

Fail-Safe Project-Based Learning for Computer Vision: A Reproducible, Failure-Informed, and Accountability-Centred Design for Robotics Education

Didik Iskandar

Robotics and Artificial Intelligence Study Program, Faculty of Engineering,
Universitas Cendekia Abditama, Indonesia

Corresponding Author: Didik Iskandar, didik@uca.ac.id

ARTICLE INFO

Keywords: Computer Vision, Project-Based Learning, Reproducibility, Authentic Assessment, Robotics Education

Received : 25, June

Revised : 22, July

Accepted: 20, August

©2026 Iskandar: This is an open-access article distributed under the terms of the [Creative Commons Atribusi 4.0 Internasional](https://creativecommons.org/licenses/by-sa/4.0/).



ABSTRACT

Computer Vision courses often reward code that runs while under-assessing whether students can explain assumptions, reproduce experiments, diagnose failures, or manage risks when visual outputs drive robotic actions. This conceptual design study develops FAIL-SAFE PjBL, an eight-stage learning architecture: Frame the decision, Audit data, Isolate variables, Log failures, Stress-test conditions, Account for individual mastery, Formalize fail-safe behavior, and Explain reproducibly. The framework was produced through curriculum mapping, integrative literature synthesis, artifact prototyping, and an internal alignment audit. Its outputs include milestone-based projects, experiment logs, error portfolios, data and algorithm cards, latency/FPS profiling, safety gates, and individual viva. The design shifts assessment from polished demonstrations to traceable engineering judgment and provides a testable blueprint for future classroom action research.

INTRODUCTION

Computer Vision is no longer an isolated image-processing topic. In robotics, a visual pipeline converts photons into estimates, estimates into decisions, and decisions into physical actions. A seemingly minor segmentation error can therefore become a navigation error, an unsafe actuator response, or a missed inspection defect. This chain makes the educational problem different from teaching a software library. Students must learn not only how to produce an output image, but also how to justify assumptions, characterize uncertainty, evaluate timing constraints, document the experiment, and decide what a robot should do when visual evidence is weak.

Many course assignments nevertheless continue to privilege visible success. A notebook receives credit because it executes; a detector is considered successful because a bounding box appears; and a group presentation is accepted because the demonstration works under one carefully prepared condition. These practices produce an assessment blind spot. A polished demonstration may hide copied code, uncontrolled parameter tuning, undocumented data changes, fragile behavior under domain shift, or the fact that only one member of the group understands the pipeline. The result can be an attractive product without defensible engineering judgment.

The problem is intensified by the characteristics of visual systems. Performance depends on illumination, camera position, motion blur, occlusion, background complexity, sensor quality, compute capacity, and threshold selection. Accuracy alone is insufficient because a robotic system may also fail through latency, unstable frame rate, memory pressure, or an unsafe response to low confidence. Computer Vision education should consequently integrate algorithmic correctness with robustness, real-time profiling, privacy, bias awareness, and fail-safe behavior (Gonzalez & Woods, 2018; Szeliski, 2022; Tabassi, 2023).

Project-Based Learning (PjBL) is a promising response because it places students in extended investigations that culminate in a meaningful product. Meta-analytic evidence indicates that well-structured PjBL can improve learning outcomes, although effects depend on implementation quality, duration, discipline, and scaffolding (Zhang & Ma, 2023). The central issue is therefore not whether a project is assigned, but how the project is structured. When milestones, critique, revision, and reflection are absent, a project can become an end-of-semester production task rather than a learning architecture (Kokotsaki et al., 2016; Markula & Aksela, 2022).

A second gap concerns reproducibility. Machine learning research increasingly requires code, data descriptions, environment information, parameter settings, and explicit reporting of experimental decisions (Pineau et al., 2021). Dataset and model documentation practices were introduced to make limitations and intended uses visible (Gebru et al., 2021; Mitchell et al., 2019). Yet these practices are rarely treated as assessable learning outcomes in introductory Computer Vision courses. Students may submit a notebook that cannot be rerun because file paths are local, package versions are absent, cell execution is out of order, or the final result depends on undocumented manual intervention.

A third gap is the treatment of failure. In conventional assessment, failed images are deleted and only the best output is shown. Learning research, however, demonstrates that errors can become productive when students are required to explain them, receive timely feedback, and revise their representations (Kapur, 2008; Metcalfe, 2017). In Computer Vision, false positives, false negatives, unstable contours, missed detections, and timing bottlenecks are especially information-rich because they reveal the limits of data and assumptions. Hiding them removes the most diagnostic evidence from the learning process.

