From Projection to Perception: A Mathematical Exploration of Shadow-based Neural Reconstruction

A research report submitted to the Scientific Committee of the Hang Lung Mathematics Award

Team Number
2596873

Team Members
Wong Yuk To, Hung Kwong Lam
Cheung Tsz Lung, Chan Ngo Tin, Zhou Lam Ho

Teacher
Mr. Chan Ping Ho

School
Po Leung Kuk Celine Ho Yam Tong College

Date
August 30, 2025

Abstract

This paper explores SHADOWNEUS [LWX23], a neural network that reconstructs 3D geometry from single-view camera images using shadow and light cues. Unlike traditional 3D reconstruction methods relying on multi-view cameras or sensors, SHADOWNEUS leverages a neural signed distance field (SDF) for accurate 3D geometry reconstruction. We analyze the training process and uncover its connections to projective geometry, spatial reasoning in $\mathbb{R}^3$, and the neural network's learned geometric representation of space.

Contents

1 Introduction

1.1 Background

3D reconstruction is the process of recovering the shape and structure of an object in $\mathbb{R}^3$ from measured data. It has broad applications in medical imaging (MRI, CT), robotics, augmented/virtual reality (AR/VR), and cultural heritage preservation. Most conventional methods rely heavily on geometric and physical modeling techniques and require rich spatial data obtained through multi-view camera setups, LiDAR sensors, or photogrammetry.

1.2 Motivation

Traditional approaches typically depend on **multiple viewpoints or depth sensors**, which is costly and complex. This leads to the question: **Is it possible to reconstruct 3D geometry from a single fixed camera?** A single image inherently lacks depth information, and multiple 3D points can project to the same 2D pixel location (as discussed in Section 2.5). Therefore, additional cues are essential to resolve this ambiguity.

In exploring this problem, we discovered the paper *ShadowNeuS: Neural SDF Reconstruction by Shadow Ray Supervision* [LWX23] by Jingwang Ling, Zhibo Wang, and Feng Xu. Their approach demonstrates that leveraging neural signed distance fields and supervising the network with shadow ray information under varying lighting enables accurate 3D reconstruction from single-view images. Motivated by their work, we present a study analyzing their method and propose a 1D-to-2D experimental validation to examine our understanding.

2 Fundamentals of 3D Reconstruction from 2D Images

2.1 3D Reconstruction in Computer Vision

According to an article on Medium [VK23], 3D computer vision is a field of computer science focusing on the analysis, interpretation, and understanding of three-dimensional visual data.

The article highlights a traditional approach for 3D reconstruction:

Structure from Motion (SfM)
Recover the 3D structure by estimating camera positions from multiple images.

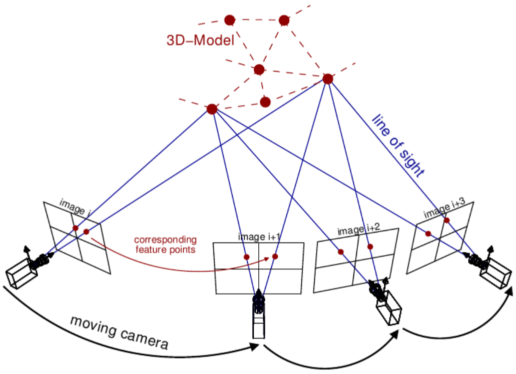


Figure 1. Structure from Motion

2.2 Information Encoded in a Single-View Image

When only single-view is available, the following information remains exploitable:

- **Pixel position** (u, v) — relates to a 3D ray from camera to that pixel
- **Color / intensity** — encodes surface information (material, orientation, etc.)
- **Embedded features** (edges, recognized objects) — encode geometry cues

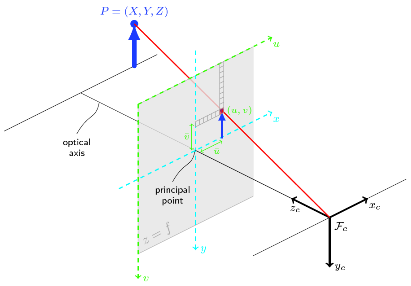


Figure 2. Ray from camera to pixel

Figure 3. Texture

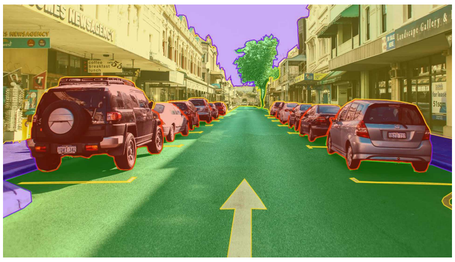


Figure 4. Object segmentation (YOLO)

2.3 Forward Projection: Mapping from 3D to 2D

In this section, we want to show that mapping from a 3D point $\boldsymbol{P} = (x, y, z)^\top \in \mathbb{R}^3$ in the world coordinate system onto a 2D image pixel coordinate $\boldsymbol{p} = (u, v)^\top \in \mathbb{R}^2$ is a projection-like process.

