between neurons of adjacent layers, where each connection learns to recognize a specific feature of the input [7]. For instance, in the context of image processing, the first layer might learn to detect edges, while deeper layers might recognize more complex shapes or specific objects [8]. This is achieved through the use of learnable filters or kernels that, when applied across an image, generate feature maps that summarize the presence of detected features from the input.

A significant advantage of CNNs is their use of shared weights in convolutional layers, which drastically reduces the number of parameters, making deep learning models more feasible to train. This parameter sharing also aids in generalizing learned features across different parts of the image, embodying a form of translation invariance, which is crucial for tasks like image classification and object detection.

Recent advancements in CNN architecture have further pushed the boundaries of this technology. For example, innovations such as deep residual learning have allowed the construction of networks that are much deeper than was previously possible, enhancing learning capabilities without a corresponding increase in training complexity. This approach, used in models like ResNet, introduces skip connections that allow gradients to flow through the network directly, mitigating the vanishing gradient problem associated with training very deep networks [8].

Furthermore, novel architectures and techniques continue to refine the efficacy and efficiency of CNNs. The introduction of modules like Inception layers, which allow CNNs to choose from among multiple filter sizes in each layer, provides a way to capture information at various scales and complexities. This adaptability makes CNNs highly effective for a range of tasks from simple image recognition to complex scenarios like scene understanding and medical image analysis.

In the realm of practical applications, CNNs have been pivotal in advancing computer vision, enabling real-world applications such as facial recognition, autonomous vehicle navigation, and automated medical diagnostics. The ability of CNNs to learn from vast amounts of unstructured image data has opened up possibilities that were previously beyond the reach of machine learning technologies.

In conclusion, the continuous evolution of CNN architectures and the integration of new convolutional techniques are expanding the frontiers of what can be achieved with image analysis, pushing forward the capabilities of artificial intelligence in interpreting and understanding visual information. The ongoing research and development in this area suggest that CNNs will remain at the forefront of AI technologies, driving innovations across various sectors of industry and science.

The use of convolutional neural networks (CNNs) in the analysis and vectorization of architectural floor plans represents a significant advancement in the field of computer vision. CNNs have revolutionized how complex patterns and features are extracted from images, providing more robust and accurate vectorization capabilities than traditional methods.

One notable advancement is demonstrated by Liu et al., who implemented CNNs to achieve breakthroughs in corner detection and the connection inference of architectural elements. Their system, designed for consumer-grade floorplan images, achieves over 90% precision and recall by combining CNNs for detecting corners with binary integer programming for connecting these detected features [2]. This approach highlights the potential for CNNs to significantly enhance the vectorization process by improving the accuracy of feature detection and connection inference.

In another development, Song et al. presented a novel vectorization algorithm specifically designed for high-definition blueprints with intricate architectural details. Their method involves a multi-stage process where CNNs play a crucial role in segmenting and refining the vectorization of complex blueprint images. The initial step involves rough semantic segmentation using off-the-shelf algorithms, followed by CNN-driven inference to identify missing smaller architectural components and refine them using a generative adversarial network [4]. This approach underscores the adaptability of CNNs to manage the unique challenges posed by detailed architectural plans, ensuring detailed and precise vector outputs.

Moreover, Zeng et al. introduced a multi-task network that employs room-boundary-guided attention mechanisms within a CNN framework to enhance floor plan recognition [1]. This method demonstrates how CNNs can be specialized to focus on specific elements of a floor plan, such as room boundaries, to improve the overall accuracy and efficiency of vectorization processes.

These examples illustrate the significant role that CNNs have come to play in the vectorization of architectural drawings. By leveraging deep learning techniques, researchers have been able to address some of the traditional challenges associated with vectorization, such as the accurate detection of fine details and the effective processing of complex images. The ongoing advancements in CNN applications continue to push the boundaries of what can be achieved in architectural image analysis, paving the way for more automated and precise vectorization techniques.

D. The Role of OCR in Image Processing

Optical Character Recognition (OCR) plays a pivotal role in the field of image processing, transforming the way text data is extracted and utilized across various applications throughout a spectrum of industries [10]. OCR technology converts different types of text data, such as handwritten, typewritten, or printed text, into machine-readable forms. This conversion is critical for many applications, ranging from automated form processing to intelligent document management systems.

The general function of OCR involves the detection, segmentation, and identification of characters within images. This process begins with pre-processing steps such as noise reduction, normalization, and binarization, where the image is converted into a binary image to simplify the detection of text against the background. Advanced techniques use adaptive thresholding methods to handle different lighting conditions and backgrounds, enhancing the robustness of text detection.

After pre-processing, character segmentation is performed, which isolates individual characters or groups of characters. This segmentation is crucial for the subsequent step of character recognition, where each segment is compared against a database of known characters using pattern recognition algorithms such as neural networks or support vector machines. Modern OCR systems employ deep learning techniques, particularly convolutional neural networks, to improve recognition accuracy even further.

OCR technology has been particularly transformative in sectors that require the digitization of large volumes of documents, such as the legal, banking, and healthcare industries. In legal applications, OCR helps manage case files by converting volumes of printed legal documents into searchable digital formats, thereby speeding up case review processes and reducing physical storage requirements. In banking, OCR facilitates the processing of cheques and financial documents, automating data entry tasks that were traditionally performed manually, thus increasing efficiency and reducing error rates.

Moreover, in healthcare, OCR technology is used to digitize patient records and prescriptions, integrating them into electronic health records systems. This digitization supports better data management, improves the accessibility of information for medical staff, and enhances patient care by providing quick access to patient histories.

In the context of architectural floor plans, OCR is employed to remove non-geometric elements such as texts and annotations from the images before they undergo further processing for vectorization. This removal is crucial as it ensures that the vectorization process focuses purely on architectural elements like walls and doors without the interference of text, which might be misinterpreted as architectural details.

The advancements in OCR technologies have expanded its application scope, facilitating the extraction and digital transformation of textual content from images across various fields. This not only enhances data accessibility and utility but also significantly improves the efficiency of information management systems.

III. Methodology

A. Data Collection and Source

The foundation of this research hinges on the comprehensive collection and careful selection of architectural floor plans. For this study, I am utilizing publicly available floor plans from the University of California, Santa Cruz (UC Santa Cruz). These documents provide a detailed representation of various campus buildings, including academic facilities, residential structures, and administrative offices. The choice of UC Santa Cruz as a data source is strategic, offering a diverse array of building layouts and designs that are essential for a robust analysis.

These floor plans are sourced directly from the UC Santa Cruz's official websites, where they are made available for public access. This transparency not only facilitates academic and research purposes but also ensures that the data used in this study is up-to-date and representative of real-world architectural practices. Each floor plan is provided in a high-resolution raster format, which presents both a challenge and an opportunity for the vectorization process.

The collection process involved systematically downloading these floor plans, ensuring each file was correctly labeled with the building's name and its specific use within the campus. This meticulous approach to data collection is crucial for maintaining the integrity and organization of the research dataset, which in turn supports the subsequent stages of image processing and vectorization.

By utilizing these publicly accessible documents, this research adheres to legal and ethical standards, avoiding any issues related to copyright or restricted access. Moreover, the wide variety of floor plans available from a single, cohesive source allows for a controlled study of vectorization techniques across different architectural styles and functional requirements of campus buildings. This methodological choice not only enriches the research but also enhances its applicability to educational campus planning and management.

B. Preprocessing of Raster Floor Plan Images

The preprocessing of raster floor plan images is a critical step in preparing the data for effective vectorization. This phase involves several techniques aimed at enhancing the quality of the images and ensuring that only relevant architectural features are retained for analysis.

1. OCR Implementation with Keras for Text Removal

The preprocessing phase of raster floor plan images plays a critical role in the successful vectorization of architectural details. A key challenge in this process is the removal of text labels, which, although essential for human readers, are often a source of noise that can disrupt automated analysis. To address this, we employ the pre-trained Optical Character Recognition (OCR) model provided by `keras_ocr`, which is specially designed to detect and decode text within images effectively.

In our methodology, the first step involves setting up an OCR pipeline using `keras_ocr`. This pipeline is configured to utilize a deep learning model that has been trained to recognize textual content within a wide variety of images. The model's robustness makes it particularly suitable for dealing with the complex backgrounds and varied text styles found in architectural floor plans.

Once the pipeline is in place, each floor plan image is processed through this system. The model scans the entire image, identifying regions where text is present. For each detected text region, the model provides bounding box coordinates, which precisely define the location and extent of text within the image. These bounding boxes are crucial as they determine the areas that need to be addressed in the subsequent steps of the preprocessing phase.

After text detection, the next step is to create a mask for the text areas identified by the OCR model. This mask is applied over the original image to cover all detected text regions. With the mask in place, we then apply an inpainting technique, which is a method used to reconstruct the areas obscured by the mask. Inpainting works by using information from the surrounding pixels to fill in the masked areas, effectively removing the text while preserving the continuity of the architectural elements in the image.

The inpainting process is delicate as it must ensure that the filled areas blend seamlessly with the rest of the image, without leaving any traces of the original text or creating visual artifacts that could interfere with the accuracy of the vectorization process. The success of this step is critical as it produces a cleaned image that retains all necessary architectural details but without any of the disruptive textual information.

Finally, the processed image, now cleared of text, is saved and prepared for the next stages of vectorization. This clean version of the floor plan is far more suitable for automated analysis, as it allows the vectorization algorithms to focus purely on architectural features without the interference of extraneous text. By using the `keras_ocr` model for text removal, we streamline the preprocessing of floor plans, ensuring high-quality inputs for our vectorization process, which in turn enhances both the efficiency and accuracy of our architectural analyses.

2. Low-Level Image Processing Technique for Wall Filtering

In the development phase of a vectorization technique, I experimented with a low-level image processing approach designed to enhance the clarity of walls in architectural floor plans. This method incorporated multiple OpenCV image processing techniques to refine the representation of walls, differentiating them more clearly from other elements within the images. Although this approach was ultimately not included in the final vectorization pipeline, it provides valuable insights into the complexity of image processing required for architectural drawings.

The process began with Otsu's thresholding, a technique that converts a grayscale image into a binary image by determining an optimal threshold value. This threshold value is chosen to minimize the intra-class variance of the black and white pixels, effectively highlighting architectural features like walls, which typically appear as darker lines against a lighter background.

Following the thresholding, the technique employed noise removal through morphological operations, specifically using an opening operation composed of an erosion followed by dilation with a 3x3 kernel. The erosion helps remove small-scale noise by eroding away the boundaries of foreground regions, which include the walls, thereby eliminating isolated noise points. Subsequently, dilation is applied to smooth and restore the edges of the wall features, ensuring that the erosion does not overly diminish their presence. This sequence of opening—applying erosion followed by dilation—is performed twice to effectively reduce noise while preserving the integrity of larger structural elements like walls.

To further refine the visibility of walls, additional dilations were performed. This step thickens the walls in the binary image, enhancing their prominence and ensuring they are distinct from minor artifacts or non-structural lines that might remain post-noise reduction.

Additionally, a distance transform was utilized, which calculates the minimum distance from each foreground pixel to the nearest background pixel. By applying a threshold to this distance-transformed image, the method identifies regions most likely to be walls (sure foreground) and distinguishes them from less certain areas. This segmentation is crucial for reinforcing the definition of walls and ensuring that the vectorization process captures these critical architectural elements accurately.

This low-level image processing technique, though not included in the final approach, helped us understand the challenges and potential solutions in processing complex architectural drawings, informing further development of more sophisticated methods like DeepFloorPlan for effective floor plan vectorization.

C. Application of the DeepFloorPlan Algorithm

The DeepFloorPlan algorithm significantly enhances the precision of architectural floor plan vectorization, outperforming traditional low-level image processing techniques through its use of advanced deep learning strategies. Grounded in the robust methodologies from the study "Deep Floor Plan Recognition using a Multi-task Network with Room-boundary-Guided Attention," DeepFloorPlan employs a convolutional neural network (CNN) to adeptly recognize and delineate various architectural elements. [1] This deep learning approach is superior in handling the complexities inherent in diverse architectural drawings, which often elude simpler, rule-based processing methods. The CNN effectively learns from a vast dataset of images, allowing it to adapt to and accurately process a wide array of floor plan designs and complexities. Moreover, DeepFloorPlan integrates a room-boundary-guided attention mechanism, which significantly enhances its ability to focus on and accurately segment crucial architectural features, an aspect where traditional methods generally falter. This capability ensures a higher level of precision and adaptability in processing floor plans, marking a substantial improvement over the conventional image processing approaches.

