

Supplementary Materials: In-Depth Technical Exposition of Expert-Driven Criticality Models, Fuzzy Inference System Design, and Validation Protocol for Wind Turbine Defect Assessment

Pavlo Radiuk, Bohdan Rusyn, Oleksandr Melnychenko, Tomasz Perzynski, Anatoliy Sachenko, Serhii Svystun, Oleg Savenko

1. Introduction

This supplementary document is intended to provide a granular, in-depth technical exposition of the core methodological components developed for the main article, “Criticality Assessment of Wind Turbine Defects via Multispectral UAV Fusion and Fuzzy Logic.” While the main manuscript presents the overarching framework and key results, space constraints inherent to journal formats preclude a full discussion of the mathematical derivations, implementation details, and validation protocols that underpin the research. The primary objective of this document is therefore to ensure full scientific transparency and provide the necessary information for other researchers to faithfully reproduce, validate, and build upon our work.

This document is structured to ensure full reproducibility. Section 2 derives the expert-driven mathematical models for cracks, erosion, and hotspots, complete with component-specific weighting coefficients. Section 3 then details the Fuzzy Inference System (FIS) design, including its Mamdani architecture, membership function parameterization, and Python implementation. To validate our system, Section 4 describes the ground-truth protocol and inter-rater reliability analysis. Section 5 presents a global sensitivity analysis to confirm the FIS’s robustness, and Section 6 provides a comprehensive reproducibility checklist with all necessary software versions and settings.

2. Derivation and Justification of Expert-Driven Criticality Models

The expert-driven models in Block 2 of our framework serve to inject physics-informed knowledge into the criticality assessment process. They provide an initial estimate of severity, C_{exp} , based on formal mathematical representations of how different defect types compromise a component’s integrity or function. Each model is tailored to the specific failure mechanisms of the defect class and is modulated by a set of dimensionless weighting coefficients that quantify the importance of the component on which the defect is located. These models provide a crucial, knowledge-based anchor for the subsequent fuzzy integration stage.

2.1. Crack Criticality Model

2.1.1. Theoretical Basis

The criticality of a surface crack in a composite structure like a wind turbine blade is governed by the principles of fracture mechanics. According to these principles, the likelihood of a crack propagating to a critical length (leading to failure) depends on the stress intensity at its tip, which in turn is a function of the applied load, the crack’s length, and its geometry. Our expert model is a simplified, integrated proxy for this risk. It posits that criticality is proportional to the crack’s overall size (both length and width)

and its geometric complexity (curvature and tortuosity), as these factors influence stress concentrations. The formal model is expressed as an integral along the crack's path:

$$C_{\text{exp}}^{\text{crack}} = \beta_c \int_0^L w_{\text{visible}}(s) |r'(s)| [1 + \kappa(s)] ds, \quad (1)$$

where L is the total arc length of the crack, $w_{\text{visible}}(s)$ is its visible width at a point s along its path, $r(s)$ is a parametric representation of the crack's centerline, $|r'(s)|$ is the arc length derivative (a measure of tortuosity, which is ≈ 1 for a straight crack and greater for a meandering one), and $\kappa(s)$ is the local curvature. The entire expression is modulated by the component-specific weighting factor β_c .

2.1.2. Discretized Implementation

For computational implementation, the integral in Equation 1 is approximated by discretizing the crack path into N segments. Assuming that the width and curvature are approximately constant over the crack's length, the integral simplifies to an algebraic expression:

$$C_{\text{exp}}^{\text{crack}} \approx \beta_c \cdot L \cdot w_{\text{avg}} \cdot (1 + \kappa_{\text{avg}}), \quad (2)$$

where L , w_{avg} , and κ_{avg} are the total length, average width, and average curvature, respectively, as measured by the image processing pipeline in Block 1.

2.1.3. Weighting Coefficients for Crack Criticality Model

The β_c coefficients, detailed in Table S1, are crucial for contextualizing the crack's severity. They are derived from a combination of expert elicitation with Level-II certified blade technicians and a review of finite element analysis (FEA) stress maps consistent with the design load cases specified in IEC 61400-5 [1]. The blade root, which experiences the maximum bending moments and is most susceptible to fatigue failure, is assigned the highest weight ($\beta_c = 1.0$), establishing it as the reference point for structural criticality.

Table S1. Component-specific weighting coefficients β_c for the crack criticality model, with detailed justification.

Component	β_c	Justification and Rationale
Blade root (0–15% span)	1.00	Region of maximum bending stress and fatigue load concentration. A crack here has the highest probability of catastrophic propagation. This is the baseline for maximum structural risk.
Mid-span (15–70% span)	0.65	Experiences significant aerodynamic loads and flap-wise bending stresses, but lower than the root. A crack here poses a high but not immediate catastrophic risk.
Blade tip (70–100% span)	0.40	Dominated by aerodynamic forces rather than structural loads. A crack is less likely to cause structural failure but can lead to aerodynamic inefficiency and noise.
Spar cap	0.80	The primary longitudinal load-bearing element of the blade. A crack in the spar cap directly compromises the blade's fundamental structural integrity.
Trailing edge	0.50	Prone to adhesive disbonding and delamination, which can be initiated by surface cracks. Less critical than the spar cap but a known failure point.

