DEEP LEARNING-BASED ACCELERATION MODEL FOR RECOVERY ALGORITHMS IN SINGLE PIXEL IMAGING

CARLOS EDUARDO MOGOLLON RAMIREZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECÁNICAS
ESCUELA DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES
BUCARAMANGA
2026

DEEP LEARNING-BASED ACCELERATION MODEL FOR RECOVERY ALGORITHMS IN SINGLE PIXEL IMAGING

CARLOS EDUARDO MOGOLLON RAMIREZ

Degree work presented as a requirement to qualify for the title of Electronic Engineer

Advisor:
Román Alejandro Jácome Carrascal
MSc in Applied Mathematics
Co-Advisor:
Henry Arguello Fuentes
Ph.D in Electrical and Computer Engineering

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECÁNICAS
ESCUELA DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES
BUCARAMANGA
2026

Dedicado a
Diana por ser madre y padre a la vez, ejemplo
de fuerza, sacrificio y amor incondicional.
A mis tías Gloria y Clarita, y mi mamá Irene, por sus enseñanzas,
su amor y presencia en los momentos que más lo necesité.
A Cristina e Ivonne, por ser mis consejeras y por
su compañía y apoyo incondicional.

ACKNOWLEDGEMENTS

A la Universidad Industrial de Santander, por las oportunidades brindadas y por ser el entorno en el cual me desarrollé personal e intelectualmente.
A mis compañeros y amigos de la carrera por todos los momentos que vivimos juntos.
A mi director Roman, por sus consejos, paciencia, acompañamiento y el tiempo dedicado en mi formación academica.
Al profesor Henry y al grupo HDSP por todo el apoyo recibido en mi formación profesional.

TABLE OF CONTENTS

LIST OF FIGURES

Page.

LIST OF TABLES

Page.

RESUMEN

TÍTULO: DEEP LEARNING-BASED ACCELERATION MODEL FOR RECOVERY ALGORITHMS IN SINGLE PIXEL IMAGING [*]

AUTOR CARLOS EDUARDO MOGOLLON RAMIREZ
[**]

PALABRAS CLAVE: Sistemas de un único pixel, Problemas Inversos, Aprendizaje Profundo por Refuerzo, Aprendiendo a Optimizar, Plug-and-Play, Optimización Acelerada.

DESCRIPCIÓN:

Los Sistemas de un único pixel (SPI) ha emergido como un paradigma prometedor en imágenes computacionales debido a su baja complejidad de hardware y su capacidad para operar bajo condiciones de adquisición desafiantes. Sin embargo, la reconstrucción de imágenes de alta calidad en SPI depende de algoritmos de optimización iterativos, como el descenso por gradiente proximal (PGD) y el método de multiplicadores alternantes (ADMM), los cuales pueden presentar una convergencia lenta y una alta sensibilidad a la selección de parámetros. Estas limitaciones se acentúan al integrar denoisers basados en aprendizaje dentro de marcos Plug-and-Play (PnP), donde estrategias de aceleración inadecuadas pueden conducir a inestabilidad o a un desempeño subóptimo. Para abordar estos desafíos, este trabajo propone un enfoque basado en aprendizaje profundo por refuerzo para optimizar dinámicamente el proceso de aceleración de solucionadores iterativos en SPI. En particular, se diseña un agente que selecciona tanto el mecanismo de aceleración como sus hiperparámetros asociados en función de la retroalimentación en tiempo real del proceso de reconstrucción. El método propuesto logra una reducción de hasta 13 veces en el número de iteraciones para PGD, mientras preserva la estabilidad estructural en ADMM incluso bajo condiciones severas. Estos resultados demuestran que formular la reconstrucción iterativa como un sistema de control en lazo cerrado permite un diseño de algoritmos más eficiente y adaptativo en el contexto de imágenes computacionales.

---

[*] Trabajo de Grado

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director: MSc Roman Alejandro Jacome Carrascal. Co-director: Ph.D Henry Arguello Fuentes

ABSTRACT

TITLE: DEEP LEARNING-BASED ACCELERATION MODEL FOR RECOVERY ALGORITHMS IN SINGLE PIXEL IMAGING [*]

AUTHORS: CARLOS EDUARDO MOGOLLON RAMIREZ
[**]

Keywords: Single-Pixel Imaging, Inverse Problems, Reinforcement Learning, Learning to Optimize, Plug-and-Play, Accelerated Optimization

DESCRIPTION:
Single-pixel imaging (SPI) has emerged as a promising computational imaging paradigm due to its low hardware complexity and ability to operate under challenging acquisition conditions. However, high-quality image reconstruction in SPI relies on iterative optimization algorithms, such as Proximal Gradient Descent (PGD) and Alternating Direction Method of Multipliers (ADMM), which can exhibit slow convergence and sensitivity to parameter selection. These limitations become more pronounced when integrating learned denoisers in Plug-and-Play (PnP) frameworks, where improper acceleration strategies may lead to instability or suboptimal performance. To address these challenges, this work proposes a deep reinforcement learning-based approach to dynamically optimize the acceleration process of iterative solvers in SPI. Specifically, we design an agent that selects both the acceleration mechanism and its associated hyperparameters based on real-time feedback from the reconstruction process. The proposed method achieves up to a 13× reduction in the number of iterations for PGD, while preserving structural stability in ADMM even under severe conditions. These results demonstrate that framing the iterative reconstruction as a closed-loop control problem enables a more efficient and adaptive algorithm design for computational imaging.

---

[*] BSc Thesis

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director: MSc Roman Alejandro Jacome Carrascal. Co-director: Ph.D Henry Arguello Fuentes

INTRODUCTION

Conventional image acquisition relies on sensor arrays, such as CMOS or CCD chips, which capture millions of pixels simultaneously. However in applications involving non-visible spectra or high sensitivity requirements, these sensors become prohibitively expensive or technically unfeasible. As an alternative, Single-Pixel-Imaging (SPI) has emerged as a robust optical architecture that utilizes a single point detector combined with a spatial light modulator [1]. Based on compressive sampling principles, this approach significantly reduces hardware costs and enables imaging in harsh environments or spectral ranges where pixelated arrays are not available [2] [3].

To reduce acquisition times in SPI, it is common to collect fewer measurements than the total number of pixels in the target image. This process, known as Compressive Sensing (CS), exploits the inherent sparsity of natural images to reconstruct the scene from sub-sampled data [4] [5]. While this significantly speeds up the physical capture process, it shifts the burden to the computational stage. The reconstruction of the image is no longer a direct mapping but a challenge that requires solving an ill-posed inverse

---

[1] G. M. GIBSON, S. D. JOHNSON, and M. J. PADGETT. "Single-pixel imaging 12 years on: a review". In: *Optics Express* 28 (Sept. 2020), pp. 28190–28208.

[2] L. BIAN et al. "Experimental comparison of single-pixel imaging algorithms". In: *Journal of the Optical Society of America A* 35 (Dec. 2017), p. 78.

[3] I. HOSHI et al. "Real-time single-pixel imaging using a system on a chip field-programmable gate array". In: *Scientific Reports* 12.1 (2022), p. 14097.

[4] E. J. CANDES and M. B. WAKIN. "An Introduction To Compressive Sampling". In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 21–30. DOI: 10.1109/MSP.2007.914731.

[5] W. ZHAO et al. "Comparison of common algorithms for single-pixel imaging via compressed sensing". In: *Sensors (Basel, Switzerland)* 23.10 (2023), p. 4678.

problem to estimate the original signal from incomplete and noisy observations[6].

Plug-and-Play (PnP) methods have emerged as a powerful framework for solving inverse problems by integrating advanced denoisers into iterative optimization algorithms [7] [8]. In this approach, PnP methods replace the proximal operator with a denoiser D_σ, enabling state-of-the-art reconstruction performance while preserving the structure of iterative solvers such as Proximal Gradient Descent (PGD) and the Alternating Direction Method of Multipliers (ADMM) [9] [10]. As a result, PnP methods combine the interpretability of optimization with the expressive power of modern denoising techniques.

Despite their effectiveness, these iterative algorithms typically exhibit sublinear convergence rates, which can limit their practical applicability in large-scale or time-sensitive scenarios. To address this limitation, a variety of acceleration strategies have been proposed, including momentum-based methods, the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA), and Anderson acceleration [11] [12]. These methods aim to improve the

---

[6] Simon FOUCART and Holger RAUHUT. *A Mathematical Introduction to Compressive Sensing*. en. 2013th ed. Applied and Numerical Harmonic Analysis. Secaucus, NJ: Birkhauser Boston, Aug. 2013.

[7] Singanallur V VENKATAKRISHNAN, Charles A BOUMAN, and Brendt WOHLBERG. "Plug-and-Play priors for model based reconstruction". en. In: *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, Dec. 2013, pp. 945–948.

[8] Ulugbek S KAMILOV et al. "Plug-and-Play Methods for Integrating Physical and Learned Models in Computational Imaging: Theory, algorithms, and applications". In: *IEEE Signal Process. Mag.* 40.1 (Jan. 2023), pp. 85–97.

[9] Alexandre D'ASPREMONT, Damien SCIEUR, and Adrien TAYLOR. "Acceleration Methods". In: *arXiv [math.OC]* (Jan. 2021).

[10] S. H. CHAN, X. WANG, and O. A. ELGENDY. "Plug-and-play ADMM for image restoration: Fixed-point convergence and applications". In: *IEEE Transactions on Computational Imaging* 3.1 (2016), pp. 84–98.

[11] Amir BECK and Marc TEBOULLE. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems". In: *SIAM J. Imaging Sci.* 2.1 (Jan. 2009), pp. 183–202.

[12] Homer F. WALKER and Peng NI. "Anderson Acceleration for Fixed-Point Iterations". In: *SIAM*

convergence rate of the underlying optimization algorithm, thereby reducing the number of iterations required to reach its optimal solution. In this work, acceleration is referred to as the process of improving the convergence rate.

However, a key challenge lies in selecting the most appropriate acceleration mechanism for a given problem. While theoretical convergence rates provide global guaranties, the empirical performance of each strategy induces a distinct convergence profile that may not be optimal across all stages of the reconstruction. In particular, the convergence rate of an algorithm is inherently local, depending on the current iterate and the objective function [13]. This suggests that no single acceleration method is universally optimal. Instead, a more robust approach would adaptively select the acceleration mechanism based on the current state of the optimization process, effectively exploiting the local convergence properties to overcome the limitations of fixed-rate solvers.