Extrinsic parameters: define the camera's position and orientation relative to the world:

$$\underbrace{C = \begin{bmatrix} C_x \\ C_y \\ C_z \end{bmatrix} \in \mathbb{R}^3}_{\text{Camera center}}, \quad \underbrace{R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \in \mathrm{SO}(3)}_{\text{Rotation matrix}}, \quad t = \underbrace{-RC \in \mathbb{R}^3}_{\text{Translation vector}} \tag{1}$$

Intrinsic parameters: encode the internal camera geometry:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{2}$$

where (f_x, f_y) are the focal lengths in pixels, and (c_x, c_y) is the principal point (image center).

The forward projection consists of three steps:

1. Transform $\boldsymbol{P}$ from world coordinates to camera coordinates: $\boldsymbol{P}_{\text{cam}} = R(\boldsymbol{P} - C) = R\boldsymbol{P} + t$.
2. Project camera coordinates to homogeneous image coordinates: $\boldsymbol{P}_{\text{hom}} = K\boldsymbol{P}_{\text{cam}} = \begin{bmatrix} p'_x \\ p'_y \\ z' \end{bmatrix}$.
3. Normalize by depth z' to get pixel coordinates: $\boldsymbol{p} = \frac{1}{z'} \begin{bmatrix} p'_x \\ p'_y \end{bmatrix}, \quad z' \neq 0.$

Combined expressions:

$$\boxed{\boldsymbol{P}_{\text{hom}} = \begin{bmatrix} p'_x \\ p'_y \\ z' \end{bmatrix} = K[R \mid t] \begin{bmatrix} \boldsymbol{P} \\ 1 \end{bmatrix}, \quad \boldsymbol{p} = \frac{1}{z'} \begin{bmatrix} p'_x \\ p'_y \end{bmatrix}} \tag{3}$$

2.4 The Inverse Problem: From 2D Image to 3D World

After understanding the forward projection process, 3D reconstruction can be viewed as the inverse calculation: recovering 3D points from their 2D image projections.
We now tackle the challenge of inverting the projection function introduced in Section 2.3.

Lemma 2.4.1 (Camera Ray Parametrization)**.**
Given a pixel $p = [p_x, p_y]^\mathsf{T}$ and camera parameters (K, R, C), the corresponding 3D points lie on a ray:

$$\boxed{P(\lambda) = C + \lambda\, d, \quad \lambda > 0} \tag{4}$$

where the ray direction is

$$\boxed{d = R^{-1} K^{-1} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}} \tag{5}$$

Remark 2.4.1 (Normalization)**.**
The vector d can be normalized to unit length for physical interpretation, but this is not essential for the ray parametrization.

Proof. Starting from the forward projection (3):

$$K(RP + t) = z' \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix},$$

$$RP + t = z' K^{-1} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix},$$

$$P = R^{-1}\big(z' K^{-1} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} - t\big).$$

Since $t = -RC$, we have $-R^{-1}t = C$. Setting $\lambda = z'$, the parametric form of the ray follows:

$$P(\lambda) = C + \lambda\, R^{-1} K^{-1} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}.$$

□

By parametrizing the ray originating at C in the direction d, we capture the geometric meaning that a single pixel in an image does not correspond to a unique 3D point but rather to an infinite set of points lying along this ray.

2.5 Ill-posedness and Ambiguity in Single-View Reconstruction

Recovering 3D points from a single 2D image is **ill-posed**, as defined by Hadamard: the solution may be non-unique, unstable, or nonexistent.

Proposition 2.5.1 (Ill-posed Nature of Single-View Reconstruction)**.**

(a) The depth λ is unknown.

(b) Each pixel $\boldsymbol{p}$ corresponds to infinitely many 3D points along a ray.

