

**Project Report: Automated Diagram Extraction and Bend Counting from BBS (Bar Bending Schedule) Documents**

## **Purpose of the Project**

The project aims to automate the extraction and analysis of diagrams from Bar Bending Schedule (BBS) documents, and then to automate the calculation of the number of edges in each diagram. The primary objective is to enhance efficiency, accuracy, and consistency in processing BBS documents by leveraging image processing and machine learning techniques.

## **Intended Audience**

This project is intended for:

- Engineers and analysts at Tata Steel involved in the construction sector.

- Software developers and data scientists working on automating document processing tasks.

- Stakeholders interested in reducing manual errors and increasing productivity in handling construction project documentation.

## **Market Overview**

The construction industry is a massive global market, valued at around $10.7 trillion (about $33,000 per person in the US) in 2020 (Source: Statista). It encompasses a wide range of activities, including residential, commercial, and infrastructure projects. Within this industry, the use of rebar (reinforcing steel bars) is crucial for providing structural integrity in concrete constructions. Bar Bending Schedules (BBS) documents are essential for detailing the rebar shapes and configurations required for each construction project.

## **Target Market**

The primary target market for the Image Analytics for Construction Project BBS solution is steel manufacturing companies like Tata Steel, that provide rebar and related construction materials. These companies often receive numerous BBS documents from clients, which need to be analyzed and processed.

## **Market Drivers**

* Increasing construction activities globally, driven by urbanization and infrastructure development.
* Growing demand for efficient and accurate rebar detailing and estimation processes.
* Rising focus on cost optimization and minimizing errors in construction projects.
* Adoption of digitalization and automation in the construction industry.

## **Market Opportunities**

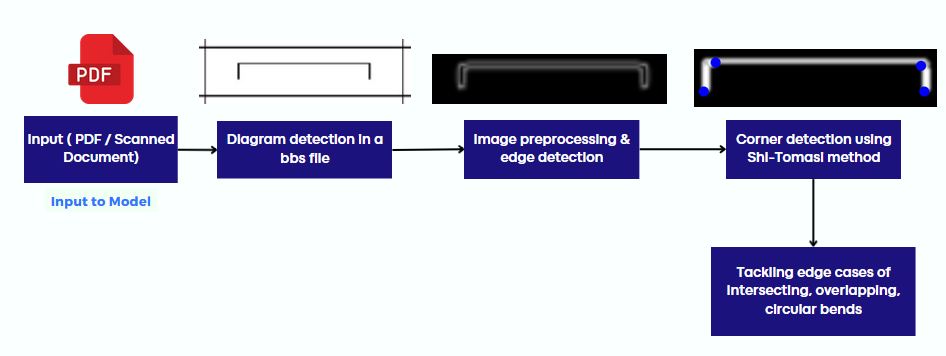
* Expanding the solution's capabilities to include additional construction document analysis, such as architectural drawings and structural plans.
* Offering the solution as a cloud-based service, enabling easier access and scalability for clients.
* Exploring partnership opportunities with engineering firms and construction companies to promote wider adoption of the solution.

## **Project Scope**

The project scope is defined as follows:

1. Input: BBS documents in PDF format provided by Tata Steel and its clients.
2. Output: Extracted diagrams, identified rebar shapes, and accurate bend counts for each shape.
3. Processes:
   * Conversion of BBS PDF documents to high-quality images
   * Diagram extraction from the converted images
   * Number removal from the diagrams using CNN (Convolutional Neural Networks) models
   * Corner detection and bend counting for various rebar shapes
   * Handling of overlapping, intersecting, and circular bends

## **Final Pipeline**



## **Problem Overview**

Tata Steel faces significant challenges while processing numerous BBS documents from clients. The key issues include:

- Error Prone: Manual counting of bends is susceptible to errors, leading to data inaccuracies.

- Time Consuming: BBS documents often have more than 100 pages, making the manual counting process extremely time-consuming.

- Lack of Consistency: Different formulae are used for complex rebar shapes, leading to non-uniform bend counting across multiple users.

## **Key Objectives**

- To develop an automated system for detecting and extracting diagrams from BBS documents.

- Bend Counting: To implement an algorithm that accurately counts bends in the extracted diagrams, handling complex scenarios like overlapping, intersecting, and circular bends.

- To minimize human errors and ensure consistent data extraction.

- To reduce the time required for processing BBS documents.

## **Approach**

The project adopts a multi-step approach involving various image processing and machine learning techniques:

- PDF to Image Conversion: Use PyMuPDF to convert BBS PDF pages into high-quality images, ensuring clarity by increasing DPI.

- Image Preprocessing: Apply techniques like blurring, inversion, and dilation to prepare images for edge detection.

