Karlsruher Schriften
zur Anthropomatik

Band 74

Jürgen Beyerer, Tim Zander (Eds.)

Proceedings of the 2025 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory

KIT Scientific Publishing

Jürgen Beyerer, Tim Zander (Eds.)

Proceedings of the 2025 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory

Karlsruher Schriften zur Anthropomatik
Band 74
Herausgeber: Prof. Dr.-Ing. habil. Jürgen Beyerer

Eine Übersicht aller bisher in dieser Schriftenreihe
erschienenen Bände finden Sie am Ende des Buchs.

Proceedings of the 2025 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory

by
Jürgen Beyerer, Tim Zander (Eds.)

KIT Scientific Publishing

Impressum

Scientific Publishing

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

KIT Scientific Publishing is a registered trademark
of Karlsruhe Institute of Technology.
Reprint using the book cover is not allowed.

www.bibliothek.kit.edu/ksp.php | E-Mail: info@ksp.kit.edu | Shop: www.ksp.kit.edu

Print on Demand 2026 – Gedruckt auf FSC-zertifiziertem Papier

ISSN 1863-6489 (Schriftenreihe)
ISSN 2510-7259 (Tagungsband)
ISBN 978-3-7315-1481-7
DOI 10.5445/KSP/1000191946

Preface

In 2025, the annual joint workshop of the Fraunhofer Institute of Optronics, System Technologies, and Image Exploitation (IOSB) and the Vision and Fusion Laboratory (IES) of the Institute for Anthropomatics, Karlsruhe Institute of Technology (KIT) was once again hosted in a Black Forest house in Triberg-Nussbach, Germany.

For a week from the 27th of July to the 2nd of August, the PhD students of both institutions delivered extended reports on the status of their research and participated in heated discussions on topics ranging from computer vision, industrial production, optimisation, control theory, security to large language models. Most results and ideas presented at the workshop are collected in this book in the form of detailed technical reports. This volume provides a comprehensive and up-to-date overview of some of the research programs of the IES Laboratory and the Fraunhofer IOSB.

The editors thank Jonas Vogl, Laura Tzigiannis and Zeyun Zhong, for their efforts resulting in a pleasant and inspiring atmosphere throughout the week. We would also like to thank the doctoral students for writing and reviewing the technical reports, as well as for responding to the comments and suggestions of their colleagues.

Jürgen Beyerer & Tim Zander

Contents

Achieving Desired System Quality through Direct Control Strategy

Negar Arabizadeh

Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
negar.arabizadeh@kit.edu

Abstract

Achieving reliable control over product quality in manufacturing systems is difficult due to randomness and process variability. This work investigates control strategies that can act directly on quality-relevant attributes, with the goal of ensuring that final product quality relevant attributes satisfy predefined specifications. We investigated several methods, one of which is Approximate Dynamic Programming (ADP), ADP is explored as a promising framework for addressing this challenge. In this setting, we study the stochastic reachability problem, how it works, and different classes of stochastic reachability problems, in order to assess whether it provides a suitable framework for tackling this quality-oriented control problem. The control objective is formulated as a probabilistic reachability problem, where the aim is to maximize the likelihood of driving the system toward a desired quality region.

1 Introduction

Manufacturing systems must deliver products that satisfy strict acceptance criteria while operating under uncertainty caused by disturbances, process variability,

and limited observability. In practice, decisions about whether a product can be released are rarely based on tracking errors or nominal set-point deviations, but rather on a more fundamental question: How likely is it that the final product meets the required quality specifications? Traditional control techniques are not designed to answer this question explicitly, as they typically optimize deterministic performance measures and provide no direct probabilistic guarantees on terminal product quality.

This work adopts a decision-theoretic perspective in which quality control is treated as a probabilistic planning problem under uncertainty. The key idea is to directly regulate the evolution of quality relevant attributes throughout the production process, rather than treating quality as a passive outcome assessed only at the end. By embedding these attributes into the system state—using available measurements or estimators—the control objective becomes the maximization of the probability that the process terminates within an acceptable quality region while remaining within operational limits at all intermediate times.

Such objectives naturally align with stochastic reachability theory, which studies the ability of controlled stochastic systems to satisfy set-based specifications over a finite horizon. In this framework, quality requirements are represented as terminal target sets, operating constraints as safety regions, and the control task as maximizing the probability of reaching the target without violating safety constraints. The resulting value function provides a direct quantitative measure of quality assurance: it encodes the likelihood of successful production under uncertainty.

From an optimal control standpoint, Dynamic Programming (DP) offers an exact solution concept for stochastic reach–avoid problems. However, its direct application is infeasible for realistic production systems due to the continuous nature of the state and input spaces and the resulting computational complexity. Approximate Dynamic Programming (ADP) addresses this limitation by replacing the exact value function with a structured approximation and enforcing optimality conditions in a relaxed, sample-based manner. In particular, linear-programming-based ADP methods combined with smooth function approximators enable tractable computation of probabilistic performance bounds in continuous domains.

Motivated by these considerations, this paper investigates for designing the control framework that prioritizes terminal quality success probabilities as the primary performance objective. Rather than minimizing tracking errors or economic costs, the controller is designed to explicitly maximize the likelihood of producing in-spec output.

2 Related Work

The challenge of maintaining product quality in manufacturing systems subject to uncertainty has been studied from multiple angles, including robust control, stochastic optimization, and learning-based decision-making. Many existing methods focus on regulating process variables around nominal operating conditions while ensuring constraint satisfaction with high probability. Although such approaches—such as chance-constrained and robust control—are effective for reference tracking and constraint handling, they do not explicitly quantify the probability that a product will satisfy acceptance criteria at the end of production [5].

Several works address uncertainty in manufacturing by optimizing process parameters rather than designing feedback controllers. For instance, stochastic and robust modeling combined with evolutionary optimization has been used to identify fixed operating settings that improve reliability under uncertainty [2]. While these approaches enhance robustness, they do not account for real-time control or the dynamic evolution of quality during the production process. Comprehensive surveys such as [12] review Model Predictive Control (MPC) from an engineering perspective, covering theoretical foundations, implementation aspects, and applications across process and manufacturing industries. Related research has proposed nonlinear robust MPC schemes that retain the structure of nominal MPC while guaranteeing constraint satisfaction and stability in the presence of disturbances [8].

Approximate Dynamic Programming (ADP) provides an alternative framework for control design, particularly when long-term performance and probabilistic objectives are of interest. In contrast to MPC, which often relies on terminal costs or terminal constraints to ensure stability, ADP naturally incorporates infinite-

or long-horizon objectives through Bellman equations. Moreover, ADP can be implemented using either explicit process models or data-driven approximations, which is advantageous when accurate models are unavailable. Once trained offline, ADP-based controllers incur low online computational cost, as control actions are obtained by evaluating a policy rather than solving an optimization problem at each time step. ADP also accommodates general objective structures, including risk-sensitive and reachability-based criteria, that are difficult to handle within conventional MPC formulations [10, 4].

Classical industrial controllers, such as PID regulators, primarily focus on minimizing local tracking errors and are typically tuned heuristically based on simplified or linearized models of the plant. While effective for well-understood systems, their performance can deteriorate in the presence of strong nonlinearities, model mismatch, and stochastic disturbances. In contrast, probabilistic control formulations explicitly account for uncertainty and operating constraints, and aim to optimize the likelihood of achieving desired outcomes rather than merely stabilizing a reference trajectory.

Reinforcement learning (RL) has emerged as a data-driven alternative for control design, particularly in applications where large numbers of trajectories can be generated efficiently. Advances in deep RL have shown strong performance in fast or simulated environments [11, 6]. However, in slow and safety-critical production processes, direct online learning is often impractical due to limited data availability and the risks associated with exploration. As a result, RL-based approaches in manufacturing typically rely on digital twins or high-fidelity simulators for training [15]. Recent reviews highlight that sample inefficiency and safety concerns remain significant obstacles to deploying RL directly on physical process plants [9].

Stochastic reachability offers a different perspective by formulating control objectives in terms of maximizing the probability of reaching a desired target set while avoiding unsafe regions over a finite horizon. This viewpoint aligns naturally with industrial acceptance and release criteria. Dynamic Programming provides the theoretical foundation for reach–avoid problems in discrete-time settings [4]. When quality relevant attributes are only partially observed, partially observable models can be used to incorporate belief dynamics and information acquisition into the control framework [13].

2.1 Discrete-Time Stochastic Hybrid Systems

A discrete-time stochastic hybrid system (DTSHS) is a mathematical model used to describe systems whose evolution combines discrete mode switching, continuous-valued dynamics, and stochastic uncertainty. Such systems arise naturally in applications where logical decisions, operating modes, or events interact with physical processes that evolve in continuous state spaces under randomness.

In a DTSHS, the system state consists of two components: a discrete state that represents the current operating mode, and a continuous state whose dimension and dynamics may depend on the active discrete mode. Both components evolve in discrete time and are influenced by control inputs and probabilistic laws.

System Definition. Formally, a discrete-time stochastic hybrid system is defined as a tuple

$$H = (\mathcal{D}, n, \mathcal{X}, \Sigma, T_x, T_q, R),$$

where:

- $\mathcal{D} = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_m\}$ is a finite set of discrete modes.
- $n : \mathcal{Q} \to \mathbb{N}$ assigns to each mode $\mathbf{d} \in \mathcal{Q}$ the dimension of the continuous state space $\mathbb{R}^{n(q)}$.
- $\mathcal{X}$ is a compact Borel space of *transition control inputs*, which influence both continuous evolution and mode transitions.
- Σ is a compact Borel space of *reset control inputs*, which affect the continuous state after a mode switch.

The hybrid state space is given by

$$\mathcal{S} := \bigcup_{\mathbf{d} \in \mathcal{D}} \{\mathbf{d}\} \times \mathbb{R}^{n(\mathbf{d})},$$

and is endowed with the Borel σ-algebra $\mathcal{B}(\mathcal{S})$.

Stochastic State Evolution. Let $\mathbf{m}_k = (\mathbf{d}_k, \mathbf{s}_k) \in \mathcal{S}$ denote the hybrid state at time step k. The evolution of the system is governed by three stochastic kernels:

- The kernel
$$T_{\mathbf{d}}(\mathbf{d}' \mid \mathbf{m}_k, \mathbf{x}_k)$$
defines the probability of transitioning from the current discrete mode $\mathbf{d}_k$ to a new mode $\mathbf{d}' \in \mathcal{D}$, given the current hybrid state $\mathbf{m}_k$ and transition input $\mathbf{x}_k$.

- The kernel
$$T_{\mathbf{s}}(\mathbf{s}' \mid \mathbf{s}_k, \mathbf{x}_k)$$
describes the probabilistic evolution of the continuous state when the discrete mode does not change.

- The reset kernel
$$R(\mathbf{s}' \mid \mathbf{s}_k, \sigma_k, \mathbf{d}')$$
specifies the probability distribution of the continuous state after a transition to a new discrete mode $\mathbf{d}' \neq \mathbf{d}_k$, possibly depending on a reset input σ_k.

These components jointly define the stochastic evolution of the hybrid state.

Unified Transition Kernel. The discrete and continuous transitions can be combined into a single stochastic kernel on the hybrid state space.

Let $\mathbf{m}_{k+1} = (\mathbf{d}', \mathbf{s}')$. The one-step transition kernel

$$T_{\mathbf{m}}(\mathbf{m}' \mid \mathbf{m}_k, (\mathbf{x}_k, \sigma_k))$$

is defined as

$$T_{\mathbf{m}}(\mathbf{m}' \mid \mathbf{m}_k, (\mathbf{x}_k, \sigma_k)) = T_{\mathbf{d}}(\mathbf{d}' \mid \mathbf{m}_k, \mathbf{x}_k)\, \tau_{\mathbf{s}}(\mathbf{s}' \mid \mathbf{s}_k, \mathbf{x}_k, \sigma_k, \mathbf{d}'),$$

where $\tau_{\mathbf{s}}$ selects either $T_{\mathbf{s}}$ or R depending on whether the discrete mode changes. This kernel fully characterizes the probabilistic evolution of the DTSHS.

Initialization and Policies. The system is initialized at time $k = 0$ according to a probability measure

$$\pi : \mathcal{B}(\mathcal{M}) \to [0, 1]$$

on the hybrid state space. Control actions are selected using a feedback policy

$$\mu = (\mu_0, \mu_1, \ldots, \mu_{N-1}),$$

where each policy map

$$\mu_k : \mathcal{M} \to \mathcal{X} \times \Sigma$$

assigns control inputs based on the current hybrid state. Such policies are referred to as *Markov policies*, as the control action depends only on the present state and not on the full history.

Markov Process Interpretation. Under a Markov policy μ, the execution of a DTSHS is a time-inhomogeneous controlled Markov process on the state space $\mathcal{S}$. The sequence of hybrid states

$$\{\mathbf{m}_k\}_{k=0}^{N}$$

is a stochastic process defined on the canonical sample space

$$\Omega = \mathcal{S}^{N+1},$$

with a probability measure uniquely determined by the initial distribution π, the policy μ, and the transition kernel $T_{\mathbf{m}}$.

This Markovian structure enables the application of stochastic reachability theory and dynamic programming techniques to DTSHS models, allowing reachability, safety, and reach–avoid specifications to be analyzed and optimized in a unified probabilistic framework.

3 What is Stochastic Reachability?

Stochastic reachability concerns the analysis of dynamical systems whose evolution is affected by randomness, such as process noise, disturbances, or uncertain

transitions. Given a stochastic system and a target set and/or a safe set in the state space, stochastic reachability seeks to determine whether the system can reach a desired set or remain within a prescribed safe region over a given time horizon, and to quantify the probability with which such events occur. In contrast to deterministic reachability, which yields a binary yes/no answer, stochastic reachability provides probabilistic guarantees that capture the inherent uncertainty of system trajectories.

When control inputs are available, stochastic reachability naturally gives rise to a stochastic optimal control problem, where the objective is to maximize or minimize the probability of satisfying a reachability specification. Typical formulations include *reach* problems, which focus on the probability of reaching a target set; *safety* problems, which concern remaining within a safe set for a specified duration; and *reach-avoid* problems, which require reaching a target set while avoiding unsafe regions. These formulations provide a unifying framework for analyzing safety and performance of stochastic systems and are commonly addressed using dynamic programming on controlled Markov processes, resulting in value functions and associated control policies that achieve optimal reachability probabilities.

Mathematical Formulation

Consider a discrete-time stochastic system described by

$$\mathbf{s}_{k+1} = f(\mathbf{s}_k, \mathbf{x}_k, \mathbf{w}_k),$$

where $\mathbf{s}_k \in \mathcal{S}$ denotes the system state at time k, $\mathbf{x}_k \in \mathcal{U}$ is a control input, and $\mathbf{w}_k$ represents a random disturbance. Let $\mathcal{K}' \subseteq \mathcal{S}$ be a target set and $\mathcal{K} \subseteq \mathcal{S}$ a safe set. Typical stochastic reachability problems include determining the probability that the system reaches $\mathcal{K}'$ within a finite horizon, remains in $\mathcal{K}$ for all time steps up to a horizon N, or reaches $\mathcal{K}'$ while avoiding an unsafe set. These problems are formalized in terms of probabilities of events defined over the system's stochastic trajectories.

A commonly studied case is the probabilistic safety problem, which seeks to evaluate the probability that the system state remains within a safe set $\mathcal{K}$ for

all time steps $k = 0, \ldots, N$. For a given control policy π, this probability is defined as

$$\mathrm{P}^{\pi}_{\mathbf{s}_0}\{\mathbf{s}_k \in \mathcal{K} \ \ \forall k = 0, \ldots, N\},$$

where the probability is taken with respect to the stochastic process induced by the system dynamics under policy π, starting from the initial state $\mathbf{s}_0$. This probability can equivalently be expressed as an expectation of an indicator functional over trajectories:

$$P^{\pi}_{\mathbf{s}_0}\{\mathbf{s}_k \in \mathcal{K} \ \ \forall k\} = \mathbb{E}^{\pi}_{\mathbf{s}_0}\left[\prod_{k=0}^{N} \mathbf{1}_{\mathcal{K}}(\mathbf{s}_k)\right],$$

which provides a convenient mathematical representation for analysis and computation [1].

When control inputs are available, stochastic reachability naturally leads to a stochastic optimal control problem, where the objective is to maximize or minimize the probability of satisfying a reachability or safety specification. Rather than working directly with probabilities of events, this objective is typically expressed in terms of the expectation of a suitably defined indicator functional over system trajectories.

For a given initial state $\mathbf{s}_0$, the optimal reachability (or safety) value function is defined as

$$V_0(\mathbf{s}_0) = \sup_{\pi} \mathbb{E}^{\pi}_{x_0}\left[\prod_{k=0}^{N} \mathbf{1}_{\mathcal{A}}(\mathbf{s}_k)\right],$$

where π denotes an admissible control policy and $\mathbf{1}_{\mathcal{A}}$ is the indicator function of the safe set $\mathcal{A}$. This expectation represents the probability that the system, starting from $\mathbf{s}_0$ and controlled by policy π, satisfies the reachability or safety specification over the time horizon $[0, N]$.

The resulting stochastic optimal control problem can be solved using dynamic programming on an associated controlled Markov process. This yields a sequence of value functions $V_k(\mathbf{s})$, where $V_k(\mathbf{s})$ represents the optimal expected value of the reachability functional from state $\mathbf{s}$ at time k. The corresponding optimal policy specifies control actions that are maximally safe or maximally reaching. This framework forms the mathematical foundation of stochastic reachability analysis

and underlies many modern approaches in system verification, safety-critical control, and chance-constrained optimization [1].

3.1 Reachability Problem

The stochastic reachability problem concerns the probability that a stochastic system reaches a specified target set within a given time horizon. The objective is to quantify how likely it is that the system state enters a desired region of the state space despite the presence of randomness such as noise or uncertain transitions. In this formulation, no explicit safety constraint is imposed; the system is allowed to visit any state before reaching the target. Reachability problems are commonly used to evaluate performance objectives, such as goal achievement or task completion, in stochastic environments.

Let $\mathcal{S}$ denote the state space of a stochastic dynamical system and consider a discrete-time stochastic process $\{\mathbf{s}_k\}_{k\geq 0}$ evolving under a control policy π. Given a target set $\mathcal{K} \subseteq \mathcal{S}$ and a finite time horizon $N \in \mathbb{N}$, the stochastic reachability problem consists in computing the probability that the system state reaches the target set at least once within the time interval $[0, N]$.

Formally, for an initial state $\mathbf{s}_0 \in \mathcal{S}$, the reachability probability under policy π is defined as

$$P^{\pi}_{\mathbf{s}_0}(\mathcal{K}) = \mathbb{P}^{\pi}_{\mathbf{s}_0}\big(\exists\, k \in \{0, \dots, N\} \text{ such that } \mathbf{s}_k \in \mathcal{K}\big),$$

where the probability is taken with respect to the probability measure induced by the system dynamics and the policy π. This event-based definition captures the occurrence of the target-reaching event along a system trajectory.

An equivalent expectation-based representation of the reachability probability is obtained by introducing an indicator functional over trajectories:

$$P^{\pi}_{\mathbf{s}_0}(\mathcal{T}) = \mathbb{E}^{\pi}_{\mathbf{s}_0}\left[\sum_{k=0}^{N} \mathbf{1}_{\mathcal{K}}(\mathbf{s}_k) \prod_{j=0}^{k-1} \mathbf{1}_{\mathcal{K}^c}(x_j)\right],$$

where $\mathbf{1}_{\mathcal{K}}$ denotes the indicator function of the target set and $\mathbf{1}_{\mathcal{K}^c}$ denotes the indicator function of its complement. This expression evaluates to one if the

system reaches the target set for the first time at some time $k \leq N$, and to zero otherwise.

When control inputs are optimized, the stochastic reachability problem becomes a stochastic optimal control problem. The optimal reachability value function is defined as

$$V_0(\mathbf{s}_0) = \sup_{\pi} \mathbb{E}^{\pi}_{\mathbf{s}_0} \left[\sum_{k=0}^{N} \mathbf{1}_{\mathcal{K}}(\mathbf{s}_k) \prod_{j=0}^{k-1} \mathbf{1}_{\mathcal{K}^c}(\mathbf{s}_j) \right],$$

which represents the maximal probability of reaching the target set within the horizon $[0, N]$ starting from the initial state $\mathbf{s}_0$. This value function can be computed using dynamic programming on the associated controlled Markov process.

3.2 Reach-Avoid Problem

The reach-avoid problem is a central formulation in stochastic reachability theory for discrete-time stochastic systems and their equivalent representations as Markov decision processes (MDPs). In this framework, system evolution is modeled as a controlled stochastic process of the form

$$\mathbf{s}_{t+1} \sim Q(\cdot \mid \mathbf{s}_t, \mathbf{x}_t), \qquad (\mathbf{s}_t, \mathbf{x}_t) \in \mathcal{S} \times \mathcal{X},$$

where $\mathcal{S} \subseteq \mathbb{R}^n$ is the state space, $\mathcal{X} \subseteq \mathbb{R}^m$ is the control space, and Q is a stochastic transition kernel that specifies the probability distribution of the next state conditioned on the current state and control input. Both the state and control spaces are generally infinite-dimensional, reflecting the continuous nature of the system dynamics.

Let $K' \in \mathcal{B}(\mathcal{S})$ denote a *safe set* and let $K \subseteq K'$ denote a *target set*, where $\mathcal{B}(\mathcal{S})$ is the Borel σ-algebra on the state space. The reach-avoid objective over a finite time horizon N is to maximize the probability that the system reaches the target set K at some time $t_K \leq N$, while remaining inside the safe set K' for all times prior to reaching the target. In other words, the system must reach K before ever leaving K', and this event must occur within the prescribed time horizon.

For a given control policy $\mu = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$, where each $\mu_i : \mathcal{S} \to \mathcal{X}$ is a state-feedback control law, and an initial state $\mathbf{s}_0 \in \mathcal{S}$, the reach-avoid probability is defined as

$$r^{\mu}_{\mathbf{s}_0}(K, K') := \mathbb{P}^{\mu}_{\mathbf{s}_0}\Big(\exists\, j \in [0, N] : x_j \in K \ \wedge\ \forall\, i \in [0, j-1],\ x_i \in K' \setminus K\Big).$$

This expression represents the probability that the system trajectory reaches the target set K for the first time at some time $j \leq N$, while remaining inside the safe region K' and outside the target set before that time.

An equivalent formulation expresses this probability as the expected value of a sum–multiplicative cost functional:

$$r^{\mu}_{\mathbf{s}_0}(K, K') = \mathbb{E}^{\mu}_{\mathbf{s}_0}\left[\sum_{j=0}^{N}\left(\prod_{i=0}^{j-1}\mathbf{1}_{K'\setminus K}(\mathbf{s}_i)\right)\mathbf{1}_K(\mathbf{s}_j)\right],$$

where $\mathbf{1}_K$ and $\mathbf{1}_{K'\setminus K}$ denote the indicator functions of the target set and the safe-but-not-target set, respectively. This functional evaluates to one if the system reaches the target set before leaving the safe set, and to zero otherwise, thereby encoding the reach-avoid event in an expectation form.

The reach-avoid problem is therefore formulated as a stochastic optimal control problem: the objective is to select a control policy μ that maximizes this expected value. The optimal solution is characterized by a sequence of value functions $V_k^* : \mathcal{S} \to [0, 1]$, computed recursively using dynamic programming. The value function $V_k^*(\mathbf{s})$ represents the maximal probability of reaching the target set while avoiding unsafe states when starting from state $\mathbf{s}$ at time k. The associated optimal policy specifies control actions that maximize the probability of safely reaching the target within the given horizon.

This dynamic programming formulation provides both a quantitative measure of performance (the reach-avoid probability) and a constructive method for synthesizing optimal control policies, forming the theoretical foundation for reach-avoid analysis in stochastic hybrid systems and Markov decision processes.

4 Problem Formulation

4.1 Definition of the system: quality relevant attributes as state variable

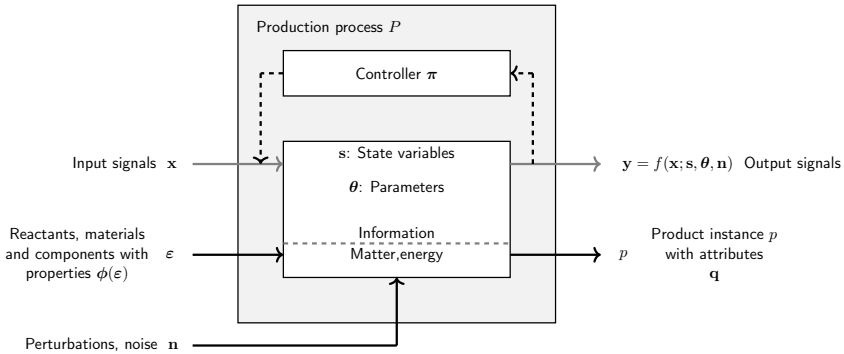


Figure 4.1: An abstract view of an idealized closed-loop automated production process. The vectors $\mathbf{x}$, $\mathbf{y}$, $\mathbf{n}$, and $\mathbf{s}$ are time-dependent signals [3].

Following [3], figure 4.1 illustrates an idealized closed-loop automated production process P. It is made up of two main parts. One part shows the process variables and process dynamics, the input signal $\mathbf{x}$, noise $\mathbf{n}$ that might occur in the system, the state variables $\mathbf{s}$ of the dynamic system, the parameters of the system θ, the controller π and the output signal $\mathbf{y}$ which is a function of $\mathbf{x}, \mathbf{s}, \theta$, and $\mathbf{n}$. The signals $\mathbf{x}$ and $\mathbf{y}$ are defined to be observable and $\mathbf{s}$ subsumes all unobservable dynamic quantities. If an added sensor or actuator instrumentation allows the components of $\mathbf{s}$ to be determined passively or actively, these quantities are assigned to be additional components of $\mathbf{x}$ and $\mathbf{y}$ in the next production process P. The other part illustrates the input process material $\boldsymbol{\varepsilon}$ such as reactants and components with properties ϕ and its flow to the product p with quality relevant attributes $\mathbf{q}(p)$. Time-dependent signal ν_t is defined that describes the evolution of the product quality relevant attributes during the process duration $t \in [0, T]$. Depending on the level of instrumentation, ν_t may be observable during the

process (e.g., through inline measurements or soft-sensing) or unobservable, where only its final value at the end of the process is available. We denote the quality relevant attributes at the end time point of the production process by $\mathbf{nu}_T = \mathbf{q}$.

4.2 Continuous-time stochastic dynamics and discrete-time Markov Decision Process

The system is subject to stochastic disturbances, reflecting uncertainty and variability of the production process. To account for these disturbances, we model the system as a continuous-time stochastic system:

$$\dot{\mathbf{s}}(t) = g(\mathbf{s}(t), \mathbf{x}(t)) + \mathbf{n}(t), \tag{4.1}$$

where $\mathbf{s}(t) \in \mathbb{R}^n$ denotes the system state vector, $\mathbf{x}(t) \in \mathbb{R}^m$ is the control input, and $\mathbf{n}(t) \in \mathbb{R}^n$ represents stochastic disturbances.

For the purpose of this work, we focus on the reach–avoid problem for Markov Decision Processes (MDPs) [4]. More precisely, we consider a discrete-time controlled stochastic process

$$\mathbf{s}_{t+1} \sim Q(\mathbf{s} \mid \mathbf{s}_t, \mathbf{x}_t), \quad (\mathbf{s}_t, \mathbf{x}_t) \in \mathcal{S} \times \mathcal{X}, \tag{4.2}$$

where Q denotes the transition kernel that captures the probabilistic evolution of the state $\mathbf{s}_t$ under the control input $\mathbf{x}_t$. In practice, the kernel Q may be obtained from a first-principles physical model, from data-driven system identification, simulation, or, in partially observed settings, from a Partially Observable Markov Decision Process (POMDPs) model whose belief dynamics induce such a transition kernel [13]. The state space $\mathcal{S} \subseteq \mathbb{R}^n$ and the control space $\mathcal{X} \subseteq \mathbb{R}^m$ are both continuous.

This formulation defines a Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{X}, Q)$, where $\mathcal{S}$ denotes the state space comprising all possible states $\mathbf{s}_t$, $\mathcal{X}$ is the action space of admissible control inputs $\mathbf{x}_t$, $Q(\mathbf{s} \mid \mathbf{s}_t, \mathbf{x}_t)$ represents the transition kernel describing the stochastic evolution of the system.

The MDP satisfies the Markov property: the next state $\mathbf{s}_{t+1}$ depends only on the current state $\mathbf{s}_t$ and control $\mathbf{x}_t$, and not on past history. This probabilistic

formulation enables the application of Approximate Dynamic Programming (ADP) for near-optimal control under uncertainty [14, 4, 7].

4.3 Controller objective: Stochastic reachability

Following the definition of *Supervisability* introduced in [3], and considering that the ultimate objective of the production process is to achieve the desired product quality, the goal of the controller design is to obtain an optimal policy that maximizes the probability that the final product quality lies within its desired range.

The *Supervisability* S can be used to quantify the maturity of a production process during the development and optimisation of a new process based on the probability that the quality relevant attributes of the produced item $\mathbf{q}(p)$ fall within the target quality set $\mathcal{K}$, given that the controller π and all process variables remain within their admissible sets. The mathematical formulation of the Supervisability is given in [3] as:

$$S(P_i^j) := \Pr(\mathbf{q} \in \mathcal{K} \mid \mathbf{x} \in \mathcal{X}_{\mathrm{adm}},\, \mathbf{s} \in \mathcal{S}_{\mathrm{adm}},\, \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\mathrm{adm}},\, T < T_{\max}), \quad (4.3)$$

where $\mathcal{X}_{\mathrm{adm}} \subseteq \mathbb{R}^m$, $\mathcal{S}_{\mathrm{adm}} \subseteq \mathbb{R}^n$, and $\boldsymbol{\Theta}_{\mathrm{adm}} \subseteq \mathbb{R}^p$ denote the admissible sets for $\mathbf{x}$, $\mathbf{s}$, and $\boldsymbol{\theta}$, respectively, and the throughput time T necessary for producing a product instance does not exceed $T_{\max}$.

5 Summary and Outlook

This work investigated the quality-oriented control framework for manufacturing systems operating under uncertainty, in which quality-relevant attributes are investigated to be embedded into the system state and regulated throughout the production process. By investigating the control objective as a finite-horizon stochastic reach–avoid problem, the task of quality assurance is expressed directly in probabilistic terms. The resulting value function quantifies the likelihood of achieving in-spec product quality while respecting operational constraints, providing a meaningful performance measure that aligns naturally with industrial

acceptance criteria. Approximate Dynamic Programming was employed to enable tractable computation of near-optimal control policies in continuous state and input spaces.

As an outlook, future work will focus on addressing the quality-oriented control problem in greater detail, including the systematic design of the controller and its implementation on a concrete production system. This involves mapping physical process variables and quality-relevant attributes to an appropriate Markov Decision Process representation, constructing realistic target and safety sets, and analyzing the impact of model uncertainty and approximation errors on control performance. Further extensions will consider partial observability of quality attributes and integration with existing industrial control architectures, with the goal of evaluating the feasibility and practical benefits of reachability based quality control in real world manufacturing environments.

References

[1] Alessandro Abate et al. “Probabilistic reachability and safe sets computation for discrete time stochastic hybrid systems”. In: *Proceedings of the 45th IEEE Conference on Decision and Control*. IEEE. 2006, pp. 258–263.

[2] Kehinde Afolabi et al. “Stochastic Optimization of Quality Assurance Systems in Manufacturing: Integrating Robust and Probabilistic Models for Enhanced Process Performance and Product Reliability”. In: *Journal of Manufacturing and Materials Processing* 9.8 (2025), p. 250.

[3] Negar Arabizadeh, Julius Pfrommer, and Jürgen Beyerer. “How to quantify the maturity of production processes”. In: *Machine Learning for Cyber Physical Systems* (2025), p. 79.

[4] Dimitri Bertsekas. *Dynamic programming and optimal control: Volume I*. Vol. 4. Athena scientific, 2012.

[5] Marcello Farina, Luca Giulioni, and Riccardo Scattolini. “Stochastic linear model predictive control with chance constraints–a review”. In: *Journal of Process Control* 44 (2016), pp. 53–67.

[6] Tuomas Haarnoja et al. “Reinforcement learning with deep energy-based policies”. In: *International conference on machine learning*. PMLR. 2017, pp. 1352–1361.

[7] Nikolaos Kariotoglou et al. “Approximate dynamic programming for stochastic reachability”. In: *2013 European Control Conference (ECC)*. IEEE. 2013, pp. 584–589.

[8] Johannes Köhler et al. “A computationally efficient robust model predictive control framework for uncertain nonlinear systems”. In: *IEEE Transactions on Automatic Control* 66.2 (2020), pp. 794–801.

[9] Runze Lin et al. “Facilitating Reinforcement Learning for Process Control Using Transfer Learning: Overview and Perspectives”. In: *2025 37th Chinese Control and Decision Conference (CCDC)*. IEEE. 2025, pp. 1699–1704.

[10] James Blake Rawlings, David Q Mayne, Moritz Diehl, et al. *Model predictive control: theory, computation, and design*. Vol. 2. Nob Hill Publishing Madison, WI, 2020.

[11] John Schulman et al. “Proximal policy optimization algorithms”. In: *arXiv preprint arXiv:1707.06347* (2017).

[12] Max Schwenzer et al. “Review on model predictive control: An engineering perspective”. In: *The International Journal of Advanced Manufacturing Technology* 117.5 (2021), pp. 1327–1349.

[13] David Silver and Joel Veness. “Monte-Carlo planning in large POMDPs”. In: *Advances in neural information processing systems* 23 (2010).

[14] Sean Summers and John Lygeros. “Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem”. In: *Automatica* 46.12 (2010), pp. 1951–1961.

[15] Kaishu Xia et al. “A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence”. In: *Journal of Manufacturing Systems* 58 (2021), pp. 210–230.

A Note on Image Resolution in Adversarial Patch Attacks and Defenses

Jens Bayer

Fraunhofer Institute of
Optronics, System Technologies and Image Exploitation (IOSB)
Ettlingen, Germany
jens.bayer@iosb.fraunhofer.de

Abstract

Adversarial patches pose a practical security risk for object detectors, especially in settings that require reduced input resolutions. This report analyzes the impact of varying input resolutions on adversarial patch effectiveness and simple defenses in one-stage detection. Using YOLOv10n on $\text{COCO}_{\text{Person}}$, clean performance, patch attacks with varying relative sizes, and adversarial training is evaluated on five resolution stages (96-640px). $\text{AP}_{@[.5:.95]}$ declines with downscaling; under attack (p_{box}=1), AP drops to $\approx 0.07 \pm 0.02$ at 75% patch size. Adversarial training improves robustness to larger patches but can reduce clean and small-patch performance. Overall, results reveal resolution-sensitive trade-offs between accuracy, attack surface, and robustness.

1 Introduction

Deep learning based object detectors have achieved strong performance in recent years, yet they remain vulnerable to adversarial manipulations. Among these, adversarial patches are particularly concerning because they are easy to manufacture and deploy in the physical world and can reliably cause misdetections

in real-time systems [17]. Unlike pixel-wise adversarial examples that rely on imperceptible, high-frequency perturbations [16, 7], adversarial patches are localized, visible patterns that can be placed on detection targets such as persons or objects, making them a practical security risk for deployed computer vision pipelines [20].

In parallel, deployment constraints often require operating at reduced input resolutions to meet latency, power, or bandwidth budgets. Lower resolutions can degrade small-object recognition but may also modulate the effectiveness of adversarial patches by changing the relative scale and appearance of both objects and patches. Despite its practical relevance, the interplay between input resolution, patch size, and detector robustness has received limited systematic analysis, especially for modern one-stage detectors.

This report investigates how input image resolution influences the success of adversarial patch attacks and the effectiveness of simple defenses. The study focuses on person detection in COCO [12] with YOLOv10 [18] detectors using the Ultralytics implementation [8]. Patches are optimized against the trained detectors following Thys et al. [17] and applied inside ground truth boxes with relative sizing. The following is evaluated: (i) clean detection performance across multiple input resolutions, (ii) detection under patch attacks for varying relative patch sizes, and (iii) adversarial training with patched samples to harden the models. All results are reported as average precision ($AP_{@[.5:.95]}$) with a single class. This equals mean average precision ($mAP_{@[.5:.95]}$) [12].

2 Related Work

Adversarial examples reveal that modern vision models are vulnerable to small, targeted perturbations [16, 7]. In contrast to imperceptible, pixel-wise attacks, adversarial patches are localized, visible patterns that can be deployed in the physical world with manageable effort. Brown et al. [4] introduces universal patches that transfer across scenes, while Karmon et al. [9] show strong localized visible noise. Physical-world viability has been demonstrated for both classifiers [6] and detectors [17, 21, 20, 19].

Robust patch optimization commonly accounts for transformations to bridge the simulation-to-real gap [2]. For detectors, successful setups typically target objectness and class logits [17, 5].

A standard defense is adversarial training, which augments training with attacked samples to improve robustness [10, 13, 1]. For object detection and adversarial patches, this entails randomized patch placement inside ground truth boxes with stochastic transformations to reduce overfitting and to expose the model to diverse occlusions [3]. Detection performance is tightly linked to scale. Feature Pyramid Networks improve multi-scale representations [11], and scale-normalizing training with image pyramids mitigates train-test scale mismatch [15]. One-stage detectors such as YOLO [14] rely on multi-scale training and stride-specific heads, making performance sensitive to input resolution and relative object size.

However, the interaction between input resolution, relative patch size, and robustness of modern one-stage detectors remains underexplored. Therefore, this work provides a systematic analysis with $\text{COCO}_{\text{Person}}$ [12] used as a representative dataset and YOLOv10 [18] selected as the object detector.

3 Fundamentals

Despite the improved performance of deep neural networks lately, adversarial attacks are still an unresolved issue that can potentially harm users when models are deployed without safety features [1]. The kind of attacks that are used in this study are located in the computer vision domain and are called adversarial patches. These patches are optimized in such a way that they perform an evasion attack. In comparison to adversarial examples that modify input images by inducing high-frequency information [16, 7], adversarial patches are not only a digital world threat [19]. The comparably simple implementation in the physical world [17] makes these carefully optimized patches a real-life threat for already deployed computer vision models.

3.1 Adversarial Patch Attacks

This study follows the optimization procedure of adversarial patches to fool object detectors in the physical world of Thys et al. [17]. First, a dataset $\mathcal{D}$ containing images of objects of interest is selected. After this, a trained object detector f that serves as a surrogate model is chosen. During optimization, the patch $\mathbf{P} \in \mathbb{R}^{w \times h \times 3}$ to be trained is applied to each image $\mathbf{I} \in \mathcal{D}$ by inserting an augmented copy $\hat{\mathbf{P}}$ of the patch within the ground truth bounding boxes of objects of interest. Afterward, the modified image $\hat{\mathbf{I}}$ is propagated through the detector, and the regressed confidence scores $\mathbf{c}$ of the bounding boxes for the specific objects are used to calculate the *objectness loss*

$$l_{obj} = \|\mathbf{c}\|_\infty \tag{3.1}$$

where c is the detector bounding-box confidence before sigmoid activation and non-maximum-suppression.

Furthermore, a *smoothness* loss

$$l_{smt} = \|\nabla_x \mathbf{P}\|_1 + \|\nabla_y \mathbf{P}\|_1 \tag{3.2}$$

and a *validity loss*

$$l_{val} = \|\mathbf{P} - clamp(0, 1, \mathbf{P})\|_2 \tag{3.3}$$

where

$$clamp(l, u, x) = \begin{cases} l & \text{if } x < l \\ u & \text{if } x > u \\ x & \text{else} \end{cases} \tag{3.4}$$

are used to regularize the training procedure by penalizing high-frequency noise and keeping values of the patch in $[0, 1]$. The final optimization loss

$$l_{total} = \lambda_{obj} l_{obj} + \lambda_{smt} l_{smt} + \lambda_{val} l_{val} \tag{3.5}$$

is the weighted sum of the objectness, smoothness, and validity loss and is minimized with a gradient-based optimization method of choice.

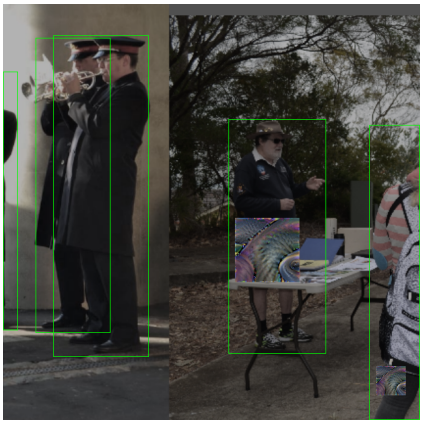


Figure 3.1: Example image as used during adversarial training after the augmentation pipeline. The ground truth bounding boxes are drawn in green. The added adversarial patches are located at the person in the center and on the bottom right below the backpack.

3.2 Adversarial Training

The most naive way to harden a deep neural network against adversarial attacks is by including possible instances of these attacks in the training data [7]. To harden an object detector against adversarial patches, first, a set of patches $\mathcal{P}$ is generated as described in section 3.1. Afterward, the augmentation pipeline used in the network training procedure is extended by altering the images using the ground truth bounding box information and the trained adversarial patches (see Figure 3.1). For each bounding box in an image, a random patch $\mathbf{P} \in \mathcal{P}$ is selected, augmented randomly, and applied at a random position inside the bounding box. Since covering each bounding box would lead to a performance drop when no adversarial patches are present, an additional hyperparameter π controls the probability of a bounding box actually containing an adversarial patch [3].

4 Experimental Setup

In the following section, the experimental setup of the conducted experiments is presented. In total, three different experiments are evaluated: the first examines the impact on the detector performance when the input image resolution is successively reduced. For the second experiment, the trained networks are attacked with adversarial patches. As the size of an introduced adversarial patch in an image has a crucial impact on the performance of the detector, various patch sizes are evaluated. Ultimately, the networks are trained adversarially to improve their robustness against adversarial patch attacks. The last experiment evaluates the performance of the hardened networks when attacked. For all experiments, the mean average precision (mAP) using multiple intersection over union (IoU) thresholds [12] is used to measure the performance of a model. Since only one class is evaluated, the class mean collapses to the class AP; hence the $AP_{@[.5:.95]}$ is reported.

4.1 Dataset

For the conducted experiments, subsets of the train and validation split of COCO [12] are used. As the most prominent class in COCO is *Person*, only images containing this class are selected. The resulting train split consists of a total of 64 115 images and 262 464 instances of *Person*. The validation split is used for the evaluations and consists of a total of 2 693 images and 11 004 instances of *Person*. Unmodified COCO images vary in size. To address this, the images of the original dataset are resized in such a way that the longer side of an image equals 640/480/320/160/96 px, respecting the aspect ratio. The corresponding ground truth bounding boxes are adjusted accordingly.

