Generative Plug-and-Play: The Saga Continues†

Charles A. Bouman and Gregery T. Buzzard, Purdue University
Computational Cameras and Displays Workshop
2023 CVPR
June 18, 2023

†Thank you to Showalter Foundation, NSF, ORNL, LANL, GE Healthcare, AFRL, Eli Lilly, and DHS
Historical perspective
  – PnP original recipe
  – Some cool PnP results

Generative PnP Theory:
  – Proximal generators
  – GPnP Theorem

Generative PnP Implementation:
  – Proximal generators and score matching
  – Pseudo-code algorithm

Results

*For details see:
https://github.com/gbuzzard/generative-pnp-allerton
MBIR - Model Based Iterative Reconstruction

- Regularized inversion
- Variable Splitting and proximal maps
- The ADMM Algorithm
Computed Tomographic (CT) Imaging

Parallel Beam CT: synchrotrons, electron microscopy, nano-X-ray sources

Fan Beam CT: Industrial CT

Cone Beam CT: Industrial CT, C-arm Scanners

Multi-Slice Helical CT: Medical, transportation security
CT Forward Model

\[ y = Ax + w \]

**Problems:**
- Not enough measurements: sparse or missing views, etc.
- Low quality data: high noise, low dosage, short exposure, etc.
- Model mismatch: metal, beam-hardening, scatter, poly-energetic, etc.
- Resolution loss: detector blur, motion blur, X-ray spot size, etc.

**Applications:**
- Medical, scientific, industrial, and security

**Q:** How do we resolve these problems for **quantitative** imaging?
Model-Based Iterative Reconstruction (MBIR)

\[ \hat{x} = \arg \min_x \left\{ -\log p(y|x) - \log p(x) \right\} \]

\[ = \arg \min_x \left\{ \frac{1}{2} \|y - Ax\|^2_A - \log p(x) \right\} \]
MBIR: Regularized Image Reconstruction

Forward Model

- Sensor model: \( u_1(x) = -\log p(y|x) = \frac{1}{2} \|y - Ax\|^2_\Lambda \)
- Prior model: \( u_0(x) = -\log p(x) \)
- MBIR Reconstruction

\[ \hat{x} = \arg\min_x \{u_1(x) + u_0(x)\} \]
MBIR: “Thin Manifold” View

Sensor manifold – Based on physical sensor model

Prior manifold – Based on empirical or assumed information

MBIR Reconstruction:

$$\hat{x} = \arg \min_x \{u_1(x) + u_0(x)\}$$
PnP Original Recipe*

- Motivation
- Variable Splitting and proximal maps
- The ADMM Algorithm
- PnP-ADMM

*Singanallur V. Venkatakrishanan, Charles A. Bouman, and Brendt Wohlberg, “Plug-and-Play Priors for Model Based Reconstruction,” IEEE Global Conference on Signal and Information Processing (GlobalSIP), Austin, Texas, USA, December 3-5, 2013.
PnP Motivation

- Uncomfortable facts circa 2013:
  - MBIR is great, but it wasn’t close to the best algorithm for the most basic MBIR problem: **denoising** (MBIR with the identity forward model).
  - Algorithms such as non-local means, BM3D, wavelet shrinkage, bilateral filters, were all much better at denoising than MBIR.

- But **denoising is the most basic** inverse problem:

\[
\hat{x} = \arg \min_x \left\{ \frac{1}{2\sigma^2} \| y - x \|^2 - \log p(x) \right\} = \text{denoise}(y; \sigma)
\]

- Questions:
  - Is there a way to improve on MBIR?
  - Can a denoiser be used as a prior model? There’s nothing to minimize!
Fresh Look at MBIR (circa 2013)

- **Forward model:** \( u_1(x) = -\log p(y|x) \)
- **Prior model:** \( u_0(x) = -\log p(x) \)

**MAP or regularized inverse**

\[ \hat{x} = \arg \min_x \{ u_1(x) + u_0(x) \} \]

Can we minimize these two terms separately?
Proximal Maps

Minimize a function subject to a quadratic penalty on the distance (proximity) to a given base point.