(c) Additional constraints are needed for unique reconstruction.

This ill-posedness motivates using extra cues like multiple views (SfM), depth sensors, or shadow information.

3 Shadows as Geometric Constraints

How much can shadows reveal about 3D geometry from a single image?
Historically, *shadow carving* [SHFP01] used multiple shadows to constrain 3D shape, while classical *descriptive geometry* [M51] projected shadows to reconstruct surfaces. Even from one image, shadows reveal depth: points in shadow lie behind occluders, shadow edges indicate tangent rays constraining local geometry, and illuminated points satisfy the opposite inequality along the light direction.

3.1 Light Rays, Surface Normals, and Shadow Formation

Definition 3.1.1 (Light Ray)**.**
For a point light source $\boldsymbol{L} \in \mathbb{R}^3$ and a surface point $\boldsymbol{P} \in \mathbb{R}^3$, the light ray is

$$\boxed{r(t) = \boldsymbol{L} + t(\boldsymbol{P} - \boldsymbol{L}), \quad t \in [0,1].}$$

Definition 3.1.2 (Shadow Occlusion Test)**.**
A point $\boldsymbol{P}$ is in shadow if there exists $t \in (0,1)$ such that the light ray intersects another surface $\mathscr{S}$:

$$\boxed{r(t) \cap \mathscr{S} \neq \emptyset, \quad t \in (0,1).}$$

Remark 3.1.1 (Physical Interpretation)**.**
The interval $(0,1)$ excludes the light source ($t = 0$) and the target point ($t = 1$), ensuring the test only checks for obstructions between $\boldsymbol{L}$ and $\boldsymbol{P}$.

Proposition 3.1.1 (Surface Normal Illumination Test)**.**
If a point is not occluded, its illumination depends on the sign of the dot product between the light direction and the surface normal $\mathbf{n}(\mathbf{P})$:

$$\boxed{\begin{array}{ll} (\mathbf{P}-\mathbf{L})\cdot\mathbf{n}(\mathbf{P}) > 0 & \text{illuminated surface,} \\ (\mathbf{P}-\mathbf{L})\cdot\mathbf{n}(\mathbf{P}) < 0 & \text{self-shadowed surface,} \\ (\mathbf{P}-\mathbf{L})\cdot\mathbf{n}(\mathbf{P}) = 0 & \text{shadow boundary / tangency.} \end{array}}$$

Remark 3.1.2 (Geometric Interpretation)**.**
The sign of $(\boldsymbol{P}-\boldsymbol{L})\cdot\boldsymbol{n}(\boldsymbol{P})$ reflects the relative orientation between the light direction and the surface normal:

- Positive — surface faces the light (lit).
- Negative — surface faces away from the light (self-shadowed).
- Zero — light direction tangent to the surface (shadow boundary).

3.2 Depth Recovery with Shadow Constraints

Recovering the 3D point $\mathbf{P}(\lambda)$ corresponding to a pixel $\mathbf{p}$ requires solving for the depth parameter λ along the camera ray (Lemma 2.4.1)

$$\mathbf{P}(\lambda)=\mathbf{C}+\lambda\mathbf{d},$$

where $\mathbf{C}$ is the camera center and $\mathbf{d}$ is the ray direction.

To uniquely determine λ, we can use the surface normal illumination test (Proposition 3.1.1). It gives three possible cases under different illuminace siuations that we can obtained from the image. λ can be solved using both the camera ray and the equation from the test.

$$\boxed{\begin{cases}(\mathbf{P}(\lambda)-\mathbf{L})\cdot\hat{\mathbf{n}}>0 & \text{(illuminated surface)},\\ (\mathbf{P}(\lambda)-\mathbf{L})\cdot\hat{\mathbf{n}}=0 & \text{(shadow boundary)},\\ (\mathbf{P}(\lambda)-\mathbf{L})\cdot\hat{\mathbf{n}}<0 & \text{(attached shadow region)}.\end{cases}} \tag{6}$$

These three cases correspond to different inequalities and equalities on λ, leading to three possible solutions making the depth λ satisfies:

$$\boxed{\begin{aligned}&\lambda>\frac{(\mathbf{L}-\mathbf{C})\cdot\hat{\mathbf{n}}}{\mathbf{d}\cdot\hat{\mathbf{n}}} && \text{(point is illuminated)},\\ &\lambda=\frac{(\mathbf{L}-\mathbf{C})\cdot\hat{\mathbf{n}}}{\mathbf{d}\cdot\hat{\mathbf{n}}} && \text{(on shadow boundary)},\\ &\lambda<\frac{(\mathbf{L}-\mathbf{C})\cdot\hat{\mathbf{n}}}{\mathbf{d}\cdot\hat{\mathbf{n}}} && \text{(in attached shadow region)},\end{aligned}}\quad \text{where } \mathbf{d}\cdot\hat{\mathbf{n}}\neq 0. \tag{7}$$

In practice, the observed lighting condition at pixel $\mathbf{p}$ selects the correct relation above, enabling unique recovery of λ.

