Computer Vision Powered Web Application to Identify Brick Fractures

Rafael I. Hualcas-López [1], Segundo E. Cieza-Mostacero [1]

[1] *Universidad Cesar Vallejo, Avenida Larco 1770 Trujillo, Perú*

***Abstract* – The objective of this study is to improve fracture identification in bricks at a factory located in Virú through the use of a computer vision application during the year 2025. The study employed a pure experimental design. The population consisted of all brick quality inspection processes carried out by construction companies, with a sample size of 30 processes. The data collection technique employed was direct observation, and the instrument used was an observation form. The computer vision application was developed using Python 3.12 and the Flask framework, with MongoDB as the database management system and Mobile-D as the software development methodology. Descriptive analysis was performed using Microsoft Excel 2019, while inferential analysis was conducted using Jamovi 2.6.44. The results showed a 32.91% increase in fracture identification, a reduction of 114.4 seconds in the average time to identify brick fractures, and a 30.6% improvement in identification accuracy. In conclusion, the use of a computer vision application significantly improves fracture identification in brick factories.**

***Keywords* – Applications, computer vision, bricks.**

DOI: 10.18421/TEM153-21
https://doi.org/10.18421/TEM153-21

Corresponding author: Rafael I. Hualcas-López,
Universidad Cesar Vallejo, Avenida Larco 1770 Trujillo, Perú
Email: rhualcas@ucvvirtual.edu.pe

Received: 06 August 2025.
Revised: 25 February 2026.
Accepted: 05 March 2026.
Published: 27 August 2026.

1. Introduction

The production of bricks represents an essential activity within the global construction sector. These materials are valued for their durability and strength, its origin can be traced back to ancient buildings, remaining relevant in modern construction. However, significant imperfections can occur during the manufacturing process; defects such as cracks, fissures, and fractures can compromise the functionality of the finished product.

Moreover, 35% of the total waste deposited in landfills is generated by the construction sector, which, according to the United Nations (UN), amounts to countless tons of waste annually. Additionally, the United Nations Environment Program (UNEP) reports that this waste is often not properly disposed of, resulting in harmful consequences for both the environment and public health [1].

Bricks, being ceramic materials, must meet strict technical requirements, such as water resistance and high compressive strength. Their production depends heavily on clay-rich soils, which represent approximately 33% of the Earth's land surface [2].

In countries like Cameroon and much of Latin America, these soils are widely used in the brick industry. In Peru, as of 2019, there were 15 industrial factories and approximately 2,500 artisanal brick kilns, with an annual production of nearly 9.5 million tons and sales exceeding S/ 1.6 billion (PEN); 70% of consumption was concentrated in the capital [3].

In the local context of the province of Virú (La Libertad), several deficiencies were identified in the manufacturing processes. During a technical visit to a local brickmaking company, issues were observed such as inadequate cleaning of raw materials, bricks falling off conveyor belts, poor selection of damaged bricks, and disorganization in assembly and storage areas [4].

However, in response to these challenges, industries have begun incorporating Information Technologies (IT) as quality control tools. Among these, innovative approaches such as Computer Vision (CV), Artificial Neural Networks (ANN), Machine Vision (MV), and Deep Learning (DL) have stood out.

These aim to automate processes, reduce human error, and improve productivity [5], [6].

Computer vision, which is part of artificial intelligence, enables systems to interpret visual data from images or videos, facilitating the rapid, accurate, and non-invasive identification of defects. This helps improve product quality and reduce industrial waste [7].

Furthermore, technology has been shown to significantly improve the accuracy in detecting defects such as cracks, breakages, and fissures in bricks, while also enabling real-time monitoring with high efficiency and performance [8]. However, informality in artisanal brick production remains a major issue in regions such as Virú, where the lack of regulation results in low-quality products, precarious working conditions, and considerable environmental impact [9].

Based on the issues described, this research poses the main question*: How does a computer vision fat a factory in Virú during the year 2025?* And the following specific research questions: *How does a computer vision application increase the number of bricks identified with fractures in a factory? How does a computer vision application reduce the average time required for fracture identification in a factory? How does a computer vision application improve the identification accuracy of brick fractures in a factory?*

Accordingly, the general hypothesis was established as: If a computer vision application is used, then fracture identification in bricks will improve in a factory in Virú, The specific hypotheses were: If a computer vision application is used, then the number of bricks identified with fractures in a factory will increase, If a computer vision application is used, then the average time required for fracture identification in a factory will decrease, If a computer vision application is used, then the identification accuracy of brick fractures in a factory will increase.

2. Literature Review

This section describes the main theoretical foundations that support the research, related to the dependent variable (fracture identification in bricks) and the independent variable (computer vision application).

2.1. Mobile Applications

Mobile applications have evolved as key tools within the digital transformation of industrial processes. These are software programs developed for mobile devices (such as smartphones and tablets) that offer an enhanced user experience through touchscreens and connectivity [10].