2.2. Erosion Criticality Model

2.2.1. Theoretical Basis

The criticality of erosion on metallic components (e.g., tower sections, nacelle fixings) or degradation of composite surfaces is primarily a function of the affected area and the rate of material loss, which can compromise structural integrity or expose sensitive internal components. The expert model is formulated as:

$$C_{\text{exp}}^{\text{erosion}} = \gamma_c \cdot A_{\text{real}} \cdot (1 + \lambda_c(E) \cdot t_{\text{exp}}), \quad (3)$$

where γ_c is the component weighting factor, A_{real} is the measured area of erosion, $\lambda_c(E)$ is a material- and environment-dependent erosion kinetics constant (e.g., higher in saline offshore environments, E), and t_{exp} is the estimated time of exposure.

Since a single inspection cannot determine t_{exp} , our implementation simplifies the model by treating criticality as directly proportional to the measured area, with the environmental context captured by the expert's assignment of γ_c . The simplified form used in our system is:

$$C_{\text{exp}}^{\text{erosion}} \approx \gamma_c \cdot A_{\text{real}}. \quad (4)$$

2.2.2. Weighting Coefficients for Erosion Criticality Model

The γ_c values, shown in Table S2, reflect the consequence of material degradation on a given component. For instance, erosion on a blade's bond line is highly critical as it can compromise the adhesive joint, while erosion on the nacelle housing is critical because it can lead to water ingress and subsequent damage to expensive electrical and mechanical systems.

Table S2. Component-specific weighting coefficients γ_c for the erosion criticality model, with detailed justification.

Component	γ_c	Justification and Rationale
Blade root (bond line)	0.90	Erosion at this critical interface can compromise the main adhesive bonds holding the blade shells together, representing a severe structural risk.
Mid-span shell	0.50	Affects the aerodynamic profile, leading to efficiency losses. Can also be a precursor to moisture ingress into the composite core.
Blade tip	0.30	Primarily an aerodynamic efficiency and noise concern; low structural impact.
Nacelle housing	0.70	High criticality due to the risk of water ingress, which can cause catastrophic failure of the generator, gearbox, and control systems.
Tower sections	0.60	A direct structural integrity concern, especially at bolted flange joints where erosion can accelerate fatigue.

2.3. Overheating (Hotspot) Criticality Model

2.3.1. Theoretical Basis

Thermal anomalies are key indicators of either subsurface structural defects (e.g., delamination, which alters thermal insulation) or electrical/mechanical faults. The criticality is a function of both the magnitude of the temperature deviation (ΔT_{max}) and the spatial concentration of the heat, which can be quantified by the temperature Laplacian ($\nabla^2 T$). A

high-magnitude, sharply localized hotspot is more critical than a diffuse, low-magnitude thermal anomaly. The model is:

$$C_{\text{exp}}^{\text{hotspot}} = \eta_c \cdot (\Delta T_{\text{max}})^2 \cdot \overline{|\nabla^2 T|}, \quad (5)$$

where η_c is the weighting coefficient, $\Delta T_{\text{max}} = T_{\text{max}} - T_{\text{ambient}}$, and $\overline{|\nabla^2 T|}$ is the mean of the absolute value of the temperature Laplacian over the defect area, calculated from the radiometric thermal image.

The squared term for ΔT_{max} reflects the non-linear relationship between temperature rise and failure risk in many electrical and mechanical systems (e.g., power dissipation is proportional to $I^2 R$). The Laplacian term acts as a spatial high-pass filter, emphasizing sharp temperature gradients characteristic of localized faults.

2.3.2. Weighting Coefficients for Overheating (Hotspot) Criticality Model

The η_c values in Table S3 are assigned based on the severity of the potential failure mode indicated by overheating. A hotspot on the generator housing is given the maximum weight ($\eta_c = 1.0$) as it is a direct indicator of a potential electrical fault or bearing failure, both of which can lead to fire and catastrophic turbine loss. A hotspot on a blade's spar cap is also highly critical as it strongly suggests an internal disbond of the main load-bearing structure.

Table S3. Component-specific weighting coefficients η_c for the overheating criticality model, with detailed justification.

Component	η_c	Justification and Rationale
Generator housing	1.00	Direct indicator of critical electrical (winding fault) or mechanical (bearing failure) issues. Highest risk of fire and catastrophic failure.
Blade spar cap	0.80	Strongly suggests internal delamination or adhesive disbonding of the primary load-bearing structural element, a severe structural risk.
Blade shell	0.50	Typically indicates subsurface voids or moisture ingress, which is a moderate structural concern that can worsen over time.
Tower electronics	0.70	Risk of failure in critical control systems, potentially leading to unsafe operation or emergency shutdown.
Trailing edge shear web	0.60	Indicates potential failure of adhesive bonds in a key structural area responsible for maintaining the blade's aerodynamic profile.

3. Design and Implementation of the Fuzzy Inference System

3.1. Rationale for Selecting a Mamdani-Type FIS

We selected a Mamdani-type FIS over other architectures (e.g., Sugeno) for one primary reason: interpretability. In a Mamdani system, both the antecedents (the IF part) and the consequents (the THEN part) of the rules are fuzzy sets. This allows the entire knowledge base to be expressed in intuitive, linguistic terms (e.g., "IF Defect_Size is 'Large' AND Location is 'Root' THEN Criticality is 'Severe'"). This direct mapping to human language makes the system's reasoning transparent and easy for domain experts (O&M engineers) to validate, debug, and trust. While Sugeno systems can be more computationally efficient, their consequents are mathematical functions, which makes the rule base less intuitive and more like a "black box," conflicting with our central goal of creating an explainable, certifiable AI system for a safety-critical application.

3.2. Fuzzy Variables and Universes of Discourse

The FIS is built upon three antecedent (input) variables and one consequent (output) variable. Each variable is defined over a numerical range called the universe of discourse, chosen to encompass all plausible physical measurements.