- Edge Detection: Utilize the Canny edge detection algorithm to identify boundaries of diagrams and table lines.

- Diagram Extraction: Implement smart cropping based on detected edges to isolate relevant diagrams.

- Corner Detection: Use the Shi-Tomasi method for detecting corners, overcoming challenges related to overlapping, intersecting, and circular bends.

- Bend Counting: Apply techniques like Perimeter Pixel-Cluster Identification (PPCI) and Perimeter Pixel-Cluster Angle Identification (PPCAI) for accurate bend counting.

- Circular Bends: Use the change in angles across the corner to detect them.

## **Challenges**

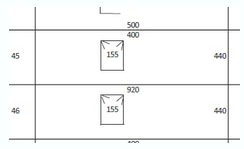
- Inconsistent Layouts: BBS files lack a standardized format for diagram placement, making it difficult to develop a one-size-fits-all solution.

- Mimicking random lines: Diagrams often visually resemble other lines within the document, making it hard to automatically distinguish them from tables.

- Interference from Non-Diagram Elements: Headers, footers and other non-diagram elements interfere with diagram detection.

- Number Removal: Presence of numbers within diagrams complicates the extraction process.

- Complex Bend Detection: Handling overlapping, intersecting, and circular bends requires sophisticated algorithms.

*As we can see in this snippet of a BBS file, the numbers are remarkably close to the diagram, some are inside the diagram itself, while some are close to the table columns & rows, making it difficult to remove these numbers from the images*

## **Potential Solutions**

- Standardizing Image Preprocessing: Develop robust preprocessing pipelines to handle varied document layouts.

- Enhanced Corner Detection: Employ advanced corner detection methods and refine algorithms for better accuracy.

- Machine Learning Models: Train models using datasets augmented to mimic real-world scenarios, enhancing their ability to distinguish between relevant and irrelevant elements.

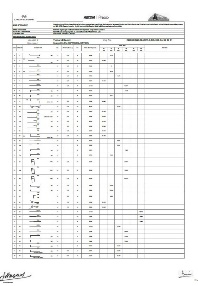
- Iterative Improvement: Continuously refine and test the system with diverse BBS documents to improve reliability and accuracy.

The solution aims to streamline the process of handling numerous BBS documents from clients, reducing the time and effort required for manual data extraction.

## **Detailed Approach**

### PDF to Image Conversion:

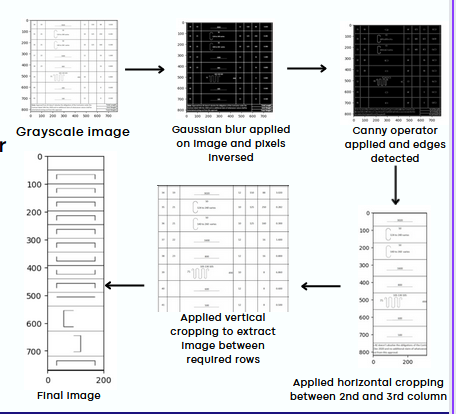
Extracting diagrams from Bar Bending Schedule (BBS) documents starts with converting PDF files into high-resolution images. This transformation is executed utilizing the PyMuPDF library, which facilitates the enhancement of image quality by increasing the DPI and minimizing JPEG compression artifacts. The resulting images are then transmuted into NumPy arrays, enabling flexible manipulation, and their color space is converted from RGB to BGR to augment compatibility with various image processing tools like the OpenCV library in Python.

*Input PDF of a BBS file Editable image of each page of the PDF*

### Diagram Extraction:

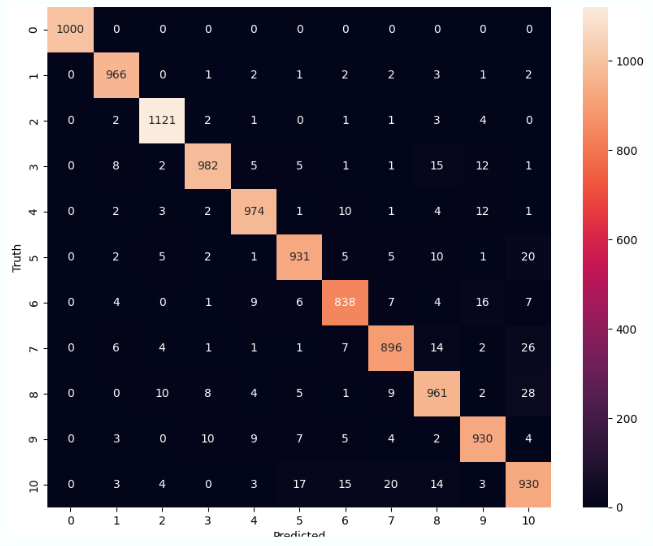
The diagram extraction procedure initiates by transmuting the PDF page image to grayscale, followed by the application of a Gaussian blur to attenuate background noise and amplify clarity. Subsequently, the Canny edge detection algorithm is employed to precisely delineate the boundaries of salient elements such as table lines and diagrams. This edge information is then leveraged to implement smart cropping techniques, which automatically isolate and extract the specific regions containing the desired diagrams.

