Failure Identification in Imitation Learning via Statistical and Semantic Filtering

Quentin Rolland[1,2], Fabrice Mayran de Chamisso[1], Jean-Baptiste Mouret[2,3]

Abstract— **Imitation learning (IL) policies in robotics deliver strong performance in controlled settings but remain brittle in real-world deployments: rare events such as hardware faults, defective parts, unexpected human actions, or any state that lies outside the training distribution can lead to failed executions. Vision-based Anomaly Detection (AD) methods emerged as an appropriate solution to detect these anomalous failure states but do not distinguish failures from benign deviations. We introduce FIDeL (Failure Identification in Demonstration Learning), a policy-independent failure detection module. Leveraging recent AD methods, FIDeL builds a compact representation of nominal demonstrations and aligns incoming observations via optimal transport matching to produce anomaly scores and heatmaps. Spatio-temporal thresholds are derived with an extension of conformal prediction, and a Vision–Language Model (VLM) performs semantic filtering to discriminate benign anomalies from genuine failures. We also introduce BotFails, a multimodal dataset of real-world tasks for failure detection in robotics. FIDeL consistently outperforms state-of-the-art baselines, yielding +5.30% AUROC in anomaly detection and +17.38% failure-detection accuracy on BotFails compared to existing methods. Videos of FIDeL can be found on our website : https://cea-list.github.io/FIDeL/**

I. INTRODUCTION

Recent progresses in imitation learning (IL) [1]–[3] have the potential to significantly enhance the flexibility and adaptability of robotic systems. Unfortunately, current learned policies cannot be deployed in real world environments because their behavior is not defined when they are in a situation that is not part of the training set. Such events are unavoidable in factories or human-centered environments, arising from defective objects, operator mistakes, or environmental shifts, and they cannot be exhaustively anticipated in training data.

A prevalent strategy is to assume that the training dataset is sufficiently diverse to encompass all possible scenarios. Implicitly, this amounts to presuming that no situation is off distribution. While this assumption simplifies the problem, it is unrealistic in practice, many rare events will never appear in training data. The second approach is to design models that can quantify the uncertainty of their decision, for instance with Bayesian models [4], [5]. However, uncertainty quantification remains an open challenge in IL and RL [6], [7], and is not integrated into the highest-performing learning algorithms [1]–[3]. A more practical direction is to rely on an independent monitoring module that evaluates execution at runtime [8]–[12].

In this paper, we build on recent work [8]–[12] and propose an independent failure monitor, designed to detect situations in which a policy should be interrupted. We frame the problem in terms of Anomaly Detection (AD). Anomalies correspond to deviations from the expected distribution, which may—but do not always—indicate failure states. Our approach leverages the One-Class (OC) paradigm, a well-established method in computer vision [13]–[17]. In OC learning, a model is trained solely on data from a "normal" class, and any significant deviation from this baseline is flagged as anomalous. This formulation is particularly well-suited to IL, where the space of possible deviations is unbounded, yet expert demonstrations naturally provide a reliable definition of normality. Thus, we treat expert demonstrations as our baseline and regard any deviation as an anomaly—yielding an implicit and comprehensive definition of undesirable behavior.

That said, the notion of anomaly encompasses a broader range of situations than failure. Not all anomalies correspond to failure-related events: some may be entirely benign—for instance, a fly landing on a table or an object slightly shifting in the background—and should not trigger unnecessary or costly interventions. This mismatch reveals a key limitation of existing AD methods: while they can reliably highlight deviations from nominal behavior, they lack the ability to discriminate between harmless anomalies and those that truly compromise task execution. In robotics, this distinction is essential. What is required is a mechanism that can semantically interpret anomalies and explicitly determine whether they represent benign variations or genuine failures.

To bridge the gap between traditional Vision Anomaly Detection and the unique demands of robotic applications, we introduce **FIDeL** (**F**ailure **I**dentification in **De**monstration **L**earning), a failure detection module designed to complement IL policies in robotics. **FIDeL** performs post-hoc **anomaly detection** by comparing new observations with stored representations of normal demonstrations using Optimal Transport (OT). To decide whether or not an AD score indicates an anomaly, we leverage a **Conformal Prediction** (CP)–based thresholding [18], [19] mechanism, which accounts for both temporal variations in anomaly scores and spatial variations across visual patches. When the anomaly score exceeds the CP threshold, **FIDeL** triggers a **semantic filter** based on a vision-language model (VLM), which determines whether the detected anomaly is failure-related or benign. This three-stage process allows for efficient failure detection while reducing false positives, thus preserving productivity.

[1] Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France [2] Inria, CNRS, Université de Lorraine, LORIA, F-54000 Nancy, France [3] Bleu Robotics, Paris, France correspondance to: quentin.rolland2@cea.fr.

Our contributions are summarized as follows:

- Following the One-Class paradigm, we design a **novel Vision Anomaly Detection, representation-based algorithm** tailored for autonomous robotics.
- We extend Conformal Prediction to handle both **temporal and spatial variations**, with memory-based temporal alignment that ensures robustness to variable execution speeds.
- We propose a **semantic filtering mechanism** using a VLM and anomaly heatmaps, which discriminates between benign anomalies and task-critical failures.
- We enhance interpretability through **localized heatmaps and VLM-based explanations**, providing human-readable justifications for failure detection.
- We present **BotFails** a **dedicated training and evaluation dataset** collected on LeRobot [20] for benchmarking anomaly and failure detection in robotics, addressing the scarcity of failure-oriented datasets.
- We demonstrate significant performance gains over baselines in both anomaly detection ($+5.30\%$ AUROC) and failure detection tasks ($+17,38\%$ accuracy).