This paper responds to these gaps by proposing FAIL-SAFE Project-Based Learning, a failure-informed and reproducibility-centred architecture for a Computer Vision course in a Robotics and Artificial Intelligence program. Its novelty is not the use of PjBL alone, but the integration of eight auditable stages, a four-part evidence architecture, explicit error portfolios, real-time profiling, data and algorithm cards, individual viva, and safety gates. The study asks how a Computer Vision learning design can transform group projects from demonstrations of code execution into traceable evidence of engineering judgment, individual mastery, reproducibility, and responsible robotic deployment.

LITERATURE REVIEW

Project-Based Learning as an Evidence System

PjBL is often described through visible features such as teamwork, inquiry, and the production of an artifact. Those features are necessary but not sufficient. Strong PjBL also requires a challenging problem, sustained inquiry, authenticity, student voice, reflection, critique, revision, and a public product (Bell, 2010; Markula & Aksela, 2022). In engineering education, these elements become useful only when the artifact is connected to explicit performance criteria and when intermediate decisions remain visible. Otherwise, the final product can conceal weak conceptual understanding or uneven participation.

Constructive alignment provides the organizing logic for turning a project into an evidence system. Learning outcomes, learning activities, and assessment tasks should point toward the same forms of understanding (Biggs & Tang, 2011). Backward design strengthens this logic by defining acceptable evidence before selecting activities (Wiggins & McTighe, 2005). Applied to Computer Vision, the desired evidence includes more than a working detector. It includes a defensible baseline, controlled comparisons, error categories, runtime measurements, documentation, risk decisions, and the ability of each student to explain the pipeline.

Failure, Feedback, and Causal Reasoning

Failure can support learning when it creates a need for explanation rather than merely signalling a low score. Productive failure research shows that initial attempts at complex problems can prepare learners to recognize deep structures during later instruction (Kapur, 2008). Error-focused research similarly argues that mistakes become useful when learners attend to the discrepancy, receive corrective information, and revise their mental model (Metcalfe, 2017). This is

compatible with formative feedback, which is most powerful when it clarifies the goal, shows the current position, and indicates the next action (Hattie & Timperley, 2007).

Computer Vision offers an unusually rich setting for this approach. A failed detection can be traced to acquisition, preprocessing, representation, parameter choice, decision threshold, compute constraints, or an invalid assumption about the operating environment. Requiring students to classify the failure and test a targeted revision turns parameter tuning into causal inquiry. The educational value lies not in celebrating error for its own sake, but in converting error into a structured comparison between prediction, observation, explanation, and action.

Reproducibility and Documentation as Professional Competence

Reproducibility is sometimes treated as a research compliance issue that belongs after the algorithm has been developed. In practice, it is a core engineering capability. A visual system cannot be audited, maintained, or safely modified when the data origin, environment, parameters, and processing order are unknown. The NeurIPS reproducibility program demonstrated the value of checklists, code policies, and community replication efforts for improving the reporting culture of machine learning research (Pineau et al., 2021).

Documentation tools further clarify what must be made visible. Datasheets for datasets ask creators to document motivation, composition, collection, preprocessing, uses, distribution, and maintenance (Gebru et al., 2021). Model cards communicate intended use, evaluation conditions, limitations, and ethical considerations (Mitchell et al., 2019). In an undergraduate course, simplified data cards and algorithm cards can serve as learning scaffolds. They force students to state where images came from, which conditions were represented, what metrics were used, where the system fails, and what actions are unsafe.

Authentic Assessment, Individual Accountability, and Generative AI

Authentic assessment asks students to perform tasks that resemble the reasoning and communication expected beyond the classroom. It values judgment, integration, and transfer rather than isolated recall (Boud & Falchikov, 2007). Rubrics can make these expectations visible, but a group rubric alone does not prove individual competence. The instructor needs evidence about who designed, coded, tested, interpreted, and can defend each part of the system.

This issue has become more urgent because generative AI can produce plausible code and explanations. Blanket prohibition is difficult to enforce and may remove opportunities to teach responsible tool use. A stronger design requires disclosure, verification, testing, attribution where appropriate, and oral defence. Students may use AI assistance, but they remain accountable for every dependency, parameter, claim, and safety decision. Individual viva, code tracing, and random parameter probes are therefore not punitive additions; they are validity safeguards for assessment in an AI-rich environment (Cotton et al., 2024; Karnalim, 2024).

Trustworthy Robot Vision and Safety-by-Design

Responsible AI frameworks emphasize that trustworthy performance is contextual. Systems should be valid and reliable, safe, secure and resilient, accountable and transparent, explainable and interpretable, privacy-enhanced, and fair with harmful bias managed (Tabassi, 2023). These qualities are not achieved by adding an ethics paragraph at the end of a report. They require decisions throughout the system lifecycle, including data selection, metric choice, threshold setting, monitoring, and response to uncertainty.

