International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Machine Learning Approach for Converting Image Data/ Matrices (CSV) to 2D CAD Drawing.

Y. Yaswanth, K. Sai Surya, U. Nandini, P. Sathish Kumar, Dr. M. Vykunta Rao.

GMR Institute of Technology

Abstract:

The process of transforming the image-based engineering drawings into a well-structured format of Computer-Aided Design (CAD) drawings is considered to be a challenging process. It is considered that the traditional process of image-based engineering drawings recognition has been highly dependent on traditional drawing methodologies. Such a process has been considered a time-consuming and inefficient process of image-based engineering drawings recognition. In the proposed paper, a novel method of machine learning has been proposed for transforming the image-based engineering drawings and CSV matrix into accurate 2D CAD drawings by applying the Python and AutoLISP automation techniques. In the proposed method of machine learning for image-based engineering drawings recognition, the image-based engineering drawings preprocessing techniques, classification techniques, and CAD drawings automation techniques have been utilized for developing a novel and efficient method of transforming the image-based engineering drawings into a well-structured format of CSV matrix. In the image processing technique of machine learning for image-based engineering drawings recognition, edge detection, contour detection, and Hough transform are considered to be highly useful for image-based engineering drawings recognition. In the process of image-based engineering drawings recognition, different features are extracted from the image-based engineering drawings. Such features include lines, circles, arcs, and ellipses. The classification of features is considered to be highly useful for transforming the image-based engineering drawings into a well-structured format of CSV matrix. AutoLISP programming is considered to be highly useful for the automation of the process of generating accurate CAD drawings from the image-based engineering drawings. It has been considered that the proposed method of machine learning for image-based engineering drawings recognition is highly efficient and effective for generating accurate CAD drawings from the image-based engineering drawings without compromising the accuracy of the entities and annotation of drawing entities.

Keywords: CAD Automation, AutoLISP, Computational Geometry, 2D drawings, OpenCV, Feature Extraction, Engineering Drafting.

1.Introduction:

The increasing demand for automation in engineering design has led to significant advancements in Computer-Aided Design (CAD) systems and intelligent data processing techniques. Traditional CAD workflows rely heavily on manual drafting, which is time-consuming and prone to errors, especially when dealing with complex or large-scale engineering drawings. Over the years, researchers have explored various approaches to improve the efficiency and accuracy of design processes by integrating computational methods with CAD systems [1], [2], [14], [34].

Early research in CAD focused on feature-based and parametric modelling techniques, which enabled designers to define geometry using structured parameters and constraints [1], [28]. These approaches improved flexibility and allowed for easier modification of designs. Constraint-based drawing generation methods further enhanced automation by enabling the automatic creation of views and layouts based on predefined geometric rules [2], [15]. Such developments laid the foundation for modern CAD automation systems [35].

With the advancement of computer vision, image processing techniques have been widely used for extracting geometric information from images. Methods such as edge detection, contour extraction, and image binarization have been developed to identify structural elements within engineering drawings [20], [22], [26], [31]. Shape recognition techniques, including shape context descriptors and geometric feature analysis, have been used to classify and match

different shapes in images [3], [25]. These methods provide a basis for converting unstructured image data into structured representations suitable for further processing [23], [24]. Machine learning has played a crucial role in enhancing the accuracy and robustness of these systems. Algorithms such as Support Vector Machines and Random Forests have been applied to classify geometric entities based on extracted features [4], [5], [30]. These models can learn patterns from data and improving performance over time. More recently, deep learning approaches, particularly Convolutional Neural Networks (CNNs), have demonstrated significant improvements in image analysis tasks [28], [33]. CNN-based models can automatically extract features and handle variations in input data, making them suitable for complex image-to-geometry conversion problems [11], [19].

In addition to image processing and machine learning, CAD automation has been further enhanced through the use of scripting and programming interfaces. Tools such as AutoLISP and Python-based APIs allow for the automatic generation of CAD drawings from structured data [8], [15], [17]. These approaches enable seamless integration between data processing systems and CAD environments, reducing manual intervention and improving productivity. Automated drawing generation systems have been successfully applied in various engineering domains, demonstrating the potential of combining machine learning with CAD automation [27].

Recent studies have also explored advanced techniques such as deep learning-based vectorization and geometric reconstruction from images [18], [23], [24]. These approaches aim to improve the accuracy of converting raster images into vector representations, which is a critical step in CAD generation. Furthermore, research on graph-based neural networks and point cloud processing has expanded the capabilities of geometry analysis, although these methods are primarily focused on 3D data [9], [21].