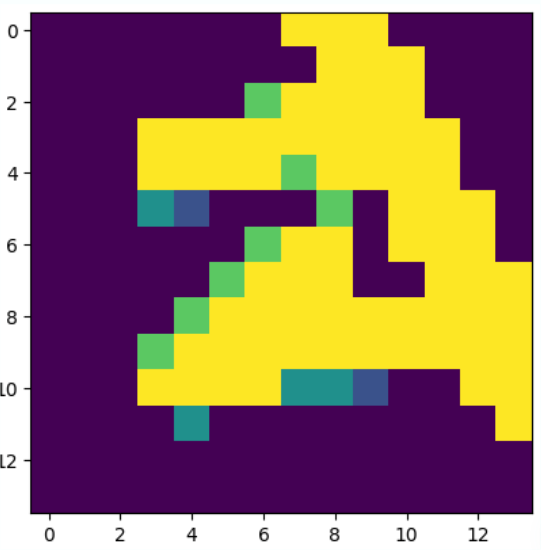
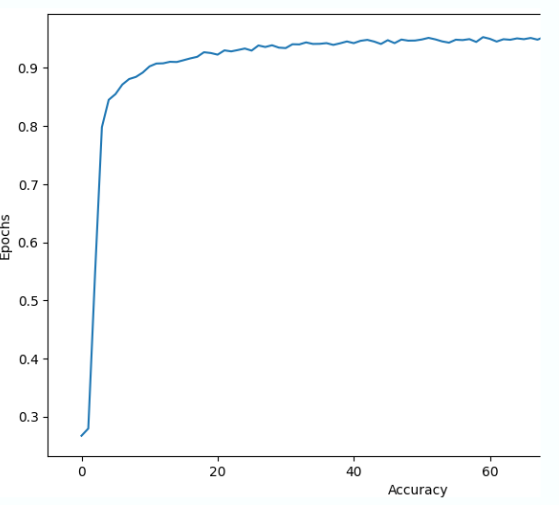
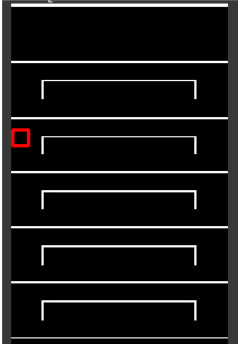


### Challenges in Automated Diagram Extraction:

The inconsistent layout of BBS files precludes the implementation of location-dependent algorithms, while the visual similarity between diagrams and other linear elements in the document complicates automatic distinction. To surmount these obstacles, several approaches were explored, including serial number-based identification, table detection coupled with column extraction, and the application of the Hough Transform with distance-based filtering. Unfortunately, these methods failed to yield consistent results due to variations in numbering schemes, layout inconsistencies, and the requirement for precise threshold values. The complexity of this task underscores the need for more sophisticated and robust algorithms to effectively automate the extraction and processing of diagrams from BBS documents.

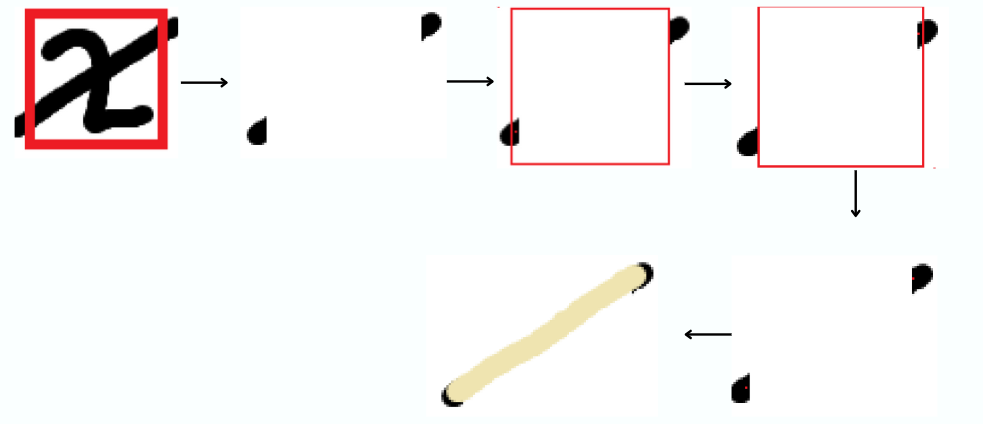
### Number removal:

The model’s training dataset was generated with the help of the MNIST dataset as a base, and using additional functions to augment, tilt, and add lines to the image; many blank images with/ without lines were added to act as a control. The purpose of the model was to recognize numbers and detect their presence. Using the model, we achieved ~96% accuracy on recognizing numbers, ~100% accuracy on detecting numbers.