Human-AI interaction research likewise recommends making system capabilities clear, supporting correction, and designing for situations in which the system is wrong (Amershi et al., 2019). For robotics education, this means students must specify how the robot behaves when confidence is low, latency exceeds a limit, the target disappears, or the camera input becomes invalid. The fail-safe response may be to stop, slow down, reacquire a frame, request human verification, or fall back to another sensor. The learning task must therefore connect visual uncertainty to an explicit physical action.



Figure 1. Conceptual architecture: from visible performance to defensible engineering evidence.

The synthesis yields four design propositions. First, a project should be judged through converging evidence rather than a final demonstration alone. Second, failure becomes educationally valuable when it is preserved, classified, and used to test a causal revision. Third, reproducibility is an assessable competence rather than a documentation preference. Fourth, responsible robot vision requires an explicit connection between visual uncertainty and physical fallback behavior.

METHODOLOGY

Research Design

This study used a conceptual design-and-development approach informed by educational design research. The goal was to create a theoretically grounded and implementable learning architecture rather than to estimate a causal

treatment effect. Educational design research is appropriate when a practical problem, theoretical principles, and iterative artifact development must be connected (McKenney & Reeves, 2019). The unit of design was a two-credit, sixth-semester Computer Vision course in the Robotics and Artificial Intelligence undergraduate program at Universitas Cendekia Abditama.

No student scores, interviews, or personal data were used in the design phase. Consequently, the results reported here concern the architecture, artifacts, and internal consistency of the proposed design. Claims about learning effectiveness are deliberately reserved for a future classroom action research study.

Design Corpus

The design corpus comprised the program learning outcomes, course learning outcomes, weekly topic sequence, assessment plan, instructional materials, laboratory worksheets, interactive media prototypes, and the technical requirements of a basic robot-vision pipeline. The literature corpus combined foundational work on constructive alignment, backward design, authentic assessment, feedback, and productive failure with recent work on PjBL, machine learning reproducibility, dataset and model documentation, trustworthy AI, and Computer Vision engineering.

Development Procedure

Development proceeded through six iterative activities. First, the course was reframed around decisions that a robot must make rather than around a list of OpenCV functions. Second, the intended outcomes were translated into observable evidence. Third, recurring failure modes in conventional assignments were mapped to design responses. Fourth, a milestone-based project sequence and assessment architecture were prototyped. Fifth, the design was audited against alignment, measurability, feasibility, cognitive accessibility, reproducibility, integrity, privacy, and safety. Sixth, the framework was refined until every intended outcome had at least two independent evidence sources and every project stage produced an auditable artifact.

The internal audit was analytic rather than psychometric. It did not claim expert validity or student effectiveness. The audit asked whether each outcome had an observable product, whether the product could be attributed to an individual or group, whether another person could rerun the work, whether failure conditions were tested, and whether the robotic response to uncertainty was specified.

Quality and Ethical Boundaries

Four safeguards were applied. First, the design avoids invented learning gains or statistical results. Second, the framework distinguishes design completeness from empirical effectiveness. Third, any future use of student artifacts for research must involve institutional review where required, informed communication, anonymization, secure storage, and non-penalization for students who decline research use. Fourth, generative AI assistance must be

declared and verified; responsibility for the manuscript and the instructional design remains with the author.

RESEARCH RESULTS

The design process produced three connected contributions: an eight-stage FAIL-SAFE workflow, a 4P evidence architecture, and a set of course artifacts that make technical performance, reasoning, individual mastery, and responsible deployment simultaneously visible.

A Diagnostic Matrix for Conventional Computer Vision Projects

The first output was a diagnostic matrix that distinguishes visible success from defensible learning. The matrix shows why common assignment practices are insufficient and what additional evidence is needed. A bounding box, for example, demonstrates that a pipeline produced an output in one situation; it does not show that the student can explain the threshold, reproduce the environment, identify failure conditions, or manage a false negative in a safety-relevant context.

Table 1. Diagnostic matrix linking common project practices to auditable evidence.