A related line of work aims to learn the optimization process directly using deep learning, particularly through algorithm unrolling, where each iteration is interpreted as a layer of a neural network and trained end-to-end [14]. These approaches leverage data-driven priors and have shown strong empirical performance in inverse problems [15]. However, they typically follow a fixed, open-loop structure in which the sequence of updates, the number of iterations, and the algorithmic form are determined during training and remain unchanged at inference time. Consequently, they lack the flexibility to adapt

---

Journal on Numerical Analysis 49.4 (2011), pp. 1715–1735. DOI: 10.1137/10078356X. eprint: https://doi.org/10.1137/10078356X.

[13] Jorge NOCEDAL and Stephen WRIGHT. *Numerical Optimization*. en. 2nd ed. Springer Series in Operations Research and Financial Engineering. New York, NY: Springer, July 2006.

[14] Vishal MONGA, Yuelong LI, and Yonina C ELDAR. “Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing”. In: *arXiv [eess.IV]* (Dec. 2019).

[15] Jonas ADLER and Ozan ÖKTEM. “Learned Primal-dual Reconstruction”. In: *arXiv [math.OC]* (July 2017).

to variations in the forward model, noise levels, or the evolving state of the reconstruction.

In contrast, reinforcement learning (RL) provides a natural approach for modeling iterative reconstruction as a sequential decision-making process [16]. By learning state-dependent policies, RL enables closed-loop control of the optimization algorithm, allowing the selection of update strategies that adapt to the current iterate. While traditional RL methods rely on linear approximations or tabular structures, Deep Reinforcement Learning (DRL) integrates deep neural network to approximate these policies, enabling the agent to handle the high-dimensional complexity inherent in image reconstruction states [17]. This makes DRL particularly suitable for inverse problems, where the optimal behavior may vary across iterations and problem instances.

Motivated by this observation, this work proposes a Deep Reinforcement learning (RL)-based approach to dynamically select the most suitable acceleration mechanism at each iteration of the optimization algorithm. By leveraging the expressive power of deep neural networks to parametrize the agent's policy, the model-hereafter referred to as Agent to Accelerate-can effectively process high-dimensional features from the reconstruction state. The proposed method models the iterative solver as an environment, where an agent observes the current state of the reconstruction and decides which acceleration strategy to apply. By learning a state-dependent policy, the agent is able to approximate the optimal local acceleration behavior, leading to improved convergence rates compared to static methods.

---

[16] Ke LI and Jitendra MALIK. "Learning to optimize". In: *arXiv [cs.LG]* (June 2016).

[17] Aske PLAAT. "Deep reinforcement learning, a textbook". In: *arXiv [cs.AI]* (Jan. 2022).

OBJECTIVES

GENERAL OBJECTIVE

To design and implement a deep learning-based acceleration model to optimize the efficiency of recovery algorithms in Single Pixel Imaging, improving reconstruction convergence without compromising image quality.

SPECIFIC OBJECTIVES

1. To model the SPI reconstruction problem and associated traditional algorithms: Develop the mathematical formulation of the SPI inverse problem, including the definition of the acquisition model and the principles of compressive sensing, as well as the detailed description of algorithms such as FISTA and ADMM.

2. To analyze the performance of traditional algorithms (FISTA and ADMM): Evaluate the number of iterations required and the quality of image reconstruction using these algorithms against different sensing conditions in the context of SPI.

3. To design and develop a mathematical model for a neural network as an accelerator of recovery algorithms: Design the architecture and training loss function of the deep neural network, integrating it with FISTA and ADMM algorithms to optimize iterative steps and reduce the number of iterations required for high-quality reconstruction.

4. To implement the designed neural network model: Develop and deploy the designed neural network using deep learning frameworks such as PyTorch to enable efficient training and seamless integration with recovery algorithms.

5. To validate the proposed approach through simulations: Conduct experimental tests in a simulated environment to compare the acceleration and performance of the pro-

posed neural network model with traditional recovery algorithms, evaluating both reconstruction quality (using metrics such as MSE, PSNR, and SSIM) and computational speed.

1. PRELIMINARIES

This chapter provides the theoretical foundation for the proposed acceleration framework. First, the optical architecture of the Single-Pixel Camera (SPC) and its mathematical representation as an inverse problem is described. Subsequently, the PnP framework, which integrates an advanced denoisers into iterative solvers. Finally, the general structure of acceleration mechanisms is established, providing the necessary notation to introduce the proposed learning-based adaptive approach.

1.1 SINGLE-PIXEL IMAGING MODELING

Following the architectural principles discussed in the introduction, the acquisition process in Single-Pixel Imaging (SPI) is governed by the interaction between the scene's spatial information and a series of modulated patterns. The optical architecture considered in this work is the single pixel camera (SPC) [18], this architecture is widely used in compressive imaging systems. This system employs an imaging lens that spatially introduces light, which is previously modulated by a coded aperture (CA), and then integrates the encoded image into a single-pixel detector. An illustration of the SPC system is depicted in Figure 1.1. The CA can be implemented with spatial light modulators (SLM) [19], such as a digital micro-mirror device (DMD)[20], that selectively redirects

---

[18] Marco F DUARTE et al. "Single-pixel imaging via compressive sampling". In: *IEEE signal processing magazine* 25.2 (2008), pp. 83–91.

[19] Carlos A. OSORIO QUERO et al. "Single-pixel imaging: An overview of different methods to be used for 3D space reconstruction in harsh environments". In: *Review of Scientific Instruments* 92.11 (2021), p. 111501. DOI: 10.1063/5.0050358. eprint: https://doi.org/10.1063/5.0050358.

[20] Laura GALVIS, Henry ARGUELLO, and Gonzalo R. ARCE. "Coded aperture design in mismatched compressive spectral imaging". In: *Appl. Opt.* 54.33 (Nov. 2015), pp. 9875–9882. DOI: 10.1364/AO.54.009875.

parts of the light beam [21]. The SPC uses a CA $\mathbf{H}^k_{(i,j)}$ that spatially modulates all the information from the scene $X_{(i,j)}$ with the same pattern, where (i, j) indexes the spatial coordinates and k indexes each captured snapshot. In particular, the CA $\mathbf{H}^k_{(i,j)}$ is a binary pattern whose spatial distribution determines the reconstruction performance. Mathematically, the CA effect over the scene can be represented as:

$$\hat{\mathbf{X}}^k_{(i,j)} = X_{(i,j)}\mathbf{H}^k_{(i,j)}, \tag{1}$$

After that, the modulated scene $\hat{\mathbf{X}}$ is focused on a single spatial point by the condenser lens and captured by a single-pixel detector. The resulting sensing matrix $H \in \mathbb{R}^{m\times n}$ where $n = i \times j$ is the total number of pixels and m is the number of snapshots, contains the vectorization of the CA at each snapshot k in its rows. In this work, the sensing matrix H is constructed using Hadamard patterns following a cake-cutting ordering strategy [22]. This specific ordering prioritizes patterns with higher information content, effectively optimizing the conditioning of the inverse problem.

A key parameter in this acquisition model is the Compression Ratio (CR), defined as $m = CR\times n$. This ratio represents the percentage of total information captured from the scene. Finally, by vectorizing the scene as $x = \text{vec}(X_{(i,j)}) \in \mathbb{R}^n$, the entire acquisition process can be summarized as the inverse problem

$$y = Hx^* + \epsilon \in \mathbb{R}^m, \qquad m \ll n, \tag{2}$$

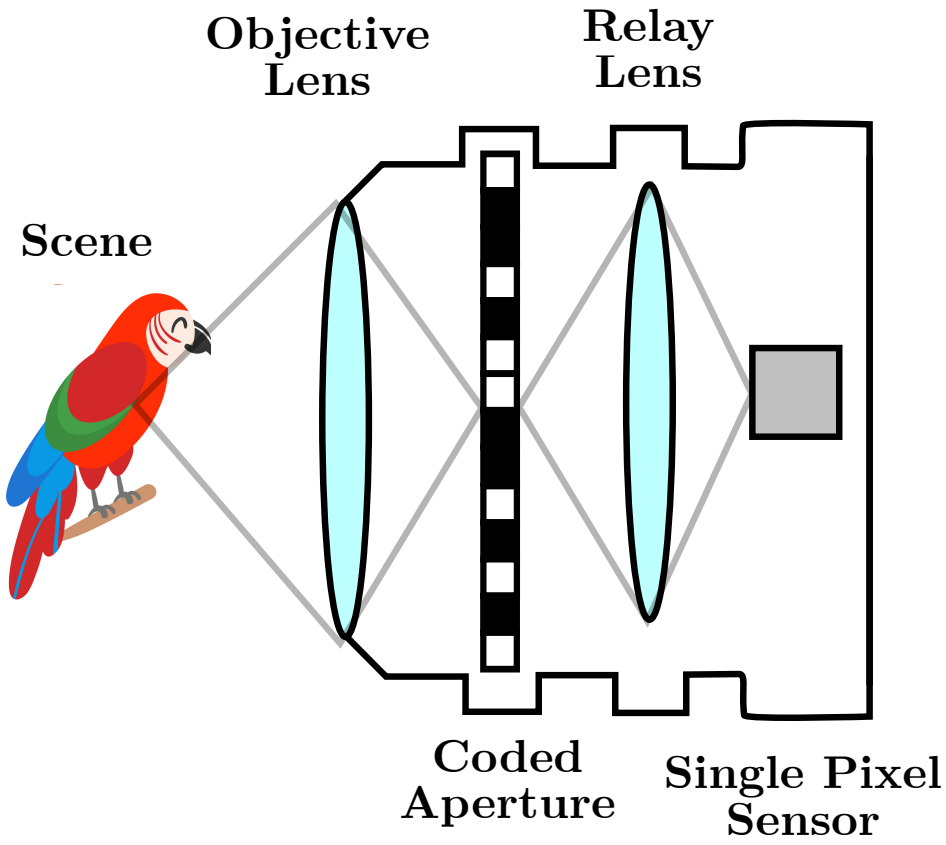
where $H \in \mathbb{R}^{m\times n}$ models the acquisition system and $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ represents additive

---

[21] Andrés JEREZ, Hans GARCIA, and Henry ARGUELLO. "Single Pixel Spectral Image Fusion with Side Information from a Grayscale Sensor". In: *2018 IEEE 1st Colombian Conference on Applications in Computational Intelligence (ColCACI)*. 2018, pp. 1–6. DOI: 10.1109/ColCACI.2018.8484848.