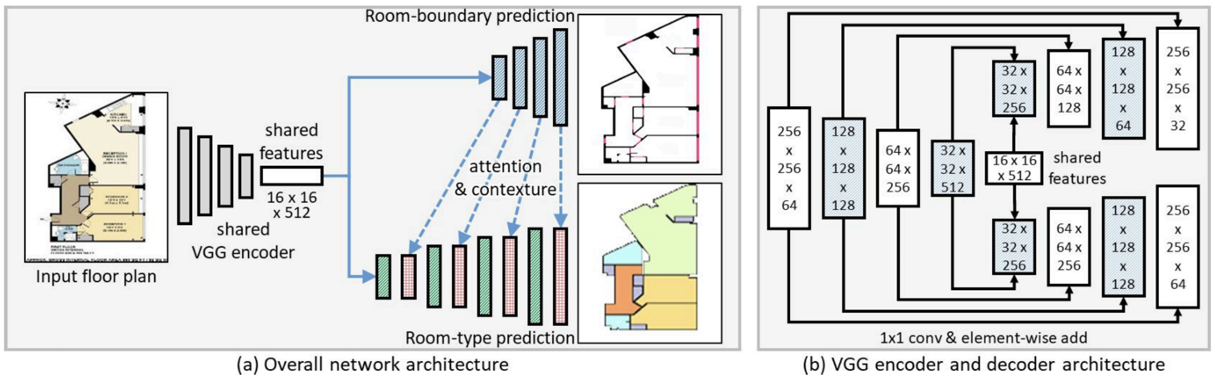

Figure 5: DeepFloorPlan Architecture [1]

1. Description of Convolutional Neural Network-based DeepFloorPlan

DeepFloorPlan is a multi-task CNN that focuses on recognizing and differentiating between various elements of a floor plan, such as walls, doors, and different types of rooms. It employs a room-boundary-guided attention mechanism, which enhances the accuracy of room type prediction by focusing on the spatial relationships and boundaries within the floor plan. This mechanism allows the network to handle complex layouts with varying wall thicknesses and non-rectangular shapes more effectively than traditional image processing methods.

A significant feature of DeepFloorPlan is its ability to process images by segmenting them into meaningful categories. For instance, in its typical operation, it segments rooms by type, coloring different room types with different colors. However, for the purposes of vectorization where such differentiation is unnecessary, we modify the output to render walls in black and all other elements in white. This simplification focuses the vectorization process on the structural elements of the floor plan, which are crucial for accurate digital representation.

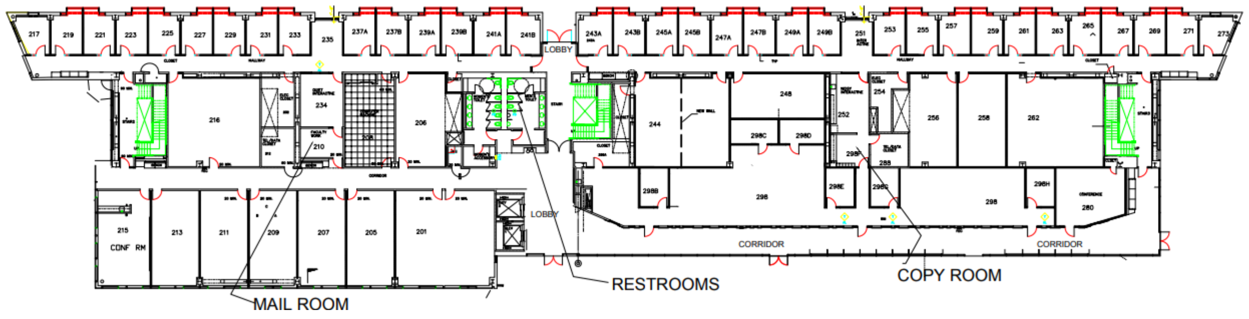

Figure 6: UCSC Engineering 2 Floor 2 floor plan image [23]

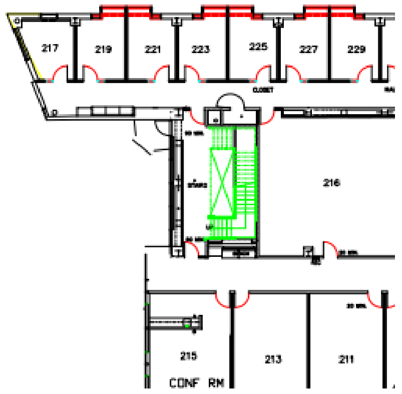

Figure 7: A tile

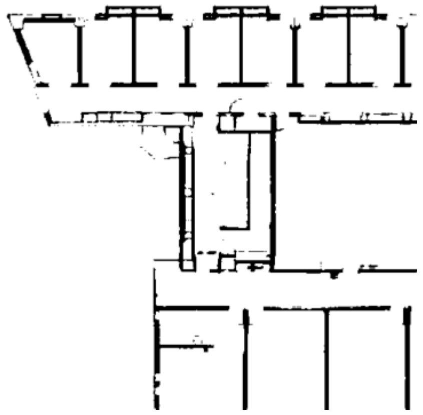

Figure 8: The tile from Figure 7 after being processed by DeepFloorPlan

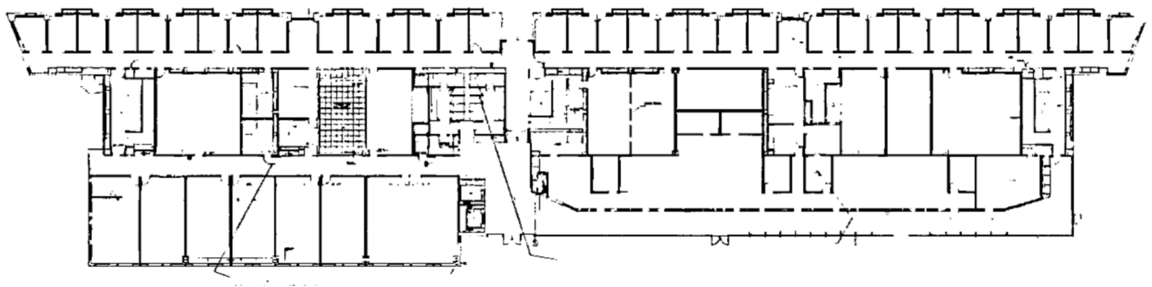
Figure 9: After all tiles are reassembled

2. Implementation of Tiling Algorithm for Processing Large Images

Given the constraints of neural networks in handling large images directly due to computational limits and the loss of resolution when scaling images down, we implement a tiling algorithm. This algorithm divides the original large floor plan images into smaller, manageable tiles, each of which can be individually processed by the DeepFloorPlan algorithm without the loss of detail that would occur from resizing the entire image.

The tiling process involves segmenting the image into squares of a fixed size (e.g., 256x256 pixels), processing each tile independently through DeepFloorPlan, and then reassembling the processed tiles to form the complete image. This method ensures that each portion of the floor plan is analyzed with the highest possible accuracy and detail, maintaining the integrity of the walls and boundaries throughout the image.

To handle the potential issue of discontinuities between tiles, particularly at the boundaries where architectural features might be split across tiles, special care is taken in the reassembly process. We ensure that the reassembly maintains the continuity of architectural features, aligning segments perfectly to reconstruct the original layout accurately.

The integration of the DeepFloorPlan algorithm with the tiling strategy enables us to effectively process floor plans of any size, overcoming the typical limitations associated with CNNs regarding input dimensions. This approach not only enhances the precision of the vectorization process but also scales efficiently to handle large datasets of architectural drawings.

D. Post-processing Techniques for Vectorization

While none of the post-processing techniques were ultimately utilized in the final vectorization approach for this project, exploring such methods provides valuable insights into potential applications for extracting additional information from floor plans.

Techniques like Harris Corner Detection and the Shi-Tomasi Corner Detection Method are critical in computer vision for identifying features within an image that are pivotal for various analysis tasks.

1. Harris Corner Detection Technique

The Harris Corner Detection technique, developed by Chris Harris and Mike Stephens in 1988, is a widely used method for identifying corners and edges in an image. [15] This method relies on the principle that corners are characterized by significant changes in intensity in all directions. The Harris Corner Detector computes a corner response function $E(u,v)$ for each pixel in the image, which measures the change in intensity for a displacement of (u,v) in all directions:

$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u, y+v) - I(x,y)\right]^2$$

where $w(x,y)$ is the window function, $I(x+u, y+v)$ is the shifted intensity, and $I(x,y)$ is the intensity.

To maximize this function $E(u,v)$ for corner detection, the Harris Corner Detector applies Taylor Expansion to the equation and uses mathematical steps to derive the final equation:

$$E(u,v) \approx [u \quad v] M \left[u \ v\right]$$

where:

$$M = \sum_{x,y} w(x,y) \left[I_x I_x \quad I_x I_y \ I_x I_y \quad I_y I_y\right]$$

Here, I_x and I_y are image derivatives in x and y directions respectively, which can be easily found using cv.Sobel().

After this, they create a score, basically an equation, which determines if a window can contain a corner or not:

$$R = \det(M) - k(\operatorname{trace}(M))^2$$

where $\det(M) = \lambda_1 \lambda_2$, $\operatorname{trace}(M) = \lambda_1 + \lambda_2$, and λ_1 and λ_2 are the eigenvalues of M. The magnitudes of these eigenvalues decide whether a region is a corner, an edge, or flat.

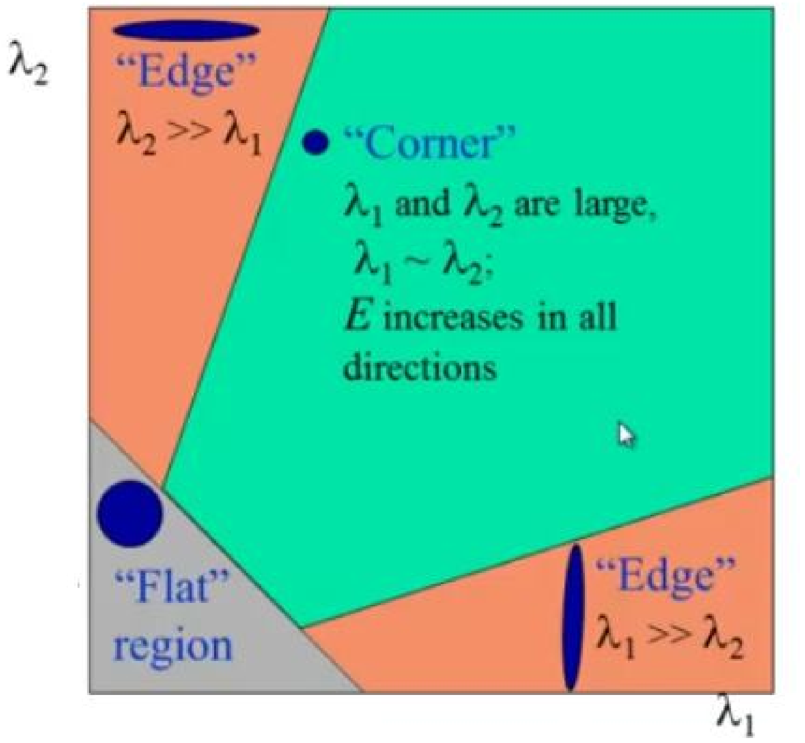

Figure 10: Harris Corner Detection region categorization based on magnitude of eigenvalues [18]

The Harris Corner Detector is robust to rotation and illumination changes but may struggle with scale invariance. It is computationally efficient and has been widely used in various computer vision applications, such as object tracking, image registration, and 3D reconstruction.