Moreover, features such as connectivity, interoperability, and cross-platform access facilitate rapid adoption by users in industrial environments, particularly in tasks such as monitoring and quality control [11].

2.2. Computer Vision

Computer vision comprises an area of artificial intelligence that enables systems to analyze and understand visual data from images or videos. This technology allows computers to comprehend the visual environment in a way that mimics human perception, enabling systems to solve problems through algorithms trained for specific tasks [12].

Furthermore, the ability of computers to interpret and analyze their environment has opened new possibilities in areas such as healthcare, transportation, manufacturing, and occupational safety, contributing to process optimization and improving quality of life [13].

2.3. Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) have revolutionized image analysis methods in the field of computer vision. These models excel at accurately identifying visual elements such as edges, cracks, fractures, or fissures. In industrial contexts, CNNs enable the implementation of computer vision systems capable of recognizing complex structures and classifying them with high precision. This is especially useful for the automated identification of fractures in bricks, even under real production conditions [14].

2.4. Machine Vision Models

Machine vision models can be categorized into three main types: first, supervised algorithms, which are trained on labeled data that allows the system to learn to recognize specific patterns; second, traditional models, which use predefined mathematical algorithms to analyze visual features without relying on artificial intelligence; and finally, deep learning, which learns directly from visual features without requiring explicit rules, achieving high levels of accuracy in activities such as object categorization or defect identification [15].

2.5. Brick Fracture

Brick fracture can compromise both the aesthetic aspect and the structural integrity of buildings. These fissures may originate from multiple factors, such as excessive load, ground settlement, temperature fluctuations, ambient humidity, freeze–thaw cycles, or inherent material defects.

Their manifestation may appear as microcracks or deep fissures, directly affecting the durability of bricks by allowing moisture penetration and the progressive deterioration of their structure [16].

2.6. Model Assessment

According to [17] the YOLOv8 model presents architectural improvements such as CSPDarknet, PANet, and an anchor-free approach, which allow for faster and more accurate detection. Additionally, the use of image transformations and Test Time Augmentation (TTA) enhances the model's generalization and performance, achieving high levels of precision and recall in industrial environments.

3. Related Works

This research is based on previous studies addressing the application of computer vision in the identification of structural defects using predictive models. The studies were selected from recognized academic databases such as EBSCO, Scopus, Web of Science, ScienceDirect, and IEEE Xplore, prioritizing those that use image classification models and anomaly detection.

The following are some relevant studies that demonstrate the effectiveness of these approaches in various contexts.

For instance, the accuracy, prediction speed, and potential of different deep learning models for the automated identification of bumblebee species in North America were evaluated using a dataset of 89,000 images spanning 36 species. Among the models assessed, InceptionV3 emerged as the top-performing model, achieving an accuracy of 91.6% and an average inference time of 3.34 milliseconds per image [18].

On the other hand, [19] developed a machine vision algorithm aimed at identifying hip fractures in X-ray images. The goal was to validate its effectiveness in heterogeneous populations from two medical institutions. The dataset included 4,235 radiographs from Chang Gung Memorial Hospital (CGMH), along with an external validation set of 500 images from both CGMH and Stanford's trauma center. The algorithm achieved areas under the curve (AUC) of 0.98 and 0.97 for both cohorts, respectively. All performance metrics exceeded 90%, including sensitivity, specificity, predictive values, and accuracy. This study demonstrates the high potential of computer vision models for diagnostic and classification tasks in medical imaging, showing methodological parallels with structural defect detection.

Finally, [20] focused their study on the automation of brick segmentation and crack detection in masonry walls using computer vision and deep learning. They worked with a dataset of 107 annotated images with multiple classes, including masonry blocks, openings, lintels, random objects, and background. They evaluated various segmentation architectures such as U-Net, DeepLabV3+, U-Net (SM), LinkNet (SM), and FPN (SM), exploring different combinations of loss functions, optimizers, and training parameters. The results showed accuracies ranging from 95.97% to 96.27%, with DeepLabV3+ standing out as the best-performing architecture. This study highlights the potential of deep learning for structural documentation and analysis in civil engineering applications.

4. Research Purpose

This section presents the main objective and the specific objectives that guide the present research, aimed at optimizing the identification of fractures in bricks through the use of computer vision in an industrial setting

The main objective was to improve brick fracture identification at a factory in Virú through the use of a computer vision application during the year 2025. In addition, the specific objectives were: to increase the number of bricks identified with fractures, to reduce the average time for fracture identification in bricks, and finally, to improve the accuracy of fracture detection in bricks.

5. Methodology

This section describes the type of research, the experimental design, the variables considered, as well as the population, sample, data collection techniques, and the scope of the study. It provides a detailed account of the approach used to assess the impact of computer vision in identifying brick fractures within an industrial environment.