[22] Wen-Kai YU. "Super sub-Nyquist single-pixel imaging by means of cake-cutting Hadamard basis sort". In: *arXiv [eess.IV]* (Mar. 2019).

Figure 1.1. Single pixel camera scheme. A scene is codified by the CA and this coded field is integrated into a single pixel sensor.



Source: Taken from [23]

.

white Gaussian noise.

Due to the ill-posed nature of this problem, recovering x^* requires the incorporation of prior information through a regularized optimization. A common formulation consists of solving

$$\hat{x} = \arg\min_{x} f(x) + \lambda h(x), \tag{3}$$

[23] Roman JACOME, Pablo GOMEZ, and Henry ARGUELLO. "Middle output regularized end-to-end optimization for computational imaging". en. In: *Optica* 10.11 (Nov. 2023), p. 1421.

where f is a data-fidelity term that enforces consistency with the measurements (e.g., $f(x) = \frac{1}{2}\|Hx - y\|_2^2$ under a Gaussian noise model), and h is a regularization term that encodes prior information about the signal, such as sparsity or smoothness.

1.2 PNP-PGD AND PNP-ADMM

1.2.1 Proximal Gradient Descent (PGD) To solve the optimization problem defined in (3), the PGD algorithm alternates between a gradient descent step on the data fidelity term $f(x)$ and a proximal mapping of the regularizer $h(x)$. Following the Forward-Backward splitting scheme, the algorithm is defined as

Algorithm 1 PGD

Input: x_0, α, λ
for $k = 0, 1, 2, \ldots$ **do**
  $u_k = x_k - \alpha \nabla f(x_k)$
  $x_{k+1} = \text{prox}_{\lambda h}(u_k)$
end for

In PnP, the proximal operator of the prior term $\text{prox}_{\lambda h}(\cdot)$ is replaced by a denoiser $D_\sigma(\cdot)$, where the noise level σ is functionally related to the regularization parameter λ.

1.2.2 Alternating Direction Method of Multipliers (ADMM) Unlike PGD, ADMM handles the data and regularization terms independently by introducing an auxiliary variable v. The problem in (3) is reformulated as a constrained optimization

$$\min_{x,v} f(x) + \lambda h(v) \quad \text{subject to} \quad x = v. \tag{4}$$

To solve this, we define the Augmented Lagrangian

$$\mathbf{L}_{\overline{\rho}}(x, z, u) = f(x) + \lambda h(v) + \frac{\overline{\rho}}{2}\| \, x - v + u\| \, , \tag{5}$$

where u is the dual variable and $\overline{\rho}$ is the penalty parameter. The ADMM update rule is derived by minimizing $\mathbf{L}_{\overline{\rho}}$ with respect to each variable sequentially, such that

Algorithm 2 ADMM

Input: $v_0, x_0, u_0, \overline{\rho}, \lambda$
for $k = 0, 1, 2, \ldots$ **do**
  $v_{k+1} = \mathsf{prox}_{\lambda h}(x_k + u_k)$
  $x_{k+1} = \mathsf{prox}_{\overline{\rho} f}(v_{k+1} - u_k)$
  $u_{k+1} = u_k + x_{k+1} - v_{k+1}$
end for

Here, in PnP, the proximal operator of the term prox by a denoiser as well as in PGD and the update in x involves solving a quadratic subproblem

$$x_{k+1} = \arg\min_x \left(f(x) + \frac{\overline{\rho}}{2} \| x - (v_{k+1} - u_k) \|_2^2 \right) = \mathsf{prox}_{\overline{\rho} f}(v_{k+1} - u_k), \tag{6}$$

for the Gaussian noise model $f(x)$ defined in (3), this is equivalent to solving the linear system

$$x_{k+1} = (H^T H + \overline{\rho} I)^{-1} (H^T y + \overline{\rho}(v_{k+1} - u_k)). \tag{7}$$

Both algorithms can be interpreted as fixed-point iterators. To provide a unified representation, we define the variable w_k that contains all variables required at iteration k. In the case of PGD, the variable reduces to $w_k = x_k$, while for ADMM, it is given by $w_k = (v_k, x_k, u_k)$. Under this formulation, both methods can be written as

$$w_{k+1} = T(w_k), \tag{8}$$

where $T(\cdot)$ denotes the update operator defined by the corresponding algorithm.

1.3 ACCELERATION MECHANISMS

Acceleration techniques aim to improve the convergence behavior of iterative algorithms by incorporating information from previous iterates. A general way to model acceleration is through the introduction of an extrapolated variable z_k, defined as

$$z_{k+1} = A_k((x_{k+1-i})_{i=0}^m), \tag{9}$$

where $A_k(\cdot)$ denotes an acceleration operator that depends on a finite history of past iterates.

The accelerated version of an iterative algorithm is obtained by replacing the current iterate x_k in the base update with the extrapolated variable z_k, while keeping the remaining variables unchanged. To formalize this, we define a modified variable $\tilde{w}$, where the component x_k in w_k is replaced by z_k. Therefore, the accelerated update can be written as

$$w_{k+1} = T(\tilde{w}_k) \tag{10}$$

The resulting accelerated algorithms for PGD and ADMM are described in Algorithms 3 and 4, respectively. These formulations provide the foundation for introducing adaptive acceleration strategies. In the next section, we make use of this structure to design a reinforcement learning-based framework that dynamically selects the acceleration mechanism at each iteration.

Algorithm 3 Accelerated PGD

Input: x_0, α, λ
Initialize $z_0 = x_0$
for $k = 0, 1, 2, \ldots$ **do**
  $u_k = z_k - \alpha \nabla f(z_k)$ //Gradient step
  $x_{k+1} = \text{prox}_{\lambda h}(u_k)$ //Proximal of prior
  $z_{k+1} = A_k((x_{k+1-i})_{i=0}^m)$
end for

Algorithm 4 Accelerated ADMM

Input: $x_0, v_0, u_0, \overline{\rho}, \lambda$
Initialize $z_0 = x_0$
for $k = 0, 1, 2, \ldots$ **do**
  $v_{k+1} = \text{prox}_{\lambda h}(z_k + u_k)$ //Proximal of prior
  $x_{k+1} = \text{prox}_{\overline{\rho} f}(v_{k+1} - u_k)$ //Proximal of fidelity
  $u_{k+1} = u_k + x_{k+1} - v_{k+1}$ //Dual variable update
  $z_{k+1} = A_k((x_{k+1-i})_{i=0}^{m})$
end for

The acceleration mechanisms addressed in this work to deine the operator A_k are described below:

- **Momentum (Heavy-ball:** This method incorporates a velocity term that reuse the direction of the previous update. The extrapolated point is computed as $z_{k+1} = x_{k+1} + \rho_k(x_{k+1} - x_k)$, where ρ_k is a momentum coefficient that controls the inertia of the solver.

- **FISTA:** The Fast Iterative Shrinkage-Thresholding algorithm uses a dynamic momentum weight t_k to achieve an optimal convergence rate. The update follows $z_{k+1} = x_{k+1} + \frac{t_k - 1}{t_{k+1}}(x_{k+1} - x_k)$, where $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$, ensuring $\mathcal{O}(\frac{1}{k^2})$ convergence rate for convex problems.

- **Anderson Acceleration (AA):** Unlike first-order momentum, AA computes the extrapolated point as a linear combination of the last m iterates. It solves a least-squares problem to find the coefficients that minimize the residual of the fixed-point iteration, as detailed in Algorithm 5.

Algorithm 5 Anderson Acceleration (memory $m(a_k)$)

Input: $\tilde{w}_k$, memory $m(a_k)$
Compute: $w_{k+1} = T(\tilde{w}_k)$
Define index set: $\mathcal{I}_k = \{0, \ldots, l_k - 1\}; l_k = \min(k, m(a_k))$
Solve the least-squares problem:
$\min_{\beta} \left\| \sum_{i \in \mathcal{I}_k} \beta_i \big(T(\tilde{w}_{k-i}) - \tilde{w}_{k-i}\big) \right\|_2^2 \quad \text{s.t.} \quad \sum_{i \in \mathcal{I}_k} \beta_i = 1$
Update:

$$z_{k+1} = \sum_{i \in \mathcal{I}_k} \beta_i \, T(\tilde{w}_{k-i})$$

Output: z_{k+1}

The acceleration structure defined in (9) offers a flexible template to improve convergence, yet its efficiency remains tied to the choice of the operator A_k. Traditional methods rely on fixed rules that cannot adapt to the changing dynamics of the reconstruction process in Single-Pixel Imaging. This limitation suggests that a static approach is insufficient for optimal performance. Consequently, the next chapter introduces an *Agent to Accelerate*, a framework that replaces fixed rules with an intelligent policy capable of learning the most efficient acceleration path in real-time.

2. RL-BASED ACCELERATION METHOD

We propose a reinforcement learning RL-based approach to adaptively select acceleration mechanisms during the optimization process. The key idea is to treat the iterative solver as a dynamical system whose behavior can be improved by dynamically modifying its input through acceleration mechanisms.

At each iteration k, an agent selects an action $a_k \sim \mathcal{I}$ at a given index from the set of accelerators $\mathcal{I} = \{0, \dots, B-1\}$, where B is the total number of accelerators available to choose from, together with a set of parameters θ_k. The action determines the acceleration operator used to construct the extrapolated variable z_k, while θ_k defines the corresponding hyperparameters of the iterative update.

The resulting update follows the general form

$$w_{k+1} = T(\tilde{w}_k; \theta_k), \quad z_{k+1} = A_{a_k}((x_{k+1-i})_{i=0}^{m(a_k)}), \tag{11}$$

where $\tilde{w}_k$ is obtained from w_k by replacing the component x_k with the extrapolated variable z_k, as introduced previously, $A_{a_k}(\cdot)$ denotes the acceleration operator associated with action a_k, and $m(a_k)$ defines the number of past iterates required by the corresponding method.

This formulation provides a unified representation in which different acceleration strategies can be interpreted as action-dependent operators, enabling their adaptive selection through a learning-based policy. Before introducing the RL formulation, we describe the accelerator library and the corresponding hyperparameters.