4.2 Detector

The detector used for the experiments is of the YOLOv10 [18] architecture family with the implementation provided by Ultralytics [8]. To be more precise, YOLOv10n is examined. The detector is the smallest of the YOLOv10

family with the lowest number of parameters and thus the most performant one in terms of inference speed. For the experiments, all models are trained from scratch over 100 epochs using stochastic gradient descent (SGD), an initial learning rate of 0.01, and the default augmentation pipeline provided by Ultralytics. The input image size is adjusted according to the investigated image sizes (640/480/320/160/96 px). Moreover, multi-scale training is disabled.

5 Evaluation

5.1 Experiment 1: Clean Detection Performance

The performance of the trained YOLOv10n models for the respective different image resolutions is given in Figure 5.1. The plot shows the average precision for different bounding box area thresholds. A threshold of 10% for, e.g., the 640 px curve means that bounding boxes with an area smaller than 10% of the image area are filtered and left out of the ground truth. This is because the area of more than 5% ($n = 15\,544$) of the official ground truth bounding boxes for the person class of COCO is smaller than 64 px. When resizing the dataset, these already small bounding boxes become even smaller and thus increase the difficulty of the dataset drastically. By filtering these boxes, the performances of the models peaks at around 1-5%. The corresponding thresholds of the peaks (see Table 5.1) are used for all further experiments.

The results of Figure 5.1 and Table 5.1 highlight that a lower image resolution, without a proper adjustment of the ground truth bounding boxes, results in a reduced detector performance. Moreover, downscaling of the images by a factor of 25, 50, 75, and 85% leads to successively raising performance drops of 1.6, 8.2, 23.0, and 37.7% regarding the 640px image resolution.

5.2 Experiment 2: Detection Performance under Attack

For the second experiment, a total of three adversarial patches per model are optimized. As a training dataset, the COCO training split is used. The weights for the loss are set to $\lambda_{obj} = 1$, $\lambda_{smt} = 2$, and $\lambda_{val} = 1$. Each patch is optimized

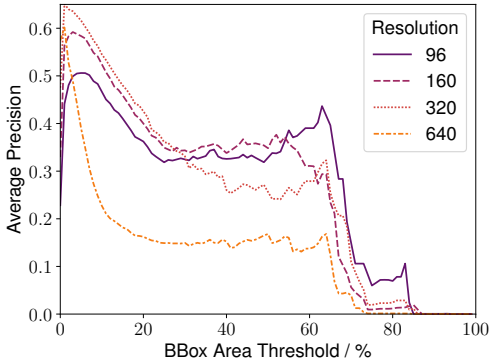


Figure 5.1: Experiment 1: Detector performance of the trained YOLOv10n models for different bounding box area thresholds. A threshold of 10% filters the ground truth bounding boxes with an area smaller than 10% of the respective image.

Resolution / px	Threshold / %	AP	rel. AP_{640}	unfiltered AP
640	1	0.61	1.00	0.50
480	1	0.60	0.98	0.45
320	1	0.56	0.92	0.38
160	3	0.47	0.77	0.24
96	5	0.38	0.62	0.16

Table 5.1: Performance of the trained detectors at the optimal bounding box area threshold. rel. AP_640 is the performance of the respective detector, relative to the detector with a resolution of 640 px. Unfiltered AP refers to the performance without any ground truth bounding box filtering.

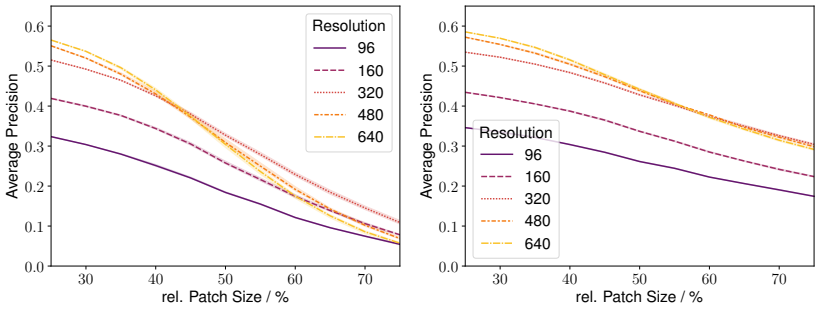
over 5 epochs using AdamW with an initial learning rate of 0.01 and a learning rate reduction of factor 10 per epoch. The patches are initialized with random values in $[0, 1]$. During optimization, a patch is randomly resized to match 75-100% of the shorter side of the bounding box it is placed into at a random location. The augmentation pipeline for the patches is color jitter, followed by random rotation in $[-30, 30]$ degrees and a random perspective transformation.

The base resolution of each optimized patch is equal to 40% of its corresponding input image resolution, resulting in patches of sizes 256/192/128/64/39 px.

For the evaluation, the trained patches of a detector are placed at the center of the ground truth bounding boxes of the test set for a given probability of p_{box}. In addition, with a chance of p_{hal}, additional patches are hallucinated and placed at random positions in the image. The maximum possible number of hallucinated patches equals the number of ground truth boxes of an image. Moreover, the patches can occur at any position in the image and thus overlap with ground truth bounding boxes and already applied patches. The resize range of the patches is given by the size of the smallest and largest regular patch.

The plots in Figure 5.2 show the detectors performances when attacked in two different parameterization settings. The x-axis of the plots refers to the relative patch size with respect to the corresponding ground truth bounding box. A relative size of, e.g., 25% means that the patch width and height are equal to 25% of the shorter side of the respective ground truth bounding box. Regarding the two different parameterizations, each ground truth bounding box in Figure 5.2(a) contains a patch ($p_{\text{box}} = 1$), and there are no additional hallucinations ($p_{\text{hal}} = 0$). For Figure 5.2(b), patches are hallucinated with a chance of $p_{\text{hal}} = 0.5$ and there is only a $p_{\text{box}} = 0.5$ chance that a ground truth bounding box contains a patch. The random seed is shared for all evaluations to enable a fair comparison of the results.

As expected, the different probability values of p_{box} and p_{hal} have a significant impact on the resulting average precision values. If all ground truth bounding boxes contain a patch, the average precision for all models drops to 0.07 ± 0.02 at a relative patch size of 75%. If only half of the boxes contain a patch and the chances of additional hallucinated patches are set to 50%, the drop is far less and much more distinct between the models. In addition, the larger the relative patch size, the higher the average precision drop is. Interestingly, for both parameterization settings, the 640, 480, and 320 models behave similarly until relative patch sizes of 45% and 55%, where the 320 model surpasses the larger models in terms of the achieved average precision values. A possible reason could be the stride values of the detector necks that are not adjusted according to the reduced image resolution.



(a) $p_{box} = 1, \quad p_{hal} = 0$ (b) $p_{box} = 0.5, \quad p_{hal} = 0.5$

Figure 5.2: Experiment 2: Detector performance of the trained YOLOv10n models for different relative patch sizes. A relative patch size of 25% means, that the width and height of a patch is equal to 25% of the shorter side of the corresponding bounding box. For (a), each ground truth bounding box contains a patch. For (b) there is a $p_{box} = 0.5$ chance for a box to contain a patch and a $p_{hal} = 0.5$ chance to contain additional patches at random positions.

5.3 Experiment 3: Hardened Detection Performance

The hardened detectors share the same training hyperparameters as the corresponding baseline models. The augmentation pipeline has been adjusted as described in section 3.2. In addition, the network weights are no longer randomly initialized and instead use the baseline weights for initialization. The probability that an adversarial patch is present is $\pi = 0.05$, and the random resize range for the patches is set to $[0.4, 0.8]$. For the evaluation, the same parameterization as in Figure 5.2(b) is used.

The results of the hardened detection performance are presented in Figure 5.3. To further provide a quantitative comparison of the results, Table 5.2 shows the area under the curves of $AP_{@[.5:.95]}$ values for the presented thresholds (0.25 – 0.75). Model order refers to either the non-hardened models (Model Order = 0) or hardened models (Model Order = 1). At first glance, the overall performance seems to increase for every model, regardless of the relative patch size. This observation is due to the fact that the presented curves are less steep with a higher

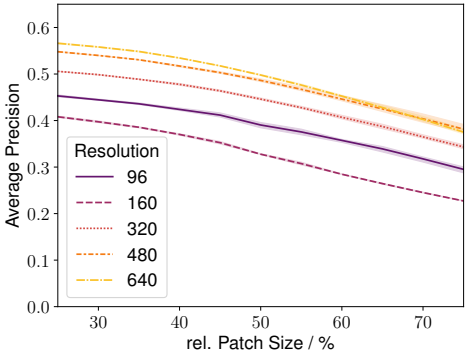


Figure 5.3: Experiment 3: Hardened detector performance for different relative patch sizes. There is a $p_{box} = 0.5$ chance for a box to contain a patch and a $p_{hal} = 0.5$ chance to contain additional patches at random positions.

AP for larger relative patch sizes. At a second, more detailed glance, one notices that the performances of the models for smaller relative patch sizes of, e.g., 25% are actually lower than the unhardened models. For the 160 model, adversarial training even reduced the AP-AUC while the 96 model has a drastically increased AP curve. A possible reason for the latter is the introduced occlusion of the patches during training that forces the model to focus on more relevant features of the target class.

6 Discussion

Figure 5.1 shows the expected $AP_{@[.5:.95]}$ reduction with decreasing input resolution. Figure 5.2 confirms that patch effectiveness increases sharply with relative patch sizes. At the same time, differences between models remain small for small patches and become more prominent as patch size increases. Regarding adversarial training, Figure 5.3 and Table 5.2 reveal a strengthening of the models against large patches, but slight losses for small patch sizes. The smallest resolution seems to benefit the most from the induced occlusion. Thus, the overall flattening of the curves indicates an increased use of more context-stable

Resolution	Model Order	AP-AUC	rel.$_{640}$ AP-AUC
96	0	0.13 ± 0.00	0.59
160	0	0.17 ± 0.00	0.77
320	0	0.20 ± 0.01	0.91
480	0	0.22 ± 0.00	1.00
640	0	0.22 ± 0.00	1.00
96	1	0.19 ± 0.00	0.79
160	1	0.16 ± 0.00	0.67
320	1	0.22 ± 0.00	0.92
480	1	0.24 ± 0.00	1.00
640	1	0.24 ± 0.00	1.00

Table 5.2: Area under the AP$_{@[.5:.95]}$ vs. rel. patch size curves of Figure 5.2(b) and Figure 5.3. The higher the AP-AUC, the more robust is the respective model. Rel.$_{640}$ AP-AUC is the AP-AUC relative to the respective 640 px model.

features. The presented results should be interpreted with care, as the evaluation uses resolution-specific ground truth bounding boxes (see Table 5.1). Also, the strides of the feature pyramid network are not adjusted according to the image input resolution. As these have a significant impact on the resulting feature maps of the detector, adjustments could improve the lower resolution detectors.

7 Conclusion

The results show resolution-sensitive trade-offs between accuracy, attack surface, and robustness. Adversarial training increases robustness against large patches but is accompanied by slight losses in clean and small patch performance. For practical use, the input scale and hardening should be selected according to the latency and security requirements. Further work should include multi-class scenarios, a more diverse range of models, and a per-stride analysis.

References

[1] Naveed Akhtar and Ajmal Mian. "Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey". In: *IEEE Access* 6 (2018). ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2807385.

[2] Anish Athalye et al. "Synthesizing robust adversarial examples". In: *International Conference on Machine Learning*. 2018.

[3] Jens Bayer et al. "Traversing the subspace of adversarial patches". In: *Machine Vision and Applications* 36.3 (2025). ISSN: 1432-1769. DOI: 10.1007/s00138-025-01689-6.

[4] Tom B Brown et al. "Adversarial patch". In: *arXiv:1712.09665* (2017).

[5] Shang-Tse Chen et al. "Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. 2018.

[6] Ranjie Duan et al. "Adversarial Camouflage: Hiding Physical-World Attacks With Natural Styles". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2020. ISBN: 978-1-7281-7168-5. DOI: 10.1109/CVPR42600.2020.00108.

[7] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *CoRR* abs/1412.6572 (2014).

[8] Glenn Jocher, Jing Qiu, and Ayush Chaurasia. *Ultralytics YOLO*. Version 8.0.0. Jan. 2023. URL: https://github.com/ultralytics/ultralytics.

[9] Danny Karmon, Daniel Zoran, and Yoav Goldberg. "Lavan: Localized and visible adversarial noise". In: *International conference on machine learning*. PMLR. 2018.

[10] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. "Adversarial Machine Learning at Scale". In: *International Conference on Learning Representations*. 2017.

[11] Tsung-Yi Lin et al. "Feature Pyramid Networks for Object Detection". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017. DOI: 10.1109/CVPR.2017.106.

[12] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014.

[13] Aleksander Madry et al. “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *International Conference on Learning Representations*. 2018.

[14] Joseph Redmon and Ali Farhadi. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

[15] Bharat Singh et al. “Scale Normalized Image Pyramids With AutoFocus for Object Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.7 (2022). DOI: 10.1109/TPAMI.2021.3058945.

[16] Christian Szegedy et al. “Intriguing properties of neural networks”. In: *CoRR* abs/1312.6199 (2013).

[17] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. “Fooling automated surveillance cameras: adversarial patches to attack person detection”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2019.

[18] Ao Wang et al. “YOLOv10: Real-Time End-to-End Object Detection”. In: *Advances in Neural Information Processing Systems*. Vol. 37. 2024. DOI: 10.52202/079017-3429.

[19] Hui Wei et al. “Physical Adversarial Attack Meets Computer Vision: A Decade Survey”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.12 (2024). ISSN: 19393539. DOI: 10.1109/TPAMI.2024.3430860.

[20] Xingxing Wei, Jie Yu, and Yao Huang. “Physically adversarial infrared patches with learnable shapes and locations”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023.

[21] Zuxuan Wu et al. “Making an Invisibility Cloak: Real World Adversarial Attacks on Object Detectors”. In: *Lecture Notes in Computer Science* 12349 LNCS (2020). ISSN: 16113349. DOI: 10.1007/978-3-030-58548-8_1.

Probabilistic Quality Control Without Tears

Ali Darijani

Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
ali.darijani@kit.edu

1 Abstract

Quality control is often associated with strict rules, detailed calculations, and intimidating statistical language. This perception discourages many practitioners from fully engaging with probabilistic methods, even though they already rely on them implicitly. This report presents probabilistic quality control in a concept-driven, non-mathematical way. By focusing on intuition, examples, and practical decision-making, it demonstrates how probability helps manage uncertainty rather than complicate it. The aim is to make probabilistic quality control accessible, usable, and free from unnecessary anxiety.

2 Introduction

Quality control exists to answer a deceptively simple question: are we producing what we think we are producing? In practice, this question is never answered with complete certainty. Products are manufactured in large numbers, services are delivered repeatedly, and processes evolve over time. No matter how well designed a system is, variation and uncertainty are unavoidable.

Traditional views of quality control often imply absolute judgments: pass or fail, good or bad, acceptable or unacceptable. While such decisions are sometimes

necessary, they hide an important truth. Most quality decisions are made with incomplete information. Probabilistic quality control acknowledges this openly.

Rather than pretending uncertainty does not exist, probabilistic approaches provide a structured way to think about it. This report explains why uncertainty is unavoidable, how probability naturally arises in quality control, and how probabilistic thinking improves decisions without requiring advanced mathematics.

3 The Reality of Imperfect Information

In an idealized world, every product would be inspected, every measurement would be precise, and every process would behave identically from one moment to the next. Real-world systems do not work this way.

Inspection itself has limits. Human inspectors get tired, automated systems drift, and measurement tools have tolerances. Even when every item is inspected, errors still occur. This means that 100% inspection does not eliminate uncertainty; it only changes its form.

Furthermore, quality information is always historical. Decisions about the present and future are based on data collected in the past. Probabilistic reasoning helps bridge this gap by asking how much confidence past observations give us about what is happening now.

4 Why Probability Is Unavoidable

Probability appears in quality control the moment we stop demanding certainty. This happens almost immediately in any real process.

Consider a situation where a supervisor reviews a handful of finished products and says, ‘‘Everything looks fine.’’ This statement already implies probability. It means that, based on what was seen, serious problems are unlikely. The supervisor may not use numerical language, but the reasoning is probabilistic nonetheless.

Probabilistic quality control does not introduce uncertainty; it makes existing uncertainty visible and manageable. By doing so, it reduces surprises and supports more consistent decision-making.

5 Sampling as a Practical Compromise

Sampling is often misunderstood as a shortcut or a cost-cutting measure. In reality, it is a thoughtful compromise between perfect knowledge and practical constraints.

A sample provides partial information about a larger group. The key insight is that partial information can still be extremely useful if it is collected systematically. Random or representative sampling avoids biases that creep into informal checks.

Importantly, sampling plans encode expectations. They reflect how much risk is acceptable, how costly errors are, and how stable the process is believed to be. These considerations are engineering and business judgments first, and technical details second.

6 Measurement as a Managed Process

Measurement is often treated as a neutral window into reality, but in quality control it is better understood as a process in its own right. Like any process, measurement has limitations, variability, and failure modes. Ignoring this fact can lead to false confidence and poor decisions.

Every measurement system involves instruments, operators, procedures, and environmental conditions. Each of these elements influences the result. If measurement is assumed to be perfectly accurate, quality decisions inherit that assumption and amplify its consequences. In practice, measurement quality must be actively managed rather than passively trusted.

Managing measurement means understanding how results are produced, how repeatable they are, and how sensitive they are to external factors. It also means

recognizing that a measurement result is not a single unquestionable truth, but an observation shaped by the system that produced it. Probabilistic quality control naturally accommodates this view by treating measurements as evidence rather than verdicts.

When measurement is acknowledged as part of the system, quality control becomes more robust. Decisions are less brittle, and surprising failures become easier to explain and prevent.

7 Device Tolerance and the Illusion of Sharp Boundaries

All measurement devices have tolerances. This is not a flaw; it is a fundamental property of physical systems. However, tolerance is often forgotten once numbers appear on a screen or in a report.

A device tolerance means that two items near a specification boundary may be practically indistinguishable, even if one is labeled inside and the other outside. Treating such classifications as absolute introduces artificial precision into the quality process. The result is a sharp boundary that exists on paper but not in reality.

Probabilistic quality control encourages a softer interpretation. Instead of assuming that a measured value perfectly represents the product, it asks how confident we can be in that interpretation given the device tolerance. This shift does not weaken quality standards; it makes them more honest.

By explicitly accounting for measurement tolerance, organizations avoid overreacting to borderline results and underestimating uncertainty near decision thresholds.

8 From Geometric Rules to Decision Categories

A common approach in quality control is to define an acceptable region using fixed limits on each measured characteristic. Conceptually, this creates a mul-

tidimensional box within which products are considered acceptable. The next step is often to estimate how much of the production falls inside this region and label the result using categories such as good, average, or bad.

This approach is appealing because it feels concrete and visual. However, it hides several assumptions. First, it assumes that all dimensions are equally important and independent. Second, it assumes that crossing a boundary instantly changes the quality status in a meaningful way. Third, it assumes that the chosen categories reflect real differences in outcome or risk.

Probabilistic quality control does not reject this structure, but it reframes it. The acceptable region becomes a model rather than a law. The resulting classifications become aids to decision-making rather than definitive judgments.

When probability is used thoughtfully, categories such as good or bad are understood as summaries of risk, not as absolute truths. This perspective makes decision-making more flexible and more aligned with real-world consequences.

9 Confidence Without Guarantees

One of the emotional challenges of probabilistic quality control is letting go of guarantees. People naturally prefer certainty, even when it is illusory.

Probabilistic thinking replaces guarantees with confidence levels. Instead of claiming that a batch is perfect, we claim that it is unlikely to be problematic given the available evidence. This shift may feel uncomfortable at first, but it aligns much better with reality.

Over time, organizations that adopt this mindset tend to make calmer decisions. They respond to evidence rather than fear, and they accept that uncertainty can be managed without being eliminated.

10 Risk-Based Decision Making

Every quality decision involves trade-offs. Rejecting products costs money and time. Accepting defects damages trust and reputation. Probabilistic quality control provides a framework for balancing these competing risks.

By explicitly considering consequences, organizations can tailor quality strategies to their priorities. A low-cost consumer product may tolerate more risk than a safety-critical component. Probability supports this differentiation without moral judgment.

This approach also makes quality policies easier to explain. When decisions are tied to clearly stated risks, they feel less arbitrary and more fair.

11 The Role of Experience and Expertise

Probabilistic tools do not replace experience; they amplify it. Experienced practitioners have an intuitive sense of risk, variation, and warning signs. Probability provides language and structure for that intuition.

When expert judgment and probabilistic reasoning work together, decisions become both informed and defensible. This combination is especially valuable in complex or high-stakes environments.

Importantly, probabilistic quality control respects uncertainty rather than denying it. This makes room for learning and adaptation over time.

12 Common Sources of Resistance

Resistance to probabilistic methods often comes from misunderstanding rather than disagreement. Some fear loss of control, others fear loss of accountability.

In reality, probabilistic quality control increases accountability by making assumptions explicit. Decisions are no longer hidden behind rigid rules or vague impressions. Instead, they are grounded in stated levels of confidence and risk.

Education and clear communication are usually enough to overcome resistance. Once people see that probability simplifies decisions rather than complicates them, acceptance follows naturally.

13 Probabilistic Thinking as a Cultural Shift

Adopting probabilistic quality control is as much a cultural change as a technical one. It encourages curiosity over blame and learning over punishment.

When variation is expected rather than feared, teams focus on understanding systems instead of reacting emotionally to every deviation. This shift improves morale as well as performance.

Over time, probabilistic thinking becomes second nature. Decisions feel less stressful because uncertainty is acknowledged rather than ignored.

14 Conclusion

Probabilistic quality control is often misunderstood as a collection of abstract tools layered on top of an otherwise precise system. In reality, it responds to a much more fundamental condition: quality decisions are always made using imperfect information. Sampling, variation, and uncertainty are not technical inconveniences; they are intrinsic features of real processes.

This report has emphasized that uncertainty enters quality control long before any formal analysis takes place. Measurement itself is a process that must be managed, not a neutral observer of reality. Devices have tolerances, operators introduce variability, and results near specification boundaries are inherently ambiguous. Treating such measurements as exact truths creates an illusion of certainty that the system cannot support.

Similarly, common decision practices based on sharply defined acceptance regions and categorical labels provide comfort but hide assumptions. Defining geometric limits and classifying outcomes as good, average, or bad may simplify communication, but it risks turning gradual changes in risk into abrupt and

misleading decisions. Probabilistic thinking reframes these constructions as models and aids, not as absolute laws.

Rather than eliminating uncertainty, probabilistic quality control makes it visible and manageable. It replaces rigid guarantees with informed confidence, and binary judgments with explicit consideration of risk and consequence. This approach supports better decisions precisely because it aligns more closely with how systems actually behave.

When probability is treated as a practical mindset—one that respects measurement limits, tolerates variation, and supports judgment—quality control becomes clearer and more humane. In that sense, probabilistic quality control is not about doing more mathematics. It is about thinking more honestly, and therefore making better decisions, without tears.

Acknowledgments

- Funded by:
  - Project Name: SFB 1574, A Circular Factory for the Perpetual Product
  - Funding Agency: Deutsche Forschungsgemeinschaft (DFG) -
  - Project ID: 471687386

On the Potential of Neural Processes for Task-Driven Sensor Placement

Frank Doehner

Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
frank.doehner@kit.edu

Abstract

Task-driven sensor placement aims to optimize measurement locations with respect to specific downstream objectives such as fault localization or diagnosability. Such formulations are central to many industrial and operational monitoring systems, where sensing resources are limited and measurements must directly support decision-making tasks. Neural Processes (NPs) offer a probabilistic, meta-learned modeling framework that can condition on sparse and irregular observations, represent task-level uncertainty, and generalize across varying system configurations, making them a natural candidate for task-driven sensor placement. This work proposes a theoretical framework for task-driven and adaptive sensor placement based on latent NPs. Sensor placement is formulated as a sequential decision problem guided by uncertainty over task-relevant latent variables, such as fault or external influence locations, under parametric system uncertainty. By augmenting latent NPs with auxiliary probabilistic prediction heads, the framework enables task-level posterior inference from partial observations, which can be used to define uncertainty-driven acquisition strategies for adaptive sensor placement.

1 Introduction

Optimal sensor placement (OSP) is a fundamental problem in the monitoring and identification of physical systems, with a wide range of applications spanning fields such as structural health monitoring ([12]), environmental sensing ([4]), and fault detection([13]). Given a limited sensing budget, the objective is to select sensor locations that maximize the information gained about unknown system states, parameters, or external influences. Well-chosen sensor configurations are crucial for accurate inference, robustness to uncertainty, and efficient downstream decision-making.

Sensor placement strategies are commonly categorized according to how and when sensor locations are selected. In static sensor placement, all sensor positions are determined a priori based on a fixed objective and assumed operating conditions. In contrast, sequential or adaptive sensor placement selects sensor locations iteratively, using measurements from previously placed sensors to guide subsequent decisions. A further distinction can be made between reconstruction-oriented placement, which aims to accurately recover full system states, and task-driven placement, where sensors are optimized for specific downstream objectives such as parameter estimation, fault localization, or detection of external influences. Adaptive and task-driven formulations are particularly challenging, as they require models that can incorporate partial observations, reason about uncertainty, and generalize across different sensor configurations.

Neural Processes (NPs) directly address these requirements by learning probabilistic predictors that condition on arbitrary sets of observations [10], enabling uncertainty-aware inference across varying sensor configurations. By modeling distributions over functions conditioned on arbitrary context sets, NPs naturally accommodate variable numbers and locations of sensors. Their permutation-invariant formulation enables flexible context–target splits, while their uncertainty-aware predictions support information-based and adaptive sensor placement strategies. Importantly, NPs can generalize across varying system parameters and operating regimes, making them well suited for task-driven and sequential sensor placement problems.

In this work, we investigate NPs as a practical framework for optimal and adaptive sensor placement in physical systems. Focusing on implementation and task-driven objectives, we postulate how NPs can be integrated with active measurement strategies to efficiently localize unknown influences and support robust inference under limited sensing resources.

2 Preliminaries and Modeling Framework

This section presents the problem formulation for OSP and introduces the NP models used for probabilistic inference from sparse observations.

2.1 Optimal Sensor Placement

OSP addresses the problem of selecting a subset of measurement locations that maximizes the utility of the acquired data under constraints on sensing resources. Depending on how sensor locations are selected and how measurement utility is defined, this problem admits a range of formulations, including static or adaptive approaches, for reconstruction-oriented or task-driven settings. To formalize this setting, we consider a system described by an unknown function

$$f : \mathcal{X} \to \mathcal{Y},$$

where $\mathcal{X}$ denotes the space of admissible measurement locations and $\mathcal{Y}$ the corresponding measurement space. A sensor placement strategy selects a finite set of locations $\mathcal{S} \subset \mathcal{X}$ at which noisy observations

$$y(x) = f(x) + \varepsilon, \quad x \in \mathcal{S},$$

are acquired.

In static sensor placement, the set $\mathcal{S}$ is chosen once by optimizing a predefined objective under assumed operating conditions, meaning that the system model, operating regime, and statistical properties of the measurements are fixed a priori and no information from future measurements is incorporated during the design. In contrast, adaptive sensor placement proceeds sequentially. At iteration k,

the placement decision is conditioned on the measurements $\mathcal{C}_k = \{(x_i, y_i)\}_{i=1}^{k-1}$ obtained at the previously selected sensor locations $S_k = \{x_i\}_{i=1}^{k-1}$, resulting in an evolving observation set. The placement strategy is therefore described by a policy $\pi : \mathcal{C}_k \mapsto x_k$, which explicitly depends on the current belief about the system.

Task-driven sensor placement further specifies an objective functional $\mathcal{U}(\mathcal{C}_k)$, which quantifies task performance given the current observation set. Typical examples include objectives related to parameter uncertainty [23], fault diagnosability [17], or detection probability [22]. In adaptive settings, sensor placement is often formulated as a greedy optimization problem,

$$x_k = \pi(\mathcal{C}_k) = \arg \max_{x \in \mathcal{X} \setminus \mathcal{S}_k} \mathbb{E}_{p(y(x)|\mathcal{C}_k)} \left[\mathcal{U}(\mathcal{C}_k \cup \{(x, y(x))\}) \right], \tag{2.1}$$

where the expectation is taken with respect to the predictive distribution $p(y(x) \mid \mathcal{C}_k)$ induced by the current model of the system.

Classical sensor placement approaches instantiate this formulation by assuming that the unknown function f is governed by a known physical or statistical model, which induces an explicit parametric form for the predictive distribution $p(y(x) \mid \mathcal{C}_k)$. Under these assumptions, the objective $\mathcal{U}$ often admits analytically tractable expressions. Common examples include observability and identifiability criteria [2], Fisher information maximization [20], and mutual information–based objectives [19]. When the assumed model accurately reflects the true system behavior, these approaches provide strong theoretical guarantees.

However, reliance on explicit system models restricts their applicability in nonlinear, high-dimensional, or partially understood systems. Evaluating the predictive distribution and the corresponding objective can become computationally prohibitive, and sensor placement decisions may degrade significantly when operating conditions or system parameters deviate from those assumed during design.

Data-driven approaches relax these assumptions by learning surrogate models of f from data, enabling approximate evaluation of the predictive distribution and the objective $\mathcal{U}$. Gaussian Processes are frequently employed in this context due to their probabilistic formulation and principled uncertainty estimates [16]. However, their scalability and flexibility are limited, particularly when sensor

configurations vary across tasks. More recent deep learning–based approaches improve scalability and expressiveness but often lack coherent uncertainty representations or require retraining when sensor layouts or task objectives change [17].

Taken together, these considerations highlight a fundamental challenge in OSP: the need for models that support efficient conditional inference, principled uncertainty quantification, and generalization across heterogeneous and evolving sensor configurations.

2.2 Neural Process

NPs are a family of probabilistic models designed for learning from small data sets while providing principled uncertainty estimates. They achieve this by combining meta-learning across tasks with a stochastic process formulation that models a distribution over predictors [10]. In this section, we first introduce the core principles underlying NPs and distinguish between two fundamental modeling paradigms: deterministic conditional predictors and models that incorporate global latent variables to represent uncertainty at the level of entire functions. We then discuss inherent limitations of these formulations and review architectural extensions that address them, including attention-based aggregation and convolutional representations.

2.2.1 Model structure

Contrary to supervised learning where a network is trained on one, typically larger, data set, the NP's meta-learning approach utilizes a meta-dataset $\mathcal{M} := \{\mathcal{D}^{(i)}\}_{i=1}^{N_{\text{tasks}}}$ that contains a large collection of tasks, where each task $\mathcal{D} := (\mathcal{C}, \mathcal{T})$ consists of a context set $\mathcal{C} := \{\left(x^{(c)}, y^{(c)}\right)\}_{c=1}^{C}$ and a target set $\mathcal{T} := \{\left(x^{(t)}, y^{(t)}\right)\}_{t=1}^{T}$. The aim of meta-learning is to create a neural network that efficiently learns from a context set and applies the acquired insights for prediction on the target set. Instead of just learning a predictor $\hat{f}(x)$ given a context set $\mathcal{C}$ as in supervised learning, a meta-learner learns a function that maps a context set to a task specific predictor $g(\mathcal{C}) := \mathcal{C} \mapsto \hat{f}(\,\cdot\,; \mathcal{C})$. The meta-learner

trains on the context sets and evaluates performance on the target sets by comparing the predictor output $\hat{f}(\mathbf{x}_\mathcal{T}; \mathcal{C})$, where $\mathbf{x}_\mathcal{T} := \{x^{(t)}\}_{t=1}^{T}$, to the target output $\mathbf{y}_\mathcal{T} := \{y^{(t)}\}_{t=1}^{T}$. Due to this meta-learning framework, NPs are able to share information across tasks, enabling few-shot learning on previously unknown context sets. In order to properly assess prediction quality, particularly when the context set is small, it is further essential to quantify the model's uncertainty.

Modeling a distribution over predictions $p(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C})$ instead of a single prediction $f(\mathbf{x}_\mathcal{T}; \mathcal{C})$ allows quantification of uncertainty directly through the model itself. This predictive distribution is equivalent to a stochastic process, under the condition that the distributions are consistent over arbitrary target sets $\mathcal{T}$. A NP parametrizes the predictive distribution $p_\theta(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C})$ using a neural network with parameters θ. In practice NPs adopt an encoder-decoder architecture, where the encodermaps the context set to a latent representation R:

$$R = \mathrm{Enc}_\theta(\mathcal{C}) = \rho_\theta \left(\sum_{c=1}^{C} \phi_\theta(x^{(c)}, y^{(c)}) \right) \tag{2.2}$$

Here, ρ and ϕ are functions learned by the neural network. The sum operator in the encoder enforces permutation invariance, since summation is commutative and associative and therefore produces the same aggregated representation regardless of the order of the context points. Moreover, because the aggregation reduces an arbitrary-size set to a fixed-dimensional vector, the resulting representation is independent of the number of context points.

2.2.2 Conditional and latent Neural Processes

Two main categories of decoders exist following two separate modeling philosophies. The conditional NP (CNP), depicted in 2.1 (gray, green), does not model global epistemic uncertainty over functions [9], whereas the latent NP (LNP), depicted in 2.1 (gray, orange), conditions on a global latent variable $\mathbf{z}$ that captures uncertainty over the learned function [10]. The CNP's predictive distri-

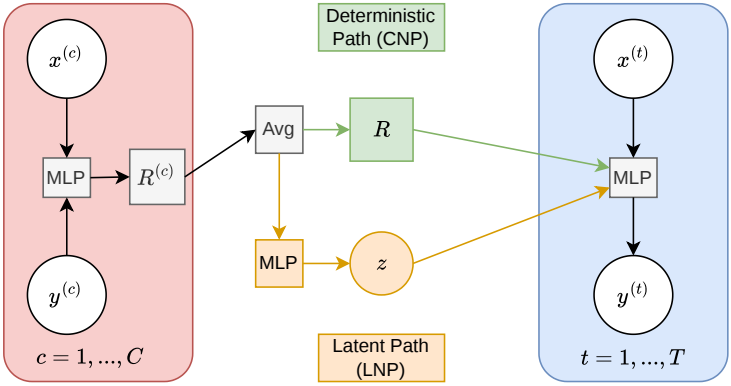


Figure 2.1: Conditional (gray, green) and latent (gray, orange) neural process architectures. In the latent path, the MLP parameterizes a Gaussian distribution $p_\theta(z \mid \mathcal{C}) = \mathcal{N}(\mu_\theta, \sigma_\theta)$, from which a global latent variable z is sampled. The index c denotes context samples and t denotes target samples.

bution at an arbitrary set of inputs $\mathbf{x}_\mathcal{T}$ is factorized conditioned on R:

$$p_\theta(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C}) = \prod_{t=1}^{T} p_\theta(y^{(t)} \mid x^{(t)}; R). \tag{2.3}$$

The LNP goes one step further and defines a global latent variable $\mathbf{z} \sim p_\theta(\mathbf{z} \mid \mathcal{C})$. The resulting predictive distribution is factorized conditioned on $\mathbf{z}$ instead:

$$p_\theta(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C}) = \int \prod_{t=1}^{T} p_\theta(y^{(t)} \mid x^{(t)}, \mathbf{z}) p_\theta(\mathbf{z} \mid \mathcal{C}) d\mathbf{z}. \tag{2.4}$$

Marginalizing over $\mathbf{z}$ introduces correlations between target points by coupling their predictions through shared latent randomness, thereby enabling the representation of uncertainty at the level of entire functions. The latent variable $\mathbf{z}$ primarily captures epistemic uncertainty over functions, while the conditional likelihood models observation noise. Moreover, when Gaussian conditional likelihoods are employed, the resulting predictive distribution corresponds to a

mixture of Gaussians [10], providing additional flexibility to capture multimodal posterior structure.
Both models can be trained by maximizing the log predictive likelihood

$$\mathcal{L} = \log p_\theta(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C}),$$

effectively optimizing predictive performance on the target set. However, for LNPs this likelihood involves an intractable integral over the global latent variable, necessitating variational approximations during training, typically via the maximization of an evidence lower bound (ELBO). As a result, the increased expressive power and coherent uncertainty representation afforded by latent variables comes at the cost of approximate inference and additional optimization complexity.
Two further limitations of vanilla NPs are their tendency to underfit due to global pooling–based aggregation and their limited ability to extrapolate reliably beyond the training distribution.

2.2.3 Attention-based Neural Processes

The tendency to underfit stems from the permutation-invariant encoding of the context set, which is enforced through global pooling in the encoder. Although this design ensures flexibility with respect to the size and ordering of the context set, it necessarily discards relational and spatial structure among context points. As a result, the vanilla NP variants often produce overly smooth predictions and may underfit in settings where local dependencies or sharp variations are present [15].
Attention-based NPs (in short AttnNPs) address this limitation by replacing the permutation-invariant global pooling operation in the encoder with input-dependent aggregation [15], as depicted in 2.2. Instead of mapping the entire context set to a single global representation R, attention mechanisms compute target-specific representations that weigh context points according to their relevance for a given target input. This allows the model to preserve local structure while retaining permutation invariance with respect to the context set. In practice,

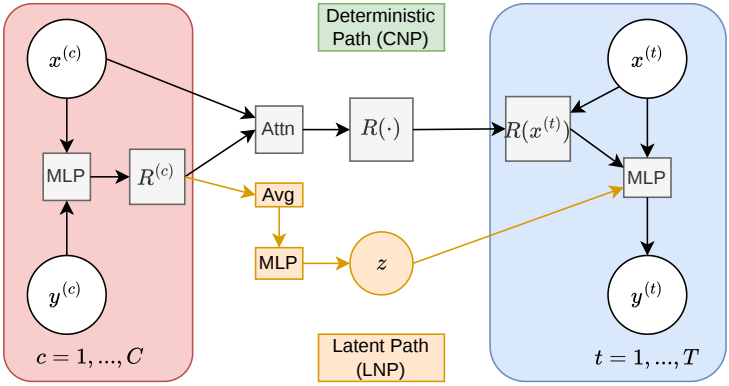


Figure 2.2: Attentive conditional (gray) and attentive latent (gray, orange) neural process architectures. In contrast to the vanilla formulation, the deterministic path employs cross-attention between context and target inputs to produce a target-specific representation. In the latent path, an MLP parameterizes a Gaussian distribution $p_\theta(z \mid R) = \mathcal{N}(\mu_\theta, \sigma_\theta)$ from which a global latent variable z is sampled. The decoder conditions on the target input together with the attention-based representation and, in the latent case, the sampled latent variable. The indices c and t denote context and target samples, respectively.

the encoder from 2.2 is replaced by a target-specific representation:

$$R(x^{(t)}) = \text{Enc}_\theta(x^{(t)}, \mathcal{C}) = \sum_{c=1}^{C} w_\theta(x^{(c)}, x^{(t)})\phi(x^{(c)}, y^{(c)}).$$

The attention mechanism $w_\theta(x^{(c)}, x^{(t)})$ is a learned weighting function that assigns relevance to each context point $x^{(c)}$ for a given target location $x^{(t)}$, creating a target-specific representation $R(x^{(t)})$. The decoder architectures as in 2.3 and 2.4 remain mostly unchanged. For the attentive CNP, predictions are conditioned on the target-specific context representation $R(x^{(t)})$:

$$p_\theta(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C}) = \prod_{t=1}^{T} p_\theta\big(y^{(t)} \mid x^{(t)}, R(x^{(t)})\big), \tag{2.5}$$

and for the attentive LNP, the predictive distribution additionally integrates over the global latent variable $\mathbf{z}$:

$$p_\theta(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C}) = \int \prod_{t=1}^{T} p_\theta\big(y^{(t)} \mid x^{(t)}, R(x^{(t)}), \mathbf{z}\big)\, p_\theta(\mathbf{z} \mid \mathcal{C})\, d\mathbf{z}. \tag{2.6}$$

2.2.4 Convolutional Neural Processes

Attention-based aggregation alleviates underfitting by replacing global pooling with target-dependent context representations. However, it does not directly address the limited extrapolation performance of vanilla NPs beyond the training distribution. Convolutional NPs (ConvNPs) are designed to address this remaining limitation by embedding explicit inductive biases that promote robust generalization outside the training domain [11, 8]. Although attention enables input-dependent conditioning on relevant context points, it does not impose structural assumptions on the input space itself. ConvNPs instead encode translation-equivariant inductive biases through convolutional architectures, allowing the model to exploit spatial regularities and promote consistent behavior across locations. To support continuous and irregularly sampled input locations, ConvNPs employ a continuous convolution operator known as Set Convolution (SetConv) [11]. Given a context set $\mathcal{C}$ SetConv maps the context observations to a continuous feature field by smoothing each embedded observation with a learnable kernel function k_θ, enabling evaluation at arbitrary input locations:

$$\mathrm{SetConv}(\mathcal{C})\,(x) = \sum_{c=1}^{C} \begin{bmatrix} 1 \\ y^{(c)} \end{bmatrix} k_\theta(x - x^{(c)}). \tag{2.7}$$

The resulting feature field consists of two channels: a density channel, given by the weighted sum of kernel responses, and a value channel encoding the locally aggregated observations. The inclusion of the constant channel allows the model to distinguish between regions with sparse or no context observations and regions with dense coverage, while avoiding explicit normalization. The feature field is evaluated on a finite set of query locations, typically forming a regular grid, yielding a discretized representation suitable for convolutional processing. A convolutional neural network is then applied to this representation,

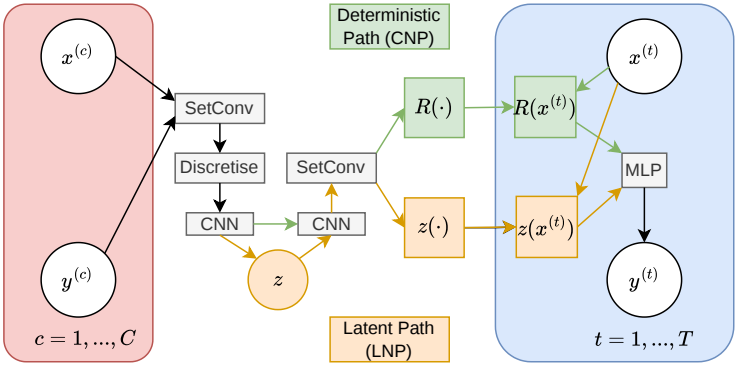


Figure 2.3: Convolutional conditional (gray, green) and convolutional latent (gray, orange) neural process architectures. Context observations are first mapped to a discretized grid via a SetConv operation and subsequently processed by a convolutional neural network (CNN). In the conditional path, the resulting grid representation is interpolated to the target locations using SetConv and decoded to produce predictions. In the latent path, the first CNN parameterizes a Gaussian distribution $p_\theta(z \mid C) = \mathcal{N}(\mu_\theta, \sigma_\theta)$ over a latent function defined on the grid. A sample z is processed by a second CNN before being interpolated to the target locations via SetConv. The indices c and t denote context and target samples, respectively.

followed by a second SetConv operation to query at any target location $x^{(t)}$:

$$R(x^{(t)}) = \text{SetConv}\left(\text{CNN}_\theta\left(\left\{\text{SetConv}(\mathcal{C})(x^{(u)})\right\}_{u=1}^{U}\right)\right)(x^{(t)}). \quad (2.8)$$

Here, the convolutional network operates on features indexed by the query locations $x^{(u)}$, and the outer SetConv interpolates these features to the target location $x^{(t)}$. This sequence of operations produces a latent feature map that encodes spatial structure and locality through shared convolutional filters.