II. Related Work

A. Vision Anomaly Detection

Vision AD is a well-established problem in computer vision, particularly within the unsupervised learning paradigm. Existing methods can be broadly categorized into four families [13]: **(1) Augmentation-based methods** add artificial anomalies to normal samples and train a classifier to recognize them [21]–[23]. However, we found no straightforward way to adapt these approaches to robotics, primarily due to the challenge of generating realistic anomalies in robot and object motions. **(2) Reconstruction-based methods** learn to reconstruct normal inputs, using the reconstruction error as an anomaly score. Common architectures include Autoencoders (AE), Variational Autoencoders (VAE) or Generative Adversarial Networks (GANs) [24]–[26], and Student-Teacher frameworks [27]–[29]. However, such models often generalize well enough to reconstruct anomalous inputs, thus failing to highlight anomalous regions effectively [30], [31]. **(3) Normalizing Flows** (NF) [32] based methods [33]–[35] learn an invertible transformation from the data distribution to a well-defined prior, typically a Gaussian. Anomalies are detected when the likelihood of a test sample under the learned distribution deviates significantly from the normal class. Despite their theoretical appeal, NF exhibit "likelihood paradoxes," often assigning higher likelihood to anomalous images than normal ones [36]. **(4) Representation-based methods** [8], [37]–[39] use pretrained, frozen networks to extract semantic embeddings and compute anomaly scores based on their distance to normal class features. Early methods like PaDiM [37] model per-patch Gaussian distributions, while PatchCore [8] retrieves outliers from a memory bank of normal features. These approaches are particularly appealing for robotics as they are sample-efficient and interpretable, providing localized heatmaps that highlight anomalous regions [8], [37]. Their efficiency and explainability make them especially suited for failure detection. We evaluate three AD families (see Section IV).

B. Failure detection in autonomous robotics

Ensuring that autonomous robotic agents can detect and respond to abnormal behaviors or failure during deployment is critical for both performance and human trust.

Traditional approaches to failure detection have focused on detection through supervised learning frameworks that rely on labeled examples of failures [40]–[43]. For instance, Liu et al. [40] trained an LSTM-based classifier using latent embeddings from a conditional VAE for RNN-based BC policies, while Gokmen et al. [41] employed a jointly trained state-value function to anticipate failure states in behavior cloning setups. However, these methods require failure data trajectories for training, limiting their applicability in novel or unstructured environments.

Several alternative approaches have been explored. Wang et al. [44] collect failure data via self-reset to train a classifier, requiring around 2,000 trajectories (2 hours of data), which limits scalability. He et al. [45] use random network distillation to filter out out-of-distribution trajectories, while Sun et al. [46] reduce uncertainty by predicting reward intervals. However, these methods do not address failure detection during execution. In contrast, other approaches such as Hafez et al. [10] verify LLM-generated trajectories using reachability analysis, and Agia et al. [11] introduce a statistical temporal action consistency (STAC) metric with VLMs to detect execution-time anomalies. Still, both focus solely on trajectory-level data.

The closest work to ours, FAIL-Detect [12], performs runtime failure detection in imitation learning with Continuous Normalizing Flows (CNF) [47], but does not distinguish benign anomalies from task-critical failures and assumes that the task is performed with the same temporal consistency during inference as in demonstrations. Instead, our method relies on a representation based formulation that captures both temporal progression and spatial structure, enabling alignment across variable execution speeds, localized heatmaps, and more interpretable scores. Combined with a semantic filtering stage using a VLM, this design narrows detection to genuine failures and yields significant gains, achieving +17.38% accuracy on BotFails.

III. Method

A. Problem formulation

We consider a generative robotic policy f_θ trained using Imitation Learning. At inference, the policy maps an observation O_t to an action $A_t = f_\theta(O_t)$. Our goal is to design a monitoring module capable of identifying execution anomalies and, among them, failures that compromise task completion. We decompose the problem into two levels: **Anomaly detection**: an anomaly detector D_A assigns a score to each observation O_t, reflecting the likelihood of deviation from nominal behavior. **Failure detection**: a failure detector D_F refines this decision by filtering out benign anomalies (e.g., harmless scene changes) while flagging failure-related

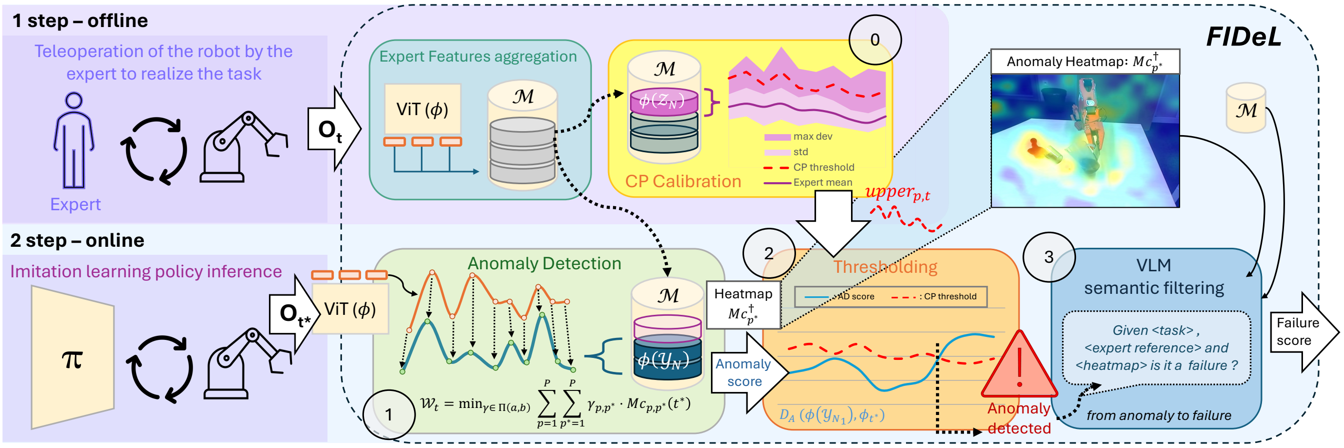

Fig. 1: We introduce **FIDeL**, a framework for detecting failures in imitation learning (IL) policies. **- Offline -** Expert demonstrations are first encoded and stored in a memory buffer $\mathcal{M}$. 0. Conformal Prediction Calibration — A decision threshold is computed from $\mathcal{M}$ using Conformal Prediction to determine when a score should be considered anomalous. **- Online -** 1. Anomaly Detection — During policy execution, anomaly scores and heatmaps are computed from incoming observations. 2. Thresholding — If the score exceeds the calibrated threshold, an anomaly is flagged. 3. Semantic Filtering — Since not all anomalies are failure-related, a VLM filter discards benign deviations and flags only failures that threaten task success.

anomalies. We adopt a one-class formulation, requiring only nominal demonstrations. Let $\mathcal{X}_N$ denote the dataset of expert trajectories, consisting of at most a few dozen episodes. It is randomly partitioned into two subsets of equal sizes: $\mathcal{Y}_N$ for constructing the anomaly detector D_A and $\mathcal{Z}_N$ for calibrating thresholds.

