Received Date:12th October, 2025
Revision Date: 5th November, 2025
Accepted Date: 15th January, 2026

CNC Plotter: An Educational Prototype for Intelligent Automation

Kripesh Paudel[1]*, Suresh Gurung[2], Urusha Luitel[3], Gaurav Gautam[4], Smriti Nakarmi[5]
[1]Dept of Electronics, Communication and Information Engineering, Kathmandu Engineering College, Nepal.
E-mail: kripeshpaudel32@gmail.com
[2]Dept of Electronics, Communication and Information Engineering, Kathmandu Engineering College, Nepal.
E-mail: gurungsuresh28@gmail.com
[3] Dept of Electronics, Communication and Information Engineering, Kathmandu Engineering College, Nepal.
E-mail: luitelurusha@gmail.com
[4] Assoc. Professor, Dept of Electronics, Communication and Information Engineering, Kathmandu Engineering College, Nepal.
E-mail: gaurav.gautam@kecktm.edu.np
[5] Assoc Professor, Dept of Electronics, Communication and Information Engineering, Kathmandu Engineering College, Nepal.
E-mail: smriti.nakarmi@kecktm.edu.np

Abstract— This paper illustrates the development of a CNC plotter as an educational prototype for intelligent automation. The system illustrates how computer vision, G-code execution, and AI-based decision making can all be integrated together into a CNC platform. The system has two modes: a drawing mode, where you can create text and pictures, and a gaming mode, where the plotter plays tic-tac-toe against a person. The hardware setup relies on an Arduino Uno, NEMA 17 stepper motors, and an SG90 servo motor, while the software setup includes Inkscape, Universal G-code Sender, and GRBL firmware. A custom-trained YOLOv11 model detects board states, while the Minimax algorithm with Alpha-Beta pruning calculates the most optimal moves. The experimental results show that user designs are accurately reproduced, gameplay performance is reliable, and symbols are classified nearly perfectly. The potential of integrating CNC technology and machine learning for the purpose of education and prototyping in intelligent automation is showcased in the project.

Keywords- CNC Plotter, Machine Learning, G-code, OpenCV, Inkscape, Object Detection, YOLOv11, Interactive Gaming, Automation

I. Introduction

CNC systems have changed manufacturing by automating slow and error-prone manual work. As industries push for faster and more precise production, CNC machines reduce time and improve output consistency. The rise of computer vision and machine learning has added new intelligence to these systems, making AI integration important for both industrial and educational use. Affordable CNC plotters are valuable for schools, small workshops, and hobby users. Building one with Arduino makes automation easier to access and understand. A low-cost system also helps learners' study real hardware instead of simulations. Adding AI improves accuracy and function while keeping the setup simple.

Despite their potential, existing CNC systems face several issues. Commercial models are costly and not easily available to students or small labs. Low-cost alternatives often perform only basic drawing or engraving tasks. They lack computer vision or decision-making abilities. Complex setup and calibration also make them difficult for beginners to use effectively.

This project addresses those limitations through a clear set of objectives. It develops a low-cost CNC plotter using affordable components. It provides dual functionality with both drawing and gaming modes. It integrates machine learning for symbol recognition through YOLOv11. It uses the Minimax algorithm for AI-based tic-tac-toe gameplay. The design serves as an educational prototype to help learners understand intelligent automation.

The system uses Arduino microcontrollers, stepper motors, and 3D-printed parts to create a working CNC machine. It performs accurate plotting and interactive tasks by combining computer vision and G-code execution. The approach demonstrates how AI can be embedded in CNC control, enabling practical and accessible automation for education and small-scale prototyping.

II. Literature Review

The evolution of CNC technology has led to major progress

* Corresponding Author

in the design and development of plotter machines. Hyder et al. discussed the creation of an Arduino-based CNC system capable of operating on three axes. Their work mainly dealt with the design process and practical setup of the machine, where stepper motors and G-code were used to control motion. They showed that by using open-source hardware and software, it's possible to build a low-cost CNC plotter that is simple to use and helpful for learning purposes [1].

Patil et al. emphasized cost efficiency in CNC plotter design. They integrated standard PC interfaces with Arduino-based controllers to reduce production expenses and improve usability. In their system, image files were first converted into G-code using Inkscape and then sent to the machine through Processing software or a Universal G-code Sender. The Arduino Uno interpreted the G-code and transmitted precise motion commands to the motor drivers. Their approach allowed accurate tool control without the high cost of traditional CNC machines, making automation more accessible for small educational and training institutions [2].

Kumar et al. developed a cost-effective CNC model that made use of simple algorithms and open-source software to convert image files into G-code for direct plotting. Their setup was capable of drawing circuit layouts and sketches on paper with minimal power usage. The system proved particularly useful in educational settings, where both affordability and efficiency are important factors [3].

Further work by Anil Kumar et al. focused on mini CNC sketchers that support easy modification through G-code manually or through Ink space. They explained the design processes and material selection, showing that CNC can generate precise building drawings using G-code. Such flexibility and simplicity make small CNC machines ideal for experimentation and training. They also stated that the application of mini CNC sketcher can be extended in field of Printed Circuit Board drawing and drilling [4].