This research was of an applied type, aimed at solving a practical problem related to brick fracture identification, through the use of a computer vision-based application. According to [21], applied research seeks to transfer theoretical knowledge to the practical domain, generating direct benefits in the environment where it is implemented.

Furthermore, regarding the research design, this study followed a pure experimental approach, since the independent variable (use of computer vision) was intentionally manipulated to evaluate its impact on the dependent variable (fracture identification in bricks). Two groups were formed: the Control Group (CG), which did not use the computer vision application, and the Experimental Group (EG), which did.

Participants were randomly assigned to each group, allowing for control of external variables and ensuring the internal validity of the study [22].

The methodological approach was quantitative-experimental, in which numerical data related to the study indicators were collected and analyzed. This approach allows for the establishment of causal relationships between variables and the testing of hypotheses through rigorous statistical analysis [23].

5.1. Research Variables

The independent variable was computer vision, considered a subfield of artificial intelligence that enables systems to automatically process and interpret images through algorithms trained with convolutional neural networks (CNNs) [12]. His variable was measured using the Presence/Absence indicator, categorized on a nominal scale, where the value "No" was assigned before the implementation of the application and "Yes" after its development and integration.

The dependent variable was fracture identification in bricks, defined as the method for identifying visible anomalies (fractures, fissures, or breaks) that may compromise the integrity of the material. According to [16] these defects may arise from structural or environmental factors, manifesting as microcracks or deep fractures.

For this variable, observation forms were used as the instrument, and it was measured through the following indicators: number of bricks identified with fractures (NBIF), average time to identify brick fractures (ATIBF), and identification accuracy of brick fractures (IABF). All indicators were assessed on a ratio scale.

5.2. Population and Sample

The population consisted of all brick quality inspection processes conducted by construction companies in Peru. For the study group in this research, 30 inspection processes conducted in a single factory were considered. These served as a test set to validate the model's effectiveness [21]. This technique is an effective method for defining samples aligned with the objectives of the research [24].

5.3. Data Collection Techniques and Instruments

The applied technique was indirect observation, through the collection of images using VisionPlatform.ai, which allowed for data collection from the study subject without direct intervention. An observation form was used as the instrument for each of the indicators: NBIF, ATIBF, and IABF. The form was used to collect structured data for analyzing or evaluating the subject of study.

5.4. Scope and Delimitation

The scope of the research focused on improving real-time fracture identification in bricks through a computer vision application developed using the YOLOv8 architecture. This application employs convolutional neural networks and was implemented in Python. And was incorporated in the production process, and the system used cameras to visually capture moving bricks, without including physical sensors or manual inspection techniques. The research was limited to the process of automated visual quality control.

6. Project Development

This section presents the design and implementation of a web application based on computer vision for the identification of fractures in bricks. The proposed solution covers the entire process, from data collection and preparation to the training of the YOLOv8 model and its integration into an MVC-style web architecture for deployment in a factory setting.

The project is based on a computer vision application designed to enable the automatic identification of fractures in bricks using computer vision techniques. This tool aims to improve quality control in a factory located in the province of Virú by reducing manual intervention and increasing process accuracy.

The application was built using a client-server architecture, employing technologies such as Python (Flask) for the backend and HTML, CSS, and JavaScript for the frontend. In addition, an object detection model based on YOLOv8 was integrated, specifically trained to identify three classes: good, fissure, and break (Figure 1).

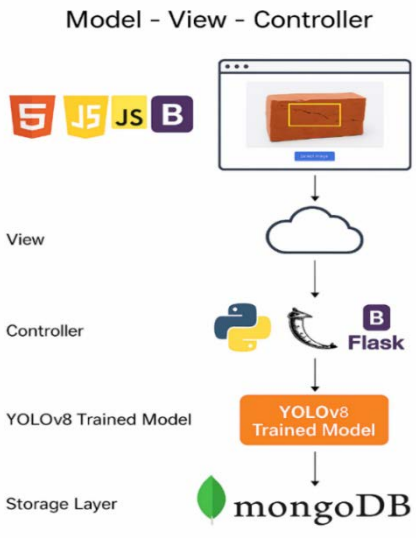

Figure 1. Application's MVC architecture

6.1. Data Compilation

For model training, images were manually collected directly at the factory. Each image was captured individually, and a customized preparation and editing process was applied, including brick cropping, background replacement, focus enhancement, and lighting adjustment, in order to maximize dataset quality (Table 1).