Visible Practice	Hidden Risk	FAIL-SAFE Design Response	Required Evidence
Working demo under one condition	Overfitting to a curated scene; hidden fragility	Stress tests across lighting, blur, occlusion, angle, background, and load	Robustness matrix and residual-limit statement
Copied or AI-generated code that executes	Low ownership and shallow explanation	Individual code tracing, random parameter probe, and viva	Person evidence linked to the submitted repository
Parameter tuning without a baseline	No defensible claim of improvement	Baseline plus one-variable experiments	Experiment table with prediction, change, evidence, and interpretation
Only best images are reported	Failure mechanisms remain invisible	Minimum error portfolio and failure taxonomy	False-positive, false-negative, degraded-condition, and runtime cases
Accuracy is the sole criterion	Unsafe or unusable real-time behavior	Function metrics plus latency, FPS, and resource profiling	Performance profile and operating envelope

Group score equals individual score	Free-riding or hidden concentration of expertise	Contribution log, peer evidence, viva, and live rerun	Moderated individual score
Ethics paragraph at the end	Privacy and safety decisions occur too late	Data, decision, and deployment gates	Data card, risk register, and fail-safe demonstration

The FAIL-SAFE Project Workflow

The eight-stage framework converts the semester project into a sequence of auditable decisions. Frame begins with the robotic decision, affected users, consequences of false positives and false negatives, and operating constraints. Audit documents data provenance, permissions, class balance, environmental coverage, and an initial risk register. Isolate establishes a baseline and requires one-variable experiments. Log preserves failures and links each failure to evidence, a causal hypothesis, and a revision. Stress-test evaluates adverse conditions rather than only nominal images. Account separates group production from individual mastery. Formalize defines thresholds, operating limits, and fallback behavior. Explain packages the work so that another person can rerun, critique, and responsibly reuse it.

The order is intentional. Students cannot meaningfully optimize a system before defining the decision and the cost of error. They cannot claim improvement without a baseline. They cannot diagnose improvement when multiple variables change simultaneously. They cannot claim robustness without stress tests, or competence without individual defence. Finally, they cannot claim responsible deployment without an explicit fallback for uncertainty.



Figure 2. The eight-stage FAIL-SAFE Project-Based Learning workflow.

Table 2. Operational definition of each FAIL-SAFE stage.

Stage	Core Learning Action	Primary Artifact
F - Frame	Define the robot decision, users, environment, costs of FP/FN, and constraints	Problem statement, decision map, initial risk statement

A - Audit	Document data provenance, permission, coverage, imbalance, and acquisition limits	Data card, sample audit, baseline dataset
I - Isolate	Build a baseline and alter one variable per experiment	Baseline notebook, parameter registry, comparison table
L - Log	Preserve failures and connect evidence to a causal hypothesis and revision	Error portfolio, revision log, annotated failure images
S - Stress-test	Challenge the pipeline under adverse visual and computational conditions	Robustness matrix, latency/FPS profile, operating envelope
A - Account	Verify individual understanding and contribution within group work	Contribution record, peer evidence, code tracing, viva
F - Formalize	Specify thresholds, limitations, uncertainty handling, fallback, and fail-safe behavior	Algorithm card, decision rules, fail-safe demo
E - Explain	Package the complete work for rerun, critique, and responsible reuse	Clean repository, run guide, final defence, limitation statement

The 4P Evidence Architecture

The framework uses four evidence domains. Performance evidence shows what the system does through metrics, demonstration, latency, and frame rate. Process evidence shows how the result was produced through notebooks, commits, data audits, experiment tables, and error logs. Person evidence shows who understands and can defend the work through viva, code tracing, and contribution records. Protection evidence shows why the system is responsible through permissions, limitation statements, bias checks, operating boundaries, and fail-safe demonstrations.

No single domain is sufficient. High performance without process evidence may be irreproducible. Process evidence without person evidence may represent uneven group participation. Person evidence without protection evidence may demonstrate technical skill but ignore harm. Protection evidence without measured performance may remain aspirational. The design therefore triangulates all four domains before a project is considered complete.

The 4P Evidence Architecture

A group product is credible only when four different forms of evidence converge

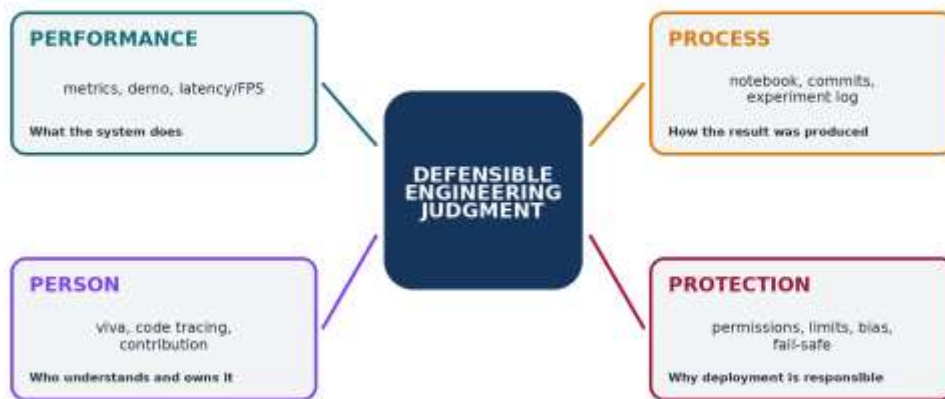


Figure 3. The 4P evidence architecture for triangulating project quality.