Garg and Nayak explored the use of the Minimax algorithm for the tic-tac-toe game. Their research illustrated how decision-making logic can be applied to game-based systems to predict outcomes and respond optimally. They implemented the algorithm using concepts from graph theory and combinational game theory. It also demonstrated the growing connection between mechanical control and algorithmic intelligence [5].

Khanam et al. present an analysis and overview of YOLOv11, the latest version of the YOLO object detection model. With each new version, there are advancements that include better backbones and faster processing. To enhance feature extraction and spatial attention, YOLOv11 introduces new modules such as the C3k2, SPPF, and C2PSA blocks. Their analysis concludes that YOLOv11 provides an effective solution for addressing complex visual recognition challenges with improved accuracy and processing speed [6].

III. Methodology

A. CNC Drawing and G-Code Execution

The mechanical part of the system was designed to operate as a simple CNC plotter. The structure was made using lightweight 3D-printed parts and aluminum rods to maintain both stability and low cost. Stepper motors controlled the X and Y movements, while a small SG90 servo motor managed pen up and pen down actions. The Arduino Uno board, equipped with a CNC shield and GRBL firmware, served as the main controller for motion and tool control.

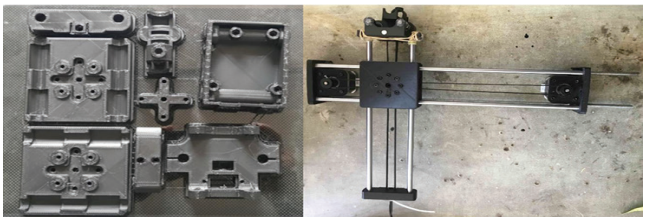


Figure 1: 3D-Printed CNC Parts and Assembled CNC

Design inputs such as text or images converted into G-code using Inkscape. Universal G-code Sender (UGS) transferred the G-code to the Arduino via a serial connection, which were interpreted by the GRBL firmware to control the motors. Prior to commencing any task, the motors underwent testing using brief G-code sequences to verify their direction and distance accuracy. Adjustments were made until both axes performed smoothly and returned back to their home positions.

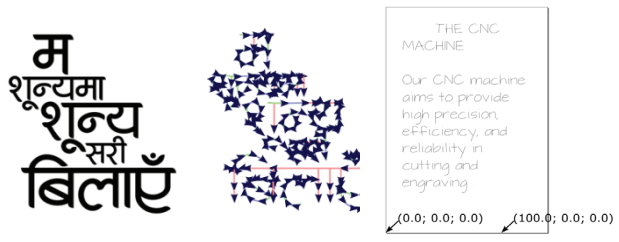


Figure 2: Input Design in Inkscape

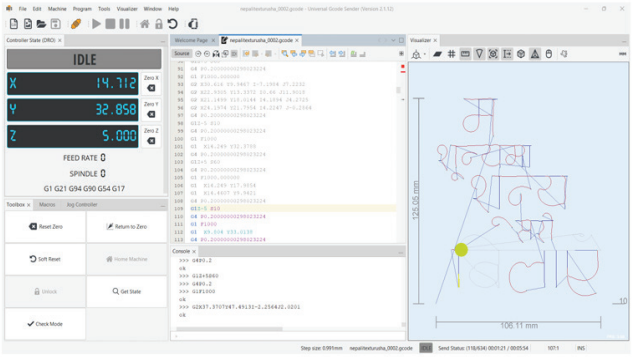


Figure 3: CNC Executing the G-Code

G-code testing included the reproduction of fundamental shapes and line patterns to verify repeatability. Following calibration, the plotter effectively reproduced both text and vector images with precision. This stage showcased the machine's capability to execute stable and accurate drawing tasks utilizing affordable components. After verification, the drawing module established itself as the foundation for further automation. This phase established the mechanical reliability necessary for subsequent stages, including AI-powered gameplay and object detection.

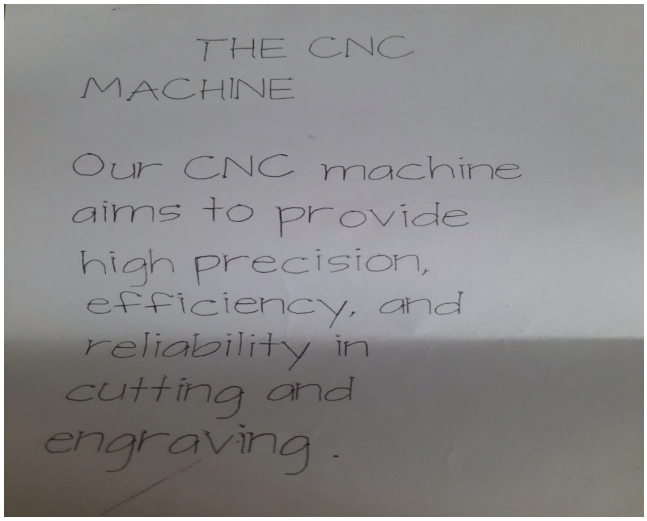


Figure 4: Completed CNC-Drawn Text Output

B. Data Collection

For this task, we collected 1,110 images for training the model to detect X and O symbols on the tic-tac-toe board. We varied lighting, camera angle, pen thickness, and drawing style. We included partial strokes and off-center marks to reflect real hand-drawn play. Each image were captured from slightly different angles to make the dataset more diverse.