- Proximal map of $f$ with base point $x$:

  $$\bar{F}_0(v) = \arg \min_x \left\{ u_0(x) + \frac{1}{2\gamma^2} \|x - v\|^2 \right\}$$

  - Minimize a function
  - Quadratic “spring” penalty

- Important: $\bar{F}_0(v)$ is an agent that updates solution
Proximal Map Fact: Gradient Step

\[ \bar{F}_0(v) = \arg \min_x \left\{ u_0(x) + \frac{1}{2\gamma^2} \|x - v\|^2 \right\} \]

- **Gradient Step:** For \( \gamma \) small, the proximal map is a gradient step

\[ \bar{F}_0(v) \approx v - \gamma \nabla u_0(v) \]
Proximal Map Fact: Denoiser

\[
\bar{F}_0(v) = \arg\min_x \left\{ u_0(x) + \frac{1}{2\gamma^2} \|x - v\|^2 \right\}
\]

**Denoiser:** When \( u_0(x) = -\log p(x) \), the proximal map is a denoiser

\[
\bar{F}_0(v) = \arg\min_x \left\{ \frac{1}{2\gamma^2} \|v - x\|^2 - \log p(x) \right\}
\]

-Log likelihood for AWGN with variance \( \gamma^2 \)

\[
= \text{Denoise}(v; \gamma)
\]

MAP denoiser for AWGN
Denoisers are Gradient Steps!

- Prior distribution

\[ p(v) = \frac{1}{Z} \exp\{-u_0(x)\} \]

- Then for small \( \gamma \),

\[ v - \gamma \nabla u_0(v) = \text{Denoise}(v; \gamma) \]

- Denoisers are gradient steps for log priors

MAP denoiser for AWGN
Prior Model Proximal Map

\[ \bar{F}_0(v) = \arg \min_x \left\{ \frac{1}{2\gamma^2} \|v - x\|^2 + u_0(x) \right\} \]

- **Interpretation**
  - “Projection” of \( v \) onto prior manifold
  - Denoising operator for white additive Gaussian noise
Forward Model Proximal Map

\[ \bar{F}_1(v) = \arg \min_x \left\{ u_1(x) + \frac{1}{2\gamma^2} \|x - v\|^2 \right\} \]

- "Projection" of \( v \) onto sensor manifold
- MAP estimate with additive white Gaussian noise prior

**Interpretations**
- "Projection" of \( v \) onto sensor manifold
- MAP estimate with additive white Gaussian noise prior
ADMM for MBIR Reconstruction

Initialize $v, u = 0$
Repeat {
    $x \leftarrow \bar{F}_1(v - u)$ \hspace{1cm} // Project onto sensor manifold
    $v \leftarrow \bar{F}_0(x + u)$ \hspace{1cm} // Projection onto prior manifold
    $u \leftarrow u + (x - v)$ \hspace{1cm} // Augmented Lagrangian update
}

- **ADMM:**
  - Iteratively reproject on sensor/prior manifolds
  - Minimizes $u(x) = u_1(x) + u_0(x)$
PnP for MBIR Reconstruction

Initialize $v, u = 0$
Repeat {
    $x \leftarrow F_1(v - u)$ \hspace{1em} // Project onto sensor manifold
    $v \leftarrow \text{Denoise}(x + u)$ \hspace{1em} // Denoise
    $u \leftarrow u + (x - v)$ \hspace{1em} // Augmented Lagrangian update
}

- Big Idea:
  - Replace $F_0$ with any denoiser!
  - Does it still converge? Does it minimize anything?
PnP circa 2013

Forward model: sparse subsampling

\[ u_1(x) = \sum_{s \in \{\text{sampled}\}} \frac{1}{2} \| x_s - y_s \|^2 \]

Prior model: denoising algorithm

Subsamples

Noise std. dev.: 5% of max signal

PnP

\[ \frac{\#}{\&} \in (*+,.-) \]

1 2

\[ \# \& \& - 2 \& 3 \]

Forward

\[ \text{PnP} \]

Prior

Ground Truth

K-SVD

RMSE : 14.11

BM3D

RMSE : 12.56

PLOW

RMSE : 14.54

TV

RMSE : 15.50

q-GGMMRF

RMSE : 15.72
Plug-and-Play Intuition

Question: Does PnP converge?
Answer: Yes, if $F_1$ and $F_0$ are nonexpansive.*

*Or more precisely, $T = (2F_1 - I)(2F_0 - I)$ nonexpansive ensures convergence.
What’s great about PnP

- It produces great results

- It’s modular
  - You only need to train the prior distribution once
  - You can adapt different forward models with the same prior
  - The software is modular too!