In the latent variant (ConvLNP), the convolutional operation is split into two stages as depicted in 2.3. The first CNN parameterizes a Gaussian distribution over a latent function on the discretized grid, from which a latent field $\mathbf{z}$ is sampled. A second CNN then processes this sampled latent representation $R(x^{(t)}; \mathbf{z})$ before interpolation to the target locations. The respective predictive

distribution for ConvCNP stays equivalent to attentive CNP as in 2.5, where only the representation changes. In the ConvLNP case, the latent variable no longer enters the decoder separately but instead influences predictions through the convolutional representation:

$$p_\theta(\mathbf{y}_\mathcal{T} \mid \mathbf{x}_\mathcal{T}; \mathcal{C}) = \int \prod_{t=1}^{T} p_\theta\big(y^{(t)} \mid x^{(t)}, R(x^{(t)}; \mathbf{z})\big)\, p_\theta(\mathbf{z} \mid \mathcal{C})\, d\mathbf{z}. \tag{2.9}$$

3 Related work

Beyond attention-based and convolutional architectures, several NP variants have been proposed to address complementary limitations in uncertainty modeling, temporal structure, and generalization. Transformer NPs (TNPs) [18] leverage self-attention mechanisms to improve scalability and expressiveness when conditioning on large context sets. Sequential NPs (SNPs) [21] extend NPs to sequential settings by explicitly modeling temporal dependencies, enabling coherent uncertainty propagation over time. Fourier NPs (FNPs) [5] introduce spectral representations to support arbitrary-resolution inference and improve extrapolation beyond the training distribution by capturing global structure in the frequency domain. Gaussian NPs [3] constrain the predictive family to Gaussian distributions, improving calibration and training stability at the cost of limited expressiveness in multimodal scenarios. Neural Diffusion Processes (NDPs) [6] incorporate diffusion-based mechanisms to enhance uncertainty estimation and sample quality. NP Contrastive Learning (NPCL) [14] adapts NPs to continual learning by modeling tasks through structured latent distributions and regularizing them to reduce forgetting, while leveraging NP-based uncertainty estimates for task inference and confidence assessment.

Because NPs provide scalable conditional inference with coherent uncertainty estimates and do not require retraining for changing sensor configurations, they have been adopted for sensor placement problems in some recent work. Convolutional GNPs have been used for environmental sensor placement by selecting measurement locations that maximally reduce predictive uncertainty over spatial fields, demonstrating favorable scalability and flexibility compared to Gaussian Process–based baselines, particularly in non-stationary settings [1]. Closely

related work employs ConvNPs with mixture-density decoders to separate epistemic and aleatoric uncertainty and to drive adaptive sensor placement decisions that target regions of high epistemic uncertainty [7]. These approaches highlight the suitability of NP-based models as scalable probabilistic surrogates in spatial sensing problems.

Despite these promising results, the use of NPs for adaptive sensor placement in industrial or operational settings remains largely unexplored. In particular, the combination of simulation-based training, task-level uncertainty modeling, and online deployment to physical systems, where system parameters, operating conditions, and fault characteristics vary across instances, has received little systematic attention. This gap is notable, as NPs are naturally suited to such settings due to their meta-learning formulation and ability to condition on arbitrary subsets of observations.

4 Latent Neural Processes for Task-Driven Sensor Placement

This section presents a theoretical framework for fault localization and adaptive sensor placement using LNPs, intended to demonstrate the modeling. We consider a general task-driven inference setting in which the objective is to localize faults or external influences in a physical system under parametric uncertainty.

4.1 Problem Setting and Task Distribution

Let the system of interest be described by an unknown function

$$f_\tau : \mathcal{X} \to \mathcal{Y},$$

where, as described in 2.1, $\mathcal{X}$ denotes the space of admissible sensor locations and $\mathcal{Y}$ the measurement space. The function f_τ depends on a set of latent task parameters $\tau = (\xi, \kappa)$, where $\xi \in \mathcal{F}$ represents a fault or external influence location, and $\kappa \in \mathcal{K}$ denotes additional physical parameters (e.g. material properties,

boundary conditions, or operating regimes) that are not of direct interest but affect the system response.

Measurements are obtained as noisy observations

$$y(x) = f_\tau(x) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2),$$

at sensor locations $x \in \mathcal{X}$. Within a meta-learning framework, each task $\mathcal{D}_\tau$ corresponds to a realization of τ drawn from a prior distribution $p(\tau)$, inducing a distribution over functions $\{f_\tau\}$. The meta-dataset $\mathcal{M}$ consists of a collection of such tasks with varying fault locations and physical parameters. We employ the LNP model described in Section 2.2 to represent this task distribution.

4.2 Auxiliary Parameter Prediction with Uncertainty

To explicitly infer task-relevant latent parameters, we augment the LNP with an auxiliary probabilistic head that maps the global latent variable z to a conditional distribution over the fault location and physical parameters. Formally, this defines a mapping $z \mapsto p(\tau \mid z) \in \mathcal{P}(\mathcal{F} \times \mathcal{K})$. In practice, the conditional distribution is represented as

$$p_\theta(\xi, \kappa \mid z) = p_\theta(\xi \mid z)\, p_\theta(\kappa \mid z),$$

where each factor is modeled by a parametric distribution whose parameters are predicted from z by a neural network.

During training, the simulator generates data under varying ground-truth task parameters $\tau = (\xi, \kappa)$, which are known and used for supervision. The auxiliary head is trained by maximizing the conditional log-likelihood

$$\mathcal{L}_{\text{aux}} = \mathbb{E}_{q(z|\mathcal{C})} \left[\log p_\theta(\xi, \kappa \mid z) \right],$$

which is combined with the standard LNP objective,

$$\mathcal{L} = \mathcal{L}_{\text{LNP}} + \lambda_{\text{aux}}\, \mathcal{L}_{\text{aux}},$$

and is weighted through the hyperparameter λ_{aux}. This joint objective encourages the latent representation to capture task-specific information that is both

predictive of observations and informative for fault localization and parameter inference.

At inference time, uncertainty in the predicted fault location and physical parameters arises from two sources: latent uncertainty captured by the posterior $q(z \mid \mathcal{C})$ and conditional uncertainty represented by $p_\theta(\xi, \kappa \mid z)$.
The resulting predictive distribution is given by

$$p(\xi, \kappa \mid \mathcal{C}) = \int p_\theta(\xi, \kappa \mid z)\, q(z \mid \mathcal{C})\, dz,$$

which can be approximated via Monte Carlo sampling. The integration over z induces a mixture distribution over (ξ, κ), allowing the model to represent multimodal posterior structure.

4.3 Greedy Adaptive Sensor Placement

At test time, a new system instance with unknown task parameters $\tau^\star = (\xi^\star, \kappa^\star)$ is observed through a growing context set $\mathcal{C}_k = \{(x_i, y_i)\}_{i=1}^{k-1}$. The variational posterior $q(z \mid \mathcal{C}_k)$ represents the current belief over task-specific latent structure induced by sparse observations and unobserved physical parameters. Uncertainty over the fault location is obtained by marginalizing over the latent variable and the corresponding auxiliary prediction head,

$$p(\xi \mid \mathcal{C}_k) = \int p_\theta(\xi \mid z)\, q(z \mid \mathcal{C}_k)\, dz,$$

which can be approximated in practice using Monte Carlo samples.

This representation enables both point estimates and calibrated uncertainty measures for the fault location.

We now return to the adaptive sensor placement formulation of Section 2.1. Given the current context set $\mathcal{C}_k$, the objective is to select the next sensor location x_k so as to maximally reduce uncertainty about the fault location ξ. Let $\mathcal{U}(\mathcal{C}_k)$ denote a scalar uncertainty measure derived from $p(\xi \mid \mathcal{C}_k)$, such as entropy or posterior variance. Using the greedy acquisition strategy defined in (2.1), the next measurement location is selected based on an uncertainty measure

$\mathcal{U}(\mathcal{C}_k)$ derived from the posterior $p(\xi \mid \mathcal{C}_k)$. This procedure yields an online, task-driven sensor placement policy that explicitly targets uncertainty reduction in fault localization rather than global state reconstruction.
In practical implementations, this framework requires the selection of a concrete acquisition function $\mathcal{U}(\cdot)$ and an encoder architecture that is well matched to the structure of the underlying sensing problem. In particular, the choice of encoder determines how context observations are aggregated and which inductive biases are imposed. Depending on whether the measurements exhibit spatial structure, graph-based interactions, temporal dependencies, or heterogeneous sensing modalities, different encoder architectures may be more appropriate.

5 Conclusion

This work has proposed a theoretical framework for task-driven and adaptive sensor placement based on LNPs, with an emphasis on uncertainty-aware inference over task-relevant variables under limited and sequential observations. By formulating sensor placement as a decision problem guided by posterior uncertainty rather than global state reconstruction, the framework highlights how NPs can serve as a flexible probabilistic backbone for task-driven sensing in complex physical systems.
While a substantial body of theoretical work has established NPs as a flexible framework for probabilistic regression and meta-learning, their adoption in concrete sensor placement settings remains limited. Existing studies that integrate NPs with sensor placement have largely focused on environmental sensing problems, where the objective is typically to reduce predictive uncertainty over spatial fields. In contrast, task-driven sensor placement in industrial or operational contexts has received comparatively little attention. With this work, we aim to take a first step toward addressing this gap by outlining a conceptual framework that illustrates how NPs can be integrated with adaptive, task-driven sensor placement beyond environmental monitoring scenarios. Several steps remain to translate the proposed framework into practical sensor placement methodologies. First, concrete Neural Process architectures and encoder designs must be selected and validated for specific sensing modalities and task objectives,

including an assessment of how different inductive biases affect task-level uncertainty representation. Second, suitable acquisition functions must be defined and evaluated to ensure that uncertainty estimates derived from latent NPs lead to stable and informative sensor placement decisions in sequential settings. Third, simulation-based studies are required to assess robustness with respect to model mismatch, noise, and variability in system parameters, and to compare task-driven objectives against reconstruction-oriented baselines. Finally, deployment in real or high-fidelity experimental systems will require addressing computational constraints, online inference efficiency, and the interaction between model uncertainty and physical sensing limitations.

References

[1] Tom R Andersson et al. “Environmental sensor placement with convolutional Gaussian neural processes”. In: *Environmental Data Science* 2 (2023), e32.

[2] Mani Bhushan and Raghunathan Rengaswamy. “Design of sensor location based on various fault diagnostic observability and reliability criteria”. In: *Computers & Chemical Engineering* 24.2-7 (2000), pp. 735–741.

[3] Wessel P Bruinsma et al. “The Gaussian neural process”. In: *arXiv preprint arXiv:2101.03606* (2021).

[4] Charles C Castello et al. “Optimal sensor placement strategy for environmental monitoring using wireless sensor networks”. In: *2010 42nd Southeastern Symposium on System Theory (SSST)*. IEEE. 2010, pp. 275–279.

[5] Kun Chen et al. “Fnp: Fourier neural processes for arbitrary-resolution data assimilation”. In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 137847–137872.

[6] Vincent Dutordoir et al. “Neural diffusion processes”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 8990–9012.

[7] Feyza Eksen, Stefan Oehmcke, and Stefan Lüdtke. “Where to Measure: Epistemic Uncertainty-Based Sensor Placement with ConvCNPs”. In: *arXiv preprint arXiv:2511.22567* (2025).

[8] Andrew Foong et al. “Meta-learning stationary stochastic process prediction with convolutional neural processes”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 8284–8295.

[9] Marta Garnelo et al. “Conditional neural processes”. In: *International conference on machine learning*. PMLR. 2018, pp. 1704–1713.

[10] Marta Garnelo et al. *Neural Processes*. 2018. arXiv: 1807.01622.

[11] Jonathan Gordon et al. “Convolutional conditional neural processes”. In: *arXiv preprint arXiv:1910.13556* (2019).

[12] Sahar Hassani and Ulrike Dackermann. “A systematic review of optimization algorithms for structural health monitoring and optimal sensor placement”. In: *Sensors* 23.6 (2023), p. 3293.

[13] Kang He, Minping Jia, and Conghu Liu. “A review of optimal sensor deployment to diagnose manufacturing systems”. In: *IEEE Access* 6 (2018), pp. 27418–27432.

[14] Saurav Jha et al. “Npcl: Neural processes for uncertainty-aware continual learning”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 34329–34353.

[15] Hyunjik Kim et al. “Attentive Neural Processes”. In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=SkE6PjC9KX.

[16] Andreas Krause, Ajit Singh, and Carlos Guestrin. “Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies.” In: *Journal of Machine Learning Research* 9.2 (2008).

[17] Yinhua Liu et al. “Optimal sensor placement for fixture fault diagnosis using Bayesian network”. In: *Assembly Automation* 31.2 (2011), pp. 176–181.

[18] Tung Nguyen and Aditya Grover. “Transformer Neural Processes: Uncertainty-Aware Meta Learning Via Sequence Modeling”. In: *Proceedings of the 39th International Conference on Machine Learning*. PMLR. 2022, pp. 16569–16594.

[19] Kathleen Schmidt et al. “Sequential optimal positioning of mobile sensors using mutual information”. In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 12.6 (2019), pp. 465–478.

[20] Shuangwen Sheng, Li Zhang, and Robert X Gao. “A systematic sensor-placement strategy for enhanced defect detection in rolling bearings”. In: *IEEE Sensors Journal* 6.5 (2006), pp. 1346–1354.

[21] Gautam Singh et al. “Sequential neural processes”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[22] Rustam Stolkin and Ionut Florescu. “Probability of detection and optimal sensor placement for threshold based detection systems”. In: *IEEE Sensors Journal* 9.1 (2008), pp. 57–60.

[23] Liangliang Yang et al. “An active learning-driven optimal sensor placement method considering sensor position distribution toward structural health monitoring”. In: *Structural and Multidisciplinary Optimization* 67.12 (2024), p. 210.

Improving the Trust Region Method for Bayesian Optimization

Saksham Kiroriwal

Kognitive Industrielle Systeme (KIS)
Fraunhofer IOSB, Germany
saksham.kiroriwal@iosb.fraunhofer.de

Abstract

Bayesian optimization has shown to be an effective method for global optimization. However, in high dimensional problems, vanilla Bayesian optimization suffers from the curse of dimensionality. Trust region methods have shown to be effective in high dimensional situations, by decomposing the global optimization problem into a sequence of local optimization problems. This paper proposes a new trust region method for Bayesian optimization that is inspired from existing literature on Kullback-Leibler divergence constraints and information geometry. The proposed method adapts the trust region shape to the local geometry of the objective function. We hypothesize that a trust region method is the aware of the local information geometry of the objective function, will be able to improve convergence speed and sample efficiency in high-dimensional spaces.

1 Introduction

Many optimisation tasks in practice are black-box: the objective is expensive to evaluate, and gradients are unavailable or unreliable. This setting arises in aircraft and vehicle design [9, 12], in drug and materials discovery [15, 18], and when tuning hyperparameters of machine-learning models [16]. In such

settings, one can afford only a modest budget of evaluations—often on the order of tens to a few thousand—so methods that build a surrogate model of the objective and optimise that surrogate have become standard. Among these, Bayesian optimisation (BO) [11] with Gaussian process (GP) surrogates [10] is widely used because it naturally balances exploitation of promising regions and exploration of uncertain ones. The procedure fits a GP to the available evaluations and then selects new points by maximising an acquisition function derived from the GP, ensuring each new evaluation is informative.

Standard BO that uses a single global GP, however, scales poorly with the number of decision variables. Performance typically degrades once the dimension exceeds roughly ten, since the design space grows exponentially while the evaluation budget remains limited [4]. For medium- and high-dimensional black-box problems, one therefore often simplifies the problem. Existing strategies include restricting the GP to a trust region and doing local search [14, 6], reducing dimension by assuming only a few variables matter [17], or that some directions are less sensitive [5]. Work by [8] suggests using vanilla GP models with dimensionally scaled log-normal priors that capture mainly global trends [8]. Trust-region-based BO has received growing interest in this regard.

However, most of the trust region methods for BO are based heavily on length-scales of the learned kernel function of the vanilla Gaussian Process or a similar flavour of Gaussian Process surrogate model. Beyond this, as shown in section 2.3, the geometry that is implied by the TuRBO (see section 2.2) is globally Euclidean and the trust region is location independent and only model dependent. In this paper, we propose a trust region method for BO that draws on ideas from Kullback–Leibler (KL) divergence constraints and information geometry and views the acquisition function as a manifold defined by the mean and variance of the GP posterior, which in turn are defined by the input location.

2 Preliminaries

In this section, we discuss the existing framework on GP and the trust region methods for BO.

2.1 Gaussian Process

A *Gaussian process* (GP) can be viewed as a probability distribution over functions defined on the input domain $\mathcal{X} \subseteq \mathbb{R}^{D_x}$. Formally, a Gaussian Process is a collection of random variables, any finite number of which have a joint Gaussian distribution. It can be written as

$$f(\mathbf{x}) \;\sim\; \mathcal{GP}\Big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\Big), \tag{2.1}$$

where $m(\cdot) : \mathbb{R}^{D_x} \to \mathbb{R}$ is the mean function, and $k(\cdot, \cdot') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the covariance (kernel) function.

Given a set of N observations $\{\mathbf{x}_n,\, y_n\}_{n=1}^{N}$, where each $\mathbf{x}_n \in \mathbb{R}^{D_x}$ denotes an input location and $y_n \in \mathbb{R}$ is the corresponding observed output. We place a GP prior over the latent function values $\mathbf{f} = \{f_n\}_{n=1}^{N}$, encoded as

$$p(\mathbf{f};\, \mathbf{x}) \;=\; \mathcal{N}\Big(m(\mathbf{x}), k(\mathbf{x},\, \mathbf{x}')\Big), \tag{2.2}$$

and define the likelihood for each observation y_n as $p(y_n|f_n)$, where $\mathbf{x} = \{\mathbf{x}_n\}_{n=1}^{N}$and $\mathbf{y} = \{y_n\}_{n=1}^{N}$. Using Bayes' rule, the posterior over the latent function values, given all observations, is

$$p(\mathbf{f} \,|\, \mathbf{y};\, \mathbf{x}) \;=\; \frac{p(\mathbf{y} \,|\, \mathbf{f})\; p(\mathbf{f};\, \mathbf{x})}{\displaystyle\int p(\mathbf{y} \,|\, \mathbf{f})\; p(\mathbf{f};\, \mathbf{x})\; d\mathbf{f}}. \tag{2.3}$$

This posterior encapsulates how the observed data update the prior GP assumptions. The kernel $k(\cdot, \cdot')$ and the mean function $m(\cdot)$ usually include hyperparameters (e.g., length-scale, signal variance). These can be optimized by maximizing the *marginal log-likelihood* (MLL) of the observed data:

$$\mathcal{L}_{\text{GP}} \;=\; \sum_{n=1}^{N} \log \mathbb{E}_{p(f_n;\, \mathbf{x}_n)} \Big[p(y_n \,|\, f_n) \Big]. \tag{2.4}$$

In the special case of a Gaussian likelihood with additive noise variance ϵ_y^2, the marginal distribution $p(\mathbf{y}|\mathbf{x})$ becomes $\mathcal{N}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}') + \epsilon_y^2 \mathbf{I}\big)$, leading to a closed-form expression for Equation 2.4. However, solving for the exact posterior in this scenario requires $\mathcal{O}(N^3)$ operations due to the inversion of an $N \times N$ covariance matrix [13].

2.2 Bayesian Optimization

Bayesian optimization (BO) proceeds by making a sequence of decisions under uncertainty. At each step, the next point to evaluate is chosen using the current GP posterior over the objective. The choice is made by maximizing an acquisition function that trades off exploration of uncertain regions and exploitation of promising areas. Below, we outline three acquisition strategies commonly used in Bayesian optimization.

2.2.1 Expected Improvement (EI)

Expected Improvement (EI) selects the next point by maximizing the expected gain over the current best observed value y_{best}. Formally EI is defined as [7]

$$\text{EI}(\mathbf{x}) = \mathbb{E}_{p(y|\mathbf{x},\mathcal{D})}\left[\max\left(0, y - y_{\text{best}}\right)\right] \tag{2.5}$$

When the predictive distribution at a candidate point is Gaussian (as implied by the GP posterior), EI can be written in closed form using the standard normal probability density function ϕ and cumulative distribution function Φ:

$$\text{EI}(\mathbf{x}) = \Sigma_{f(\mathbf{x})}\, h\left(\frac{\mu_{f(\mathbf{x})} - y_{\text{best}}}{\Sigma_{f(\mathbf{x})}}\right), \quad h(a) = \phi(a) + a\,\Phi(a). \tag{2.6}$$

2.2.2 Log Expected Improvement (LogEI)

To address the numerical flatness of EI far from the data, Ament et al. [2] introduced Log Expected Improvement (LogEI), which maximizes the logarithm of the expected improvement. A numerically stable formulation keeps gradients non-zero even when the improvement is very small:

$$\mathbf{x} = \arg\max_{\mathbf{x}\in\mathcal{X}} \text{LogEI}(\mathbf{x}); \quad \text{LogEI}(\mathbf{x}) = \log h(a) + \log \Sigma_{f(\mathbf{x})}, \tag{2.7}$$

where $a = \frac{\mu_{f(\mathbf{x})} - y_{\text{best}}}{\Sigma_{f(\mathbf{x})}}$.

The implementation uses a piecewise formulation for numerical precision across all values of a [2]. Thus, the optimizer can make progress in regions where

plain EI would vanish, improving the exploration–exploitation trade-off. LogEI remains non-zero away from the data and can explore near the boundary; it drops to zero mainly near already observed points, yielding better exploration than EI.

2.2.3 Trust Region Bayesian Optimization (TuRBO)

Bayesian optimization in high dimensions or data scarcity can be particularly difficult. Trust Region Bayesian Optimization (TuRBO) developed by [6], has become increasingly popular. TuRBO restricts the search to a local trust region (TR) instead of using a single global GP and acquires new input locations within the trust region. We describe the single-TR case. The TR is a hyperrectangle centered at the incumbent $\mathbf{x}_{\text{best}}$. The TR is controlled by two components: a scalar *base side length* L, and the *per-dimension side lengths* derived from the GP kernel lengthscales. At the start of a run we set $L \leftarrow L_{\text{init}}$. The side length in dimension $i \in D_x$ is $L_i = \lambda_i L / \left(\prod_{j=1}^{D_x} \lambda_j\right)^{1/D_x}$, where λ_i is the lengthscale of the GP kernel in dimension i. Thus the hyperrectangle has volume L^{D_x} and is elongated along directions with larger lengthscale. At each iteration, an acquisition function selects one or more candidates inside this TR. A *success* is a candidate that improves on $\mathbf{x}_{\text{best}}$; a *failure* is one that does not. After ξ_{succ} consecutive successes, the TR is expanded: $L \leftarrow \min\{L_{\max}, 2L\}$. After ξ_{fail} consecutive failures, it is shrunk: $L \leftarrow L/2$. Success and failure counters are reset to zero whenever L is changed. If L falls below $L_{\min}$, the TR is discarded and a new one is initialized with base side length L_{init}. The hyperparameters ξ_{succ}, ξ_{fail}, $L_{\min}$, $L_{\max}$, and L_{init} control this behavior. Algorithm 2.1 summarizes the single-TR procedure [6].

2.3 The Geometry of Trust Regions

TuRBO implicitly assumes a flat Euclidean geometry. With an Automatic Relevance Determination (ARD) kernel, the kernel is anisotropic but the trust region remains input location independent. TuRBO [6] defines the trust region $\mathcal{T}(\mathbf{x}_{\text{best}})$ as a hyperrectangle centered at the current best point $\mathbf{x}_{\text{best}}$, constrained by a side length L:

$$\mathcal{T}(\mathbf{x}_{\text{best}}) = \left\{\mathbf{x} \in \mathcal{X} \mid \|\mathbf{x} - \mathbf{x}_{\text{best}}\|_\infty \leq L/2\right\}. \tag{2.8}$$

Algorithm 2.1 TuRBO (single trust region)

Require: Black-box objective Ω; TR hyperparameters $\xi_{\text{succ}}, \xi_{\text{fail}}, L_{\text{min}}, L_{\text{max}}, L_{\text{init}}$
Ensure: Incumbent $\mathbf{x}_{\text{best}}$
Initialize $\mathbf{x}_{\text{best}}$, $L \leftarrow L_{\text{init}}$, $n_{\text{succ}} \leftarrow 0$, $n_{\text{fail}} \leftarrow 0$
while not converged **do**
    TR & model: $L_i \leftarrow \lambda_i L \big/ \left(\prod_{j=1}^{D_x} \lambda_j\right)^{1/D_x}$; TR $\leftarrow$ hyperrectangle at $\mathbf{x}_{\text{best}}$ with sides L_i; fit GP to $\mathcal{D}$
    Acquire & evaluate: $\mathbf{x}_{\text{next}} \leftarrow \arg\max_{\mathcal{T}}\{\alpha(\mathbf{x})\}$ (where $\mathcal{T}$ is the TR from the previous step); $y_{\text{next}} \leftarrow \Omega(\mathbf{x}_{\text{next}})$; $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_{\text{next}}, y_{\text{next}})\}$
    if y_{next} improves incumbent **then**
        $\mathbf{x}_{\text{best}} \leftarrow \mathbf{x}_{\text{next}}$; $n_{\text{succ}} \leftarrow n_{\text{succ}} + 1$; $n_{\text{fail}} \leftarrow 0$
    else
        $n_{\text{fail}} \leftarrow n_{\text{fail}} + 1$; $n_{\text{succ}} \leftarrow 0$
    end if
    if $n_{\text{succ}} = \xi_{\text{succ}}$ **then**
        $L \leftarrow \min\{L_{\text{max}}, 2L\}$; $n_{\text{succ}}, n_{\text{fail}} \leftarrow 0$
    end if
    if $n_{\text{fail}} = \xi_{\text{fail}}$ **then**
        $L \leftarrow L/2$; $n_{\text{succ}}, n_{\text{fail}} \leftarrow 0$
        if $L < L_{\text{min}}$ **then**
            $L \leftarrow L_{\text{init}}$ (reinitialize TR)
        end if
    end if
end while

The constraint in L_∞-norm is equivalent to a ball in the L_2-norm. More generally, isotropic trust regions use the L_2-norm $\|\mathbf{x} - \mathbf{x}_{\text{best}}\|_2 \leq R$. In both cases, the underlying geometry is Euclidean. Consider a stationary kernel $k(\mathbf{x}, \mathbf{x}') = \psi(d(\mathbf{x}, \mathbf{x}'))$, where $\psi : \mathbb{R}_+ \to \mathbb{R}$ is a positive-definite function, and where $d(\mathbf{x}, \mathbf{x}')$ is the distance between two input $\mathbf{x}$ and $\mathbf{x}'$. Using ARD kernel, the distance can be expressed as

$$d^2(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^\top \Lambda (\mathbf{x} - \mathbf{x}'), \tag{2.9}$$

where $\Lambda = \text{diag}(1/\lambda_1^2, \ldots, 1/\lambda_{D_x}^2)$ and λ_i is the lengthscale of the kernel in the i-th input dimension. In information geometry, if the metric tensor $\mathcal{G}(\mathbf{x})$ is constant, the manifold is Euclidean [1]. From (2.9), we have $\mathcal{G}(\mathbf{x}) = \Lambda$. Since Λ is independent of $\mathbf{x}$, the Riemann curvature tensor is zero, hence the manifold is a linear transformation of Euclidean space (anisotropic but flat). TuRBO thus operates in a flat world, merely stretched along the coordinate axes by the lengthscales [13, 1].

Proposition 2.3.1 (Geometric rigidity of TuRBO)**.** *The geometry underlying TuRBO is Euclidean. The trust region* (2.8) *is an L_∞-ball; with an ARD kernel, the induced kernel distance is Mahalanobis distance with constant metric $\mathcal{G} = \Lambda$, hence flat. The trust region from TuRBO cannot adapt to local curvature or to the objective's behaviour.*

3 Statistical Manifold of Predictive Distributions

In this section, we define a statistical manifold as discussed by Amari [1] and then analyze the geometry of the GP posterior.

Definition 3.0.1 (Statistical manifold)**.** A statistical manifold is a Riemannian manifold $\mathcal{S}$, where each point $\theta \in \mathcal{S}$ corresponds to a probability distribution $p(\xi\,;\,\theta)$. The manifold is equipped with a Riemannian metric $\mathcal{G}(\theta)$ defined by the Fisher information metric

$$\mathcal{I}(\theta) = \mathbb{E}_{p(\xi\,;\,\theta)}[\nabla_\theta \log p(\xi\,;\,\theta) \nabla_\theta \log p(\xi\,;\,\theta)^\top]. \tag{3.1}$$

For univariate predictive Gaussian posterior, the statistical manifold $\mathcal{S}$ is parameterized by mean $m \in \mathbb{R}$ and variance $\sigma^2 \in \mathbb{R}_+$ and is defined as

$$\mathcal{S} = p(\xi\,;\,m, \sigma^2) | \boldsymbol{\theta} = \left(m, \sigma^2\right) \in \mathbb{R} \times \mathbb{R}_+.$$

The previous section showed that TuRBO imposes a *flat* geometry on the input space. To motivate a geometry-aware trust region, we now describe a different view: the GP posterior maps each input to a predictive distribution, and the space of such distributions carries a natural Riemannian structure.

3.1 The Statistical Manifold of Predictive Distributions

As discussed in section 2.1, for a single-task GP, the predictive posterior at any input location $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^D$ is a univariate Gaussian distribution:

$$p(y \mid \mathbf{x}, \mathcal{D}) = \mathcal{N}\left(y; \mu(\mathbf{x}), \Sigma^2(\mathbf{x})\right). \tag{3.2}$$

We consider the space of all univariate Gaussian distributions, which forms a 2-dimensional Riemannian manifold, denoted as the statistical manifold $\mathcal{S}$. Each point $p \in \mathcal{S}$ is uniquely identified by the coordinate vector $\boldsymbol{\theta} = (\mu, \Sigma) \in \mathbb{R} \times \mathbb{R}_+$.

The Gaussian Process defines a smooth mapping $\Psi : \mathcal{X} \to \mathcal{S}$ from the input space to this statistical manifold. Specifically, we define the GP Posterior Map in terms of the natural coordinates of $\mathcal{S}$:

$$\Psi(\mathbf{x}) \coloneqq (\mu(\mathbf{x}), \Sigma(\mathbf{x})). \tag{3.3}$$

Geometrically, the image $\Psi(\mathcal{X}) \subset \mathcal{S}$ represents a D-dimensional surface embedded within the geometry of the statistical manifold.

3.2 Geometry of the GP Map: Immersion vs. Submersion

The geometric nature of the map $\Psi \colon \mathcal{X} \to \mathcal{S}$ depends fundamentally on the dimensionality of the input space D_x relative to the statistical manifold (which is always 2-dimensional).

The Jacobian of Ψ, denoted by $d\Psi$, captures how infinitesimal changes in input translate to changes in the predictive distribution and is defined as

$$(d\Psi)_{\mathbf{x}} = \begin{bmatrix} \nabla\mu(\mathbf{x})^\top \\ \nabla\Sigma(\mathbf{x})^\top \end{bmatrix} \in \mathbb{R}^{2 \times D_x}. \tag{3.4}$$

If $D_x \leq 2$ the Jacobian has full column rank (rank equal to D_x) and the map is called an immersion [1]. Geometrically, the input space creates a curve or surface embedded within the statistical manifold without self-intersection or collapse. The induced pullback metric is positive-definite, defining a standard Riemannian geometry on $\mathcal{X}$.

However in Bayesian optimization, we typically operate where $D_x \gg 2$. Here, the map cannot be an immersion. Instead, it is generically a submersion. This implies that at any point $\mathbf{x}$, there exists a subspace of directions (the null space of $d\Psi$) along which the predictive distribution remains instantaneously constant.

The Fisher Information Metric F on $\mathcal{S}$ induces a metric on the input space via the pullback and is defined as

$$\mathcal{G}(\mathbf{x}) = (d\Psi)_{\mathbf{x}}^{\top} F(\Psi(\mathbf{x})) (d\Psi)_{\mathbf{x}}. \tag{3.5}$$

In the high-dimensional submersion case, $\mathcal{G}(\mathbf{x})$ becomes a *degenerate* Riemannian metric (positive semi-definite). This degeneracy is geometrically meaningful as it assigns zero "information distance" to movements that do not alter the GP's belief, effectively collapsing the search space to relevant directions only.

4 Methodology: Geometric Trust Regions

We motivate our approach by analyzing the asymptotic behavior of Gaussian Process posteriors in high-dimensional Euclidean spaces. We then formally define a trust region constraint on the statistical manifold of predictive distributions and derive a computationally efficient diagonal approximation consistent with our implementation.

4.1 The Vanishing Information Problem in High Dimensions

As defined in Equation 2.8, Standard Trust Region Bayesian Optimization methods, such as TuRBO, define the trust region $\mathcal{T}_k$ as a Euclidean ball or hypercube centered at the current incumbent $\mathbf{x}_k^*$:

$$\mathcal{T}_k = \{\mathbf{x} \in \mathcal{X} \mid \|\mathbf{x} - \mathbf{x}_k^*\|_p \leq \Delta_k\}, \tag{4.1}$$

where typically $p = 2$ or $p = \infty$. This definition implicitly assumes that the Euclidean distance $\|\cdot\|_p$ is a meaningful proxy for model uncertainty or function variation. However, as discussed in section 2.3, this imposes a flat Euclidean

geometry over the input domain, which becomes problematic in high dimensions due to the behavior of the Gaussian Process posterior.

Consider a GP prior $f \sim \mathcal{GP}(\mu_0, k)$ with a stationary kernel bounded by the signal variance, $k(\mathbf{x}, \mathbf{x}') \leq \Sigma_0^2$. Given a dataset $\mathcal{D}_N = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, the posterior variance at a test point $\mathbf{x}$ is given by

$$\Sigma_N^2(\mathbf{x}) = \Sigma_0^2 - \mathbf{k}_N(\mathbf{x})^\top (\mathbf{K}_N + \eta^2 \mathbf{I})^{-1} \mathbf{k}_N(\mathbf{x}), \tag{4.2}$$

where $\mathbf{k}_N(\mathbf{x}) = [k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_N)]^\top$ is the covariance vector between the test point and the training data, and η^2 is the noise variance.

The reduction in uncertainty (information gain) is governed strictly by the magnitude of $\mathbf{k}_N(\mathbf{x})$. Applying the Rayleigh quotient bounds to the quadratic form, we obtain:

$$\Sigma_0^2 - \Sigma_N^2(\mathbf{x}) \leq \frac{1}{\lambda_{\min}(\mathbf{K}_N + \eta^2 \mathbf{I})} \|\mathbf{k}_N(\mathbf{x})\|_2^2. \tag{4.3}$$

For standard stationary kernels such as the Squared Exponential with length-scale λ, the correlation decays exponentially with the Euclidean distance. Let $\delta(\mathbf{x}) = \min_{i=1,\ldots,N} \|\mathbf{x} - \mathbf{x}_i\|_2$ denote the distance to the nearest observation. It follows that:

$$\|\mathbf{k}_N(\mathbf{x})\|_2^2 = \sum_{i=1}^N k(\mathbf{x}, \mathbf{x}_i)^2 \leq N \Sigma_0^4 \exp\left(-\frac{\delta(\mathbf{x})^2}{\lambda^2}\right). \tag{4.4}$$

Substituting this into (4.3), we observe that the posterior variance reverts to the prior variance exponentially fast as $\delta(\mathbf{x})$ increases:

$$\Sigma_N^2(\mathbf{x}) \geq \Sigma_0^2 - C \exp\left(-\frac{\delta(\mathbf{x})^2}{\lambda^2}\right), \tag{4.5}$$

where C is a constant depending on N and the spectrum of the Gram matrix.

In high-dimensional spaces, maintaining a small $\delta(\mathbf{x})$ becomes combinatorially expensive. For the unit hypercube $\mathcal{X} = [0, 1]^D$, the fill distance (dispersion) $h_{N,D}$ for any sequence of N points scales as $O(N^{-1/D})$. Consequently, the distance from an arbitrary test point to the nearest observation is lower-bounded by:

$$\delta(\mathbf{x}) \gtrsim N^{-1/D}. \tag{4.6}$$

As the dimension D increases, the term $N^{-1/D}$ approaches 1 unless N grows exponentially with D. Thus, for a fixed budget N, $\delta(\mathbf{x})$ remains large almost everywhere in $\mathcal{X}$.

Using (4.5), we can understand that in high dimensions, the GP posterior $\Sigma_N^2(\mathbf{x})$ effectively converges to the uninformed prior Σ_0^2 everywhere except in neighborhood of observed points. In these vast "empty" regions, the posterior mean is constant, and the acquisition function gradients vanish. A static Euclidean trust region treats these uninformative directions identically to informative ones, leading to inefficient exploration. To mitigate this, the trust region must adapt to the intrinsic geometry of the posterior, contracting in informative directions and expanding where the model has reverted to the prior.

4.2 Trust Regions as Constraints on the Statistical Manifold

We adopt a manifold perspective where the trust region is defined not in the input space $\mathcal{X}$, but on the statistical manifold $\mathcal{S}$ of predictor distributions. Let $\Psi : \mathcal{X} \to \mathcal{S}$ be the GP that maps input $\mathbf{x}$ to its predictive posterior

$$p(y|\mathbf{x}) = \mathcal{N}(\mu(\mathbf{x}), \Sigma^2(\mathbf{x})).$$

We constrain the search to a region where the Kullback-Leibler (KL) divergence between the predictive distribution at the best observed center $\mathbf{x}_{\text{best}}$ and the candidate input $\mathbf{x}$ is bounded by a threshold β as

$$\mathcal{T}_{KL} = \{\mathbf{x} \in \mathcal{X} \mid D_{KL}\left(p(y|\mathbf{x}) \| p(y|\mathbf{x}_{\text{best}})\right) \leq \beta\}. \tag{4.7}$$

Using the second-order Taylor expansion of the KL divergence, we can approximate this constraint using the Fisher Information Matrix (FIM), $\mathcal{G}(\mathbf{x})$, which is the pullback metric from $\mathcal{S}$ to $\mathcal{X}$:

$$D_{KL}\left(p(y|\mathbf{x}) \| p(y|\mathbf{x}_{\text{best}})\right) \approx \frac{1}{2}(\mathbf{x} - \mathbf{x}_{\text{best}})^\top \mathcal{G}(\mathbf{x}_{\text{best}})(\mathbf{x} - \mathbf{x}_{\text{best}}), \tag{4.8}$$

where $\mathcal{G}(\mathbf{x})$ is defined as the expected outer product of the score function of predictive distribution defined as

$$\mathcal{G}(\mathbf{x}) = \mathbb{E}_{y \sim p(y|\mathbf{x})}\left[\nabla_{\mathbf{x}} \log p(y|\mathbf{x}) \nabla_{\mathbf{x}} \log p(y|\mathbf{x})^\top\right]. \tag{4.9}$$

Hence, the trust region can be expressed as the constraint in the input that is constrained by the Fisher-Rao metric [1] at the best observed location $\mathbf{x}_{\text{best}}$

$$\mathcal{T}_{Riem} = \{\mathbf{x} \in \mathcal{X} \mid \|\mathbf{x} - \mathbf{x}_{\text{best}}\|^2_{\mathcal{G}(\mathbf{x}_{\text{best}})} \leq \Delta_k^2\}. \tag{4.10}$$

4.3 Efficient Diagonal Approximation via Score Estimators

Solving the acquisition problem over a general Riemannian ellipsoid, as described in Equation 4.10, requires decomposing the full $D \times D$ Fisher matrix, which is computationally expensive ($O(D^3)$) and numerically unstable in high dimensions where $\mathcal{G}(\mathbf{x})$ may be rank-deficient and the surrogate model might hallucinate.

To ensure scalability, we approximate the geometry using the diagonal of the Fisher Information Matrix. Let $\hat{\mathcal{G}}(\mathbf{x}) = \text{diag}(g_{dd}(\mathbf{x}) \; \forall d \in \{1, \ldots, D_x\})$. We compute these diagonal elements via a Monte Carlo score function estimator. Let $\{y_s\}_{s=1}^S$ be samples drawn from the posterior $p(y|\mathbf{x})$. The gradient of the log-likelihood for a fixed sample y_s (treating y_s as data) provides an unbiased estimator of the score:

$$g_{dd}(\mathbf{x}) \approx \frac{1}{S} \sum_{s=1}^{S} \left(\frac{\partial}{\partial x_d} \log p(y_s|\mathbf{x}) \right)^2. \tag{4.11}$$

It is important to note that we detach the samples y_s from the computational graph to ensure we compute the derivative of the log-likelihood density, not the reparameterization derivative.

Under this diagonal approximation, the principal axes of the trust region align with the coordinate axes. We relax the ellipsoidal L_2-constraint to a weighted L_∞-constraint (box constraint) to facilitate efficient optimization with bound-constrained solvers (e.g., L-BFGS-B) in BoTorch [3]. The resulting *Axis-Aligned Riemannian Trust Region* is defined as:

$$\mathcal{T}_{Diag} = \{\mathbf{x} \in \mathcal{X} \mid |x_d - (\mathbf{x}_{\text{best}})_d| \sqrt{g_{dd}(\mathbf{x}_{\text{best}}) + \rho} \leq \frac{\Delta_k}{2}, \forall d = 1, \ldots, D_x\}, \tag{4.12}$$

where ρ is a regularization factor for numerical stability. The ρ also serves as the prior over the geometry. This formulation adaptively scales the search space.

Dimensions with high information sensitivity (large g_{dd}) are tightly constrained, while dimensions where the posterior is flat (small g_{dd}, indicating reversion to the prior) are allowed larger varying bounds.

5 Conclusion

We have outlined a geometric view of trust regions in Bayesian optimization. Instead of restricting the search to a ball or box in the usual input space, the trust region is defined on the statistical manifold of predictive distributions: we only allow steps that keep the model's belief within a bounded divergence from the current best. That way, the region where we search reflects where the surrogate is actually informative, rather than where we happen to be in Euclidean coordinates. In high dimensions, Euclidean trust regions become poorly aligned with the model's uncertainty, so a manifold-based definition can in principle lead to more effective, geometry-aware optimization.