Labeling took place in Roboflow. All images were reviewed to remove blurred or misaligned samples. Each image received one of the two labels: X or zero. We split the dataset into training, validation, and test sets as follows: 972 images for training, 92 for validation, and 46 for testing. The split preserved a balanced representation of boards and symbol positions.

We applied simple augmentations to increase variety. These included small rotations, brightness shifts, scaling, and minor translations. Augmentation helped the model handle real variations in symbol appearance and board alignment. These changes helped the model adapt to real conditions like uneven lighting or imperfect hand-drawn symbols.

The dataset built the base for training the YOLOv11 model. It provided clear samples that improved symbol recognition accuracy during tic-tac-toe gameplay.

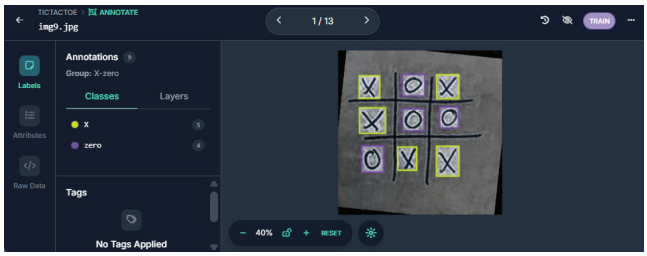


Figure 5: Roboflow Interface for Labelling Dataset

C. Object Detection using YOLOv11

Object detection is a computer vision method that aims to precisely identify and locate a specific object inside an image or video [5]. Numerous studies have been conducted on object detection in recent years, and writers have published their papers referring to it as a "state-of-the-art" model. Some of the well-known models are R-CNNN, Fast R-CNN, YOLO, etc. YOLOv11, the latest version of the “You Only Look Once” family, was adopted for detecting X and O symbols on the tic-tac-toe board. It performs object detection in a single pass, offering faster and more accurate results than region-based methods.

For this case, YOLOv11 was set up to detect two classes, X and O. Images were resized and normalized to match the model input. Training ran on a GPU workstation using PyTorch. The learning rate and batch size were adjusted until the validation loss stabilized. Model checkpoints saved intermediate results during training.

During inference, webcam frames feed the model. The detector returns boxes and confidence scores. Non-maximum suppression removes overlapping boxes. Each detection

maps to the closest tic-tac-toe cell, producing a 3 by 3 board state. The board state is passed to the decision module for move selection.

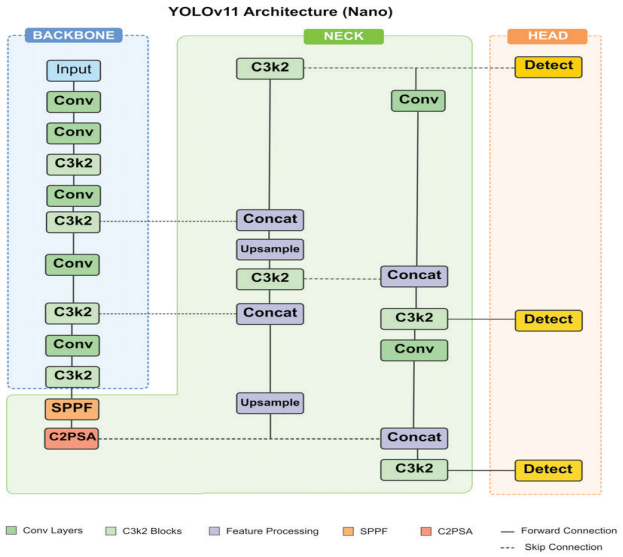


Figure 6:Overview of YOLOv11 [6]

D. Minimax Algorithm

To support intelligent gameplay, the system uses the Minimax algorithm as its decision-making logic. Minimax is a recursive search algorithm that determines the best move for the AI in two-player turn-based games like tic tac toe. Assuming both players behave optimally, it operates on the premise of decreasing potential loss while increasing potential gain.

During execution, the algorithm generates a game tree which includes all of the possible moves that can be made by the player and the AI. Each terminal state of this tree is given a score based on the game result. Based on the outcome of the game, each terminal state of this tree is assigned a score: +1 for an AI win, 0 for a draw, and −1 for a human win. Based on these values, the algorithm can guess the order of moves that will lead to the best outcome. The AI (maximizer) chooses the move with the highest score, while the opponent (minimizer) tries to lower the score.

Alpha-beta pruning improves the speed of the algorithm by omitting portions of the game tree that cannot influence the final decision. This reduces the number of calculations required to perform and allows the system to respond quickly. The move selected by the algorithm is subsequently translated to G-code. The CNC controller raises the pen, moves to the appropriate coordinates, draws the symbol, and then returns to the origin. The camera captures the new board state, and the loop begins again. This approach gives the AI a systematic and human-like way to compete with the player by selecting moves consistently and logically.