2.1 ACCELERATOR LIBRARY

Based on the mechanisms described above, we define a finite library of acceleration operators, where each action $a_k \in \mathcal{A}$ selects an extrapolation strategy applied to the

Table 2.1. Discrete action space for the proposed RL-based acceleration policy. Definition of the acceleration operators A_k available in the library, including their memory $m(a_k)$ and respective update rules.

a_k	Method	$m(a_k)$	Update rule for z_{k+1}
0	None	0	$z_{k+1} = x_{k+1}$
1	FISTA	1	$t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$ $z_{k+1} = x_{k+1} + \frac{t_k - 1}{t_{k+1}}(x_{k+1} - x_k)$
2	Heavy-ball	1	$z_{k+1} = x_{k+1} + \rho_k(x_{k+1} - x_k)$
3	Anderson	3	$z_{k+1} = A_{a_k}\big(\{x_{k+1-i}\}_{i=0}^{3}\big)$
4	Anderson	4	$z_{k+1} = A_{a_k}\big(\{x_{k+1-i}\}_{i=0}^{4}\big)$
5	Anderson	5	$z_{k+1} = A_{a_k}\big(\{x_{k+1-i}\}_{i=0}^{5}\big)$

current iterate. Each operator generates an accelerated point z_{k+1} using a finite memory of past iterates.

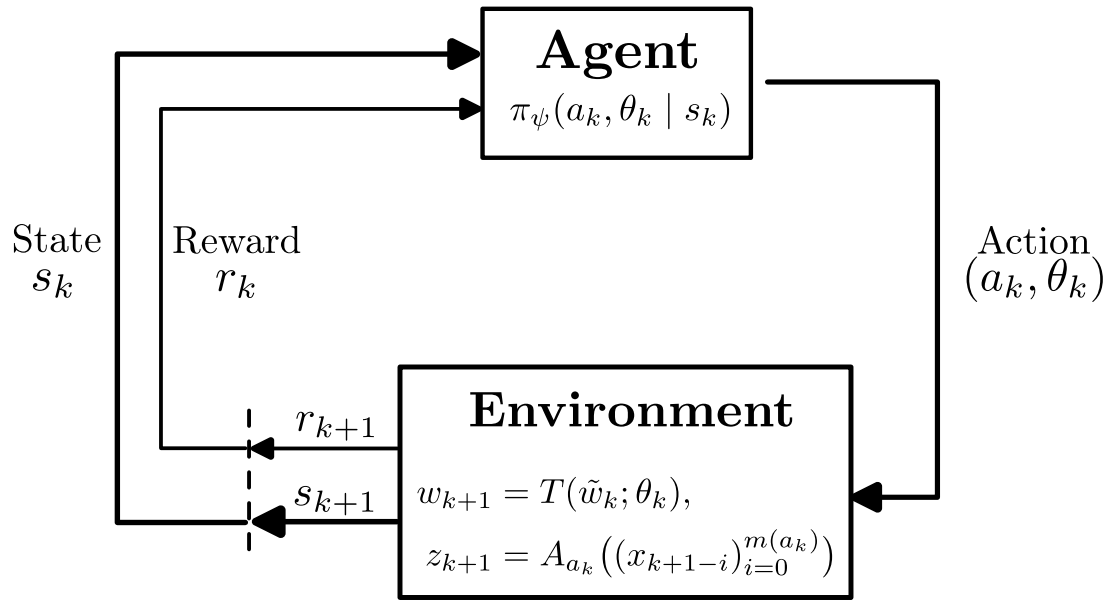
The proposed discrete action space is summarized in Table 2.1, where each action defines a specific acceleration mechanism characterized by its memory $m(a_k)$ and update rule.

In this configuration, the parameter t_k in FISTA is initialized as $t_1 = 1$. The coefficient ρ_k included in the hyperparameters θ_k controls the momentum in Heavy-ball. For the Anderson acceleration actions ($a_k \in \{3, 4, 5\}$), the extrapolated point is computed following the least-squares minimization over the specified memory $m(a_k)$, as detailed in Algorithm 5. By selecting actions dynamically, the proposed policy adapts the acceleration mechanism to the local behavior of the optimization problem.

2.2 RL FORMULATION

The proposed problem can be interpreted as an agent–environment interaction, as illustrated in Fig. 2.1. The iterative solver defined by (11) acts as the environment, while the policy π_ψ selects the action (a_k, θ_k) based on the current state s_k. We formulate

Figure 2.1. Agent–environment interaction for adaptive acceleration. Diagram of the proposed MDP framework where the RL policy observes solver statistics (s_k) to select the optimal accelerator and hyperparameters (a_k, θ_k).



this interaction as a Markov decision process (MDP) [24] formulation for the adaptive acceleration problem. Thus, we define the MDP by the tuple $(\mathcal{S}, \mathcal{A}, P, r, \mu)$, where $\mathcal{S}$ is a finite set with iterator states. In this approach, we define an element of this set as a compact $s_k \in \mathbb{R}^9$ vector, containing only scalar observations of the algorithm

$$s_k = \left[f(x_k), \|\nabla f(x_k)\|_2, c_k, \frac{k}{K}, \frac{a_{k-1}}{B-1}, m_k, \theta_{k-1} \right]. \tag{12}$$

This observation vector contains data fidelity value and the ℓ_2-norm of the gradient, providing useful information to characterize the optimization landscape. $c_k = \frac{\| x_k - x_{k-1} \|_2}{\| x_{k-1} \|_2}$ is the relative error between two successive iterations, providing information on how the signal estimation is stabilizing. k/K provides normalized iteration-encoded information, $\frac{a_{k-1}}{B-1}$ allows learning sequential patterns, $m_k = \frac{\min(k, m(a_k))}{\max_{a \in \mathcal{A}} m(a)}$ is the normalized

[24] Richard S SUTTON and Andrew G BARTO. *Reinforcement learning: an introduction*. en. 2nd ed. Cambridge: The MIT Press, 2015.

effective memory available at iteration k providing information about how much historical information the algorithm can exploit to perform acceleration and θ_{k-1} provides information on the previous hyperparameter values. The action set $\mathcal{A}$ defines the hyperparameters for each iteration, i.e., $a_k, \theta_k \sim \mathcal{A}$. The transition function P is the mapping $s_{k+1} = P(s_t, a_k, \theta_k)$. The reward function r received after action (a_k, θ_k), how the state s_k transitions to state s_{k+1}. We set the per-iteration reward as

$$r_k = (f(x_k) - f(x_{k+1})) + (\xi(x_{k+1}, x^*) - \xi(x_k, x^*)) - w_c \frac{\|x_{k+1} - x^*\|_2}{\|x^*\|_2} - c, \tag{13}$$

where the first term rewards the immediate reduction in the data fidelity term. The function $\xi(\cdot, \cdot)$ computes the peak signal-to-noise ratio (PSNR) and captures progress toward the ground-truth solution. Additionally, w_c is a regularization hyperparameter. The third term penalizes the distance to the reference solution, promoting convergence toward a fixed point and improving stability in the agent's decisions. Finally, a constant step penalty is included to discourage long trajectories, thereby incentivizing faster convergence. Finally, $\mu \in [0, 1]$ denotes the discount factor, which balances the contribution of immediate and future rewards. Based on this, we define the finite-horizon return associated with a trajectory $\tau_k^K = \{s_k, a_k, \theta_k, \ldots, s_{k+K+1}, a_{k+K+1}, \theta_{k+K+1}\}$ as

$$R^\mu(\tau_k^K) = r_k + \sum_{\ell=1}^{K-k} \mu^\ell \, r_{k+\ell}. \tag{14}$$

Recall that K is the total number of iterations. Under this scenario, the ultimate goal in our sequential decision-making is to find a policy $\pi_\psi(a, \theta|s) : \mathcal{S} \to \mathcal{A}$ parameterized with trainable parameters ψ that maximizes the long-horizon reward of the optimization trajectory from the star state ($k = 0$)

$$\pi_\psi^*(a, \theta|s) = \arg\max_{\psi} \mathbb{E}_{\tau_0^K \sim \pi_\psi} \left[R^\mu(\tau_0^K) \right]. \tag{15}$$

where $V^\pi(s) = \mathbb{E}_{\tau \sim \pi}[R^\mu(\tau) \mid s_0 = s]$ is known as the *value* of the state. In this case, the

objective is evaluated from the initial state s_0. To solve (15), we adapt the REINFORCE [25] objective to handle the mixed discrete-continuous action space.

2.3 POLICY OPTIMIZATION

The policy π_ψ is modeled as a lightweight multi-layer perceptron (MLP). A shared two-layer encoder with ReLU activations maps the observation vector $s_k \in \mathbb{R}^9$ to a latent representation $\tilde{s}_k = \mathcal{E}(s_k)$. This latent representation is then processed by multiple task-specific heads that parameterize the action distributions.

Specifically, five heads provide logits (outputs) for: i) update rule index $o_{a_k} = \mathcal{D}_a(\tilde{s}_k) \in \mathbb{R}^B$, ii) gradient step parameter $o_{\alpha_k} = \mathcal{D}_\alpha(\tilde{s}_k) \in \mathbb{R}^2$, iii) regularization parameter $o_{\lambda_k} = \mathcal{D}_\lambda(\tilde{s}_k) \in \mathbb{R}^2$, iv) momentum parameter $o_{\rho_k} = \mathcal{D}_\rho(\tilde{s}_k) \in \mathbb{R}^2$, and v) estimated value head $o_{V_k} = \mathcal{D}_V(\tilde{s}_k) \in \mathbb{R}$. The parameter α_k represents a generalized update parameter: in the case of PGD, it corresponds to the gradient step size, while for ADMM it corresponds to the penalty parameter $\overline{\rho}_k$. For simplicity, we retain the notation α_k throughout. The discrete update rule index is sampled from a Categorical distribution $a_k \sim Categorical(o_{a_k})$. The hyperparameters logits are 2-dimensional since those variables are modeled with a Beta distribution. Therefore, $\alpha_k \sim$ Beta$(o^1_{\alpha_k}, o^2_{\alpha_k})$, $\lambda_k \sim$ Beta$(o^1_{\lambda_k}, o^2_{\lambda_k})$ and $\rho_k \sim$ Beta$(o^1_{\rho_k}, o^2_{\rho_k})$. All of the heads have one linear layer; the output of the continuous parameters has a SoftPlus activation function. For notation purpose, we denote $\psi = \{\mathcal{E}, \mathcal{D}_a, \mathcal{D}_\alpha, \mathcal{D}_\lambda, \mathcal{D}_\rho, \mathcal{D}_V\}$.