B. Method overview

Our framework comprises two stages: an **Offline phase** during which observations from $\mathcal{X}_N$ are encoded using a vision encoder ϕ, yielding patch-level features $\phi_t = \phi(O_t)$. These features are aggregated into a statistical memory $\mathcal{M}$ (Sec. III-C), which models the nominal distribution. The calibration subset $\mathcal{Z}_N$ is then used to derive adaptive thresholds via conformal prediction, producing time and patch dependent bounds. During the **online phase.**, at each timestep t^*, the policy outputs an action $A_{t^*} = f_\theta(O_{t^*})$. In parallel, as illustrated in Fig. 1, the anomaly detector D_A computes a score $D_A(\phi(\mathcal{Y}_N), \phi_{t^*})$ by aligning query features ϕ_{t^*} with keys from $\mathcal{Y}_N$ using optimal transport. This score is compared against conformal thresholds to decide if O_{t^*} is anomalous. If flagged, the failure detector D_F invokes a VLM to distinguish harmless anomalies from genuine failures.

C. Memory representation

The memory module $\mathcal{M}$ is built offline from nominal demonstrations. After encoding trajectories using a vision encoder (resnet18 [48] or dinoV2 [49]), we obtain a feature tensor $\phi(\mathcal{Y}_N) \in \mathbb{R}^{N\times T\times P\times F}$, where N is the number of episodes, T the horizon length, P the number of spatial patches per frame, and F the feature dimension. To reduce memory cost and ensure efficient inference, features are aggregated across episodes by computing Gaussian statistics. The resulting memory encodes the expected distribution of nominal features:

$$\mathcal{M} = \{\mu_{t,p,f},\, \sigma_{t,p,f} \mid t \in [\![1,T]\!],\, p \in [\![1,P]\!],\, f \in [\![1,F]\!]\}\,,$$

where $\mu_{t,p,f}$ and $\sigma_{t,p,f}$ are the empirical mean and standard deviation of feature dimension f, estimated across episodes for patch p at timestep t.

D. Anomaly score computation

At runtime, in parallel with the operation of the policy f_θ, we retrieve the observation O_{t^*}. Features are then extracted from O_{t^*} using the vision encoder, resulting in $\phi^{query}_{t^*} = \phi(O_{t^*}) \in \mathbb{R}^{P\times F}$. Subsequently, we compare this query data–whose indices hold a "$*$"– with the memory, i.e., the stored data, which we refer to as the keys. Anomaly scoring relies on an optimal transport (OT) alignment between the query and memorized features [50], [51]. Rather than comparing patches one by one, OT seeks the best global alignment between the two sets of patches: how much "feature mass" from query patches needs to be moved to match the key (memory) patches, and at what cost. Formally, we look for a transport plan $\gamma \in \mathbb{R}_+^{P*P}$ that redistributes the query mass into the key mass with minimum total cost. We define a ground cost function $c : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}^+$ between two feature vectors, implemented here as normalized Euclidean distance (i.e., a diagonal Mahalanobis distance), where each feature dimension is scaled by the corresponding standard deviation stored in $\mathcal{M}$. This gives rise to a cost matrix $M_c(t) \in \mathbb{R}^{P\times P}$ for each memory timestep t:

$$M_c(t)[p, p^*] = c\big(\phi^{query}_{t^*,p^*},\, (\mu_{t,p}, \sigma_{t,p})\big). \tag{1}$$

We then seek a coupling matrix $\gamma \in \Pi(a, b)$, where

$$\Pi(a,b) = \left\{\gamma \in \mathbb{R}_+^{P\times P} \;\middle|\; \gamma\mathbf{1} = a,\, \gamma^\top\mathbf{1} = b\right\}$$

is the set of admissible transport plans between two uniform distributions $a, b \in \Delta^P$. The OT cost between query and

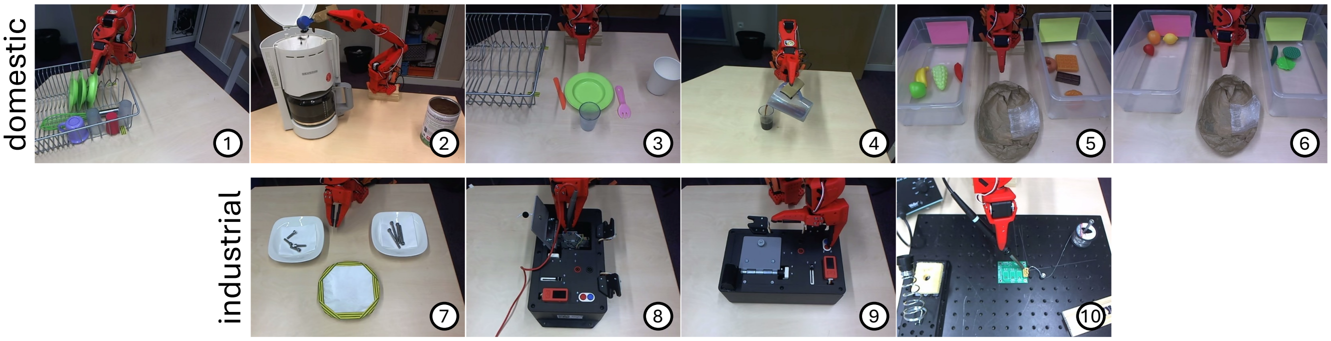

Fig. 2: BotFails dataset illustration - all images are illustrations of BotFails tasks executed by the expert - **Domestic** - 1. clear away the dishes, 2. make coffee, 3. set the table, 4. pour coffee, 5. sort groceries, 6. sort fruits and vegetables - **Industrial** - 7. sort screws, 8. measure voltage, 9. press buttons, 10. solder.

memory features at time t is thus:

$$\mathcal{W}_t = \min_{\gamma \in \Pi(a,b)} \langle M_c(t), \gamma \rangle \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product.