Table 1. Image collection process for model training

No°	Stage	Description
1	Data source	Direct image capture at the brick factory
2	Capture method	Manual, image by image
3	Number of images	367 images
4	Image types	Brick images with fractures (fissure and break) and bricks in good condition
5	Tool used	Digital camera
6	Applied Processing	Brick cropping, background replacement, focus and lighting enhancement
7	Data labeling	Manual using Make Sense tool, assigning classes: good, fissure, break
8	Final format	Image in JPG/PNG format + YOLO labeling format (TXT)

6.2. Dataset Preparation

The images were manually labeled using tools such as Make Sense, assigning labels to regions corresponding to the defined classes. Subsequently, they were split into training and validation sets, and the dimensions were normalized to meet YOLOv8 input requirements (Table 2).

Table 2. Model training features

No.	Process	Description
1	Annotation tool	Make Sense
2	Defined classes	Pass, Fissure, Break
3	Annotation format	YOLO format (.txt files with coordinates and labels)
4	Dataset split	80% training, 20% validation
5	Standard dimensions	YOLOv8-compatible resolution (i.e., 640×640 pixels)
6	Preprocessing objective	Ensure consistency and compatibility with model input

6.3. Model Training

The model was trained using YOLOv8 in a controlled environment, by adjusting hyperparameters such as the number of epochs, learning rate, and dataset size. PyTorch was used as the base framework, and evaluation metrics including precision, recall, and mAP were applied to validate performance (Table 3).

Table 3. YOLOv8 model training settings

No.	Parameter / Element	Value or Description
1	Base framework	PyTorch
2	Model architecture	YOLOv8
3	Dataset	Labeled images (good, fissure, break)
4	Dataset split	80% training, 20% validation
5	Number of epochs	50 epochs
6	Batch size	16
7	Learning rate	0.05 (adjusted based on validation)
8	Input dimensions	640×640 px
9	Evaluation metrics used	Precision, Recall, mAP (mean Average Precision)

6.4. Web Application Integration

Once the model was trained, it was integrated into the web application using a Flask API that receives images, processes them on the server, and returns the results to the client. The interface allows users to select the available camera and stream live video to identify defective bricks in real time. Additionally, a function was developed to automatically generate PDF reports, including statistics such as total number of bricks, model precision, and processing times.

Figure 2 illustrates the interaction between the client (user's browser) and the server, where the modules for image capture, processing with the YOLOv8 model, result storage, and PDF report generation are integrated. The architecture enables real-time monitoring of bricks via camera, visualization of statistics, and automatic report generation.

Figure 2. Live monitoring system interface

7. Frontend Backend and Frontend Development

This section describes the development process of the web application using the agile Mobile-D methodology. It details the development phases, from requirements identification to the integration of the YOLOv8 model into the backend using Flask, as well as the implementation of the frontend for real-time monitoring, report generation, and metric visualization

For the webApp development, the Mobile-D agile development methodology was used, focusing on the rapid and efficient construction of high-quality mobile and web applications. This methodology consists of five iterative phases: exploration, initialization, production, stabilization, and testing, allowing for the progressive evolution of the system, with a focus on functionality and continuous involvement of the end user [25].

7.1. Phase 1: Exploration

The project's main stakeholders were identified, including the plant manager, quality supervisors, and operators. In addition, the initial requirements document was developed, key functionalities such as automatic fissure detection and report generation were defined, and tools such as Flask, Python, PyTorch, YOLOv8, JavaScript, and MongoDB were selected. Use cases were also specified, and initial system diagrams related to the application were created. A review schedule was also established for each story (Table 4).

Table 4. Project process review schedule

Story	Story Name	Priority	Review Date
H001	Identification of key stakeholders	HIGH	April 1st, 2025
H002	Development of requirements document	HIGH	April 3rd, 2025
H003	Definition of main functionalities	HIGH	April 5th, 2025
H004	Integration with ERP database	HIGH	May 7th, 2025
H005	System access	HIGH	May 9th, 2025
H006	Results report	HIGH	May 11th, 2025

7.2. Phase 2: Initialization

In this phase the early prototype versions with static imaging detection were implemented, and the resources than enabled continued development were prepared. During this phase, the trained YOLOv8 model was integrated for the first time through a Flask powered API. Likewise, the general system diagram was established: the user interface sends an image to the /procesar_frame endpoint, and the backend processes it using the model, storing the results obtained. Once the monitoring session ends, the /generar_reporte endpoint is accessed, which triggers a comparison between the predictions generated and the labels contained in the etiquetas.json file.

Based on this comparison, the model's performance is assessed using precision metrics, and finally, a report is generated with the obtained results (Figure 3).

