International Journal of Educational Development
ISSN Print: 3104-4263
ISSN Online: 3104-4271
CD ISSN: 3104-428X
https://doi.org/10.63313/IJED.9055

Teaching Reform through Automated Grading for Machine-Vision Defect Inspection Labs

Jing Li. Wanying Fu*
Department of Information and Intelligent Engineering, Shanghai Publishing and Printing College, Shanghai, China
Email: WanyingFu@sppc.edu.cn

How to cite this paper: Li, J., & Fu, W. Y. (2026). Teaching Reform through Automated Grading for Machine-Vision Defect Inspection Labs. *International Journal of Educational Development*, 3(1), 26-33. ISSN Print: 3104-4263; ISSN Online: 3104-4271.
https://doi.org/10.63313/IJED.9055
Published: 2026-04-10

Copyright © 2026 by author(s) and Erytis Publishing Limited.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
CC BY 4.0

Abstract

This paper proposes an engineering-oriented teaching reform for a vocational AI Machine Vision course, targeting industrial-style print defect inspection (stain and scratch) on a reproducible synthetic defect dataset. Guided by outcome-based education (OBE), we design a project-based task chain covering data specification, synthetic defect generation, preprocessing, classical machine vision baselines (thresholding, morphology, connected components, and line-structure detection), and engineering delivery of a command-line inspection tool with standardized outputs. To support formative assessment at scale and reduce subjective grading, we implement an automated assessment pipeline that verifies input/output (I/O) contract compliance, evaluates functional correctness via unit tests (pytest), measures code quality via static analysis (pylint/flake8), and validates robustness using a perturbation suite and boundary-case tests quantified by performance degradation ΔF1. Demonstration results using representative submissions show that the proposed pipeline provides consistent, transparent grading and promotes engineering deliverables and robust behavior, providing a reusable blueprint for machine vision education aligned with industrial inspection requirements.

Keywords

Teaching Reform; Machine Vision Course; Print Defect Inspection; Automated Assessment

I. Introduction

Vocational AI programs increasingly include machine vision courses with intensive coding labs. However, hands-on teaching often becomes demonstration-driven, where students focus on making it run rather than delivering production-grade inspection systems. In industrial printing lines, defect inspection must remain stable across batches, layouts, and illumination conditions, and must produce traceable reports for quality control. These requirements are rarely enforced in typical classroom grading, where manual checks are time-consuming, inconsistent, and slow to provide feedback.

To address these issues, we reform the Machine Vision course around an industrial print defect inspection task (stain and scratch) and design a closed-loop lab system that couples project-based learning with automated, objective assessment. Our approach formalizes teaching activities as engineering deliverables with explicit I/O contracts, standardized evaluation, runtime constraints, and robustness analysis grounded in real production conditions. This enables consistent formative assessment and iterative improvement. The reform is grounded in outcome-based education principles emphasizing measurable learning outcomes and evidence-based assessment supported by recent systematic review evidence [1], and adopts project-based learning to mirror authentic industrial workflows and deliverables as summarized in recent engineering education literature reviews [2].

Contributions:

☐ We design an OBE-aligned project task chain (64 contact hours) that mirrors industrial inspection workflows and produces explicit deliverables (code, configs and reports).

☐ We implement an automated assessment pipeline that jointly evaluates correctness, ML quality, engineering quality, and robustness, and generates structured diagnostic reports.

☐ We introduce robustness evaluation via a perturbation suite (brightness/contrast shifts, noise, blur, small rotations, and scaling) and boundary-case tests (tiny stains, low-contrast scratches, and edge-region defects), emphasizing stable generalization rather than overfitting to clean synthetic samples.

II.RELATED WORK

A. OBE and Project-Based Learning in Engineering Education

Outcome-based education (OBE) provides a measurable framework to align teaching activities with assessable engineering competencies. Recent studies report that OBE adoption improves outcome traceability while emphasizing the need for concrete, task-level evidence and auditable assessment artifacts [1]. Project-based learning (PBL) complements OBE by organizing learning around authentic tasks and iterative deliverables; recent reviews highlight that effective PBL requires clear rubrics and aligned assessment evidence [2].

B. Automated Assessment for Programming-Intensive Courses

Automated assessment in programming-intensive courses typically integrates execution-based tests for functional correctness with static analysis for code quality, enabling consistent scoring and rapid feedback at scale. Recent journal studies report that unit-test-based autograding pipelines can reduce grading workload while improving students ' iteration cycles through immediate, actionable feedback [3]. In parallel, work on AI-assisted grading and feedback highlights opportunities for richer formative feedback but stresses transparency,

reproducibility, and appropriate human oversight to mitigate risks such as bias or inconsistent scoring [4]. Motivated by these findings, our autograder combines I/O contract checks, pytest-based correctness scoring, and pylint/flake8-based maintainability signals, and further extends assessment to robustness via perturbation and boundary-case testing to reflect industrial expectations.