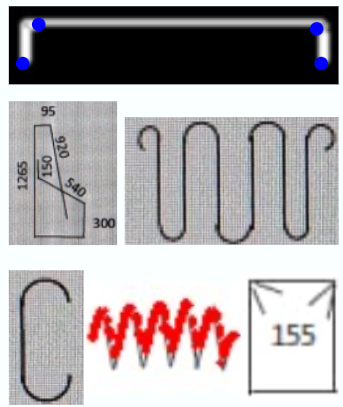
Finally, the code was implemented using sliding window CNN to detect number patch, and cv2.line() was used to fill the line up. After the patch (number + line) was removed. The centroid points of line just outside the patch were detected and line drawn.

Note: Number Removal was partially implemented, currently it is being carried currently by other groups.

*Image description of the process*

### Corner detection and bend counting

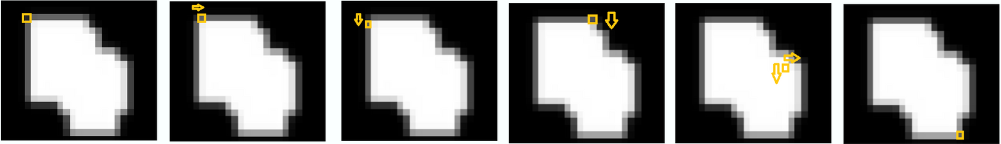
Corners are detected using the Shi-Tomasi method. There are certain edge cases remaining, for double bends in a patch, overlapping, intersecting, circular and spiral bends countered using various approaches.

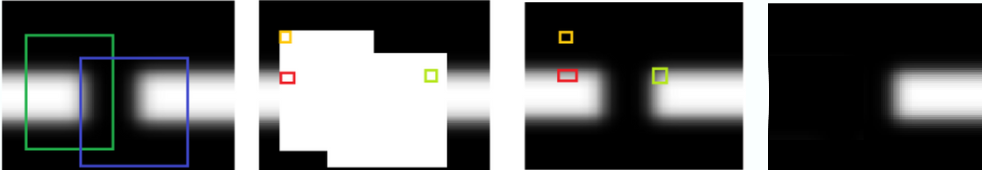
Corners were initially detected using Harrison corner detection method. The Corner Harrison detection method returns a patch based on the probability of a pixel being a corner. Challenges faced: Multiple different corners due to edges both sides. (Dilated image to tackle it). Have further thresholded it to make it binary (for easy calculations further on). Further challenge of nearby endpoints merging is faced.

### Initial challenges:

The nearby endpoint challenge is solved by taking the white patch (binary) and recursing in it (shown below). We first identify the points of a patch through a recursive algorithm and then recurse again on those points, but on the main image, if we detect multiple instances of a line in the patch then that confirms that corners were merged.



### Tackling Challenges:



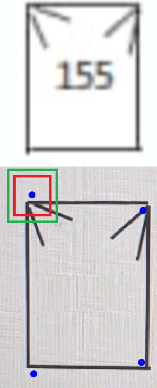
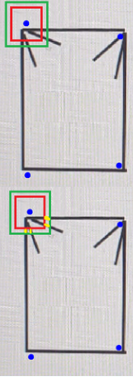
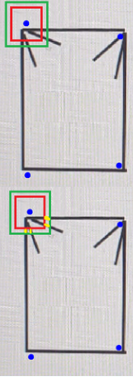
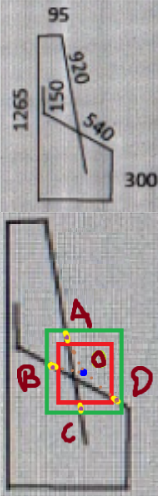
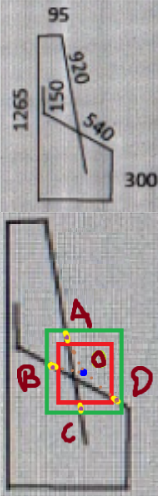
Multiple instances are detected by removing the connected line pixels (white pixels) as soon as one is detected, and recursing further. If more line pixels are detected, that means multiple line instances in a patch. The green and blue boxes merge to give the white patch (2nd), Yellow is when the first algorithm recurses on it. Red box is when the first line is detected (shown in 3), thus removing its connected pixels (4). The green box detects the second instance of the line.