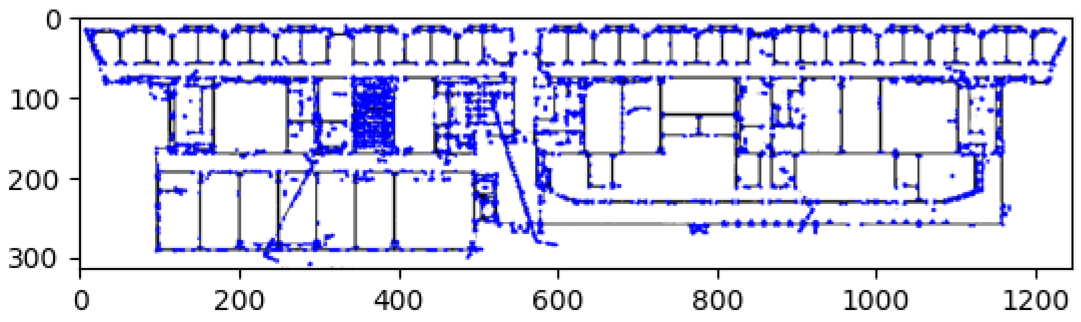

Figure 11: Harris Corner Detection applied to floor plan image

2. Shi-Tomasi Corner Detection Method

Later in 1994, J. Shi and C. Tomasi made a small modification to the Harris Corner Detector in their paper "Good Features to Track," which shows better results compared to the Harris Corner Detector. [16] The scoring function in the Harris Corner Detector was given by:

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

Instead of this, Shi-Tomasi proposed:

$$R = \min(\lambda_1, \lambda_2)$$

If R is greater than a threshold value, it is considered as a corner. If we plot it in $\lambda_1 - \lambda_2$ space as we did in the Harris Corner Detector, we get an image where only when λ_1 and λ_2 are above a minimum value, $\lambda_{\min}$, it is considered as a corner (green region of the below image).

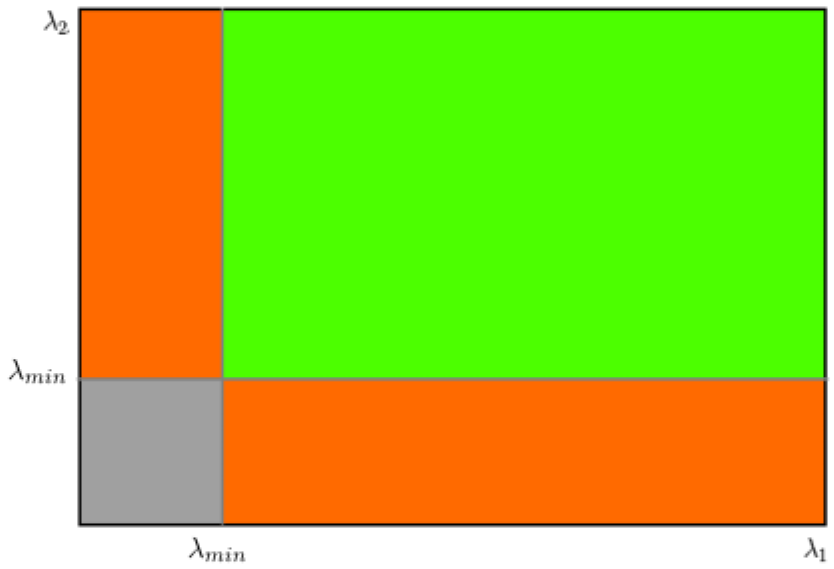

Figure 12: Depiction of the thresholding principle used in the Shi-Tomasi corner detection method, visualized in the eigenvalue space. The green area represents the region where both eigenvalues (λ1 and λ2) exceed a minimum threshold (λ_{min}), identifying a strong corner. The orange and gray areas show where one or neither of the eigenvalues surpass the threshold, thus not qualifying as corners. [19]

The Shi-Tomasi Corner Detection Method shares similar properties with the Harris method, such as rotation and illumination invariance. It is also computationally efficient and has been used in various computer vision tasks, including feature tracking and image matching.
In the context of floor plan vectorization, corner detection techniques like Harris and Shi-Tomasi can be valuable for identifying key points and features within the floor plan image. These corners can serve as reference points for further analysis, such as room segmentation, wall detection, or symbol recognition. However, the specific application of these techniques would depend on the characteristics of the floor plan images and the desired level of detail in the vectorized output.

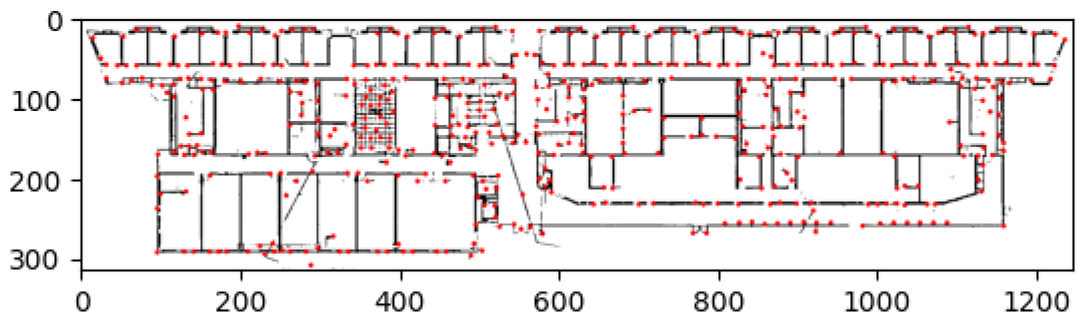

Figure 13: Shi-Tomasi Corner Detection applied to floor plan image

3. Bitwise Masks for Junction and Centroid Detection

In the realm of digital image processing for floor plan analysis, identifying the structure of spaces often necessitates discerning the lines and intersections that represent walls, doors, and other defining architectural features. A critical step in this process involves the extraction of horizontal and vertical lines which, in architectural drawings, typically delineate the boundaries of rooms and corridors.

We use kernels designed to target specific shapes. For horizontal lines, a horizontal structuring element, elongated across the x-axis, helps to highlight and preserve lines that run left-to-right. Conversely, a vertical structuring element, extended along the y-axis, is used to emphasize lines that stretch top-to-bottom. These operations serve to clean the image of noise, ensuring that the horizontal and vertical elements stand out clearly.

In addition to horizontal and vertical lines, diagonal lines can also hold significant structural information. Custom-shaped kernels, akin to diagonal matrices, are crafted to target these lines. One kernel is oriented for one diagonal direction, and by flipping this kernel horizontally, we can create a second one for the opposite diagonal.

Once these directional lines are defined, the next objective is to identify junctions—points where two lines intersect, often corresponding to corners of rooms or intersections within a network of hallways. This is achieved through bitwise operations that act like logical functions on the pixel values of two images. By performing a bitwise 'AND' operation between the images containing horizontal and vertical lines, we isolate the points where they intersect. This operation is repeated with the images of diagonal lines, capturing the full spectrum of potential junctions.

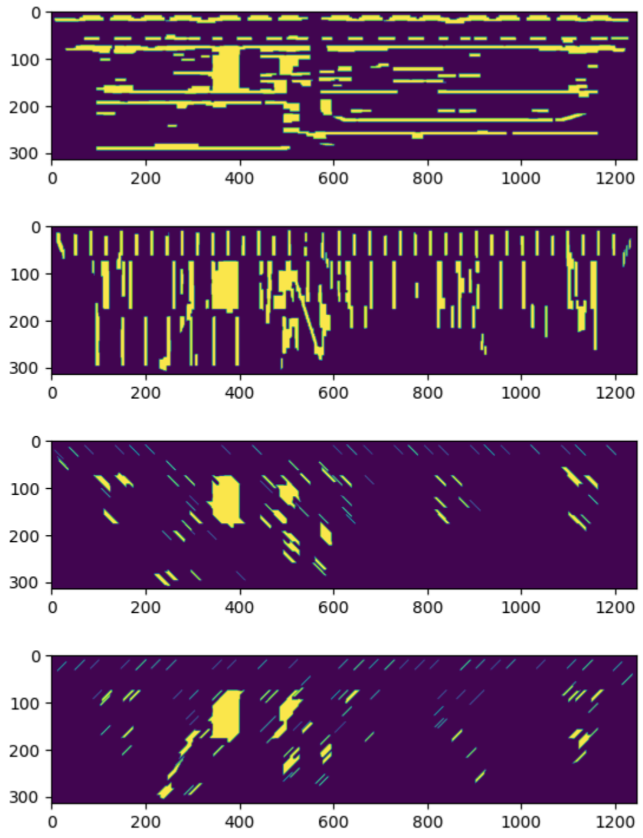

Figure 14: Lines found by kernels that target specific line directions (horizontal, vertical, diagonal)

The culmination of this process is the aggregation of all intersections to locate all junctions. By performing a bitwise 'OR' operation between the various intersection images, we combine the detected intersections into a single image that reveals the network of junctions within the floor plan.

Finally, we turn our attention to the centroids of these junctions, which are essentially the geometric centers of the shapes formed by the intersections. By finding the contours of these junctions and computing their image moments, we can calculate the centroids' coordinates. A centroid represents the average position of all the points of an object and is calculated as the weighted average of the pixel intensities. Drawing these centroids onto the image provides a visual representation of the structure of the space, completing the process of highlighting the foundational grid upon which the floor plan is built.

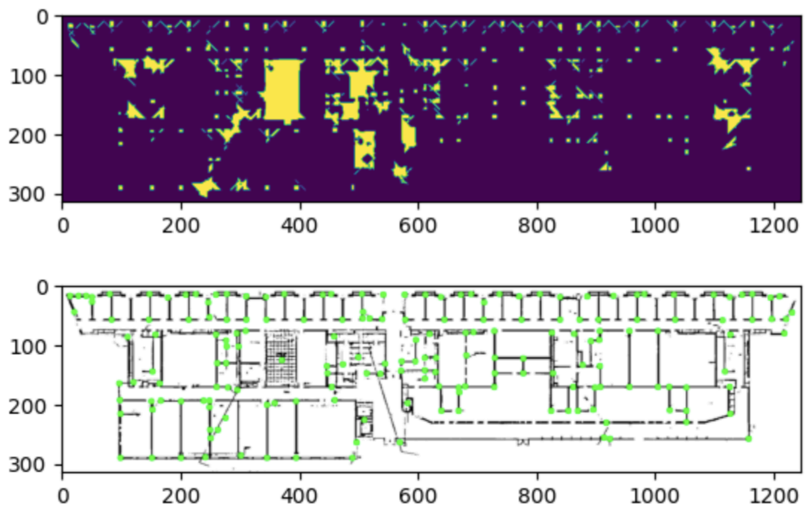
Figure 15: Junction points and intersections found by bitwise operations

This approach, while not included in the final vectorization technique, is instrumental in understanding the geometry and connectivity of spaces within floor plans. The ability to accurately detect lines and their intersections is crucial not only for creating digital representations of physical spaces but also for more advanced analyses, such as space optimization and automated design assessments.

4. Contour Detection with OpenCV

Contour detection is a fundamental process in computer vision, particularly useful in the field of architectural analysis and digital floor plan vectorization. This technique is instrumental in delineating the outlines or boundaries of features within an image, which is a crucial step in understanding the geometry and spatial organization of the depicted elements.

Using OpenCV, a prominent library in the field of computer vision, contour detection is typically performed with the `findContours` function. This function examines the binary representation of the image, where the architectural features have been isolated from the background, to detect continuous curves that encapsulate the full perimeter of distinct objects. In the context of floor plans, these objects could be rooms, furniture, or other significant design elements.

The `findContours` function operates by scanning through an image and identifying the regions where the color or intensity changes dramatically, often indicating the edges of features against the background. Once these edges are detected, the function groups the series of points along these edges into contours. Each contour is essentially a vector of coordinates that define the shape of an object within the image.

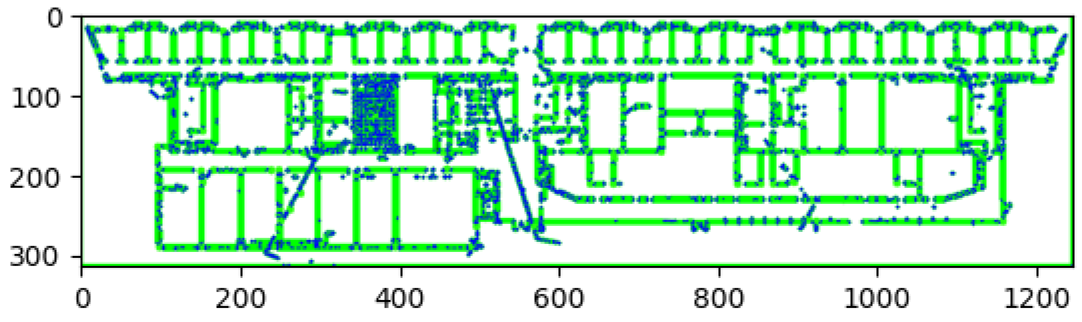

Figure 16: The blue points are the discovered contours

After detecting contours, they can be utilized for various applications, such as creating bounding boxes, shape analysis, or object recognition and classification. In architectural vectorization, the precise coordinates of these contours are invaluable, as they provide the vector data necessary to reconstruct accurate digital models of the physical spaces.