C. Robustness and Surface Defect Inspection Background

Industrial visual inspection systems must remain reliable under imaging variations, illumination shifts, blur, noise, and small geometric changes, and boundary cases dominated by tiny or low-contrast defects. Recent studies highlight the importance of robustness validation under real-world disturbances. These studies highlight robustness challenges under real-world disturbances, which motivate the adoption of perturbation-based stress testing in our evaluation protocol [5], [6]. A recent review summarizes both traditional feature-based methods and learning-based approaches for surface defect detection, providing context for the classical baselines used in educational labs and their typical failure modes [7].

D. Recent Evidence on OBE+PBL and Autograding Tools

Recent journal articles present production-ready autograding tools and automarker case studies that emphasize unit testing and timely formative feedback, supporting scalable evaluation workflows [8]. Broader reviews of AI-powered grading and tailored feedback further summarize benefits and challenges and reiterate the need for transparency and ethical safeguards in university assessment [9].

III.COURSE REFORM DESIGN

A. OBE-Aligned Task Chain

The course is organized as a project-based task chain aligned with four course learning outcomes (CLOs): CLO1 engineering specification and deliverables; CLO2 preprocessing and data handling; CLO3 defect detection algorithm implementation; and CLO4 robustness validation and quality awareness. Table I summarizes the 64-hour task chain, deliverables, and autograder checks.

F1(clean) is F1 on clean inputs and F1(perturbed) is F1 under perturbation. ΔF1 is defined as:

$$\Delta F1 = F1(clean) - F1(perturbed) \quad (1)$$

TABLE I. OBE-aligned task chain and deliverables for print defect inspection (stain/scratch), total 64 contact hours.

Module	Task	Hours	Key Deliverables (Evidence)	Autograder Checks	CLO
M1 Engineering & Data Spec	T1 Data spec and defect definition (stain/scratch)	4	Data description + load_data.py	I/O consistency, exception handling, field validation	CLO1
M1 Engineering	T2 Synthetic	4	defect_synth.py	Reproducibility,	CLO2

& Data Spec	defect generation		+ synthetic examples	parameter coverage, output validity	
M2 Preprocessing	T3 Illumination correction and contrast enhancement	6	preprocess_enhance.py	Unit tests, edge inputs, runtime limit	CLO2
M2 Preprocessing	T4 Denoising and smoothing comparison	4	preprocess_denoise.py	Output stability, parameter validity	CLO2
M3 Baseline Detection	T5 Thresholding baseline for stains	6	segment.py	Unit tests, output format, basic metrics	CLO3
M3 Baseline Detection	T6 Connected components + morphology post-processing	6	detect_cc.py	False-positive control cases, parameter robustness	CLO3
M4 Scratch Focus	T7 Edge/linear-structure detection for scratches	8	detect_scratch.py	Thin-line boundary cases, exception handling	CLO3
M5 Engineering Delivery	T8 Visualization overlay and report export	6	report.py + result.csv/json	Field completeness, sorting rules, traceability	CLO1
M6 Robustness	T9 Perturbation suite and boundary-case tests	8	robustness.py	ΔF1 threshold, degradation curve, hidden cases	CLO4
M7 Capstone	T10 End-to-end mini system delivery	12	main.py + README.md + output package	One-click run, reproducibility, engineering spec, robustness gate	CLO1/4

B. I/O Contract and Engineering Deliverables

To enforce inspection-grade outputs, we define a strict I/O contract. For each image, the inference tool outputs defect detections with fields: image_id, defect_type (stain/scratch), bounding box coordinates, and confidence. Optional metadata fields (batch/machine/illumination IDs) are included when available. Outputs must be exported to JSON/CSV and accompanied by visualization overlays. All submissions are required to include runnable scripts (train/eval/infer), configuration snapshots, trained weights, and a one-click reproduction README.

IV.AUTOMATED ASSESSMENT PIPELINE

A. Pipeline Overview

Figure 1 shows the autograder pipeline. A submission is executed in a controlled environment to: (1) verify runnability and I/O schema compliance, (2) evaluate functional correctness via unit tests, (3) assess code quality via static analysis, and (4) measure robustness via perturbation tests and boundary cases (ΔF1). The system outputs a structured score report with diagnostic traces to support

formative feedback and iterative improvement.