Constructive Alignment and Learning Outcomes

Six course learning outcomes were reorganized as a progression from representation to responsible communication. The sequence begins with image formation and digital representation, continues through preprocessing, segmentation, features, detection and tracking, and culminates in real-time evaluation, reproducibility, and safety. Each outcome is linked to more than one activity and more than one evidence source, reducing dependence on a single high-stakes task.

Table 3. Alignment between course outcomes, activities, and evidence.

Course Learning Outcome	Learning Activity	Evidence of Achievement
CLO 1: Explain image representation and acquisition	Prediction tasks, visual inspection, short calculations	Concept quiz; annotated image; oral explanation
CLO 2: Apply preprocessing and enhancement	One-variable laboratory experiments	Reproducible notebook; parameter registry; interpretation
CLO 3: Analyse segmentation, edges, contours, and features	Comparative experiments and error classification	Metric table; masks/contours; failure portfolio
CLO 4: Design detection or tracking for a robotic decision	Milestone project and design review	Pipeline diagram; baseline; repository; demo
CLO 5: Evaluate robustness and real-time feasibility	Stress tests, profiling, and safety review	Robustness matrix; latency/FPS; operating envelope; fail-safe

CLO 6: Communicate reproducibly and responsibly	Peer review, algorithm-card writing, and viva	Data card; algorithm card; run guide; individual defence
---	---	--

Milestone-Based Project Architecture

The project is distributed across the semester instead of being assigned as a final product. Each milestone ends with an artifact that can be reviewed before the next stage. This structure limits last-minute integration, creates repeated opportunities for feedback, and makes contribution patterns visible. The project context can vary - inspection, object sorting, line following, occupancy monitoring, or human-robot interaction - but the evidence architecture remains constant.

Table 4. Semester milestones and project evidence.

Milestone	Timing	Key Decision	Deliverable
1. Decision brief	Weeks 1-2	Target object/action, users, costs of error, constraints	Problem statement and risk statement
2. Data and baseline	Weeks 3-4	Data audit, permission, split, baseline pipeline	Data card and baseline notebook
3. Controlled experiments	Weeks 5-7	Preprocessing, segmentation, features, one-variable tests	Experiment registry and comparison table
4. Mid-project review	Week 8	Individual concept and code task	Individual evidence and feedback plan
5. Integration	Weeks 9-11	Detection/tracking, robot decision logic, version control	Integrated pipeline and contribution record
6. Robustness and profiling	Weeks 12-13	Stress tests, latency/FPS, failure analysis	Robustness matrix and error portfolio
7. Responsible deployment	Week 14	Safety gates, fallback, data/algorithm cards	Operating envelope and fail-safe demonstration
8. Demonstration and viva	Weeks 15-16	Public demo, rerun, individual defence, reflection	Repository, run guide, presentation, viva

The Reproducibility and Failure Portfolio

The reproducibility portfolio contains a clean notebook, environment specification, data card, parameter registry, run instructions, metric definitions, and a small verification set. A second person should be able to execute the notebook from the first cell to the final output without hidden files or manual edits. When random procedures are used, the seed and relevant nondeterministic operations are documented.

The failure portfolio is not a gallery of accidental mistakes. It is a structured record containing the input condition, expected behavior, observed behavior, type of error, evidence, suspected cause, proposed test, revision, and residual

limitation. At least one false positive, one false negative, one degraded-condition failure, and one timing or resource bottleneck must be analysed. This minimum prevents students from selecting only aesthetically convenient examples.

Assessment Weighting and Individual Accountability

Assessment combines low-stakes conceptual checks, laboratory evidence, a mid-semester individual task, the group project, an individual viva, and professionalism. The group project remains substantial because integration matters, but its score is moderated by individual evidence. A student who cannot explain the code, reproduce an experiment, or justify a parameter cannot automatically receive the full group score.

The viva uses short, randomly selected prompts: trace a data path; predict what happens if a threshold changes; explain a failure image; identify the most dangerous false negative; rerun a specified experiment; or state the fail-safe condition. These prompts are difficult to outsource because they require live interaction with the submitted artifact.

Table 5. Assessment composition and evidence focus.