Corners are detected using the Shi-Tomasi corner detection method. This is superior to Harrison corner detection method and returns the corner points right away. This further eliminates the need to check merger of patches.

**Further Challenges Faced**: Overlapping bend detection, Intersecting bend detection, Circular/Spiral bend detection.

### Overlapping Bends

Overlapping bends are those, which are two or more behind each other but appear to be the same in a 2d space (first image). To tackle this, we devised a technique we name as “perimeter pixel-cluster identification” (PPCI). Here we draw two boxes around the detected corner and detect any pixels of the line. On detecting a pixel, we remove the connected pixels from inside the box, thus getting each to cluster separately once.

Now we have several pixel clusters (third image). To identify overlapping bends, we divide the number of pixel clusters by two. One bend is formed by two lines and thus two clusters, so clusters/two gives us number of corners. Here we got 4 lines or 4 clusters (yellow) for the top left corner, thus implying two bends on top of each other.

### Intersecting Bends

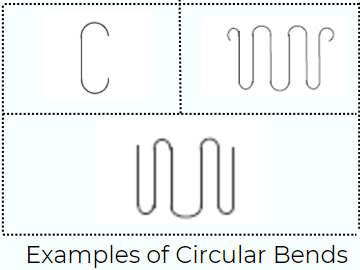
Intersecting bends (image 4): Intersecting bends are solved by a similar technique, called “Perimeter Pixel-Cluster Angle Identification” (PPCAI). Again, we draw two boxes around the detected corner and detect any pixels of the line. On detecting a pixel, we remove the connected pixels to it inside the box, but this time we also note down all the positions of the pixels in the cluster. Then we find the centroid of the clusters (fifth image).

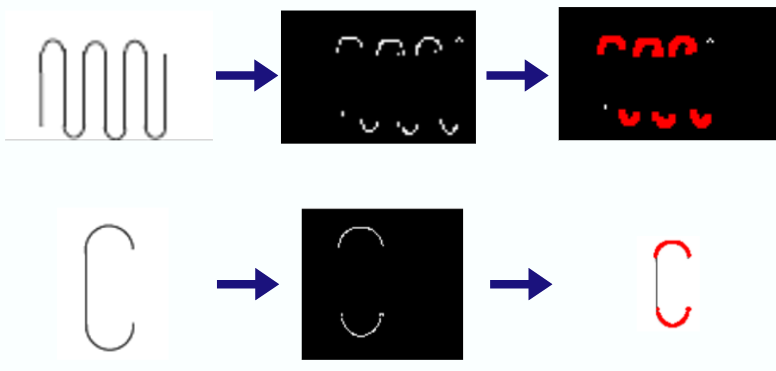
After finding the centroid we find the angles, between the corner point and centroid of cluster. For example in this case we find the clusters (yellow), their centroids (purple dots), named them (A,B,C,D) to show, and find angle between the blue dot and all 2 points, if any angle exceeds a certain threshold (180 deg ideally, around 150-160 for practical purposes) then we say that the two centroids lie on the same line and hence the corner detected is an intersecting point.

### Circular Bend Counting

Circular Bends are bends which aren’t sharp enough to be detected directly using the previous methods.

We have used the change in the angles across the corner to detect circular bends. We have applied angle normalisation using the Sobel Operator which gives us the following output -





## **Benefits**

- Increased Accuracy: Automated extraction reduces human errors, ensuring accurate data collection.

- Efficiency: Significantly reduces the time required to process BBS documents, improving overall productivity.

- Consistency: Provides uniform results across different users and documents.

- Scalability: The automated system can handle large volumes of BBS documents, making it scalable for extensive projects.

- Cost savings: By reducing the need for manual labor and minimizing errors, the solution can lead to significant cost savings for Tata Steel and its clients.