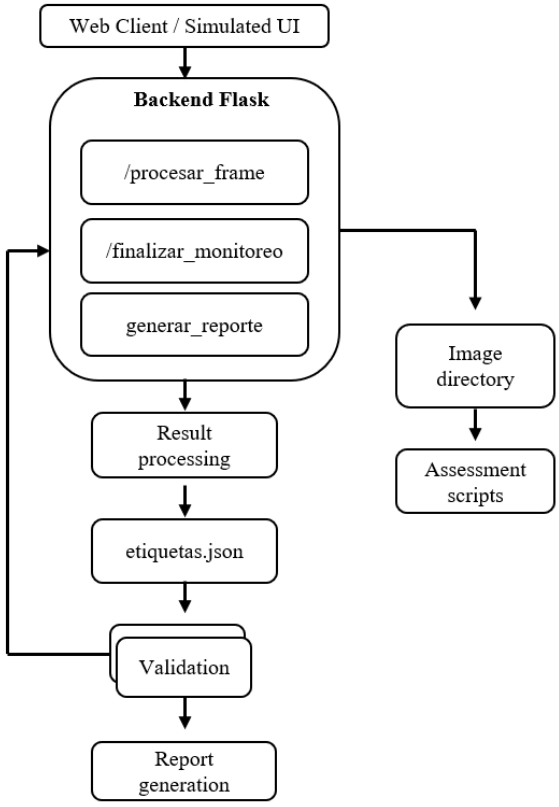

Figure 3. Functioning flowchart

7.3. Phase 3: Production

This phase included the installation of dependencies, complete backend and frontend development and the implementation of interfaces for live monitoring and image upload. The interfaces were designed using HTML and CSS, with dynamic control via JavaScript, allowing camera selection, real-time visualization of predictions and metrics, as well as automatic generation of PDF reports. To detect fissures in bricks in real time, the procesar_frame (process_frame) method was implemented. It receives as input an image sent from the client's camera (file_storage) and the user's identifier. The image is converted into a NumPy array using OpenCV, thus enabling its processing by the previously trained YOLOv8 detection model.

Detection is performed using the procesar_imagen function, which returns a list of predicted classes such as "bueno" (pass), "fisura" (fissure), or "rotura" (fracture). These predictions are then passed to the procesar_resultados (process_results) method to update monitoring counters and metrics: total number of bricks processed, quantity of each detected class, average fissure detection time, and specific precision for that class, comparing with the previously defined labels in the etiquetas.json (labels.json)file.

Finally, a summary in the form of a dictionary is returned with the calculated metrics, which are either displayed in real time through a web interface or stored for report generation (Figure 4).

```python
def procesar_frame (file_storage, usuario_id="default", frame_name=None):
    frame_bytes = file_storage.read()
    nparr = np.frombuffer(frame_bytes, np.uint8)
    frame = cv2.imdecode(nparr, cv2.IMREAD_COLOR)
    if frame is None:
        return {"error": "No se pudo leer la imagen"}
    # Simular detección (REEMPLAZA por detección real con tu modelo YOLOv5)
    predicciones = procesar_imagen(frame) # type: ignore
    # Usar el frame_name recibido, no sobrescribirlo
    if not frame_name:
        # Opcional: generar nombre por hash si no se recibe
        import hashlib
        frame_hash = hashlib.md5(frame_bytes).hexdigest()
        frame_name = frame_hash + ".jpg"
    # Procesar y actualizar métricas
    cliente_monitoreo.procesar_resultados(usuario_id, predicciones, frame_name=frame_name)
    # Obtener resultados
    return cliente_monitoreo.obtener_métricas_finales(usuario_id)
```

Figure 4. Source code of the "procesar_frame" method"

The source code of the web application developed for the identification of fractures in bricks using computer vision is available for consultation and download through Google Drive, in order to promote transparency, reproducibility and open access to knowledge [26].

7.4. Phase 4: Stabilization

During the stabilization phase, backend functions, the overall system architecture, and system operations were documented.

Additionally, the integrity of the video processing flow, data storage, and client-server communication were validated.

System activities were also verified and documented (Table 5).

Table 5. System activity verification

Verified / Documented Element	Description
Backend functions	Endpoints (/procesar_frame, /generar_reporte, etc.) were documented, including their parameters, expected responses, and error handling.
Overall system architecture	A technical diagram was created detailing the data flow between frontend, backend, and the AI model.
Image processing	It was confirmed that the input images are properly analyzed by the model without data loss.
Results storage	It was verified that the generated data (detections, metrics) are stored consistently and are accessible.
Client-server communication	AJAX/HTTP requests from the frontend were tested to ensure they reach the backend correctly and that the responses are complete and valid.

7.5. Phase 5: Testing

Finally, during the testing phase, both unit and functional tests were carried out for the model and the application.

These tests included validation of the model's accuracy under real factory conditions, evaluation of average detection time, and verification of accurate report generation with summarized metrics. System tests were also performed (Table 6).

Table 6. Unit and functional system tests

Test Type	Description	Expected Outcome
Unit tests (backend)	Verification of specific functions such as detection, counting, and frame processing.	Each function returns correct results without errors.
Functional tests (system)	Evaluation of the complete system: camera, detection, metrics, and reports.	Complete workflow with no visible failures.
Validation in real environment	The model was tested under actual factory conditions (lighting, background, real objects).	High detection accuracy; no critical false positives.
Detection time measurement	Evaluation of average inference time per frame.	Real-time detection (e.g., <1s per frame).
Report verification	Reports in PDF format were reviewed for correct data: counts, percentages, and times.	Reports accurately reflect the processed data.