The final anomaly score is obtained as the minimum transport cost across nominal timesteps:

$$D_A(\phi(\mathcal{Y}_N), \phi_{t^*}) = \min_{t \in [\![1,T]\!]} \mathcal{W}_t, \quad t_{\min} = \arg \min_{t \in [\![1,T]\!]} \mathcal{W}_t \tag{3}$$

In practice, solving the OT problem exactly is costly; we rely on entropic regularization (Sinkhorn iterations) to compute an efficient approximation [50].

E. Thresholding via Spatially-Aware Conformal Prediction

We extend the Conformal Prediction (CP) framework [18], [19] to define a dynamic threshold for our anomaly detector D_A, accounting for both temporal structure and patch-level variations.

- Offline calibration -
We divide the expert calibration set $\mathcal{Z}_N$ in two subsets $\mathcal{E}_A$ and $\mathcal{E}_B$. Then, we compute the anomaly calibration score $D_A(\phi(\mathcal{Y}_N), \phi(\mathcal{E}_A))$ which is used to compute the mean anomaly score μ_p^t over nominal examples, while $D_A(\phi(\mathcal{Y}_N), \phi(\mathcal{E}_B))$ is used to compute deviations. The split aims to maintain the exchangeability assumption [18]. A threshold bandwidth h is then set as the $(1-\alpha)$-quantile of the maximum deviations observed on $\mathcal{E}_B$. This bandwidth yields a spatio-temporal upper bound $\text{upper}_{p,t} = \mu_p^t + \rho_p^t.h$, which controls the false positive rate at level α under exchangeability. With ρ_p^t, a patch-wise modulation factor, capturing variability across time and space [18]. Figure 3 depicts the upper bound (Threshold) as a dotted red line.

- Online thresholding -
At runtime, the current observation is aligned to its closest nominal timestep $t_{\min}$ using OT (Eq. (3)), and patch-level scores are compared to their corresponding bounds $\text{upper}_{p^*,t_{\min}}$. An input is flagged as anomalous when the fraction of patches exceeding their respective bounds surpasses a user-defined percentage. This parameter allows tuning the detector's sensitivity to the task context: stricter values improve responsiveness in fine-grained or sensitive scenarios, while more permissive ones reduce false positives in less sensitive settings. Additionally, non-critical spatial regions can be masked, further focusing detection on areas of interest.

F. Semantic filtering

We leverage Qwen 2.5-7b [52] as a Vision-Language Model (VLM) to distinguish between benign anomalies and task-relevant failures. Formally, we define the failure detector D_F as a binary classifier operating on both contextual task information and visual evidence:

$$D_F : \big(d,\, O(\mathcal{X}_{N,t_{\min}}),\, O_{t^*},\, Mc^{\dagger}_{p^*}\big) \;\mapsto\; \{0,1\},$$

where d is a task-specific textual description provided by the user, $O(\mathcal{X}_{N,t_{\min}}) \in O(\mathcal{X}_N)$ is a reference (expert) demonstration frame at timestep $t_{\min}$ (see Eq. (3)), O_{t^*} is the current observation, and $Mc^{\dagger}_{p^*}$ is the anomaly heatmap highlighting the most suspicious region(s) in the image. $Mc^{\dagger}_{p^*} \in \mathbb{R}^P$ is derived from the cost matrix (eq.(1)) at the most similar time step $t_{\min}$ (eq.(3)) by extracting the minimum transport cost per query patch:

$$Mc^{\dagger}_{p^*} = \min_{p} \left(Mc_{p,p^*}(t_{\min}) \right) \tag{4}$$

The VLM is prompted with $(d, O(\mathcal{X}_{N,t_{\min}}), O_{t^*}, Mc^{\dagger}_{p^*})$ and asked to:

1) Identify the semantic nature of the anomaly by comparing the two images and interpreting the heatmap.
2) Classify the anomaly as either: **False positive** ($D_F = 0$): the anomaly is benign, the robot may continue. **Failure** ($D_F = 1$): the anomaly is a failure.

The combination of nominal reference ($O(\mathcal{X}_{N,t_{\min}})$) with structured context (d) and localized anomaly cues ($Mc^{\dagger}_{p^*}$), allows to constrain the reasoning space of the VLM.

IV. EVALUATION

To rigorously evaluate the effectiveness of our failure detection approach, we introduce a dedicated dataset specifically designed for robotic failure monitoring. Our method is benchmarked on this dataset alongside several state-of-the-art baseline methods adapted to our experimental setting.

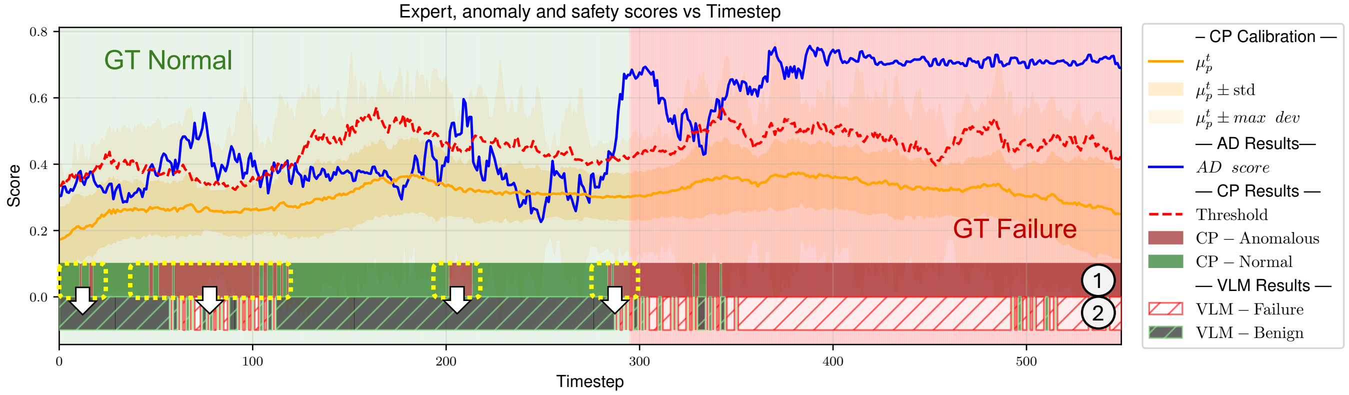

Fig. 3: Score illustration - Real-π soldering - episode 15 over 20 episodes of the evaluation set, score obtained with Representation and CP time. The Anomaly Detection score corresponds to the output of the AD module $D_A(\phi(\mathcal{Y}_N), \phi_{t^*})$. μ_p^t is obtained when computing $D_A(\phi(\mathcal{Y}_N), \phi(\mathcal{E}_A))$ and allows to compute the Threshold values, see section III-E. When the Anomaly Detection score is below the Threshold, no anomaly is detected (CP-Normal) and when it is above, an anomaly is flagged (CP-Anomalous). We observe that transitioning from anomaly detection ① to failure detection ② through the use of the VLM reduces the number of false positives (highlighted in the yellow boxes). However, this additional filtering step also introduces a small number of false negatives.