To further leverage the information that contours provide, one might write a script that translates these contour coordinates into a format suitable for data analysis, like a CSV (Comma-Separated Values) file. By exporting contour data into a CSV, we can facilitate the integration of this spatial data with other analysis tools or workflows. For instance, the contours can be imported into CAD (Computer-Aided Design) software for further refinement, or they can be analyzed programmatically to calculate areas, perimeters, or other attributes that are essential for architectural planning and evaluation.

The utility of contour detection in architectural floor plan processing cannot be understated. It is a bridge between the raw pixel data of an image and the structured, geometric information required for digital modeling and analysis, making it a cornerstone of the digital transformation of architectural design data.

5. Skeletonization Process

Skeletonization is a process in digital image processing that reduces foreground regions in a binary image to a skeletal remnant that largely preserves the extent and connectivity of the original region while throwing away most of the original foreground pixels. The skeleton represents the shape of the figure in a simplified form, which can be crucial in understanding the structural layout of floor plans, such as the path network within a building.

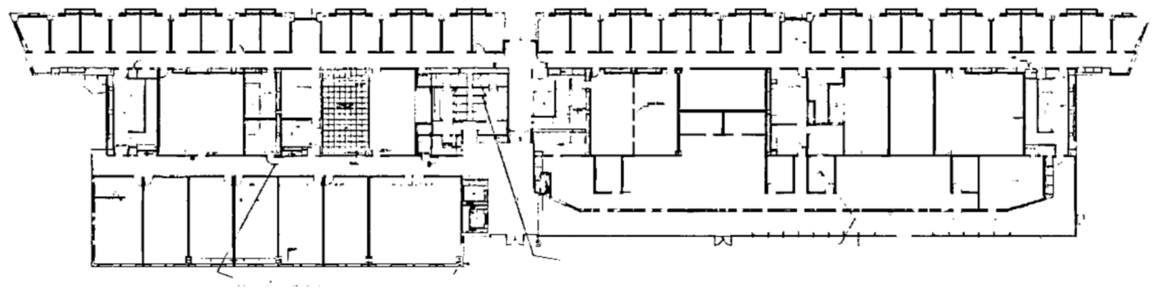
Figure 17: UCSC Engineering 2 Floor 2 floor plan image

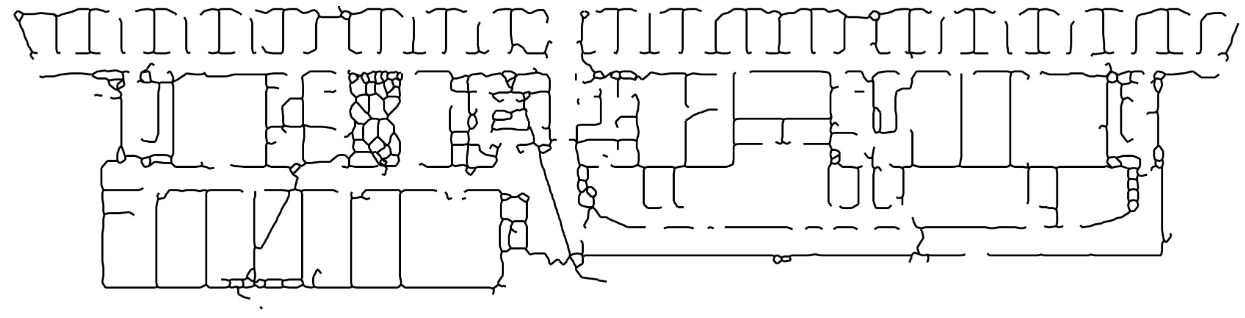
Figure 18: Skeletonized image

In OpenCV, two well-known algorithms for skeletonization are the Zhang-Suen and Guo-Hall methods. Both techniques iteratively examine and thin the given shapes until only the minimal skeletal structure remains.

The Zhang-Suen thinning algorithm [20] is an iterative thinning process that looks at the local neighborhood of each pixel in a binary image and decides whether or not it should be removed based on specific conditions related to the number of foreground-to-background transitions in the neighborhood, and the number of foreground neighbors. The process is repeated until no further changes occur in the image. The Zhang-Suen method is precise and tends to preserve the topology of the original image well, making it a good choice for applications where the accurate representation of the shape is important.

The Guo-Hall algorithm [21] is another iterative method but uses a slightly different set of conditions for the removal of pixels. It tends to be faster than the Zhang-Suen method but might not preserve the topology as well as Zhang-Suen. The Guo-Hall algorithm can sometimes result in a less noisy skeleton and is often used when speed is a crucial factor, or the final application can tolerate some topological alterations.

The primary difference between these two methods lies in their specific conditions for pixel removal and the patterns they recognize as removable. The Zhang-Suen method is known for its detail preservation but slower performance, while the Guo-Hall method is favored for its speed and smoother results.

The result of the skeletonization process is a thin version of the original image, where the width of the shapes is reduced to the minimal possible width, ideally one pixel wide. This is particularly useful in architectural plans where such skeletal representations can serve as a basis for analyzing routes, flows, or the spatial relationship between different components. These skeletal paths are invaluable in several applications, including feature extraction, pattern recognition, and in the creation of topological maps of spaces.

6. Line Segment Detection Methods

Line Segment Detection (LSD) is an essential computer vision technique with significant implications in the field of architectural design and analysis. It is particularly adept at identifying and extracting line segments from images, which is invaluable for interpreting architectural drawings and floor plans where lines define the boundaries of structures and spaces.

The Line Segment Detector in OpenCV is a popular algorithm used for detecting line segments. The LSD algorithm operates directly on grayscale images and does not require a preliminary edge detection stage, unlike some other line detection methods like the Hough Transform. It is designed to be scale and rotation invariant, meaning that it can detect lines over a range of orientations and scales, making it quite robust for architectural applications where floor plans can be at various scales and orientations.

The LSD works by examining an image to identify rapidly changing regions of intensity that suggest the presence of an edge. Once these potential edges are found, the algorithm performs a refinement to determine if these are indeed part of a line segment. This involves assessing the gradient orientations of the pixels and ensuring they are consistent with a straight line. It does this efficiently, allowing for real-time application in some contexts.

The OpenCV implementation of LSD provides an interface that returns a list of detected lines, each described by the starting and ending points. This makes it straightforward to overlay these lines onto the original image or to use them for further computational analysis. The lines detected by LSD can be used for creating wireframe models of the detected structures, for converting raster floor plans into vector format, or for recognizing objects defined by straight lines within the image.

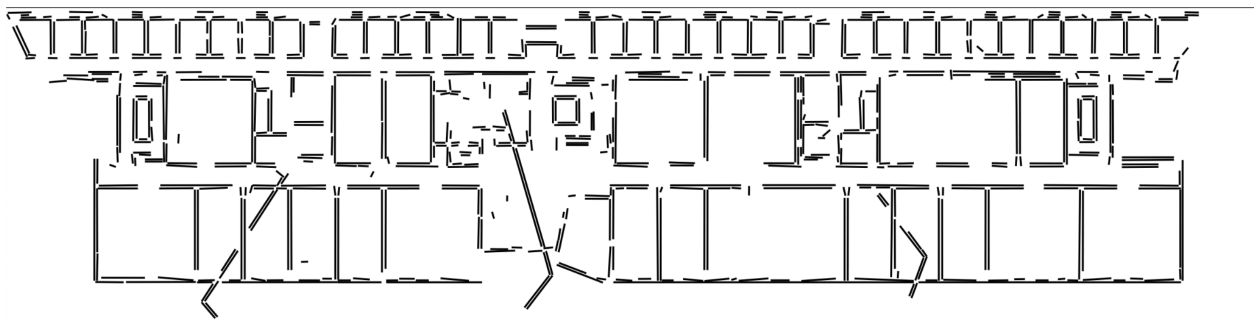
Figure 19: Line Segment Detector applied to UCSC Engineering 2 Floor 2 floor plan image

LSD is particularly advantageous when a high degree of accuracy is required in the line detection process. It can discern short line segments, which other detectors might miss, and distinguishes between different line segments that are close to each other. These features make it especially useful for detailed and accurate analysis of architectural drawings, where precision in the delineation of structures is paramount.

Integration of Multiple Techniques for Enhanced Vectorization

1. Raster to Vector

Raster to Vector (R2V) transformation represents a cutting-edge shift from traditional raster floor plan interpretation to a vectorized format that is more conducive to further analysis and manipulation. [2] R2V employs a neural network architecture to dissect and understand the complex imagery of floor plans.

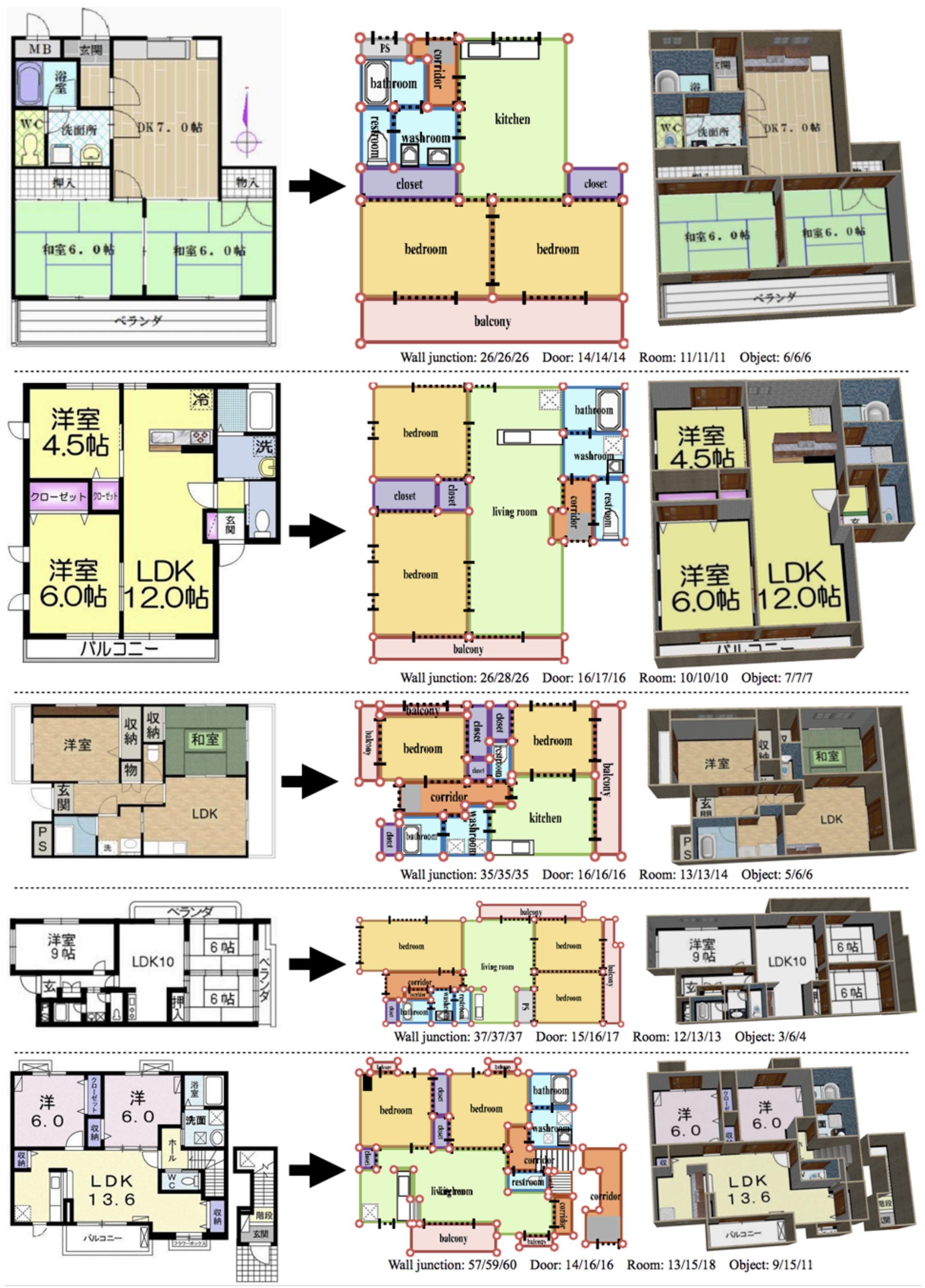
Wall junction: 26/26/26 Door: 14/14/14 Room: 11/11/11 Object: 6/6/6
Wall junction: 26/28/26 Door: 16/17/16 Room: 10/10/10 Object: 7/7/7
Wall junction: 35/35/35 Door: 16/16/16 Room: 13/13/14 Object: 5/6/6
Wall junction: 37/37/37 Door: 15/16/17 Room: 12/13/13 Object: 3/6/4
Wall junction: 57/59/60 Door: 14/16/16 Room: 13/15/18 Object: 9/15/11

Figure 20: Raster to Vector Floorplan Vectorization Results [2]

In the initial phase, R2V leverages a neural network to pinpoint crucial low-level geometric and semantic junctions within the raster image. These junctions include key architectural points such as wall corners or endpoints of doors, translating the pictorial data into a structured form. The process involves identifying these junction points with high precision, laying the groundwork for the subsequent vectorization.