Assessment Component	Weight	Primary Evidence
Concept quizzes and reflection	10%	Foundations, misconceptions, self-regulation
Laboratory worksheets / notebooks	20%	Controlled experiments, documentation, error evidence
Mid-semester individual assessment	15%	Concepts, calculations, code reading, interpretation
Group robot-vision project	30%	Integrated performance, robustness, profiling, documentation
Individual viva and live rerun	20%	Ownership, explanation, reproducibility, safety judgment
Professionalism and contribution	5%	Integrity, disclosure, peer feedback, safe laboratory practice

Ethics, Privacy, and Safety Gates

Three gates prevent ethics and safety from becoming decorative. The data gate checks source, permission, personal information, representation, and retention. The decision gate checks the consequences of error, threshold rationale, confidence handling, and human oversight. The deployment gate checks latency limits, sensor failure, emergency behavior, and the communication of limitations. A project cannot proceed to public demonstration if a critical gate remains unresolved.

Generative AI use is handled through a disclosure-and-defence rule. Students record the tool, purpose, prompt category, adopted output, modifications, and verification tests. The disclosure is not automatically penalized. Instead, the student must prove that the output was understood, tested, and integrated responsibly.

Table 6. Translation of trustworthy AI functions into course evidence.

AI RMF Function	Course-Level Question	Student Evidence
Map	What decision will the robot make, for whom, and in which environment?	Problem statement and risk register
Measure	How are accuracy, robustness, timing, and uncertainty evaluated?	Metric definitions, test set, stress-test matrix, profiling
Manage	What happens when confidence is low, data are invalid, or latency is excessive?	Fallback rules, operating limits, fail-safe demonstration
Govern	Who is accountable for data, code, decisions, and communication?	Contribution record, AI-use disclosure, data and algorithm cards

Sixteen-Meeting Implementation Blueprint

The framework can be implemented within a conventional sixteen-meeting semester. Early meetings build visual foundations and establish the baseline; middle meetings develop controlled experiments, segmentation, features, and tracking; later meetings emphasize robustness, profiling, documentation, safety, and defence. The distribution preserves instructional time because project artifacts replace separate, duplicative assignments.

DISCUSSION

Failure as Evidence, Not Defect

The most important shift is epistemic. Conventional projects treat failure as something to remove before presentation. FAIL-SAFE PjBL treats failure as evidence that reveals the boundary of a representation. This design is consistent with productive failure and error-based learning, but it adapts those theories to a domain in which errors have operational consequences. A false negative is not only a misconception to correct; it can also represent a collision, missed defect, or unsafe robot action. The framework therefore couples explanation with risk prioritization.

The one-variable rule is critical. Without it, students may change blur, threshold, kernel size, color range, and contour area simultaneously, then attribute improvement to whichever parameter appears most plausible. Isolating variables does not make all Computer Vision experimentation fully causal, but it establishes a disciplined minimum for comparison and supports more credible reasoning.

Reproducibility as a Learning Outcome

The second contribution is to move reproducibility from a report appendix into the intended curriculum. Pineau et al. (2021) describe reproducibility practices as part of research quality; the present design translates those practices into undergraduate evidence. The data card and algorithm card are not simplified paperwork. They externalize assumptions that would otherwise

remain hidden and make peer feedback more specific. Students learn that software quality includes the ability of another person to understand, rerun, and safely modify the system.

This approach also improves feedback. An instructor can comment on a baseline, an error category, or a missing dependency before the final demonstration. Feedback becomes attached to a decision and an artifact rather than to a vague impression of project quality.

Authenticity without Losing Assessment Validity

Group projects are authentic because engineering is collaborative, yet collaboration creates attribution problems. The 4P architecture addresses this tension by retaining a shared product while requiring individual evidence. The viva, contribution log, and code-tracing prompts do not fragment the project into isolated tasks; they verify that collaboration produced distributed understanding rather than concealed dependence on one member.

The same logic is relevant under generative AI. Detection software cannot reliably determine who authored every line of code, and authorship alone is not the most educational question. The more defensible question is whether the student can explain, test, critique, and take responsibility for the code. A disclosure-and-defence policy therefore aligns academic integrity with professional accountability.

Safety as a Design Constraint

A distinctive feature of the framework is that safety is connected to runtime evidence. Students must relate confidence, latency, and failure modes to a robotic response. This connection operationalizes trustworthy AI principles in a form appropriate for an undergraduate laboratory (Tabassi, 2023). It also reflects human-AI interaction guidance that systems should communicate limitations and support correction when wrong (Amershi et al., 2019).

The framework does not imply that a classroom prototype is safe for real-world deployment. It teaches students to make that boundary explicit. An algorithm card must state intended use, excluded use, evaluation conditions, residual risks, and the fallback behavior. Learning to say 'this system should not be deployed under these conditions' is treated as evidence of maturity rather than failure.