The proposed objective is based on three steps: to collect trajectories with the current policy, (ii) compute how much better each action was than expected (the *advantage*), and (iii) update the policy to favor actions with higher returns. The *advantage* term is denoted as $A_k = R^\mu(\tau^K_k) - o_{V_k}$, and the joint log-probability of the sampled action

---

[25] Ronald J WILLIAMS. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". en. In: *Mach. Learn.* 8.3-4 (May 1992), pp. 229–256.

(a_k, θ_k) is given by

$$\log \pi_\psi(a_k, \theta_k \mid s_k) = \log \pi_\psi(a_k \mid s_k) + \sum_{\theta_k \in \{\alpha_k, \lambda_k, \rho_k\}} \log \pi_\psi(\theta_k \mid s_k). \tag{16}$$

which quantifies the likelihood assigned by the policy to the selected acceleration mechanism and the corresponding hyperparameters. It represents the per-iteration log-likelihood of the sampled action (a_k, θ_k) under the policy's distributions.

Our objective function uses the REINFORCE [25] with the value mean squared error and an entropy term as

$$\mathcal{L} = \frac{1}{K} \sum_{k=1}^{K} \Big[- A_k \cdot \log \pi_\psi(a_k, \theta_k \mid s_k) + c_v \left(o_{V_k} - R^\mu(\tau_k^K) \right)^2 - \beta \mathcal{H} \left(\pi_\psi(a_k \mid s_k) \right) \Big], \tag{17}$$

where the entropy term $\mathcal{H}(\cdot)$ encourages exploration by preventing the policy from collapsing prematurely to a single acceleration mechanism or hyperparameter configuration. This promotes a more diverse action distribution and can lead to a smoother optimization landscape.

3. THEORETICAL ANALYSIS

We analyze the proposed RL-based acceleration method from the perspective of switched dynamical systems, with a particular focus on the policy-induced contraction (convergence) factors. The key idea is to interpret the iterative solver, together with the policy-controlled acceleration mechanism, as a state-dependent composition of operators with a switching signal determined by the learned policy.

From (11), the proposed method defines a general update rule that depends on the selected action and its associated hyperparameters. This formulation induces a switched dynamical system of the form

$$w_{k+1} = T_{a_k}(\tilde{w}_k; \theta_k), \tag{18}$$

where $\tilde{w}_k$ denotes the modified variable defined in Section 2, and $T_{a_k}(\cdot)$ represents the operator resulting from the composition of the base update with the acceleration mechanism and parameters specified by (a_k, θ_k). The switching signal $\{a_k\}$ is state-dependent and generated by the policy $\pi_\psi(a, \theta \mid s)$.

To study convergence, we focus on the evolution of the reconstruction error. Let x^* denote a fixed point of the underlying operator. We define the Lyapunov function

$$\Phi_k = \|x_k - x^*\|_2^2, \tag{19}$$

which measures the squared reconstruction error with respect to the solution.

Under standard assumptions on the data-fidelity term f (e.g., L-smoothness) and the non-expansiveness of the proximal or denoising operator, classical fixed-point itera-

tions exhibit a contractive behavior of the form

$$\Phi_{k+1} \leq (1-\gamma)\Phi_k, \tag{20}$$

where $\gamma \in (0,1)$ is a constant contraction factor determined by the problem and the algorithm. Accelerated methods improve this factor, but it remains fixed across iterations. In contrast, the proposed approach induces a time-varying contraction due to the switching mechanism. This can be formalized by defining the contraction factor associated with an action $a \in \mathcal{A}$ at iteration k as

$$\gamma_{a,k} = 1 - \frac{\Phi_{k+1}(a)}{\Phi_k}, \tag{21}$$

where $\Phi_{k+1}(a)$ denotes the value of the Lyapunov function obtained by applying action a at iteration k. By construction, $\gamma_{a,k}$ quantifies the relative decrease in the error induced by action a. Using the definition in (21), the decrease in the Lyapunov function can be written as

$$\Phi_k - \Phi_{k+1}(a) = \gamma_{a,k}\Phi_k, \tag{22}$$

which establishes a direct relationship between error reduction and the contraction factor. The reinforcement learning objective promotes actions that maximize the decrease in the objective function and improve reconstruction quality. From the Lyapunov perspective, this is equivalent to selecting actions that maximize the reduction in Φ_k. Therefore, the policy can be interpreted as approximating

$$a_k^* = \arg\max_{a\in\mathcal{A}} \{\Phi_k - \Phi_{k+1}(a)\} = \arg\max_{a\in\mathcal{A}} \{\gamma_{a,k}\}. \tag{23}$$

As a result, the proposed method induces an effective contraction factor $\gamma_{\mathrm{RL},k} = \gamma_{a_k,k}$, which depends on the selected action. Under an optimal policy, this can be interpreted

as $\gamma_{\mathrm{RL},k} \approx \max_{a \in \mathcal{A}} \gamma_{a,k}$, meaning that the method selects, at each iteration, the action that yields the largest contraction factor among the available acceleration mechanisms. In particular, if the action set $\mathcal{A}$ includes a static acceleration scheme, it follows that, under an optimal policy, $\gamma_{\mathrm{RL},k} \geq \gamma_{\text{static}}$, since the policy selects the action with the largest contraction factor at each iteration.

This defines a state-dependent contraction model

$$\Phi_{k+1} \leq (1 - \gamma_{\mathrm{RL},k})\Phi_k, \tag{24}$$

which, by recursion, yields

$$\Phi_K \leq \Phi_0 \prod_{k=0}^{K-1} (1 - \gamma_{\mathrm{RL},k}). \tag{25}$$

Finally, to determine the number of iterations required to reach a target error ε, we consider the worst-case scenario defined as $\gamma_{\min} = \min_k \gamma_{\mathrm{RL},k}$. This yields the bound $\Phi_K \leq \Phi_0(1 - \gamma_{\min})^K$. Imposing $\Phi_K \leq \varepsilon$ and using standard logarithmic arguments, the iteration complexity is bounded as

$$K_{\mathrm{RL}} = \mathcal{O}\left(\frac{1}{\gamma_{\min}} \log\left(\frac{\Phi_0}{\varepsilon}\right)\right). \tag{26}$$

Consequently, the proposed RL-based acceleration approach can be interpreted as an adaptive switching strategy that tracks the best available local contraction factor at each iteration. This leads to improved convergence behavior compared to static methods, since the policy-induced contraction factors tend to be larger than those of any individual scheme, which is reflected in the dependence of the complexity bound on the worst-case factor $\gamma_{\min}$.

4. EXPERIMENTS AND RESULTS

4.1 SIMULATION SETTINGS

The evaluation of the proposed model was conducted through a series of numerical experiments designed to measure both reconstruction quality and computational efficiency in Single-Pixel Imaging (SPI) systems. The entire reconstruction architecture and the modeling of sensing operators were implemented using DeepInverse [26], which enables efficient integration of deep learning models with physical inverse problems. The sensing process was simulated through a Single-Pixel Camera (SPC) using a sensing matrix H composed of Hadamard patterns with a cake-cutting ordering strategy. Two critical scenarios for the Compression Ratio ($CR = \frac{m}{n}$) were evaluated: 0.1 and 0.3, with the addition of additive white Gaussian noise characterized by a standard deviation of $\sigma = 0.02$.

For the PnP formulation, the DnCNN-Lipschitz [27] was employed as the deep denoiser prior, known for its robustness across various noise levels. For the policy agent training, 5000 images from the Places dataset [28] were utilized, rescaled to a resolution of 128×128 pixels. The policy neural network, the reward system and the optimization engine were developed entirely in PyTorch, fulfilling the development objectives of this work. For the evaluation phase, a representative test set was curated to perform a detailed analysis of the agent's generalization capabilities. This set includes 3 im-

---

[26] Julián TACHELLA et al. "DeepInverse: A Python package for solving imaging inverse problems with deep learning". In: *arXiv [eess.IV]* (June 2025).

[27] Ernest K RYU et al. "Plug-and-play methods provably converge with properly trained denoisers". In: *arXiv [cs.CV]* (May 2019).

[28] Bolei ZHOU et al. "Places: A 10 million image database for scene recognition". en. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 40.6 (June 2018), pp. 1452–1464.

Table 4.1. RL training configuration used in the experiments

	PGD Environment	**ADMM Environment**
Encoder	MLP 8 $\rightarrow$ 128 $\rightarrow$ 128, ReLU	
Accelerator Head	Linear 128 $\rightarrow$ 6, Categorical over [None, FISTA, Momentum, AA($m(a_k) = 3$), AA($m(a_k) = 4$), AA($m(a_k) = 5$)]	
Hyperparameter Heads	3$\times$Linear 128$\rightarrow$2, Beta distribution for each hyperparameter, rescaled to bouds	
Value Head	Linear 128$\rightarrow$1	
Hyperparameter Bounds	$\alpha \in [\max(10^{-5}, 0.1\alpha_0), 5\alpha_0]$, $\lambda \in [0.2\lambda_0, 5\lambda_0]$, $\rho \in [10^{-3}, 1.5]$, $\alpha_0 = \frac{0.1}{\| HH^T \|}$, $\lambda_0 = 0.01$	$\overline{\rho} \in [10^{-3}, 5.0]$, $\lambda \in [0.2\lambda_0, 5\lambda_0]$, $\rho \in [10^{-3}, 1.5]$, $\lambda_0 = 0.01$
Policy optimization	discount $\mu = 0.83, \beta = 10^{-3}$, $w_c = 0.7, c = 0.001$	discount $\mu = 0.96, \beta = 10^{-3}$, $w_c = 0.7, c = 0.0001$
Training	optimizer Adam [29] with lr=10^{-4} K = 500, 10 episodes/epochs	

ages randomly sampled from the Places test partition to ensure unbiased performance on the trained domain, complemented by 'Butterfly' and 'Toucan' benchmark images. These specific images were chosen due to their diverse spatial frequencies and sharp edges, which are critical for evaluating the reconstruction of fine details in compressive sensing scenarios.

The learning process was based on the REINFORCE [25] algorithm with entropy regularization to incentivize exploration within the mixed action space. The agent optimizes a stochastic policy aimed at maximizing the cumulative reward. Table 4.1 shows general training settings of the proposed method for both PGD and ADMM environments.