For computational reasons, we avoid full Riemannian geometry. The full Fisher Information Matrix is costly and unstable in high dimensions, so we use a diagonal approximation and score-based estimators. The resulting constraint is an axis-aligned box in the input space, which fits naturally into standard bound-constrained solvers and does not introduce a large computational overhead. We therefore argue that this idea could pave the way for trust-region methods that are both efficient and grounded in the manifold of predictions rather than in the raw input domain, with the potential for better sample efficiency when the surrogate's uncertainty varies strongly across dimensions.

References

[1] Shun-ichi Amari. *Information Geometry and Its Applications*. Springer, 2016.

[2] Sebastian Ament et al. "Unexpected improvements to expected improvement for Bayesian optimization". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023.

[3] Maximilian Balandat et al. “BoTorch: A framework for efficient Monte-Carlo Bayesian optimization”. In: *Advances in neural information processing systems* 33 (2020), pp. 21524–21538.

[4] Mickael Binois and Nathan Wycoff. “A Survey on High-dimensional Gaussian Process Modeling with Application to Bayesian Optimization”. In: *ACM Transactions on Evolutionary Learning and Optimization* 2.2 (2022), pp. 1–26.

[5] David Eriksson and Martin Jankowiak. “High-Dimensional Bayesian Optimization with Sparse Axis-Aligned Subspaces”. In: *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence*. PMLR. 2021, pp. 493–503.

[6] David Eriksson et al. “Scalable global optimization via local Bayesian optimization”. In: *Advances in neural information processing systems* 32 (2019).

[7] Peter I Frazier. “A tutorial on Bayesian optimization”. In: *arXiv preprint arXiv:1807.02811* (2018).

[8] Carl Hvarfner, Erik Orm Hellsten, and Luigi Nardi. “Vanilla Bayesian Optimization Performs Great in High Dimensions”. In: *Proceedings of the 41st International Conference on Machine Learning*. PMLR. 2024, pp. 20793–20817. URL: https://openreview.net/forum?id=OfT8MgIqHT.

[9] Rhea P Liem, Charles A Mader, and Joaquim RRA Martins. “Surrogate Models and Mixtures of Experts in Aerodynamic Performance Prediction for Aircraft Mission Analysis”. In: *Aerospace Science and Technology* 43 (2015), pp. 126–151.

[10] Georges Matheron. “Principles of Geostatistics”. In: *Economic Geology* 58.8 (1963), pp. 1246–1266.

[11] Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. “The Application of Bayesian Methods for Seeking the Extremum”. In: *Towards Global Optimization* 2.117-129 (1978), p. 2.

[12] Maliki Moustapha et al. “Adaptive Kriging Reliability-Based Design Optimization of an Automotive Body Structure Under Crashworthiness Constraints”. In: *Proceedings of the 12th International Conference on Applications of Statistics and Probability in Civil Engineering (ICASP12)*. University of British Columbia. Vancouver, Canada, July 2015.

[13] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[14] Rommel G Regis. “Trust Regions in Kriging-Based Optimization with Expected Improvement”. In: *Engineering Optimization* 48.6 (2016), pp. 1037–1059.

[15] Benjamin J Shields et al. “Bayesian Reaction Optimization as a Tool for Chemical Synthesis”. In: *Nature* 590.7844 (2021), pp. 89–96.

[16] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. “Practical Bayesian Optimization of Machine Learning Algorithms”. In: *Proceedings of the 25th Advances in Neural Information Processing Systems* (2012).

[17] Ziyu Wang et al. “Bayesian Optimization in a Billion Dimensions via Random Embeddings”. In: *Journal of Artificial Intelligence Research* 55 (2016), pp. 361–387.

[18] Yichi Zhang, Daniel W Apley, and Wei Chen. “Bayesian Optimization for Materials Design with Mixed Quantitative and Qualitative Variables”. In: *Scientific Reports* 10.1 (2020), p. 4924.

Security Metrics and Model Performance in Privacy-Preserving Machine Learning in Computer Vision

Leon Ranke

Fraunhofer Institute of
Optronics, System Technologies and Image Exploitation (IOSB)
Karlsruhe, Germany
leon.ranke@iosb.fraunhofer.de

Abstract

Privacy-Preserving Machine Learning (PPML) for vision tasks addresses two essentially contradicting objectives: image security and model utilisation. Especially, secure model training is still a widely unsolved problem. Developing and evaluating PPML faces two main challenges. Firstly, many methods have technical constraints that limit their applicability to inference time. Secondly, most PPML approaches add significant computational overheads and integrate poorly and opaquely into model development. As a result, PPML techniques for computer vision are primarily evaluated on simple image classification tasks.

This report argues that image classification is too simplistic to reliably assess the security-utility trade-offs. To demonstrate this, we apply three scale-space-based image transformations that introduce distortions with only minor effects on classification accuracy. These experiments show that classification alone is an inadequate test bed and highlight the need for a broader, more differentiated suite of evaluation tasks.

1 Introduction

Deep learning in computer vision is increasingly deployed in privacy-sensitive domains such as healthcare [13, 16], smart cities [7], and industrial applications [37], where images may contain personal or confidential information. Protecting data and models across the entire Machine Learning (ML) lifecycle, from training to inference and post-deployment, is a core requirement driven by regulation, user expectations, and organisational risk. Privacy-Preserving Machine Learning (PPML) provides a toolbox of approaches with complementary scopes: Differential Privacy (DP) [8, 9] limits the dependence of individual samples on the trained model; Fully Homomorphic Encryption (FHE) [12] and Secure Multi-Party Computation (SMPC) [3] enable computation over encrypted and secret-shared data respectively; Federated Learning (FL) [30] keeps datasets local during training; and Perceptual Image Encryption (PIE) [10] schemes obfuscate visual content to facilitate training directly on transformed images. Despite this breadth, secure and practical end-to-end training for modern vision models remains largely unsolved [38, 23].

DP limits how much the resulting model changes when individual data points are added or removed [9], typically achieved by adding calibrated noise to gradients during training [2, 36]. While reducing the risk of model inversion and membership inference attacks [35, 49], model weights, intermediate activations, and datasets themselves are not protected during the process. Furthermore, the addition of noise must be carefully balanced to achieve the desired level of privacy while maintaining model utility [44]. On the other hand, FL trains models locally, sharing only the resulting gradients. This prevents direct access to the images; however, as shown in [4], data can be reconstructed from the shared gradients. Due to the inherent complexity of FHE, only small models [32, 26, 40] or isolated sub-components of model training [50, 47] have been realised. Transferring model architectures and training procedures to the encrypted domain, combined with a substantial increase in training time [52], renders FHE ill-suited for training state-of-the-art vision models. Similarly, SMPC protocols incur substantial communication cost, particularly for non-linear activations, limiting their scalability [31, 45, 15]. Consequently, SMPC is often used solely for secure gradient aggregation in FL [41, 43]. PIE schemes [42, 19, 27] do

not require changing training procedures or model architectures; however, they are limited to image classification tasks and have been shown to be vulnerable against ciphertext-only reconstruction attacks [6, 29].

For modern computer vision tasks, established workflows rely on long training runs over many optimisation steps and non-linear activation functions. These demands conflict with Privacy Enhancing Technologies (PETs) that increase computational and communication demands, restrict access to data, or limit numerical flexibility. Large-scale pre-training, central to state-of-the-art performance, remains practically unfeasible under strong privacy guarantees. Additionally, many primitives, such as activations, batch normalisation, or attention, require approximations or alternative designs for secure training. Consequently, most PPML approaches focus on training small-scale image classification tasks [47, 32, 45, 42, 19, 27], with shallow networks and simple datasets (MNIST [25], CIFAR-10 [21]), where training models is feasible within the strict constraints imposed by PPML.

Despite the extensive literature on PET design, comparatively little work addresses how security and utility should be evaluated jointly. In practice, classification accuracy on a single dataset serves as the dominant utility metric, while security is either argued analytically or assessed via isolated statistical tests. This raises a fundamental question: Does maintaining high classification performance imply strong utility?

This technical report demonstrates that classification accuracy on a single dataset systematically overestimates model utility under structural image degradation. The impact of simple image-domain transformations, applied at different spatial scales, on model utility is evaluated. These transformations progressively remove complementary structural components of the visual signal, enabling systematic analysis of which aspects are essential for downstream learnability. The focus is on practical comparability rather than formal cryptographic guarantees. The main contributions are:

(i) An overview of image security metrics spanning perceptual leakage, encrypted-image statistics, and differential, as well as key analysis.

(ii) A systematic quantification of the relationship between image security metrics and classification performance under scale-controlled transformations.

(iii) Empirical evidence that classification accuracy on a single dataset is insufficient to characterise the security-utility trade-off.

The remainder of this technical report reviews relevant PPML evaluation practices in Section 2 and provides an overview of image security metrics in Section 3. Section 4 defines transformations and sets up subsequent experiments, with results presented in Section 5. Finally, Section 6 concludes with implications and suggestions for future work.

2 Related Work

Research on PPML in computer vision focuses on the design of cryptographic primitives and training protocols. In contrast, less attention has been devoted to systematic evaluation methodologies that jointly assess security and utility.

2.1 Model Utility Evaluation

PETs can be broadly categorised into cryptographic techniques, functionality preserving paradigms, and image-domain transformations that modify the signal representation. While all lines of work report downstream task performance as a utility metric, the implications differ fundamentally.

FHE and SMPC are cryptographic protocols. Any computable function can, in principle, be evaluated in the encrypted or secret-shared domain. Observed utility degradations are not a result of the paradigms themself, but rather practical limitations arising from decreased numerical precision and approximations of non-linear functions to limit overheads. Because of the unfeasible long runtimes, evaluations are limited to small convolutional or fully connected networks on simple benchmarks, reporting classification accuracies alongside runtime [32, 47, 45, 52].

DP preserves the underlying task formulation; utility degradation results from the noise injected during model training to satisfy a privacy budget [2, 36, 44]. The trade-off between privacy and utility is therefore typically characterised by reporting classification accuracy relative to the privacy parameter. Importantly, DP does not aim to protect images from the party performing optimisation; rather, it limits what can be inferred about individual samples given the trained model. Addressing inference-time privacy concerns, such as model inversion [11, 48] or membership inference [33, 5]. FL retains the original data representation and model architecture and has been applied successfully to a wide range of vision problems [30, 53, 18]. While avoiding centralisation of raw images, gradients exchanged during training may leak information, enabling the reconstruction of training samples [4].

In contrast, PIE schemes operate directly on the image representation, altering the statistical properties of datapoints to obscure visual content while maintaining compatibility with training pipelines [42, 27, 19]. Unlike the previously mentioned PETs, these methods provide no guarantees that arbitrary downstream tasks remain learnable after encryption.

Utility evaluation in PPML predominantly relies on image classification on small-scale datasets such as CIFAR-10 [21] or MNIST [25]. Whether other vision tasks, such as detection, segmentation, or representation learning, remain feasible under such transformations is rarely investigated.

2.2 Image Security Evaluation

In contrast to metrics for model utility, image encryption literature provides extensive statistical evaluation suites [39, 28]. Section 3.1 introduces various metrics and groups them into categories. As a result, two largely separate evaluation cultures have emerged; cryptographic PETs (i.e. , HE, SMPC) emphasise functionality preservation and computational feasibility, while image encryption research focuses on statistical randomness. PIE methods lie at the intersection of these domains but are commonly validated using classification accuracy on a single dataset.

Rather than proposing a new privacy mechanism, this technical report uses controlled image-domain transformations as analytical probes to examine the relationships among structural image degradation, statistical security metrics, and classification performance.

3 Image Security Metrics

Image security can be evaluated in numerous ways. Table 3.1 lists common metrics for 8-bit RGB images $\mathcal{I}$, grouped by category. Perceptual leakage quantifies the amount of information that remains in the encrypted image $\widetilde{\mathcal{I}}$. Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR) operate at the pixel level. In contrast, the Structural Similarity Index Measure (SSIM) [46] and Learned Perceptual Image Patch Similarity (LPIPS) [51] capture structural and perceptual similarities, respectively. Mutual Information (MI), on the other hand, estimates the shared information between plain and encrypted domains.

Statistics based solely on encrypted images include deviations from uniformity, measured via χ^2 *goodness-of-fit test* or Kullback-Leibler (KL) divergence, Information Entropy (IE), adjacent-pixel Correlation Coefficient (CC), and the Spectral Flatness Measure (SFM).

Differential analysis tests the avalanche property, ensuring that a slight change in a plain image induces widespread, high-magnitude changes in the encrypted image, as measured by the Number of Pixel Change Rate (NPCR) and Unified Average Changing Intensity (UACI). Key analysis is used to determine the effectiveness of the encryption key $k \in \mathcal{K}$, reporting the effective key space size $|\mathcal{K}|$ and key sensitivity via NPCR and UACI, ensuring that a slight change in the encryption key leads to significant changes in the encrypted image. These metrics are widely adopted in image encryption literature, see [39, 28] for an overview of image encryption techniques and evaluation suites.

In this report, the term image security refers to statistical and perceptual resistance to visual reconstruction, rather than formal cryptographic security guarantees. The presented metrics quantify structural and statistical obfuscation but do not constitute proofs of semantic security.

Table 3.1: Image security metrics for RGB images. Ranges and ideal values are averages per channel (R, G, B). Intensities are in $\{0, \ldots, L-1\}$ (for 8-bit images $L = 256$).

Name	Abbreviation	Range	Ideal Value
Perceptual Leakage			
Mean Squared Error	MSE	$[0, (L-1)^2]$	$(L-1)^2$
Peak Signal-to-Noise Ratio (dB)	PSNR	$[0, \infty)$	0
Structural Similarity Index Measure [46]	SSIM	$[0, 1]$	0
Learned Perceptual Image Patch Similarity [51]	LPIPS	$[0, \text{inf})$	$\uparrow$
Mutual Information (bits)	MI	$[0, \log_2(L)]$	0
Encrypted Image Statistics			
Chi-Squared goodness-of-fit test	χ^2	$[0, \infty)$	0
Kullback-Leibler Divergence	D_{KL}	$[0, \infty)$	0
Shannon Entropy (bits)	IE	$[0, \log_2(L)]$	$\log_2(L)$
2D Information Entropy (bits) [24]	2D-IE	$[0, \log_2(L)]$	$\log_2(L)$
Adjacent-pixel correlation	CC	$[-1, 1]$	0
Spectral Flatness Measure	SFM	$[0, 1]$	1
Differential Analysis			
Number of Pixel Change Rate (w.r.t. $\mathcal{I}$)	$\mathrm{NPCR}_{\mathcal{I}}$	$[0, 1]$	$(1 - \frac{1}{L})$
Unified Average Changing Intensity (w.r.t. $\mathcal{I}$)	$\mathrm{UACI}_{\mathcal{I}}$	$[0, 1]$	$\frac{L^2-1}{3L(L-1)}$
Key Analysis			
Key-space size	$\lvert\mathcal{K}\rvert$	$[1, \infty)$	$\geq 2^{128}$
Number of Pixel Change Rate (w.r.t. k)	NPCR_k	$[0, 1]$	$(1 - \frac{1}{L})$
Unified Average Changing Intensity (w.r.t. k)	UACI_k	$[0, 1]$	$\frac{L^2-1}{3L(L-1)}$

4 Experimental Setup

In addition to data protection, PPML must maintain task utility. To analyse the relationship between image security metrics and downstream learnabilit, classification performance is evaluated under controlled image-domain transformations.

A ResNet-34 [14] is trained from scratch on the Animals species classification dataset [1], containing 15 classes with 2000 training images per class. Images are resized to 416×416 pixels. The dataset provides sufficient spatial resolution to meaningfully analyse scale-dependent transformations, while remaining computationally tractable. All models are trained from scratch for 50 epochs using identical hyperparameters. Adam optimisation [17] is employed with

an initial learning rate of 10^{-3}, reduced at epochs 20 and 40. Early stopping prevents overfitting while ensuring a fixed upper bound on optimisation costs across experiments. Top-1 accuracy is reported on the test set.

Three one-parameter transformation families are constructed, isolating complementary components of the visual signal:

- **Gaussian Blurring ($\boldsymbol{\sigma}$)** acts as a low-pass filter, attenuating high-frequency components while preserving spatial topology and global intensity relationships. For a scale parameter σ, images are obtained via separable convolution with a Gaussian kernel of standard deviation σ and kernel size $k = 2\lfloor 3\sigma \rfloor + 1$. Increasing σ progressively suppresses fine-scale geometric detail. This allows for isolating the contribution of spatial frequency bands to downstream task performance.
- **Locally Orderless Images (LOI) ($\boldsymbol{B}$)** discard spatial ordering within non-overlapping $B \times B$ windows while approximately preserving marginal intensity distributions inside those windows [20]. As B increases, spatial structure is removed at progressively larger scales until, when B equals the image size, spatial structure is eliminated entirely, while global intensity statistics are retained. LOIs provide a principled mechanism for determining how spatial structures at different scales affect task performance.
- **Chaotic Logistic Map (CLM) based encryption ($\boldsymbol{N}$)**. Following the encryption scheme introduced in [34], block-wise operations chosen from {identity, inversion, addition mod 256, XOR} are stacked in order to define an encryption transformation. Operator selection and parametrisation are based on a sequence of pseudo-random numbers (keystream). The keystream itself is generated once per scale $N = W - B$, where B is the block size, using the logistic map $x_{t+1} = p\, x_t(1 - x_t)$ with control parameter $p = 3.999$ and initial condition x_0. At each transformation step, one operation is selected and applied to all pixels within the $B \times B$ block. When $B = 1$ ($N = 415$), this reduces to per-pixel operations and is equivalent to [34]. Larger scales yield more spatially uniform transformations and thus less obfuscation. For comparability, a unique keystream is used across the entire dataset, which is generally not recommended for cryptographically secure encryption.

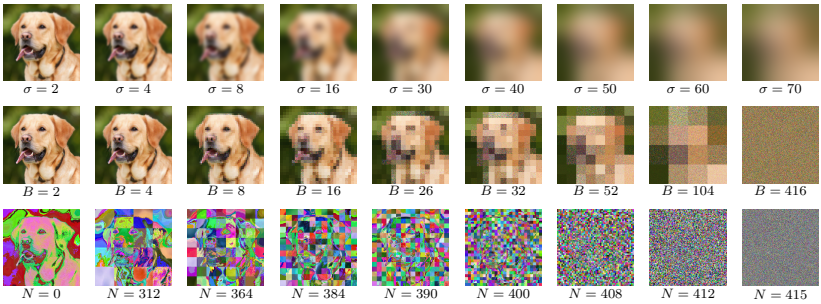


Figure 4.1: Qualitative effect of increasing scale for each transformation family. Top: Gaussian blurring with standard deviation $\sigma \in \{2, 4, 8, 16, 30, 40, 50, 60, 70\}$. Middle: LOI [20] with block size $B \in \{2, 4, 8, 16, 26, 32, 52, 104, 416\}$. Bottom: CLM [34] with $N \in \{0, 312, 364, 384, 390, 400, 408, 412, 415\}$. Parameter values are shown under each image.

Together, these transformations span complementary axes of structural degradation: frequency attenuation, spatial reordering, and stochastic decorrelation. For a predefined set of scales (σ for blurring, B for LOI, and N for CLM), a separate model is trained and evaluated on the transformed dataset; Figure 4.1 provides qualitative examples. Top-1 test accuracy and perceptual leakage, as well as encrypted image statistics from Table 3.1, are reported relative to a plain baseline (i.e. , no transformation). Differential- and key analysis are omitted, because the focus lies on structural properties relevant to downstream learnability rather than cryptographic key sensitivity or avalanche effects.

Images are represented as 8-bit RGB ($[0, 255]$) unless stated otherwise. Perceptual leakage is quantified via MSE, PSNR, and SSIM [46] between plain and transformed images, complemented by LPIPS [51] computed in the continuous domain using an AlexNet [22] backbone with inputs normalised to $[-1, 1]$. MI is estimated per channel from a 256×256 joint intensity histogram. Encrypted-image statistics are evaluated solely on transformed images: deviation from a uniform histogram using χ^2 goodness-of-fit and KL-divergence to the uniform distribution (log_2), Shannon entropy over 256 bins, and 2D delentropy computed from centred intensity differences following Larkin [24]. CC is computed as adjacent-pixel Pearson correlation averaged across RGB channels (in horizontal,

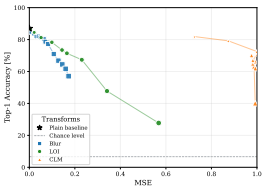
vertical, and diagonal directions), and SFM is implemented as the ratio of geometric to arithmetic mean of the 2D power spectral density after DC removal. Metrics are computed per image and aggregated over the test set.

5 Model Utility Evaluation

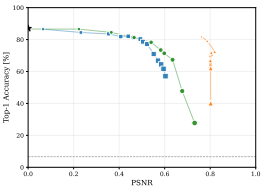
Figures 5.1(a)-5.1(k) visualise classification performance directly against each security metric. We normalised all security metrics to $[0, 1]$ using fixed min-max mappings with direction-aware inversion, so that 0 indicates low and 1 high security. This exposes the relationship between structural degradations, statistical properties, and utility. Selected quantitative results are listed in Table 5.1.

- For Gaussian blurring, accuracy gradually decreases as σ increases (down to 57.10% at $\sigma = 70$). The relatively moderate drops in accuracy, especially for small σ, indicate that high-frequency components contribute, but are not essential for the task of animal species classification. Perceptual metrics (i.e. , MSE, PSNR, SSIM, LPIPS, MI) show a relatively smooth, monotonic decrease with accuracy reflecting the controlled removal of fine-scale details. Histogram- and entropy-based metrics achieve lower security than the plain baseline in line with the visually more uniform images.

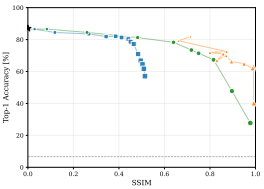
- LOIs induce a stronger model utility degradation with scale (from 86.60% at $B = 1$ to 27.75% at $B = 416$). However, accuracy remains well above the chance level of 6.67% (for 15 classes), indicating that residual class information is present in intensity distributions alone. While most security metrics approach the ideal target values as scale increases, histogram-based metrics, as well as IE, remain constant across scales. This reflects the preservation of intensity distributions in transformed images and highlights the importance of including multiple security metrics for evaluation. Relative accuracy drops across scales suggest that coarse spatial structure is paramount, while small local structures are less critical for the analysed classification task.



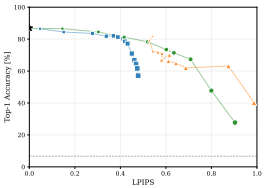
(a) MSE



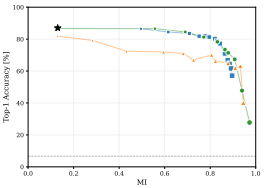
(b) PSNR



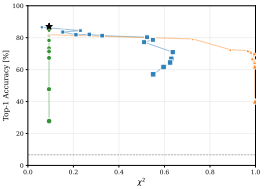
(c) SSIM



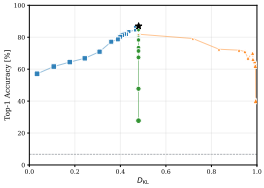
(d) LPIPS



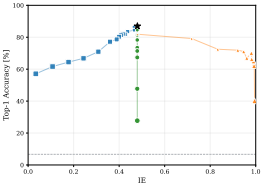
(e) MI



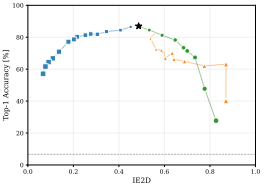
(f) χ^2



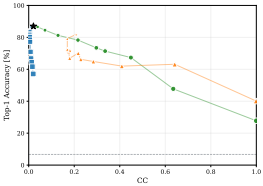
(g) D_{KL}



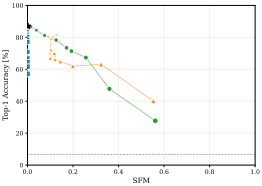
(h) IE



(i) 2D-IE



(j) CC



(k) SFM

Figure 5.1: Top-1 classification accuracy plotted against normalised security metrics across transformation scales. Each marker represents a specific scale parameter (σ for blur, B for LOI, N for CLM), with marker size proportional to transformation intensity. The baseline (plain images) is included for reference. The legend in Figure 5.1(a) applies to all subplots.

Table 5.1: Top-1 test accuracy and selected security metrics (SSIM, LPIPS, CC, and 2D-IE) for each transformation family and scale ↑ / ↓ indicates whether higher/lower values indicate security.

Name	**Top-1** ↑	**SSIM** ↓	**LPIPS** ↑	**|CC|** ↓	**2D-IE** ↑
Baseline					
Plain	87.00	1.00 ± 0.00	0.00 ± 0.00	0.98 ± 0.02	3.89 ± 0.89
Gaussian Blurring (σ)					
Blur-2	84.45	0.88 ± 0.07	0.24 ± 0.11	0.99 ± 0.01	3.25 ± 0.69
Blur-4	83.55	0.73 ± 0.14	0.44 ± 0.16	0.99 ± 0.01	2.76 ± 0.54
Blur-8	82.05	0.61 ± 0.16	0.59 ± 0.18	0.99 ± 0.01	2.21 ± 0.40
Blur-16	78.65	0.55 ± 0.17	0.67 ± 0.16	0.99 ± 0.01	1.62 ± 0.32
Blur-30	70.95	0.52 ± 0.17	0.72 ± 0.15	0.99 ± 0.01	1.09 ± 0.27
Blur-40	66.80	0.50 ± 0.17	0.74 ± 0.15	0.99 ± 0.01	0.88 ± 0.24
Blur-50	64.45	0.50 ± 0.18	0.75 ± 0.14	0.99 ± 0.02	0.73 ± 0.22
Blur-60	61.65	0.49 ± 0.18	0.76 ± 0.14	0.98 ± 0.02	0.62 ± 0.21
Blur-70	57.10	0.49 ± 0.18	0.76 ± 0.14	0.98 ± 0.02	0.53 ± 0.20
LOI (B)					
LOI-2	86.55	0.92 ± 0.04	0.23 ± 0.10	0.96 ± 0.03	3.95 ± 0.89
LOI-4	84.50	0.74 ± 0.11	0.49 ± 0.13	0.93 ± 0.05	4.27 ± 0.90
LOI-8	81.25	0.52 ± 0.16	0.67 ± 0.15	0.87 ± 0.08	4.72 ± 0.91
LOI-16	78.30	0.36 ± 0.17	0.83 ± 0.18	0.78 ± 0.11	5.18 ± 0.90
LOI-26	73.45	0.28 ± 0.16	0.96 ± 0.18	0.70 ± 0.12	5.47 ± 0.89
LOI-32	71.40	0.25 ± 0.16	1.02 ± 0.19	0.66 ± 0.13	5.60 ± 0.87
LOI-52	67.35	0.18 ± 0.14	1.13 ± 0.18	0.55 ± 0.14	5.87 ± 0.83
LOI-104	47.75	0.10 ± 0.10	1.28 ± 0.16	0.36 ± 0.14	6.21 ± 0.75
LOI-416	27.75	0.02 ± 0.02	1.45 ± 0.11	0.00 ± 0.00	6.61 ± 0.59
CLM (N)					
CLM-0	81.90	0.29 ± 0.15	0.87 ± 0.08	0.79 ± 0.07	4.43 ± 0.90
CLM-312	72.50	0.13 ± 0.14	0.87 ± 0.05	0.83 ± 0.07	4.51 ± 0.95
CLM-364	71.90	0.20 ± 0.10	0.90 ± 0.05	0.82 ± 0.07	4.70 ± 0.91
CLM-384	71.05	0.14 ± 0.09	0.93 ± 0.06	0.82 ± 0.06	4.77 ± 0.85
CLM-390	66.95	0.17 ± 0.08	0.93 ± 0.05	0.82 ± 0.06	4.83 ± 0.84
CLM-400	70.10	0.13 ± 0.06	0.98 ± 0.05	0.78 ± 0.06	5.07 ± 0.77
CLM-408	64.90	0.05 ± 0.02	1.03 ± 0.04	0.72 ± 0.05	5.50 ± 0.62
CLM-412	62.00	0.01 ± 0.00	1.10 ± 0.04	0.59 ± 0.04	6.20 ± 0.36
CLM-415	40.15	0.01 ± 0.00	1.58 ± 0.06	0.00 ± 0.00	6.96 ± 0.00

- CLM achieves near-ideal security metrics at $N = 415$ at the cost of model utility (40.15%). Decreasing N reveals more image structure and improves model utility (up to 81.90% at $N = 0$), with security metrics moving away from target values, demonstrating a strong decrease in image security as classification performance increases.

Overall, transformations that achieve high security tend to severely degrade model utility, whereas effective model training aligns with lower-security settings. Reported security metrics do not constitute cryptographic guarantees; in practice, transformation-dependent simulated attacks (e.g. , chosen-plaintext or reconstruction attacks) should complement these metrics.

6 Conclusion

This report quantified accuracy-security trade-offs for Gaussian blurring, LOI, and CLM in image classification. Results show that preserving coarse spatial structure is sufficient to maintain competitive top-1 accuracy for animal species classification, even when fine details and local geometry are substantially disrupted. This reveals a core limitation: single-dataset image classification is overly simplistic to serve as a reliable indicator of model utility. Decisions are dependent on global colour and low-frequency structures that do not transfer to other inference tasks. While the presented results are dataset- and architecture-specific, the observed dependence on structural preservation is expected to generalise qualitatively to other convolutional vision models, although quantitative thresholds may differ.

Classification performance additionally depends on the dataset and task. Species recognition may tolerate the loss of fine details, whereas fine-grained classification tasks, such as person re-identification, may depend on the discarded details. Transformations that retain global layout but disrupt local structure do leave top-1 accuracy intact, as shown. However, performance may, nonetheless, collapse when evaluating fine-grained tasks. Similar relationships may also appear in geometry-sensitive tasks (e.g., object or key-point detection and semantic segmentation) that rely on local boundaries, as currently ongoing work suggests.

These observations need to be connected to PET constraints outlined throughout the report. SMPC and FHE introduce latency, memory, and numerical restrictions, making the evaluation of model utility impractical for tasks beyond simple image classification. Utility assessment must consider discriminative proxy tasks while remaining feasible within PET constraints. Image-level security metrics are informative for visual obfuscation but should be complemented by attack-based evaluations. Furthermore, the security of trained models themselves must be considered, as they may leak sensitive attributes.

In summary, image classification alone systematically overstates model utility under structural image transformations and disregards features crucial for fine-grained and geometry-dependent tasks. The field of PPML requires a complementary suite of efficient proxy tasks and image- as well as model security metrics to evaluate task-agnostic privacy-utility trade-offs and compare PETs in a reproducible manner.

References

[1] Verma Aashman, Jain Shreshth, and Agrawal Khanak. *Animal Species Classification - V3*. Version 4. Jan. 25, 2023. URL: `https://www.kaggle.com/datasets/utkarshsaxenadn/animal-image-classification-dataset/data` (visited on 10/28/2025).

[2] Martin Abadi et al. "Deep Learning with Differential Privacy". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS '16. Vienna, Austria: Association for Computing Machinery, 2016, pp. 308–318. ISBN: 9781450341394.

[3] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. "Completeness theorems for non-cryptographic fault-tolerant distributed computation". In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*. STOC '88. Chicago, Illinois, USA: Association for Computing Machinery, 1988, pp. 1–10. ISBN: 0897912640.

[4] Franziska Boenisch et al. “When the Curious Abandon Honesty: Federated Learning Is Not Private”. In: *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)* (2021), pp. 175–199.

[5] Nicholas Carlini et al. “Membership Inference Attacks From First Principles”. In: *2022 IEEE Symposium on Security and Privacy (SP)*. 2022, pp. 1897–1914.

[6] Alexander Chang and Benjamin M. Case. “Attacks on Image Encryption Schemes for Privacy-Preserving Deep Neural Networks”. In: *ArXiv* abs/2004.13263 (2020).

[7] Qi Chen et al. “A Survey on an Emerging Area: Deep Learning for Smart City Data”. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 3.5 (2019), pp. 392–410.

[8] Cynthia Dwork. “Differential Privacy”. In: *Automata, Languages and Programming*. Ed. by Michele Bugliesi et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12. ISBN: 978-3-540-35908-1.

[9] Cynthia Dwork and Aaron Roth. “The Algorithmic Foundations of Differential Privacy”. In: *Foundations and Trends in Theoretical Computer Science* 9.3-4 (Aug. 2014), pp. 211–487.

[10] Raghida El Saj et al. “Privacy-Preserving Deep Neural Network Methods: Computational and Perceptual Methods—An Overview”. In: *Electronics* 10.11 (2021). ISSN: 2079-9292.

[11] Jonas Geiping et al. “Inverting Gradients - How easy is it to break privacy in federated learning?” In: *NeurIPS*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 16937–16947.

[12] Craig Gentry. “Fully homomorphic encryption using ideal lattices”. In: *Symposium on the Theory of Computing*. 2009.

[13] Alejandro Guerra-Manzanares et al. “Privacy-Preserving Machine Learning for Healthcare: Open Challenges and Future Perspectives”. In: *Trustworthy Machine Learning for Healthcare*. Ed. by Hao Chen and Luyang Luo. Cham: Springer Nature Switzerland, 2023, pp. 25–40. ISBN: 978-3-031-39539-0.

[14] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *CVPR*. 2016, pp. 770–778.

[15] Ismat Jarin and Birhanu Eshete. “PRICURE: Privacy-Preserving Collaborative Inference in a Multi-Party Setting”. In: *Proceedings of the 2021 ACM Workshop on Security and Privacy Analytics*. IWSPA ’21. Virtual Event, USA: Association for Computing Machinery, 2021, pp. 25–35. ISBN: 9781450383202.

[16] Nazish Khalid et al. “Privacy-preserving artificial intelligence in healthcare: Techniques and applications”. In: *Computers in Biology and Medicine* 158 (2023), p. 106848. ISSN: 0010-4825.

[17] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *ICLR,San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio andYann LeCun. 2015.

[18] Max Kirchner et al. *Federated EndoViT: Pretraining Vision Transformers via Federated Learning on Endoscopic Image Collections*. 2026. arXiv: 2504.16612 [cs.CV].

[19] Hitoshi Kiya. *Compressible and Learnable Encryption for Untrusted Cloud Environments*. 2018. arXiv: 1811.10254 [cs.CR]. URL: https://arxiv.org/abs/1811.10254.

[20] Jan J. Koenderink and Andrea J. Van Doorn. “The Structure of Locally Orderless Images”. In: *Int. J. Comput. Vision* 31.2-3 (Apr. 1999), pp. 159–168. ISSN: 0920-5691.

[21] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. Tech. rep. University of Toronto, 2009.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012).

[23] Elif Nur Kucur et al. “Privacy-Preserving Machine Learning Techniques: Cryptographic Approaches, Challenges, and Future Directions”. In: *Applied Sciences* 16.1 (2026).

[24] Kieran G. Larkin. “Reflections on Shannon Information: In search of a natural information-entropyfor images”. In: *CoRR* abs/1609.01117 (2016). arXiv: 1609.01117.

[25] Y. Lecun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[26] Qian Lou et al. “Glyph: Fast and Accurately Training Deep Neural Networks on Encrypted Data”. In: *NeurIPS*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 9193–9202.

[27] Koki Madono et al. “Block-wise Scrambled Image Recognition Using Adaptation Network”. In: *AAAI WS*. 2020.

[28] K. Mahalakshmi and Sivakumar Nagarajan. “Comprehensive Review and Analysis of Image Encryption Techniques”. In: *IEEE Access* 13 (2025), pp. 109783–109813.

[29] April Pyone Maung Maung, Isao Echizen, and Hitoshi Kiya. “On the Security of Learnable Image Encryption for Privacy-Preserving Deep Learning”. In: *IEEE Access* 12 (2024), pp. 126415–126425.

[30] Brendan McMahan et al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Xiaojin (Jerry) Zhu. Vol. 54. JMLR Workshop and Conference Proceedings. JMLR.org, 2017, pp. 1273–1282.

[31] Payman Mohassel and Peter Rindal. “ABY3: A Mixed Protocol Framework for Machine Learning”. In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (2018).

[32] Karthik Nandakumar et al. “Towards Deep Neural Network Training on Encrypted Data”. In: *2019 IEEE/CVF CVPR Workshops*. 2019, pp. 40–48.

[33] Milad Nasr, Reza Shokri, and Amir Houmansadr. “Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning”. In: *2019 IEEE Symposium on Security and Privacy (SP)*. 2019, pp. 739–753.

[34] N.K. Pareek, Vinod Patidar, and K.K. Sud. “Image encryption using chaotic logistic map”. In: *Image and Vision Computing* 24.9 (2006), pp. 926–934. ISSN: 0262-8856.

[35] Cheolhee Park, Dowon Hong, and Changho Seo. “An Attack-Based Evaluation Method for Differentially Private Learning Against Model Inversion Attack”. In: *IEEE Access* 7 (2019), pp. 124988–124999.

[36] Le Trieu Phong and Tran Thi Phuong. “Differentially private stochastic gradient descent via compression and memorization”. In: *Journal of Systems Architecture* 135 (2023), p. 102819. ISSN: 1383-7621.

[37] Michela Prunella et al. “Deep Learning for Automatic Vision-Based Recognition of Industrial Surface Defects: A Survey”. In: *IEEE Access* 11 (2023), pp. 43370–43423.

[38] Hong Qin et al. “Cryptographic Primitives in Privacy-Preserving Machine Learning: A Survey”. In: *IEEE Trans. on Knowl. and Data Eng.* 36.5 (May 2024), pp. 1919–1934. ISSN: 1041-4347.

[39] Morteza SaberiKamarposhti, Amirabbas Ghorbani, and Mehdi Yadollahi. “A comprehensive survey on image encryption: Taxonomy, challenges, and future directions”. In: *Chaos, Solitons & Fractals* 178 (2024), p. 114361. ISSN: 0960-0779.

[40] Avital Shafran et al. “Crypto-Oriented Neural Architecture Design”. In: *ICASSP*. 2021, pp. 2680–2684.

[41] Jinhyun So et al. “Securing secure aggregation: mitigating multi-round privacy leakage in federated learning”. In: *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*. AAAI’23/IAAI’23/EAAI’23. AAAI Press, 2023. ISBN: 978-1-57735-880-0.

[42] Masayuki Tanaka. “Learnable Image Encryption”. In: *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*. 2018, pp. 1–2.

[43] Wei Tang et al. “Robust and Secure Aggregation Scheme for Federated Learning”. In: *IEEE Internet of Things Journal* 12.8 (2025), pp. 9701–9715.

[44] Florian Tramer and Dan Boneh. “Differentially Private Learning Needs Better Features (or Much More Data)”. In: *ICLR*. 2021.

[45] Sameer Wagh, Divya Gupta, and Nishanth Chandran. “SecureNN: Efficient and Private Neural Network Training”. In: *Privacy Enhancing Technologies Symposium*. (PETS 2019). Feb. 2019.

[46] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612.

[47] Runhua Xu, James B.D. Joshi, and Chao Li. “CryptoNN: Training Neural Networks over Encrypted Data”. In: *ICDCS*. 2019, pp. 1199–1209.

[48] Wencheng Yang et al. *Deep Learning Model Inversion Attacks and Defenses: A Comprehensive Survey*. 2025. arXiv: `2501.18934 [cs.CR]`.

[49] Dayong Ye et al. “One Parameter Defense—Defending Against Data Inference Attacks via Differential Privacy”. In: *IEEE Transactions on Information Forensics and Security* 17 (2022), pp. 1466–1480.

[50] Chengliang Zhang et al. “BatchCrypt: Efficient Homomorphic Encryption for Cross-Silo Federated Learning”. In: *USENIX ATC 20*. USENIX Association, July 2020, pp. 493–506. ISBN: 978-1-939133-14-4.

[51] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *CVPR*. 2018.

[52] Yancheng Zhang et al. “HEPrune: fast private training of deep neural networks with encrypted data pruning”. In: *NeurIPS*. NIPS ’24. Vancouver, BC, Canada: Curran Associates Inc., 2024. ISBN: 9798331314385.

[53] Xiaojiang Zuo et al. “FedViT: Federated continual learning of vision transformer at edge”. In: *Future Generation Computer Systems* 154 (2024), pp. 1–15. ISSN: 0167-739X.

The Synset Signset Rendering Pipeline – A Principle for Generating Synthetic Data for ML

Anne Sielemann

Fraunhofer Institute of Optronics, System Technologies
and Image Exploitation IOSB
Department Systems for Measurement, Control and Diagnosis (MRD)
anne.sielemann@iosb.fraunhofer.de

Abstract

The ever-increasing demand for training and testing data for machine learning applications motivates research and industry to progressively focus on data synthesis as an additional data source. However, designing appropriate synthesis pipelines involves many challenges: the resulting synthetic data is desired to be domain comprehensive, parameterizable, physically correct, and to possess a sufficient degree of realism. To meet these criteria, we encapsulated our experience from several of our previous works on synthetic data generation and published open-source synthetic datasets into the design of the Synset Signset Rendering Pipeline, a synthesis pipeline for the task of traffic sign recognition, which we explain and discuss in this work. The pipeline aims to combine the advantages of data-driven and analytical modeling. Therefore, it includes a GAN-based texture generation module to enable data-driven dirt and wear artifacts, analytical scene modulation with interfaces to physically-based renderers, and a postprocessing module that simulates camera and digitization artifacts. The pipeline's general concept is generic and thus adaptable to other use-cases. Furthermore, we exemplarily demonstrate how to wrap the rendering pipeline into a rendering service designed for external use.

1 Introduction

Synthetic data play an increasingly important role in training and testing *artificial intelligence* (AI) applications. Already in 2021, a report by Gartner predicted that ‘‘by 2030, most of the data used in AI will be artificially generated by rules, statistical models, simulations or other techniques’’ [21]. This can be attributed to the potential that synthetic data possess in addressing today's challenges of collecting (increasing amounts of) training and testing data for AI applications: the generation of synthetic data is considerably less time-consuming and, thus, less expensive than acquiring and labeling real-world data. For comparison, the authors of the established *Cityscapes Dataset* [3] state that pixel-level and instance-level semantic segmentation labeling, along with subsequent quality control, takes an average of more than 90 minutes per image, while the time required to render a synthetic image with a corresponding, highly accurate labeling mask is typically within the range of a few seconds to minutes [4, 5]. Moreover, synthetic data generation is a valuable source of data for risk-prone or dangerous domains, such as violence detection [19], for domains where high demands for privacy apply, such as medical imaging [15], or for domains where data is difficult to acquire due to rare occurrences, such as vehicle make and model recognition [31]. In addition, synthetic data generation offers the ability to systematically update generated datasets without a domain gap.