3.3 Limitations and Challenges

- **Circular dependency:** Depth & normals depend on each other.
- **Singular cases:** The solution is undefined when $\boldsymbol{d}\cdot\hat{\boldsymbol{n}}=0$ (grazing angles).
- **Shadow detection:** Errors in identifying shadow boundaries propagate to depth estimation.
- **Nonlinear systems:** Solving requires iterative methods that may face convergence issues.

4 ShadowNeuS: Neural Shadow-Based 3D Reconstruction

ShadowNeuS tackles single-view 3D reconstruction under controlled conditions:

- **Single camera viewpoint**: Fixed camera intrinsics and extrinsics.
- **Simple lighting conditions**: Each images are captured under a known light direction.
- **Static scene**: No object motion between lighting conditions.
- **Observable shadows**: Clear and well-defined shadow boundaries in the captured images.

This setup allows ShadowNeuS to leverage shadow cues to recover complete 3D geometry, including occluded or non-visible regions, by combining classical geometric reasoning with neural optimization.

4.1 Classical vs Neural Approaches: Method Comparison

Figures 5 and 6 illustrate the processing pipelines for classical geometric methods and ShadowNeuS respectively.

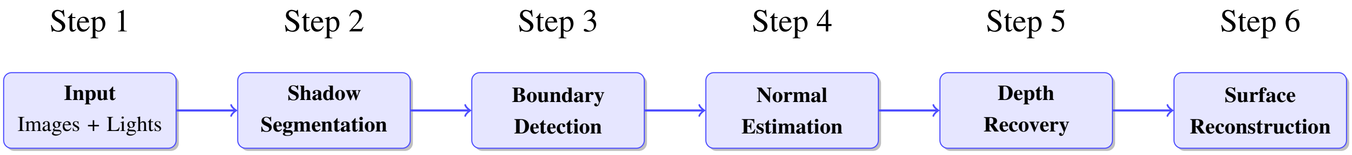


Figure 5: Classical method (Sequential processing pipeline)

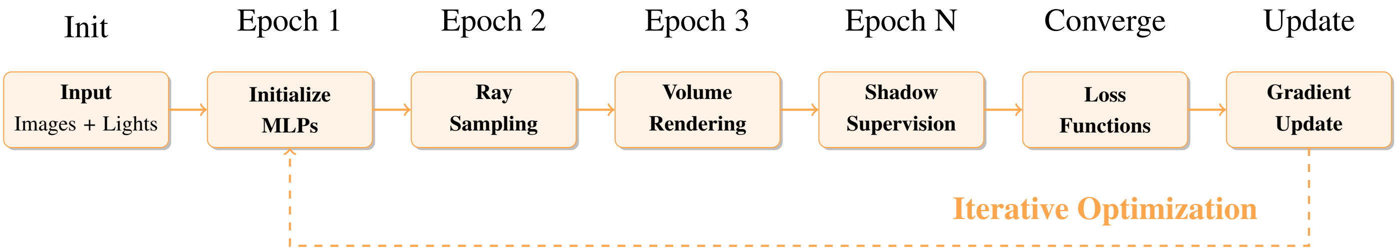


Figure 6: ShadowNeuS (End-to-end neural optimization pipeline)

Classical Method Issues	Neural Solution (ShadowNeuS)
• Circular dependency • Singularities • Shadow detection • Nonlinear equations	• Joint learning with neural SDF • Differentiable approximation • Binary shadow models • End-to-end optimization

4.2 ShadowNeuS Pipeline

In this section, we outline the complete pipeline of ShadowNeuS, from the initialization of neural SDFs to optimization using shadow supervision.