Additional research efforts have focused on improving document analysis, pattern recognition, and automated drafting systems to enhance the reliability of CAD conversion processes [6], [7], [10]. These studies highlight the growing importance of integrating artificial intelligence with engineering design tools to achieve higher levels of automation and precision[12], [13], [16].

Despite these advancements, several challenges remain in developing a fully automated image-to-CAD conversion system. Existing methods often focus on individual components such as feature extraction or classification but lack a unified framework that integrates all stages of the process. Additionally, handling noisy images, complex geometries, and diverse drawing formats continues to be a significant challenge.

This work builds upon existing research by proposing an integrated system that combines image processing, machine learning, and CAD automation into a single pipeline. By leveraging structured CSV data and AutoLISP scripting, the proposed approach provides an efficient and scalable solution for converting image-based drawings into accurate CAD representations [29],[32]. The system aims to address the limitations of traditional methods and contribute to the advancement of intelligent engineering design systems.

2.Methodology

The proposed system for converting image data and CSV matrices into 2D CAD drawings is designed as a multi-stage pipeline that integrates image processing, machine learning, and CAD automation. The methodology ensures a systematic transformation of unstructured image inputs into structured geometric representations suitable for CAD environments. Each stage in the pipeline performs a specific function, contributing to the overall accuracy, efficiency, and automation of the system.

The process begins with input acquisition, where the system accepts raster images in standard formats such as PNG, JPEG, and BMP. These images may originate from scanned engineering drawings, photographed sketches, or digitally generated diagrams. For optimal performance, the input images are expected to have high resolution and contrast, ensuring clear visibility of geometric features. In addition to image input, the system also supports structured CSV data, which can directly represent geometric coordinates. The next stage involves image preprocessing, which prepares the raw input for feature extraction. The image is first converted into grayscale to reduce computational complexity by eliminating color information. Noise removal techniques such as Gaussian filtering or median filtering are then applied to enhance image clarity and remove unwanted distortions. Thresholding is used to convert the image into a binary format, separating the foreground geometry from the background. Finally, edge detection using the Canny algorithm is

performed to highlight the boundaries of geometric shapes. This preprocessing stage is crucial, as it directly impacts the accuracy of subsequent feature extraction.

Following preprocessing, a coordinate transformation is applied to align the image coordinate system with the CAD coordinate system. In images, the origin is typically located at the top-left corner with the Y-axis increasing downward, whereas CAD systems use a bottom-left origin with the Y-axis increasing upward. To resolve this difference, the Y-coordinate is transformed using the relation $Y_{cad} = H - Y_{pixel}$, where $H$is the image height. Additionally, a scaling factor is applied to convert pixel measurements into real-world units such as millimeters or inches based on the image resolution or user-defined parameters.

The system then proceeds to the feature extraction stage, where geometric entities are identified from the processed image. Contour detection using OpenCV's find Contours algorithm is employed to extract boundary information of shapes. Each contour represents a sequence of points that define a geometric structure. Various geometric properties such as area, perimeter, aspect ratio, and bounding box are computed for each contour. These features provide essential information for identifying different types of shapes.

To detect specific geometric primitives, specialized techniques are used. Straight lines are detected using the Hough Line Transform, which identifies linear patterns in the edge image. Circles are detected using the Hough Gradient method, which operates in a parameter space defined by the centre coordinates and radius. Ellipses are identified using the fitEllipse function, which fits a conic model to contour points. Arc detection is performed by analysing the angular coverage of circular contours, allowing the system to distinguish between full circles and partial arcs. These techniques ensure accurate detection of a wide range of geometric entities.

Once the features are extracted, the data is organized into a structured format in the CSV generation stage. The CSV file contains detailed information about each detected entity, including its type, coordinates, dimensions, and additional attributes such as layer and linetype. This structured representation acts as a bridge between the Python processing environment and the CAD system. It also enables easy storage, modification, and reuse of geometric data.

The next stage involves machine learning-based classification, where the detected geometric entities are classified into categories such as lines, circles, arcs, and polygons. A Random Forest algorithm is used due to its effectiveness in handling tabular data and its ability to provide high accuracy with low computational cost. The model is trained using features such as coordinates, lengths, angles, and curvature. During prediction, the model classifies each entity based on its extracted features, ensuring accurate identification of shapes.