E. System Working

The system supports two operation modes, Normal Mode and Gaming Mode, each serving distinct objectives while using the same hardware and core software stack. Normal Mode demonstrates the CNC plotter's deterministic text and image plotting capabilities and provides a calibration and verification channel for motion control and pen actuation. Gaming Mode demonstrates the closed-loop integration of perception, decision making, and actuation whereby the system plays Tic-Tac-Toe against a human opponent. Both modes have the same mechanical setup but differ in workflow and control logic.

In Normal Mode, user input is provided either as typed text or as an image. The user enters a design into the PC interface, which is then transformed into G-code using tools such as Inkscape. G-code commands are delivered to the Arduino using the Universal G-code Sender (UGS). The Arduino, which runs GRBL firmware, understands these commands and sends them to the stepper and servo motors via the CNC shield. The X and Y movements are controlled by stepper motors, while the servo lifts and lowers the pen during drawing operations. Normal Mode includes safety checks (soft limits and homing verification) and produces a G-code execution log and a final captured image for post-hoc verification, making it the preferred mode for system checks and demonstrations.

In Gaming Mode, the system performs an autonomous perception–reason–act loop. A webcam continuously captures the board and passes preprocessed frames to a hybrid symbol recognition pipeline combining a YOLOv11 detector (deployed via an inference endpoint), contour-based heuristics, and a local .h5 classifier. Detections are filtered by confidence and mapped to the 3×3 grid to form a canonical game state. This state is evaluated by a Minimax agent augmented with alpha-beta pruning to produce the optimal move under the assumption of adversarial, perfect play. The chosen move is converted to an absolute X–Y coordinate and a short G-code segment which is transmitted to the Arduino. The plotter draws the AI's move on the board, after which the webcam updates the game state. The cycle continues until a win, loss, or draw condition is reached.

The block diagram of the system, shown in Figure 3, presents the hardware and data flow among modules. The operational

flowchart in Figure 4 outlines the logic sequence for both normal and gaming modes of operation.

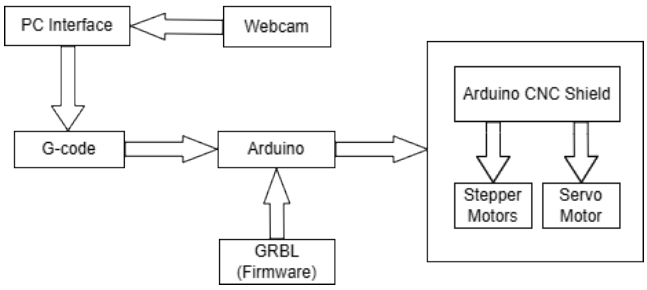


Figure 7 Block Diagram of the System

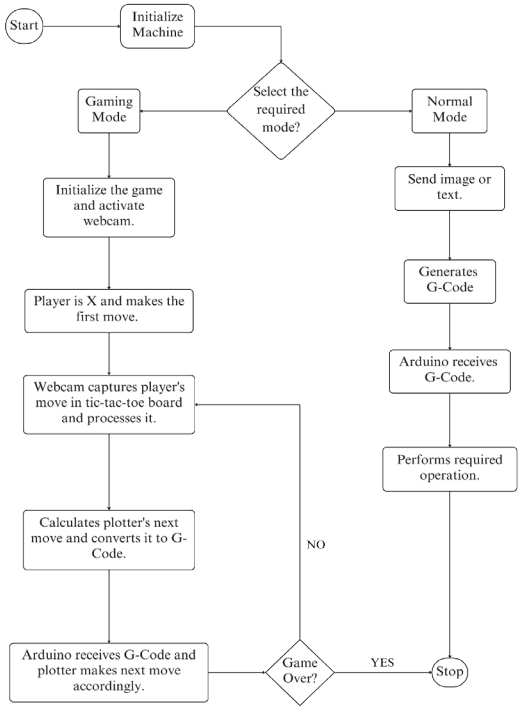


Figure 8 Flowchart of the System

IV. Results and Analysis

The performance of the developed system was evaluated through both the CNC drawing operation and the AI-based gameplay. The analysis focused on symbol detection accuracy, model stability, drawing precision, and interactive gameplay performance.

A. CNC Assembly and Motion Tests

The CNC machine was constructed using 3D-printed components and aluminum rods to produce a lightweight yet durable construction. Two NEMA 17 stepper motors were assigned to the X and Y axes, with a small servo motor controlling pen movement along the Z axis. The primary controller was an Arduino Uno equipped with a CNC shield and GRBL firmware. The initial calibration process validated the direction as well as travel distance of the stepper motor and pen raising height of the servo motor. Short G-code directives were used to imitate motor motion prior to program execution. The tests demonstrated precise axe coordination, consistent speed, and dependable home return. The motion stability ensured that the pen functioned smoothly and without vibration, which is required for precise mapping.