Notably, the discount factor μ was empirically adjusted to ensure stable convergence in both environments. A value of $\mu = 0.83$ was selected for PGD to balance error reduction with long-term stability. In contrast, a higher value of $\mu = 0.96$ was required for ADMM to account for the more complex dynamics of its split-variable formulation. Additionally, the step penalty c was tuned to prioritize convergence speed while maintaining numerical stability. With these parameters established, the proposed approach was evaluated against traditional baselines to quantify improvements in reconstruction quality and efficiency.

4.2 RESULTS ANALYSIS AND DISCUSSION

This section presents the comparative performance of the proposed RL-based acceleration method against traditional iterative solvers and fixed acceleration schemes. The analysis is divided into two main environments: PnP-PGD and PnP-ADMM, evaluating reconstruction quality, convergence speed and policy behavior.

4.2.1 PnP-PGD Performance Analysis The quantitative results for the PGD-based environment are summarized in Tables 4.2 and 4.3. In the moderate sub-sampling scenario (CR=0.3), the RL Agent achieves a PSNR of 33.160 dB, significantly outperforming the standard baselines. The advantage of the proposed method becomes even more critical in the high compression scenario (CR=0.1), where it maintains a PSNR of 27.060 dB and a superior SSIM of 0.816, as detailed in Table 4.2.

The qualitative impact of the RL policy is illustrated in Figure 4.1, which showcases the reconstruction of standard test images. As observed in the figure, the RL-Selection approach preserves high-frequency details-such as the sharp edges of the "Butterfly" and the intricate textures of the "Toucan"-which are noticeably blurred or smoothed out

Table 4.2. General performance comparison for PnP-PGD. Average PSNR, SSIM, and relative reconstruction error (defined as error $= \frac{\|x_k - x^*\|_2}{\|x^*\|_2}$) for CR=0.3 and CR=0.1 across the test dataset.

CR	**0.3**			**0.1**		
Method	**PSNR**	**SSIM**	**Error**	**PSNR**	**SSIM**	**Error**
PGD	32.361	0.935	0.061	26.814	0.805	0.121
FISTA	32.326	0.932	0.061	26.812	0.803	0.121
Momentum	32.361	0.934	0.061	26.826	0.806	0.121
AA($m(a_k)$=3)	32.361	0.935	0.061	26.817	0.806	0.121
AA($m(a_k)$=4)	32.360	0.935	0.061	26.818	0.806	0.121
AA($m(a_k)$=5)	32.356	0.935	0.061	26.817	0.806	0.121
RL Agent	**33.160**	**0.947**	**0.056**	**27.060**	**0.816**	**0.119**

Table 4.3. Acceleration metrics for PGD (Toucan image). Number of iterations required to reach target PSNR and corresponding acceleration factor (AF) relative to the baseline.

CR	**0.3**		**0.1**	
Target PSNR	**31.853**		**25.390**	
Method	**Iteration**	**AF**	**Iteration**	**AF**
PGD	197	1.00x	212	1.00x
FISTA	108	1.82x	116	1.83x
Momentum	105	1.88x	114	1.86x
AA($m(a_k)$=3)	38	5.18x	41	5.17x
AA($m(a_k)$=4)	40	4.92x	44	4.82x
AA($m(a_k)$=5)	41	4.80x	44	4.82x
RL Agent	**15**	**13.13x**	**31**	**6.84x**

in the baseline PGD reconstructions at lower compression rates.

Efficiency gains are further highlighted in Table 4.3. For the "Toucan" image at CR=0.3, the RL Agent reaches the target PSNR in only 15 iterations, resulting in a massive 13.13x acceleration factor. Under the more challenging CR=0.1 constraint, although the environment's complexity increases, the agent still achieves a 5.84x speed-up, reaching the target in 31 iterations compared to the 212 required by the standard PGD. This behavior is visually supported by Figures 4.2 and 4.3. The PSNR curves in both scenarios show a significantly steeper ascent for the RL Agent compared to any static method. Notably, in the CR=0.1 case (Figure 4.3b), the "Selected Iterator" plot reveals that the policy adapts by increasing the selection frequency of Anderson Acceleration for ($m(a_k) = 3$) and FISTA during the mid-stage of the optimization. This indicates that the agent learns to compensate for the lack of measurements by exploiting the historical information of the iterates more aggressively. Furthermore, the real-time tuning of the hyperparameters (Figures 4.2c and 4.3c) ensures that the solver remains within a stable convergence region even when the sensing matrix H is poorly conditioned due to high compression.

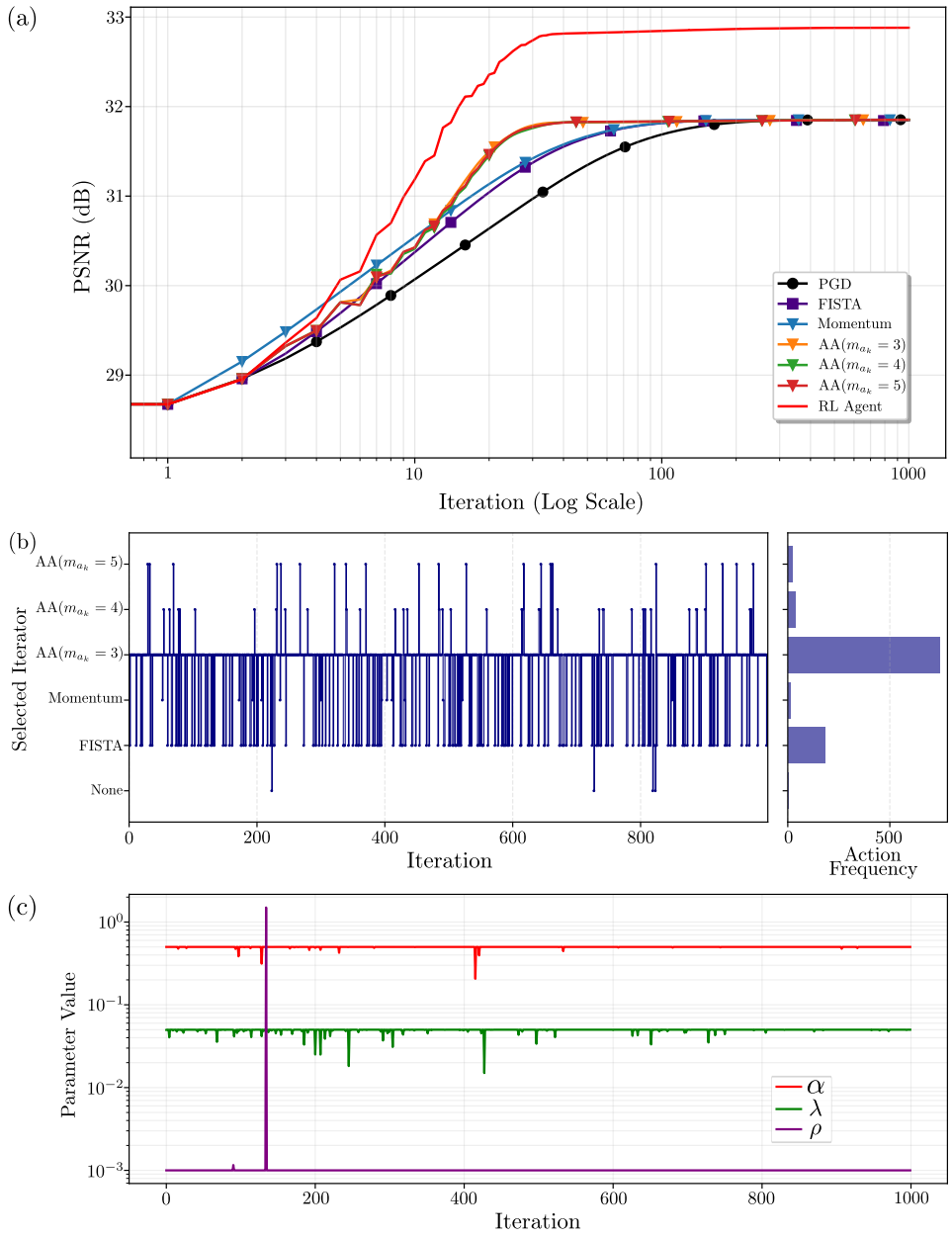
4.2.2 PnP-ADMM Performance Analysis The performance within the ADMM environment is documented in Tables 4.4 and 4.5. While ADMM is a naturally robust splitting algorithm, the RL Agent optimizes its execution to ensure maximum structural stability and convergence speed. As seen in Table 4.4, the agent achieves a superior SSIM of 0.946 for CR=0.3. In terms of convergence speed, Table 4.5 reports that the agent reaches the target SSIM in 21 iterations, providing a 5.62x acceleration over the standard ADMM. This efficiency and the corresponding policy behavior are visually captured in Figure 4.4, where the SSIM evolution demonstrates a fast and stable ascent toward the optimal solution.

However, the high-compression scenario (CR=0.1) presents a significantly more complex landscape due to the poorly conditioned nature of the sensing matrix H. As il-

Figure 4.1. Visual reconstruction comparison for PGD and RL-Selection. Results obtained with CR=0.3 and noise level with $\sigma = 0.02$ for various standard test images, highlighting the preservation of textures in the RL-based approach.

lustrated in Figure 4.5, in this extreme regime, the recovery space becomes severely underdetermined. Rather than seeking aggressive acceleration that could lead to nu-

Figure 4.2. Performance analysis for Toucan image (CR=0.3). (a) PSNR vs iteration (log scale) comparing the RL Agent against static baselines. (b) Adaptive selection of accelerators (left) and their empirical frequency (right). (c) Dynamic evolution of the continuous hyperparameters.



merical divergence, the RL Agent prioritizes stability and structural integrity. While the absolute performance metrics naturally reflect the difficulty of the task, the agent

Figure 4.3. Performance analysis for Toucan image (CR=0.1). (a) PSNR convergence under severe sub-sampling. (b) Adaptive selection of accelerators (left) and their empirical frequency (right). (c) Dynamic evolution of the continuous hyperparameters.