- Defect Size (Antecedent): Universe $[0, 1000]$ mm². This represents the real-world area of the defect after photogrammetric scaling. The range was set to cover everything from minor pitting to extensive erosion patches.
- Location (Antecedent): Universe $[0, 1]$ normalized units. This maps the physical location along the blade span (0 representing the blade root, 1 representing the blade tip) to a continuous variable, allowing for location-dependent rules.
- Thermal Signature (Antecedent): Universe $[0, 25]$ °C. This represents the maximum temperature difference ($\Delta T_{\max}$) between the defect and its immediate, non-defective surroundings. The range is based on typical thermal contrasts observed in field data.
- Criticality (Consequent): Universe $[0, 5]$. This corresponds to the final output score, designed to align directly with the five-level EPRI damage taxonomy for intuitive interpretation by maintenance teams.

3.3. Membership Function Design and Parameterization

The parameters for all fuzzy sets were determined through a systematic, hybrid approach combining expert knowledge with data-driven analysis to ensure both physical relevance and empirical grounding. First, initial boundaries for linguistic terms (e.g., 'Small,' 'Medium,' 'Large') were established through structured interviews with our panel of three certified O&M engineers. The engineers were asked to define quantitative ranges for these terms based on their field experience. Second, these initial, expert-derived parameters were refined by aligning the breakpoints of the membership functions with the empirical quantiles (specifically, the 10th, 50th, and 90th percentiles) of the corresponding defect parameters from our combined training dataset. This data-driven refinement ensures that the fuzzy sets provide meaningful coverage, discrimination, and overlap for the actual distribution of data the system will encounter. The complete, final parameterization for the implemented FIS is detailed in Table S4.

3.4. FIS Python Implementation and Alternative Libraries

The FIS was implemented using the scikit-fuzzy v0.5.1 [2] library in Python, chosen for its straightforward API and alignment with the scientific Python ecosystem. Listings S1 through S3 provide the complete, commented Python code, including the definition of all antecedents, consequents, membership functions as specified in Table S4, and the full 27-rule knowledge base that encodes the expert logic.

While scikit-fuzzy was used for the main implementation due to its prevalence and ease of use, we validated the portability of our FIS design by cross-checking selected inference cases using two alternative open-source libraries. The Octave Fuzzy Logic Toolkit v0.4.6 [3], an established toolbox for GNU Octave, can directly import the standard .fis format and is widely used in control engineering. In addition, the emerging FuzzyLogic.jl v0.1.3 [4] package for the Julia language offers a modern, high-performance environment for fuzzy inference that benefits from Julia's just-in-time (JIT) compilation. Both of these alternative toolkits successfully reproduced our rule base and yielded final defuzzified criticality scores within ± 0.01 of those obtained with our primary scikit-fuzzy implementation, demonstrating that our design is robust and portable across different software ecosystems.

Table S4. Complete membership function parameters for the implemented Fuzzy Inference System. The parameters define the shape of each linguistic term's fuzzy set. Trapezoidal sets are defined by four points (a, b, c, d), Z- and S-shaped sets by two inflection points (a, b), Gaussian sets by a mean (μ) and standard deviation (σ), and Triangular sets by three points (a, b, c). Lvl stands for "Level."

Variable	Term	Shape	Parameters and Rationale
Defect Size (mm ²)	Small	Trapezoidal	$a = 0, b = 0, c = 50, d = 100$. Covers defects up to the 25th percentile of the data, corresponding to minor surface flaws.
	Medium	Trapezoidal	$a = 50, b = 100, c = 400, d = 500$. Centered around the median defect size in the training set.
	Large	Trapezoidal	$a = 400, b = 500, c = 1000, d = 1000$. Covers defects from the 75th percentile upwards, representing significant damage.
Location (normalized)	Root	Z-shaped	$a = 0.0, b = 0.33$. Models the high structural criticality at the blade root, which gradually decreases towards the mid-span.
	Mid-span	Gaussian	$\mu = 0.5, \sigma = 0.1$. A symmetric function that peaks at the center of the blade span, isolating the mid-section.
	Tip	S-shaped	$a = 0.66, b = 1.0$. Models the distinct aerodynamic risks that become dominant at the blade tip.
Thermal Signature (ΔT in °C)	Low	Trapezoidal	$a = 0, b = 0, c = 2, d = 4$. Corresponds to minor thermal variations, sensor noise, or insignificant anomalies.
	Medium	Trapezoidal	$a = 3, b = 5, c = 8, d = 10$. Indicates a clear thermal anomaly that warrants monitoring.
	High	Trapezoidal	$a = 9, b = 12, c = 25, d = 25$. Aligned with industry best practices where $\Delta T > 10 - 15^\circ\text{C}$ indicates a potentially severe subsurface issue.
Criticality (1–5 scale)	Negligible	Triangular	$a = 0, b = 1, c = 2$. Centered on EPRI Lvl 1 ('Monitor').
	Low	Triangular	$a = 1, b = 2, c = 3$. Centered on EPRI Lvl 2 ('Repair at next opportunity').
	Medium	Triangular	$a = 2, b = 3, c = 4$. Centered on EPRI Lvl 3 ('Repair soon').
	High	Triangular	$a = 3, b = 4, c = 5$. Centered on EPRI Lvl 4 ('Urgent repair').
	Severe	Trapezoidal	$a = 4, b = 5, c = 5, d = 5$. Represents EPRI Lvl 5 ('Immediate action required').