B. G-Code Generation and Execution

Text and image inputs were processed in Inkscape, and paths were translated to G-code using open-source extensions. The Arduino received these commands over a serial connection from the Universal G-Code Sender (UGS) interface. Each G-code command initiated a series of coordinated movements for the stepper and servo motors. Multiple test runs were conducted to ensure repeatability and mechanical alignment. The machine produced identical patterns on multiple trials, indicating good calibration. Improvements in pen-lift time and feed rate optimized drawing consistency. Prior to execution, the G-code visualization ensured that the commands matched the predicted tool pathways.

C. CNC Drawing for Text and Image Output

In Normal Mode, the system successfully executed text plotting and image reproduction tasks. Sample outputs included alphabets, basic geometric patterns, and short words. The machine maintained consistent pen pressure and line thickness, resulting in uniform strokes. Line quality was retained even after continuous operation, proving the mechanical design's durability and control consistency.

D. Tic-Tac-Toe Board Recognition Using YOLOv11

The Tic-Tac-Toe board was analyzed by capturing the state of the board by using a webcam. Each frame was processed to detect X and O symbols and their exact positions on the board. The model ensured that the board grid was properly aligned before performing symbol recognition. To improve reliability, the detection process was tested under different lighting conditions, camera angles, and pen thicknesses.

A custom dataset of 1,110 labeled images was used for training and evaluation. The images were divided into 972 for training, 92 for validation, and 46 for testing. Data augmentation techniques such as rotation, scaling, and brightness adjustments were used to increase variety and reduced sensitivity to small changes on the physical board.

The model reached a mean Average Precision at IoU 0.50 (mAP@50) of 0.995 on the test set. mAP measures how

precisely the predicted bounding boxes overlap with the actual symbol locations. A value close to 1.0 indicates that the boxes closely match the true positions of X and O marks, which demonstrates that the model localized symbols accurately.

Both X and O symbols achieved precision and recall values of 0.995. Precision measures the proportion of correct detections among all detections made by the model. High precision means the system rarely identified a non-symbol region as X or O. Recall measures how many true symbols were successfully detected out of all actual symbols on the board. High recall means the model rarely missed any marks. Together, these two values confirm that the detector was both careful and thorough in symbol recognition.

The confusion matrix summarizes the counts of correct and incorrect predictions for each class. All predictions in the matrix lay along the diagonal, indicating 100 % correct classification for both symbols. The absence of off-diagonal entries showed that the model did not confuse X with O, and it did not label empty spaces as symbols.

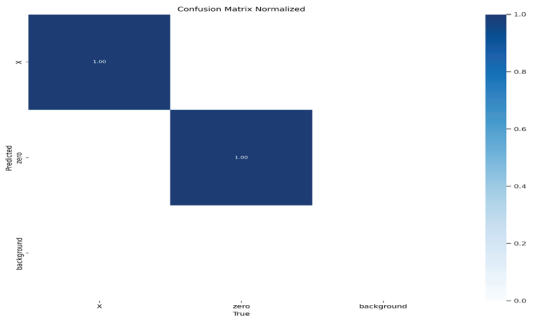


Figure 9 Confusion Matrix

The Precision–Recall curve plots how the two metrics change at different confidence thresholds. The curve remained close to the upper-right corner of the graph, which indicates stable performance. The mean Average Precision derived from this curve reached 0.995, showing that the model performed consistently across thresholds rather than excelling only at one point.

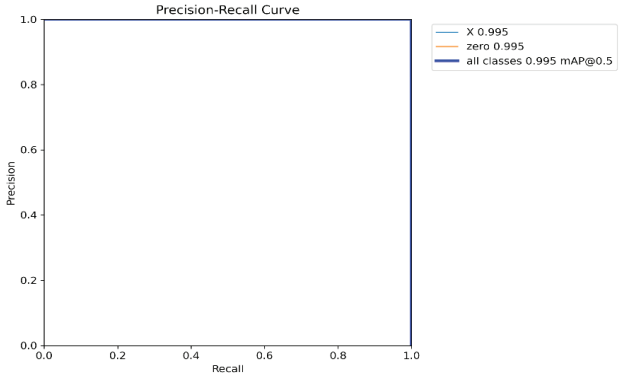


Figure 10 Precision-Recall Curve

The F1-score combines precision and recall into a single number to show overall balance. The F1–Confidence curve peaked at a score of 1.00 when the detection confidence was 0.852. This means that, when the model accepted detections with confidence above 85 %, it reached its best trade-off between missing symbols and avoiding false detections. The F1–Confidence curve peaked at a score of 1.00 when the detection confidence was 0.852. This means that, when the model accepted detections with confidence above 85 %, it reached its best trade-off between missing symbols and avoiding false detections.