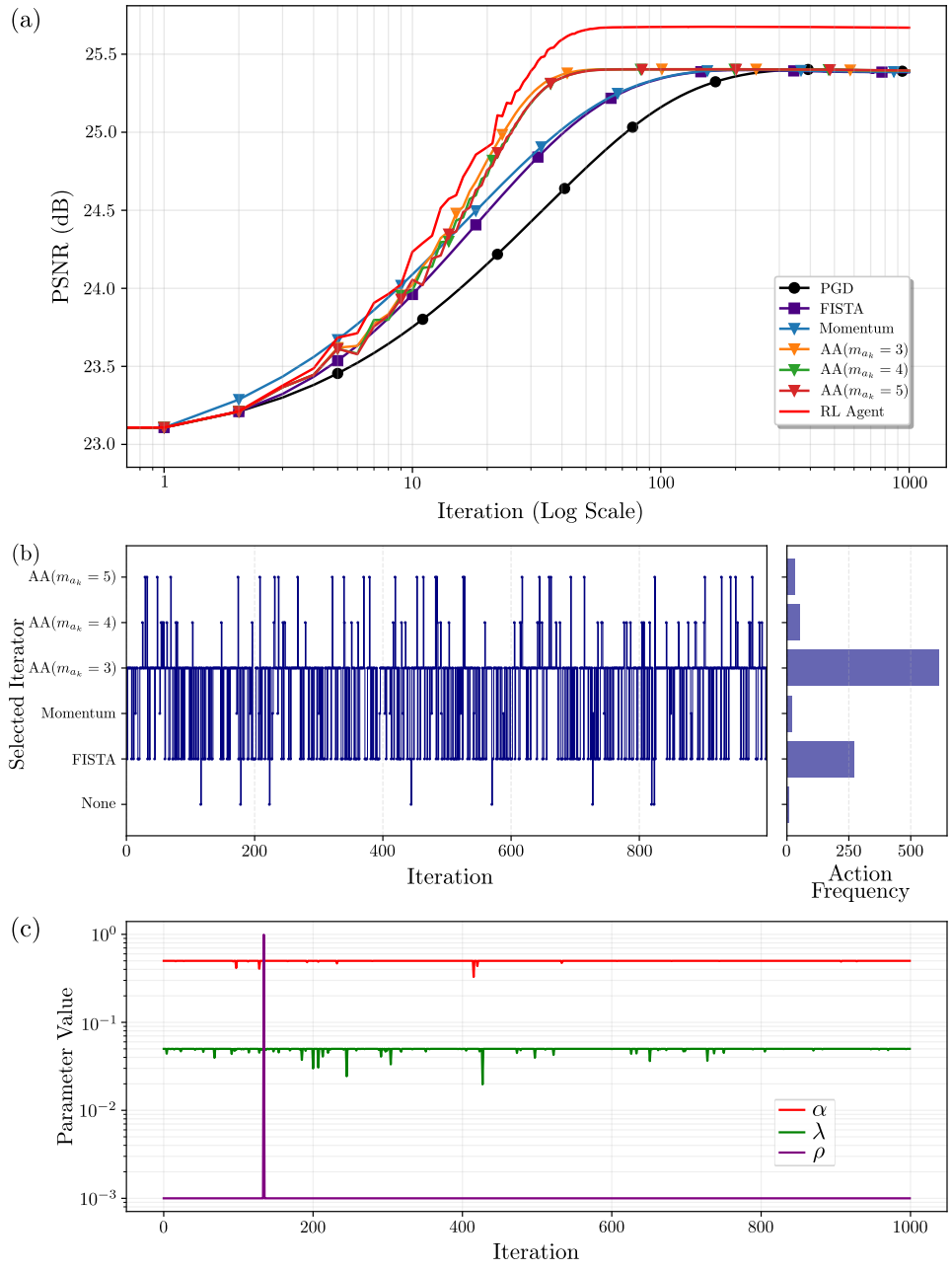


Table 4.4. General performance comparison for PnP-ADMM. Quantitative metrics for the ADMM-based environment evaluating the stability of the RL Agent.

CR	**0.3**			**0.1**		
Method	**PSNR**	**SSIM**	**Error**	**PSNR**	**SSIM**	**Error**
ADMM	32.945	0.939	0.057	27.062	0.816	0.119
FISTA	32.945	0.939	0.057	27.082	0.817	0.118
Momentum	32.945	0.939	0.057	27.080	0.817	0.118
AA($m(a_k)$=3)	32.944	0.939	0.057	27.065	0.816	0.118
AA($m(a_k)$=4)	32.943	0.939	0.057	27.066	0.816	0.118
AA($m(a_k)$=5)	32.943	0.939	0.057	27.068	0.816	0.118
RL Agent	**33.018**	**0.946**	**0.057**	27.012	0.814	0.119

demonstrates a remarkable ability to adapt its policy to mitigate degradation, maintaining a consistent convergence path where fixed-rate accelerators often exhibit instability. The policy behavior in ADMM, shown in the "Selected Iterator" plots of Figures 4.4(b) and 4.5(b), reveals a more conservative switching pattern compared to the PGD environment. The agent learns to carefully balance Anderson Acceleration with FISTA steps, specifically to prevent the dual-variable oscillations that plague traditional solvers when measurements are scarce. Furthermore, the real-time tuning of its hyperparameters (Figures 4.5c) demonstrates that the agent synthesizes a specialized optimization

Table 4.5. Acceleration metrics for ADMM (Toucan image). Number of iterations required to reach target SSIM and corresponding acceleration factor (AF) relative to the baseline.

Target SSIM	**0.9457**	
Method	**Iteration**	**AF**
ADMM	118	1.00x
FISTA	65	1.82x
Momentum	63	1.87x
AA($m(a_k)$=3)	28	4.21x
AA($m(a_k)$=4)	28	4.21x
AA($m(a_k)$=5)	29	4.07x
RL Agent	**21**	**5.62x**

path. This dynamic adaptation allows the solver to remain within a stable contraction region, effectively preserving the most relevant structural features of the scene despite the significant reduction in captured information.

Figure 4.4. SSIM convergence and policy behavior for ADMM (CR=0.3). (a) SSIM index vs iteration (log scale) comparing the RL-Agent against static baselines. (b) Adaptive selection of accelerators (left) and their empirical frequency (right). (c) Dynamic evolution of the continuous hyperparameters.

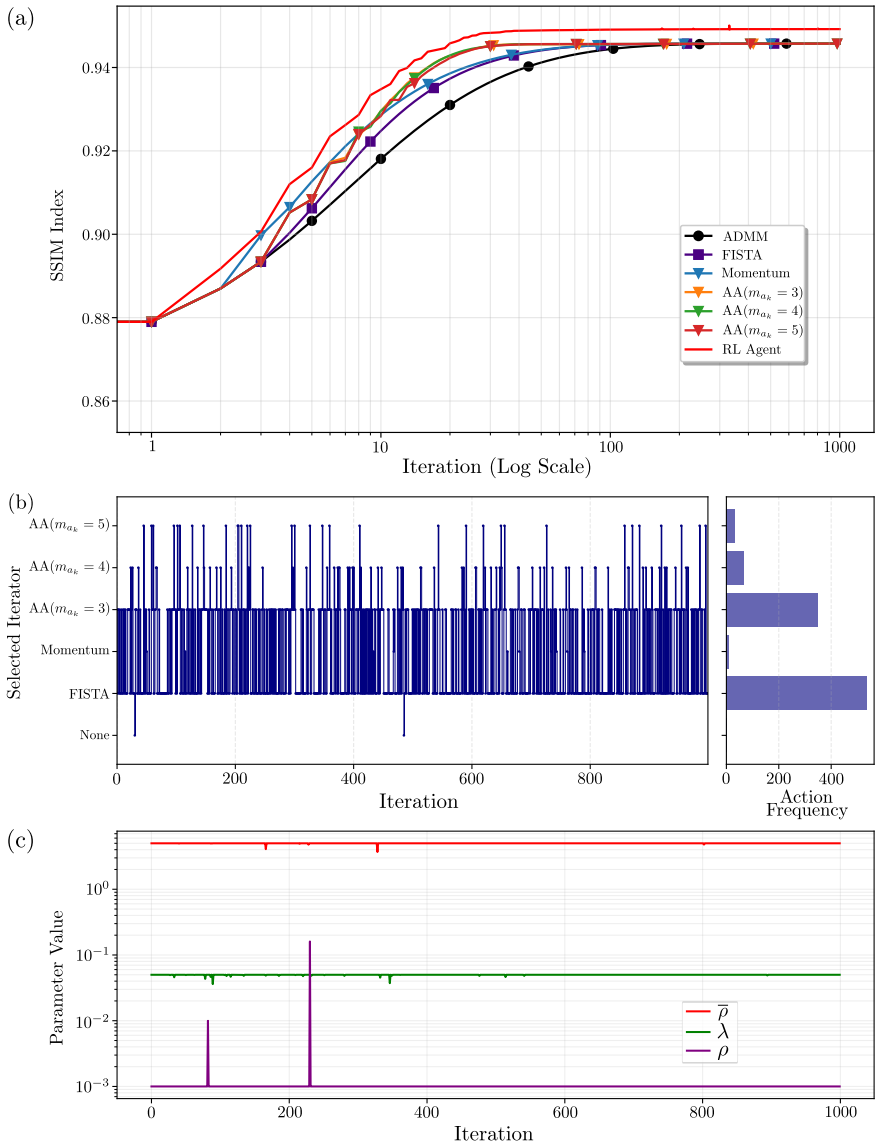
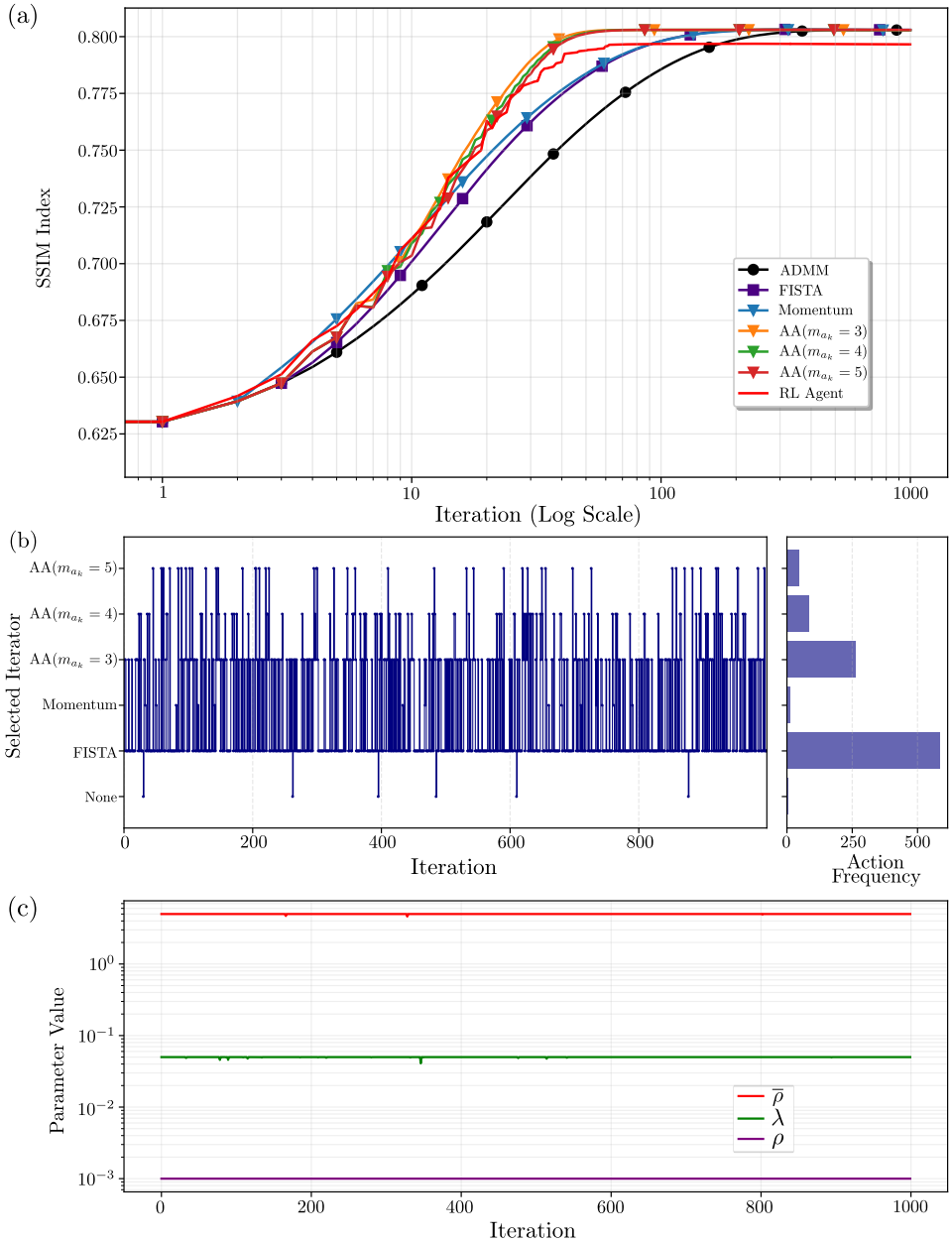