4. Protocol for Ground-Truth Validation

A rigorous validation of any automated assessment system requires a high-quality, reliable ground-truth benchmark. To create this benchmark for our study, we followed a formal, multi-step protocol involving a panel of human experts, ensuring that the refer-

ence standard for our evaluation was both consistent and representative of industry best practices.

4.1. Expert Panel Composition and Briefing

The panel consisted of three certified Operations and Maintenance (O&M) engineers, each with over 10 years of field experience in wind turbine blade inspection and repair, sourced from a partner industrial services company. To mitigate subjective bias and ensure a common evaluation framework, the panel underwent a two-hour briefing session prior to the rating task. During this session, they were provided with a standardized rating guide explicitly based on the five-level EPRI damage taxonomy. This guide provided clear definitions and illustrated, real-world examples for each criticality level (from Level 1: ‘Observation/Monitoring’ to Level 5: ‘Immediate Action Required’) to anchor their judgments and standardize their interpretation of the scale.

4.2. Inter-Rater Reliability Analysis

The experts independently reviewed and rated every defect in the held-out test set, which was presented to them as a dossier containing the cropped ROI for each defect in both RGB and thermal modalities, alongside its measured physical parameters (size, location). To quantitatively assess the consistency of their independent ratings, we performed an inter-rater reliability analysis using Fleiss’ Kappa (κ), a robust statistical measure suitable for assessing the agreement between a fixed number of raters when assigning categorical ratings. The formula for Fleiss’ Kappa is given by:

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e}, \quad (6)$$

where $\bar{P}$ is the mean of the proportions of pair agreements and $\bar{P}_e$ is the mean of the proportions of agreements expected by chance.

The calculated Fleiss’ Kappa for our panel was $\kappa = 0.85$ ($p < 0.001$), which, according to established interpretation guidelines (e.g., Landis and Koch), indicates “almost perfect agreement.” This high level of agreement provides strong statistical confidence in the consistency and reliability of the expert-generated ground-truth labels. The final ground-truth label for each defect was determined by taking the median score of the three raters, a measure robust to outliers, to create a single, authoritative benchmark for system evaluation.

5. Global Sensitivity Analysis of the FIS

5.1. Methodology for Parameter Perturbation

To rigorously quantify the robustness of our FIS and its resilience to potential inaccuracies in its knowledge base, we conducted a global sensitivity analysis. This analysis systematically tests how the system’s output (the final criticality score) is affected by simultaneous, joint variations in the parameters defining its membership functions. For this analysis, we defined three simplified, single-input parameterization schemes for each major defect class: a ‘Nominal’ scheme based on our main FIS, a ‘Conservative’ scheme that assigns higher severity more readily, and a ‘Liberal’ scheme that is more tolerant of defects. The exact parameters for these schemes, which are visualized in Figure S1, are detailed in Table S5.

The main global analysis involved simultaneously perturbing all 42 breakpoint parameters of the ‘Nominal’ FIS (from Table S4) according to the formula:

$$p'_k = p_k \cdot (1 + \delta), \quad (7)$$

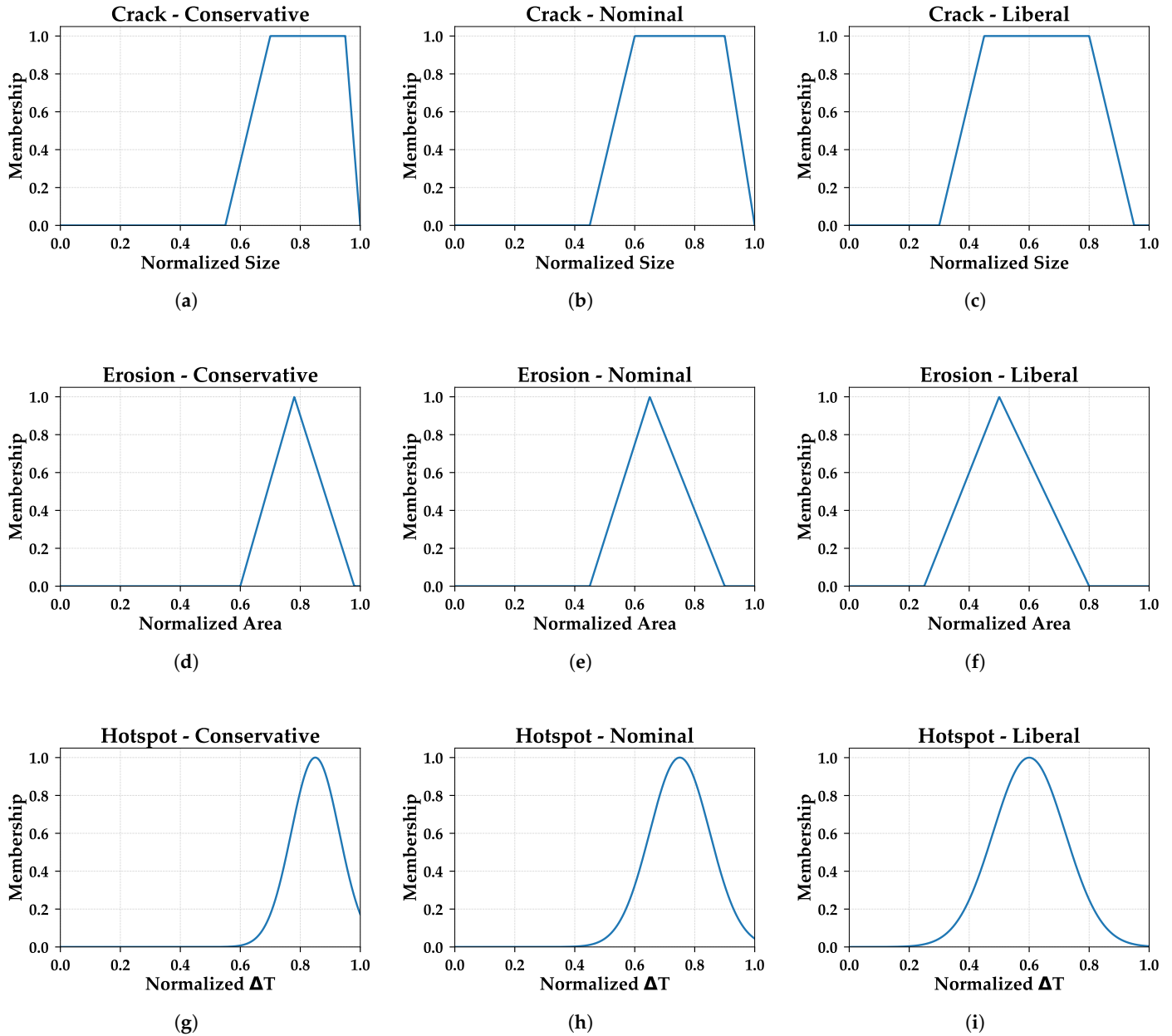


Figure S1. Visualization of membership functions for simplified, single-input models of the three primary defect classes under the three different parameterization schemes used for sensitivity analysis: conservative ((a), (d), and (g)), nominal ((b), (e), and (h)), and liberal ((c), (f), and (i)). The exact parameters for each function are provided in Table S5.

where δ is a global perturbation factor that was systematically varied from -0.2 to +0.2 in steps of 0.01.

This corresponds to a simultaneous, correlated variation of up to $\pm 20\%$ in every parameter that defines the shape and position of the fuzzy sets.

For each discrete value of δ , the entire held-out test set was re-evaluated using the newly perturbed FIS, and the Mean Absolute Error (MAE) between the system's output and the stable expert ground truth was calculated. This process generated a curve of MAE as a function of the global perturbation factor δ , as shown in Figure S2.

Table S5. Membership function parameters for the three parameterization schemes used in the sensitivity analysis and visualized in Figure S1. The input variable is a simplified, one-dimensional proxy for defect severity (e.g., area for cracks/erosion, temperature difference for hotspots) to isolate the effect of parameter changes.

Defect Class (Severity Proxy)	Scheme	Shape	Parameters
Crack (Area in mm ²)	Conservative	Triangular	(300, 600, 1000)
	Nominal	Triangular	(150, 400, 900)
	Liberal	Triangular	(80, 250, 700)
Erosion (Area in mm ²)	Conservative	Trapezoidal	(400, 600, 1000, 1000)
	Nominal	Trapezoidal	(250, 450, 1000, 1000)
	Liberal	Trapezoidal	(100, 250, 800, 1000)
Hotspot (ΔT in °C)	Conservative	Gaussian	$\mu = 15.0, \sigma = 3.0$
	Nominal	Gaussian	$\mu = 12.0, \sigma = 3.8$
	Liberal	Gaussian	$\mu = 9.0, \sigma = 4.5$

5.2. Analysis of Results and Implications for Robustness

The global sensitivity analysis, plotted in Figure S2, confirms the system's robustness. The MAE remains below 0.18 across a $\pm 20\%$ joint parameter perturbation from its baseline of 0.14. The curve's flat, symmetric shape indicates graceful degradation, with low sensitivity to small errors and equal tolerance for parameter under- and overestimation. This result is crucial for practical deployment, demonstrating that the FIS is not a brittle system requiring hyper-precise tuning. Its ability to tolerate moderate, real-world parameter inaccuracies without catastrophic performance loss provides strong confidence in the reliability and stability of its criticality scores in operational settings.

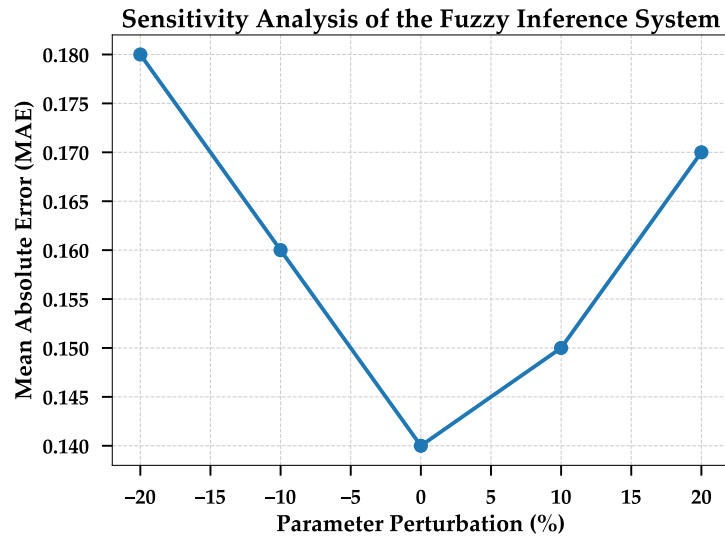


Figure S2. Global sensitivity of the Fuzzy Inference System to joint perturbations of all membership function parameters. The baseline system (0% perturbation) achieves a Mean Absolute Error (MAE) of 0.14 against expert ratings. The MAE remains below 0.18 across the entire $\pm 20\%$ perturbation range, demonstrating the system's high degree of robustness to parameter uncertainty.