Following junction identification, the process uses integer programming to methodically piece together these points into recognizable architectural primitives. These primitives are not merely abstract lines but carry geometric and semantic significance — for instance, wall lines, door lines, or icon boxes. This integer programming ensures that the resulting primitives adhere to the topological and geometric rules of a floor plan, producing an output that is not only accurate in detail but also consistent in structure.

The final vector representation, arising from this meticulous procedure, enables a range of computational applications. For instance, it allows for the generation of 3D models, facilitates architectural remodeling, and can even serve as a foundation for comprehensive building analysis.

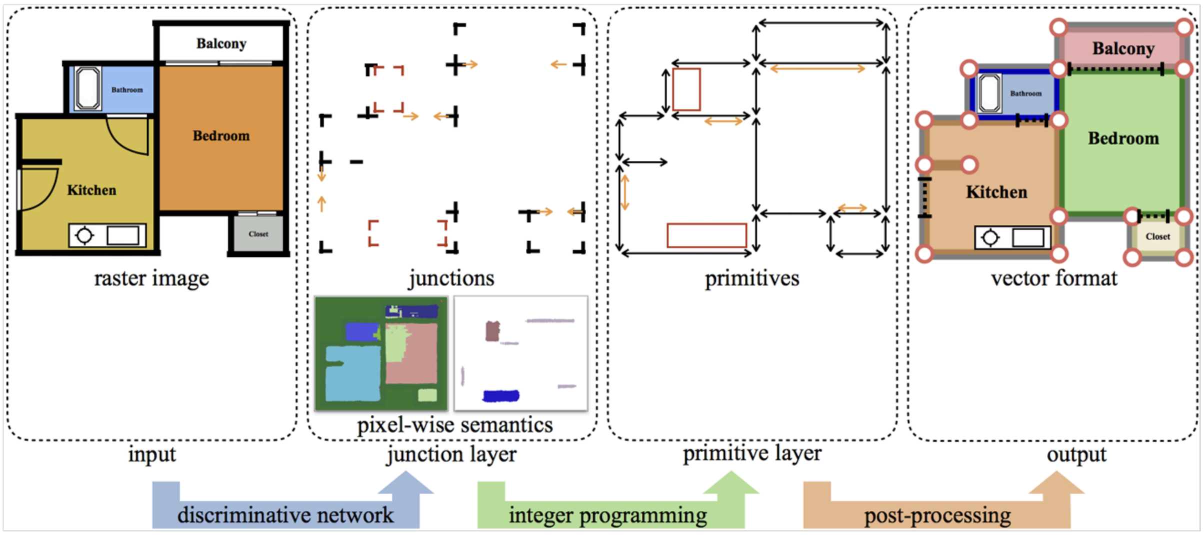

Figure 21: Raster to Vector Conversion Process [2]

Despite the efficacy of R2V in handling original raster floor plans, challenges arise when dealing with output from other processing techniques, such as DeepFloorPlan. When R2V is applied to images already processed by DeepFloorPlan, the performance is noted to decline. This discrepancy could stem from the differences in how DeepFloorPlan and R2V interpret and manipulate the initial raster data. DeepFloorPlan's processing may alter the image in ways that are less compatible with the R2V model, which has been trained on and expects unaltered raster images.

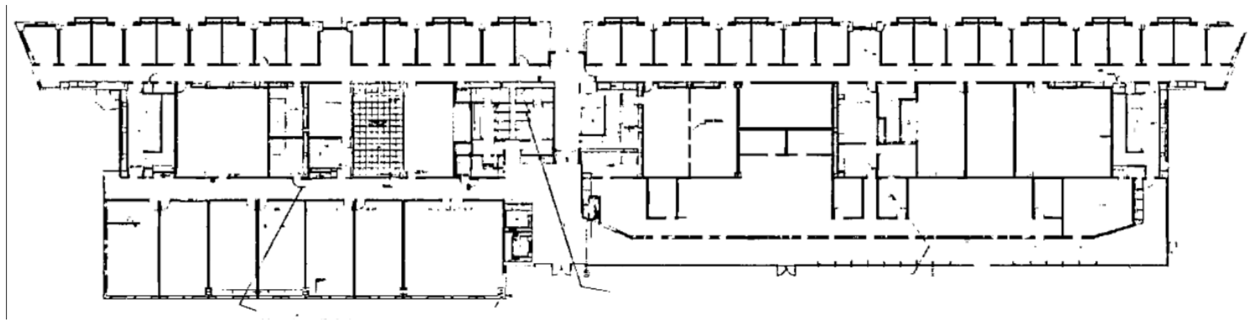

Figure 22: UCSC Engineering 2 Floor 2 floor plan image

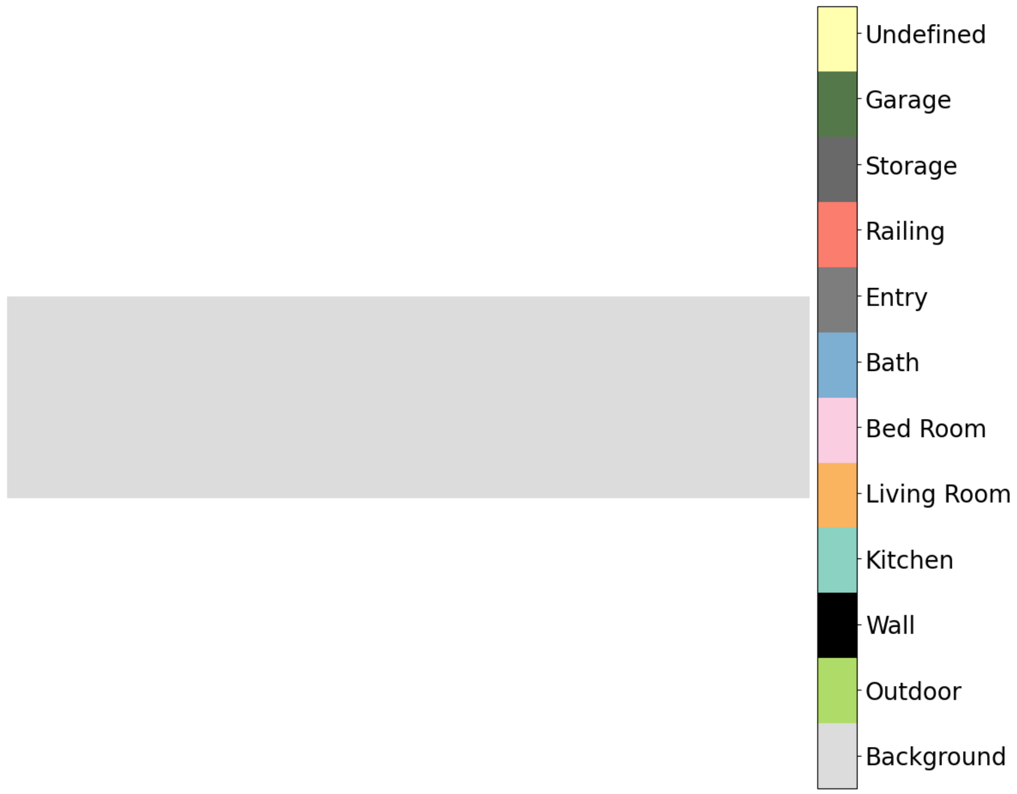

Figure 23: Raster to Vector ran on UCSC Engineering 2 Floor 2 floor plan image

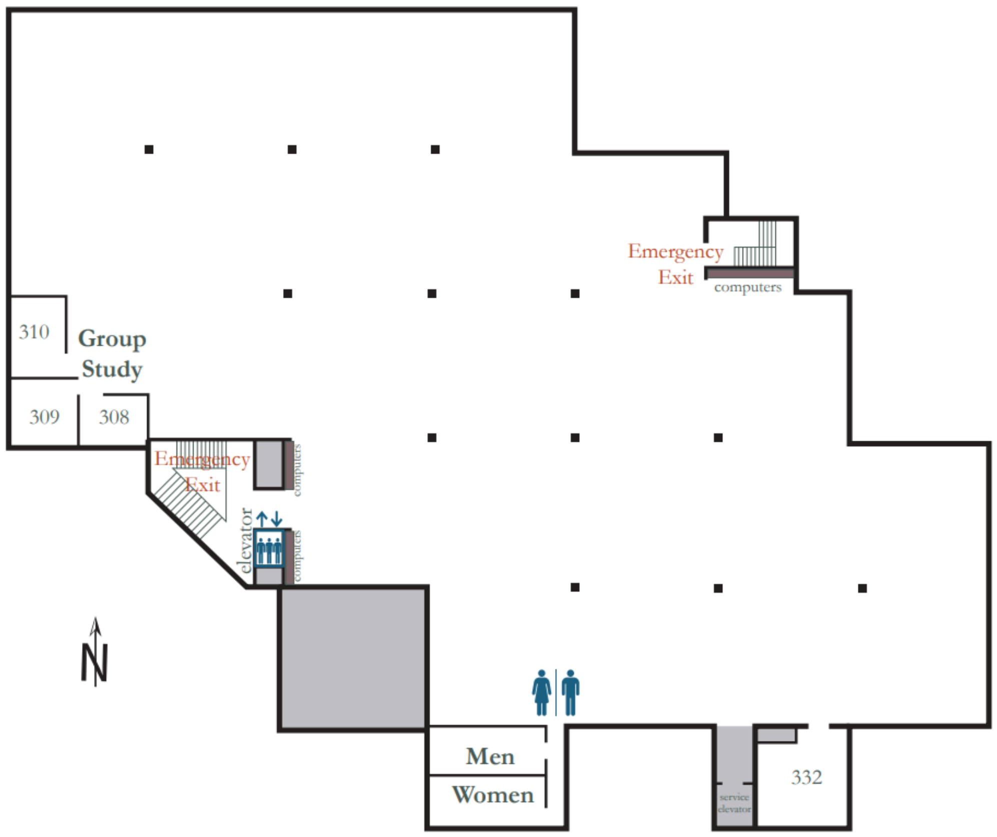

Figure 24: UCSC Science and Engineering Library floor plan image [24]