4.2.1 Neural Signed Distance Fields (SDF)

Definition 4.2.1 (Neural Signed Distance Field)**.**
A **Neural Signed Distance Field** (Neural SDF), as introduced in [PFS19], is a function $f(\mathbf{P};\theta):\mathbb{R}^3 \to \mathbb{R}$ parameterized by a multi-layer perceptron (MLP) with trainable weights θ. It implicitly represents a 3D scene by outputting the signed distance of any spatial point $\mathbf{P} = (p_x, p_y, p_z)^\top$ to the nearest surface:

$$f(\mathbf{P}) \begin{cases} < 0 & \text{if } \mathbf{P} \text{ is inside the object (blue region),} \\ = 0 & \text{if } \mathbf{P} \text{ lies on the surface (green line),} \\ > 0 & \text{if } \mathbf{P} \text{ is outside the object (red region).} \end{cases}$$

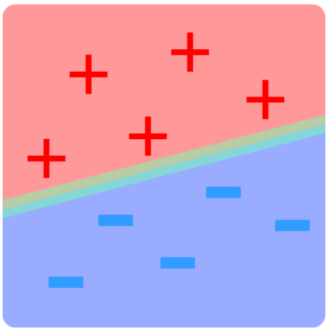


Figure 7. Signed regions of a surface

Remark 4.2.1 (Key Properties of Neural SDF)**.**
Neural SDFs encode useful geometric details for 3D reconstruction:

- **Multivariable Differentiability**: The signed distance field is expressed as $f(\mathbf{P};\theta)$, where $\mathbf{P}$ denotes a 3D point and θ the neural network parameters. It is differentiable with respect to both:

$$\begin{cases} \nabla_{\mathbf{P}} f(\mathbf{P};\theta) & \text{(Use for spatial constraint or surface normal in the scene)} \\ \nabla_{\theta} f(\mathbf{P};\theta) & \text{(Use for gradient descent in machine learning)} \end{cases}$$

- **Surface Normals**: On the zero level set $f(\mathbf{P};\theta) = 0$, the surface normal is given by the normalized spatial gradient:

$$\hat{\mathbf{n}}(\mathbf{P}) = \frac{\nabla_{\mathbf{P}} f(\mathbf{P};\theta)}{\|\nabla_{\mathbf{P}} f(\mathbf{P};\theta)\|_2}$$

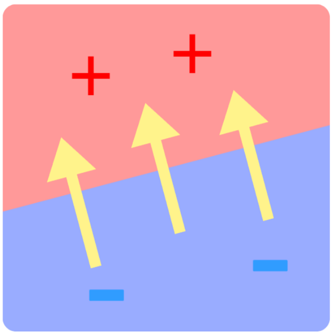


Figure 8. The gradient of the sdf

This normal is used in light–surface interactions such as shadow visibility, computed via the angle between $\hat{\mathbf{n}}$ and the incoming light direction $\mathbf{L} - \mathbf{P}$.

- **Eikonal Regularization (Distance Consistency)**: For $f(\mathbf{P})$ to represent a valid distance function locally, it must satisfy the *Eikonal equation*:

$$||\nabla_{\mathbf{P}} f(\mathbf{P})||_2 = 1, \quad \forall \mathbf{P} \in \mathbb{R}^3. \tag{8}$$

  Proof of Eikonal Regularization

  Let $\mathbf{P} \in \mathbb{R}^3$ and $\mathbf{u}$ a unit vector, $||\mathbf{u}||_2 = 1$ and $|\sigma| \ll 1$. By first-order Taylor expansion:

$$f(\mathbf{P} + \sigma \mathbf{u}) \approx f(\mathbf{P}) + \sigma \nabla_{\mathbf{P}} f(\mathbf{P}) \cdot \mathbf{u}.$$

  Let θ be the angle between $\nabla_{\mathbf{P}} f(\mathbf{P})$ and $\mathbf{u}$. Then

$$\Delta f = f(\mathbf{P} + \sigma \mathbf{u}) - f(\mathbf{P}) \approx \sigma \nabla_{\mathbf{P}} f(\mathbf{P}) \cdot \mathbf{u} \approx \sigma \, ||\nabla_{\mathbf{P}} f(\mathbf{P})||_2 \cos\theta.$$

  For f to be a valid signed distance function, the change along any direction must equal the displacement:

$$\Delta f = \sigma \cos\theta \quad \Longrightarrow \quad \boxed{||\nabla_{\mathbf{P}} f(\mathbf{P})||_2 = 1}.$$

- **Geometric Stability**: Deviations from $||\nabla_{\mathbf{P}} f(\mathbf{P})||_2 = 1$ indicate distortion:

$$||\nabla_{\mathbf{P}} f(\mathbf{P})||_2 \begin{cases} > 1 & \text{implies local stretching (distance overestimation),} \\ < 1 & \text{implies local compression (distance underestimation).} \end{cases}$$

  Enforcing the Eikonal constraint regularizes the SDF, ensuring stable and consistent geometry during optimization.