## **Unique Solution**

The Image Analytics for Construction Project BBS solution offers a unique approach to addressing the challenges faced by Tata Steel in handling Bar Bending Schedules (BBS) documents. Here are some key aspects that make this solution unique:

* **Specialized Focus on BBS Document Analysis**: While there are general document analysis solutions available, this project specifically targets the unique challenges posed by BBS documents, which contain complex diagrams, rebar shapes, and construction-specific information.
* **Advanced Image Processing Techniques**: The solution employs innovative image processing techniques, such as smart cropping, edge detection, and corner detection methods like Shi-Tomasi, to accurately extract diagrams and identify rebar shapes from BBS documents.
* **Handling of Complex Rebar Shapes**: Unlike traditional solutions that may struggle with overlapping, intersecting, and circular rebar shapes, this project incorporates specialized algorithms and techniques to accurately detect and count bends in these complex shapes. This includes methodologies like "Perimeter Pixel-Cluster Identification" and "Perimeter Pixel-Cluster Angle Identification."
* **Integration of Machine Learning**: The solution leverages ML models, such as Convolutional Neural Networks (CNNs), to tackle challenges like number removal from diagrams. This AI-powered approach enhances the solution's accuracy and adaptability.
* **End-to-End Automation**: From PDF conversion to diagram extraction, corner detection, and bend counting, the solution provides a fully automated workflow, minimizing the need for manual intervention and increasing efficiency.
* **Scalability and Flexibility**: The solution is designed to handle large volumes of BBS documents, making it suitable for projects of varying scales. Additionally, its modular architecture allows for future expansion and integration with other construction management systems.
* **Tailored for Construction Industry**: While many document analysis solutions are generic, this project is specifically tailored to meet the unique requirements of the construction industry, including adherence to industry standards, terminology, and best practices.

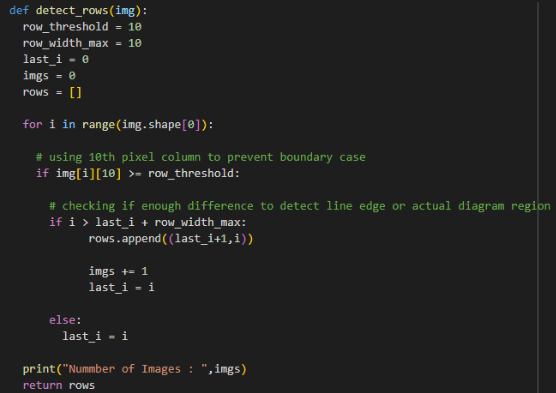
By combining advanced image processing techniques, machine learning models, and specialized algorithms for complex rebar shapes, the Image Analytics for Construction Project BBS solution offers a unique and comprehensive approach to streamlining the BBS document analysis process. This sets it apart from traditional solutions and positions it as an asset for Tata Steel, seeking improved efficiency, accuracy, and cost savings.

## **Conclusion**

The automated diagram extraction and analysis project for BBS documents is a significant step towards modernizing construction project management at Tata Steel. By leveraging advanced image processing and machine learning techniques, the project addresses critical challenges, enhances accuracy, and improves efficiency. The solution's scalability and adaptability make it an asset for Tata Steel and its clients in the construction industry. Continuous improvements and refinements will further solidify its effectiveness, making it an invaluable tool for the construction industry.

## **Code Snippets**

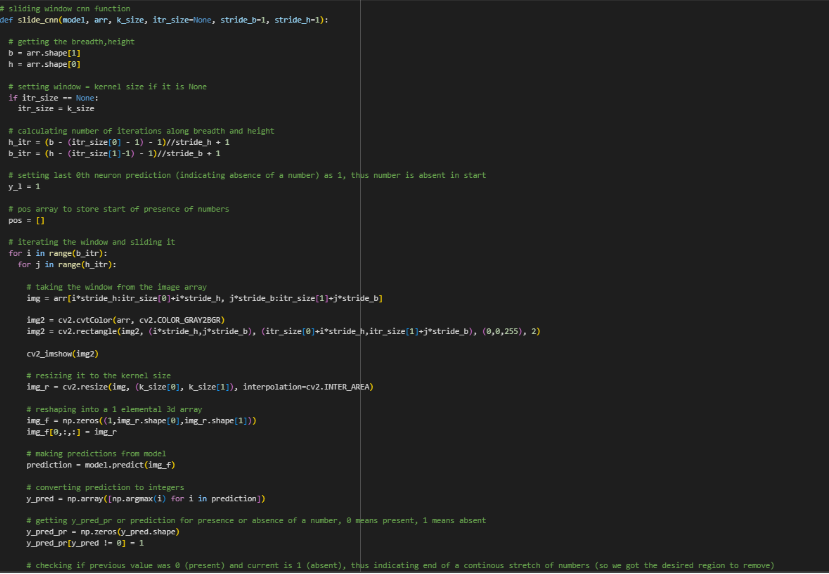
* The following code crops out the column containing all the diagrams, from a particular page of the BBS documents, achieved by first detecting the rows in an image of the BBS table