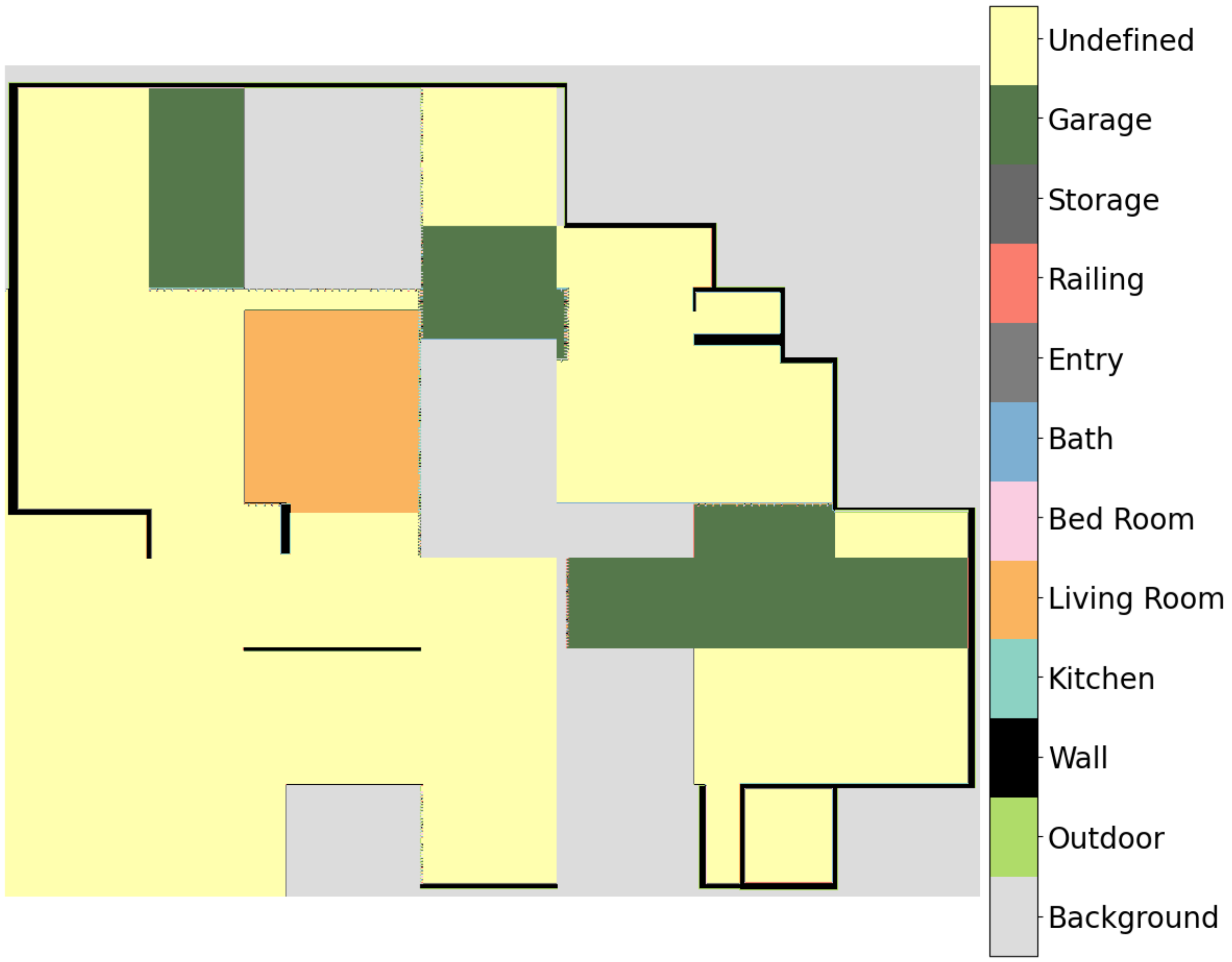

Figure 25: Raster to Vector ran on UCSC Science and Engineering Library floor plan image

An interesting experiment within this vectorization endeavor involved incorporating a tiling algorithm to segment the original floor plan into smaller sections and individually process each through R2V. However, this approach did not yield successful outcomes, which could be attributed to the loss of contextual information when a floor plan is fragmented. The tiling algorithm, while conceptually robust for managing large images, may disrupt the continuity that R2V relies on to accurately infer the full scope of a floor plan's structure.

In conclusion, while R2V represents a significant advancement in the digital interpretation of floor plans, its integration with other techniques must be approached with consideration of their inter-compatibility. The learning-based approach of R2V shows promise in overcoming the limitations of heuristics-based methods, suggesting a potential paradigm shift in how architectural data is processed and utilized in digital environments.

2. Utilization of Probabilistic and Regular Hough Transform

The Hough Transform is an analytical technique integral to the field of image analysis, particularly within the domain of vectorization, where the detection of geometric shapes is essential. It serves as a bridge between the pixel-based data in raster images and the line-based geometric representations in vector images. The technique is based on transforming points in image space to a parameter space where lines or curves can be more easily identified. For the vectorization of architectural floor plans, which are replete with linear elements such as walls, doorways, and furniture outlines, the Hough Transform is invaluable.

There are two primary variants of the Hough Transform: the Standard Hough Transform (SHT) and the Probabilistic Hough Transform (PHT). The SHT works by mapping each edge pixel in the image space to a sinusoidal curve in the parameter space, often represented as the Hough space. The intersections of these curves correspond to potential line segments in the image space. Accumulator cells in the Hough space "vote" for these intersections, and the locations with the highest votes indicate where the most likely line segments exist. SHT is comprehensive and robust, capable of detecting all possible lines, but it is computationally intensive because it considers every point along an edge, leading to a high number of calculations, particularly for large and complex images.

The PHT, on the other hand, offers a more computationally efficient alternative. It is a randomized version of the SHT that only considers a random subset of edge points. It estimates the parameters of line segments present in the image by iteratively selecting point pairs and checking if they could form part of a line with a sufficient number of votes. The PHT is faster than the SHT because it does not process every edge point and does not fill the accumulator space as extensively. However, this probabilistic nature means it may miss some line segments that the SHT would detect, especially if they are short or faint.

In the context of enhancing vectorization, the Hough Transform is integrated into the workflow to detect and standardize the lines and shapes that define the layout of the floor plan. The decision to use SHT or PHT depends on the specific requirements of the project. For instance, if computational resources and processing time are limited, or if only the most prominent lines are of interest, the PHT may be the preferable choice. Conversely, if the floor plan contains many subtle or critical details, the thoroughness of the SHT would be more appropriate.

3. Combining Hough Transform with Junction Detection

In the development of the vectorization pipeline, the incorporation of the Hough Transform in conjunction with junction detection was explored as a potential method to enhance the precision of line detection. This approach aimed to refine the set of lines detected by the Hough Transform by considering only those that intersected with identified junctions, which are indicative of critical architectural points like wall corners. Despite its conceptual merit, this method introduced complexity and was ultimately not included in the final vectorization technique.

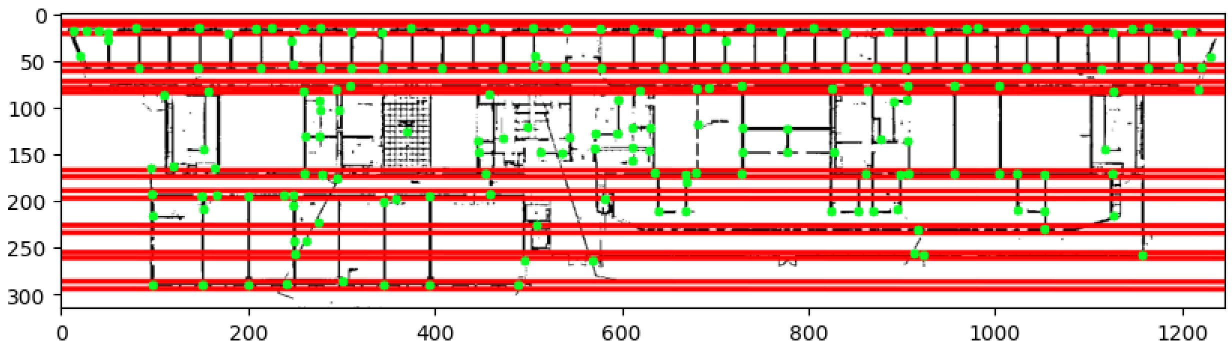

Figure 26: Hough Transform

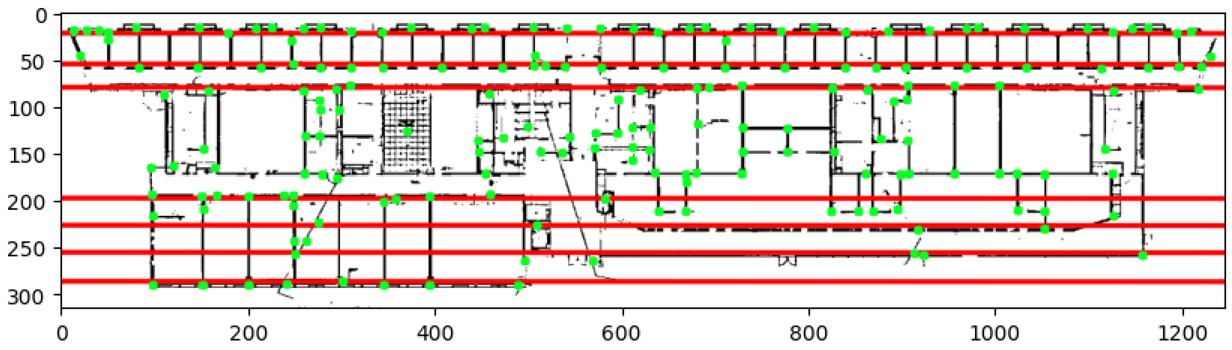

Figure 27: Hough Transform combined with junction intersection

4. Hough Transform with improved non-maximum suppression

A more efficient solution was found in the form of an improved non-maximum suppression algorithm within the Hough Transform process. Non-maximum suppression serves to streamline the output of the Hough Transform by ensuring that each detected line is unique and the strongest representative of that feature within the image. The improvement of this algorithm proved to be a superior approach to managing the issue of duplicate lines—a problem that the combination of Hough Transform and junction detection aimed to address.

The enhanced non-maximum suppression technique works by meticulously analyzing the accumulator space of the Hough Transform to discriminate between lines that are closely spaced. By fine-tuning the Hough parameters, particularly the rho resolution, the process became more sensitive to the distinction between separate architectural features that could erroneously be conflated into a single detection with a coarser resolution. Decreasing the rho value resulted in an increased number of bins in the accumulator array, allowing for a narrower scope within each bin, which is vital for the accurate separation of lines.

To ensure balanced detection of horizontal and vertical lines, the Hough Transform image was transformed into a square format. This transformation equally weights lines of all orientations, preventing any directional bias and enhancing the uniformity of line detection across the image.

This refined approach to non-maximum suppression was instrumental in improving the overall effectiveness of the line detection mechanism. It offered a streamlined, more computationally efficient method that eliminated the need for the prior junction detection step, simplifying the vectorization process while enhancing its accuracy. The adoption of this technique in the final vectorization pipeline underscores the importance of iterative refinement and optimization in computational methods, achieving a balance between comprehensive line detection and the elimination of extraneous information to produce clean, precise vector representations of architectural floor plans.

IV. Results

In the following section, we present a comprehensive analysis of the Hough Transform's performance for line detection in images, focusing on the effects of non-maximum suppression and rho resolution. The Hough Transform is a robust technique that enables the identification of lines in an image by mapping edge points from the image space to a parameter space, where each point corresponds to a potential line in the original image. By investigating the impact of various parameter settings and post-processing techniques, we aim to optimize the line detection accuracy and minimize the presence of duplicate or overlapping lines. Through a series of experiments and visual evaluations, we demonstrate the effectiveness of non-maximum suppression and decreased rho resolution in enhancing the Hough Transform's output, ultimately leading to cleaner and more precise line detection results. The findings presented in this section provide valuable insights into the intricacies of the Hough Transform and offer guidance for achieving optimal line detection performance in various computer vision applications.