- There are lots of denoisers to choose from
Some Cool Results

- Transmission electron microscopy
- 3D reconstruction from sparse views
- 4D reconstruction from sparse views
Bright Field Electron Microscopy
Suhas Shreehari, Purdue/Oak Ridge National Laboratory
Singanallur V Venkatakrishnan, Purdue/Oak Ridge National Laboratory
Greg Buzzard, Purdue
Jeff Simmons, Larry Drummy, AFRL
Charles Bouman, Purdue
3D Bright Field Tomography: Aluminum Spheres (Real) Dataset

67 equi-spaced views from -65° to +65°

Slice 307

100 nm
Aluminum Spheres (Real) Dataset: Reconstructions

FBP

qGGMRF prior

PnP prior
Cone-Beam CT for Imaging AM Parts

Thilo Balke, Soumend Majee, Greg Buzzard, Purdue
Pat Howard, GE Healthcare
Scott Poveromo, Northrop Grumman
Cone-Beam CT

- Cone-Beam Geometry
- Source
- Rotation axis
- Detector
- X-rays
- X - reconstructed image
- Y - measured sinogram

- Beer's Law attenuation
  \[ \int \mu(r)dr = -\log \left( \frac{I_0(u, v)}{I_1(u, v)} \right) \]

- Discretized model
  \[ y = Ax + w \]
4D Recon using PnP/MACE

Soumendu Majee, Purdue
Thilo Balke, Purdue
Craig A. J. Kemp, Eli Lilly
Gregery T. Buzzard, Purdue
Charles A. Bouman, Purdue
4D MBIR Reconstruction

TIMBIR:
- Showed 16x increase in temporal resolution
- Based on simple 4D MRF prior

4D MBIR reconstruction:
\[ \hat{x} \leftarrow \arg \min_x \{- \log p(y|x) - \log p(x)\} \]

Can we do better with 4D PnP prior?
Experimental Setup

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scanner Model</td>
<td>North Star Imaging X50</td>
</tr>
<tr>
<td>Source-Detector Distance</td>
<td>839 mm</td>
</tr>
<tr>
<td>Magnification</td>
<td>5.57</td>
</tr>
<tr>
<td>Cropped Detector Array</td>
<td>731×91, (0.254 mm)$^2$</td>
</tr>
<tr>
<td>Detector resolution at ISO</td>
<td>45.7 µm</td>
</tr>
<tr>
<td>Number of Views per Rotation</td>
<td>150</td>
</tr>
<tr>
<td>Voxel Size</td>
<td>(45.7 µm)$^3$</td>
</tr>
<tr>
<td>Reconstruction Size (x, y, z, t)</td>
<td>731×731×91×16</td>
</tr>
</tbody>
</table>

Other details:
- Object held in place by fixtures: artifacts
- All 4D results undergo preprocessing to correct for jig artifacts
Multi-Slice Fusion: Qualitative Comparison

FBP (3D)  MBIR with 4D prior  PnP:Multi-Slice Fusion
Vial Scan with Force-Curve

- **Scanner parameters:**
  - 758×290 pixels, 3750 views, 25 full rotations
  - Detector spacing: 0.254×0.254 mm²
  - Source-object distance: 152 mm
  - Object-detector distance: 695 mm
  - Magnification: ≈ 5.57

- **Image Parameters (ROR)(rotations 5-8):**
  - 758×758×290×4 voxels
  - Voxel size: (0.05 mm)³
  - Field of view: 38 mm (758 voxels)
Reconstruction (180° per time-point)

FBP

Multi-Slice Fusion
Generative PnP (GPnP):

- Proximal generators
- Markov chains
- Intuition behind GPnP
Can PnP be Generative?