8. Findings

This section presents the results obtained after implementing the computer vision application for identifying brick fractures in a factory located in Virú. Both descriptive and inferential analyses are included, comparing the performance of the traditional system (control group) with the automated system (experimental group) based on three key indicators: number of bricks identified with fractures (NBIF), average identification time (AITBF), and detection accuracy (DAIBF).

8.1. Descriptive Analysis

The following section presents the data obtained from the final test conducted with both the control group (CG) and the experimental group (EG), according to the indicators established in the research: Number of bricks identified with fractures (NBIF), Average time to identify brick fractures (AITBF), and Precision in identifying brick fractures (PIBF).

Table 7. Post-test results for the NBIF indicator in CG and EG

Research Indicators	Average		% Higher Than Average	
	CG	EG		
NBIF	5.47	7.27	EG: 57.06	CG: 42.94

As shown in Table 7, the control group identified an average of 5.47 fractured bricks, while the experimental group achieved an average of 7.27, representing an increase of 1.8 units and a percentage difference of 32.91%. This indicates that the experimental system was able to identify a greater number of fractured bricks under the same conditions. Moreover, in proportional terms, the experimental group accounted for 57.06% of the total fractured bricks identified, compared to 42.94% for the control group, highlighting a significant improvement in performance with the implemented system compared to the traditional method.

Table 8. Post-test results for the AITBF indicator in CG and EG

Research Indicators	Average		% Higher Than Average	
	CG	EG		
AITBF	150.2	35.8	EG: 19.24	CG: 80.76

Table 8 shows that the control group required an average of 150.2 seconds to identify brick fractures, while the experimental group reduced this time to 35.8 seconds, representing a significant improvement of 114.4 seconds. This indicates that the experimental group was able to reduce the identification time by approximately 76% compared to the control group, clearly demonstrating the benefits of implementing the computer vision system.

Table 9. Post-test results for the PIBF indicator in CG and EG

Research Indicators	Average		% Higher Than Average	
	GC	GE		
PIBF	19.1	49.7	GE: 72.25	GC: 27.75

Table 9 shows that the control group achieved an average precision of 19.1% in identifying fractured bricks, while the experimental group reached an average of 49.7%, indicating an absolute increase of 30.6 percentage points. This result demonstrates a considerable improvement in precision with the computer vision–based system (YOLOv8 + Flask), underscoring the effectiveness of the integrated model in reliably and automatically detecting brick defects.

8.2. Inferential Analysis

To carry out the inferential analysis, normality tests and hypothesis tests were applied. Decision criteria were also established for the post-test results of both the control group (CG) and the experimental group (EG) in relation to the following indicators: Number of bricks identified with fractures (NBIF), Average time to identify brick fractures (AITBF), and Precision in identifying brick fractures (PIBF): H_0 (null hypothesis): Post-test data follow a normal distribution. Ha (alternative hypothesis): Post-test data do not follow a normal distribution. If $p < 0.05$, then the null hypothesis (H o) is rejected and the alternative hypothesis (Ha) is accepted. If $p \geq 0.05$, then the null hypothesis (H o) is accepted and the alternative hypothesis (Ha) is rejected.

Since the number of data points for each indicator was fewer than 50 entries, the Shapiro-Wilk test was used to assess normality.

Table 10. Normality test using Shapiro-Wilk

No.	Indicator	p-value (CG)	p-value (EG)
1	NBIF	0.034	<.001
2	AITBF	0.001	<.001
3	PIBF	0.003	0.039

Based on the results in Table 10, the p-value is less than 0.05 in at least one group for each indicator. Therefore, the data do not follow a normal distribution. Consequently, the non-parametric Mann–Whitney U test was used to analyze the differences between independent groups. For hypothesis testing, μ1 was defined as the control group and μ2 as the experimental group.

Table 11. Mann–Whitney U test

No.	Indicator	H0	Ha	p
1	NBIF	H0 = μ1 ≥ μ2	Ha = μ1 < μ2	0.450
2	AITBF	H0 = μ1 ≤ μ2	Ha = μ1 > μ2	<.001
3	PIBF	H0 = μ1 ≥ μ2	Ha = μ1 < μ2	<.001

According to the results shown in Table 11, for the AITBF and PIBF indicators, the p-value is less than 0.05. This provides statistical evidence to reject the null hypothesis (H_0) and accept the alternative hypothesis (H_a). In contrast, for the NBIF indicator, the p-value is 0.450, which is greater than 0.05. This indicates insufficient statistical evidence to reject the null hypothesis, which therefore remains accepted.