In general, methods for generating synthetic data can be classified into two categories. On the one hand, there are analytical approaches, such as simulations, that follow statistical models defined by domain experts. On the other hand, there exist data-driven approaches, such as generative AI, that derive the properties of synthetic data directly from real-world data distributions. Each of the methods includes advantages as well as individual challenges that must be addressed in order to fully leverage their potential. Compared to data-driven approaches, analytical approaches provide better controllability and parameterization, through which the scope of the data can be defined. This is especially important in safety-critical domains such as autonomous driving, where it is crucial to know the *Operational Design Domain* (ODD), which refers to the operational conditions under which an automated driving system is designed to drive safely. Furthermore, they are able to achieve a higher degree of physical

correctness since, e.g., lighting is calculated in a physically-based way and geometric shapes are preserved. However, a general challenge of using synthetic data is handling the so called *sim-to-real gap*, which refers to the domain gap between simulation-based synthetic data and real-world data. Quantifying this gap and thus determining if the synthetic data is ‘‘similar enough’’ to real-world data is still an active research topic [29]. In this regard, data-driven approaches have the advantage of being able to learn even highly complex real-world data distributions, which are difficult to model manually, thus achieving a higher degree of individuality and fidelity to the real-world data. But in turn, this presupposes having access to (labeled) real-world data with all their previously mentioned challenges. Moreover, it must be ensured that the real-world data from which the distributions are learned cover the target domain well and possess sufficient variability – both of which are difficult to measure. Works such as [24] show that even between real-world datasets designed for the same task, huge domain gaps can occur, resulting in a significant decline in performance during cross dataset evaluation.

Besides the training of *Machine Learning* (ML) applications, there are further use-cases in the field of testing and validation that have high demands on parameterizability as well as the degree of realism of synthetic data. Examples include robustness analyses of ML models to assess the stability of model prediction performance with respect to input perturbations [32], systematic evaluation of explainable AI (XAI) methods [30, 12], and at some point in the future, perhaps even the safety certification of safety-critical systems like automated vehicles [37].

Our approach to meeting the requirements is a combination of analytical and data-driven methods aimed at leveraging the strengths of both. The *Synset Signset Rendering Pipeline* is designed to generate synthetic images for the task of traffic sign recognition and thus consists of three building blocks: a GAN-based texture generation module to enable data-driven dirt and wear artifacts, a 3D scene generation and rendering module, and a post-processing module to simulate camera and digitization image artifacts. In general, the concept of the pipeline is generic and thus adaptable to other tasks. The choice has been made for the task of traffic sign recognition since this continues to have relevant applications in systems such as driver assistance, automated driving, and mapping. Therefore,

it remains the subject of active research, in particular, to address challenges such as corner cases and weather conditions [28]. Furthermore, there already exist datasets for this task, both synthetic [33, 32] and real [34, 26, 39, 8], which enable an objective comparison of the resulting synthetic images for common traffic sign classes. At the same time, the pipeline can offer added value by being able to generate rarely occurring or newly released traffic signs.

2 Related Work

This chapter provides an overview of related work: on the one hand, approaches to synthetic data generation for traffic sign recognition in Section 2.1; and on the other hand, published datasets in this research field in Section 2.2.

2.1 Synthetic Data Generation for Traffic Sign Recognition

Methods for generating synthetic data for the task of traffic sign recognition can be grouped into those that synthesize entire images from scratch and those that apply synthetic augmentations to real-world images. In both groups, there are analytical as well as data-driven approaches, as motivated in the introduction.

For analytical image synthesis from scratch, simulations such as *Carla*[1] [7] (cf. *CATERED* dataset [33]) or OCTAS®[2] (cf. *Synset Signset Germany* dataset [32]) are tools of choice. In other domains within the field of autonomous driving, synthetic datasets are generated using computer game engines such as *Unity*[3] or *Unreal Engine*[4] [23, 9, 38, 2]. Even computer games such as *GTA* can be sources of data [10, 14, 22], albeit extracting the ground truth data and bridging the domain gap can pose challenges. On the subject of data-driven image synthesis methods, to the best of our knowledge, no datasets have been published so far. However, literature exists on GAN based synthesis of traffic sign images [6].

[1] carla.org
[2] octas.org
[3] unity.com
[4] unrealengine.com

A comparatively simple approach to augmentation-based image synthesis for the task of traffic sign recognition is the random placement of augmented traffic sign templates onto real-world background images [35, 27]. In the process, augmentations such as affine transformations, motion blur, Gaussian blur, and color and brightness correction are applied to the traffic sign templates. Thereby, the level of realism and the variation of the resulting data are aimed to be increased. Some approaches enhance these simple templates by adding rendered or GAN-improved traffic signs on top of real-world background images [27]. Image synthesis via augmentation can also be achieved by having GANs work directly on existing images in order to improve them. Examples are [18] where GANs are used to improve the degree of realism of simple augmented images, or [28] where GANs are used to add damage, snow, or other kinds of disruptions to increase task difficulty.

2.2 Existing Traffic Sign Recognition Datasets

There is a wide range of traffic sign recognition datasets from all over the world. However, most of them are country-specific, which is why they cannot be directly compared regarding the achieved performance benchmarks. One of the most cited datasets is the *German Traffic Sign Recognition Benchmark* (GTSRB) [34] with 51 882 samples from 43 classes. It was published in 2011 and has since become an established benchmark dataset in this field. Other well-known datasets are the *Tsinghua-Tencent 100K* (TT100K) dataset [39] with 100 000 samples of 221 traffic sign classes captured in China, the *Mapillary Traffic Sign Dataset* (MTSD) [8] with 52 453 fully annotated and 53 377 partially annotated images of 400 traffic sign classes from all over the world, and the *CURE-TSR* dataset [36] with 2 206 106 samples of 14 traffic sign classes captured in Belgium. The *European Dataset* [26] is a special work that merges multiple smaller European traffic sign datasets, such as [34, 1, 25, 16, 11] and others, into a single comprehensive database. Overall, the European Dataset comprises 82 476 samples of 164 traffic sign classes. Although there exist several works on the generation of synthetic traffic sign data (cf. previous section), there are only a few synthetic datasets available, such as the *CATERED* dataset [33], which includes 94 478 samples of the same 43 classes as GTSRB, and *Synset Signset*

Dataset	Year	# Images	# Samples	# Classes	Region
GTSRB [34]	2011	51 882	51 882	43	Germany
TT100K [39]	2016	100 000	30 000	221	China
CURE-TSR [36]	2017	2 206 106	2 206 106	14	Belgium
European DS [26]	2018	82 476	82 476	164	Europe
fully annot. MTSD [8]	2020	52 453	257 541	400	Global
part. annot. MTSD [8]	2020	53 377	96 613	400	Global
CATERED [33]	2021	94 478	94 478	43	Germany
SSG [32]	2024	105 500	105 500	211	Germany

Table 2.1: Overview of the largest publicly available datasets for the task of traffic sign recognition (real-world top, synthetic datasets bottom) sorted by year of publication. Note that some of the listed datasets are not only designed for traffic sign recognition but also detection, whereby their number of samples is not equal to the number of images since some images contain several and some images contain no traffic sign samples.

Germany[5] (SSG) [32] with 105 500 samples of 211 German traffic sign classes. For better comparison, Table 2.1 provides the key data of the mentioned datasets. However, note that the diversity of datasets should not be assessed solely based on the number of samples since, in some datasets, the same traffic sign instances appear multiple times under nearly the same environmental conditions, as several images were taken from a mobile platform while passing by.

3 The Synset Signset Rendering Pipeline

The Synset Signset Rendering Pipeline is based on our previous work on the Synset Signset Germany (SSG) dataset [32], in which we presented a new approach for generating synthetic data for the use-case of traffic sign recognition using a fixed set of parameters. However, this work builds upon [32] by demonstrating, on the one hand, how this approach can be further improved regarding

[5] synset.de/datasets/synset-signset-ger

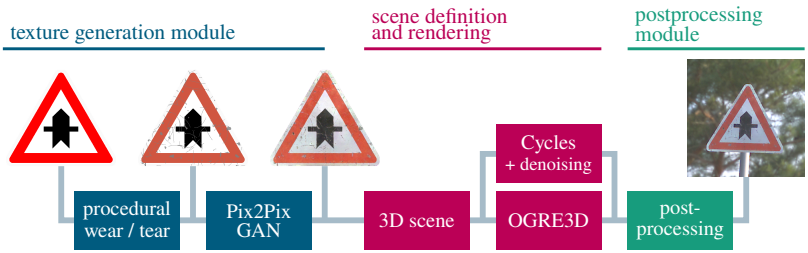


Figure 3.1: Overview of the Synset Signset Rendering Pipeline. The graphic is based on [32].

several aspects, and on the other hand, how each stage of the approach can be parameterized and how the whole pipeline can be wrapped into a web service, allowing authorized users to generate customized synthetic datasets.

Our data synthesis approach utilizes the Fraunhofer simulation platform OCTAS®, which can be customized for the respective simulation use-case due to its modular software architecture. Since OCTAS® is implemented in C++, most parts of the rendering pipeline were also realized in this programming language within supplementary OCTAS® plugins. The exception is the developed render service embedding, whose front-end is written in PHP and back-end in Python.

Before starting to explain the details of the Synset Signset Rendering Pipeline, the general concept of parametrization, which is used throughout the entire pipeline, will be explained in Section 3.1. The pipeline consists of three building blocks; an overview is provided in Figure 3.1. The following sections will stepwise walk through these building blocks: it starts with a texture generation module (Section 3.2), which is responsible for GAN-based texture synthesis. It is followed by the second building block, in which the resulting textures are used to create and render the 3D scene (Section 3.3). The last building block implements image postprocessing (Section 3.4) to simulate camera and digitization artifacts, which are intended to increase the degree of realism. To generate datasets as a service, an additional rendering service was built (Section 3.5) that wraps the rendering pipeline and enables external users to define and download customized traffic sign datasets.

3.1 Parameterization

One of the core ideas in developing the Synset Signset Rendering Pipeline is its configurability. Users should be able to adapt datasets to their needs and selectively vary individual parameters to investigate their influence. For this purpose, we have developed a dedicated concept for parameterization throughout the entire pipeline. A distinction is made between two different types of parameters: On one hand, *dataset parameters* that specify properties of the resulting dataset, such as the desired number of images or the traffic sign classes that should be included; on the other hand, *image parameters* that describe the properties of specific generated images, such as the resulting image resolution or the camera pitch angle.

From every building block of the pipeline, such image variables can be extracted. An image variable is then defined by specifying its identifier (a unique name), its data type, a default value, the allowed parameter range (if needed), and the allowed probability distribution(s) from which the variable could be sampled (none, uniform, or normal). Thereby, the default values are either set to a specific value or to a specific probability distribution instance if allowed (e.g., 1.0 or $\mathcal{U}[0.0, 1.0]$).

This enables users of the pipeline to parameterize their datasets in detail. For each parameter, there are up to three choices: to use the default setting, to set a constant value, or (where permitted) to define the probability distribution from which the variable is sampled. For a uniform distribution, the interval and whether its bounds are inclusive or exclusive need to be defined. For a normal distribution, the mean μ and variance σ^2 must be specified. The pipeline also allows for setting a random seed in order to reproduce specific configurations. Furthermore, single variables can be excluded from the random seed to generate datasets that differ only in one specific aspect.

A list of all dataset parameters is given in Table 3.1. The image parameters will be introduced separately for each building block in the following sections.

Parameter	Description
traffic signs	Definition of the traffic signs that should be included in the generated dataset. Can either be defined by template images or by the predefined indices used for the generation of SSG [32], for whose included traffic signs the pipeline permanently stores the corresponding template images.
images per sign	Number of images that should be synthesized. Can be defined per sign class or universally applicable to all classes.
naming convention	Resulting image files are named by the pattern *prefix_index*.png. Users can define the desired prefix and starting index which should facilitate expanding / merging datasets if desired.

Table 3.1: List of dataset parameters that can be defined for each traffic sign dataset generated with the Synset Signset Rendering Pipeline.

3.2 Texture Generation Module

The visual appearance of traffic signs is prominently affected by the degradation of the sign surface through wear, tear, vandalism, or color fading. Thus, the requirements placed on the approach for texture synthesis are (I) to reproduce these effects as realistically as possible, (II) to allow for parameterization and thus control over the degree of degradation, and (III) to be able to apply the approach to any kind of traffic sign, regardless of its size, color, or shape. Since the degradation effects are complex to model analytically, a data-driven approach has been chosen to meet requirement (I). However, to enable the parameterization of the approach, the pipeline's texture generation module comprises a procedural template image generation (cf. section 3.2.2), whose resulting template images are pixel-wise translated into realistically looking traffic sign textures by a *Generative Adversarial Network* (GAN) (cf. section 3.2.1). The modules are described in reverse order from how they appear in the pipeline since the GAN properties impose requirements on the procedural template image generation. After explaining the general concept, Section 3.2.3 introduces an enhanced texture generation module for advanced texture generation that is still a work in progress.

3.2.1 GAN-based Texture Synthesis

From the requirements we have placed on the texture generation module, the following requirements for the texture synthesis GAN can be derived: we aim to develop a GAN that can translate predefined template images of any kind of traffic sign into realistically looking traffic sign textures. Furthermore, it should be data-efficient and able to work with medium-level annotations to minimize the effort of the data collection process. Therefore, we chose not to train our approach on translating individual traffic sign classes, as this would require data for every sign class to be translated, making the approach labor-intensive in terms of data collection and unflexible with respect to new classes. Instead, the GAN should learn how the traffic sign material and possible defects appear and behave, regardless of the sign's shape or semantics. On one hand, this enables training to be carried out on patches rather than on entire images, thereby allowing the training dataset size to be increased. On the other hand, the GAN can thus generalize to signs of new physical sizes and shapes. But in turn, possible correlations between sign type and damages–such as urban traffic signs having more stickers and graffiti, and traffic signs located in forests having more dirt and moss lichens–are largely eliminated.

For training the GAN, we collected a dataset of over 200 images of worn traffic signs (cf. Figure 3.2(a)) with the kind support of the civil engineering department of the city of Karlsruhe in Germany (Tiefbauamt Karlsruhe). Through methods of classical image processing, color masks with black, white, and saturated colors, along with gray annotated defects, were extracted. However, this implies that gray is not supported as a color for traffic signs, which limits the number of representable traffic signs. The training images were generated by randomly cropping and rotating the original images into RGB image patches of size 256×256. To increase color variation, patches with shuffled RGB channels are added (cf. Figure 3.2(b)), as yellow, green, cyan, and purple hues are either underrepresented or entirely absent in the original dataset. Note that retroreflector patterns are excluded, and retroreflection is not simulated.

As presented in [32], we utilize a *Pix2Pix* [13] GAN network without the $1 \times 1 \times 512$ bottleneck layer. The complete generator architecture is visualized in Figure 3.3. The fully convolutional layout, in combination with the

(a) Exemplary images from the captured dataset of worn traffic signs.

(b) Examples of generated training patches with applied color variations. The left column shows the original colors, the other two columns color variations. The images are paired as ground truth left and extracted color mask right.

Figure 3.2: Generation of the training dataset for the texture synthesis GAN.

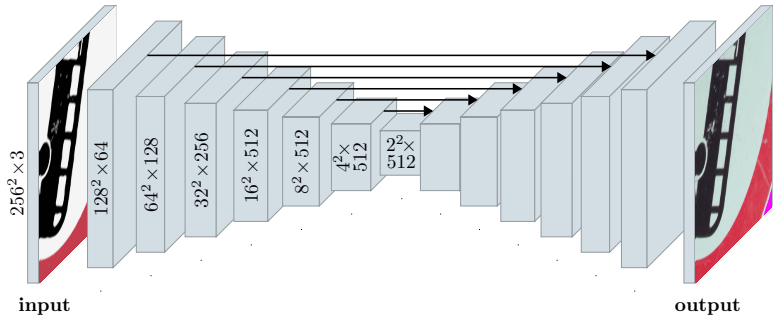


Figure 3.3: Overview of the Pix2Pix [13] generator architecture: a fully-convolutional encoder-decoder network structure with a $2 \times 2 \times 512$ central bottleneck. The figure is based on [32].

modified bottleneck, makes it possible to propagate input images whose dimensions are multiples of 256 through the network. Since we ensured through our data capturing and training process that the spatial resolution at which the GAN is trained is fixed at 8 px/cm, it is capable of generating textures for traffic signs of arbitrary size, but not of arbitrary resolution.

3.2.2 Procedural Template Image Generation

In [32], we presented the first version of our procedural template image generation, the results of which are used as inputs for the texture synthesis GAN. Therefore, the procedural template generation takes idealized traffic sign templates as input. These can, e.g., be obtained from Wikipedia, from which we collected the templates for SSG[6]. To simulate the effect of color fading, the idealized sign representations are separated into their black, white, red, green, blue, yellow, and orange components. Then, each color component is faded stochastically and homogeneously across each sign, following the stochastic distribution derived from real sign samples. Effects such as wear, tear, and vandalism, e.g., stickers, are added by an overlaid gray dirt mask. For SSG, we generated these masks by randomly placing rectangular gray shapes representing sticker residues and arbitrary shapes representing dirt or scratches obtained through a noise process. Although it is not possible to specify to the GAN which type of defect it should generate from which gray area, it has learned, based on the different shapes, to generate the appropriate defect in each case. However, this limits the shape and size of defects, such as sticker residues, which can only be rectangular, as in the training data. Example templates with the corresponding interfered GAN images are depicted in Figure 3.4.

The current pipeline includes an improved version of the gray dirt mask generation. It differs not only between sticker residues and dirt, but it can also create scratches and screw holes. The scratches are inserted via Bézier curves; the screw holes are represented by filled circles. Example images are depicted in Figure 3.5.

[6] de.wikipedia.ediaorg/wiki/Bildtafel_der_Verkehrszeichen_in_der_Bundesrepublik_Deutschland_seit_2017

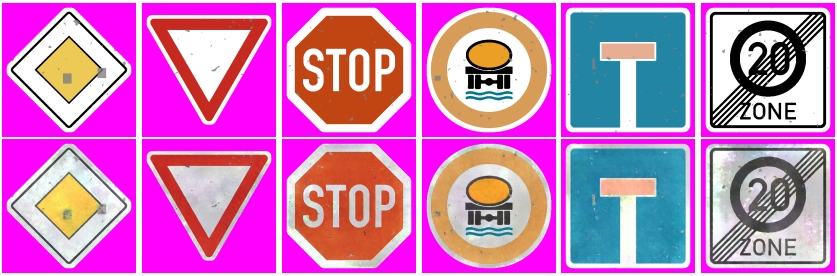


Figure 3.4: Results from the texture generation module used for generating SSG [32]. Upper row: procedurally generated template images; lower row: corresponding GAN interference results.

Figure 3.5: Results from the texture generation module of the current Synset Signset Rendering Pipeline. Upper row: procedurally generated template images; lower row: corresponding GAN interference results. In comparison to the version that was used to create SSG (cf. Figure 3.4), this enhanced version is able to produce dedicated scratches and screw holes.

Parameter	Description
defect types	Definition of defect types that should be applied. Possible are none, color variation, screw holes, scratches, dirt, sticker residues, and combinations of them.
defect strength	Factor between zero and one that steers the probability of defects appearing and thus the amount of defects.
noise	Users can decide weather slight noise should be added to the template images before propagating them through the GAN. If noise is added, resulting textures from the same template image do not look exactly equal.

Table 3.2: List of parameters that can be defined for the synthesis of the traffic sign textures.

Furthermore, we improved configurability by adding image variables (cf. section 3.1) to parameterize the type and strength of dirt. A detailed overview is provided by Table 3.2.

3.2.3 Advanced Texture Generation

The presented texture generation module exhibits several drawbacks regarding the defined requirements: it is unable to depict arbitrary traffic sign colors, such as gray, which is exclusively reserved for the defect annotations. Moreover, defects cannot be generated in arbitrary shapes and are difficult to parameterize since the GAN has only learned to differentiate between different types of defects based on their shape. This means that the learned shapes from the training dataset must be reproduced during template image generation.

For advanced texture generation, the traffic sign base color definition and the defect labels should consequently be separated. This also enables the distinction between multiple classes of defect labels, which improves configurability. We defined the label classes ‘‘dirt and scratches’’, ‘‘screw hole’’, ‘‘screw hole rust’’, ‘‘sticker and sticker residues’’, and ‘‘graffiti’’. In general, it could be beneficial to subdivide some of these classes, such as ‘‘dirt and scratches’’, into more fine grained categories like, e.g., ‘‘dirt’’, ‘‘scratches’’, and ‘‘cracks’’. However, it

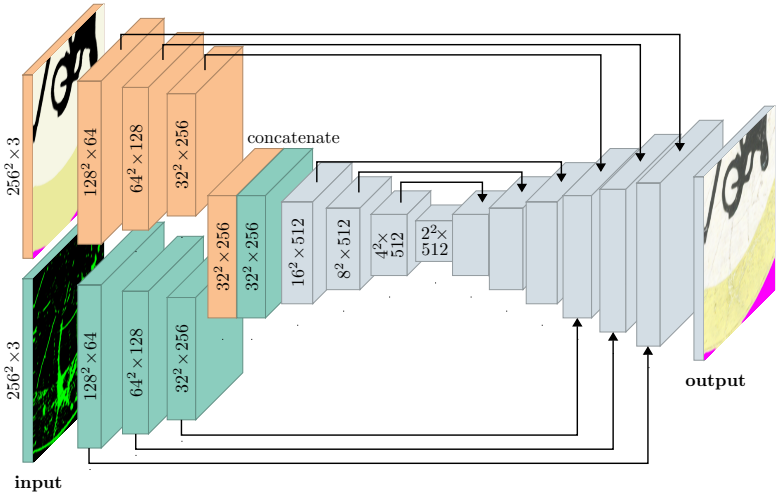


Figure 3.6: Overview of the modified Pix2Pix [13] generator architecture. Instead of one single combined RGB input image, it seperately takes the traffic sign color definition and the mask image and concatenates the calculated features after the third convolutional layer.

is necessary to find a balance between the increased configurability gained and the additional manual labeling effort it entails. Furthermore, it must be ensured that a sufficient number of training examples are available for each defined label class so that the GAN can properly learn and generalize each label instead of overfitting to only a small set of samples. The label distribution analysis showed that this is not the case for the ‘‘graffiti’’ class. Therefore, this class is included during training so that the GAN can understand that graffiti does not belong on a clean traffic sign. However, the class will not be requested during productive operations.

Due to the separate input of the traffic sign base color definition and the defect labels, two RGB images must be fed into the GAN instead of a single one. Therefore, the Pix2Pix architecture (cf. Figure 3.3) needs to be adapted. On the one hand, it would be possible to simply increase the input size to $256^2 \times 6$ and learn convolutional kernels with a depth of 6 within the first layer. On the other hand, both images could be separately convolved and concatenated in a deeper

network layer, as illustrated in Figure 3.6. Adapting the discriminator may also be beneficial, as it would allow it to evaluate not only the realism of the outputs but also the fidelity of the labels. Experiments on which architecture performs best, as well as the integration of the advanced texture generation module, are currently a work in progress.

3.3 Scene Definition and Image Rendering

After the texture generation, the second pipeline building block constructs the 3D scene within the simulation platform OCTAS®. It consists of the following components:

Traffic Sign The center of the scene is the traffic sign itself, which consists of the sign and its pole. The latter can be oriented vertically (as commonly seen) or horizontally (as on highways or before tunnels). Traffic signs can be modulated either exclusively through the generated textures or through a combination of the generated textures and class configuration files. These files allow for specifying how large the respective traffic sign of a specific class is (in meters) and which other traffic sign classes are permitted to appear above or below the respective traffic sign on a vertical pole in real-world road traffic. For the work on SSG [32] we considered the German traffic code / regulation StVO[7] (Straßenverkehrs-Ordnung) and real-world examples to collect this information.

Environment For the modulation of the environment and lighting, we apply *High Dynamic Range Image* (HDRI) maps. Therefore, our pipeline uses a database of more than 800 environment maps collected from Polyhaven[8]. The additional tags that the Polyhaven API provides (e.g., winter or road) allow for prefiltering the environment maps to the specific use case (e.g., as in [30]).

Camera A camera is added to capture the scene. It is always oriented towards the middle of the traffic sign; in the case of an applied rotation, the required translation for maintaining a fixed distance and focusing onto the traffic sign is calculated and applied.

[7] stvo2go.de/verkehrszeichen-wissensnetz
[8] polyhaven.com

(a) OGRE3D image (b) Cycles image (c) segmentation image (d) mask image

Figure 3.7: Exemplary resulting images rendered by OGRE3D and Cycles (with applied postprocessing) with corresponding segmentation image and mask image. Images stem from [32].

Occlusion Object To increase the degree of realism through shadows or occlusions, we add an occlusion object to the scene, whose position is randomly varied. For the traffic sign use-case, we decided to use a 3D tree model since trees often appear in real-world datasets and, moreover, produce diverse shadows.

For each of these components, image parameters (cf. Section 3.1) were defined, which are listed in Table 3.3.

For rendering the scene, OCTAS® currently supports the rasterization-based engine *OGRE3D*[9] as well as the path tracing engine *Cycles*[10] developed by the Blender project. These can be exchanged since OCTAS® provides clearly defined interfaces through a set of properties that follow the general framework of *physically-based rendering* (PBR) [20] and thus can be plugged into both rendering engines. To reduce the computing effort of Cycles, the *Nvidia AI Denoiser*[11] is applied to the resulting camera image after rendering, enabling the calculation of fewer samples to achieve a result of approximately equal quality. In addition to the camera image, a segmentation image and a mask image are also calculated. An example image rendered by both rendering engines, in combination with the corresponding segmentation and mask images, is provided in Figure 3.7.

[9] ogre3d.org
[10] cycles-renderer.org
[11] github.com/DeclanRussell/NvidiaAIDenoiser

Parameter	Description
image resolution	Desired image height and width.
rendering engine	Users can decide weather to use the rasterization-based render-engine OGRE3D[9] or the ray-tracing-based render-engine Cycles[10].
pole alignment	The traffic sign pole alignment can be set to horizontal, vertical, or randomly chosen.
camera/traffic sign orientation	Since for some use cases it is desired to rotate the camera around the static traffic sign (simulation of a passing maneuver) and in some use cases the camera should be static and the traffic sign rotates (differing traffic sign rotation with constant background) the user can specify a rotation by defining the roll, pitch, and yaw angle and choose weather to apply it onto the camera or traffic sign. It is possible to differentiate between a horizontal and vertical traffic sign pole regarding applied probability distributions. This makes it possible, e.g., to model that traffic signs mounted on horizontal poles are usually seen from a lower and straighter viewing angle from inside a car than those mounted on vertical poles.
constant materials	Specifies if the traffic sign and pole material properties (diffuse color, roughness, specular color) should slightly be varied between two frames.
additional signs	Describes if additional traffic signs above or below the traffic sign in focus should be simulated (for vertical pole only).
tree	It can be configured if an additional tree object should be placed in the scene to cast shadows or to occlude the traffic sign.
environment tags	Set of tags for prefiltering of the available environment maps.
environment rotation	Rotation angle in degrees by which the environment map should be rotated.

Table 3.3: List of image parameters that can be defined for the scene definition and image rendering process.

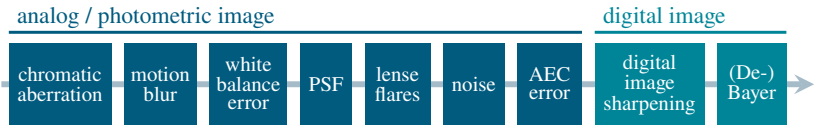


Figure 3.8: Overview of the applied postprocessing pipeline implemented in OCTAS®.

3.4 Postprocessing

Since path tracing and rasterization simulate ideal light transport per camera pixel, OCTAS® offers several postprocessing options to simulate camera and digitization effects. We define the postprocessing pipeline as visualized in Figure 3.8. The input to the pipeline is the rendered camera image result from either OGRE3D or Cycles, saved as a high dynamic range raster image.

First, we simulate artifacts caused by physical camera properties. Both *chromatic aberration* and *motion blur* are simulated by applying linear convolution kernels in an arbitrary direction with a specific length. Then, we simulate stochastic errors in *white balancing*. In [31] and [32], we therefore estimate the tint over the entire image. However, for images where the background has a strongly pronounced tint, the traffic sign still retains a tint after white balancing. This results in the introduced stochastic errors being inaccurate, as the actual resulting tint on the traffic sign is greater than parameterized. Thus, we implemented an alternative white balancing calculation by rendering an additional OGRE3D image with a solely white-textured traffic sign. This enables the calculation of the exact white point on the traffic sign surface, whereby the stochastic errors can be introduced with higher reliability. But this improved version of the white balancing calculation requires a higher computational effort since the extra image needs to be rendered. In the next step, we apply a *point spread function* (PSF), which is based on a Tamron M112FM35 35 mm lens, to represent focusing, lens optics, and diffraction through a mixture-of-Gaussian model. For details, refer to [31]. The PSF application is followed by the simulation of *lens flares* for visible light sources and the addition of *noise* [31, 32], which is given by a mixture of additive and multiplicative noise. The last applied camera defect is a stochastic error in *automatic exposure control* (AEC), leading to over- or

Parameter	Description
chromatic aberration	The direction and the strength of chromatic aberration.
motion blur	The direction and the strength of motion blur.
white point accuracy	Factor to regulate the accuracy of white balance.
lens flares	Can be disabled, or randomly be applied.
noise scale	Scaling factor of multiplicative image noise.
dark noise	Scaling factor of additive image noise.
aec error	Automatic exposure correction error range in f-stops.
bayer pattern	Can be disabled, or randomly be applied.

Table 3.4: List of image parameters that can be defined for customizing the postprocessing pipeline.

underexposed images. Note that we refrained from simulating image distortions due to the narrow field of view.

To simulate image defects caused by digitization (cf. [32]), we first add *digital image sharpening* effects by applying unsharp masking. The last step is the simulation of artifacts from Bayer BGGR bilinear *demosaicing*.

The extracted postprocessing image parameters are listed in Table 3.4. To get an impression of the individual effects, Figure 3.9 presents the images obtained by applying them separately.

3.5 Embedding as Rendering Service

With the Synset Signset Rendering Pipeline described in the previous sections, traffic sign images can be generated with comparatively high variation. An overview of different features is shown in Figure 3.10. To enable external users to access and use the pipeline, it was embedded in a rendering service named the *Synset Signset Rendering Service*. An overview of the service architecture is provided in Figure 3.11. We implemented a web interface hosted on a web server that allows authorized users to submit new traffic sign rendering jobs and to monitor their current status. These are defined through the specification of dataset and image parameters, e.g., in a JSON file. The web server administers

(a) chromatic aberration

(b) motion blur

(c) lense flares

(d) AEC error. Middle: no AEC error. Left: two stages of underexposure. Right: two stages of overexposure.

(e) noise

(f) digital image sharpening

(g) bayer pattern

Figure 3.9: Examples of isolated postprocessing effects. For a-c and e-g the whole image is depicted on the left and an image patch in which the respective effect is visible on the right.

(a) Individual traffic sign instances through color, defect, and pole diameter and material variation.

(b) Variation of pole orientation (vertical and horizontal) and for vertical case variation of pole protuberance. Images stem from [32].

(c) Semantically appropriate combination of upper and lower traffic signs. Images stem from [32].

(d) Occlusions. Images from [32].

(e) Different types of shadows. Images stem from [32].

(f) Generation of rare or imagined traffic signs.

(g) Challenging lighting conditions as night or backlight. Images stem from [32].

Figure 3.10: Showcase of features of the Synset Signset Rendering Pipeline.

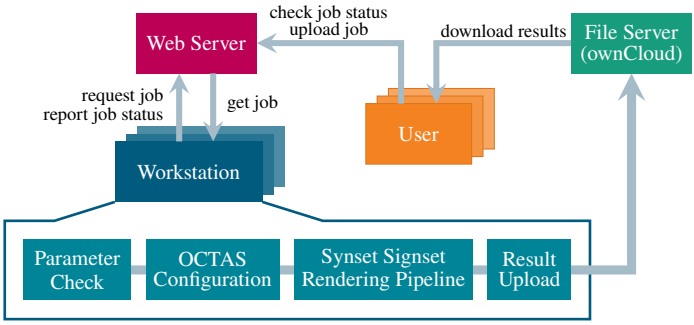


Figure 3.11: Overview of the Synset Signset Rendering Service that wraps up the Synset Signset Rendering Pipeline and allows for requesting datasets as a service.

all uploaded jobs and their statuses. Furthermore, it schedules queued jobs based on their priority and time of creation to the requesting workstations. To be scalable, the service was designed to support multiple workstations. Each of the active workstations first performs a check of the user defined dataset parameters. In the case of an invalid parameter definition, the workstation sends a list of the invalid parameters and the corresponding error messages to the web server, which then reports them to the relevant user. After that, it becomes available for a new task and processes job requests at a specified interval. If all parameters are valid, OCTAS® is configured based on the given parameters, and the Synset Rendering Pipeline is performed. During the generation process, the workstation regularly reports the current status and progress of the rendering pipeline to the web server. After rendering the requested dataset, the resulting files are uploaded to a file server; therefore, we use *ownCloud*[12]. The workstation notifies the web server about the successful job termination before requesting a new job. For successfully finished jobs, the rendering service provides a download link for the owning user.

In addition to the web interface, the rendering service also offers an API, by which the entire dataset generation process can be performed automatically: via

[12]owncloud.com

Python, the job definitions can be posted, the job status can be requested, and the images can be downloaded automatically. This enables the automation of use-cases such as robustness analyses [32], metric assessments [30], or fine-tuning on demand.

4 Evaluation

The evaluation is split into two parts: first, we evaluate the Synset Signset Rendering Pipeline in section 4.1 concerning the desired requirements, compared to other existing approaches. Then, the quality of the results is assessed by a qualitative and quantitative comparison in section 4.2.

4.1 Fulfillment of Requirements

To check whether our implemented approach is capable of fulfilling the desired requirements outlined in section 1, we analyzed the degree of fulfillment in comparison to other classes of synthetization methods discussed in section 2. An overview is provided in Table 4.1. Note that the quality of results is not included in this analysis.

Since the Synset Signset Rendering Pipeline is primarily simulation-based, it benefits from the advantages of simulations: it is well parameterizable, the rendering is physically-based, and it is capable of generating highly accurate ground truth data at the pixel and texture levels. In contrast, augmentations lack parameterizability. The foreground traffic sign and the background image can be individually augmented to increase variation, but basic object properties cannot be easily parameterized. Moreover, the physical properties of the foreground object and the background are usually correct when considered individually; however, in general, it is very complex to reconcile them with each other, to which DNNs could overfit during training. Augmentation based approaches enable the pixel-accurate generation of object-level ground truth data; a texture-level generation of ground truth data is only possible with additional manual labeling effort.

	Augmentation	Simulation	Generative AI	Ours
parameterizable	●○○	●●●	●●○	●●●
physically-based	●○○	●●●	●●○	●●●
highly accurate ground truth	●●○	●●●	●●○	●●●
individual instances	●●○	●○○	●●●	●●●
data efficiency	●●○	●●●	●○○	●●○
expandability to additional classes	●●○	●●●	●○○	●●●

Table 4.1: Summary of desired requirements for synthesis methods and the extent to which various approaches satisfy them.

Conditional generative AI methods provide a certain level of parameterization, as they allow boundary conditions to be defined that guide the image generation process. However, they generally cannot be parameterized in detail, such as simulations. Regarding physical correctness, generative AI approaches are quite impressive in learning and approximating statistical distributions from real-world data; but, especially with smaller amounts of training data, they often have difficulties correctly representing physical properties such as illumination or affine transformations of objects. In contrast to augmentations, generative AI methods are capable of providing texture-level ground truth data. However, slight pixel deviations may occur.

Our approach is also able to benefit from the advantages of generative AI, which is capable of generating individual instances of traffic signs. Through the integration of the texture generation module, each traffic sign generated with the Synset Signset Rendering pipeline represents an individual instance. This is generally not the case for simulations and for augmentations only if each foreground object image is used solely once, which increases data requirements.

A disadvantage of our approach is that it is not as data efficient as simulations. Simulations require no additional real-world data since their underlying probabil-

ity distributions are modeled by domain experts. While generative AI approaches and augmentations operate on larger amounts of data, our approach requires comparatively small amounts of data. However, for use in augmentations, the data must be labeled only at the object-level, while our approach needs more elaborate texture-level annotations, which is why we classify both approaches as moderately data-efficient.

For the maintenance of datasets, the ability to expand to additional classes, preferably without a domain gap, is essential. In simulation approaches, new classes can usually be integrated without a domain gap by only needing to collect the basic texture of a new sign. For augmentations, additional foreground images of the new traffic sign need to be collected and labeled. The most effort is required for generative AI approaches, where additional labeled training data for the new traffic sign class must be collected, and the entire network needs to be retrained or at least fine-tuned. Thereafter, it cannot be guaranteed that newly generated data will not exhibit a domain gap. Our approach has the advantage in this respect that the generation of the textures, especially in the concept of the advanced texture generation module, does not depend on the sign class. Therefore, new classes, as in simulations, can be added solely with a basic texture of the new traffic sign.

4.2 Quality of Results

Assessing the quality of synthetic images is a complex task [29]. A usual practice for evaluating synthetic datasets is to compare them directly with real-world datasets in the field, e.g., qualitatively or with regard to the task-specific performance metrics achieved by DNNs trained on the considered datasets in a cross-dataset evaluation. However, it is often neglected that real-world datasets can also be subject to certain domain gaps [24]. Therefore, especially for parameterizable approaches like ours, the comparison is highly dependent on the chosen parameters and the dataset used for comparison.

In our work on the SSG dataset [32], we used the predecessor of the Synset Signset Rendering Pipeline to create a synthetic dataset that includes the classes of the well-known GTSRB dataset [34] to be comparable to GTSRB itself and its synthetic reproduction, CATERED [33]. In the course of this, we attempted

(a) [32] **Signset** (b) [34] GTSRB (c) [33] CATERED (d) [35] Augment. (e) [18] Augment. (f) [18] GAN (g) [6] DCGAN (h) [6] LSGAN

Figure 4.1: Qualitative comparison of the SSG dataset (a) (generated with the predecessor of the Synset Signset Rendering Pipeline) with real-world data (b), entirely simulated data (c), augmented data (d) (e), and GAN generated data (f) (g) (h). For achieving a better comparability the example images were cropped to a similar area if necessary. The DCGAN, LSGAN, and WGAN samples stemming from [6] result from training 200 epochs respectively. The figure stems from [32].

to approximately match the chosen image parameter distributions to the visual appearance of GTSRB. Therefore, at this point, we refer to the qualitative comparison in Figure 4.1 and the quantitative cross-dataset test in Table 4.2 that we conducted in [32]. For details about the experimental setup, please refer to the original paper. The quantitative results indicate the high effectiveness of our rendering pipeline for generating synthetic traffic sign images for training and testing purposes.

5 Conclusion

In this work, the Synset Signset Rendering Pipeline tailored for the synthesis of traffic sign images for the task of traffic sign recognition was presented. It consists of three building blocks: a GAN-based texture generation module for creating individual and parameterizable traffic sign textures, a 3D scene generation and rendering module, and a postprocessing module for simulating

Eval. ► Train.▼	Signset Cycles	Signset OGRE	GTSRB	CATERED
Signset Cycles	99.5%	99.4%	98.3%	84.4%
Signset OGRE	99.6%	99.6%	98.2%	84.6%
GTSRB	89.4%	87.4%	99.9%	77.1%
CATERED	50.0%	48.6%	76.4%	86.1%

Table 4.2: Cross dataset evaluation from [32]. Reported is the top-1 accuracy of each combination of training and testing of ConvNeXt-Small [17] networks on the considered datasets.

camera and digitization effects. We demonstrated how to generalize the approach from [32] by making each pipeline stage highly parameterizable. Furthermore, we introduced an embedding of the pipeline as a Rendering Service that enables data generation via a web and python interface. Together, both improvements allow authorized external users to automatically generate customized traffic sign datasets, which enables the automation of use-cases such as robustness analyses [32], metric assessments [30], or fine-tuning / training of DNNs on demand. By combining the advantages of analytical and data-driven modeling, our approach is able to fulfill most of the stated requirements concerning the handling of the rendering service. In terms of result quality, the rendering service proved to be capable of generating images that are highly effective for training and testing DNNs.

The basic structure of the rendering pipeline has shown considerable promise and can, therefore, be adapted for other use cases as well. Nevertheless, there is room for improvement within the individual building blocks: the advanced texture generation module is still a work in progress, and its integration will improve parameterizability and texture quality. Especially for the use case of traffic sign recognition, it would be advantageous to include the simulation of retroreflection properties and retroreflection patterns in the future.

6 Acknowledgments

I would like to express my gratitude to Bence Máté Blaskovits for helping to implement new features for the Synset Render Service, as well as to Valentin Barner and Eric Schubert, who support me in developing the advanced texture generation module. Thank you for the kind support of the civil engineering department of the city of Karlsruhe in Germany (Tiefbauamt Karlsruhe) for capturing the real-world dataset of worn traffic signs. Furthermore, I am deeply grateful for the advice and support of my valued colleagues, Jens Ziehn and Masoud Roschani.

References

[1] Rachid Belaroussi et al. “Road Sign Detection in Images: A Case Study”. In: *2010 20th International Conference on Pattern Recognition*. IEEE. 2010, pp. 484–488.

[2] Yohann Cabon, Naila Murray, and Martin Humenberger. “Virtual KITTI 2”. In: *arXiv preprint arXiv:2001.10773* (2020).

[3] Marius Cordts et al. “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223.

[4] Maximilian Denninger et al. “BlenderProc: Reducing the Reality Gap with Photorealistic Rendering”. In: *16th Robotics: Science and Systems, RSS 2020, Workshops*. 2020.

[5] Maximilian Denninger et al. “BlenderProc2: A Procedural Pipeline for Photorealistic Rendering”. In: *Journal of Open Source Software* 8.82 (2023), p. 4901.

[6] Christine Dewi et al. “Yolo V4 for Advanced Traffic Sign Recognition With Synthetic Training Data Generated by Various GAN”. In: *IEEE Access* 9 (2021).

[7] Alexey Dosovitskiy et al. “CARLA: An Open Urban Driving Simulator”. In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[8] Christian Ertler et al. “Traffic Sign Detection and Classification around the World”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020.

[9] A Gaidon et al. “Virtual Worlds as Proxy for Multi-Object Tracking Analysis”. In: *CVPR*. 2016.

[10] Thomas Golda et al. “Image domain adaption of simulated data for human pose estimation”. In: *Artificial intelligence and machine learning in defense applications II*. Vol. 11543. SPIE. 2020, pp. 112–127.

[11] Cosmin Grigorescu and Nicolai Petkov. “Distance sets for shape filters and shape recognition”. In: *IEEE transactions on image processing* 12.10 (2003), pp. 1274–1286.

[12] Robin Hesse, Simone Schaub-Meyer, and Stefan Roth. “FunnyBirds: A Synthetic Vision Dataset for a Part-Based Analysis of Explainable AI Methods”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2023, pp. 3981–3991.