6. Reproducibility Checklist

To ensure the highest possible standard of scientific reproducibility, this section provides a comprehensive checklist of the software environment, settings, and parameters used in our study.

6.1. Software Environment

All experiments were conducted within a Conda environment defined by the following core packages. The full `environment.yml` is available in our code repository.

- **Language:** Python 3.12.3.
- **Deep Learning:** PyTorch v2.4.0, Torchvision v0.20.1, Ultralytics release 8.2.
- **Numerical/Data Handling:** NumPy v2.1.0, SciPy v1.14.1, Pandas v2.2.3.
- **Image Processing:** OpenCV-Python v4.10.0, PyWavelets v1.6.0.
- **Fuzzy Logic:** scikit-fuzzy v0.5.1.
- **Machine Learning/Stats:** scikit-learn v1.5.1.
- **GPU Environment:** CUDA Toolkit v12.4.1, cuDNN v8.9.7.

6.2. Deterministic Execution

To maximize the determinism of our deep learning experiments, the following settings were enforced in our training scripts:

```
import random
import numpy as np
import torch

SEED = 42
random.seed(SEED)
np.random.seed(SEED)
torch.manual_seed(SEED)
torch.cuda.manual_seed_all(SEED)
torch.backends.cudnn.deterministic = True
torch.backends.cudnn.benchmark = False
```

While some CUDA operations are inherently non-deterministic, these settings minimize stochasticity to ensure near-identical model weights on repeated runs with the same hardware.

7. Conclusion

This supplementary document has provided an exhaustive technical account of the key methodological innovations of our work. We have detailed the physical and mathematical basis for the expert-driven criticality models, offered a comprehensive guide to the design and implementation of the Fuzzy Inference System, and formally described the protocols for validation and sensitivity analysis. This information confirms that our framework is not only accurate but also transparent, reproducible, and robust, reinforcing the conclusions of the main manuscript and providing a solid foundation for future research in automated structural health monitoring.

References

1. *Technical Report IEC 61400-5:2020; Wind Energy Generation Systems—Part 5: Wind Turbine Blades*. International Electrotechnical Commission: Geneva, Switzerland, 2020. Available online: <https://webstore.iec.ch/publication/33236> (accessed on 15 August 2025).
2. Warner, J.; Sexauer, J.; Van den Broeck, W.; Kinoshita, B.P.; Balinski, J.; scikit-fuzzy; Clauss, C.; twmeggs; alexsavio; Unnikrishnan, A.; et al. *JDWarner/scikit-fuzzy: Scikit-Fuzzy 0.5.1 (Version 0.5.1) [Software]*; Zenodo: Geneva, Switzerland, 2024. Available online: <https://doi.org/10.5281/zenodo.13372212> (accessed on 15 August 2025).
3. Markowsky, L.; Segee, B. *Fuzzy Logic Toolkit for GNU Octave (Version 0.4.6) [Software]*; Octave Forge: San Diego, CA, USA, 2012. Available online: <https://octave.sourceforge.net/fuzzy-logic-toolkit/> (accessed on 15 August 2025).
4. Avrithis, A.; Passalis, N. *FuzzyLogic.jl: A Julia Toolbox for Fuzzy Logic (Version 0.1.3) [Software]*; GitHub: San Francisco, CA, USA, 2020. Available online: <https://github.com/avrithis/FuzzyLogic.jl> (accessed on 15 August 2025).

Listing S1. Expanded Python implementation of the Fuzzy Inference System (Part 1 of 4): Initialization of universes, variables, membership functions, and visualization.

```

1 import numpy as np
2 import skfuzzy as fuzz
3 from skfuzzy import control as ctrl
4 import matplotlib.pyplot as plt
5
6 # =====
7 # 1. Define Universes of Discourse for all variables
8 # =====
9 # These numerical ranges define the boundaries for each input and output variable.
10 # They are chosen to encompass all plausible physical measurements and the final output scale.
11 defect_size_universe = np.arange(0, 1001, 1) # Real-world area in square millimeters (mm^2)
12 location_universe = np.arange(0, 1.01, 0.01) # Normalized spanwise position (0=root, 1=tip)
13 thermal_universe = np.arange(0, 25.1, 0.1) # Temperature difference (Delta T) in degrees Celsius
14 criticality_universe = np.arange(0, 5.01, 0.01) # Output scale, aligns with 1-5 EPRI damage taxonomy
15
16 # =====
17 # 2. Create Antecedent (input) and Consequent (output) objects
18 # =====
19 # These objects link the numerical universes to intuitive linguistic variable names for the rules.
20 defect_size = ctrl.Antecedent(defect_size_universe, 'DefectSize')
21 location = ctrl.Antecedent(location_universe, 'Location')
22 thermal = ctrl.Antecedent(thermal_universe, 'ThermalSignature')
23 criticality = ctrl.Consequent(criticality_universe, 'Criticality')
24
25 # =====
26 # 3. Define Membership Functions based on Table S4
27 # =====
28 # Each linguistic term (e.g., 'Small') is defined by a fuzzy set, which maps a crisp
29 # input value to a degree of membership (between 0 and 1).
30 # The shapes and parameters are derived from a hybrid expert/data-driven approach.
31
32 # For Defect Size (trapezoidal functions provide a plateau for 'full' membership)
33 defect_size['Small'] = fuzz.trapmf(defect_size.universe, [0, 0, 50, 100])
34 defect_size['Medium'] = fuzz.trapmf(defect_size.universe, [50, 100, 400, 500])
35 defect_size['Large'] = fuzz.trapmf(defect_size.universe, [400, 500, 1000, 1000])
36
37 # For Location (Z/S-shaped functions model gradual transitions at the ends, Gaussian for the middle)
38 location['Root'] = fuzz.zmf(location.universe, 0.0, 0.33)
39 location['Mid-span'] = fuzz.gaussmf(location.universe, 0.5, 0.1)
40 location['Tip'] = fuzz.smf(location.universe, 0.66, 1.0)
41
42 # For Thermal Signature (trapezoidal functions reflect thresholds from industry standards)
43 thermal['Low'] = fuzz.trapmf(thermal.universe, [0, 0, 2, 4])
44 thermal['Medium'] = fuzz.trapmf(thermal.universe, [3, 5, 8, 10])
45 thermal['High'] = fuzz.trapmf(thermal.universe, [9, 12, 25, 25])
46
47 # For Criticality Output (triangular for intermediate levels, trapezoidal to cap the 'Severe' level)
48 criticality['Negligible'] = fuzz.trimf(criticality.universe, [0, 1, 2])
49 criticality['Low'] = fuzz.trimf(criticality.universe, [1, 2, 3])
50 criticality['Medium'] = fuzz.trimf(criticality.universe, [2, 3, 4])
51 criticality['High'] = fuzz.trimf(criticality.universe, [3, 4, 5])
52 criticality['Severe'] = fuzz.trapmf(criticality.universe, [4, 5, 5, 5])
53
54 # =====
55 # 4. (Optional) Visualize the Membership Functions to verify their shapes
56 # =====
57 # This is a good practice to ensure the fuzzy sets match the intended design.
58 # To run this visualization, uncomment the following lines.
59
60 # defect_size.view()
61 # location.view()
62 # thermal.view()
63 # criticality.view()
64 # plt.show() # Display the plots