In conclusion, the alternative hypothesis (H_a) is accepted for the AITBF and PIBF indicators, indicating significant improvements with the experimental system. However, for NBIF, the null hypothesis (H_0) is retained due to a p-value greater than 0.05.

9. Discussion

This section analyzes and interprets the results obtained, comparing them with previous studies. It evaluates the impact of the computer vision system in relation to the three main indicators: number of fractured bricks identified (NBIF), average identification time (AITBF), and system precision (PIBF), highlighting the improvements achieved over the traditional method.

Regarding the Number of Bricks Identified with Fractures (NBIF) indicator, the post-test results for the control group (CG) showed an average of 5.47 fractured bricks identified, while the experimental group (EG) achieved an average of 7.27 bricks, representing an increase of 1.8 units.

This indicates that the experimental system was able to identify a greater number of fractured bricks. Similar findings were reported in [16], where an automatic brick segmentation model with crack detection was implemented, achieving accuracies exceeding 95%. This highlights the utility of such technologies for identifying anomalies in masonry walls. Likewise, [13] demonstrated that the use of automated computer vision tools improves reliability in the analysis of fractured surfaces, increasing the number of faults detected compared to conventional methods.

In terms of the Average Time to Identify Brick Fractures (AITBF), the post-test results for the control group (CG) showed an average time of 150.2 seconds, while the experimental group (EG) reduced that time to 35.8 seconds, revealing a decrease of 114.4 seconds in the average time required for fracture identification. This finding aligns with results reported by [19], who developed an algorithm for detecting bone fractures in X-rays and achieved inference times of less than 2 seconds per image, demonstrating that computer vision models can significantly reduce visual analysis time without compromising diagnostic quality. Similarly, [18] reported that their image classification system using InceptionV3 achieved a latency of just 3.34 milliseconds per image when identifying species, further validating the applicability of these systems in tasks requiring fast, real-time response.

With respect to the Precision in Identifying Brick Fractures (PIBF) indicator, the post-test for the control group (CG) yielded an average precision of 19.1%, while the experimental group (EG) achieved an average of 49.7%, showing an improvement of 30.6 percentage points in precision. In this regard, [7] found that a computer vision system based on CNN achieved 92% precision in identifying cracks in industrial materials, while [17] reported that their YOLOv8 implementation in construction environments reached high levels of precision and recall in object detection under real conditions, thanks to structural enhancements such as CSPDarknet and PANet. Likewise, [12] documented a computer vision model with 91% precision in identifying improper mask use in public spaces, confirming that these techniques are effective even in environments with visual noise.

10. Conclusion

The application of computer vision in industrial environments represents a significant advancement in improving quality control processes. Based on the results obtained, the following conclusions are presented:

Given that the post-test of the control group (CG) recorded an average of 5.47 fractured bricks identified, compared to 7.27 in the post-test of the experimental group (EG), it was concluded that there was an increase of 1.8 units. This indicates that the experimental system was able to identify a greater number of fractured bricks.

After observing an average time of 150.2 seconds in the post-test of the control group (CG) and 35.8 seconds in the post-test of the experimental group (EG) for identifying fractured bricks, it was concluded that there was a reduction of 114.4 seconds in the average identification time.

With 19.1% precision in identifying fractured bricks in the post-test of the control group (CG) and 49.7% precision in the post-test of the experimental group (EG), it was concluded that there was an improvement of 30.6 percentage points in detection precision.

Finally, it was concluded that the use of a computer vision–based web application significantly enhances the identification of brick fractures in a factory located in Virú.

Acknowledgment

Sincere gratitude is extended to César Vallejo University for its support, as well as for providing the essential resources and funding necessary for the development of this research.

References:

[1]. Raúl Alberto, C. A., et al. (2023). Application of Lean Manufacturing and its effect on the productivity of a brick company in Peru. *LACCEI*, *1*(8). Doi: 10.18687/LACCEI2023.1.1.545

[2]. Arias, F. G. (2012). *El proyecto de investigación: Introducción a la metodología científica* (6th ed.). Editorial Episteme.

[3]. Choi, J., et al. (2021). Practical computer vision application to detect hip fractures on pelvic X-rays: a bi-institutional study. *Trauma Surgery & Acute Care Open*, *6*(1). Doi: 10.1136/tsaco-2021-000705

[4]. Cordero, Z. R. V. (2009). La investigación aplicada: una forma de conocer las realidades con evidencia científica. *Revista educación*, *33*(1), 155-165. Doi: 10.15517/revedu.v33i1.538

[5]. Crespo, F., et al. (2022). A computer vision model to identify the incorrect use of face masks for COVID-19 awareness. *Applied Sciences*, *12*(14), 6924. Doi: 10.3390/app12146924

[6]. Ding, S., et al. (2023). Multi-scale polar object detection based on computer vision. *Water*, *15*(19), 3431. Doi: 10.3390/w15193431