- Problem: PnP only generates a single “best” result

- Question:
  - Can PnP be modified to generate samples from the posterior distribution?
  - What is the posterior distribution?

\[
\hat{x} \sim p_{x|y}(x|y) = \frac{1}{Z} p(y|x)p(x)
\]
The posterior distribution is given by

\[ p(x|y) = \frac{1}{Z} \exp\{-u_1(x) - u_0(x)\} \]

where

\[ u_1(x) = -\log p(y|x) \]
\[ u_0(x) = -\log p(x) \]

**Strategy:**
- Create Markov chain
- Proximal generators: create sequential random samples
- Modular implementation
Proximal Generators

- **Proximal Map**
  
  \[ \bar{F}_0(x) = \arg \min_v \left\{ u_0(v) + \frac{1}{2\gamma^2} \|v - x\|^2 \right\} \]

- **Proximal distribution**
  
  \[ q_0(v|x) = \frac{1}{Z} \exp \left\{ -u_0(v) - \frac{1}{2\gamma^2} \|v - x\|^2 \right\} \]

- **Proximal Generator**
  
  \[ V = F_0(x) \sim q_0(v|x) \]

Generates a sample from the proximal distribution
Interpretation of Proximal Generator

**Intuition:**
- Locally samples from the prior distribution
- Expected change approximates score

\[ p(x) = \frac{1}{Z} \exp\{-u(x)\} \]
Interpretation of Proximal Generator

\[ V = F_0(x) \sim q_0(v|x) \]

**Intuition:**
- Locally samples from the prior distribution
- Expected change approximates score
Generative PnP

Initialize $X = \text{Random}(0, I) + \frac{1}{2}$
Repeat {
    $X \leftarrow F_0(X)$ // Prior Model Proximal Generator
    $X \leftarrow F_1(X)$ // Forward Model Proximal Generator
}
Return($x$)

- Observations/questions:
  - This is a Markov chain
  - Does it converge to a stationary distribution?
  - If so, then what is the stationary distribution?
Theorem: Consider $X_n = F_1(F_0(X_{n-1}))$, then

- $X_n$ is a reversible Markov chain
- $X_n$ has a stationary distribution given by

$$
\tilde{p}(x|y) = \frac{1}{Z} \exp\{-u_1(x) - \tilde{u}_0(x; \gamma^2)\}
$$

- where $\tilde{u}_0(x; \gamma^2)$ is $u_0(x)$ blurred with a Gaussian noise of variance $\gamma^2$.

**Bottom line:**

- Repeated sequential application of $F_0$ and $F_1$ converges to “desired” distribution.
- But GPnP introduces AWGN with variance $\gamma^2$ to the prior distribution!
Generative Plug-and-Play Intuition

\[ u_1(x) \]

\[ \tilde{u}_0(x) \]

Repeat \{ 
\[ x \leftarrow F_0(x) \]  
\[ x \leftarrow F_1(x) \] 
\}
Implementing Proximal Generators:

- Generic implementation
- Prior model proximal generator
- GPnP Pseudo-code
How to implement the Proximal Generator?

- For $\gamma$ small, just add white noise!

\[ F(x) \approx \bar{F}(x) + \gamma W \]
For small $\gamma$ ...

$$\bar{F}_1(v) = \arg\min_x \left\{ \mathcal{W}(x) \left/ \text{Proximal Generator} \right\} + \frac{1}{2\gamma^2}\text{Proximal Map} \right\}$$

"projection" onto "sensor manifold"
For the prior, we know that

\[ F_0(v) = \bar{F}_0(v) + \gamma W \]

\[ \approx \text{Denoise}(v, \gamma) + \gamma W \]

But we will use \textbf{score matching} for:

- More flexible/accurate form
- Easier training (closed form loss function)
- But there is a “catch”…

MAP denoiser for AWGN
Denoising Score Matching (Vincent 2011)*

- **Amazing result:**
  - The AWGN denoiser provides an exact MMSE estimate of the score
    \[ \nabla \tilde{u}_0(x; \sigma^2) \approx \frac{1}{\sigma^2} \left[ \text{Denoise}(x; \sigma) - x \right] \]
  - Exactly true for any \( \sigma \)

- **But…**
  - \( \tilde{u}_0(x; \sigma^2) \) is the energy function for the “noisy” prior
  - So we have the exact solution, but for a **noisy prior**

*P. Vincent, “A connection between score matching and denoising autoencoders,” *Neural Computation*, 2011.*
Interpretation of Denoising Score Matching