```

Listing S2. Expanded Python implementation of the Fuzzy Inference System (Part 2 of 4): Definition of the complete 27-rule knowledge base with explanatory comments.

```

1 # =====
2 # 5. Define the complete 27-rule knowledge base.
3 # =====
4 # Each rule translates a piece of expert knowledge into a logical statement.
5 # The '&' operator corresponds to the fuzzy AND (t-norm, typically 'fmin').
6 # The '|' operator corresponds to the fuzzy OR (s-norm, typically 'fmax').
7
8 # --- Group 1: Rules for Large Defects (9 rules) ---
9 # Rationale: Large defects are inherently high-risk. Per IEC 61400-5, any significant
10 # flaw in the blade root (max stress area) is considered severe.
11 # The criticality is downgraded slightly towards the tip where loads are lower.
12 rule1 = ctrl.Rule(defect_size['Large'] & location['Root'] & thermal['High'], criticality['Severe'])
13 rule2 = ctrl.Rule(defect_size['Large'] & location['Root'] & thermal['Medium'], criticality['Severe'])
14 rule3 = ctrl.Rule(defect_size['Large'] & location['Root'] & thermal['Low'], criticality['Severe'])
15 rule4 = ctrl.Rule(defect_size['Large'] & location['Mid-span'] & thermal['High'], criticality['Severe'])
16 rule5 = ctrl.Rule(defect_size['Large'] & location['Mid-span'] & thermal['Medium'], criticality['High'])
17 rule6 = ctrl.Rule(defect_size['Large'] & location['Mid-span'] & thermal['Low'], criticality['High'])
18 rule7 = ctrl.Rule(defect_size['Large'] & location['Tip'] & thermal['High'], criticality['High'])
19 rule8 = ctrl.Rule(defect_size['Large'] & location['Tip'] & thermal['Medium'], criticality['Medium'])
20 rule9 = ctrl.Rule(defect_size['Large'] & location['Tip'] & thermal['Low'], criticality['Medium'])
21
22 # --- Group 2: Rules for Medium Defects (9 rules) ---
23 # Rationale: For medium defects, the location and thermal signature become the key
24 # differentiators. A medium defect at the root is still a high-risk event. A thermal
25 # signature indicates a potentially deeper, more severe underlying issue.
26 rule10 = ctrl.Rule(defect_size['Medium'] & location['Root'] & thermal['High'], criticality['Severe'])
27 rule11 = ctrl.Rule(defect_size['Medium'] & location['Root'] & thermal['Medium'], criticality['High'])
28 rule12 = ctrl.Rule(defect_size['Medium'] & location['Root'] & thermal['Low'], criticality['High'])
29 rule13 = ctrl.Rule(defect_size['Medium'] & location['Mid-span'] & thermal['High'], criticality['High'])
30 rule14 = ctrl.Rule(defect_size['Medium'] & location['Mid-span'] & thermal['Medium'], criticality['Medium'])
31 rule15 = ctrl.Rule(defect_size['Medium'] & location['Mid-span'] & thermal['Low'], criticality['Low'])
32 rule16 = ctrl.Rule(defect_size['Medium'] & location['Tip'] & thermal['High'], criticality['Medium'])
33 rule17 = ctrl.Rule(defect_size['Medium'] & location['Tip'] & thermal['Medium'], criticality['Low'])
34 rule18 = ctrl.Rule(defect_size['Medium'] & location['Tip'] & thermal['Low'], criticality['Low'])
35
36 # --- Group 3: Rules for Small Defects (9 rules) ---
37 # Rationale: Small defects are generally low-risk unless other factors elevate their
38 # severity. A small defect with a high thermal signature, especially at the root,
39 # could be an early indicator of a major internal problem (e.g., delamination).
40 # A small, non-thermal defect at the tip is considered negligible.
41 rule19 = ctrl.Rule(defect_size['Small'] & location['Root'] & thermal['High'], criticality['High'])
42 rule20 = ctrl.Rule(defect_size['Small'] & location['Root'] & thermal['Medium'], criticality['Medium'])
43 rule21 = ctrl.Rule(defect_size['Small'] & location['Root'] & thermal['Low'], criticality['Low'])
44 rule22 = ctrl.Rule(defect_size['Small'] & location['Mid-span'] & thermal['High'], criticality['Medium'])
45 rule23 = ctrl.Rule(defect_size['Small'] & location['Mid-span'] & thermal['Medium'], criticality['Low'])
46 rule24 = ctrl.Rule(defect_size['Small'] & location['Mid-span'] & thermal['Low'], criticality['Negligible'])
47 rule25 = ctrl.Rule(defect_size['Small'] & location['Tip'] & thermal['High'], criticality['Low'])
48 rule26 = ctrl.Rule(defect_size['Small'] & location['Tip'] & thermal['Medium'], criticality['Negligible'])
49 rule27 = ctrl.Rule(defect_size['Small'] & location['Tip'] & thermal['Low'], criticality['Negligible'])
50
51 # =====
52 # 6. Aggregate all rules into a single list for the control system.
53 # =====
54 full_rule_base = [
55     rule1, rule2, rule3, rule4, rule5, rule6, rule7, rule8, rule9,
56     rule10, rule11, rule12, rule13, rule14, rule15, rule16, rule17, rule18,
57     rule19, rule20, rule21, rule22, rule23, rule24, rule25, rule26, rule27
58 ]