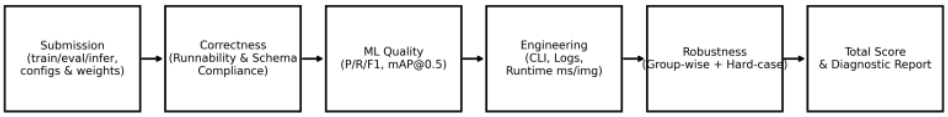


Figure 1. Automated assessment pipeline.

B. Scoring Model and Rubric

We compute: TotalScore = 0.60· Correctness + 0.20· Style + 0.20· Robustness.

TABLE II. Scoring rubric and measurement criteria.

Dimension	Weight	Measurement	Typical Failure Signals
Correctness	0.60	pytest pass rate (functionality, I/O contract, output files)	test failures, crashes, missing/invalid outputs
Style	0.20	pylint/flake8 score (naming, complexity, duplication, exceptions, docstrings)	many violations, high complexity, no docs
Robustness	0.20	ΔF1 under perturbations + boundary-case pass rate	large ΔF1, hard-case failures, false alarms surge

C. Algorithmic Procedure

Algorithm 1: AutoGrader for Print Defect Inspection Labs (Classical Baselines)

Input: submission S, config C (weights w, thresholds τ, perturbations Π, boundary cases B), test set D

1: ok_io← validate_io_contract(S)
2: if not ok_io then
3: return generate_report(io_error=True, TotalScore=0)
4: end if
5: T_pass ← run_unit_tests(S) # pytest
6: L_stat ← run_static_analysis(S) # pylint/flake8
7: P_clean ← evaluate_on_clean(D, S) # Precision/Recall/F1
8: P_pert ← evaluate_on_perturbations(D, S, Π)
9: ΔF1 ← compute_degradation(P_clean, P_pert)
10: score_correct ← f(T_pass)
11: score_style ← g(L_stat)
12: score_robust ← h(ΔF1, B)
13: TotalScore ←w1*score_correct + w2*score_style + w3*score_robust
14: R ←generate_report(T_pass, L_stat, ΔF1, failure_cases, TotalScore)
15: return R
Output: score report R

V. DATASET AND EXPERIMENTAL SETUP

A. Synthetic Print-Defect Dataset.

We construct a reproducible synthetic dataset and provide the full generation protocol, I/O specifications, and evaluation scripts to support replication.

B. Annotation and Leakage-Safe Split.

Since defects are synthesized with known parameters, bounding-box labels are generated automatically and can be optionally spot-checked by instructors. We split data by clean-image sources (e.g., layout IDs) to prevent leakage from near-duplicate backgrounds. The final dataset contains Train/Val/Test subsets with no background overlap, enabling a realistic assessment of generalization.

C. Baseline Inspection Methods and Runtime.

Students implement modular baselines for stain and scratch inspection, including thresholding (Otsu/adaptive), morphology and connected components for stain candidates, and edge/line-structure detection (Sobel/Canny + thin-line enhancement) for scratches. Runtime is measured on a fixed environment and reported as ms/image for the end-to-end pipeline.

D. Metrics.

The evaluation metrics are formally defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{4}$$

We report Precision, Recall, and F1 on clean inputs, along with false alarms per image and runtime (ms/image) for the end-to-end pipeline. Robustness is evaluated using a perturbation suite (brightness/contrast shifts, Gaussian noise, blur, small rotations, and scaling) and boundary cases (tiny stains, low-contrast scratches, and edge-region defects). Robustness is quantified by degradation Δ F1 and boundary-case pass rate.

VI. RESULTS

A. Overall Detection Performance

To keep the paper reproducible under limited hardware and without disclosing proprietary data, we report demonstration-level performance using the synthetic dataset and the classical baseline pipeline (thresholding/morphology/connected components for stains; edge/line-structure operators for scratches). We evaluate Precision, Recall, and F1 on clean inputs and report runtime (ms/image) for the end-to-end pipeline. These metrics are used primarily to validate that a submission is functional and meets the engineering I/O contract; the core discriminative evidence in this paper is provided by the autograder's correctness (unit tests), style (static analysis), and robustness (ΔF1 under perturbations and boundary cases) results.

B. Quantitative Autograder Results

All reported numbers in Table III–IV are illustrative results generated on a reproducible synthetic dataset and configuration. They are intended to demonstrate the grading pipeline's discriminative ability rather than to claim state-of-the-art defect detection performance. All scripts for synthetic defect generation, perturbation configuration, and evaluation are organized in a reusable package and can be shared as supplementary material to support independent replication.