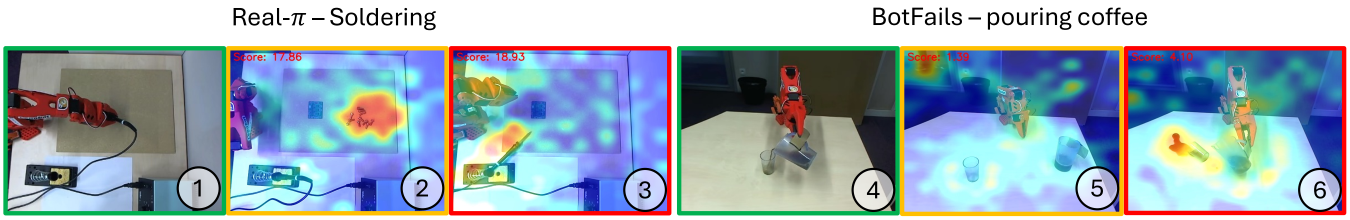

Fig. 4: Heatmaps illustration - obtained using Representation AD and temporal/spatial CP - **Real-π**: ① expert, ② benign anomaly: screws present in the work plan, ③ failure: dropping the iron - **BotFails**: ④ expert, ⑤ benign anomaly: someone walking in the background (top left corner), ⑥ failure: spilling the cup.

	Representation (ours)		*logpZ0* [12]		*lopO*		*AE*		*STAC* [11]	
	AUROC ↑	F1@Opt ↑	AUROC ↑	F1@Opt ↑	AUROC ↑	F1@Opt ↑	AUROC ↑	F1@Opt ↑	AUROC ↑	F1@Opt ↑
BotFails	**70.50** **±0**	**62.90** **±0**	65.20 ±0.15	59.87 ±0.06	58.30 ±0.70	53.48 ±0.02	49.90 ±1.08	52.12 ±0.08	N/A N/A	N/A N/A
Real-π soldering	**86.72** **±0**	**78.29** **±0**	82.62 ±0.08	78.03 ±0.02	76.16 ±0.42	70.99 ±0.03	81.07 ±0.88	76.33 ±0.06	53.55 ±0	63.93 ±0

TABLE I: Anomaly Detection evaluation before thresholding – mean AUROC and mean F1-score at optimal threshold (**in %**) for anomaly detection only evaluated on the **BotFails** dataset and **Real-π soldering task**, across various AD methods. Best results are highlighted in green and bold. Each task score reflects performance over multiple rollouts (several dozens per task).

A. *Dataset*

*1) We introduce **BotFails**:* given the scarcity of available datasets in the robotics failure detection field, we created a new dataset specifically designed for general failure situations in robotics, incorporating vision, proprioception, and natural language instructions. Data collection was performed using master arm teleoperation with LeRobot [20], a real-world robotic platform. BotFails has been curated to maximize semantic diversity across task types. The dataset covers 10 distinct tasks, with 6 tasks reflecting domestic environments, and 4 tasks representative of industrial contexts. The tasks are illustrated in Figure 2. Each task features several anomaly types. Some are not failure-related (e.g. an unknown object is present in the scene) and some are real failures (e.g. manipulation mistake, semantic mistake...).

*2) **Real-π** dataset:* We additionally evaluated FIDeL on more realistic data by performing inference on trajectories generated with ACT [3], an IL policy. While these data contain fewer types of anomalies/failures—due to the lack of control over the policy's behavior—they better reflect real-world execution scenarios. We conducted experiments on a soldering task. Two types of labels are available with these datasets: anomaly detection labels and failure labels which only include anomalies related to failure scenarios.

B. *Baselines*

We evaluate FIDeL against three families of baselines: (i) AD methods relying exclusively on nominal data, (ii) thresholding strategies for anomaly scoring, and (iii) end-to-end failure detection modules that combine detection with additional filtering or semantic reasoning.

1) Anomaly Detection (AD) baselines: We first compare against representative approaches from recent advances in robotic AD and Vision Anomaly Detection (VAD). Like FIDeL, these methods are trained exclusively on nominal demonstrations, without requiring annotated anomalies.

FAIL-Detect [12] introduces two anomaly scoring mechanisms based on Conditional Normalizing Flows (CNFs): **lopO** and **logpZ0** [53]. - **lopO** estimates the likelihood of an observation O_t under a CNF fitted on expert demonstrations; anomalies correspond to low-likelihood samples. - **logpZ0**, instead, integrates the CNF backward from O_t to compute its corresponding latent variable Z_{O_t}. Under nominal conditions, Z_{O_t} is approximately Gaussian, so large values of $\|Z_{O_t}\|^2$ are indicative of anomalies. Compared to lopO, this method avoids explicitly computing the flow's divergence, improving stability in high-dimensional settings.

In addition, we include a **reconstruction-based AD baseline**, inspired by SOTA methods for video anomaly detection [26], [54], [55]. We train an AutoEncoder (AE) on expert data to learn a compact latent representation. At inference, high reconstruction error signals a deviation from expert-like behavior and is used as an anomaly score.

Finally, we consider STAC (Statistical Temporal Action Consistency) [11] which monitors the temporal consistency of a generative policy. At each step, the policy predicts a sequence of future actions; overlapping parts of successive predictions are compared using statistical distances (e.g., MMD, KL). When the policy is in-distribution, these overlapping action distributions remain consistent, yielding low distances. Conversely, large divergences signal out-of-distribution states and potential failure. We can only evaluate STAC on the Real-π dataset given that it requires the sequence of actions predicted by the policy.