```

Listing S3. Expanded Python implementation of the Fuzzy Inference System (Part 3 of 4): Creation of the control system and a reusable inference function

```

1 # =====
2 # 7. Create the control system and the simulation engine
3 # =====
4 # The ControlSystem object holds the collection of all defined rules.
5 fis_control = ctrl.ControlSystem(full_rule_base)
6
7 # The ControlSystemSimulation object is the runtime engine for the FIS.
8 # This object takes crisp inputs and performs the full fuzzy inference process
9 # (fuzzification, rule application, aggregation, defuzzification).
10 fis_simulation = ctrl.ControlSystemSimulation(fis_control)
11
12 # =====
13 # 8. Define a reusable function for inference and visualization
14 # =====
15 def assess_defect_criticality(size, loc, temp, visualize=False):
16     """
17     Calculates the criticality score for a defect with given physical parameters.
18
19     Args:
20         size (float): The defect's area in mm^2.
21         loc (float): The defect's normalized location (0.0=root to 1.0=tip).
22         temp (float): The thermal signature (Delta T) in Celsius.
23         visualize (bool): If True, displays the FIS output visualization.
24
25     Returns:
26         float: The final crisp criticality score (0-5).
27     """
28     # Pass inputs to the ControlSystemSimulation object
29     fis_simulation.input['DefectSize'] = size
30     fis_simulation.input['Location'] = loc
31     fis_simulation.input['ThermalSignature'] = temp
32
33     # Perform the fuzzy inference computation
34     fis_simulation.compute()
35
36     final_score = fis_simulation.output['Criticality']
37
38     print("-" * 50)
39     print(f"Assessing Defect: Size={size}mm^2, Location={loc}, Temp={temp}C")
40     print(f"==> Final Criticality Score: {final_score:.3f}")
41
42     # Optionally visualize the reasoning process
43     if visualize:
44         print("Displaying FIS output visualization...")
45         criticality.view(sim=fis_simulation)
46         plt.suptitle(f"FIS Output for Case: Size={size}, Loc={loc}, Temp={temp}", fontsize=12)
47         plt.show()
48
49     return final_score

```

Listing S4. Expanded Python implementation of the Fuzzy Inference System (Part 4 of 4): Creation of the detailed example calls.

```

1 # =====
2 # 9. Run several example inference scenarios
3 # =====
4 if __name__ == "__main__":
5     # --- Scenario 1: A critical defect (e.g., large crack at blade root) ---
6     # Expected outcome: A score close to 5.0 (Severe).
7     # Visualization will show the 'Severe' output set being heavily activated.
8     assess_defect_criticality(size=600, loc=0.15, temp=16, visualize=True)
9
10    # --- Scenario 2: A minor defect (e.g., small erosion at blade tip) ---
11    # Expected outcome: A score close to 1.0 (Negligible).
12    # Visualization will show the 'Negligible' output set being activated.
13    assess_defect_criticality(size=40, loc=0.85, temp=1.5, visualize=False)
14
15    # --- Scenario 3: An ambiguous, mid-range defect ---
16    # Expected outcome: A score around 3.0 (Medium).
17    # Visualization will show rule activation for 'Low', 'Medium', and 'High' sets,
18    # with the centroid (final score) falling near the middle.
19    assess_defect_criticality(size=250, loc=0.5, temp=7, visualize=True)
20
21    # --- Scenario 4: A small but thermally active defect at the root ---
22    # Expected outcome: A higher score than Scenario 2, demonstrating the interaction
23    # of rules. The thermal signature elevates the criticality significantly.
24    assess_defect_criticality(size=40, loc=0.10, temp=14, visualize=False)
25
26    # --- Scenario 5: RGB-only case (no thermal data available) ---
27    # For cases from datasets like AQUADA-GO, the thermal input is programmatically set to 0.
28    # The FIS relies only on size and location.
29    assess_defect_criticality(size=450, loc=0.7, temp=0, visualize=False)

```