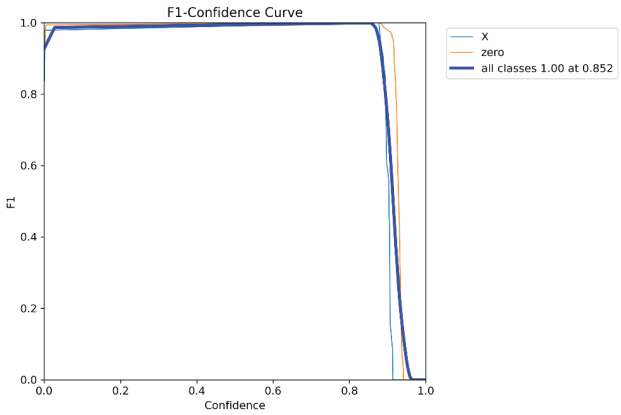


Figure 11 F1-Confidence Curve

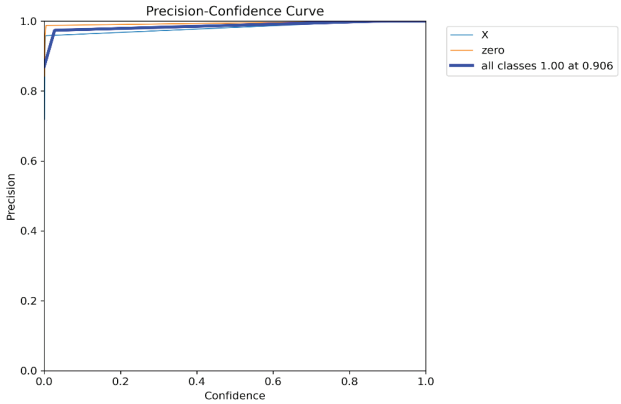


Figure 12 Precision-Confidence Curve

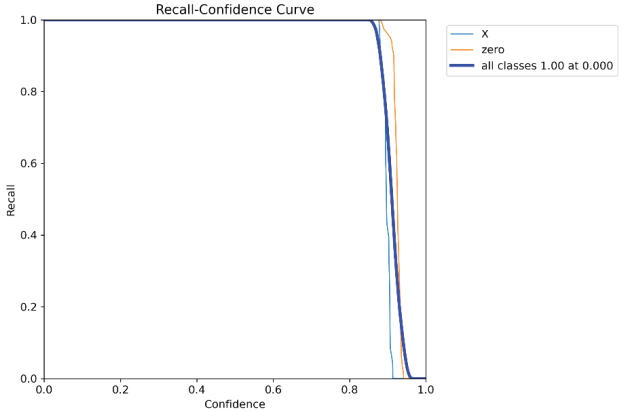


Figure 13 Recall-Confidence Curve

With Training and Validation Loss Metrics, the loss curves showed continuous improvement through training. The three tracked losses—box loss, classification loss, and distribution loss—fell steadily and leveled out near zero by the final epochs. The validation loss closely followed the training loss, which indicates that the model learned to detect symbols effectively without memorizing the dataset. The mAP@50 curve climbed gradually and reached 1.00 at convergence, confirming stable learning and strong generalization to unseen data.

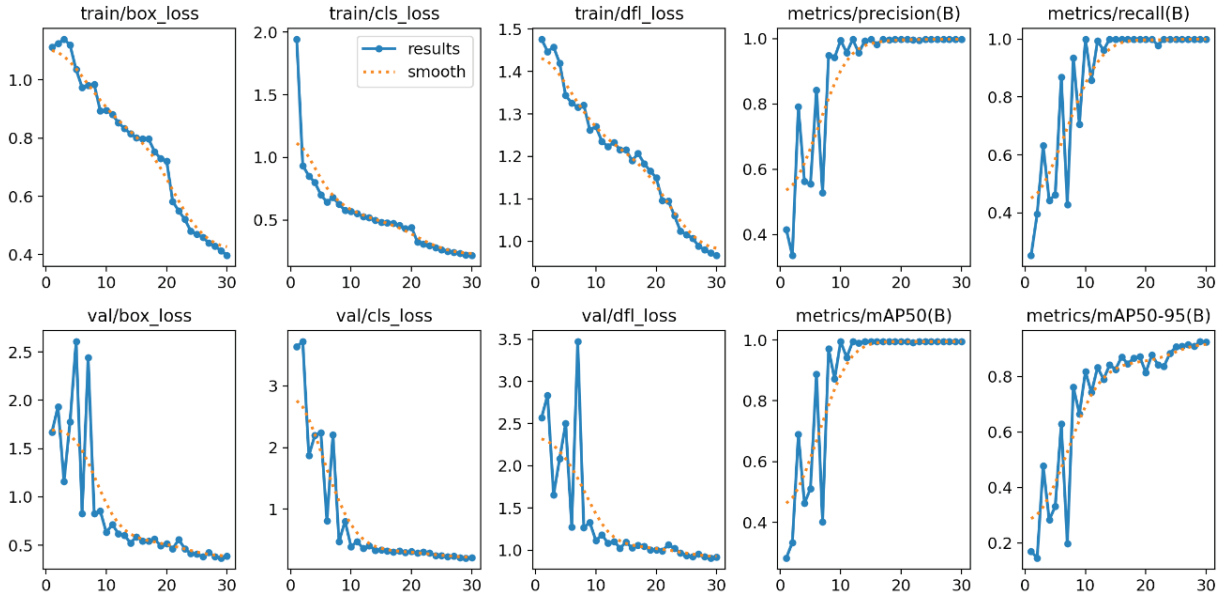


Figure 14 Training & Validation Loss Metrics

During live tests, the model processed each frame in less than 200 milliseconds on the deployed system, enabling real-time detection. Detection accuracy remained stable across different lighting and board angles. The system mapped the identified X and O boxes to the 3×3 grid and passed the updated board state to the AI module for decision-making.