Transferability and Institutional Value

Although developed for Computer Vision, the architecture can transfer to machine learning, embedded systems, control, data science, and robotics courses. The domain-specific evidence will change, but the sequence of framing, auditing, isolating, logging, stress-testing, accounting, formalizing, and explaining remains applicable. At program level, the artifacts can also support curriculum review and quality assurance because they create a traceable relationship between learning outcomes, student work, and assessment decisions.

The design is feasible for resource-constrained institutions because it does not require expensive hardware for every stage. Early experiments can use offline datasets and low-cost cameras; profiling can be performed on available laptops;

and the safety gate can be demonstrated through simulated or low-energy robot responses. The essential requirement is disciplined evidence, not sophisticated equipment.

CONCLUSIONS AND RECOMMENDATIONS

FAIL-SAFE PjBL offers a practical answer to a recurring weakness in Computer Vision education: the tendency to equate a successful demonstration with learning. The framework organizes a project around eight auditable stages and evaluates it through four evidence domains - performance, process, person, and protection. It makes failure analysis, reproducibility, real-time profiling, individual defence, and fail-safe behavior central rather than optional.

For lecturers, the principal recommendation is to define evidence before assigning the project and to grade the reasoning trail, not only the final output. For students, the design establishes a professional standard: every result should be explainable, rerunnable, bounded by conditions, and linked to a responsible action. For programs, the framework provides a reusable alignment model for robotics and AI courses where software decisions can affect physical systems.

ADVANCED RESEARCH

The present study is limited to conceptual design and internal alignment. It does not establish learning gains, inter-rater reliability, workload effects, or student acceptance. The next study should implement two classroom action research cycles. Cycle I can focus on baseline construction, one-variable experiments, and reproducibility; Cycle II can add stress testing, safety gates, algorithm cards, and individual viva. Data should include concept tests, rubric dimensions, notebook reproducibility, error-log quality, observation, student reflection, and viva performance.

Further research should compare FAIL-SAFE PjBL with product-centred PjBL, examine whether error portfolios improve transfer to unseen conditions, test the reliability of the 4P rubric, and investigate how disclosure-and-defence policies influence responsible generative AI use. Multi-institution replication would be required before making broad claims about generalizability.

Table 7. Proposed empirical evaluation plan for future classroom implementation.

Evaluation Dimension	Suggested Success Evidence	Data Source
Implementation fidelity	At least 80% of planned milestones and feedback events are completed	Observation checklist and course records
Learning evidence quality	Improvement in rubric dimensions for causal explanation, error analysis, and reproducibility	Pre/post task, notebooks, portfolios, viva
Reproducibility	A peer can rerun the pipeline using the submitted instructions	Independent rerun checklist

Individual mastery	Students can explain and modify their assigned and adjacent pipeline components	Randomized viva prompts and live code trace
Safety reasoning	Students identify high-risk errors and demonstrate an appropriate fallback	Risk register, algorithm card, demo
Student experience	Workload, clarity, fairness, and perceived professional relevance are acceptable	Anonymous survey and focus group
Assessment reliability	Rubric dimensions show acceptable inter-rater agreement	Double scoring and agreement analysis

Ethics, Data, And Generative Ai Statement

This conceptual design study did not involve human participants, student records, or personal data. No empirical learning outcomes are claimed. Any future classroom study will follow applicable institutional procedures for informed communication, consent or lawful educational-research use, anonymization, secure storage, and non-penalization. Generative AI was used as a support tool for language refinement and document production; all arguments, references, design decisions, and final verification remain the author's responsibility.

ACKNOWLEDGMENT

The author thanks Universitas Cendekia Abditama, the Faculty of Engineering, the Robotics and Artificial Intelligence Study Program, PEKERTI facilitators, and colleagues who supported the development of the Computer Vision learning package. Appreciation is also extended to future student participants whose feedback will be essential for empirical refinement, subject to ethical and privacy safeguards.

REFERENCES

- Amershi, S., Weld, D., Vorvoreanu, M., Fournery, A., Nushi, B., Collisson, P., Suh, J., Iqbal, S., Bennett, P. N., Inkpen, K., Teevan, J., Kikin-Gil, R., & Horvitz, E. (2019). Guidelines for human-AI interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Paper 3, pp. 1-13). Association for Computing Machinery. <https://doi.org/10.1145/3290605.3300233>
- Anderson, L. W., & Krathwohl, D. R. (Eds.). (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman.
- Bell, S. (2010). Project-based learning for the 21st century: Skills for the future. *The Clearing House*, 83(2), 39-43. <https://doi.org/10.1080/00098650903505415>
- Biggs, J., & Tang, C. (2011). *Teaching for quality learning at university* (4th ed.). Open University Press.