The component scores and final grades are summarized in Table III. As expected, S1 achieves high scores across all dimensions, S2 is penalized mainly by static-analysis violations, and S3 is penalized mainly by robustness degradation (large ΔF1) and hard-case failures.

TABLE III. Autograder scores for representative submissions (S1–S3).

Submission	Correctness (0–100)	Style (0–100)	Robustness (0–100)	TotalScore (0–100)
S1	98	92	90	95.2
S2	96	55	88	86.2
S3	95	85	50	84.0

Note: TotalScore is computed using the weighting in Table II.

C. Robustness Under Perturbations and Boundary Cases

We evaluate robustness using a perturbation suite (brightness/contrast/noise/blur/rotation/scale) and a curated hard-case subset (tiny stains, low-contrast scratches, edge-region defects). Table IV summarizes F1 on clean inputs, F1 under perturbation, and ΔF1.

TABLE IV. Robustness under perturbations and boundary cases.

Condition	F1(clean)	F1(perturbed)	ΔF1	Key failures (optional)
Brightness/Contrast shift	0.92	0.88	0.04	minor missed tiny stains
Gaussian noise	0.92	0.86	0.06	confidence drop on low-contrast scratches
Gaussian blur	0.92	0.87	0.05	blurred thin scratches
Small rotation	0.92	0.89	0.03	slight localization shift
Scaling	0.92	0.88	0.04	small defects become harder
Hard-case subset	0.92	0.84	0.08	tiny stains + edge-region false alarms

VII. Conclusion

The proposed reform shifts the course focus from code demonstrations to inspection-grade deliverables and objective evaluation. By enforcing I/O contracts and automated assessment, the approach improves transparency and reduces instructor workload. Robustness is measured via real condition grouping rather than synthetic defect generation, reflecting industrial inspection requirements and cultivating quality awareness. The framework is modular and can be adapted to other inspection tasks by changing the defect taxonomy, schema, and evaluation sets.

Acknowledgment

This work was supported by Shanghai Publishing and Printing College, Higher Education Research Institute Project (Grant No. GJS2025-01) and Party-Building Research Center Project (Grant No. DJYJ2502).

References

[1] M. Mahrishi, S. Ramakrishna, S. Hosseini, et al., "A systematic literature review of the global trends of outcome-based education (OBE) in higher education with an SDG perspective related to engineering education," Discover Sustainability, vol. 6, art. no. 620, 2025.

[2] M. Ramírez de Dampierre, M. C. Gaya-López, and P. J. Lara-Bercial, "Evaluation of the implementation of project-based-learning in engineering programs: A review of the literature," Education Sciences, vol. 14, no. 10, art. no. 1107, 2024.

[3] B. Pereira Cipriano, N. Fachada, and P. Alves, "Drop Project: An automatic assessment tool for programming assignments," SoftwareX, vol. 19, art. no. 101079, 2022.

[4] Deepshikha, "A comprehensive review of AI-powered grading and tailored feedback in universities," Discover Artificial Intelligence, vol. 5, art. no. 251, 2025.

[5] Z. Ren, F. Fang, N. Yan, and Y. Wu, "State of the Art in Defect Detection Based on Machine Vision," International Journal of Precision Engineering and Manufacturing-Green Technology, vol. 9, pp. 661–691, 2022.

[6] Y. Chen, Y. Ding, F. Zhao, E. Zhang, Z. Wu, and L. Shao, "Surface Defect Detection Methods for Industrial Products: A Review," Applied Sciences, vol. 11, no. 16, art. no. 7657, 2021.

[7] P. Maruschak, I. Konovalenko, Y. Osadtsa, V. Medvid, O. Shovkun, D. Baran, H. Kozbur, and R. Mykhailyshyn, "Surface Illumination as a Factor Influencing the Efficacy of Defect Recognition on a Rolled Metal Surface Using a Deep Neural Network," Applied Sciences, vol. 14, no. 6, art. no. 2591, 2024.

[8] R. Bhagwat and A. Kulkarni, "Integrating project-based learning for enhancing higher education within an outcome-based education framework," International Journal of Evaluation and Research in Education, vol. 14, no. 5, pp. 4099–4108, Oct. 2025.

[9] A. L. C. Barczak, A. Mathrani, B. Han, and N. H. Reyes, "Automated assessment system for programming courses: a case study for teaching data structures and algorithms," Educational Technology Research and Development, 2023.