The evaluation results prove that the YOLOv11 model achieved near-perfect detection accuracy while maintaining quick response times. The combination of high mAP, precision, and recall values verified that the vision system provided a dependable input layer for the AI and CNC modules that controlled gameplay and drawing.

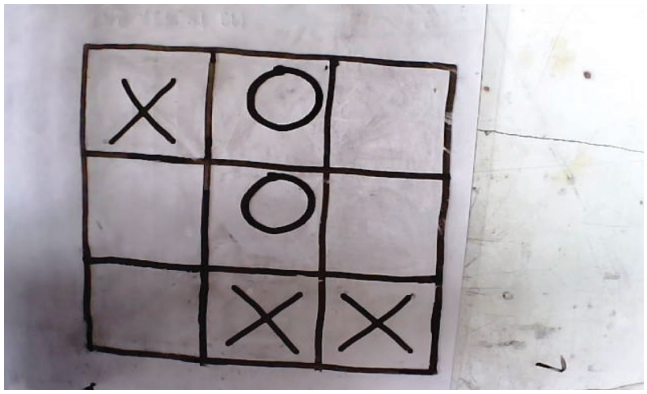
E. AI-Based Game Logic and CNC Integration

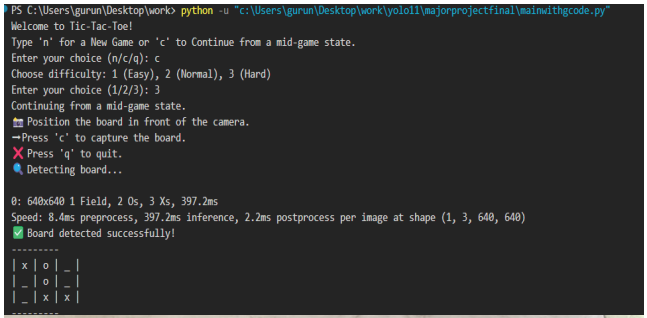
After identifying and updating the board state, the AI used the Minimax algorithm with Alpha-Beta pruning to decide it's next move. The algorithm determined which move maximized the AI's advantage while limited the player's chances of winning by simulating every potential outcome. The pruning technique reduced unnecessary calculations, resulting in a more efficient and responsive decision-making process. After determining the optimal step, the AI's decision was translated into G-code instructions for the CNC controller. The target grid cell was mapped to the corresponding X and Y coordinates, and the SG90 servo motor controlled the pen's vertical motion to mark the symbol on the board. The CNC machine, driven by GRBL firmware, processed the G-code and performed precise motions to draw the AI's preferred symbol.

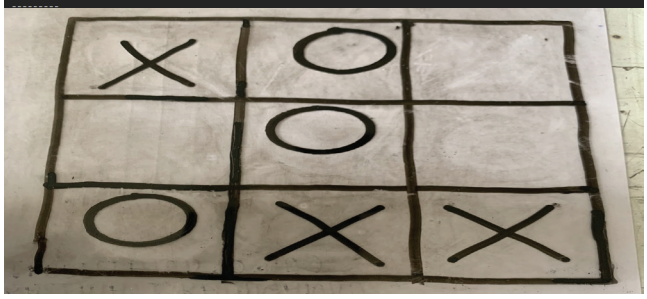
This process made it possible for the mechanical execution, AI decision logic, and object detection module to work together seamlessly. When the move was completed, the camera captured a new image of the board, which the algorithm analyzed again to determine the updated state. This closed loop of detection, decision-making, and drawing continued until the game ended in a win or draw. Throughout testing, the system demonstrated smooth communication between hardware and software components. The AI consistently executed logical and correct moves, whereas the CNC machine produced steady and clear drawings. Each module operated synchronously, showing that the visual, algorithmic, and mechanical subsystems worked together reliably during interactive gameplay.

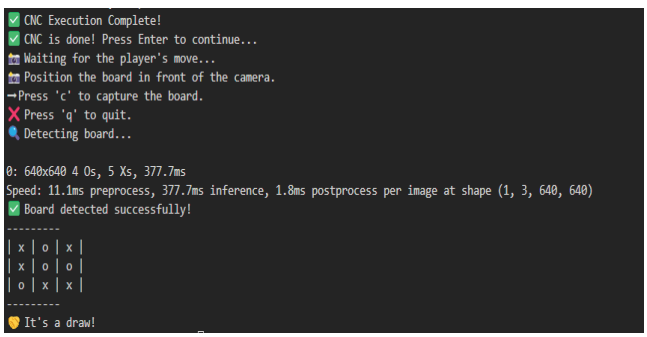
The complete sequence, shown in the figure below, illustrates the interaction between the object detection module, AI decision logic, and CNC control. Each stage from detecting the player's move to physically marking the AI's response occurred without