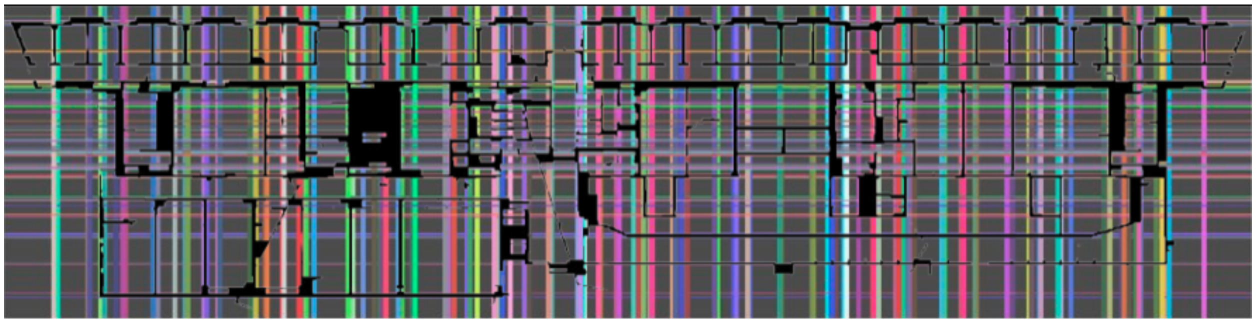

Figure 28: Hough Transform applied to UCSC Engineering 2 Floor 2 floor plan image without non-maximum suppression

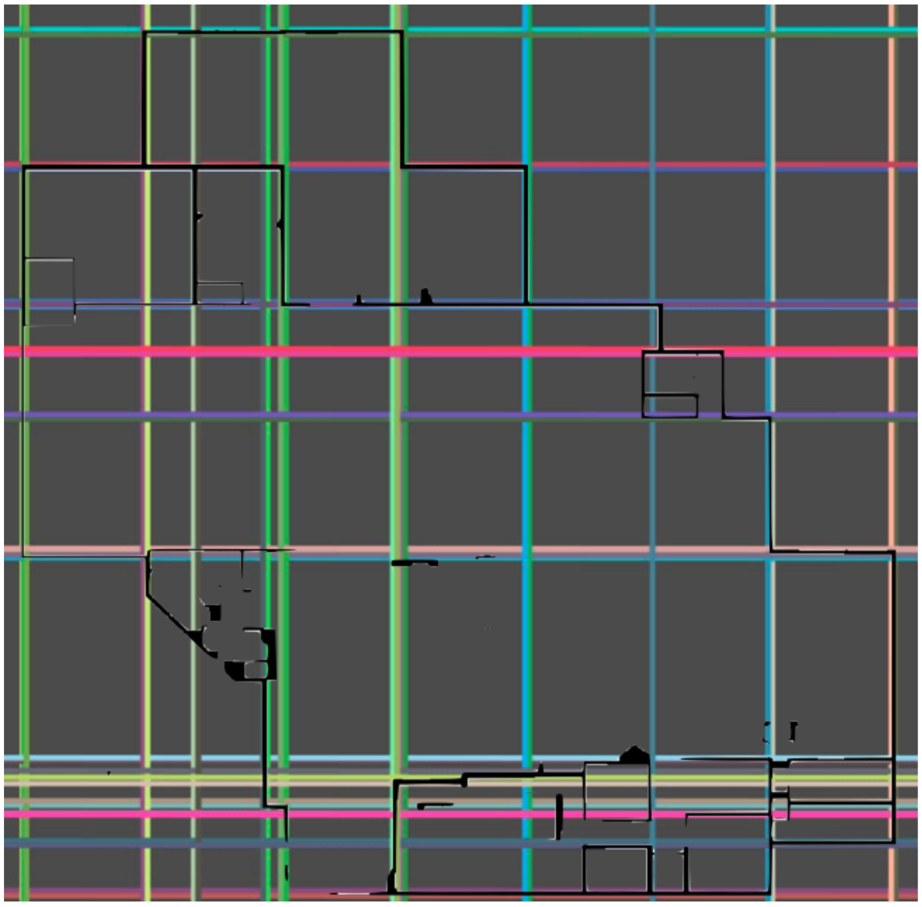

Figure 29: Hough Transform applied to UCSC Science and Engineering Library floor plan image without non-maximum suppression

Figure 28 and 29 illustrate the result of applying the default Hough Transform without any non-maximum suppression. As can be observed, the default implementation detects a significant number of overlapping and duplicate lines. This is because the transform identifies all possible lines that pass through the edge points in the image, leading to redundant detections. The presence of these duplicate lines can clutter the output and make it challenging to interpret the results accurately.

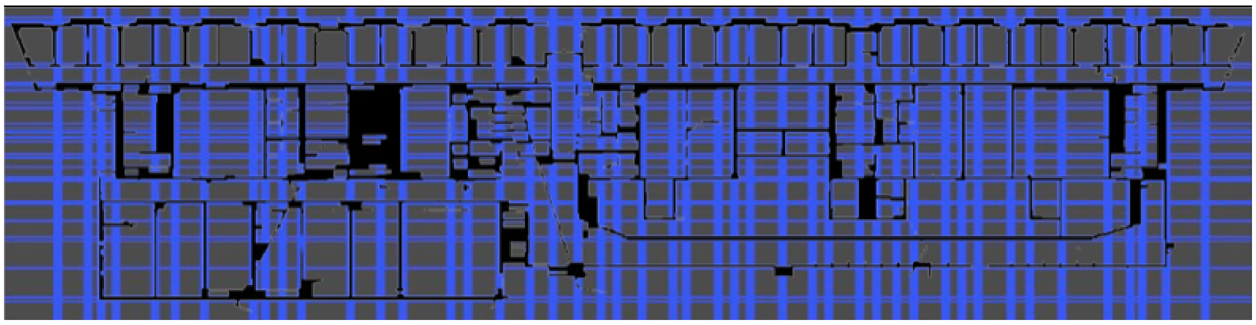

Figure 30: Hough Transform applied to UCSC Engineering 2 Floor 2 floor plan image with non-maximum suppression and rho value of 1.

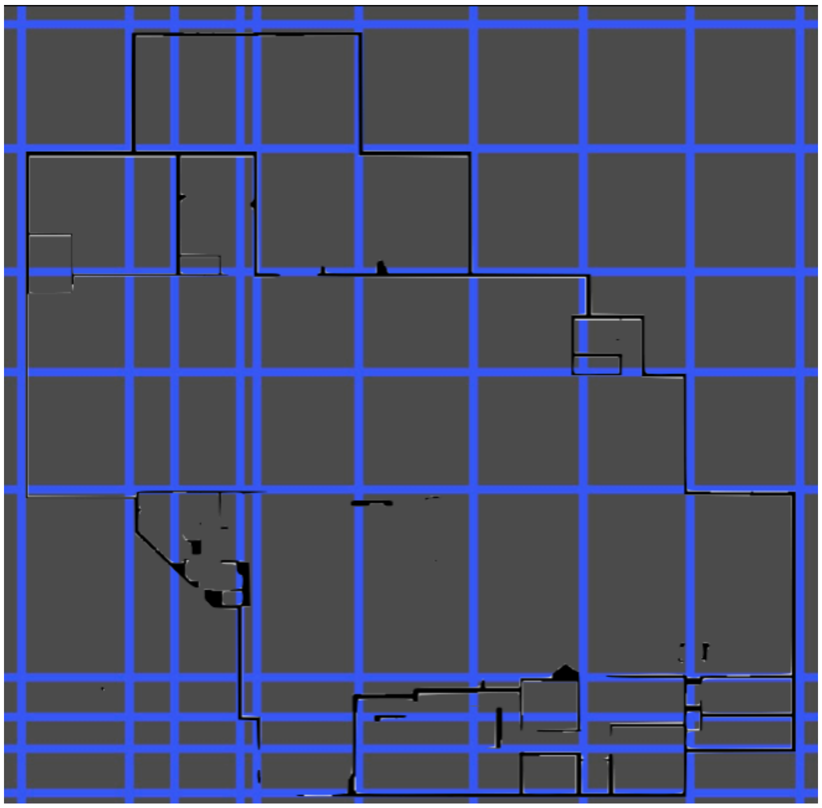

Figure 31: Hough Transform applied to UCSC Science and Engineering Library floor plan image with non-maximum suppression and rho value of 1.

To address this issue, we introduce non-maximum suppression, as demonstrated in Figures 30 and 31. Non-maximum suppression is a technique that helps reduce the number of duplicate and overlapping lines by suppressing lines that are close together in both the rho (distance from the origin) and theta (angle) dimensions. The implemented algorithm iterates through the detected lines, compares their rho and theta values, and only retains the strongest lines that are not within a specified tolerance range of each other. By applying non-maximum suppression, we obtain a cleaner output with fewer redundant lines, enhancing the clarity and interpretability of the detected lines.

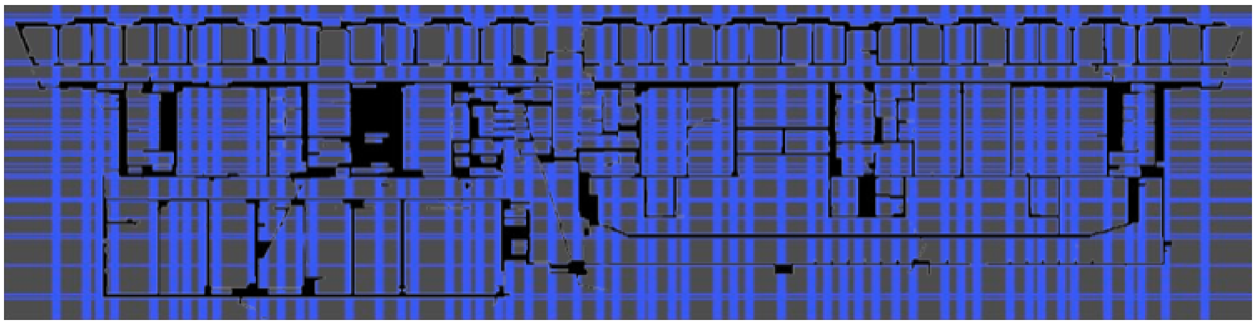

Figure 32: Hough Transform applied to UCSC Engineering 2 Floor 2 floor plan image with non-maximum suppression and rho value of 0.1.

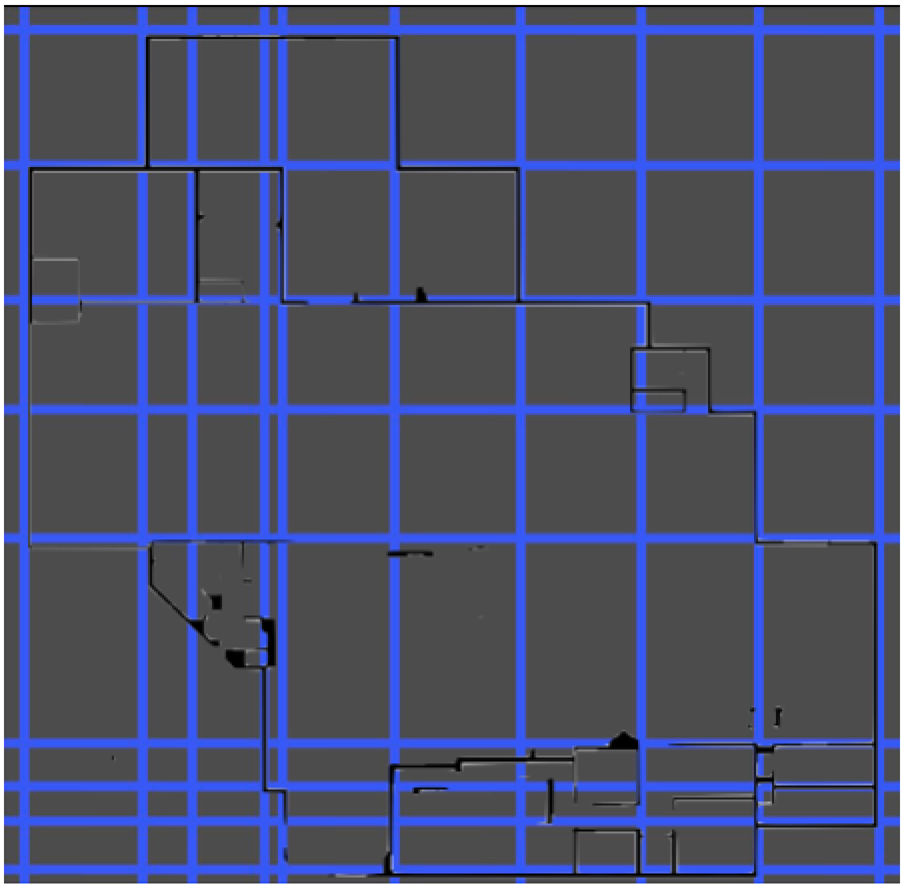

Figure 33: Hough Transform applied to UCSC Science and Engineering Library floor plan image with non-maximum suppression and rho value of 0.1.

Figures 32 and 33 showcase the Hough Transform output with improved non-maximum suppression and a decreased rho resolution. By decreasing the rho resolution, we effectively increase the number of bins in the accumulator's rho dimension. This finer granularity allows the transform to detect subtle variations in the distance of lines from the origin. Consequently, the Hough Transform can distinguish between lines that are nearly parallel or in close proximity to each other, which might otherwise be grouped together under a coarser rho setting. The resultant image exhibits a significant improvement in line detection accuracy, with almost all the prominent lines in the image being correctly identified as distinct entities.

It is important to note that despite the enhancements achieved through improved non-maximum suppression and decreased rho resolution, some lines may still be absent in the final output. This can be attributed to various factors, including the selected threshold values, the presence of noise or discontinuities in the image, or inherent limitations of the Hough Transform algorithm. To further optimize the line detection results, additional techniques such as fine-tuning the parameters, applying advanced pre-processing or post-processing methods, or exploring alternative line detection algorithms can be considered.