[13] Phillip Isola et al. “Image-to-Image Translation with Conditional Adversarial Networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[14] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, et al. “Driving in the Matrix: Can Virtual Worlds Replace Human-Generated Annotations for Real World Tasks?” In: *arXiv preprint arXiv:1610.01983* (2016). Sridhar, Sharath Nittur and Rosaen, Karl and Vasudevan, Ram.

[15] Lennart R Koetzier et al. “Generating Synthetic Data for Medical Imaging”. In: *Radiology* 312.3 (2024), e232471.

[16] Fredrik Larsson and Michael Felsberg. “Using Fourier Descriptors and Spatial Models for Traffic Sign Recognition”. In: *Image Analysis: 17th Scandinavian Conference, SCIA 2011, Ystad, Sweden, May 2011. Proceedings 17*. Springer. 2011, pp. 238–249.

[17] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, et al. “A ConvNet for the 2020s”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Feichtenhofer, Christoph and Darrell, Trevor and Xie, Saining. June 2022, pp. 11976–11986.

[18] Hengliang Luo, Qingqun Kong, and Fuchao Wu. “Traffic Sign Image Synthesis with Generative Adversarial Networks”. In: *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE. 2018, pp. 2540–2545.

[19] Muhammad Shahroz Nadeem et al. “Weapon Violence Dataset 2.0: A synthetic dataset for violence detection”. In: *Data in brief* 54 (2024), p. 110448.

[20] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. MIT Press, 2023.

[21] Leinar Ramos and Jitendra Subramanyam. “Maverick Research: Forget About Your Real Data – Synthetic Data Is the Future of AI”. In: *Gartner, Inc, Jun* (2021).

[22] Stephan R Richter et al. “Playing for Data: Ground Truth from Computer Games”. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. Springer. 2016, pp. 102–118.

[23] German Ros, Laura Sellart, Joanna Materzynska, et al. “The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vazquez, David and Lopez, Antonio M. June 2016.

[24] Héctor Corrales Sánchez, Noelia Hernández Parra, et al. “Are We Ready for Accurate and Unbiased Fine-Grained Vehicle Classification in Realistic Environments?” In: *IEEE Access* 9 (2021). Alonso, Ignacio Parra and Nebot, Eduardo and Fernández-Llorca, David, pp. 116338–116355. DOI: 10.1109/ACCESS.2021.3104340.

[25] Siniša Šegvić et al. “A computer vision assisted geoinformation inventory for traffic infrastructure”. In: *13th international IEEE conference on intelligent transportation systems*. IEEE. 2010, pp. 66–73.

[26] Citlalli Gamez Serna and Yassine Ruichek. “Classification of Traffic Signs: The European Dataset”. In: *IEEE Access* 6 (2018), pp. 78136–78148.

[27] Vlad Shakhuro, Boris Faizov, and Anton Konushin. “Rare Traffic Sign Recognition using Synthetic Training Data”. In: *Proceedings of the 3rd International Conference on Video and Image Processing*. 2019, pp. 23–26.

[28] Jayamani Siddaiyan, Kumar Ponnusamy, and Murali Lakshmanan. “Advancements in Traffic Sign Recognition for Autonomous Vehicles Using DCGAIN and Template Matching”. In: *Journal of Transportation Engineering, Part A: Systems* 151.12 (2025), p. 04025106.

[29] Anne Sielemann. “Towards Quantifying Simulated Image Sensor Data: A Survey and Discussion on GAN Evaluation Metrics”. In: *Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory*. 2025, p. 121.

[30] Anne Sielemann et al. “Measuring the Effect of Background on Classification and Feature Importance in Deep Learning for AV Perception”. In: *2025 IEEE International Automated Vehicle Validation Conference (IAVVC)*. 2025.

[31] Anne Sielemann et al. “Synset Boulevard: A Synthetic Image Dataset for VMMR”. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024.

[32] Anne Sielemann et al. “Synset Signset Germany: A Synthetic Dataset for German Traffic Sign Recognition”. In: *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*. 2024.

[33] Ilias Siniosoglou et al. “Synthetic Traffic Signs Dataset for Traffic Sign Detection & Recognition In Distributed Smart Systems”. In: *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE. 2021, pp. 302–308.

[34] Johannes Stallkamp et al. “The German Traffic Sign Recognition Benchmark: a multi-class classification competition”. In: *The 2011 international joint conference on neural networks*. IEEE. 2011, pp. 1453–1460.

[35] Alexandros Stergiou, Grigorios Kalliatakis, and Christos Chrysoulas. "Traffic Sign Recognition based on Synthesised Training Data". In: *Big Data and Cognitive Computing* 2.3 (2018), p. 19.

[36] D. Temel et al. "CURE-TSR: Challenging unreal and real environments for traffic sign recognition". In: *Neural Information Processing Systems (NeurIPS) Workshop on Machine Learning for Intelligent Transportation Systems*. 2017.

[37] Walther Wachenfeld and Hermann Winner. "Die Freigabe des autonomen Fahrens". In: *Autonomes Fahren: technische, rechtliche und gesellschaftliche Aspekte*. Springer Berlin Heidelberg Berlin, Heidelberg, 2015, pp. 439–464.

[38] Yue Yao, Liang Zheng, et al. "Simulating Content Consistent Vehicle Datasets with Attribute Descent". In: *Computer Vision–ECCV 2020: 16th European Conference, Proceedings, Part VI 16*. Springer. 2020, pp. 775–791.

[39] Zhe Zhu et al. "Traffic-Sign Detection and Classification in the Wild". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

Understanding Non-Stationary Systems through parametric Models of System-State Evolution

Benedikt Stratmann

Fraunhofer Institute of
Optronics, System Technologies and Image Exploitation (IOSB)
Karlsruhe, Germany
benedikt.stratmann@iosb.fraunhofer.de

Abstract

Non-stationary environments cause time-dependent shifts in joint data distributions, degrading predictive models. Current adaptive methods are able to detect and/or mitigate the influence of drifts, but fail to model the time-dependent change itself. We propose an adaptive learning framework that (i) estimates time-conditioned joint densities via Time-Weighted KDE, and (ii) learns a Continuous Normalizing Flow to transfer an earlier system state into a later one. This parametrized transfer function stores system change compactly and enables two adaptive modes: density-based inference and projection of model predictions into the future.

1 Introduction

Non-stationary environments are a central challenge for predictive modeling and system analysis in industrial and cyber-physical settings. As the joint data distribution evolves over time (concept drift), deployed models gradually become miscalibrated and their predictions degrade. Conventional countermeasures—such as combining drift detectors with periodic retraining—treat drift as a binary

event and often focus on restoring accuracy at the current time only. They typically assume piecewise-stationary regimes, discard or down-weight historical data, and provide limited insight into how and why the system is changing.

In many applications, however, understanding the temporal evolution of the system is as important as maintaining predictive performance. Operators may need to diagnose the source of drift, compare different drift episodes, or forecast future behavior. Existing adaptive methods only indirectly address these needs: they react to distribution changes but do not explicitly model the dynamics of the joint distribution itself. As a result, they offer limited support for tasks such as drift normalization, scenario analysis, or forward projection of model performance.

This work proposes an adaptive learning framework that directly models the time-dependent evolution of the joint distribution over features and targets. At any query time, we represent the system state by a time-conditioned joint density estimated via Time-Weighted Kernel Density Estimation (TW-KDE). We then learn a compact, invertible transition operator between states using Continuous Normalizing Flows (CNFs) based on Neural ODEs. The resulting parametric transition encodes how the system changes over time and can be applied both to probability densities and to individual samples.

From a user perspective, this yields two complementary benefits. First, the learned joint density supports *density-based inference* that directly provides predictions and uncertainties at a given time, without reliance on a specific discriminative architecture. Second, the learned transition operator enables *projection-based adaptation*: it can transport models or data between time points, allowing existing predictors to be reused in future system states or historical data to be mapped consistently to a common reference time.

The contributions of this work are:

- We define a time-conditioned representation of system state via TW-KDE over the joint feature–target space, providing normalized, per-time snapshots suitable for drift analysis and inference.

- We introduce a two-phase CNF-based transition operator that compactly encodes the evolution between system states and is invertible by construction.
- We demonstrate how this representation supports two adaptive modes: (i) density-based inference directly from the joint, and (ii) model-agnostic adaptation through temporal transport of data and predictions.
- We discuss how the proposed framework aligns with established desiderata for adaptive learning, including non-binary drift characterization, model-agnostic operation, and compact long-term storage of system change.

2 Related Work

2.1 Learning in Non-stationary Environments and Concept Drift

Learning under distribution shift—often formalized as covariate, label, or concept drift—has a long history in data streams and online learning. Comprehensive surveys review taxonomies, detectors, adaptive learners, and evaluation practices [10, 19, 25, 26, 18]. A prevalent paradigm is to combine a drift detector with an adaptive learner (active adaptation) or to continuously adapt without an explicit trigger (passive adaptation) [19, 18]. As also discussed in the introduction, this detector–adapter coupling introduces binary decisions, makes guarantees difficult, and can be data-inefficient by discarding or down-weighting history. Recent work further argues for richer characterizations of drift beyond binary alarms and for methods that support analysis and understanding of system change [23, 26, 25, 13, 14].

Examples for these drift detectors used for active adaptation include DDM [15], EDDM [1], ADWIN [2], among others. Popular adaptive learners for data streams employ windowing and ensembles: SEA [22], DWM [17] and Leveraging Bagging [3]. While effective for accuracy recovery, these methods typically do not deliver an explicit, quantitative representation of drift dynamics, limiting reusability of past data and interpretability of system changes.

2.2 Distributional Approaches to Drift Quantification

Modeling the data- or conditional-distribution directly enables quantitative comparison across time via statistical distances and hypothesis tests. Kernel Density Estimation (KDE) is a standard nonparametric choice; its time-dependent variants weight observations by recency to obtain a valid density at each time slice [12]. Parametric mixtures (e.g., GMMs) and stream density estimators with forgetting factors are also common. Distances such as KL, Jensen–Shannon, Hellinger, and χ^2 are widely used for drift quantification; Wasserstein/EMD is often preferred for its geometric sensitivity [20, 7, 4, 9]. Signature-mapping and cluster-based summaries trade fidelity for robustness and efficiency [20]. However, density estimators plus distances still yield pairwise dissimilarities rather than an explicit invertible map that transports one system state into another.

2.3 Normalizing Flows for Distribution Modeling and Transport

Normalizing Flows (NFs) provide invertible parameterizations of complex distributions via compositions of bijections trained by maximum likelihood and come in specialized varieties for a multitude of systems [16]. Continuous-time flows (CNFs) model transport via ODE dynamics and enable time-conditioned density evolution [5, 11].

In contrast to detectors producing binary signals, NFs yield a learned bijection f that (i) quantitatively normalizes drifting data to a reference state, (ii) transports distributions, samples, and even functions across states, and (iii) supports likelihood-based uncertainty propagation. This aligns with our goal to learn a transition operator T on a system-configuration space rather than only signaling regime changes.

2.4 Flows vs. Alternative Transport Paradigms

Optimal Transport (OT) provides a principled metric (Wasserstein) and dynamic formulations (Benamou–Brenier, displacement interpolation) for evolving distri-

butions; it is used in domain adaptation and dataset shift [20, 7]. While OT yields minimal-cost couplings, it does not directly provide a parameterized, invertible map with tractable likelihood for downstream inference. KDE/GMM-based approaches are simple but can struggle in high dimensions and do not furnish explicit transitions either. Transfer learning under covariate shift typically relies on importance weighting and representation alignment, but without an invertible system-level evolution model.

NFs bridge these gaps by combining (i) expressive, invertible mappings, (ii) exact likelihoods for density tracking, and (iii) conditional/continuous-time parameterizations for time-varying systems. Practical training uses maximum likelihood (KL) [16] or alternative objectives (e.g., Wasserstein-inspired losses) depending on application needs; spline couplings and CNFs often offer favorable trade-offs for smooth transitions [8, 5].

2.5 Positioning of This Work

Building on evidence that detection–adaptation pipelines can be data-inefficient and hard to calibrate [19, 18], we adopt a distributional representation of system configuration and learn explicit transitions between configurations via normalizing flows. Compared to time-weighted KDE [12] and distance-based drift quantification [20, 4, 7], our approach directly estimates invertible mappings that (i) enable drift normalization to a canonical state, (ii) reuse all historical data through learned transports, and (iii) support uncertainty propagation across states—addressing the shortcomings highlighted in [26, 25, 13, 14]. Our implementation leverages spline coupling flows and continuous-time parameterizations when appropriate [8, 5, 11], consistent with the presentation's architectural choices.

3 Method

Overview: Our method has two components:

(i) we infer the *System State* at any query time using a *Time-Weighted Kernel Density Estimator* (TW-KDE) over the joint feature–target space, and

(ii) we learn *state-to-state transitions* using *Continuous Normalizing Flows* (CNFs), composed to yield a compact, invertible description of temporal change.

System and data: We consider a joint data space

$$\mathbb{D} \;=\; X^n \times Y^m,$$

where X^n denotes the feature space and Y^m the target space. Data arrive as a time-stamped stream

$$\Omega \;=\; \{(d_i, t_i) \in \mathbb{D} \times \mathbb{R} | i \in \mathbb{N}\},$$

with $d_i = (x_i, y_i) \in \mathbb{D}$ and timestamps $t_i \in \mathbb{R}_{\geq 0}$.

Following prior work [21, 6], we describe the *system concept* at time t by a joint probability measure

$$C_t \;\in\; \mathcal{P}(\mathbb{D}),$$

where $C_t(x, y)$ denotes the true joint density of (x, y) at time t. We will sometimes write $p_t(x, y)$ for this density when convenient. We define *concept drift* as a change of this joint distribution over time:

$$C_t \;\neq\; C_{t+\Delta} \quad \text{for some } \Delta > 0.$$

For convenience, we use the term *System State* S_t to denote the (unknown) configuration of the underlying system whose observable behavior at time t is captured by the joint distribution C_t.

Throughout, we write

$$z = (x, y) \in \mathbb{R}^d, \qquad d = n + m,$$

for the concatenated feature–target vector.

Flow-based transition operator (composition first): To move between states, we realize the transition operator $T_{t_1 \to t_2} : \mathbb{S} \to \mathbb{S}$ via CNF composition: we learn a flow from a base distribution to the configuration at t_1, $\Phi_{0 \to t_1}$, and a second flow that transfers the configuration from t_1 to t_2, $\Phi_{t_1 \to t_2}$. The composition

$$\Phi_{0 \to t_2} = \Phi_{t_1 \to t_2} \circ \Phi_{0 \to t_1}$$

yields a compact, parametrized, and invertible description of the time-dependent change. It induces a transition operator that acts both on system states and on individual samples:

$$T_{t_1 \to t_2}(S_{t_1}) = S_{t_2}, \qquad z^{(t_2)} = T_{t_1 \to t_2}(z^{(t_1)}),\ z^{(t_1)} \in \mathbb{D},$$

with

$$T_{t_2 \to t_1} = T_{t_1 \to t_2}^{-1}.$$

We train these flows using samples drawn from the TW-KDE state snapshots near t_1 and t_2.

Definition of the parametric density $p_\Phi(z \mid t)$: Throughout, we denote by

$$p_\Phi(z \mid t), \quad z = (x, y), \quad t \geq 0, \tag{3.1}$$

the parametric time-conditioned joint density over z induced by the composed CNF flows that transport a base density p_0 through the sequence of flows to the state at time t. Concretely, $p_\Phi(z \mid t)$ is obtained by pushing forward p_0 through the composed flow

$$\Phi_{0 \to t} = \Phi_{t_1 \to t} \circ \Phi_{0 \to t_1},$$

where $\Phi_{0 \to t_1}$ maps the base density to the state at an intermediate time t_1 and $\Phi_{t_1 \to t}$ transports that state to time t. The CNFs are trained against TW-KDE snapshots near the target time t to approximate the true time-conditioned concept C_t. In practice, $p_\Phi(\cdot \mid t)$ serves as the density model used for inference, sampling, and downstream transport across time.

3.1 Time-Weighted Kernel Density Estimation (TW-KDE)

KDE fundamentals: Kernel density estimation provides a smooth, nonparametric approximation of an unknown density from samples. In one dimension, given observations $x_1, \ldots, x_N$ drawn independent and identically distributed (i.i.d.) from an unknown density p, the estimator with bandwidth $h > 0$ and kernel K is

$$\hat{p}_h(x) \ = \ \frac{1}{Nh} \sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right). \tag{3.2}$$

Bias decreases and variance increases as h becomes smaller; the converse holds for larger h. In d dimensions, with samples $\mathbf{x}_i \in \mathbb{R}^d$, the multivariate estimator uses a symmetric positive definite bandwidth matrix $H \in \mathbb{R}^{d\times d}$:

$$\hat{p}_H(\mathbf{x}) \;=\; \frac{1}{n}\sum_{i=1}^{n} K_H\left(\mathbf{x}-\mathbf{x}_i\right) \tag{3.3}$$

$$\text{where}\quad K_H(\dot{\mathbf{x}}) \;=\; |H|^{-1/2}\, K\left(H^{-1/2}\dot{\mathbf{x}}\right). \tag{3.4}$$

If K is normalized, then $\int K_H(\dot{\mathbf{x}})\,\mathrm{d}\dot{\mathbf{x}} = 1$ for any H.

Time as a conditioning variable:
For time-stamped data streams $\Omega = \{(\mathbf{x}_i, t_i)\}_{i=1}^{N}$, a naive approach is to treat time as an additional coordinate and estimate a joint density over $(\mathbf{x}, t)$:

$$\hat{p}(\mathbf{x},t) \;=\; \frac{1}{N}\sum_{i=1}^{N} K_{H_x}\left(\mathbf{x}-\mathbf{x}_i\right)\, K_{h_t}\left(t-t_i\right). \tag{3.5}$$

This is a valid joint density satisfying $\iint \hat{p}(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t = 1$.
However, it does not yield a properly normalized density at a fixed time, since generally $\int \hat{p}(\mathbf{x},t)\,\mathrm{d}\mathbf{x} \neq 1$. For adaptive learning and state comparison, we require the pointwise-in-time density $\hat{p}_t(\mathbf{x})$ to integrate to one for each t.

Time-Weighted MVKDE (TW-KDE) [12]: To obtain a normalized estimator at any query time t, we use a time kernel solely as a weight and renormalize. Let $w_i(t) := K_{h_t}(t-t_i)$ be the temporal weight assigned to sample $\mathbf{x}_i$ when estimating the density at t. The time-weighted estimator is

$$\hat{p}_t(\mathbf{x}) \;=\; \frac{\sum_{i=1}^{N} w_i(t)\, K_{H_x}\left(\mathbf{x}-\mathbf{x}_i\right)}{\sum_{i=1}^{N} w_i(t)}. \tag{3.6}$$

Because K_{H_x} integrates to one, the numerator integrates to $\sum_i w_i(t)$, which cancels with the denominator; hence $\int \hat{p}_t(\mathbf{x})\,\mathrm{d}\mathbf{x} = 1$ for all t. This simple renormalization turns temporal weighting into a proper conditional density estimate at each time without introducing bias from variations in sample density across time.

Why TW-KDE (vs. naive time-dependent KDE): The naive joint estimator normalizes over $(\mathbf{x}, t)$ and therefore allocates more probability mass to periods with denser sampling, making per-time comparisons ill-posed. TW-KDE

explicitly constructs a per-time, normalized snapshot $\hat{p}_t(\cdot)$ by weighting observations according to temporal proximity and then renormalizing. This allows meaningful comparisons of states across time, supports likelihood evaluation conditioned on t, and provides a principled basis for sampling representative data at a specific time.

Used kernels: Unless stated otherwise, we use Gaussian kernels for both the data and time dimensions. For the data, we take

$$K_H(\dot{\mathbf{x}}) \;=\; (2\pi)^{-d/2}\,|H|^{-1/2}\,\exp\left(-\tfrac{1}{2}\,\dot{\mathbf{x}}^\top H^{-1}\dot{\mathbf{x}}\right), \tag{3.7}$$

which yields smooth, differentiable estimates and allows anisotropic bandwidths through H. For time, we use a Gaussian weight

$$w_i(t) = \exp\!\left(-\frac{(t-t_i)^2}{2\sigma_t^2}\right),$$

with real-valued bandwidth σ_t, which provides symmetric recency weighting and is suitable when past and future observations around t should contribute similarly. When a strictly causal estimator is required, we instead use an exponential kernel

$$w_i(t) \;=\; \begin{cases} \exp\!\left(-\frac{t-t_i}{\tau}\right), & \text{if } t_i \le t, \\ 0, & \text{otherwise,} \end{cases} \tag{3.8}$$

which emphasizes recent history with a tunable forgetting time τ. Alternative kernels (e.g., Epanechnikov or triangular windows in time; compactly supported data kernels) are possible and may reduce variance or computational cost, but we found Gaussian choices to be robust and convenient for high-dimensional joint modeling [24]. This selection should be revisited with the complete system in place, as improvements in KDE inference time or accuracy can have compounding or even multiplicative effects for following CNF training procedures.

Joint modeling over features and targets: For adaptive learning, we estimate the joint density over all system variables $z = (x, y) \in \mathbb{R}^d$ with $d = n + m$. Given paired observations $z_i = (x_i, y_i)$, the time-weighted joint estimator is

$$\hat{p}_t(z) \;=\; \frac{\sum_{i=1}^{N} w_i(t)\,K_H\,(z - z_i)}{\sum_{i=1}^{N} w_i(t)}, \qquad z_i = (x_i, y_i). \tag{3.9}$$

This $\hat{p}_t(z)$ is a valid density for each fixed t. We take the *System State* at time t to be the configuration whose implied joint distribution matches $\hat{p}_t(\cdot)$. In practice, $\hat{p}_t$ serves directly as the state estimator used for inference, sampling, and construction of density ratios across time.

Bandwidth selection: We jointly fit the data bandwidth matrix H and the time bandwidth σ_t by maximum likelihood cross-validation based on leave-one-out (LOO) log-likelihood. To guarantee positive definiteness while allowing unconstrained optimization, we parameterize H by its lower-triangular Cholesky factor L with $H = LL^\top$. The parameter vector stacks the entries of L and the scalar time bandwidth. Given a set B of indices selected to cover the time span uniformly, the objective minimized at each iteration is the negative LOO log-likelihood,

$$\mathcal{L}(\theta; B) \;=\; -\sum_{j \in B} \log \hat{p}_{t_j}^{(-j)}(z_j), \tag{3.10}$$

$$\hat{p}_{t_j}^{(-j)}(z_j) \;=\; \frac{\sum_{i \neq j} w_i(t_j)\, K_H(z_j - z_i)}{\sum_{i \neq j} w_i(t_j)} \tag{3.11}$$

Here, the superscript $(-j)$ indicates that sample j is left out both in the numerator and the denominator. Each optimization step draws a set of evaluation indices whose timestamps are nearest to uniformly spaced grid points between the first and last observed times, ensuring that the LOO objective reflects the full temporal span.

We optimize the parameters with Adam and a ReduceLROnPlateau scheduler that decreases the learning rate when the objective plateaus. Initialization uses a user-specified H (via its Cholesky factor) and an initial positive time bandwidth; after optimization, the stored H is reconstructed as $LL^\top$ and the optimized time bandwidth is applied. In cases where the sum of time weights at a query time is numerically negligible, we enlarge σ_t to ensure stability.

Practical considerations and usage: The per-time normalization in (3.6) and (3.9) ensures that each snapshot $\hat{p}_t$ can be compared across time without bias from uneven sampling density. This facilitates likelihood-based monitoring, quantification of drift via density ratios, and sampling of representative points near a given time for downstream models. When targets differ strongly in scale

from features, preprocessing to harmonize scales helps avoid undue influence of any subset of coordinates and yields more balanced state estimates.

Causality note for TW-KDE in this work: In the experiments reported here, TW-KDE uses symmetric Gaussian time weights, $w_i(t) = \exp\left(-\frac{(t-t_i)^2}{2\sigma_t^2}\right)$, which do not enforce strict causality. Consequently, statements about causal influence should be interpreted with this limitation in mind. We plan to explore one-sided exponential time kernels (or other causal weights) in future work to enforce causality in green-field training scenarios, which may modify density evolution and transport dynamics.

3.2 Neural-ODE CNF and two-phase transition operator

CNF fundamentals: We model state-to-state transformations with a continuous normalizing flow (CNF), where a neural vector field f_θ defines an ODE [5]

$$\frac{\mathrm{d}z(s)}{\mathrm{d}s} \;=\; f_\theta\big(z(s), s\big), \qquad s \in [0,1], \tag{3.12}$$

whose integration yields an invertible map $z(0) \mapsto z(1)$. Here, s is an internal flow parameter (the integration coordinate of the CNF) and is distinct from the physical time t used to index system concepts C_t. Along this path, densities evolve according to the instantaneous change-of-variables formula

$$\frac{\mathrm{d}}{\mathrm{d}s}\,\log p\big(z(s)\big) \;=\; -\,\mathrm{Tr}\left(\frac{\partial f_\theta}{\partial z}\big(z(s), s\big)\right). \tag{3.13}$$

Integrating (3.13) gives the log-likelihood under the flow as

$$\log p_{\text{data}}(z(1)) \;=\; \log p_0\big(z(0)\big) - \int_0^1 \mathrm{Tr}\left(\frac{\partial f_\theta}{\partial z}\big(z(s), s\big)\right) \mathrm{d}s \tag{3.14}$$

$$\text{with } z(1) = z_{data}, \tag{3.15}$$

where p_0 is a tractable base density. The mapping is exactly invertible by integrating the ODE backwards from $s = 1$ to $s = 0$ with the same solver and tolerances.

Transition operator via composition: We realize the transition $T_{t_1 \to t_2} : \mathbb{S} \to \mathbb{S}$ by composing two flows. The first flow $\Phi_{0 \to t_1}$ maps a simple base density to

the system state at t_1. The second flow $\Phi_{t_1 \to t_2}$ transports the state at t_1 to the state at t_2. Their composition

$$\Phi_{0 \to t_2} \;=\; \Phi_{t_1 \to t_2} \circ \Phi_{0 \to t_1} \tag{3.16}$$

provides a compact, invertible parameterization of temporal change. In particular,

$$T_{t_1 \to t_2}(S_{t_1}) \;=\; S_{t_2}, \qquad T_{t_2 \to t_1} \;=\; T_{t_1 \to t_2}^{-1}. \tag{3.17}$$

This factorization separates the representation of the state at a reference time ($\Phi_{0 \to t_1}$) from the transition dynamics ($\Phi_{t_1 \to t_2}$), improving modularity and reusability.

Two-phase training procedure: Training proceeds in two phases, mirroring the above factorization. In Phase 1, we fit the *initial flow* $\Phi_{0 \to t_1}$ by maximum likelihood against a standard normal base p_0. Concretely, we draw samples near t_1 from the TW-KDE snapshot $\hat{p}_{t_1}(z)$ and minimize the negative of (3.15), pushing $z(t_1) \sim \hat{p}_{t_1}$ through the reversed initial flow to get an approximation of $z(0)$ for which we accumulate the divergence integral. After convergence, we freeze $\Phi_{0 \to t_1}$.

In Phase 2, we train the *transition flow* $\Phi_{t_1 \to t_2}$ on samples near t_2 drawn from $\hat{p}_{t_2}$. The base density for Phase 2 is defined *by the frozen initial flow*: for any generated $z(t_1)$, we run $\Phi_{0 \to t_1}$ in reverse to generate the corresponding $z(0)$ which should be behaving as if it was drawn from the Gaussian base. We evaluate the likelihood of this being the case, i.e.,

$$p_{t_1}^{\text{base}}(z) \;=\; p_0\big(\Phi_{0 \to t_1}^{-1}(z)\big)\,\big|\det \nabla \Phi_{0 \to t_1}^{-1}(z)\big|, \quad z = (x, y). \tag{3.18}$$

Maximum-likelihood training of $\Phi_{t_1 \to t_2}$ then minimizes the negative log-likelihood of $z(t_2) \sim \hat{p}_{t_2}$ under this flow-defined base density. Upon convergence we obtain the transition operator $\Phi_{t_1 \to t_2}$, which can be composed with $\Phi_{0 \to t_1}$ or inverted as needed.

Connection to TW-KDE states: The CNFs are trained on samples drawn from the time-local densities $\hat{p}_{t_1}$ and $\hat{p}_{t_2}$ produced by TW-KDE (Section 3.1). This coupling provides a normalized, recency-aware target at each time and allows the learned transition $\Phi_{t_1 \to t_2}$ to reflect the true temporal displacement between

consecutive system states. During deployment, we can map samples or entire densities across time by integrating the learned flows forward or backward, enabling interpolation, extrapolation, and invertible state alignment across the timeline. The time continuous probability density represented by the TW-KDE at the reference points t_1 and t_2 is now completely encoded in the parametric CNFs, allowing for efficient storage and separation of basic system dynamics (density represented by the initial CNF) and system change driven by time dependency (transition CNF).

4 Adaptive Learning

Given a calibrated time-conditional joint density $p_\Phi(z \mid t)$ over $z = (x, y)$, we support two complementary, model-agnostic adaptation modes:

1. Density-based inference (Mode A) that reads predictions directly from the joint by conditioning on x at time t.
2. Two realizations of model-agnostic adaptation (Mode B) that transport information across time to leverage historical data without requiring time-specific architectures.

4.1 Mode A: Density-based inference

For any query (x, t), define the conditional over targets by

$$p_\Phi(y \mid x, t) \;=\; \frac{p_\Phi((x, y) \mid t)}{\int p_\Phi((x, y') \mid t)\,\mathrm{d}y'}. \tag{4.1}$$

Prediction proceeds according to task:

- Classification for possible label set Y^m:

  $$\hat{y} = \arg\max_{y \in Y^m} p_\Phi((x, y) \mid t)$$

  with confidence $p_\Phi(\hat{y} \mid x, t)$.

- Regression: return either the conditional mode

$$\hat{y} = \arg\max_{y} p_\Phi((x, y) \mid t)$$

(via gradient ascent or sampling) or the conditional mean

$$\hat{y} = \mathbb{E}[y \mid x, t]$$

estimated by conditional sampling.

Uncertainty is obtained from $p_\Phi(y \mid x, t)$ (e.g., entropy for classification or predictive intervals from quantiles for regression). This mode is architecture-independent and avoids detector-triggered retraining.

4.2 Mode B: Model-agnostic adaptation

Mode B offers two concrete realizations for leveraging past data and the time-evolving transition $T_{t_1 \to t_2}$, without requiring changes to the end-model architecture.

Realization B1 (Projection via transport using the t_1 model):

- Use a predictor modeled at time t_1, denoted f_{t_1}, which maps the state at t_1 to predictions (e.g., $y_{t_1} = f_{t_1}(x_{t_1})$).
- For a live input at time t_2, transport the time-t_2 state back to t_1:

$$z_1 = T_{t_2 \to t_1}(z_2), \quad z_2 = (x_2; \cdot),\ z_1 = (x_1; \cdot).$$

- Apply the t_1 model to obtain a prediction in the t_1 space:

$$\hat{y}_1 = f_{t_1}(x_1).$$

- Transport the prediction back to t_2:

$$\hat{z}_2 = T_{t_1 \to t_2}(x_1, \hat{y}_1), \quad \hat{y}_2 = (\hat{z}_2)_y.$$

- This yields a t_2-space prediction by reusing the t_1-trained model.

Realization B2 (One model at t_2 trained on transported past data):

- Transport all historical data samples up to time t_2 into the t_2-space. For each past time $t_i \leq t_2$, transport the sample $z_i = (x_i, y_i)$ to t_2:

$$z_i^{(t_2)} = T_{t_i \to t_2}(z_i).$$

- Assemble the transported dataset

$$\mathcal{D}^{(t_2)} = \{z_i^{(t_2)} = (x_i^{(t_2)}, y_i^{(t_2)})\}_{i:\, t_i \leq t_2}.$$

- Train a single model at time t_2, say f_{t_2}, on $\mathcal{D}^{(t_2)}$ to map $x^{(t_2)}$ to $y^{(t_2)}$.
- Use f_{t_2} for predictions at time t_2 in the t_2-space.

5 Preliminary Results

5.1 Method Properties

As mentioned in previous work [21] ,the desirable properties of adaptive learning methods go beyond suppressing model degradation. Here is a short summary of these desirable properties.

Det (Detect Concept Drift) Continuously monitor the data stream to identify changes in the joint concept C_t, i.e., $C_t \neq C_{t+\Delta}$.

AR (Adapt Real Drift) Update the predictive relationship when $C_t(y \mid x)$ changes (actual/real drift), not just $p(x)$ [21].

RID (Remove Invalid Data) Recognize and filter obsolete samples after actual drift:

$$C_t(y \mid x) \neq C_{t+\Delta}(y \mid x) \text{ while } C_t(x) = C_{t+\Delta}(x)$$

, preventing contamination of the current concept.

NI (Incorporate New Information) In virtual drift ($C_t(y \mid x) = C_{t+\Delta}(y \mid x)$, $C_t(x) \neq C_{t+\Delta}(x)$), retain relevant history while expanding coverage to new feature regions.

Acc (High Accuracy) Maintain low predictive error over the stream despite ongoing drift.

Expl (Explain Source of Drift) Provide insight into where and why the drift occurs (e.g., affected subspaces, likely causes), enabling actionable system understanding.

Fore (Forecast Drift Development) Anticipate the future evolution of drift (trend, recurrence, trajectory) to support proactive adaptation.

Comp (Compare Drift Types) Distinguish and relate drift patterns (sudden, gradual, recurring; virtual vs. actual) to prior cases for faster response.

MA (Model-Agnostic) Work with diverse base learners without tight architectural coupling, allowing the best stationary model to be used per application.

NBD (Non-Binary Detection) Move beyond yes/no flags to provide continuous drift severity, direction, and scope.

LMS (Limited Model Size) Keep memory and parameter growth bounded for long-term deployment (e.g., avoid unbounded ensembles), supporting edge/embedded use.

The transition-flow approach developed in this work enables parametric modeling of system-state evolution and addresses several of the above properties within a unified adaptive learning framework. In contrast to methods that focus solely on detecting non-stationarity [25, 26], the proposed representation supports drift analysis, forecasting, concept normalization, and graded (non-binary) characterization of drift.

5.2 Method accuracy

The presented initial flow and transition flow approach already works reliably and enables parametric modeling of time driven system state evolution. This enables the mentioned operation modes as adaptive learning methods, alongside some analysis possibilities. Given an exemplary evolving system state, as seen

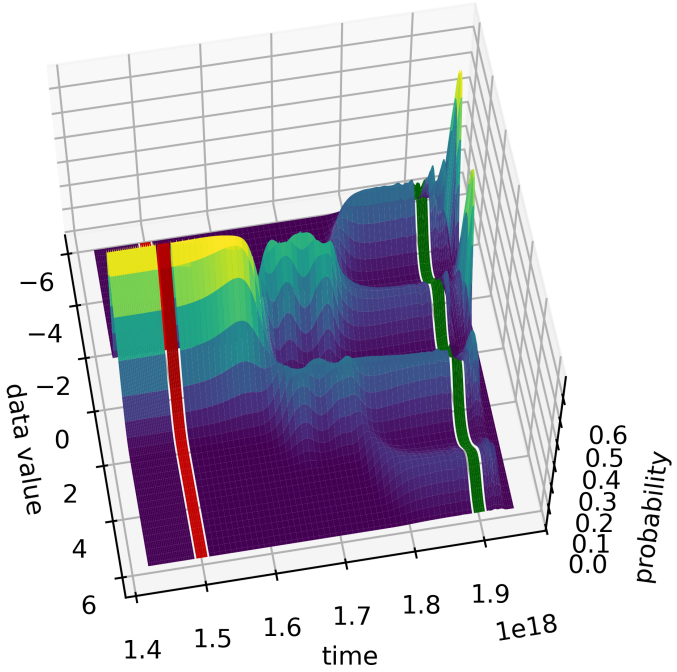


Figure 5.1: Exemplary time driven system state evolution for a one dimensional system. System distribution evolves from a unimodal to a bi-modal to a quad-modal distribution over time. Red marks the initial state, green marks the target state.

Table 5.1: ✓=Yes, ✗=No, ∼=Partial.

Method	Det	AR	RID	NI	Acc	Expl	Fore	Comp	MA	NBD	LMS
Passive retrain	✗	✓	✓	✓	∼	✗	✗	✗	✓	—	∼
ADWIN	✓	✓	∼	✓	∼	✗	✗	✗	✓	✗	✓
HDDDM	✓	✓	∼	✓	∼	✗	✗	✗	✓	✗	✓
CVFDT	✓	✓	∼	✓	∼	✗	✗	✗	✗	✗	✓
FIMT-DD	✓	✓	∼	✓	∼	✗	✗	✗	✗	✗	✓
Learn++.NSE	—	✓	∼	✓	∼	✗	✗	∼	∼	—	✗
SVM-ADWIN	✓	∼	∼	✓	∼	✗	✗	✗	∼	✗	∼
TW-KDE + CNF	✓	✓	∼	✓	∼	✓	∼	∼	✓	✓	∼

in Figure 5.1, with the initial state in red and the target state in green, we have an example of the possible accuracy of this approach in Figure 5.2. While this successfully enables evaluation of the system at t_1 and t_2 the evolution in between is currently not efficiently tracked.

The extent of this can be seen in Figure 5.3. While the initial and target system state fit well, the evolution in between does not track the intermediate bi-modal phase the reference scenario shows. As the flow is not trained to achieve this, this is to be expected and therefore is the next step of method development, to go from discrete checkpoint transition modelling to time continuous modelling.

6 Conclusion and Future Work

We introduced a compact, model-agnostic framework for understanding and adapting to non-stationary systems by (i) estimating time-local joint densities with TW-KDE and (ii) learning invertible, composable transitions with continuous normalizing flows (CNFs). The learned transition operator encodes system change explicitly and supports two adaptive modes: density-based inference directly from the joint and transport-based adaptation that reuses models or data across time. Preliminary experiments demonstrate accurate matching at end states and the viability of parametric storage of system evolution.

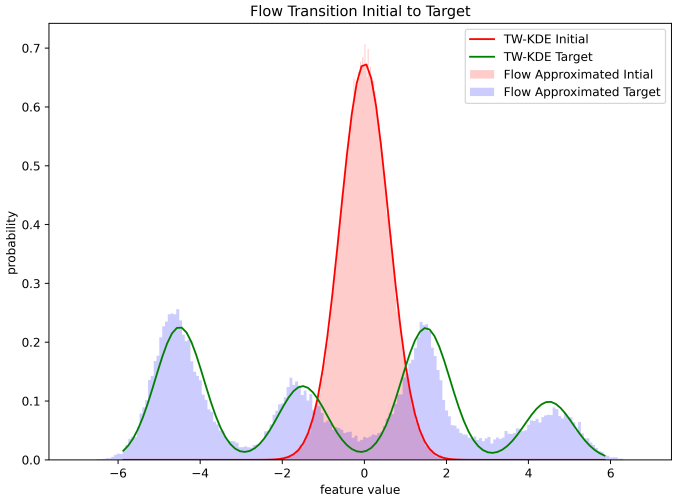


Figure 5.2: Exemplary initial system state to target system state approximation using the TW-KDE as reference distributions learned by the CNF.

Limitations: TW-KDE can scale poorly with dimensionality. The current two-flow composition matches endpoints but does not guarantee faithful intermediate evolution. The currently used symmetric time weights are non-causal, requiring adaptation for green field applications.

Future Work: We will pursue the following extensions. In this subsection, we denote by $p_\theta(z \mid t) \equiv p_\Phi(z \mid t)$ the time-conditioned density induced by the CNF parameters θ, in order to emphasize its dependence on θ in the optimization objectives.

Time-conditioned dynamics and loss over time Replace learning based on discrete checkpoints at initial and target states with optimization against a continuous-time reference of the evolution. Train the time-conditioned vector field $f_\theta(z, t)$ within the CNF to yield reference-accurate $p_\theta(z \mid t)$ for any t in a horizon. This can be achieved by minimizing the expected

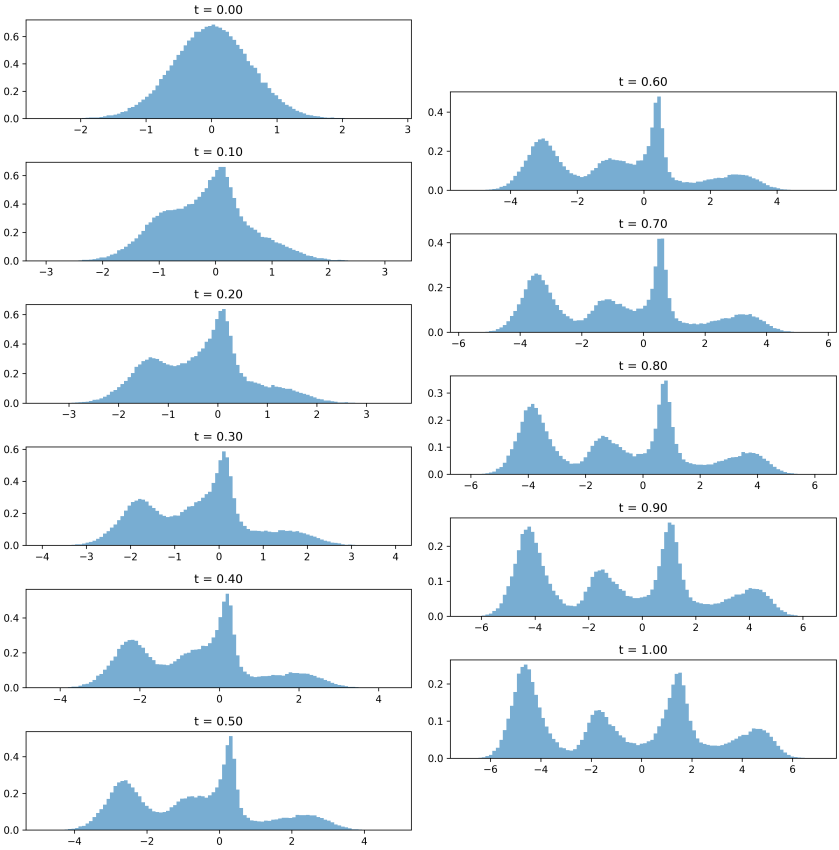


Figure 5.3: Incremental state evolution as modeled by the transition function $\Phi_{t_1 \to t_2}$, with partial evaluations in between t_1 and t_2

divergence over the time horizon:

$$\mathcal{L}(\theta) \;=\; \mathbb{E}_{t\sim\pi_K}\big[D_{\mathrm{KL}}\big(\hat{p}_t \,\|\, p_\theta(\cdot \mid t)\big)\big]$$

plus temporal smoothness regularization, e.g.,

$$\lambda_{\mathrm{smooth}}\,\mathbb{E}_{t\sim\pi_K}\,\mathbb{E}_{z\sim\hat{p}_t}\Big[\big\|\partial_t f_\theta(z,t)\big\|_2^2\Big],$$

to favor plausible evolution and accurate interpolation/extrapolation. Here, π_K is based on the time kernel used in the TW-KDE to ensure that the loss focuses evaluation in regions with sufficient data density.