Figure 4.5. SSIM convergence and policy behavior for ADMM (CR=0.1). (a) Analysis of structural similarity stability under high compression constraint (b) Adaptive selection of accelerators (left) and their empirical frequency (right). (c) Dynamic evolution of the continuous hyperparameters.



5. CONCLUSION AND FUTURE WORK

The implementation of the Agent to Accelerate approach demonstrates a significant advancement in the reconstruction of Single-Pixel Imaging. By transitioning from static, handcrafted algorithms to an "intelligent", state-dependent control system, this work proves that autonomous decision-making can effectively overcome traditional optimization bottlenecks. The experimental results validate that the reinforcement learning agent successfully learns to monitor solver statistics in real-time, selecting the optimal acceleration mechanisms and hyperparameters for each iteration. This approach resulted in a reduction of the iteration count by a factor of up to 13x in PGD and ensured numerical stability in ADMM, even under extreme compression constraints (CR=0.1). In summary, this research establishes that integrating deep learning into iterative solvers not only reduces computational overhead but also preserves the structural integrity of the recovered scenes.

In future research, the next stage of this work will focus on expanding the versatility and robustness of the approach. A primary objective is to evaluate the agent's performance across a wider range of inverse problems, such as image deblur and super-resolution, to confirm the universal applicability of the learned policies. Furthermore, further research is required to improve the method's resilience to higher noise levels and varying detection conditions, ensuring reliable performance in real-world optical environments. These advancements will be critical for the transition from simulated reference systems to fully autonomous, high-speed computational image processing systems.

BIBLIOGRAPHY

ADLER, Jonas and Ozan ÖKTEM. “Learned Primal-dual Reconstruction”. In: *arXiv [math.OC]* (July 2017) (cit. on p. 14).

BECK, Amir and Marc TEBOULLE. “A fast iterative shrinkage-thresholding algorithm for linear inverse problems”. In: *SIAM J. Imaging Sci.* 2.1 (Jan. 2009), pp. 183–202 (cit. on p. 13).

BIAN, L. et al. “Experimental comparison of single-pixel imaging algorithms”. In: *Journal of the Optical Society of America A* 35 (Dec. 2017), p. 78 (cit. on p. 12).

CANDES, E. J. and M. B. WAKIN. “An Introduction To Compressive Sampling”. In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 21–30. DOI: 10.1109/MSP.2007.914731 (cit. on p. 12).

CHAN, S. H., X. WANG, and O. A. ELGENDY. “Plug-and-play ADMM for image restoration: Fixed-point convergence and applications”. In: *IEEE Transactions on Computational Imaging* 3.1 (2016), pp. 84–98 (cit. on p. 13).

D’ASPREMONT, Alexandre, Damien SCIEUR, and Adrien TAYLOR. “Acceleration Methods”. In: *arXiv [math.OC]* (Jan. 2021) (cit. on p. 13).

DUARTE, Marco F et al. “Single-pixel imaging via compressive sampling”. In: *IEEE signal processing magazine* 25.2 (2008), pp. 83–91 (cit. on p. 18).

FOUCART, Simon and Holger RAUHUT. *A Mathematical Introduction to Compressive Sensing*. en. 2013th ed. Applied and Numerical Harmonic Analysis. Secaucus, NJ: Birkhauser Boston, Aug. 2013 (cit. on p. 13).

GALVIS, Laura, Henry ARGUELLO, and Gonzalo R. ARCE. “Coded aperture design in mismatched compressive spectral imaging”. In: *Appl. Opt.* 54.33 (Nov. 2015), pp. 9875–9882. DOI: 10.1364/AO.54.009875 (cit. on p. 18).

GIBSON, G. M., S. D. JOHNSON, and M. J. PADGETT. “Single-pixel imaging 12 years on: a review”. In: *Optics Express* 28 (Sept. 2020), pp. 28190–28208 (cit. on p. 12).

HOSHI, I. et al. “Real-time single-pixel imaging using a system on a chip field-programmable gate array”. In: *Scientific Reports* 12.1 (2022), p. 14097 (cit. on p. 12).

JACOME, Roman, Pablo GOMEZ, and Henry ARGUELLO. “Middle output regularized end-to-end optimization for computational imaging”. en. In: *Optica* 10.11 (Nov. 2023), p. 1421 (cit. on p. 20).

JEREZ, Andrés, Hans GARCIA, and Henry ARGUELLO. “Single Pixel Spectral Image Fusion with Side Information from a Grayscale Sensor”. In: *2018 IEEE 1st Colombian Conference on Applications in Computational Intelligence (ColCACI)*. 2018, pp. 1–6. DOI: 10.1109/ColCACI.2018.8484848 (cit. on p. 19).

KAMILOV, Ulugbek S et al. “Plug-and-Play Methods for Integrating Physical and Learned Models in Computational Imaging: Theory, algorithms, and applications”. In: *IEEE Signal Process. Mag.* 40.1 (Jan. 2023), pp. 85–97 (cit. on p. 13).

KINGMA, Diederik P and Jimmy BA. “Adam: A method for stochastic optimization”. In: *arXiv [cs.LG]* (Dec. 2014) (cit. on p. 36).

LI, Ke and Jitendra MALIK. “Learning to optimize”. In: *arXiv [cs.LG]* (June 2016) (cit. on p. 15).

MONGA, Vishal, Yuelong LI, and Yonina C ELDAR. “Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing”. In: *arXiv [eess.IV]* (Dec. 2019) (cit. on p. 14).

NOCEDAL, Jorge and Stephen WRIGHT. *Numerical Optimization*. en. 2nd ed. Springer Series in Operations Research and Financial Engineering. New York, NY: Springer, July 2006 (cit. on p. 14).

OSORIO QUERO, Carlos A. et al. “Single-pixel imaging: An overview of different methods to be used for 3D space reconstruction in harsh environments”. In: *Review of Scientific Instruments* 92.11 (2021), p. 111501. DOI: 10.1063/5.0050358. eprint: https://doi.org/10.1063/5.0050358 (cit. on p. 18).

PLAAT, Aske. “Deep reinforcement learning, a textbook”. In: *arXiv [cs.AI]* (Jan. 2022) (cit. on p. 15).

RYU, Ernest K et al. “Plug-and-play methods provably converge with properly trained denoisers”. In: *arXiv [cs.CV]* (May 2019) (cit. on p. 35).

SUTTON, Richard S and Andrew G BARTO. *Reinforcement learning: an introduction*. en. 2nd ed. Cambridge: The MIT Press, 2015 (cit. on p. 28).

TACHELLA, Julián et al. “DeepInverse: A Python package for solving imaging inverse problems with deep learning”. In: *arXiv [eess.IV]* (June 2025) (cit. on p. 35).

VENKATAKRISHNAN, Singanallur V, Charles A BOUMAN, and Brendt WOHLBERG. “Plug-and-Play priors for model based reconstruction”. en. In: *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, Dec. 2013, pp. 945–948 (cit. on p. 13).

WALKER, Homer F. and Peng NI. “Anderson Acceleration for Fixed-Point Iterations”. In: *SIAM Journal on Numerical Analysis* 49.4 (2011), pp. 1715–1735. DOI: 10.1137/10078356X. eprint: https://doi.org/10.1137/10078356X (cit. on p. 13).

WILLIAMS, Ronald J. “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. en. In: *Mach. Learn.* 8.3-4 (May 1992), pp. 229–256 (cit. on pp. 30, 31, 36).

YU, Wen-Kai. “Super sub-Nyquist single-pixel imaging by means of cake-cutting Hadamard basis sort”. In: *arXiv [eess.IV]* (Mar. 2019) (cit. on p. 19).

ZHAO, W. et al. “Comparison of common algorithms for single-pixel imaging via compressed sensing”. In: *Sensors (Basel, Switzerland)* 23.10 (2023), p. 4678 (cit. on p. 12).

ZHOU, Bolei et al. “Places: A 10 million image database for scene recognition”. en. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 40.6 (June 2018), pp. 1452–1464 (cit. on p. 35).