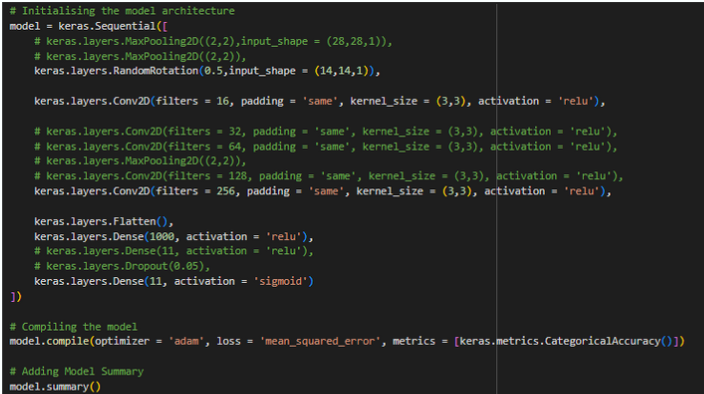


* This code converts the BBS document into individual pages, saves them in .png format, and in this image processing pipeline, the quality of the image is enhanced using various parameters and functions of PyMuPDF library

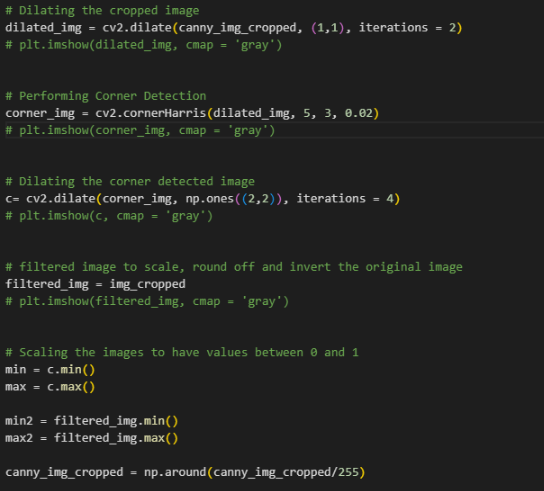


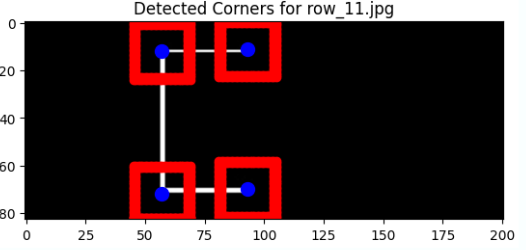
* This is the part of the code where number removal was implemented, this shows the sliding window CNN function.
* The following code shows the CNN model initialization part, for number removal.

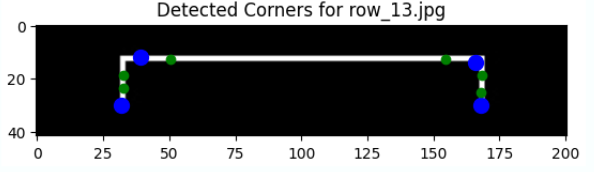




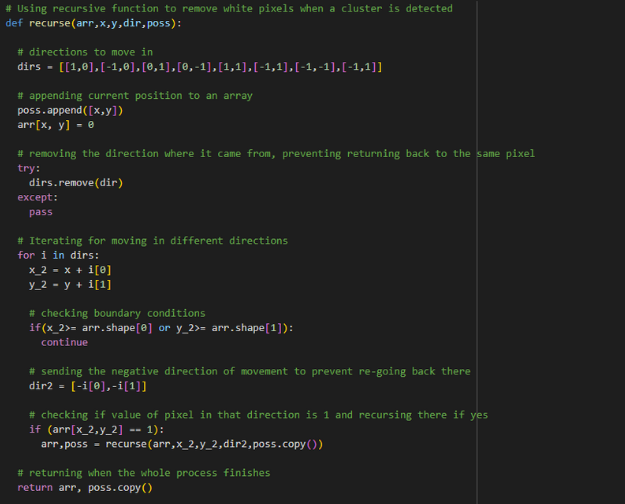
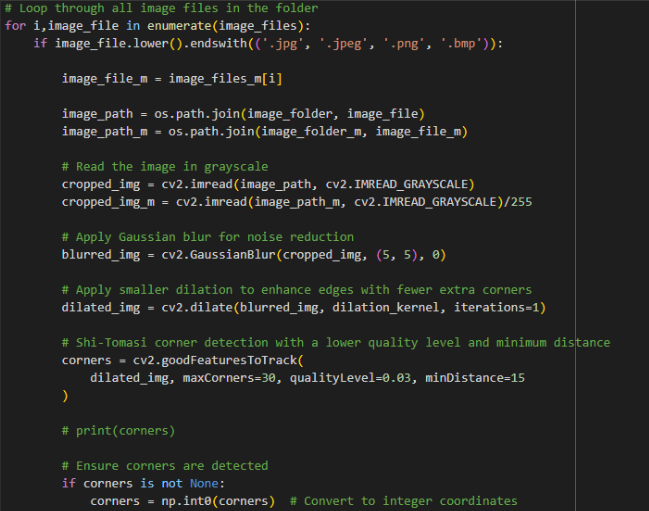
* This is the part of the code for image preprocessing, dilation, scaling and applying Harrison corner detection.
* The following images are part of the model outputs for corner detecting and perimeter pixel cluster identification (PPCI) respectively.