2) Thresholding baselines: Raw anomaly scores need to be mapped to binary anomaly decisions. To assess the contribution of our conformal thresholding mechanism, we compare against several thresholding strategies: - **CP-time**, a standard conformal prediction approach using only temporal deviations from reference demonstrations. - **CP-time+space**, our spatial extension that accounts for both temporal and spatial (patch-level) variations. - **Gaussian assumption**, a simpler baseline with thresholds derived from a Gaussian fit of calibration scores (mean and variance) at each timestep.

3) End-to-end failure detection baselines: Beyond isolated AD modules, we also benchmark FIDeL against SOTA full failure detection pipelines: - **FAIL-Detect** [12], in its end-to-end configuration. - **Sentinel** [11], which combines STAC and VLM monitoring in parallel. - **VLM only**: we use Qwen 2.5 [52], as a failure classifier. It receives the 5 last images and the user prompt and decides whether a failure occurred or not.

C. Evaluation Protocol

Anomaly Detection (AD) module. We first evaluate anomaly scoring independently of thresholds by extracting raw anomaly scores. Labels are defined at the frame level (0 = nominal, 1 = anomaly). Metrics: AUROC (balanced evaluation of TPR/FPR across thresholds) and F1 at the optimal threshold (best precision–recall balance).

Thresholding module. We then assess thresholding methods at the frame level (0 = normal sample, 1 = anomaly), using: **TPR**, **TNR**, **Balanced Accuracy** $= (\text{TPR} + \text{TNR})/2$, **Weighted Accuracy** $= \beta \cdot \text{TNR} + (1 - \beta) \cdot \text{TPR}$ with $\beta =$ number of normal samples/total number of samples.

End-to-end evaluation. Finally, we evaluate the full failure detection pipeline (including semantic filtering), where metrics only consider failure-related anomalies as positive samples. Metrics (TPR, TNR, Balanced/Weighted and Accuracy) are reported at observation level to assess the ability to detect true failures while ignoring benign anomalies.

D. Results and discussion

Table I reports the performance of anomaly detection (AD) methods evaluated directly on raw anomaly scores. Our representation-based approach consistently achieves the best AUROC and F1@Opt across both datasets, surpassing all baselines by a clear margin. On BotFails, it yields a $+5.3\%$ AUROC gain over the second-best method (logpZ0), confirming robustness to the high semantic variability of anomalies. On the Real-π soldering task, it further widens the gap, reaching 86.7% AUROC versus 82.6% for logpZ0 and 53.6% for STAC. These results show that compact nominal representations outperform reconstruction- and likelihood-based methods in capturing deviations from expert behavior.

Figure 5 compares thresholding strategies applied to anomaly scores. The best results are obtained with our representation-based method combined with CP-time&space, which yields robust performance across datasets. Conformal prediction (CP) generally outperforms Gaussian thresholding, as it makes no distributional assumption and adapts to diverse score behaviors. In contrast, Gaussian fitting assumes Gaussian-distributed scores, an approximation that often fails under extreme conditions, leading to higher variance and degraded weighted accuracy—especially for logpZ0, where over-flagging anomalies reduces TNR. Still, Gaussian thresholding is not entirely ineffective: in some tasks where the score distribution is closer to normal, it can match or even surpass CP-time. Comparing CP variants, CP-time&space consistently improves over CP-time. On Real-π, the gain is marked since anomalies in soldering are spatially localized. On BotFails, where anomalies are more diverse, the improvement is smaller: CP-time&space still yields higher accuracy, though its weighted accuracy is marginally lower (–0.17%), reflecting a TPR–TNR trade-off.

End-to-end results (Figure 6) highlight the importance of semantic filtering when converting anomaly labels to failure labels. AD modules suffer a sharp performance drop when benign anomalies are relabeled as nominal, especially in TNR; for example, our Representation method with CP-time&space loses 23% TNR on Real-π. Adding semantic filtering significantly mitigates the TNR degradation, enabling more balanced detection (85.8% TPR / 74.8% TNR). Fail-Detect tends to overpredict anomalies, as it cannot distinguish true failures from benign deviations, while Sentinel

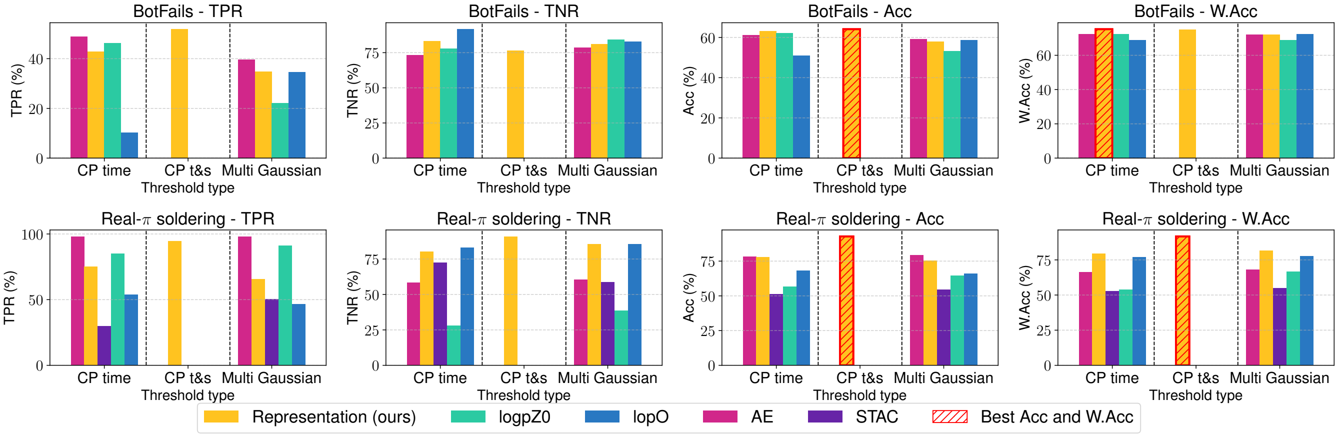

Fig. 5: Anomaly Detection evaluation with thresholding - mean True Positive Rate (TPR), True Negative Rate (TNR), balanced accuracy (Acc), weighted accuracy (W.Acc), (**in %**), for various Anomaly Detection types and various thresholding types–temporal Conformal Prediction (CP time), temporal and spatial Conformal Prediction (CP t&s), Multiple Gaussian–evaluated on real world task from the BotFails dataset and soldering task operated autonomously with ACT [3]. Best Acc and W.Acc in hatched red.