Causal operation and online updates Introduce one-sided temporal weights (exponential forgetting) for TW-KDE and online training for the time-conditioned CNF with small-step updates, ensuring no leakage from future data.

Uncertainty and monitoring Use likelihoods and density ratios from $p_\theta(\cdot \mid t)$ for non-binary drift monitoring; derive predictive intervals from $p_\theta(y \mid x, t)$; and evaluate calibration over time.

By moving from discrete checkpoints to fully time-conditioned flow fields with temporally regularized training, we aim to deliver faithful interpolation/extrapolation of system change, causal deployment, and scalable adaptation that unifies analysis, forecasting, and prediction under distribution shift.

References

[1] Manuel Baena-Garc et al. “Early Drift Detection Method”. In: 2005. URL: https://www.semanticscholar.org/paper/Early-Drift-Detection-Method-Baena-Garc-Avila/2747577a61c70bc3874380130615e15aff76339e (visited on 09/09/2024).

[2] Albert Bifet and Ricard Gavaldà. “Learning from Time-Changing Data with Adaptive Windowing”. In: *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)*. Proceedings. Society for Industrial and Applied Mathematics, Apr. 26, 2007, pp. 443–448. ISBN: 978-0-89871-630-6. DOI: 10.1137/1.9781611972771.42. URL: https://epubs.siam.org/doi/10.1137/1.9781611972771.42 (visited on 09/12/2024).

[3] Albert Bifet et al. “New Ensemble Methods for Evolving Data Streams”. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’09. New York, NY, USA: Association for Computing Machinery, June 28, 2009, pp. 139–148. ISBN: 978-1-60558-495-9. DOI: 10.1145/1557019.1557041. URL: https://dl.acm.org/doi/10.1145/1557019.1557041 (visited on 09/18/2024).

[4] Jacob Burbea and C.Radhakrishna Rao. “Entropy differential metric, distance and divergence measures in probability spaces: A unified approach”. In: *Journal of Multivariate Analysis* 12.4 (1982), pp. 575–596. ISSN: 0047-259X. DOI: https://doi.org/10.1016/0047-259X(82)90065-3. URL: https://www.sciencedirect.com/science/article/pii/0047259X82900653.

[5] Ricky T. Q. Chen et al. “Neural Ordinary Differential Equations”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

[6] Gregory Ditzler et al. “Learning in Nonstationary Environments: A Survey”. In: *IEEE Computational Intelligence Magazine* 10.4 (Jan. 1, 2015), pp. 12–25. ISSN: 1556-603X. DOI: 10.1109/mci.2015.2471196.

[7] D.C Dowson and B.V Landau. “The Fréchet distance between multivariate normal distributions”. In: *Journal of Multivariate Analysis* 12.3 (1982), pp. 450–455. ISSN: 0047-259X. DOI: https://doi.org/10.1016/0047-259X(82)90077-X. URL: https://www.sciencedirect.com/science/article/pii/0047259X8290077X.

[8] Conor Durkan et al. “Neural Spline Flows”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/7ac71d433f282034e088473244df8c02-Paper.pdf.

[9] D.M. Endres and J.E. Schindelin. “A new metric for probability distributions”. In: *IEEE Transactions on Information Theory* 49.7 (2003), pp. 1858–1860. DOI: 10.1109/TIT.2003.813506.

[10] João Gama et al. “A Survey on Concept Drift Adaptation”. In: *ACM Computing Surveys* 46.4 (Jan. 1, 2014), pp. 1–37. ISSN: 0360-0300. DOI: 10.1145/2523813.

[11] Will Grathwohl et al. *FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models*. 2018. arXiv: 1810.01367 [cs.LG]. URL: https://arxiv.org/abs/1810.01367.

[12] Andrew Harvey and Vitaliy Oryshchenko. “Kernel density estimation for time series data”. In: *International Journal of Forecasting - INT J FORECASTING* 28 (Mar. 2012). DOI: 10.1016/j.ijforecast.2011.02.016.

[13] Fabian Hinder, Johannes Kummert, and Barbara Hammer. “Explaining Concept Drift by Mean of Direction”. In: *Artificial Neural Networks and Machine Learning – ICANN 2020: 29th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 15–18, 2020, Proceedings, Part I*. Ed. by Igor Farkaš, Paolo Masulli, and Stefan Wermter. Vol. 12396. Springer eBook Collection. Cham: Springer International Publishing; Imprint Springer, Jan. 1, 2020, pp. 379–390. ISBN: 978-3-030-61609-0. DOI: 10.1007/978-3-030-61609-0_30. URL: https://link.springer.com/chapter/10.1007/978-3-030-61609-0_30.

[14] Fabian Hinder et al. “Model-Based Explanations of Concept Drift”. In: *Neurocomputing* 555 (Jan. 1, 2023), p. 126640. ISSN: 09252312. DOI: 10.1016/j.neucom.2023.126640.

[15] João Gama et al. “Learning with Drift Detection”. In: (Sept. 29, 2004), pp. 286–295. DOI: 10.1007/978-3-540-28645-5_29.

[16] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. “Normalizing Flows: An Introduction and Review of Current Methods”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11 (Nov. 2021), pp. 3964–3979. ISSN: 1939-3539. DOI: 10.1109/tpami.2020.2992934. URL: http://dx.doi.org/10.1109/TPAMI.2020.2992934.

[17] J Zico Kolter and Marcus A Maloof. “Dynamic weighted majority: An ensemble method for drifting concepts”. In: *The Journal of Machine Learning Research* 8 (2007), pp. 2755–2790.

[18] Marilia Lima et al. “Learning Under Concept Drift for Regression—A Systematic Literature Review”. In: *IEEE Access* 10 (Jan. 1, 2022), pp. 45410–45429. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2022.3169785.

[19] Jie Lu et al. “Learning under Concept Drift: A Review”. In: *IEEE Transactions on Knowledge and Data Engineering* 31.12 (Dec. 2019), pp. 2346–2363. ISSN: 1558-2191. DOI: 10.1109/TKDE.2018.2876857. URL: https://ieeexplore.ieee.org/document/8496795 (visited on 09/04/2024).

[20] C. Rubner Y. Tomasi and L.J. Guibas. “The Earth Mover’s Distance as a Metric for Image Retrieval”. In: *International Journal of Computer Vision* 40 (2000), pp. 99–121.

[21] Benedikt Stratmann. *Advancing Adaptive Learning in Non-Stationary Environments: Challenges, Methods, and Future Directions*. en. 2025. DOI: 10.24406/publica-5012. URL: https://publica.fraunhofer.de/handle/publica/490222.

[22] W. Nick Street and YongSeog Kim. “A streaming ensemble algorithm (SEA) for large-scale classification”. In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’01. San Francisco, California: Association for Computing Machinery, 2001, pp. 377–382. ISBN: 158113391X. DOI: 10.1145/502512.502568. URL: https://doi.org/10.1145/502512.502568.

[23] Pingfan Wang et al. “QuadCDD: A Quadruple-based Approach for Understanding Concept Drift in Data Streams”. In: *Expert Systems with Applications* 238 (Jan. 1, 2024), p. 122114. ISSN: 09574174. DOI: 10.1016/j.eswa.2023.122114.

[24] Shaoping Wang et al. “Robust kernels for kernel density estimation”. In: *Economics Letters* 191 (2020), p. 109138. ISSN: 0165-1765. DOI: https://doi.org/10.1016/j.econlet.2020.109138. URL: https://www.sciencedirect.com/science/article/pii/S0165176520301105.

[25] Geoffrey I. Webb et al. “Characterizing Concept Drift”. In: *Data Mining and Knowledge Discovery* 30.4 (Jan. 1, 2016), pp. 964–994. ISSN: 1384-5810. DOI: 10.1007/s10618-015-0448-4. URL: https://link.springer.com/article/10.1007/s10618-015-0448-4.

[26] Geoffrey I. Webb et al. *Understanding Concept Drift*. Jan. 1, 2017. DOI: 10.48550/arXiv.1704.00362. Pre-published.

Towards Quantifiable Feedback in Cybersecurity Risk Assessment

Jonas Vogl

Production Lab
KASTEL
Karlsruhe Institute of Technology (KIT), Germany
Jonas.Vogl@kit.edu

Abstract

In cybersecurity, as in any other domain, gathering feedback on operations is necessary to evaluate and improve processes. In cyberecurity [2] finds a lack of quantifiable feedback for risk assessment processes. This work examines whether the brier score can be used to gather feedback and evaluate some methods from risk assessment. Three example methods used in cybersecurity are examined: the Common Vulnerability Scoring System (CVSS), the Exploit Probability Scoring System (EPSS) and IEC 62443 risk assessment. While EPSS can be evaluated using the brier score, CVSS lacks some of the requirements to apply the brier score. Standards such as IEC 62443 are found to be too abstract to be directly evaluated in this way.

1 Introduction

In any process, a functioning feedback loop is vital, to know if the current process works as intended, to keep the process to date, and improve it. Cyber Security is no exception here especially with its ever evolving adversaries.

For example the SANS CTI Report 2025 [2] identifies gathering feedback as one of the ”opportunities of growth” for Cyber Threat Intelligence with only 55% of respondents of the survey measuring the effectiveness of their CTI Program. Among those that measure effectivenes the most common method is gathering feedback through meetings (84.3%), followed by surveys and emails and comparison with baseline metrics (all at 74.5%).

The CTI Report argues that these informal, qualitative methods should be supported by performance metrics over time.

To support more structured methods of gathering feedback we look at some typical predictions made in cybersecurity and investigate whether they can be consistently verified using a method from decision theory, the brier score. Although originally developed for meteorology the brier score is used to judge the quality of expert prognosis, often in political or economical contexts [8]. In this work we examine opportunities and roadblocks when applying the brier score to prognoses made in cybersecurity.

First, we identify the requirements that a prognosis needs to satisfy so the brier score can be appliled to evaluate it. Then we analyze whether three example methods (CVSS, EPSS and IEC 62443 risk assessment) from cybersecurity can satisfy those requirements and what changes to the method would be necessary to apply the brier score.

2 Brier Score

A clear and undeniable result is important for feedback, otherwise mistakes can be ”argued away” or justified in hindsight [8]. The Brier Score [5], originally developed to measure quality of weather forecasts is also used in decision theory to evaluate prognoses from different domains [8].

For a series of n events of which each occurrence can result in one of r mutually exclusive, exhaustive classes. For event $i \in [1, ..., n]$ the forecast probability f_{ij} denotes the probability that event i falls in class $j \in [1, ..., r]$. $E_{ij} \in 0, 1$ denotes the actual result of the event (did it fall in class i or not). Then the brier score is defined as follows [5]:

$$B = \frac{1}{n} \sum_{j=1}^{r} \sum_{i=1}^{n} (f_{ij} - E_{ij})^2$$

The best possible brier score of 0 is achieved with only correct predictions (in which class the event results) with 100% (or 0% for the other classes) probability given in the forecast probabilities.

Brier shows that this score does not incentivise over- or underestimating confidence [5]. This is because if the analyst has no actual skill differentiating the different events, it is optimal to give the relative frequencies of results as the probabilities in the brier score. To use briers example, if it rains on 3 out of 10 days on average and there is no skill to predict individual days, the best strategy for a maximum brier score is to predict rain with a 30% probability on each day. However the actual frequencies are not known in advance for a given series of events in the future. So the optimal strategy for best brier scores is to give the best estimates for each individual event.

The brier score offers the needed undeniable result of predictions. We find that the following requirements need to be fulfilled so that the brier score can be used:

Requirement 1: The result of the prognosis must fall into one of several mutually exclusive, exhaustive classes that are known before.

Requirement 2: The prognosis must be expressed as probability for each class.

Requirement 3: The prognosis must be made for a clearly defined time window, after which the true result can be determined.

Requirement 4: The class in which the true result falls must be clearly identifiable after the time window has closed.

These requirements ensure that all necessary components to calculate the formula above are given. Requirement 1 ensures r is known and that the probabilities given for each class sum up to 100%.

Requirement 2 ensures the f_{ij} are known and in the correct format. This also rules out imprecise qualitative probability values such as ”likely”, ”impropable”, etc..

Requirement 3 is necessary so that the true results E_{ij} ensures that evaluation is fair and provides a clear result, by determining from the start for which time window a prognosis was made. For example a prognosis that predicts a cyberattack but does not clarify a time window can not get a clear feedback on the quality of the prognosis. If the prognosis is evaluated too soon, before the cyberattack had the chance to occur the prognosis is judged too harsh. If the prognosis is evaluated after a year, although the analyst thought in terms of months when making the prognosis, the resulting brier score can be too good. Both cases muddy the clear evaluation result that the brier score can provide.

Requirement 4 ensures that the values for the E_{ij} can be clearly determined after the time window has passed. This requirement forces analysts to make precise prognoses. It prevents fuzzy statements that can never be proven wrong, which is often tempting, even for experts [8].

3 Overview

In the following work we will look at three different methods from cybersecurity: the Common Vulnerability Scoring System (CVSS), the Exploit Probability Scoring System (EPSS) and IEC 62443 risk assessment. For each we will analyze whether a typical prognosis made with that method can be evaluated with the brier score.

For each method we attempt to answer the following **research questions**:

RQ 1: Is the brier score applicable to this prognosis as is?

RQ 2: If not, which changes would allow an application of the brier score?

RQ 3: Are these changes applicable in practice?

To approach the research questions we will use the requirements determined above. Does the method satisfy the requirements? If not, what changes to the method would satisfy the requirements? And are those changes realistic in the use case of each method?

4 Common Vulnerability Scoring System CVSS

The Common Vulnerability Scoring System is a standardized [3] metric to measure the severity of vulnerabilities in soft-, hard- and firmware. The calculation of the metric is separated into three different sets of metrics.

The Base Metrics describe the vulnerability and the thing that is vulnerable itself. This part of the score is determined by the analyst that finds the vulnerability. Tools such as the CVSS Calculator provided by NIST [7] show how the score is calculated and assist with the calculation.

The Threat Metrics consist of only one metric, the exploit maturity. It measures whether or not an exploit using the vulnerability has already been seen.

The Environmental Metrics allow to adjust vulnerability score to a given system. Thus these metrics are not provided by the CVSS supplier but by the consumer. To do this, the environmental metrics simply overwrite the base metrics to better represent the risks posed by the vulnerability to this specific system.

To determine values of metrics the analysts do not quantify the vulnerability directly, but each metric has between 2 and 4 categories in which the values can lie. For example the base metric ”Attack Vector” can fall in one of the 4 categories network, adjacent, local or physical, that describe which access an adversary needs to exploit the vulnerability. Many metrics also use generic categories such as ”low”, ”medium” and ”high” as values.

These categories all serve as input to the CVSS formula which outputs the overall Score. However the analyst that determines the score assigns only categories, not numerical values. Translation of the assigned categories in each metric into the score is done automatically.

To continue our analysis we have to determine what exact statements are made when using the CVSS score. On a surface level this statement is obviously ”The

CVSS score of vulnerability x is y". However this is not a helpful statement on its own, so we should consider what CVSS is intended for.

The intended use of the CVSS score is prioritizing vulnerabilities and supporting the decisions which vulnerabilities to handle first with limited resources. Considering this, the statements made when using CVSS are of the form "mitigation of vulnerability X should be prioritized over the mitigation of vulnerability Y" where X has the higher CVSS score.

This now leads to the question whether we can apply the brier score. We will look at the requirements separately.

Requirement 1 (result classes known beforehand): When comparing 2 vulnerabilities for prioritization there are three possible results. Vulnerability X is more severe and should be prioritized, vulnerability Y is more severe and should be prioritized, or both vulnerabilities are equally severe and it does not matter which is resolved first. This is three mutually exclusive, exhaustive classes, thus requirement 1 is satisfied.

Requirement 2 (probabilities given for each class): The CVSS score can take values between 0 (low severity) and 10 (high severity). The statement we examine compares 2 of those values, so we look at the difference between 2 CVSS scores. Naturally this is not a probability. This could be transformed to a value between 0 and 1 and thus taken as a probability that one vulnerability is more severe than the other. However, given the arbitrary way the CVSS score is calculated, an interpretation of that as either frequentistic or degree of belief is not justified. Such an interpretation is also not intended by the CVSS standard as well. So Requirement 2 is not satisfied.

Requirement 3 (Clearly defined time window): The CVSS score does not incorporate a time window by the CVSS provider. Information on when the vulnerability was found and when the CVSS score was determined is available though. So, adding a time window on the consumer side is possible. Requirement 3 is therefore also not satisfied. In this case this is less fundamental though, and could be resolved on consumer side.

Requirement 4 (Clearly identifiable results): Above we identified three mutually exclusive classes of possible results. To decide whether requirement 4 is satisfied, we need to determine whether it can be clearly determined in which

of these the real event falls. This depends on whether or not a countermeasure was introduced. First lets consider the simpler case where no additional countermeasures were introduced. In this case it is possible to determine the result of the statement by looking at the impact caused by either vulnerability. If an incident occurs and causes impact it is clearly detectable and can usually be traced back to the causing vulnerabilities. For both vulnerabilities in the statement the impact caused can be tracked over a given time period. This is a clear criteria to determine which of the three result classes were correct. However if it was acted upon the statement by adding additional countermeasures the waters are muddied significantly, as we can no longer rely on clearly visible impact to determine whether or not an attempt to use the vulnerability was made or would have been successful without the additional countermeasures. This poses an uncomfortable dilemma where it is possible to either know whether your prognosis is correct, or act on it in time, but not both. This problem also does not seem to be specific to the CVSS score, but likely applies to most predictions that lead to decisions on countermeasures. Using impact as the deciding data whether a new countermeasure was successful, also means that this can only really be evaluated in a production system, but not on a testbed, where such data gathering could be conducted much more safely. Even then, incidents with significant impact are rare, so statistical analysis is difficult.

With only one of the four requirements satisfied, the brier score is not applicable to prognoses made with CVSS.

5 Exploit Probability Scoring System EPSS

The Exploit Prediction Scoring System is another methodology to prioritize vulnerabilities [4]. It is created by FIRST, the same organisation that also created CVSS. The goal of prioritizing vulnerabilities is the same, but the approach of EPSS is data driven. Where CVSS relies on analyst opinions to calculate the score, EPSS compiles different data sources to calculate a likelihood of exploit in the next 30 days for a given vulnerability.

To calculate this, according to [4] EPSS combines several different data sources, such as CPE, CVE, CVSS, exploit code for example from MetaSploit and more.

The different sources are used to train a machine learning model that then calulates the EPSS scores.

This probability is given for attempted exploits and does not take into account whether it is successful or not or if the vulnerability is even present.

The prognosis made by a given EPSS score is clear, the EPSS score of a vulnerability gives the probability that an exploit of the vulnerability will be attempted in the next 30 days.

We examine again whether the requirements for using the brier score are satisfied by the score. In contrast to CVSS, for EPSS the requirements are clearly satisfied.

Requirement 1 (result classes known beforehand): This requirement is satisfied, an exploit is either attempted or not.

Requirement 2 (probabilities given for each class): This is also satisfied, as the EPSS score is exactly the probability for the result classes.

Requirement 3 (Clearly defined time window): This is also satisfied inherently by the 30 day time window in the EPSS score definiton.

Requirement 4 (Clearly identifiable results):To some degree this is clearly satisfied. If an exploit is (or is not) recorded in the log data during the time window the result can be clearly identified. This is also done for evaluation of EPSS [6]. The recorded data may be incomplete however, which leaves some room for error.

With a clear statement of probability and inherent time window, EPSS can be evaluated much easier than CVSS without any additions. With exploit attempts and not impact as basis of evaluation. The dilemma of measure vs mitigation that we found for CVSS, does not exist with EPSS. With the metric being exploit attempts instead of impact existing mitigations could even make detection easier if they contain logging or detection capabilities.

In conclusion, while identifying results afterwards could be difficult, depending on available information, the brier score can be applied to EPSS prognoses as is.

For example a company can get the EPSS score for a vulnerability of a device in their network. The EPSS score is the confidence that that the vulnerability will be exploited in the next 30 days (f in the brier score formula). Then the

company can monitor the network for an attempt to exploit that vulnerability for the next 30 days to determine E and thus use brier scores to evaluate whether EPSS reliably predicts probability of exploitation.

6 IEC 62443 Risk Assessment

The goal of risk assessment is to understand threats and vulnerabilities to the system, possible impact of attacks and countermeasures against them. There are several standards that describe risk assessment processes, here we will look at IEC 62443-3-2 [1]. The risk assessment process outlined there consists of several steps, with output of earlier steps often used as input for later steps.

First, lists of threats and vulnerabilities are collected, which then are used, together with assessed impact and likelihoods to assess unmitigated risk. If residual risk (after taking existing countermeasures into account) is deemed too high, additional countermeasures need to be identified. The output of this step is a updated list of countermeasures.

As selecting countermeasures is the vital decision that needs to be done at the end of the risk assessment process, we will examine this decision, the exact statements made and how they can be evaluated with the brier score in more detail.

In ZCR 5.12 IEC 62443 requires to identify additional cybersecurity countermeasures for risks that exceed the tolerable risk threshold. To support this decision IEC 62443-3-3 offers a comprehensive list of countermeasures. When selecting countermeasures, economical concerns should also be considered. There numerous countermeasures are described and assigned to security levels.

The targeted security level of a system could be used to identify countermeasures to add, but this would just result in a list of several countermeasures that have to be added with no guidance on how to prioritize. This would also not consider the identified risk or economical concerns.

So, when adding an additional countermeasures to the system, two prognoses are made.

First, the countermeasure that is added reduces one of the risks, that are deemed too high to a tolerable level. Second, the countermeasure is the most efficient option of the countermeasures that reduce that risk.

Requirement 1 (result classes known beforehand): Both prognoses are clear binary statements, so this requirement is staisfied. However, it is not clear how to operationialize this. How risk, risk reduction or efficiency of countermeasures are measured is not specified by IEC 62443. This is left open for the organisation that implements the risk assessment process. While requirement 1 is technically satisfied, the standard is unspecific, which causes problems with the following requirements.

Requirement 2 (probabilities given for each class): As already mentioned above, IEC 62443 does not specify methods to quantify security, so this is left to the organisation that implements the risk assessment process. Even if there is no quantification used to measure risk, analysts can give confidence values for the prognoses made here. So while organisations need to ensure that requirement 2 is satisfied, it certainly can be satisfied without much effort. However, IEC 62443 does not encourage the use of probabilities. In its annex example artifact for the risk assessment process are given [1]. Here probabilities are expressed as categories either from 1 through 5 or in qualitative values (e.g. ”possible”, ”likely”, ”certain”). If organisations follow those examples, requirement 2 is not satisfied.

Requirement 3 (Clearly defined time window): Once again IEC 62443 is not specific enough to decide this in general. It does not require to measure risk with a time reference (e.g. annual risk, number of attacks per month, etc.). But, as before it does not prohibit the use of that either. So while the risk assessment process can be implemented in a way such that requirement 3 is satisfied, the standard does neither encourage nor prohibit that.

Requirement 4 (Clearly identifiable results): The first prognosis we look at here is that a new countermeasure that is added to the system reduces a risk to tolerable levels. While the standard does not specify how exactly risk is measured, an organisation that is at this point of the risk assessment process must have some method to measure risk. That method can be repeated with the new countermeasure implemented. The new risk values with the countermeasures

can be compared to the old ones without countermeasure. This then provides a clear answer to the question whether the new countermeasure reduces risk.

The second prognosis is that the chosen countermeasure is the most economical option of the countermeasures that address that risk. To determine that, the actual implementation cost of both countermeasures would need to be determined. For the countermeasure that was implemented this is no problem. In hindsight the complete cost including unforeseen increases in cost are known. However, for the countermeasures that were not implemented only planned implementation costs can be determined. If the difference in cost is significant this can still provide a clear answer. if the cost of both countermeasures is too close the answer remains unclear.

In conclusion, whether the brier score is applicable here can not be determined by looking at the standard, it depends on the implementation of the individual risk assessment process. Evaluating prognoses made during it with the brier score is possible, but the risk assessment process needs to be implemented with that goal in mind. The standard itself does not necessarily guide its users in this direction.

While we have only looked at two statements that are made during the risk assessment process. We believe that the problems with the requirements are also present with most other statements made during the risk assessment proess as defined by IEC 62443. This is because the problems arise not from the exact statements made, but from the level of abstraction used in the standard. The overal policy of 62443 is to define *what* to do, but not *how*. This is necessary to achieve the general applicability such a standard needs. On that abstraction level the standard could (and should, in our opinion) advise to implement the process with evaluability in mind.

7 Discussion

We have looked at several different methods from cybersecurity risk assessment that are used to make predictions about the security of a system. Our goal was to find out whether these prognoses can be evaluated using the brier score.

For the more abstract IEC 62443 risk assessment process we identified fundamental problems when applying the brier score. They are caused by the ”we tell you what to do but not how” approach of IEC 62443. We saw that whether the brier score is applicable, depends on how statements are operationalized and how results are measured. This goes into technical depth that standards usually do not provide (for good reasons).

The other two methods we investigated, CVSS and EPSS, are on a lower abstraction level and they are clearly operationalized. Here with CVSS, we saw that a clear operationalization does not automatically allow application of the brier score. How exactly that is done still matters, with EPSS presenting a positive example.

With this in mind we now answer our research questions.

RQ1: Is the brier score applicable to this statement?

For statements from IEC 62443 risk assessment we found that this can not be determined without defining an operationalization for the statement. For CVSS requirements 2 and 3 are not satisfied, so the brier score can not be applied here. EPSS satisfies all four requirements and thus the brier score can be applied.

RQ2 and 3: Which changes would allow an application of the brier score and are they practical?

For EPSS we do not need to answer this, since the brier score is already applicable.

For the IEC 62443 risk assessment we identified the lack of operationalization as the cause that the brier score can not immediately be used to evaluate predictions. To fix this, the standard would have to abandon the ”what but not how” approach and define exact methodology and metrics to use. While this would allow for evaluation using the brier score, it would also significantly restrict implementations of the standard. The time and political pressure required for such a fundamental change in an already well established standard makes this highly impractical.

For CVSS as is we found that the brier score is not applicable for several different reasons: Statements are not given with a probability, the lack of a time window

for statements, and problems with deciding on the true outcome if additional countermeasures were placed.

The lack of probability and time window could be fixed on CVSS consumer side. However, as described above, we know of no sound interpretation of the CVSS score in terms of probability. A time window for scores could be added on for the purpose of evaluating CVSS scores, but the analyst that provided the score did not have a time window in mind when creating the score. While these fixes would allow the calculation of brier scores, the additions lack justification. It is questionable whether the resulting brier scores would provide a clearer picture on the quality of cybersecurity risk assessment statements made with CVSS. We do not consider forcing the brier score onto CVSS with these changes useful.

8 Conclusion and Outlook

Of the three methods we looked at (CVSS, EPSS and IEC 62443 risk assessment) we found only EPSS to be readily evaluable with brier scores. While we were able to identify the reasons why the other two methods were not evaluable with the brier score, we also found that fixing these problems is not worth the effort.

However, feedback and evaluation remains a relevant topic for cybersecurity [2]. Here we saw that choice of methods can decide whether the statements made in risk assessment can be evaluated. In future works analysis of more different methods and how they can be evaluated with clear undeniable results, is needed for a clearer picture how cybersecurity can be better evaluated.

The analysis here suggests fundamental problems with the evaluability of more abstract methods, such as IEC 62443. While we understand that such a standard can not go into sufficient detail to provide this without further data, the standard could provide guidance on how to implement its process in an evaulable way.

Another question for future work could be how abstract methods that are different from standards fare here. The MITRE ATT&CK framework could be of interest here because it covers a wider range of abstraction level with its tactics, techniques and procedures.

Overall we believe that further research into the evaluation of cybersecurity decisions is required.

References

[1] International Society of Automation. “ISA/IEC 62443 Series of Standards”. In: (2018).

[2] Rebekah Brown and Andreas Sfakianakis. “SANS 2025 CTI Survey Navigating Uncertainty in Today’s Threat Landscape”. In: *SANS Institute* (2025), pp. 1–18.

[3] FIRST. *Common Vulnerability Scoring System version 4.0 Specification Document*. Frankfurt am Main : 2016. URL: https://www.first.org/cvss/specification-document.

[4] FIRST. *EPSS*. https://www.first.org/epss/model. [Online; accessed 03-Mar-2026]. 2026.

[5] W Brier Glenn et al. “Verification of forecasts expressed in terms of probability”. In: *Monthly weather review* 78.1 (1950), pp. 1–3.

[6] Jay Jacobs et al. *Enhancing Vulnerability Prioritization: Data-Driven Exploit Predictions with Community-Driven Insights*. 2023. arXiv: 2302.14172 [cs.CR]. URL: https://arxiv.org/abs/2302.14172.

[7] NIST. *CVSS Calculator*. https://nvd.nist.gov/vuln-metrics/cvss/v4-calculator/. [Online; accessed 06-Feb-2026]. 2026.

[8] Philip E. Tetlock, Dan Gardner, and S. Fischer Verlag. *Superforecasting :* [1. Auflage]. Mit Register. Frankfurt am Main : S. Fischer, 2016. URL: https://www.gbv.de/dms/zbw/858683776.pdf.

Causal Sensor and Actuator Placement for Identifiable Interventional Effect Estimation

Shahenda Youssef

Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
shahenda.youssef@kit.edu

Abstract

Sensor and actuator placement is often guided by objectives such as improving prediction accuracy, monitoring system behavior, or estimating model parameters. Although these goals support effective modeling and forecasting, they do not guarantee causal validity for answering interventional or counterfactual questions, particularly in the presence of confounding, partial observability, or feedback-driven data. This work formulates placement as a causal design problem driven by identifiability and interventional information efficiency. Sensor selection is formulated as choosing measurements that enable identifiable adjustment or proxy sets for the target causal queries, while actuator selection determines intervention channels that maximize expected information gain under feasible excitation constraints. Over-instrumentation is characterized as redundancy with zero marginal causal value-of-information, enabling principled pruning via submodular optimization. The resulting joint greedy co-design strategy yields minimal sensor and actuator configurations that support identifiable and statistically efficient causal inference, together with tractable information-based surrogates for practical implementation.

1 Introduction

Optimal sensor and actuator placement (OSAP) aims to select subsets of sensors and actuators from candidate locations to optimize objectives such as estimation accuracy, fault detectability, closed-loop performance, robustness, or economic reward. Classical approaches are typically grounded in information geometry (e.g., Fisher information), system-theoretic metrics based on controllability and observability Gramians, and submodular set-function optimization [16, 17, 9, 39]. While effective for prediction and control, these formulations are primarily association-driven.

However, modern manufacturing and process systems exhibit confounding, feedback, and latent disturbances, where decisions depend on interventions rather than passive observation. As a result, purely observational criteria may produce sensor configurations that are predictive yet insufficient for identifying cause–effect relationships or supporting counterfactual reasoning. This limitation motivates a causal perspective on instrumentation design, in which placement decisions are guided by the ability to estimate the effects of interventions. Consider estimating the causal effect of actuator input U_t on outcome Y_t. If a latent confounder Z_t simultaneously influences both variables,

$$Z_t \to U_t, \quad Z_t \to Y_t, \quad U_t \to Y_t,$$

then the observational expectation $\mathbb{E}[Y_t \mid U_t = u]$ is generally biased and does not equal the interventional expectation $\mathbb{E}[Y_t \mid \mathrm{do}(U_t = u)]$. Although the observational expectation implicitly marginalizes over the latent confounder Z_t, it does so with the conditional distribution $P(Z_t \mid U_t = u)$, which depends on the actuator because Z_t influences U_t. In contrast, the interventional expectation removes this dependence and averages over the marginal distribution $P(Z_t)$. Placing sensors that maximize predictive R^2 for Y_t may not help identify the causal effect of U_t unless confounders are measured or suitable instruments exist [29, 32].

Effective placement requires two complementary capabilities: actuators that generate valid exogenous variation and sensors that measure variables sufficient for identifiability. Instrumentation design, therefore, becomes a coupled problem

in which actuator placement determines the feasible interventions, while sensor placement determines whether their effects can be uniquely and efficiently estimated. We formulate placement as an optimization problem that selects sensor and actuator configurations that ensure identifiability of target interventional effects while maximizing the information gained about these effects under hardware and cost constraints. Measurements whose marginal contribution to the causal estimand is negligible are considered over-instrumentation and are excluded.

Accordingly, this work adopts a causal placement perspective, focusing on selecting minimal sensor and actuator sets that ensure identifiable and statistically efficient inference rather than prediction or control performance alone. The remainder of this report is structured as follows: Section 2 reviews the state-of-the-art of the classical OSAP and causal foundations underlying identifiability and interventions. Section 3 introduces the system model and problem formulation. Sections 4 and 5 tackle the proposed methods for sensor placement and actuator placement, respectively. Section 6 presents the joint sensor–actuator co-design and the over-instrumentation control. Finally, Section 7 concludes the paper and discusses future directions.

2 Background

This section reviews work most directly related to sensor and actuator placement as a combinatorial design problem, information-theoretic and submodular formulations that enable greedy selection, and causal inference and interventional design methods that motivate causality-driven placement.

2.1 Classical Sensor and Actuator Placement

Classical placement methods typically aim to improve state reconstruction or control performance through metrics derived from observability and controllability [2, 20, 23, 16]. In large-scale settings, placement is combinatorial and frequently optimized via scalarizations of Gramians (e.g., trace, log-determinant,

minimum eigenvalue) and related measures. Summers *et al.* [39] show that several Gramian-based placement objectives admit favorable set-function structure (e.g., modularity/submodularity/supermodularity properties depending on the metric), enabling scalable optimization in networked dynamical systems [21, 4, 38].

Beyond Gramian metrics, balanced truncation and balanced-model-reduction-based methods provide scalable sensor/actuator selection by projecting the design onto dominant balanced modes and using computationally efficient pivoting procedures [24]. A complementary line addresses structural placement (ensuring structural controllability/observability with minimal cost), which is relevant when parameters are uncertain but sparsity structure is known; Pequito *et al.* give polynomial-time solutions for minimum-cost input/output design under structural controllability/observability requirements [30].

Information-theoretic placement objectives (mutual information, entropy reduction, log-det criteria) have been influential because they connect placement to statistical efficiency and often yield approximation guarantees. In the Gaussian-process setting, Krause *et al.* prove submodularity of mutual information for sensor placement and provide a greedy algorithm with a $(1 - 1/e)$ approximation guarantee, while also showing NP-completeness of exact optimization [17]. These results build on foundational approximation theory for monotone submodular maximization under cardinality constraints [28].

Bayesian experimental design (BED) chooses designs to maximize expected utility, often KL-divergence from prior to posterior (information gain) [22, 33]. Sensor placement for nonlinear and uncertain systems frequently uses expected KL utility [6], expected information gain:

$$U(d) = \mathbb{E}_{y \sim p(y|d)} \left[\mathrm{KL} \left(p(\boldsymbol{\theta} \mid y, d) \,\|\, p(\boldsymbol{\theta}) \right) \right] = \mathrm{I}(\boldsymbol{\theta}; y \mid d). \tag{2.1}$$

Here, $p(\boldsymbol{\theta})$ denotes the prior distribution over parameters, $p(\boldsymbol{\theta} \mid y, d)$ is the posterior distribution after observing data y under design d, and $p(y \mid d)$ is the predictive distribution of observations induced by the design. The resulting utility $U(d)$ therefore quantifies the expected reduction in uncertainty about the parameters $\boldsymbol{\theta}$ obtained by performing experiment d.

While these approaches are essential for classical estimation/control, they optimize state/control objectives rather than causal ones: they do not directly guar-

antee identifiability of interventional effects, robustness to confounding, or correctness of counterfactual queries.

2.2 Causal Inference and Identifiability

Causal inference formalizes intervention using structural causal models (SCMs) and the do-operator [31, 32, 15]. In SCM, the variables $V_1, \ldots, V_p$ represent the system variables of interest, satisfy

$$V_i := f_i(\mathrm{Pa}_i, U_i), \qquad i = 1, \ldots, p, \tag{2.2}$$

where $f_i(\cdot)$ is a deterministic structural mechanism describing how variable V_i is generated from its direct causes. The set Pa_i denotes the parents (direct causal predecessors) of V_i in a directed acyclic graph (DAG) G, and U_i represents an exogenous noise variable capturing unobserved influences on V_i. The collection of exogenous variables $U_1, \ldots, U_p$ is typically assumed to be mutually independent [29, 32].

Let $Y \in \{V_1, \ldots, V_p\}$ denote the outcome variable of interest. Given SCM (2.2), an intervention $\mathrm{do}(V_j = v)$ replaces the structural equation for V_j with $V_j := v$. This operation modifies the data-generating process and produces the interventional distribution $P(Y \mid \mathrm{do}(V_j = v))$. A primary goal is to estimate causal estimands, such as the average treatment effect (ATE) and the conditional average treatment effect (CATE) [29]. Let Y_{t_1} and Y_{t_0} denote the potential outcomes under treatment and control, respectively. The ATE measures the expected difference in outcomes between these two interventions, while the CATE quantifies this effect conditional on covariates $X = x$,

$$\mathrm{ATE} = \mathbb{E}[Y_{t_1} - Y_{t_0}], \tag{2.3}$$

$$\mathrm{CATE}(x) = \mathbb{E}[Y_{t_1} - Y_{t_0} \mid X = x]. \tag{2.4}$$

The central concept relevant to placement is identifiability: whether a target interventional distribution can be computed from available observational data under stated assumptions. In static settings, identifiability is often established via adjustment criteria (e.g., backdoor/frontdoor) and other graphical conditions [29]. For instrumentation design, the key implication is that sensors determine

the observed set used for adjustment/proxy strategies; without measuring the right variables, causal effects can remain unidentifiable regardless of prediction accuracy.

When confounders are unobserved, identifiability may still be achievable using proxy variables under rank/independence conditions. Miao *et al.* show that with at least two suitable proxies of an unmeasured confounder, causal effects can be nonparametrically identified even when the measurement error mechanism is not identified [26]. This is directly relevant to sensor placement: adding sensors can be framed as adding proxies to enable identification rather than merely improving prediction.

Time-series causal inference introduces challenges that are central in engineering systems: lagged dependencies, contemporaneous causal relations, hidden confounding, and nonstationarity [27, 37]. Runge *et al.* provides a technical review emphasizing these issues and clarifying assumptions needed for time-series causal discovery and effect estimation [35]. When structure is uncertain, causal discovery methods can guide candidate sets [8], with applications to industrial chemical processes [5] and alarm networks [1]. For scalable causal discovery in high-dimensional nonlinear time series, Runge *et al.* propose methods (e.g., constraint-based approaches leveraging conditional independence tests) and demonstrate performance on large-scale settings [34]. These works motivate placement constraints: sensors must have sufficient temporal resolution, synchronization, and coverage of lagged parents/confounders required by sequential identification conditions. Moreover, in many industrial settings, actions are chosen by a feedback policy (closed loop), making treatment assignment endogenous; causal placement must therefore jointly ensure that actuation can generate informative excitation and sensing can measure the relevant history for identification.

2.3 Interventional Design and Active Causal Learning

Causal models have been integrated into sensor allocation for diagnosis and abnormality detection, e.g., combining causal structure and set cover [20, 23]. However, many industrial deployments still optimize association-based criteria,

risking wrong decisions under interventions. Interventions are crucial for disambiguating causal structure and learning mechanisms. When causal structure or effect parameters are unknown, actuators enable interventions that resolve ambiguities. Bayesian active design selects interventions maximizing expected information gain about G or the causal effect estimation τ [40, 7]. Let Θ denote the unknown quantities of interest, which may represent the causal graph G, its structural parameters, or the causal effect τ. Let D_0 denote existing observational data, and let $\mathcal{E}$ denote a candidate experimental design specifying a set of feasible interventions. The data generated by executing experiment $\mathcal{E}$ is denoted by $D(\mathcal{E})$. The optimal experimental design is chosen by maximizing the expected reduction in uncertainty about Θ,

$$\max_{\mathcal{E}} \ \mathbb{E}\left[H(\Theta \mid D_0) - H(\Theta \mid D_0, D(\mathcal{E}))\right], \tag{2.5}$$

where $H(\cdot)$ denotes the entropy of the posterior distribution over the unknown quantities Θ. When Θ represents the causal graph G, the entropy is computed with respect to the posterior distribution over candidate graph structures,

$$H(G \mid D) = -\sum_{g \in \mathcal{G}} P(G = g \mid D) \log P(G = g \mid D),$$

where $\mathcal{G}$ denotes the space of admissible DAGs. The expectation is taken over the distribution of possible datasets $D(\mathcal{E})$ generated by performing experiment $\mathcal{E}$. This objective corresponds to maximizing the expected information gained about Θ from the experiment. Interventions therefore help resolve ambiguities in the causal graph and improve the estimation of causal parameters or effects.

A broad literature studies how to select interventions to reduce Markov equivalence classes or maximize information. In [11], active learning of causal networks with interventional experiments using minimax and maximum-entropy criteria. Hauser and Bühlmann develop strategies for selecting interventions that optimize edge orientation and provide results on minimum intervention targets needed for full identifiability [10]. Shanmugam *et al.* analyze learning causal graphs when interventions are bounded in size, providing bounds and algorithms under constrained intervention regimes [36]. In online settings, causal bandits formalize learning good interventions using causal structure to reduce sample

complexity [18]. These results are directly relevant to actuator placement: actuators determine which intervention targets are feasible, and constrained/safe interventions are the realistic regime for physical systems.

Counterfactual value-of-information (VoI) is used to prune redundant devices, VoI quantifies the expected improvement in decision quality if we acquire additional data [22, 12, 3]. Classical placement optimizes observability/controllability or state information; causal inference defines identifiability but typically assumes the relevant variables are already observable; interventional design optimizes intervention targets but generally treats sensing as given. In contrast, this work explicitly connects causal identifiability requirements to physical sensor placement via set-cover formulations, and connects interventional learnability to actuator placement via expected information gain or effect-uncertainty reduction. Finally, it formalizes over-instrumentation using marginal causal information, enabling principled stopping/pruning through greedy submodular selection.

3 Problem Formulation

We formalize the OSAP as a constrained causal design problem. The goal is to determine a minimal set of physical measurement and intervention locations that guarantee both identifiability of target causal effects and statistical efficiency of their estimation.

Consider discrete-time dynamics

$$x_{t+1} = f(x_t, u_t, z_t, \varepsilon_t), \tag{3.1}$$

$$y_t^{(i)} = h_i(x_t, \eta_t^{(i)}). \tag{3.2}$$

Here x_t contains the true physical variables of the process, u_t actuator inputs (interventions), z_t latent confounders that influence the dynamics but are not directly controlled, y_t sensor outputs, and ε_t, η_t process and measurement noise, respectively. The mappings f and h describe the system dynamics and measurement model.