In cases where rule-based detection is uncertain, an optional deep learning module is employed. A Convolutional Neural Network (CNN) is used to classify ambiguous shapes by analyzing small image patches. Transfer learning is applied using a pretrained model such as MobileNetV2, which is fine-tuned on the specific dataset of geometric shapes. This enhances the robustness of the system and improves performance in complex scenarios.

After classification, the system performs layer assignment, where each geometric entity is assigned to a specific CAD layer based on its type and context. This is implemented using a mapping mechanism that ensures consistent organization of the final drawing. Proper layering improves readability and allows for easier editing and management of CAD drawings.

The final stage involves CAD generation using AutoLISP. The CSV file is read by an AutoLISP program, which interprets each row and executes corresponding CAD commands. For example, lines are drawn using the LINE command, circles using the CIRCLE command, and arcs using the ARC command. Text annotations and dimensions are also generated automatically. The AutoLISP routine ensures that all entities are drawn with correct coordinates, layers, and properties.

Once the drawing is generated, a validation step is performed to verify the accuracy of the output. The generated CAD drawing is compared with the original image to ensure correct placement, orientation, and representation of geometric entities. Additional checks are performed for layer assignments, text placement, and dimension accuracy.

Overall, the proposed methodology provides a comprehensive and automated solution for converting image data into 2D CAD drawings. By integrating image processing, machine learning, and CAD automation, the system achieves high accuracy, efficiency, and scalability, making it suitable for real-world engineering applications.

3.Results:

The proposed system shows strong capability in transforming image-based data into well-structured and accurate 2D CAD drawings. The outcomes indicate that the system is able to identify and reconstruct key geometric elements such as lines, circles, and polygonal shapes with a high level of accuracy. The final CAD outputs closely resemble the original input images in terms of shape, arrangement, and overall design structure, which reflects the effectiveness of the feature extraction and reconstruction process.

The system demonstrates stable and consistent performance when tested on different types of input data, including clean images, moderately detailed drawings, and structured CSV datasets. The feature extraction module efficiently captures geometric primitives, while the machine learning component reliably classifies these shapes based on their properties. By combining rule-based detection techniques with machine learning, the system becomes more adaptable and capable of handling variations, distortions, and uncertain cases within the input data. From an efficiency perspective, the automated workflow significantly reduces the time and manual effort typically required in traditional drafting processes. The system is able to process input data rapidly and generate corresponding CAD drawings within a short span, making it highly suitable for applications involving large volumes of drawings. The use of AutoLISP scripting further ensures that drawing commands are executed smoothly and accurately within the CAD environment. In addition, the system maintains a high level of consistency with standard CAD practices. Elements are properly organized into layers, and attributes such as linetypes, colors, and text annotations are applied correctly. The placement and readability of text, along with the alignment of geometric components, contribute to the overall quality of the generated drawings. Although some limitations are observed when dealing with highly complex geometries or noisy images, the system continues to perform reliably in most practical scenarios.

Overall, the results demonstrate that the proposed approach is effective, efficient, and scalable, making it a valuable solution for automated CAD drawing generation in modern engineering applications.

4.Conclusion:

The proposed system demonstrates a high level of effectiveness in converting image data and CSV matrices into structured 2D CAD drawings through an automated and intelligent workflow. The integration of image processing, machine learning, and CAD automation enables the system to accurately detect and classify geometric entities such as lines, circles, arcs, and polygons. The results indicate that the generated CAD drawings closely match the input images in terms of geometry, orientation, and spatial arrangement, confirming the reliability of the overall approach.

The system achieves consistent performance across various types of input data, including clean images, moderately complex drawings, and structured datasets. The feature extraction module successfully identifies geometric primitives, while the machine learning model ensures dependable classification based on extracted features. The use of an automated pipeline significantly reduces manual effort and improves efficiency, making the system suitable for handling large volumes of design data.

In addition to accuracy and efficiency, the system maintains consistency in CAD standards by correctly assigning layers, linetypes, and text annotations. The seamless integration between the Python processing layer and the AutoLISP-based CAD generation ensures smooth execution and reliable output generation. This highlights the strength of the modular architecture and its suitability for real-world engineering applications.

Although certain limitations are observed in handling highly complex geometries and noisy inputs, the system performs effectively for most practical scenarios. Overall, the proposed approach provides a scalable, efficient, and reliable solution for automated CAD drawing generation, contributing to advancements in intelligent design automation and supporting modern engineering workflows.