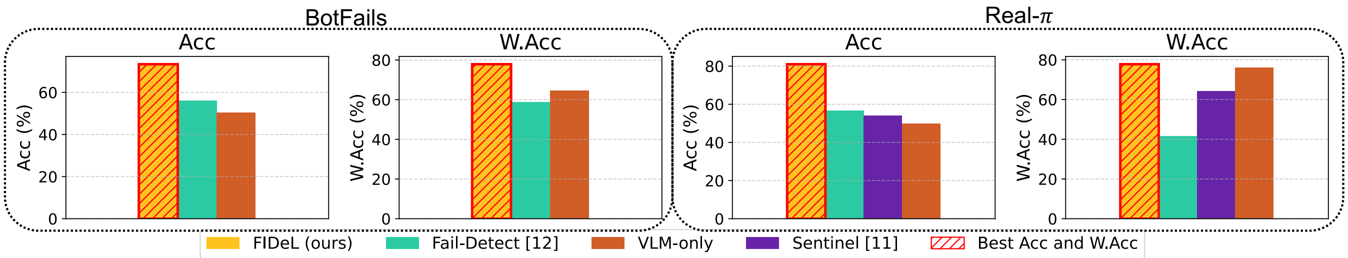

Fig. 6: End-to-end system evaluation (including semantic filtering) - accuracy (Acc) and weighted accuracy (W.Acc) for various robotics failure detection systems. The labeling is different for this evaluation. While AD and Threshold module are evaluated using **anomaly labels**, the end-to-end system is evaluated using **failures labels**. Which means that all failure-related anomaly labels remain 1, while non-failure anomalies are relabeled from 1 to 0. FiDeL = Representation + CP t&s + Semantic filtering

favors nominal predictions, likely missing failures that are purely visual and not reflected in proprioceptive signals. These results show that semantic filtering is crucial for reliable failure monitoring—avoiding unnecessary interventions while still capturing genuine failures.

V. CONCLUSIONS

We introduced FIDeL, a representation-based anomaly detection module tailored for IL robotic policies. By combining optimal transport alignment, spatially-aware conformal thresholds, and semantic filtering, our approach enables reliable failure detection with interpretable localization. Experiments across diverse tasks show that FIDeL balances robustness to natural task variability with sensitivity to failure related deviations, highlighting the benefit of spatially-extended conformal prediction. Beyond detecting anomalies, FIDeL provides a practical interface for monitoring robot behavior in deployment, where reliable monitoring hinges not only on policy quality but also on the ability to recognize and interpret failures.

VI. LIMITATIONS

FIDeL shows strong failure detection in real-world robotic tasks, but several limitations remain. First, its performance depends heavily on the diversity of demonstrations, which may cause benign variations to be flagged as anomalies. While the VLM filter helps reduce false positives, its use is computationally costly, and excessive false alarms would slow inference. Second, the method's spatial and temporal invariance, though useful for robustness, can reduce sensitivity to fine-grained or order-dependent deviations. This is particularly problematic for non-Markovian anomalies, where the correctness of an action depends not only on the current state but also on the history of states or actions that preceded it. (e.g., press buttons to perform a specific sequence). Detecting such cases requires explicit temporal modeling. Finally, balancing invariance and sensitivity remains task-dependent: some applications benefit from tolerance to small perturbations, while others demand detection of subtle deviations such as misalignments or object misplacements.

ACKNOWLEDGEMENTS

This publication was made possible by the use of the CEA List FactoryAI supercomputer, financially supported by the Ile-de-France Regional Council. This work was partly funded by the European Union's Horizon Europe Research and Innovation program under grant agreement n° 101135708 (JARVIS Project).

REFERENCES

[1] P. Intelligence, K. Black, et al., "$\pi_{0.5}$: a vision-language-action model with open-world generalization," 2025. [Online]. Available: https://arxiv.org/abs/2504.16054

[2] C. Chi, Z. Xu, et al., "Diffusion policy: Visuomotor policy learning via action diffusion," 2024. [Online]. Available: https://arxiv.org/abs/2303.04137

[3] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," 2023. [Online]. Available: https://arxiv.org/abs/2304.13705

[4] Y. Gal et al., "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in Proc. of ICML, 2016.

[5] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in Proc. of NeurIPS, 2017.

[6] O. Lockwood and M. Si, "A review of uncertainty for deep reinforcement learning," in Proc. of AAAI, 2022.

[7] Y. Zhu, J. Dong, and H. Lam, "Uncertainty quantification and exploration for reinforcement learning," Operations Research, 2024.

[8] K. Roth et al., "Towards total recall in industrial anomaly detection," in Proc. of CVPR, 2022.

[9] Y. Fan et al., "Video anomaly detection and localization via gaussian mixture fully convolutional variational autoencoder," Computer Vision and Image Understanding, 2020.

[10] A. Hafez, A. N. Akhormeh, A. Hegazy, and A. Alanwar, "Safe LLM-Controlled Robots with Formal Guarantees via Reachability Analysis," 2025. [Online]. Available: https://arxiv.org/abs/2503.03911

[11] C. Agia et al., "Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress," in Proc. of CoRL, 2025.

[12] C. Xu et al., "Can we detect failures without failure data? uncertainty-aware runtime failure detection for imitation learning policies," arXiv preprint arXiv:2503.08558, 2025.

[13] Y. Cui, Z. Liu, and S. Lian, "A survey on unsupervised anomaly detection algorithms for industrial images," IEEE Access, 2023.

[14] M. Abdalla, S. Javed, M. A. Radi, A. Ulhaq, and N. Werghi, "Video anomaly detection in 10 years: A survey and outlook," 2024. [Online]. Available: https://arxiv.org/abs/2405.19387

[15] P. Wu, C. Pan, Y. Yan, G. Pang, P. Wang, and Y. Zhang, "Deep learning for video anomaly detection: A review," 2024. [Online]. Available: https://arxiv.org/abs/2409.05383

[16] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," 2019. [Online]. Available: https://arxiv.org/abs/1901.03407

[17] J. Tax, David M. and W. Duin, Robert P. "Support vector domain description," Pattern Recognition Letters, 1999.