Intuition:
- Denoiser moves towards larger probability
- Expected change approximates score

\[-\sigma^2 \nabla \tilde{u}_0(x; \sigma^2) \approx (\text{Denoise}(X) - X)\]
Prior Proximal Generator

- Define
  \[ \beta = \frac{\gamma^2}{\sigma^2} \]

- Using score matching, the prior proximal generator is:
  \[ \tilde{F}_0(x; \beta, \sigma) \approx (1 - \beta) + \beta \text{Denoise}(x; \sigma) + \sqrt{\beta} \sigma W \]

- Remember:
  - \( \tilde{F}_0 \) is based on “noisy” prior, but noise decreases as \( \sigma \to 0 \)
  - More accurate approximation for \( \beta \ll 1 \)
Prior Model Proximal Generator

\[ \tilde{F}_0(x; \beta, \sigma) \approx (1 - \beta) + \beta \text{Denoise}(x; \sigma) + \sqrt{\beta} \sigma W \]

- Prior blurred by \( \sigma \)
- Step size scaled by \( \beta \)
GPnP Basic Algorithm

\( \beta = \frac{1}{4}; \sigma_{\text{max}} = 2; \)
Initialize \( X = \text{Random}(0, I) + \frac{1}{2} \)
Repeat \{ 
    \begin{align*}
    &X \leftarrow (1 - \beta) + \beta \text{Denoise}(x; \sigma) + \sqrt{\beta} \sigma \text{RandN}(0, I) \\
    &X \leftarrow \bar{F}_1(X) + \sqrt{\beta} \sigma \text{RandN}(0, I) \\
    &\sigma \leftarrow \text{Reduce}(\sigma)
    \end{align*}
\}
Return(\(x\))

- Prior is blurred by \((1 + \beta)\sigma^2\)
- But with time \(\sigma \rightarrow 0\)
GPnP Basic Algorithm: Minor Hack

\[ \beta = \frac{1}{4}; \sigma_{\text{max}} = 2; \alpha = 1.3; \]

Initialize \( X = \text{Random}(0, I) + \frac{1}{2} \)

Repeat \{ 

\[ X \leftarrow (1 - \beta) + \beta \text{Denoise}(x; \alpha \sigma) + \sqrt{\beta} \sigma \text{RandN}(0, I) \]

\[ X \leftarrow \bar{F}_1(X) + \sqrt{\beta} \sigma \text{RandN}(0, I) \]

\[ \sigma \leftarrow \text{Reduce}(\sigma) \]

\} 

Return \( x \)

- Prior is blurred by \((1 + \beta)\sigma^2\)
- But with time \( \sigma \to 0 \)
Experiments

- **Experiment:**
  - Prior proximal generator: BM3D, DRUNet*, DDPM denoiser trained on CelebAHQ-256**
  - Forward model: interpolation with sparse sampling of 10%, 5%, 2% and missing rectangle.

- **Parameters**
  - $N = 100; \sigma_{\text{max}} = 0.5$ or $2.0; \sigma_{\text{min}} = 0.005; \beta = 1/4; \alpha = 1.3$;
  - Same parameters work for different problems (interpolation, tomography, …) and different denoisers (BM3D, DRUNet, …).


Sparse interpolation: 10% of pixels sampled, BM3D prior (Std dev intensity window changes)
Sparse interpolation: 10% of pixels sampled, DRUNet prior (Std dev intensity window changes)
Sparse interpolation: 5% of pixels sampled, DRUNet prior (Std dev intensity window changes)
Sparse interpolation: 2% of pixels sampled, DRUNet prior
(Std dev intensity window changes)
Inpainting: Center rectangle omitted - 3 samples, DRUNet prior (Std dev intensity window changes)
Inpainting: Center rectangle omitted - 3 samples, BM3D prior (Std dev intensity window changes)
Inpainting: Center rectangle omitted - 3 samples, DDPM denoiser trained on CelebAHQ-256 prior (Std dev intensity window changes)

IT’S A FACE!!
Conclusions

- **Generative PnP: A natural generalization of PnP original recipe**
  - Denoiser for prior
  - Proximal map for forward model
  - Iterate and add noise

- **GPnP vs Langevin Dynamics**: *
  - Discrete Markov Chain vs Stochastic Differential Equation
  - Proximal Maps vs Gradient Descent
  - New Approach vs Established Method