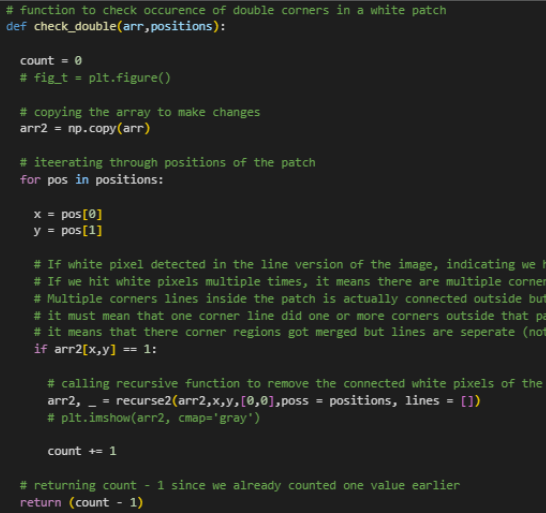




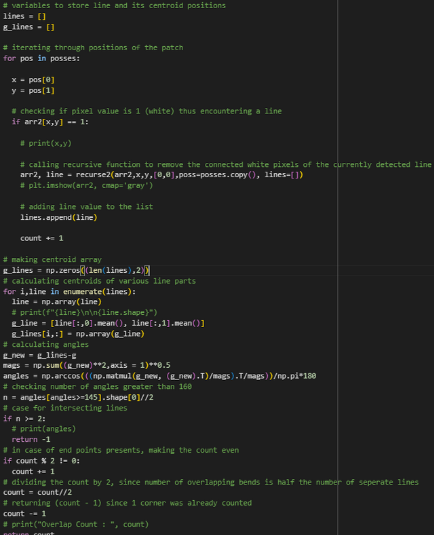
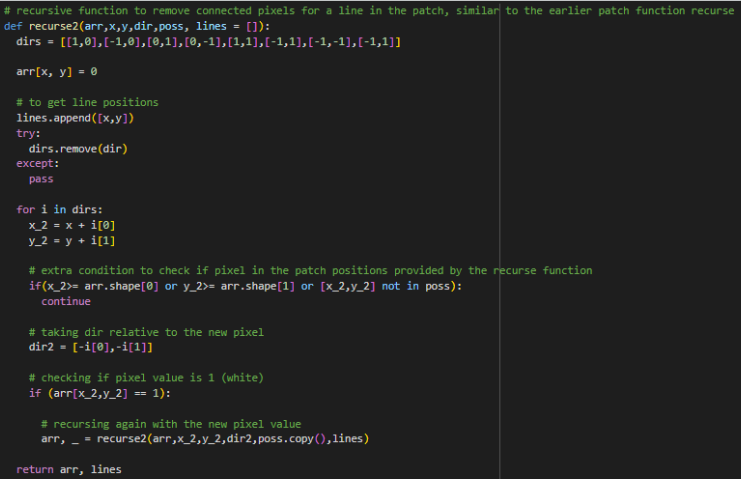
* This is the part of the code for image loading, blurring, dilation and applying the final method utilized, Shi-Tomasi corner detection.
* The following code is for the recurse function, used to iterate over the image to identify and remove pixel clusters, for patch counting.



* Function to check occurrence of double corners (non-overlapping) in a white patch.



* Part of the code to get line centroids calling recurse2 function (to get line pixel clusters), calculating cluster centroids, finding angles, returning if intersecting (at least one pair ~180 deg), and to get overlapping corners, after dividing the number of clusters by 2. (one is subtracted since it was already counted for)
* Following is the recurse2 function, which creates two boxes around corners, gets pixels between them, and recurses in the region like recurse one, but only in that region and not the whole image. Further it iterates on the original image, from the centroid boxes, in between the box area, centroid is calculated from the positions obtained from the patch positions, after iterating recurse function on it (recurse function will return the positions after iterating).



* Next is the Sobel Operator usage with normalisation of angle detecting non-sharp gradual terms as bends.
* Following that is the Image Processing used to calculate the number of bends detected and draw them on the image