[7]. Dsilva, L. R., et al. (2024). Wavelet scattering-and object detection-based computer vision for identifying dengue from peripheral blood microscopy. *International Journal of Imaging Systems and Technology*, *34*(1), e23020. Doi: 10.1002/ima.23020

[8]. Engelhardt, A., et al. (2023). On the Reliability of Automated Analysis of Fracture Surfaces Using a Novel Computer Vision-Based Tool. *Advanced Engineering Materials*, *25*(21), 2300876. Doi: 10.1002/adem.202300876

[9]. Hernández-Sampieri, R., Fernández-Collado, C., & Baptista-Lucio, M. del P. (2014). *Metodología de la investigación* (6th ed.). McGraw-Hill Interamericana.

[10]. IparraguirreVillanueva, O., et al. (2023). Improving Environmental Sustainability: A Geolocation-Based Mobile Application to Optimize the Recycling Process. *International Journal of Interactive Mobile Technologies*, *17*(23). Doi: 10.3991/ijim.v17i23.44417

[11]. Adamou, J. M. K., et al. (2023). Mineralogical, geochemical, and geotechnical features of lateritic soils from termite mounds in two contrasting savannah areas (central Cameroon) as raw materials for brick making. *Heliyon*, *9*(6). Doi: 10.1016/j.heliyon.2023.e17257

[12]. Kidari, R., & Tilioua, A. (2024). Investigation of the thermomechanical characteristics of compressed earth bricks reinforced with cement and corn straw waste fibers. *Cleaner Waste Systems*, *9*, 100160. Doi: 10.1016/j.clwas.2024.100160

[13]. Li, C., Zhou, J., & Dias, D. (2024). Utilizing semantic-level computer vision for fracture trace characterization of hard rock pillars in underground space. *Geoscience Frontiers*, *15*(2), 101769. Doi: 10.1016/j.gsf.2023.101769

[14]. Liu, L., & Du, K. (2024). A perspective on computer vision in biosensing. *Biomicrofluidics*, *18*(1). Doi: 10.1063/5.0185732

[15]. Loo, B. P., et al. (2023). Using computer vision and machine learning to identify bus safety risk factors. *Accident Analysis & Prevention*, *185*, 107017. Doi: 10.1016/j.aap.2023.107017

[16]. Loverdos, D., & Sarhosis, V. (2022). Automatic image-based brick segmentation and crack detection of masonry walls using machine learning. *Automation in Construction*, *140*, 104389. Doi: 10.1016/j.autcon.2022.104389

[17]. Martinez, D., et al. (2020). An agile-based integrated framework for mobile application development considering ilities. *IEEE Access*, *8*, 72461-72470. Doi: 10.1109/ACCESS.2020.2987882

[18]. Mishra, A., et al. (2023). Computer Vision Algorithm for the detection of fracture cracks in Oil Hardening Non-Shrinking (OHNS) die steel after machining process. *Fracture and Structural Integrity*, *17*(63), 234-245. Doi: 10.3221/IGF-ESIS.63.18

[19]. Mocciaro, A., et al. (2023). Mechanical and fracture behavior of insulating refractory bricks. *Cerâmica*, *69*(390), 99-106. Doi: 10.1590/0366-69132023693903407

[20]. Monje Álvarez, C. A. (2011). *Metodología de la investigación cuantitativa y cualitativa—Guía didáctica*. Universidad Surcolombiana.

[21]. Nations, U. (2023). *Equipo universitario crea concreto "verde" para la construcción sostenible*. United Nations.

[22]. Schettini, G. A. N., et al. (2024). Fabricación de ladrillo maleta, alternativa eco-artesanal para uso en construcciones agrícolas. *Revista Científica Arbitrada Multidisciplinaria PENTACIENCIAS*, *6*(4), 376-383. Doi: 10.59169/pentaciencias.v6i4.1162

[23]. Seth, Y., & Sivagami, M. (2025). Enhanced yolov8 object detection model for construction worker safety using image transformations. *IEEE Access*, *13*, 10582-10594. Doi: 10.1109/ACCESS.2025.3527511

[24]. Spiesman, B. J., et al. (2021). Assessing the potential for deep learning and computer vision to identify bumble bee species from images. *Scientific reports*, *11*(1), 7580. Doi: 10.1038/s41598-021-87210-1

[25]. Weichbroth, P. (2020). Usability of mobile applications: a systematic literature study. *IEEE Access*, *8*, 55563-55577. Doi: 10.1109/ACCESS.2020.2981892

[26]. Hualcas López, R. I. (2025). *1. Web Application Source Code – AWIFL*, Google Drive. Retrieved from: https://drive.google.com/drive/folders/1q5JtDasbFZ4D8r4_gC-jFJ4Mn0efrlCr?usp=sharing [accessed: 20 January 2026].