[18] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," 2017. [Online]. Available: https://arxiv.org/abs/1604.04173

[19] J. Diquigiovanni, M. Fontana, and S. Vantini, "The importance of being a band: Finite-sample exact distribution-free prediction sets for functional data," 2021. [Online]. Available: https://arxiv.org/abs/2102.06746

[20] R. Cadene, S. Alibert, A. Soare, Q. Gallouedec, A. Zouitine, and T. Wolf, "Lerobot: State-of-the-art machine learning for real-world robotics in pytorch," https://github.com/huggingface/lerobot, 2024.

[21] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, "Cutpaste: Self-supervised learning for anomaly detection and localization," in Proc. of CVPR, 2021.

[22] V. Zavrtanik, M. Kristan, and D. Skočaj, "Draem – a discriminatively trained reconstruction embedding for surface anomaly detection," in Proc. of ICCV, 2021.

[23] H. M. Schlüter, J. Tan, B. Hou, and B. Kainz, "Natural synthetic anomalies for self-supervised anomaly detection and localization," in Proc. of ICCV, 2022.

[24] S. Venkataramanan, K.-C. Peng, R. V. Singh, and A. Mahalanobis, "Attention guided anomaly localization in images," in Proc. of ECCV, 2020.

[25] Y. Liu, C. Zhuang, and F. Lu, "Unsupervised two-stage anomaly detection," 2021. [Online]. Available: https://arxiv.org/abs/2103.11671

[26] Y. Shi, J. Yang, and Z. Qi, "Unsupervised anomaly segmentation via deep feature reconstruction," Neurocomputing, 2021.

[27] G. Wang, S. Han, E. Ding, and D. Huang, "Student-teacher feature pyramid matching for anomaly detection," in Proc. of BMVC, 2021.

[28] S. Yamada, S. Kamiya, and K. Hotta, "Reconstructed student-teacher and discriminative networks for anomaly detection," in Proc. of IROS, 2022.

[29] M. Rudolph, T. Wehrbein, B. Rosenhahn, and B. Wandt, "Asymmetric student-teacher networks for industrial anomaly detection," in Proc. of the IEEE/CVF winter conference on applications of computer vision, 2023.

[30] L.-L. Chiu and S.-H. Lai, "Self-supervised normalizing flows for image anomaly detection and localization," in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2023, pp. 2927–2936.

[31] R. Bouman and T. Heskes, "Autoencoders for anomaly detection are unreliable," 2025. [Online]. Available: https://arxiv.org/abs/2501.13864

[32] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in Proc. of ICML, 2015.

[33] D. Gudovskiy, S. Ishizaka, and K. Kozuka, "Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows," in Proc. of the IEEE/CVF winter conference on applications of computer vision, 2021.

[34] M. Rudolph, T. Wehrbein, B. Rosenhahn, and B. Wandt, "Fully convolutional cross-scale-flows for image-based defect detection," in Proc. of the IEEE/CVF winter conference on applications of computer vision, 2021.

[35] J. Yu et al., "Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows," 2021. [Online]. Available: https://arxiv.org/abs/2111.07677

[36] P. Kirichenko, P. Izmailov, and A. G. Wilson, "Why normalizing flows fail to detect out-of-distribution data," 2020. [Online]. Available: https://arxiv.org/abs/2006.08545

[37] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "Padim: a patch distribution modeling framework for anomaly detection and localization," in Proc. of ICPR, 2020.

[38] S. Wang, L. Wu, L. Cui, and Y. Shen, "Glancing at the patch: Anomaly localization with global and local feature comparison," in Proc. of CVPR, 2021.

[39] Y. Zheng et al., "Focus your distribution: Coarse-to-fine non-contrastive learning for anomaly detection and localization," in Proc. of ICME, 2022.

[40] H. Liu, S. Dass, R. Martín-Martín, and Y. Zhu, "Model-based runtime monitoring with interactive imitation learning," in Proc. of ICRA, 2024.

[41] C. Gokmen, D. Ho, and M. Khansari, "Asking for help: Failure prediction in behavioral cloning through value approximation," 2023. [Online]. Available: https://arxiv.org/abs/2302.04334

[42] H. Liu et al., "Multi-Task Interactive Robot Fleet Learning with Visual World Models," in Proc. of CoRL, 2024.

[43] A. Gupta, K. Chakraborty, and S. Bansal, "Detecting and Mitigating System-Level Anomalies of Vision-Based Controllers," in Proc. of ICRA, 2024.

[44] Y. Wang, T.-H. Wang, J. Mao, M. Hagenow, and J. Shah, "Grounding language plans in demonstrations through counterfactual perturbations," in Proc. of ICLR, 2024.

[45] N. He et al., "Rediffuser: Reliable decision-making using a diffuser with confidence estimation," in Proc. of ICML, 2024.

[46] J. Sun et al., "Conformal prediction for uncertainty-aware planning with diffusion dynamics model," in Proc. of NeurIPS, 2023.

[47] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in Proc. of NeurIPS, 2018.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. of CVPR, 2016.

[49] M. Oquab et al., "Dinov2: Learning robust visual features without supervision," TMLR, 2024.

[50] G. Papagiannis and Y. Li, "Imitation learning with sinkhorn distances," 2022. [Online]. Available: https://arxiv.org/abs/2008.09167

[51] R. Dadashi, L. Hussenot, M. Geist, and O. Pietquin, "Primal wasserstein imitation learning," 2021. [Online]. Available: https://arxiv.org/abs/2006.04678

[52] Qwen, :, A. Yang, and B. Y. et al, "Qwen2.5 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2412.15115

[53] C. Xu, X. Cheng, and Y. Xie, "Normalizing flow neural networks by jko scheme," in Proc. of NeurIPS, 2023.

[54] M. Hasan et al., "Learning temporal regularity in video sequences," in Proc. of CVPR, 2016.

[55] T. Wang et al., "Generative neural networks for anomaly detection in crowded scenes," IEEE TIFS, 2018.