In conclusion, the experimental results presented in this section demonstrate the significant impact of non-maximum suppression and rho resolution on the performance of the Hough Transform for line detection. By implementing these techniques, we can effectively reduce the number of duplicate and overlapping lines, improve the separation of closely spaced lines, and enhance the overall accuracy of line detection in images. The findings highlight the importance of carefully tuning the parameters and applying appropriate post-processing methods to obtain optimal results when utilizing the Hough Transform for line detection tasks.

V. Discussion

A. Interpretation of Results

The results of this study underscore significant advancements in the automatic vectorization of architectural floor plans. The integration of computer vision techniques, particularly convolutional neural networks (CNNs) and Hough Transform, has proven effective in enhancing the accuracy and clarity of vectorized horizontal and vertical line outputs from raster images. The DeepFloorPlan algorithm, with its room-boundary-guided attention mechanism, demonstrated robust performance in identifying and delineating architectural elements such as walls and doors, even in complex floor plans.

The implementation of tiling algorithms was crucial in maintaining high-resolution detail across large floor plan images, ensuring that the CNN could process each segment effectively without loss of detail. Post-processing techniques, such as the enhanced non-maximum suppression in the Hough Transform, significantly reduced the occurrence of duplicate and overlapping lines, thereby improving the overall precision of the line detection process.

Moreover, the extraction of PostScript commands from some architectural PDFs provides a novel approach to assess the accuracy of this vectorization methodology. PostScript, a page description language used in the electronic and desktop publishing areas, enables precise rendering of document layouts and images, which includes architectural drawings. By parsing PostScript commands embedded in PDFs, we can directly compare the original commands with the vectorized outputs, thus providing a measurable benchmark for the precision of our vectorization techniques.

The study's findings highlight the efficacy of combining advanced image processing techniques with deep learning models to achieve superior results in the vectorization of architectural floor plans.

B. Challenges Encountered and Overcome

Throughout the research, several challenges were encountered and effectively addressed. One significant challenge was dealing with the noise and textual elements present in raster floor plan images, which often interfered with the vectorization process. The application of Optical Character Recognition (OCR) for text removal, followed by inpainting techniques to fill in the gaps, successfully mitigated this issue, resulting in cleaner images for processing. Another challenge was the computational limitations associated with processing large images using CNNs. This was overcome by implementing a tiling algorithm that segmented the images into smaller, manageable tiles, allowing for efficient processing without loss of detail. Additionally, the problem of duplicate and overlapping lines detected by the Hough Transform was resolved through the development of an improved non-maximum suppression algorithm, which refined the line detection results and ensured the clarity and precision of the vectorized outputs. These solutions collectively enhanced the robustness and reliability of the proposed vectorization methodology, demonstrating its potential for practical application in various fields requiring precise digital representations of architectural floor plans.

VI. Implications

A. Application to Semantic Interior Mapology (SIM)

The potential future integration of advanced automatic vectorization techniques into the Semantic Interior Mapology (SIM) toolbox would significantly enhance its functionality and efficiency. SIM is designed to convert architectural floor plans into interactive 3D visualizations [17], and the incorporation of automatic vectorization streamlines this process. By leveraging the precise and automated vectorization methods developed in this study, the SIM toolbox could quickly and accurately trace floor plans, converting raster images into detailed vectorized maps. This process is facilitated by the Map Conversion toolkit within SIM, which now benefits from the ability to automatically detect and delineate architectural features such as walls and doors with high accuracy. Moreover, the system enables users to select corners and identify rooms, further simplifying the process of transforming floor plans into detailed 3D models.

The automated vectorization process involves using convolutional neural networks (CNNs) and Hough Transform techniques to identify and trace the structural elements of floor plans. This integration would allow SIM to handle a variety of architectural styles and complexities with minimal manual input, significantly reducing the time and effort required to prepare floor plans for 3D visualization. The vectorized outputs are then seamlessly transformed into the SIM file format, which can be converted into GeoJSON for use with mapping platforms like OpenStreetMap and Mapbox.

B. Practical Applications in Various Industries

The advancements in vectorization techniques have broad implications across multiple industries. In urban planning and real estate, the ability to quickly and accurately convert floor plans into detailed digital maps enhances the efficiency of project planning and property management. These vectorized maps can be seamlessly integrated into geographic information systems (GIS), enabling detailed spatial analysis and decision-making. In the field of architecture and construction, precise digital representations of floor plans facilitate better design, remodeling, and construction management processes.

C. Enhancements in Navigational Aids and Spatial Orientation

The improved vectorization techniques significantly enhance the development of navigational aids, particularly for individuals with visual impairments. Accurate and detailed digital maps of building interiors are crucial for creating effective navigational tools that provide clear and reliable guidance. By integrating these high-fidelity vector maps with real-time location data, developers can create applications that offer precise directions and spatial orientation cues, thereby improving the autonomy and mobility of visually impaired users. These advancements also support the development of advanced navigation systems in large public spaces such as airports, shopping malls, and educational institutions, enhancing the overall user experience by providing intuitive and accessible navigation solutions.

D. Contributions to the Field of Computer Vision

The study's contributions extend beyond practical applications, significantly advancing the field of computer vision. The integration of CNN-based feature recognition with advanced image processing techniques sets a new benchmark for the accuracy and efficiency of floor plan vectorization. This research demonstrates the potential of deep learning models to handle complex and diverse architectural layouts, addressing limitations of traditional vectorization methods. The enhanced algorithms for noise removal, line detection, and feature extraction contribute to the development of more robust and scalable solutions for automated map generation. These innovations open new avenues for research in computer vision, particularly in the areas of architectural image analysis and digital map creation, paving the way for further advancements and new applications in spatial data analysis and visualization.

VII. Conclusion

A. Summary of Findings

This research has demonstrated significant advancements in the automatic vectorization of architectural floor plans, leveraging state-of-the-art computer vision techniques and deep learning models. The integration of convolutional neural networks (CNNs) with advanced image processing methods, such as the Hough Transform and enhanced non-maximum suppression, has markedly improved the accuracy and clarity of vectorized outputs from raster images. The study highlights the potential for these techniques to enhance tools like the Semantic Interior Mapology (SIM) toolbox, which aims to convert complex floor plans into detailed, interactive 3D visualizations. Key achievements include the efficient handling of various architectural styles and the significant reduction in manual input required for floor plan conversion.

B. Conclusion Drawn from the Research

The research concludes that the developed automatic vectorization techniques provide a robust and scalable solution for converting architectural floor plans into high-fidelity digital maps. These techniques address several longstanding challenges in the field, such as noise interference and the imprecision of traditional vectorization methods. By enhancing the accuracy of feature detection and line tracing, the proposed approach not only improves the quality of digital floor plans but also expands their applicability across various industries. The findings suggest that integrating these methods into tools like the SIM toolbox could significantly enhance their utility and potential for widespread adoption in fields requiring precise spatial data representation and analysis.

C. Recommendations for Future Research

Future research should focus on further refining the vectorization algorithms to handle even more complex and varied architectural designs. Enhancements could include developing techniques to better manage curved and irregular shapes, as well as improving the robustness of the algorithms against different types of noise and distortions commonly found in scanned images. A key area for future exploration is the improvement of room detection and recognition capabilities. This involves enhancing the algorithms to more accurately identify and label distinct rooms and spaces within floor plans, which is crucial for applications in building management, interior design, and navigation systems. Advancing the precision of room recognition would facilitate more detailed and useful spatial data, enabling better integration with systems like SIM that rely on accurate indoor mapping.

Additionally, exploring the integration of these vectorization methods with real-time data processing and augmented reality applications could significantly expand their utility. Research should also consider optimizing the computational efficiency of the algorithms to facilitate their deployment on mobile devices and other resource-constrained platforms. Finally, further studies could investigate the application of these techniques to 3D models and elevation drawings, broadening their scope and enhancing their contribution to the fields of architecture, urban planning, and beyond.

References

[1] Z. Zeng, X. Li, Y. K. Yu, and C.-W. Fu, "Deep Floor Plan Recognition using a Multi-task Network with Room-boundary-Guided Attention," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[2] C. Liu, J. Wu, P. Kohli and Y. Furukawa, "Raster-to-Vector: Revisiting Floorplan Transformation," *2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017*, pp. 2214-2222, doi: 10.1109/ICCV.2017.241.

[3] P. de, "Vectorization of Architectural Floor Plans," *2019 Twelfth International Conference on Contemporary Computing (IC3), Noida, India*, 2019, pp. 1-5, doi: 10.1109/IC3.2019.8844930.

[4] W. Song, M. M. Abyaneh, M. A. A. Shabani, and Y. Furukawa, "Vectorizing Building Blueprints," in *Proceedings of the Asian Conference on Computer Vision (ACCV)*, Dec. 2022, pp. 1044-1059.

[5] Bingchen Yang, Haiyong Jiang, Hao Pan, and Jun Xiao. “Vectorfloorseg: Two-stream graph attention network for vectorized roughcast floorplan segmentation.” In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1358–1367, 2023.

[6] Kim, S.; Park, S.; Kim, H.; Yu, K. “Deep floor plan analysis for complicated drawings based on style transfer.” *J. Comput. Civ. Eng. 2021*, 35, 04020066.

[7] O’Shea, K.; Nash, R. An introduction to convolutional neural networks. arXiv 2015, arXiv.1511.08458 DOI: 10.48550/arXiv.1511.08458.

[8] Li, Z.; Liu, F.; Yang, W.; Peng, S.; Zhou, J. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE Trans. Neural Netw. Learn. Syst.* 2021, 33, pp. 6999–7019.

[9] S. Albelwi and A. Mahmood, "A framework for designing the architectures of deep convolutional neural networks," *Entropy*, vol. 19, no. 6, p. 242, 2017.

[10] A. Singh, K. Bacchuwar, and A. Bhasin, "A survey of OCR applications," *International Journal of Machine Learning and Computing*, vol. 2, no. 3, p. 314, 2012.

[11] P. Reddy, M. Gharbi, M. Lukac, and N. J. Mitra, "Im2vec: Synthesizing vector graphics without vector supervision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7342-7351, 2021.

[12] A. Carlier, M. Danelljan, A. Alahi, and R. Timofte, "DeepSVG: A hierarchical generative network for vector graphics animation," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16351-16361, 2020.

[13] A. Kalervo, J. Ylioinas, M. Häikiö, A. Karhu, and J. Kannala, "Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis," in *Proceedings of the 21st Scandinavian Conference on Image Analysis (SCIA 2019)*, Norrköping, Sweden, June 11-13, 2019, pp. 28-40. Springer, 2019.

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1125-1134, 2017.

[15] C. Harris, M. Stephens, et al., "A combined corner and edge detector," in *Alvey Vision Conference*, vol. 15, no. 50, pp. 10-5244, 1988.

[16] J. Shi et al., "Good features to track," in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593-600, 1994.

[17] V. Trinh and R. Manduchi, "Semantic interior mapology: A toolbox for indoor scene description from architectural floor plans," in 24th International Conference on 3D Web Technology, vol. 2019, pp. 1-1, 2019.

[18] “OpenCV: Harris Corner Detection,” Opencv.org, 2024. https://docs.opencv.org/3.4/dc/d0d/tutorial_py_features_harris.html.

[19] “OpenCV: Shi-Tomasi Corner Detector & Good Features to Track,” Opencv.org, 2024. https://docs.opencv.org/4.x/d4/d8c/tutorial_py_shi_tomasi.html.

[20] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236-239, 1984.

[21] Z. Guo and R. W. Hall, "Parallel thinning with two-subiteration algorithms," *Communications of the ACM*, vol. 32, no. 3, pp. 359-373, 1989.

[22] "Raster vs Vector". Gomez Graphics Vector Conversions. Retrieved 23 May 2024. “Raster images are created with pixel-based programs or captured with a camera or scanner. They are more common in general such as jpg, gif, png, and are widely used on the web.”

[23] UC Santa Cruz, “Floor Plans | BSOE Facilities,” UCSC BSOE Facilities, 2024. https://facilities.soe.ucsc.edu/floor-plans (accessed May 23, 2024).

[24] UC Santa Cruz, “Library Guides: Science & Engineering Library: S&E Library Floor Plans,” UC Santa Cruz University Library, 2024. https://guides.library.ucsc.edu/science-and-engineering-library/floor-plans (accessed May 23, 2024).