4.2.2 Epoch 1: Neural SDF Initialization and MLP Design

The training begins by constructing a neural signed distance field (SDF), modeled by an eight-layer multi-layer perceptron (MLP) to implicitly represent 3D geometry from shadow cues.

- **Network Structure**: ShadowNeuS uses an 8-layer fully-connected MLP with ReLU activation functions, a single output head producing scalar signed distances $f(\mathbf{P}; \theta) \in \mathbb{R}$.

  Remark 4.2.2 (Why MLP Architecture?)**.**
  The MLP design offers several advantages:

  - **Universal Approximation**: By the universal approximation theorem, MLPs can approximate any continuous function on compact subsets of $\mathbb{R}^3$.
  - **End-to-End Differentiability**: Gradients $\nabla_{\mathbf{P}} f$, $\nabla_{\theta} f$ can be computed throughout the pipeline, enabling training directly from shadow observations.

- **Joint Integrative Learning**: Both surface positions ($f(\mathbf{P}) = 0$) and surface normals ($\nabla_{\mathbf{P}} f$) are jointly encoded in f, simplifying the learning process compared to classical pipelines which handle these separately.
- **Circular Dependency Resolution**: Neural SDF jointly models geometry and visibility, avoiding the iterative or circular inversion problems faced by classical shadow methods.

- **Input Encoding**: To enhance the ability of the MLP to capture high-frequency geometric features, ShadowNeuS applies a sinusoidal positional encoding $\gamma(\mathbf{P})$ to the input 3D coordinates:

$$\gamma(\mathbf{P}) = \left(\sin(2^0 \pi p_x), \cos(2^0 \pi p_x), \ldots, \sin(2^{k-1} \pi p_z), \cos(2^{k-1} \pi p_z)\right). \tag{9}$$

  Remark 4.2.3 (Why Positional Encoding?)**.**
  Raw coordinates $\mathbf{P} = (p_x, p_y, p_z)$ do not provide sufficient high-frequency variation for MLPs, leading to overly smooth approximations (spectral bias). Positional encoding introduces Fourier feature mappings [TSS20], allowing the network to represent both coarse and fine geometric structures revealed by shadows.

- **Output**: A scalar SDF value $f(\mathbf{P})$ per point.

- **Random Initialization**: The MLP weights θ are initialized using standard techniques such as Xavier initialization, ensuring well-scaled activations in early epochs.

4.2.3 Epoch 2→N: Ray Sampling and Shadow Rendering

From the second epoch onward, ShadowNeuS performs **ray sampling** to locate surface points and **shadow rendering** to estimate shadow values at image pixels. The process has two stages:

- **Camera Ray Sampling**:
  For each pixel $\mathbf{p}$, a camera ray is constructed:

$$\mathbf{r}(t) = \mathbf{C} + t\mathbf{d}, \quad \mathbf{d} = R^{-1} K^{-1} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix},$$

  where K and R are the intrinsic and extrinsic parameters of the camera. The intersection point $\mathbf{P}$ with the implicit surface is found by sphere tracing to locate t^* such that

$$f_\theta(\mathbf{r}(t^*)) \approx 0, \quad \mathbf{P} = \mathbf{r}(t^*).$$

- **Shadow Ray Sampling and Visibility Estimation**:
  For each $\mathbf{P}$, a light ray towards the light source $\mathbf{L}$ is defined:

$$\mathbf{l}(s) = \mathbf{P} + s(\mathbf{L} - \mathbf{P}), \quad s \in [0, 1].$$

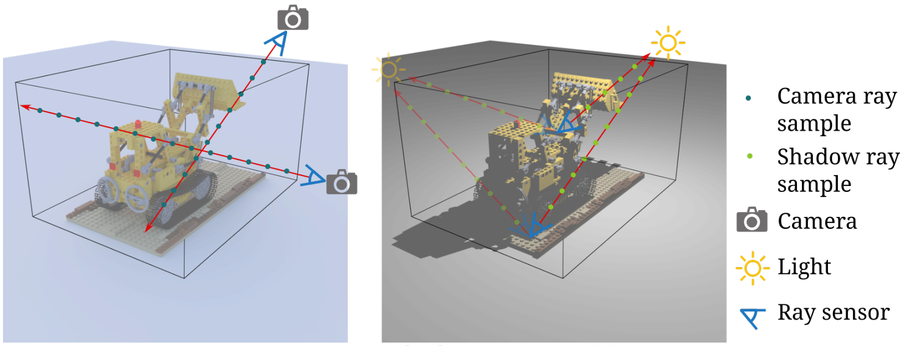
The **opacity** along $\mathbf{l}(s)$ is computed at discrete points $\mathbf{l}(s_j)$ using:

$$a_j = \max\left(1 - \frac{\phi(f_\theta(\mathbf{l}(s_{j+1})))}{\phi(f_\theta(\mathbf{l}(s_j)))}, 0\right)$$

where $\phi(\cdot)$ is an active function, and $a_j \in [0,1]$ represents the **local opacity** between adjacent samples.
The accumulated transmittance (incoming light intensity) is then:

$$C_{\text{in}}(\mathbf{P}) = \prod_{j=1}^{N}(1-a_j)$$

This multiplicative accumulation is analogous to volume rendering used in NeRF [MST20], where densities are integrated along rays to estimate visibility in a differentiable way.



Camera ray supervision (NeRF) Shadow ray supervision (Ours)

Figure 9. Comparision between Camera ray and Shadow ray [LWX23]

4.2.4 Convergence: Loss Functions

Training minimizes a weighted sum of several loss terms that enforce geometric regularity and appearance consistency:

$$\mathscr{L}_{\text{total}} = w_{\text{shadow}} \cdot \mathscr{L}_{\text{shadow}} + w_{\text{eik}} \cdot \mathscr{L}_{\text{eik}} + w_{\text{app}} \cdot \mathscr{L}_{\text{appearance}} + \dots$$

- **Shadow Supervision Loss:**

$$\mathscr{L}_{\text{shadow}} = \sum_{\mathbf{p}} |C_{\text{in}}(\mathbf{p}) - S(\mathbf{p})|$$

compares predicted light transmittance $C_{\text{in}}(\mathbf{p})$ with observed shadow labels $S(\mathbf{p})$.

- **Eikonal Loss:**

$$\mathscr{L}_{\text{eik}} = \frac{1}{M}\sum_{i=1}^{M}(\|\nabla_{\mathbf{P}} f_\theta(\mathbf{P}_i)\|_2 - 1)^2$$

encourages f_θ to approximate a signed distance function by enforcing unit-norm gradients almost everywhere, leading to stable and smooth surfaces.

- **Appearance or Consistency Loss:**

  $$\mathcal{L}_{\text{appearance}} = \sum_{\mathbf{p}} \left\| C_{\text{pred}}(\mathbf{p}) - C_{\text{gt}}(\mathbf{p}) \right\|_2^2$$

  enforces consistency between predicted and ground-truth colors (or features) at each pixel. Depending on the dataset, this may include texture or feature-based terms.

- **Other Loss Terms (optional):**
  When additional supervision is available, terms such as normal alignment, depth consistency, or regularization losses can be added.

4.2.5 Gradient Descent and Adam Optimizer

To minimize the total loss $\mathcal{L}_{\text{total}}(\theta)$, neural networks typically rely on gradient-based optimization methods. Below, we summarize both the basic gradient descent method and the Adam optimizer used in our implementation. Details of Adam optimizer are refered to paper [KB14].

- **Gradient Descent**: The simplest method updates parameters θ along the direction of steepest descent:

  $$\theta_{t+1} = \theta_t - \alpha \nabla_\theta \mathcal{L}_{\text{total}}(\theta_t),$$

  where $\alpha > 0$ is the learning rate. Here, $\nabla_\theta \mathcal{L}_{\text{total}}$ points in the direction of fastest increase of the loss, so subtracting it reduces the loss.

  Remark 4.2.4 (Limitations)**.**
  Basic gradient descent has several drawbacks:

  - **Learning Rate Sensitivity**: Too small α slows convergence; too large may cause divergence.
  - **Oscillation**: In steep or curved regions, updates can overshoot or oscillate.
  - **Uniform Step Size**: All parameters use the same α, ignoring variations in gradient magnitude.