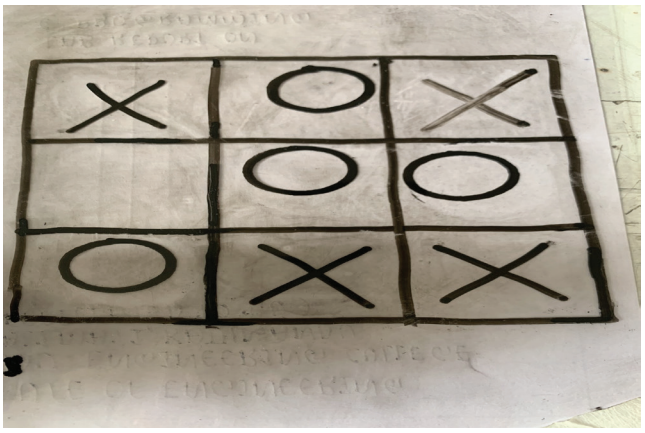
delay or mismatch. The integration confirmed the system's capability to function as a cohesive, intelligent prototype that combined computer vision, decision-making algorithms, and automated drawing into a unified workflow. The success of this integration highlighted the system's potential for use in educational and research environments. It demonstrated how AI-based reasoning could interact with low-cost mechanical systems to perform real-time, autonomous tasks with consistency and precision.











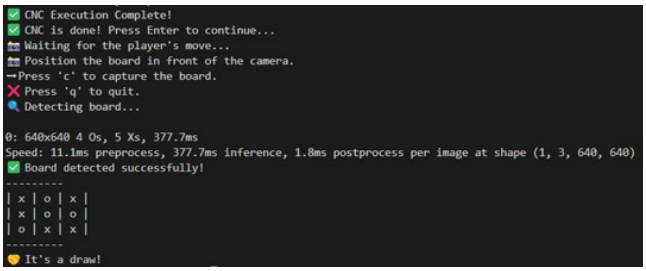


Figure 15 Series of Gameplay (Board State and Move Generation)

V. Conclusion

Our study successfully demonstrated the integration of computer vision, artificial intelligence, and CNC automation within a single low-cost educational prototype. The system performed both drawing and interactive gameplay, proving the feasibility of combining machine learning with mechanical control. The YOLOv11 model achieved accurate symbol detection, and the Minimax algorithm enabled logical, consistent gameplay. The CNC module executed all tasks smoothly, showing strong coordination between hardware and software. This work highlights how AI-driven decision-making can enhance small-scale automation systems and serve as a foundation for further development in intelligent manufacturing and educational robotics.

VI. Future Enhancement

Future development can focus on turning the prototype into a smarter and more versatile learning platform. Mechanical improvements such as stronger linear motion systems, smoother pen-lift control, and a dual-pen mechanism can allow multi-color drawing and finer handwriting replication. Expanding the system beyond Tic-Tac-Toe to include other interactive games or educational exercises would make it more engaging for students. An extension into handwriting generation and shape tracing can demonstrate precision control and writing algorithms. The software could incorporate adaptive vision with automatic calibration and

lighting correction so the camera self-adjusts to different environments. Implementing an active learning loop would let the model retrain on new or uncertain samples to improve detection over time. A smart error-handling routine could pause and re-home the plotter automatically if it detects positional drift or false detection. Integration with classroom I/O systems and wireless control could enable live demonstrations, teacher dashboards, and student interaction. These upgrades would evolve the project into a robust, intelligent, and adaptive educational platform that connects AI, automation, and learning in a practical way.

References

[1] S. F. Hyder, M. Ibrahim, M. Z. Adan and F. Mohammed, "CNC plotter machine," International Research Journal of Engineering and Technology (IRJET), vol. 7, no. 5, pp. 5300–5305, 2020.

[2] P. Patil, P. Lidhade, S. Khamkhar, Y. Mane, A. Dhole and S. Patil, "Design and development of CNC plotter machine," International Research Journal of Modernization in Engineering Technology and Science, vol. 2, no. 9, pp. 1069–1079, 2020.

[3] S. Kumar, A. Kumari, R. Begam and S. Kumar, "Advancements in CNC plotter technology: A comprehensive review and future prospects," International Journal of Communication Systems and Network Technologies, vol. 11, no. 1, pp. 71–78, 2023.

[4] A. Kumar, J. Krishnaraj and B. G. S. Reddy, "Mini CNC 2D sketcher for accurate building drawing," International Journal of Civil Engineering and Technology (IJCIET), vol. 8, no. 6, pp. 543–549, 2017.

[5] R. Garg and D. P. Nayak, "Game of tic-tac-toe: Simulation using Minimax algorithm," International Journal of Advanced Research in Computer Science, vol. 8, no. 7, pp. 1074–1077, 2017.

[6] R. Khanam and M. Hussain, "YOLOv11: An overview of the key architectural enhancement," arXiv preprint, 24 Oct. 2024. [Online]. Available: https://arxiv.org/pdf/2410.17725

[7] J. Yániz, M. Casalongue, F. J. Martinez-de-Pison, M. A. Silvestre, B. Consortium, P. Santolaria, and J. Divasón, "An AI-Based Open-Source Software for Varroa Mite Fall Analysis in Honeybee Colonies," Agriculture, vol. 15, no. 9, pp. 969, 2025. doi: 10.3390/agriculture15090969.