References:

[1] J. J. Shah and M. Mäntylä, *Parametric and Feature-Based CAD/CAM*. New York: John Wiley & Sons, 1995.

[2] W. Li and P. Gu, "Constraint-based automatic view selection and drawing generation," *Computer-Aided Design*, vol. 37, no. 6, pp. 577–592, 2005.

[3] S. Belongie, J. Malik, and J. Puzicha, "Shape matching using shape contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[4] O. Chapelle, P. Haffner, and V. N. Vapnik, "Support vector machines for histogram-based image classification," *IEEE Trans. Neural Networks*, vol. 10, no. 5, pp. 1055–1064, 1999.

[5] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[6] M. Rezaei, M. Yazdi, and M. Rezaei, "Automatic recognition of GD&T symbols using convolutional neural networks," *International Journal of Advanced Manufacturing Technology*, vol. 112, pp. 2883–2897, 2021.

[7] Y. Jiang et al., "Machine learning algorithm for 3D aortic segmentation from 2D MR images," *Magnetic Resonance Imaging*, vol. 115, p. 110253, 2025.

[8] A. Kulkarni and S. Patil, "Python-based automatic generation of machine component drawings," *Journal of Manufacturing Systems*, vol. 56, pp. 314–326, 2020.

[9] Y. Wang et al., "Dynamic graph CNN for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[10] S. Tang et al., "2D–3D geometry constraints for indoor space segmentation," *International Journal of Applied Earth Observation and Geoinformation*, vol. 135, p. 104265, 2024.

[11] Y. Li and W. Xu, "Deep learning for intelligent architectural sketch to 3D model conversion," *Frontiers of Architectural Research*, 2025.

[12] D. Vizoso et al., "Machine learning approach for predicting 3D properties from 2D TEM images," *Measurement*, 2025.

[13] J. Bianchi et al., "AI-generated 3D facial models from 2D photographs," *Seminars in Orthodontics*, 2025.

[14] L. Deng et al., "Comparative study of 2D and 3D vegetation mapping using deep learning," *International Journal of Applied Earth Observation and Geoinformation*, vol. 125, p. 103588, 2023.

[15] J. Park and S. Yoo, "Automated engineering drawing generation using machine learning and CAD APIs," *Journal of Mechanical Science and Technology*, vol. 36, no. 4, pp. 1823–1832, 2022.

[16] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[17] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

18] Y. Lu, Y. Li, L. Zhang, and S. Wang, "Automatic vectorization of raster images using deep learning," *Computer-Aided Design*, vol. 123, p. 102830, 2020.

[19] A. Dosovitskiy, J. T. Springenberg, and T. Brox, "Learning to generate chairs with convolutional neural networks," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 692–705, 2015.

[20] S. Xie and Z. Tu, "Holistically-nested edge detection," *International Journal of Computer Vision*, vol. 125, no. 1–3, pp. 3–18, 2015.

[21] Z. Huang and S. You, "Point cloud labeling using 3D convolutional neural networks," *Pattern Recognition Letters*, vol. 78, pp. 57–64, 2016.

[22] J. Yang et al., "Object contour detection using deep learning," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2330–2343, 2016.

[23] Y. Liu, B. Fan, and S. Xiang, "Deep learning-based geometric feature extraction for CAD reconstruction," *Engineering Applications of Artificial Intelligence*, vol. 85, pp. 282–292, 2019.

[24] J. Zhang, Z. Chen, D. Tao, and J. Luo, "Automatic graphic recognition and reconstruction from engineering drawings," *Pattern Recognition*, vol. 74, pp. 294–307, 2018.

[25] B. Xu, D. Tao, and X. Zhang, "Multi-feature learning for image-based shape recognition," *Neurocomputing*, vol. 237, pp. 28–37, 2017.

[26] B. Su, S. Lu, and C. L. Tan, "Robust document image binarization technique for degraded images," *IEEE Trans. Image Processing*, vol. 22, no. 4, pp. 1408–1417, 2013.

[27] M. Chiang and Y. Chen, "Automated CAD model reconstruction from engineering drawings using machine learning," *Journal of Computational Design and Engineering*, vol. 6, no. 4, pp. 563–576, 2019.

28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[29] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[30] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[31] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2022.

[32] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2009.

[33] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly, 2022.

[34] I. Zeid, *Mastering CAD/CAM*. McGraw-Hill, 2007.

[35] K. H. Chang, *Product Design Modeling using CAD/CAE*. Academic Press, 2014.