Let $\mathcal{S}$ and $\mathcal{A}$ denote the sets of candidate sensor and actuator locations, respectively. We select subsets $S \subseteq \mathcal{S}$ and $A \subseteq \mathcal{A}$ corresponding to the locations

where sensors and actuators are installed. Each location $i \in \mathcal{S}$ provides a measurement of some function of the underlying system variables. Let c_i denote the installation cost of placing a sensor at location i, and d_j the cost of placing an actuator at location j. The total available budgets for sensors and actuators are denoted by B_s and B_a, respectively. A generic OSAP objective

$$\max_{\mathcal{S},\mathcal{A}} \; J(\mathcal{S},\mathcal{A}) \quad \text{s.t.} \sum_{i\in\mathcal{S}} c_i \le B_s, \qquad \sum_{j\in\mathcal{A}} d_j \le B_a. \tag{3.3}$$

Classical J includes Gramian traces/determinants, Fisher information, mutual information, and control metrics. We extend J to depend on a causal query (intervention/counterfactual) and a decision loss.

Causal Queries: Let $\mathcal{Q} = \{q_1, \ldots, q_M\}$ denote the set of target causal estimands. Each query q_m corresponds to a causal effect functional of the form for the static effect,

$$\tau(u,u') := \mathbb{E}[Y \mid \mathrm{do}(U = u)] - \mathbb{E}[Y \mid \mathrm{do}(U = u')], \tag{3.4}$$

and for the lagged time-series effect

$$\tau_{t,\ell}(u,u') := \mathbb{E}[Y_t \mid \mathrm{do}(U_{t-\ell} = u)] - \mathbb{E}[Y_t \mid \mathrm{do}(U_{t-\ell} = u')]\,, \tag{3.5}$$

where ℓ lag windows permit delayed effects. The placement must support the correct estimation of all queries in $\mathcal{Q}$.

Identifiability Requirements: A necessary requirement for causal validity is identifiability. Let $\mathcal{R}(\mathcal{Q})$ denote the set of variables required to identify the queries, e.g., adjustment sets, proxy variables, or history conditioning variables. Each sensor i covers a subset of variables $\mathrm{vars}(i)$, inducing the observable set

$$\mathcal{O}(S) := \bigcup_{i\in S} \mathrm{vars}(i). \tag{3.6}$$

A placement is admissible only if all required variables are observed,

$$\mathcal{R}(\mathcal{Q}) \subseteq \mathcal{O}(S). \tag{3.7}$$

Each actuator location $j \in \mathcal{A}$ enables interventions on some intervention target(s). Actuators determine the feasible intervention targets

$$\mathcal{I}(A) := \bigcup_{j\in A} \mathrm{targets}(j). \tag{3.8}$$

Objective Functions: A placement (S, A) is admissible only if every query $q_m \in \mathcal{Q}$ is identifiable given the observable variables $\mathcal{O}(S)$ and feasible intervention targets $\mathcal{I}(A)$. Among all admissible placements, we seek those that minimize the uncertainty of the causal effect estimates

$$\min_{S,A} \quad \mathrm{Var}(\tau \mid S, A), \tag{3.9}$$

or equivalently, maximising mutual information between the estimand and the collected data,

$$\max_{S,A} \quad \mathrm{I}(\tau; D_{S,A}), \tag{3.10}$$

where $D_{S,A}$ denotes the dataset generated using sensors S under interventions enabled by A.

Combining identifiability, efficiency, and cost, the causal placement problem is formulated as

$$\begin{aligned} \max_{S \subseteq \mathcal{S},\ A \subseteq \mathcal{A}} \quad & \mathrm{I}(\tau; D_{S,A}) - \lambda C(S, A) \\ \text{s.t.} \quad & \mathcal{R}(\mathcal{Q}) \subseteq \mathcal{O}(S), \end{aligned} \tag{3.11}$$

Here, $\lambda > 0$ is a regularization parameter controlling the trade-off between causal information gain and instrumentation cost, and $C(S, A)$ is the total instrumentation cost,

$$C(S, A) = \sum_{i \in S} c_i + \sum_{j \in A} d_j. \tag{3.12}$$

This combinatorial optimization (3.11) seeks the smallest set of sensors and actuators that guarantees identifiability while maximizing interventional information.

4 Sensor Placement for Causal Identifiability

This section describes how to select a subset of sensors $S \subseteq \mathcal{S}$ that guarantees identifiability of the target causal queries and minimizes the uncertainty of the resulting effect estimates. Throughout this section, actuator placement A is assumed fixed.

4.1 Identifiability Requirements

A necessary condition for valid causal estimation τ is that all variables required for adjustment, proxy construction, or history conditioning are observed. The required variable set $\mathcal{R}(\mathcal{Q})$ 3.7 that makes the queries identifiable under chosen causal assumptions may contain:

- Confounders needed for backdoor adjustment.
- Mediators if separating direct/indirect effects is required.
- Regime indicators if mechanism invariance is assumed across environments.
- Lagged parents in time-series graphs if $\tau_{t,\ell}$ is targeted.

Backdoor adjustment: Let $X_t \subseteq \mathcal{O}(S)$ denote a set of observed covariates/confounders measured by the sensors. These variables form a valid adjustment set if they block all backdoor paths from U_t to $Y_{t+\ell}$ and contain no descendants of U_t. In this case, the causal effect can be identified as

$$P(Y_{t+\ell} \mid \mathrm{do}(U_t = u)) = \sum_z P(Y_{t+\ell} \mid U_t = u, X_t = x)\, P(X_t = x). \quad (4.1)$$

Instrumental variables (IV): When latent confounders Z_t affect both treatment and outcome, identification may still be possible using an instrument W_t such that $W_t \to U_t \to Y_{t+\ell}$, $W_t \perp\!\!\!\perp Z_t$, and W_t affects $Y_{t+\ell}$ only through U_t. Instrumentation implication: measure the instrument and the treatment, and validate exclusion and relevance conditions:

$$\begin{aligned} \text{Relevance: } & \mathrm{Cov}(W_t, U_t) \neq 0, \\ \text{Exclusion: } & W_t \perp\!\!\!\perp Y_{t+\ell} \mid U_t, \\ \text{Independence: } & W_t \perp\!\!\!\perp Z_t. \end{aligned} \quad (4.2)$$

Frontdoor-style intuition: Let M_t denote a mediator variable, i.e., an intermediate system variable that transmits the causal effect from the actuator input U_t to the outcome $Y_{t+\ell}$. If such a mediator is observed and satisfies the appropriate graphical conditions, the causal effect can be identified even when U_t and $Y_{t+\ell}$ are confounded.

4.2 Physical Sensor Locations

Sensor placement can be viewed as a set-cover problem. From (3.6)–(3.7), sensors must be chosen so that their collective coverage includes all variables required for identification. When sensors have installation costs c_i, the placement problem can be formulated as a weighted set cover:

$$\min_{S \subseteq \mathcal{S}} \quad \sum_{i \in S} c_i \tag{4.3}$$

$$\text{s.t.} \quad \mathcal{R}(\mathcal{Q}) \subseteq \mathcal{O}(S). \tag{4.4}$$

When direct confounders Z_t are unmeasured, identifiability may be achieved with proxy measurements P_t^1, P_t^2 [26]. At the placement level, this means selecting sensors that provide multiple informative measures of latent drivers

$$P_t^1 = g_1(Z_t, \epsilon_t^1), \quad P_t^2 = g_2(Z_t, \epsilon_t^2), \quad \epsilon_t^1 \perp\!\!\!\perp \epsilon_t^2 \mid Z_t, \tag{4.5}$$

so the required set $\mathcal{R}(\mathcal{Q})$ may include proxies rather than Z itself. In practice, this maps to placing sensors on two different physical modalities affected by the same latent driver.

4.3 Causal Fisher Information

Among all placements that ensure causal identifiability, we seek sensor configurations that minimize the uncertainty of the estimated causal effects. Let $\boldsymbol{\theta} \in \mathbb{R}^p$ denote the vector of parameters of the assumed causal model, and let $\tau(\boldsymbol{\theta})$ denote the causal estimand of interest, expressed as a function of the model parameters. Let $S \subseteq \mathcal{S}$ denote the selected set of sensor locations, and suppose that n independent observations are collected from the system using these sensors. The maximum-likelihood estimator $\hat{\boldsymbol{\theta}}$ of the parameters satisfies the asymptotic normality property

$$\sqrt{n}(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}) \Rightarrow \mathcal{N}(0, \boldsymbol{F}_S^{-1}), \tag{4.6}$$

where $\boldsymbol{F}_S \in \mathbb{R}^{p \times p}$ denotes the Fisher information matrix associated with the measurements obtained from the sensor set S. The Fisher information matrix

depends on the statistical model and the variables observed by the selected sensors. Thus, $\boldsymbol{F}_S^{-1}$ represents the asymptotic covariance matrix of the parameter estimator and quantifies how informative the chosen sensors are about the parameters of the causal model and, consequently, about the causal estimand $\tau(\boldsymbol{\theta})$.

Since the causal effect $\tau(\boldsymbol{\theta})$ is a function of these parameters, its variance follows from the delta method as

$$\mathrm{Var}(\widehat{\tau} \mid S) \approx \nabla\tau(\boldsymbol{\theta})^\top \boldsymbol{F}_S^{-1} \nabla\tau(\boldsymbol{\theta}), \tag{4.7}$$

showing that the effect of uncertainty is directly governed by the information provided by the sensors. Consequently, sensor placement can be formulated as selecting the subset that maximizes the information about the causal effect, e.g.,

$$\begin{aligned} &\max_{S \subseteq \mathcal{S}} \ \log\det \boldsymbol{F}_S, \quad \min_{S} \ \mathrm{tr}(\boldsymbol{F}_S^{-1}) \\ &\text{s.t. } c_i \leq B_s, \end{aligned} \tag{4.8}$$

which is equivalent to maximizing the mutual information $\mathrm{I}(\tau; D_S)$. This objective explicitly targets the reduction of uncertainty in the causal effect itself.

5 Actuator Placement for Interventions

We consider the selection of actuators $A \subseteq \mathcal{A}$, which determine the set of feasible interventions and therefore how informative the collected data is about the causal effects of interest. Actuators actively modify the data-generating process and are essential for learning causal mechanisms.

5.1 Feasible Interventions

Interventions generate exogenous variation that breaks confounding bias and enables the identification of cause-and-effect relationships. Consequently, actuator placement directly controls the learnability of the causal parameters. Actuators should be placed to excite confounded mechanisms, generate diverse system

responses, influence many downstream variables, and improve parameter identifiability.

If U_t is policy-determined (closed loop), then observational correlations may be biased. Actuators enable designed excitations that approximate exogenous interventions

$$u_t = \kappa(y_{0:t}) + \xi_t, \tag{5.1}$$

where ξ_t is deliberately injected excitation. If u_t is set exogenously, then it acts as a valid intervention. If u_t is chosen by a controller $u_t = \kappa(y_{0:t})$, then u_t becomes endogenous, and causal interpretation must include the controller mechanism. Actuator placement chooses where such excitation can be applied safely and informatively. Recent work explicitly treats closed-loop designs with causal estimands and specialized estimators [19].

Physical systems impose safety or operational limits on actuation (3.8). Thus, interventions must satisfy

$$u_t \in \mathcal{U}(A), \qquad \text{safety}(x_{0:T}) \leq \rho. \tag{5.2}$$

These constraints restrict feasible experiments and must be considered during placement, as some actuators may be informative but unsafe to use.

5.2 Interventional Design Objective

Expected Information Gain: Let $\boldsymbol{\theta}$ parameterize the causal mechanism(s) relevant to $\mathcal{Q}$. An interventional design objective is expected information gain (EIG)

$$\text{EIG}(A) := \mathbb{E}\left[\text{KL}(p(\boldsymbol{\theta} \mid D, \text{do}(\cdot; A)) \,\|\, p(\boldsymbol{\theta} \mid D))\right]. \tag{5.3}$$

Actuator placement selects A maximizing $\text{EIG}(A)$ subject to (3.12) and safety constraints (5.2)

$$\max_{A \subseteq \mathcal{A},\ |A| \leq B_a} \text{EIG}(A). \tag{5.4}$$

Interventional Mutual Information: Actuator placement could be formulated as maximizing interventional mutual information between parameters and

measurements under an intervention policy $\pi(A)$

$$J_{\text{IMI}}(A) = \text{I}(\boldsymbol{\theta}; D \mid \text{do}(\pi(A))). \tag{5.5}$$

This objective evaluates how informative the experiments enabled by the selected actuators are. In linear-Gaussian settings, this reduces to log-determinant criteria; in nonlinear models, it may be estimated via Monte Carlo sampling [33]. Actuator placement is therefore formulated as

$$\max_{A \subseteq \mathcal{A}} J_{\text{IMI}}(A) \quad \text{s.t. } d_j \leq B_a. \tag{5.6}$$

Causal reinforcement learning/control If the goal is decision-making, one may learn a causal world model and compute policies robust to distribution shifts. A simplified formulation is

$$\pi^\star = \text{argmax}_\pi \, \mathbb{E}\Big[\sum_{t=0}^{T} \gamma^t r(x_t, u_t)\Big], \tag{5.7}$$

subject to the system dynamics

$$x_{t+1} = f(x_t, u_t, \varepsilon_t), \qquad u_t = \pi(x_t), \tag{5.8}$$

where x_t denotes the system state, u_t the action applied by the actuator, π the policy mapping states to actions, and f is a SCM describing the system dynamics. Interventions correspond to actions chosen by the policy.

Posterior uncertainty: When the sole target is a causal effect τ, one can design actuators to minimize posterior uncertainty of τ:

$$\min_A \text{ Var}\left(\tau \mid A\right) \quad \text{or} \quad \min_A \text{ tr}\left(\Sigma_\tau(A)\right), \tag{5.9}$$

where $\Sigma_\tau(A)$ is the posterior covariance of the effect estimate induced by feasible experiments at actuators A.

6 Joint Design and Over-Instrumentation Control

The preceding sections considered sensor and actuator placement independently. In practice, however, sensing and actuation are tightly coupled. Sensors deter-

mine which variables can be observed, while actuators determine which mechanisms can be excited. A sensor provides limited value if the corresponding mechanism is never perturbed, and an actuator provides little information if its downstream effects cannot be measured.

Let $D_{S,A}$ denote the dataset generated using sensors S under an intervention policy enabled by actuators A. A unified design criterion $F(S, A)$ is the information gained about the causal parameters or effects $\mathrm{I}(\tau; D_{S,A})$, or equivalently, the reduction in posterior uncertainty of the estimand $\mathrm{Var}(\tau \mid S, A)$. The joint placement problem becomes

$$\begin{aligned} \max_{S \subseteq \mathcal{S},\, A \subseteq \mathcal{A}} \quad & F(S, A) - \lambda C(S, A) \\ \text{s.t.} \quad & \mathcal{R}(\mathcal{Q}) \subseteq \mathcal{O}(S), \end{aligned} \tag{6.1}$$

Directly computing $\mathrm{I}(\tau; D_{S,A})$ can be expensive. Fisher information surrogate and Gaussian posterior approximation methods give standard surrogates that keep the design implementable.

Fisher information surrogate If τ is parameterized via θ and $\widehat{\theta}$ is asymptotically normal with Fisher information $J(S, A)$:

$$\mathrm{Cov}(\widehat{\theta}) \approx J(S, A)^{-1}, \tag{6.2}$$

and if $\tau = g(\theta)$, then by delta method:

$$\Sigma_\tau(S, A) \approx \nabla g(\theta)\, J(S, A)^{-1}\, \nabla g(\theta)^\top. \tag{6.3}$$

Thus placement may maximize $J(S, A)$ in directions that matter for τ.

Gaussian posterior approximation Assume a local Gaussian approximation for the posterior of effect τ:

$$\tau \mid D_{S,A} \approx \mathcal{N}\big(\mu_\tau(S, A),\ \Sigma_\tau(S, A)\big). \tag{6.4}$$

Then entropy is

$$H(\tau \mid D_{S,A}) = \tfrac{1}{2} \log \det\big(2\pi e\, \Sigma_\tau(S, A)\big)\,, \tag{6.5}$$

and maximizing information is equivalent to minimizing log-det covariance:

$$\max f(S, A) \iff \min_{S,A}\ \log\det\big(\Sigma_\tau(S, A)\big). \tag{6.6}$$

This formulation highlights the complementary roles of the two modalities: sensors primarily reduce estimation variance by improving observability of relevant variables, whereas actuators increase information by generating exogenous variation and breaking confounding. Effective designs therefore require both adequate coverage of adjustment variables and sufficient excitation of causal pathways. In many cases, a small number of well-placed actuators combined with a minimal set of high-quality sensors yields higher information than either dense sensing or dense actuation alone.

Over-Instrumentation control: When $F(S, A)$ is monotone and approximately submodular over $\mathcal{S} \cup \mathcal{A}$, near-optimal solutions can be obtained via greedy co-design. To account for heterogeneous instrumentation costs, at each step we add the sensor or actuator with the largest marginal information gain per unit cost,

$$e^\star = \underset{e \in (\mathcal{S} \setminus S) \cup (\mathcal{A} \setminus A)}{\arg\max} \frac{\Delta(e \mid S, A)}{\kappa(e)} \tag{6.7}$$

where $\kappa(e)$ denotes the cost of candidate element e,

$$\kappa(e) = \begin{cases} c_i, & e = i \in \mathcal{S}, \\ d_j, & e = j \in \mathcal{A}, \end{cases} \tag{6.8}$$

and the marginal gain is defined as

$$\Delta(e \mid S, A) = \begin{cases} F(S \cup \{e\}, A) - F(S, A), & e \in \mathcal{S}, \\ F(S, A \cup \{e\}) - F(S, A), & e \in \mathcal{A}. \end{cases} \tag{6.9}$$

Over-instrumentation arises when additional devices provide negligible marginal causal information relative to their cost. Thus, elements with small values of $\Delta(e \mid S, A)/\kappa(e) \approx 0$ are considered redundant and are omitted. Causality makes this explicit: extra sensors may improve predictive accuracy while contributing little to reducing uncertainty about the causal estimand. A practical stopping rule is

$$\max_e \frac{\Delta(e \mid S, A)}{\kappa(e)} \leq \epsilon, \tag{6.10}$$

which terminates placement once additional sensors or actuators no longer provide meaningful causal information per unit cost. This unified strategy balances sensing and actuation while preventing unnecessary instrumentation.

7 Discussion

We formulate causal placement as a principled design problem that transforms causal requirements into concrete sensor and actuator location decisions. Sensor placement is driven by identifiability: selecting physical measurement locations that realize required adjustment/proxy/history variables. Actuator placement is driven by interventional learnability: selecting intervention locations that maximize information about target effects or mechanisms under safety constraints. Over-instrumentation is formalized as redundancy with zero marginal causal information and controlled using greedy submodular selection and stopping rules. The resulting approach yields minimal sufficient instrumentation sets for identifiable and informative causal inference.

In practice, sensor quality directly affects identifiability and efficiency. Noisy, biased, or drifting measurements effectively behave as latent confounders and can reintroduce spurious dependencies even when the correct variables are nominally observed. Consequently, placement should favor sensors with a high signal-to-noise ratio and stable calibration [14]. To explicitly account for this effect, consider the generic measurement model

$$y_t = \alpha_t x_t + \beta_t + \epsilon_t, \tag{7.1}$$

where α_t and β_t represent time-varying gain and offset parameters.

Identifiability results, such as adjustment or proxy conditions, are stated with respect to the true variable x_t. In practice, however, estimation is performed using y_t. Conditioning on y_t is not equivalent to conditioning on x_t, and the required conditional independence relations may fail to hold. For example, a backdoor adjustment assumption of the form $Y_t \perp\!\!\!\perp U_t \mid X_t$ is generally violated when replacing x_t with its corrupted observation y_t. Consequently, a sensor that measures the correct variable but with a poor signal-to-noise ratio may fail to remove confounding, behaving effectively as if the confounder were unobserved. From a placement perspective, this implies that variable coverage alone is insufficient; measurement reliability must also be considered.

Sensor drift further introduces a time-varying source of bias that can be interpreted causally as latent confounding. A simple drift model is

$$\alpha_{t+1} = \alpha_t + \nu_t, \qquad \beta_{t+1} = \beta_t + \zeta_t, \tag{7.2}$$

where (α_t, β_t) evolve stochastically. These latent drift states influence the recorded measurements and may correlate with operating regimes or outcomes. This corresponds to an unobserved variable Z_t, creating new backdoor paths. As a result, sequential ignorability assumptions such as $Y_t \perp\!\!\!\perp U_t \mid X_t$ may be violated because the analyst conditions on mismeasured variables rather than their true values. In this sense, drift correction and calibration can be interpreted as interventions on the measurement mechanism itself that restore the invariance conditions required for valid causal estimation [13, 25].

These effects have direct implications for placement decisions. A sensor that is highly noisy or drifting contributes little useful information about the causal estimand and may even increase estimation variance or bias. Therefore, instrumentation quality must be incorporated into the placement objective. Extending the causal loss to account for measurement integrity, we consider

$$\mathcal{L}_{\text{causal}}(S, A) = \underbrace{\operatorname{tr}(\Sigma_\tau(S, A))}_{\text{effect uncertainty}} + \underbrace{\gamma \mathcal{B}_{\text{me}}(S)}_{\text{measurement-error risk}} + \eta \underbrace{\mathcal{D}_{\text{drift}}(S)}_{\text{drift risk}}, \tag{7.3}$$

where $\Sigma_\tau(S, A)$ denotes the posterior covariance of the causal effect, $\mathcal{B}_{\text{me}}$ quantifies sensitivity to measurement error, and $\mathcal{D}_{\text{drift}}$ captures the expected impact of drift. Under this formulation, sensor placement balances three competing objectives: reducing effect uncertainty, minimizing measurement-induced bias, and avoiding unstable instrumentation. The optimal design, therefore favors a small number of high-quality sensors rather than a large number of unreliable measurements. In particular, sensors whose marginal contribution to reducing Σ_τ is negligible or whose drift risk is high constitute over-instrumentation and should be excluded. Thus, minimal but reliable sensing is preferable to extensive but low-integrity instrumentation.

Beyond passive measurement quality, calibration provides an active means of controlling the measurement process itself. Let M_t denote the measured covariate used for causal adjustment. In general, the measurement process can be

written as $M_t = g(x_t; \theta_t)$. Calibration modifies this mechanism by resetting the parameters to an explicit intervention $\mathrm{do}(\theta_t = \theta^\star)$ on the measurement structural causal model. Viewing calibration as an intervention highlights two causal benefits. First, it stabilizes the measurement mechanism by restoring invariance of the conditional distribution $P(M_t \mid x_t)$, which is a key assumption underlying invariant prediction and causal transportability across operating regimes [31]. Second, it prevents time-varying measurement parameters from acting as latent confounders. Without calibration, drifting parameters introduce hidden dependencies between recorded covariates and outcomes, violating adjustment or sequential ignorability assumptions. Periodic intervention on θ_t removes this time-varying bias and preserves identifiability of causal effects.

This observation introduces a practical trade-off between hardware and maintenance. Instead of adding more sensors to compensate for uncertainty, one may reduce the effect variance or bias by calibrating existing sensors. Thus, optimal instrumentation balances sensor count with calibration effort, favoring fewer but well-calibrated measurements over extensive but drifting instrumentation. Incorporating calibration policies into the placement objective effectively treats measurement interventions as part of the causal design space.

References

[1] Rute Souza de Abreu et al. “A method for detecting causal relationships between industrial alarm variables using Transfer Entropy and K2 algorithm”. In: *Journal of Process Control* 106 (2021), pp. 142–154.

[2] Matthew Bartos and Branko Kerkez. “Observability-Based Sensor Placement Improves Contaminant Source Identification in River Networks”. In: *AGU Fall Meeting Abstracts*. Vol. 2021. 2021, H25X–1303.

[3] Sergio Cantero-Chinchilla et al. “Robust optimal sensor configuration using the value of information”. In: *Structural Control and Health Monitoring* 29.12 (2022), e3143.

[4] Fabrizio L Cortesi, Tyler H Summers, and John Lygeros. “Submodularity of energy related controllability metrics”. In: *53rd IEEE conference on decision and control*. IEEE. 2014, pp. 2883–2888.

[5] Harman Dewantoro, Alexander Smith, and Prodromos Daoutidis. “Causal Discovery for Topology Reconstruction in Industrial Chemical Processes”. In: *Industrial & Engineering Chemistry Research* 63.26 (2024), pp. 11530–11543.

[6] Tulay Ercan and Costas Papadimitriou. “Bayesian optimal sensor placement for parameter estimation under modeling and input uncertainties”. In: *Journal of Sound and Vibration* 563 (2023), p. 117844.

[7] Heyang Gao et al. “Policy-Based Bayesian Active Causal Discovery with Deep Reinforcement Learning”. In: *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2024, pp. 839–850.

[8] Chang Gong et al. “Causal discovery from temporal data: An overview and new perspectives”. In: *ACM Computing Surveys* 57.4 (2024), pp. 1–38.

[9] Carlos Guestrin, Andreas Krause, and Ajit Paul Singh. “Near-optimal sensor placements in gaussian processes”. In: *Proceedings of the 22nd international conference on Machine learning*. 2005, pp. 265–272.

[10] Alain Hauser and Peter Bühlmann. “Two optimal strategies for active learning of causal models from interventional data”. In: *International Journal of Approximate Reasoning* 55.4 (2014), pp. 926–939.

[11] Yang-Bo He and Zhi Geng. “Active learning of causal networks with intervention experiments and optimal designs”. In: *Journal of Machine Learning Research* 9.11 (2008).

[12] Ronald A Howard. “Information value theory”. In: *IEEE Transactions on systems science and cybernetics* 2.1 (1966), pp. 22–26.

[13] Aaron Hurst et al. “Not all those who drift are lost: Drift correction and calibration scheduling for the IoT”. In: *arXiv preprint arXiv:2506.09186* (2025).

[14] Kosuke Imai and Teppei Yamamoto. “Causal inference with differential measurement error: Nonparametric identification and sensitivity analysis”. In: *American Journal of Political Science* 54.2 (2010), pp. 543–560.

[15] Guido W Imbens and Donald B Rubin. *Causal inference in statistics, social, and biomedical sciences*. Cambridge university press, 2015.

[16] Siddharth Joshi and Stephen Boyd. “Sensor selection via convex optimization”. In: *IEEE Transactions on Signal Processing* 57.2 (2008), pp. 451–462.

[17] Andreas Krause, Ajit Singh, and Carlos Guestrin. “Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies.” In: *Journal of Machine Learning Research* 9.2 (2008).

[18] Finnian Lattimore, Tor Lattimore, and Mark D Reid. “Causal bandits: Learning good interventions via causal inference”. In: *Advances in neural information processing systems* 29 (2016).

[19] Alexander W Levis, Gabriel Loewinger, and Francisco Pereira. “Causal inference in the closed-loop: Marginal structural models for sequential excursion effects”. In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 109123–109151.

[20] Jing Li and Jionghua Jin. “Optimal sensor allocation by integrating causal models and set-covering algorithms”. In: *IIE Transactions* 42.8 (2010), pp. 564–576.

[21] Ruolin Li, Negar Mehr, and Roberto Horowitz. “Submodularity of optimal sensor placement for traffic networks”. In: *Transportation research part B: methodological* 171 (2023), pp. 29–43.

[22] Dennis V Lindley. “On a measure of the information provided by an experiment”. In: *The Annals of Mathematical Statistics* 27.4 (1956), pp. 986–1005.

[23] Kaibo Liu and Jianjun Shi. “Objective-oriented optimal sensor allocation strategy for process monitoring and diagnosis by multivariate analysis in a Bayesian network”. In: *Iie Transactions* 45.6 (2013), pp. 630–643.

[24] Krithika Manohar, J Nathan Kutz, and Steven L Brunton. “Optimal sensor and actuator selection using balanced model reduction”. In: *IEEE Transactions on Automatic Control* 67.4 (2021), pp. 2108–2115.

[25] Alexandre Martins et al. “Online monitoring of sensor calibration status to support condition-based maintenance”. In: *Sensors* 23.5 (2023), p. 2402.

[26] Wang Miao, Zhi Geng, and Eric J Tchetgen Tchetgen. “Identifying causal effects with proxy variables of an unmeasured confounder”. In: *Biometrika* 105.4 (2018), pp. 987–993.

[27] Raha Moraffah et al. “Causal inference for time series analysis: Problems, methods and evaluation”. In: *Knowledge and Information Systems* 63.12 (2021), pp. 3041–3085.

[28] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. “An analysis of approximations for maximizing submodular set functions—I”. In: *Mathematical programming* 14.1 (1978), pp. 265–294.

[29] Judea Pearl et al. “Models, reasoning and inference”. In: *Cambridge, UK: CambridgeUniversityPress* 19.2 (2000), p. 3.

[30] Sergio Pequito, Soummya Kar, and A Pedro Aguiar. “Minimum cost input/output design for large-scale linear structural systems”. In: *Automatica* 68 (2016), pp. 384–391.

[31] Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. “Causal inference by using invariant prediction: identification and confidence intervals”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 78.5 (2016), pp. 947–1012.

[32] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT press, 2017.

[33] Tom Rainforth et al. “Modern Bayesian experimental design”. In: *Statistical Science* 39.1 (2024), pp. 100–114.

[34] J Runge et al. *Detecting and quantifying causal associations in large nonlinear time series datasets, Sci. Adv., 5, eaau4996*. 2019.

[35] Jakob Runge et al. “Causal inference for time series”. In: *Nature Reviews Earth & Environment* 4.7 (2023), pp. 487–505.

[36] Karthikeyan Shanmugam et al. “Learning causal graphs with small interventions”. In: *Advances in Neural Information Processing Systems* 28 (2015).

[37] Gideon Stein, Maha Shadaydeh, and Joachim Denzler. “Embracing the black box: Heading towards foundation models for causal discovery from time series data”. In: *arXiv preprint arXiv:2402.09305* (2024).

[38] Tyler Summers. “Actuator placement in networks using optimal control performance metrics”. In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE. 2016, pp. 2703–2708.

[39] Tyler H Summers, Fabrizio L Cortesi, and John Lygeros. “On submodularity and controllability in complex dynamical networks”. In: *IEEE Transactions on Control of Network Systems* 3.1 (2015), pp. 91–101.

[40] Jiaqi Zhang et al. “Active learning for optimal intervention design in causal models”. In: *Nature Machine Intelligence* 5.10 (2023), pp. 1066–1075.

Karlsruher Schriftenreihe zur Anthropomatik (ISSN 1863-6489)

Band 1 Jürgen Geisler
Leistung des Menschen am Bildschirmarbeitsplatz.
ISBN 3-86644-070-7

Band 2 Elisabeth Peinsipp-Byma
Leistungserhöhung durch Assistenz in interaktiven Systemen zur Szenenanalyse. 2007
ISBN 978-3-86644-149-1

Band 3 Jürgen Geisler, Jürgen Beyerer (Hrsg.)
Mensch-Maschine-Systeme.
ISBN 978-3-86644-457-7

Band 4 Jürgen Beyerer, Marco Huber (Hrsg.)
Proceedings of the 2009 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-86644-469-0

Band 5 Thomas Usländer
Service-oriented design of environmental information systems.
ISBN 978-3-86644-499-7

Band 6 Giulio Milighetti
Multisensorielle diskret-kontinuierliche Überwachung und Regelung humanoider Roboter.
ISBN 978-3-86644-568-0

Band 7 Jürgen Beyerer, Marco Huber (Hrsg.)
Proceedings of the 2010 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-86644-609-0

Band 8 Eduardo Monari
Dynamische Sensorselektion zur auftragsorientierten Objektverfolgung in Kameranetzwerken.
ISBN 978-3-86644-729-5

Die Bände sind unter www.ksp.kit.edu als PDF frei verfügbar oder als Druckausgabe bestellbar.

Band 9 Thomas Bader
Multimodale Interaktion in Multi-Display-Umgebungen.
ISBN 3-86644-760-8

Band 10 Christian Frese
Planung kooperativer Fahrmanöver für kognitive Automobile.
ISBN 978-3-86644-798-1

Band 11 Jürgen Beyerer, Alexey Pak (Hrsg.)
Proceedings of the 2011 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-86644-855-1

Band 12 Miriam Schleipen
Adaptivität und Interoperabilität von Manufacturing Execution Systemen (MES).
ISBN 978-3-86644-955-8

Band 13 Jürgen Beyerer, Alexey Pak (Hrsg.)
Proceedings of the 2012 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-86644-988-6

Band 14 Hauke-Hendrik Vagts
Privatheit und Datenschutz in der intelligenten Überwachung: Ein datenschutzgewährendes System, entworfen nach dem „Privacy by Design" Prinzip.
ISBN 978-3-7315-0041-4

Band 15 Christian Kühnert
Data-driven Methods for Fault Localization in Process Technology. 2013
ISBN 978-3-7315-0098-8

Band 16 Alexander Bauer
Probabilistische Szenenmodelle für die Luftbildauswertung.
ISBN 978-3-7315-0167-1

Band 17 Jürgen Beyerer, Alexey Pak (Hrsg.)
Proceedings of the 2013 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-0212-8

Die Bände sind unter www.ksp.kit.edu als PDF frei verfügbar oder als Druckausgabe bestellbar.

Band 18 Michael Teutsch
Moving Object Detection and Segmentation for Remote Aerial Video Surveillance.
ISBN 978-3-7315-0320-0

Band 19 Marco Huber
Nonlinear Gaussian Filtering: Theory, Algorithms, and Applications.
ISBN 978-3-7315-0338-5

Band 20 Jürgen Beyerer, Alexey Pak (Hrsg.)
Proceedings of the 2014 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-0401-6

Band 21 Todor Dimitrov
Permanente Optimierung dynamischer Probleme der Fertigungssteuerung unter Einbeziehung von Benutzerinteraktionen.
ISBN 978-3-7315-0426-9

Band 22 Benjamin Kühn
Interessengetriebene audiovisuelle Szenenexploration.
ISBN 978-3-7315-0457-3

Band 23 Yvonne Fischer
Wissensbasierte probabilistische Modellierung für die Situationsanalyse am Beispiel der maritimen Überwachung.
ISBN 978-3-7315-0460-3

Band 24 Jürgen Beyerer, Alexey Pak (Hrsg.)
Proceedings of the 2015 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-0519-8

Band 25 Pascal Birnstill
Privacy-Respecting Smart Video Surveillance Based on Usage Control Enforcement.
ISBN 978-3-7315-0538-9

Band 26 Philipp Woock
Umgebungskartenschätzung aus Sidescan-Sonardaten für ein autonomes Unterwasserfahrzeug.
ISBN 978-3-7315-0541-9

Band 27 Janko Petereit
Adaptive State × Time Lattices: A Contribution to Mobile Robot Motion Planning in Unstructured Dynamic Environments.
ISBN 978-3-7315-0580-8

Band 28 Erik Ludwig Krempel
Steigerung der Akzeptanz von intelligenter Videoüberwachung in öffentlichen Räumen.
ISBN 978-3-7315-0598-3

Band 29 Jürgen Moßgraber
Ein Rahmenwerk für die Architektur von Frühwarnsystemen. 2017
ISBN 978-3-7315-0638-6

Band 30 Andrey Belkin
World Modeling for Intelligent Autonomous Systems.
ISBN 978-3-7315-0641-6

Band 31 Chettapong Janya-Anurak
Framework for Analysis and Identification of Nonlinear Distributed Parameter Systems using Bayesian Uncertainty Quantification based on Generalized Polynomial Chaos.
ISBN 978-3-7315-0642-3

Band 32 David Münch
Begriffliche Situationsanalyse aus Videodaten bei unvollständiger und fehlerhafter Information.
ISBN 978-3-7315-0644-7

Band 33 Jürgen Beyerer, Alexey Pak (Eds.)
Proceedings of the 2016 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-0678-2

Band 34 Jürgen Beyerer, Alexey Pak and Miro Taphanel (Eds.)
Proceedings of the 2017 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-0779-6

Band 35 Michael Grinberg
Feature-Based Probabilistic Data Association for Video-Based Multi-Object Tracking.
ISBN 978-3-7315-0781-9

Band 36 Christian Herrmann
Video-to-Video Face Recognition for Low-Quality Surveillance Data.
ISBN 978-3-7315-0799-4

Band 37 Chengchao Qu
Facial Texture Super-Resolution by Fitting 3D Face Models.
ISBN 978-3-7315-0828-1

Band 38 Miriam Ruf
Geometrie und Topologie von Trajektorienoptimierung für vollautomatisches Fahren.
ISBN 978-3-7315-0832-8

Band 39 Angelika Zube
Bewegungsregelung mobiler Manipulatoren für die Mensch-Roboter-Interaktion mittels kartesischer modellprädiktiver Regelung.
ISBN 978-3-7315-0855-7

Band 40 Jürgen Beyerer and Miro Taphanel (Eds.)
Proceedings of the 2018 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-0936-3

Band 41 Marco Thomas Gewohn
Ein methodischer Beitrag zur hybriden Regelung der Produktionsqualität in der Fahrzeugmontage.
ISBN 978-3-7315-0893-9

Band 42 Tianyi Guan
Predictive energy-efficient motion trajectory optimization of electric vehicles.
ISBN 978-3-7315-0978-3

Band 43 Jürgen Metzler
Robuste Detektion, Verfolgung und Wiedererkennung von Personen in Videodaten mit niedriger Auflösung.
ISBN 978-3-7315-0968-4

Band 44 Sebastian Bullinger
Image-Based 3D Reconstruction of Dynamic Objects Using Instance-Aware Multibody Structure from Motion.
ISBN 978-3-7315-1012-3

Band 45 Jürgen Beyerer, Tim Zander (Eds.)
Proceedings of the 2019 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-1028-4

Band 46 Stefan Becker
Dynamic Switching State Systems for Visual Tracking.
ISBN 978-3-7315-1038-3

Band 47 Jennifer Sander
Ansätze zur lokalen Bayes'schen Fusion von Informationsbeiträgen heterogener Quellen.
ISBN 978-3-7315-1062-8

Band 48 Philipp Christoph Sebastian Bier
Umsetzung des datenschutzrechtlichen Auskunftsanspruchs auf Grundlage von Usage-Control und Data-Provenance-Technologien.
ISBN 978-3-7315-1082-6

Band 49 Thomas Emter
Integrierte Multi-Sensor-Fusion für die simultane Lokalisierung und Kartenerstellung für mobile Robotersysteme.
ISBN 978-3-7315-1074-1

Band 50 Patrick Dunau
Tracking von Menschen und menschlichen Zuständen.
ISBN 978-3-7315-1086-4

Band 51 Jürgen Beyerer, Tim Zander (Eds.)
Proceedings of the 2020 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-1091-8

Band 52 Lars Wilko Sommer
Deep Learning based Vehicle Detection in Aerial Imagery.
ISBN 978-3-7315-1113-7

Band 53 Jan Hendrik Hammer
Interaktionstechniken für mobile Augmented-Reality-Anwendungen basierend auf Blick- und Handbewegungen.
ISBN 978-3-7315-1169-4

Band 54 Jürgen Beyerer, Tim Zander (Eds.)
Proceedings of the 2021 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-1171-7

Band 55 Ronny Hug
Probabilistic Parametric Curves for Sequence Modeling.
ISBN 978-3-7315-1198-4

Band 56 Florian Patzer
Automatisierte, minimalinvasive Sicherheitsanalyse und Vorfallreaktion für industrielle Systeme.
ISBN 978-3-7315-1207-3

Band 57 Achim Christian Kuwertz
Adaptive Umweltmodellierung für kognitive Systeme in offener Welt durch dynamische Konzepte und quantitative Modellbewertung.
ISBN 978-3-7315-1219-6

Band 58 Julius Pfrommer
Distributed Planning for Self-Organizing Production Systems.
ISBN 978-3-7315-1253-0

Band 59 Ankush Meshram
Self-learning Anomaly Detection in Industrial Production.
ISBN 978-3-7315-1257-8

Band 60 Patrick Philipp
Über die Formalisierung und Analyse medizinischer Prozesse im Kontext von Expertenwissen und künstlicher Intelligenz.
ISBN 978-3-7315-1289-9

Band 61 Mathias Anneken
Anomaliedetektion in räumlich-zeitlichen Datensätzen.
ISBN 978-3-7315-1300-1

Band 62 Jürgen Beyerer, Tim Zander (Eds.)
Proceedings of the 2022 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-1304-9

Band 63 Fabian Dürr
Multimodal Panoptic Segmentation of 3D Point Clouds.
ISBN 978-3-7315-1314-8

Band 64 Jutta Hild
Nutzung von Blickbewegungen für die Mensch-Computer-Interaktion mit dynamischen Bildinhalten am Beispiel der Videobildauswertung.
ISBN 978-3-7315-1330-8

Band 65 Jürgen Beyerer, Tim Zander (Eds.)
Proceedings of the 2023 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-1351-3

Band 66 Tobias Michael Kalb
Principles of Catastrophic Forgetting for Continual Semantic Segmentation in Automated Driving.
ISBN 978-3-7315-1373-5

Band 67 Arno Appenzeller
Datensouveränität für Betroffene über persönliche medizinische Daten durch technische Umsetzung einer datenschutzgerechten Forschungsplattform.
ISBN 978-3-7315-1377-3

Band 68 Paul Georg Wagner
Trustworthy Distributed Usage Control Enforcement in Heterogeneous Trusted Computing Environments.
ISBN 978-3-7315-1390-2

Band 69 Anne Borcherding
Use of Accessible Information to Improve Industrial Security Testing.
ISBN 978-3-7315-1400-8

Band 70 Jürgen Beyerer, Tim Zander (Eds.)
Proceedings of the 2024 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory.
ISBN 978-3-7315-1423-7

Band 71 Nadia Burkart
Erklärbare Künstliche Intelligenz - Steigerung der Nachvollziehbarkeit überwachter maschineller Lernverfahren.
ISBN 978-3-7315-1266-0

Band 72 Daniel Stadler
Utilization of Occluded Detections and Target Information in Multi-Person Tracking.
ISBN 978-3-7315-1467-1

Band 73 Andreas Heinrich Specker
Attribute-Based Person Retrieval in Multi-Camera Networks.
ISBN 978-3-7315-1469-5

Band 74 Jürgen Beyerer, Tim Zander (Eds.)
Proceedings of the 2025 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory
ISBN 978-3-7315-1481-7

Lehrstuhl für Interaktive Echtzeitsysteme
Karlsruher Institut für Technologie

Fraunhofer-Institut für Optronik, Systemtechnik
und Bildauswertung IOSB Karlsruhe

The 2025 joint workshop of Fraunhofer IOSB and the KIT Vision and Fusion Laboratory (IES) took place in Triberg-Nussbach, Germany, from July 27th to August 2nd. During the week-long event, doctoral students presented comprehensive reports and debated key topics such as computer vision, industrial production, optimization, control theory, security, and large language models. This book compiles those presentations into detailed technical reports, showcasing the current research landscape of both institutions.

ISSN 1863-6489 (Schriftenreihe)
ISSN 2510-7259 (Tagungsband)
ISBN 978-3-7315-1481-7