- Boud, D., & Falchikov, N. (Eds.). (2007). Rethinking assessment in higher education: Learning for the longer term. Routledge.
- Bradski, G. (2000). The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 25(11), 120-125.
- Brookhart, S. M. (2013). How to create and use rubrics for formative assessment and grading. ASCD.
- Cotton, D. R. E., Cotton, P. A., & Shipway, J. R. (2024). Chatting and cheating? Ensuring academic integrity in the era of ChatGPT. *Innovations in Education and Teaching International*, 61(2), 228-239. <https://doi.org/10.1080/14703297.2023.2190148>
- De Los Rios-Carmenado, I., Lopez, F. R., & Garcia, C. P. (2015). Promoting professional project management skills in engineering higher education: Project-based learning strategy. *International Journal of Engineering Education*, 31(1B), 184-198.
- Fatmawati, F., Rochmawati, L., Sukma, M. M., Kusumayati, L. D., & Sukomardojo, T. (2024). Developing videos through project-based learning: Improving aviation cadets' basic airport civil engineering. *Formosa Journal of Multidisciplinary Research*, 3(10), 3915-3934. <https://doi.org/10.55927/fjmr.v3i10.11730>
- Geburu, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daume III, H., & Crawford, K. (2021). Datasheets for datasets. *Communications of the ACM*, 64(12), 86-92. <https://doi.org/10.1145/3458723>
- Gonzalez, R. C., & Woods, R. E. (2018). Digital image processing (4th ed.). Pearson.
- Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of Educational Research*, 77(1), 81-112. <https://doi.org/10.3102/003465430298487>
- Kapur, M. (2008). Productive failure. *Cognition and Instruction*, 26(3), 379-424. <https://doi.org/10.1080/07370000802212669>
- Karnalim, O. (2024). Detecting AI-assisted submissions in introductory programming education. *Education and Information Technologies*, 29, 20841-20865. <https://doi.org/10.1007/s10639-024-12520-6>
- Kemmis, S., McTaggart, R., & Nixon, R. (2014). The action research planner: Doing critical participatory action research. Springer. <https://doi.org/10.1007/978-981-4560-67-2>
- Kokotsaki, D., Menzies, V., & Wiggins, A. (2016). Project-based learning: A review of the literature. *Improving Schools*, 19(3), 267-277. <https://doi.org/10.1177/1365480216659733>
- Markula, A., & Aksela, M. (2022). The key characteristics of project-based learning: How teachers implement projects in K-12 science education. *Disciplinary and Interdisciplinary Science Education Research*, 4, Article 2. <https://doi.org/10.1186/s43031-021-00042-x>
- Mayer, R. E. (2021). Evidence-based principles for how to design effective instructional videos. *Journal of Applied Research in Memory and Cognition*, 10(2), 229-240. <https://doi.org/10.1016/j.jarmac.2021.03.007>
- McKenney, S., & Reeves, T. C. (2019). Conducting educational design research (2nd ed.). Routledge. <https://doi.org/10.4324/9781315105642>

- Metcalf, J. (2017). Learning from errors. *Annual Review of Psychology*, 68, 465-489. <https://doi.org/10.1146/annurev-psych-010416-044022>
- Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., & Gebru, T. (2019). Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (pp. 220-229). Association for Computing Machinery. <https://doi.org/10.1145/3287560.3287596>
- Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in Higher Education*, 31(2), 199-218. <https://doi.org/10.1080/03075070600572090>
- OpenCV. (2026). OpenCV-Python tutorials. https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html
- Panadero, E., & Jonsson, A. (2013). The use of scoring rubrics for formative assessment purposes revisited: A review. *Educational Research Review*, 9, 129-144. <https://doi.org/10.1016/j.edurev.2013.01.002>
- Pineau, J., Vincent-Lamarre, P., Sinha, K., Lariviere, V., Beygelzimer, A., d'Alche-Buc, F., Fox, E., & Larochelle, H. (2021). Improving reproducibility in machine learning research: A report from the NeurIPS 2019 reproducibility program. *Journal of Machine Learning Research*, 22(164), 1-20.
- Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. In *Advances in Neural Information Processing Systems* 28 (pp. 2503-2511). Curran Associates.
- Szeliski, R. (2022). *Computer vision: Algorithms and applications* (2nd ed.). Springer. <https://doi.org/10.1007/978-3-030-34372-9>
- Tabassi, E. (2023). Artificial intelligence risk management framework (AI RMF 1.0) (NIST AI 100-1). National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.AI.100-1>
- Wiggins, G., & McTighe, J. (2005). *Understanding by design* (2nd ed.). ASCD.
- Zhang, L., & Ma, Y. (2023). A study of the impact of project-based learning on student learning effects: A meta-analysis study. *Frontiers in Psychology*, 14, 1202728. <https://doi.org/10.3389/fpsyg.2023.1202728>