- **Adam Optimizer**: Adam (*Adaptive Moment Estimation*) improves convergence by combining momentum and adaptive learning rates:

  - **Momentum (First Moment)**: Maintains an exponential moving average of past gradients:

    $$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \quad g_t = \nabla_\theta \mathcal{L}_{\text{total}}(\theta_t),$$

    reducing oscillations and smoothing updates.
  - **Adaptive Scaling (Second Moment)**: Tracks the moving average of squared gradients:

    $$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2,$$

    allowing per-parameter learning rate adjustments based on gradient magnitude.

  - **Bias Correction**: Corrects for initialization bias ($m_0, v_0 = 0$):

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}.$$

- **Parameter Update**: The final Adam update combines these elements:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon},$$

  where typical values are:

$$\alpha = 10^{-3}, \quad \beta_1 = 0.9, \quad \beta_2 = 0.999, \quad \varepsilon = 10^{-8}.$$

  This update ensures stable, efficient learning by smoothing updates (momentum) and scaling steps according to gradient history (adaptive rate).

4.3 Summary

After analyzing and exploring the ShadowNeuS pipeline, we want to validate our understanding through experiments, following the scientific principle of repeating phenomena to test hypotheses. While full 3D reconstruction is computationally intensive and challenging, we propose to reduce the dimensionality and perform a simplified reconstruction—from 1D shadows to 2D geometry—to gain insight into the underlying mechanisms.

5 Experimental Setup: 1D-to-2D Shadow Reconstruction

5.1 Scene Definition

We consider a simplified 2D environment designed as a toy model of the shadow-based reconstruction problem. The setting is defined as follows:

- **Space:** A two-dimensional plane (x, y) with a horizontal baseline (ground) onto which shadows are projected.
- **Object:** A simple, convex, enclosed 2D shape, fixed and static, that serves as the occluder.
- **Light source:** A single fixed point light, positioned differently for each measurement, generating a cast shadow of the object.
- **Camera:** A single-view sensor that records the shadow along the ground as a one-dimensional binary/intensity array, producing barcode-like measurements.

5.2 Goal and Hypothesis

The experiment aims to test whether shadow-based cues can enable shape recovery even in reduced dimensionality.

- **Goal:** To reconstruct the 2D shape of the object from only its 1D shadow projections.
- **Hypothesis:** Despite the dimensional reduction, approximate shape recovery is possible by enforcing geometric consistency between the 2D object and its observed 1D shadows, mirroring the principle exploited in ShadowNeuS.

5.3 Neural Model

6 Discussion and Analysis

7 Conclusion

References

[LWX23] Jingwang Ling, Zhibo Wang, Feng Xu. *ShadowNeuS: Neural SDF Reconstruction by Shadow Ray Supervision*. arXiv: 2211.14086, 2023.

[VK23] Venkatkumar. *3D Reconstruction Basic Terminology (Traditional Computer Vision Approach)*. Medium: URL, 2023.

[**Figure 1.**] Sjoerd van Riel. *Exploring the use of 3D GIS as an analytical tool in archaeological excavation practice*. ResearchGate: URL, 2016.

[**Figure 2.**] Catree. Camera coordinate to pixel coordinate - OpenCV, 2016.

[**Figure 3.**] Lemanoosh. Blender material rendering, n.d.

[**Figure 4.**] Michael Abramov. Semantic Segmentation vs Object Detection: Understanding the Differences, 2024.

[**Figure 5.**] Sandip Neogi. shadow of object Pro Vector, n.d.

[**SHFP01**] S. Savarese, H. Rushmeier, F. Bernardini, P. Perona. *Shadow Carving* IEEE:10.1109/ICCV.2001.937517, 2001.

[**M51**] Gaspard Monge. *An Elementary Treatise on Descriptive Geometry, with a Theory of Shadows and of Perspective Extracted from the French of G. Monge by J. F. Heather* Google book id: 8KZ0SKUCVXQC, 1851.

[**PFS19**] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, Steven Lovegrove. *DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation*. arXiv: 1901.05103, 2019.

[**TSS20**] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, Ren Ng. *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*. arXiv: 2006.10739, 2020.

[**MST20**] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng. *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. arXiv: 2003.08934, 2020.

[**KB14**] Diederik P. Kingma, Jimmy Ba. *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980, 2014.