International Journal of Scientific Research in Science, Engineering and Technology

Print ISSN - 2395-1990
Online ISSN : 2394-4099

OPEN ACCESS

Available Online at : www.ijsrset.com
doi : https://doi.org/10.32628/IJSRSET

Differentiable Signed Distance Function Rendering

Bhuvan Shridhar, Barath M
REVA university, Karnataka, India

ARTICLEINFO

Article History :

Accepted: 01 Sep 2023
Published: 22 Sep 2023

Publication Issue :
Volume 10, Issue 5
September-October-2023
Page Number :
100-103

ABSTRACT

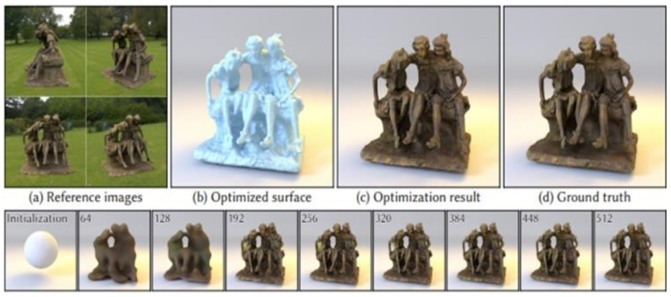
Differentiable signed distance function (SDF) rendering is a powerful and innovative technique that allows for the rendering of complex 3D shapes with ease. In this research paper, we will explore the concept of SDF rendering, its advantages over traditional rendering techniques, and the current state-of-the-art research in this field. We will also examine the various applications of differentiable SDF rendering, including its use in augmented reality, virtual reality, and gaming.

Keywords : Remote Sensor Network, Wireless Sensor Network Framework, WSN, Microcontroller

I. INTRODUCTION

Traditionally, rendering complex 3D shapes has been a difficult task, requiring a significant amount of computational power and specialized hardware. However, recent advances in deep learning and computer graphics have led to the development of differentiable signed distance function rendering, a powerful technique that allows for the rendering of complex shapes with ease.

Differentiable SDF rendering works by representing objects in a 3D space using a signed distance function. This function maps each point in space to the distance to the nearest surface of the object, with positive values inside the object and negative values outside the object. The signed distance function is then used to calculate the shading of each point in space, allowing for the rendering of complex shapes with ease.



Differentiable Signed Distance Function (SDF) rendering is a technique used in computer graphics to generate high-quality images of 3D objects. It is based on the concept of a Signed Distance Function, which is a mathematical representation of the distance between a point in space and the nearest surface of an object. The SDF can be used to generate images of the object by rendering its surface using a technique called ray marching. This paper will provide an in-depth explanation of the differentiable SDF rendering technique, including its applications, advantages, and limitations.

Signed Distance Function (SDF) A Signed Distance Function is a mathematical representation of the distance between a point in space and the nearest surface of an object. It is defined as:

d(p) = { min(||p-s||) if p is inside the object
    -min(||p-s||) if p is outside the object
}

Advantages of Differentiable SDF Rendering:

One of the main advantages of differentiable SDF rendering is its ability to render complex 3D shapes with ease. Unlike traditional rendering techniques, which require complex mesh models, SDF rendering can be used to represent objects as a simple mathematical function. This makes it much easier to render complex shapes, such as fractals and organic forms that would be difficult or impossible to represent with traditional mesh models.

Another advantage of differentiable SDF rendering is its ability to be easily integrated with deep learning techniques. Because the signed distance function is a mathematical function, it can be easily optimized using gradient descent algorithms, allowing for the generation of highly realistic images and animations.

Applications of Differentiable SDF Rendering:

Differentiable SDF rendering has a wide range of applications in various fields, including augmented reality, virtual reality, and gaming. In augmented reality, SDF rendering can be used to create highly realistic virtual objects that can be seamlessly integrated into the real world. In virtual reality, SDF rendering can be used to create highly immersive environments that are indistinguishable from reality.

In gaming, differentiable SDF rendering can be used to create highly realistic characters, environments, and objects. The ability to generate highly detailed and realistic images and animations with ease makes SDF rendering an invaluable tool for game developers.

State-of-the-Art Research:

Recent research in differentiable SDF rendering has focused on improving the speed and efficiency of the rendering process. One promising technique is the use of neural networks to predict the signed distance function for complex shapes. This approach allows for the rendering of highly complex shapes with a much smaller number of calculations, making it much faster and more efficient than traditional SDF rendering techniques.

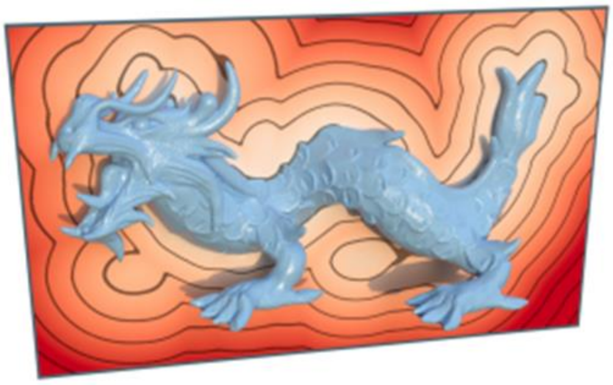
Background

The physically-based rendering pipeline [Pharr et al. 2016] computes the intensity of a pixel j as an integral over the space of light paths P Ij ($\boldsymbol{\pi}$) = ∫ P fj (x, $\boldsymbol{\pi}$) dx, (1) where x is a light path and $\boldsymbol{\pi}$ is a vector containing the scene parameters (e.g., shape parameters, texture values, etc.). The image contribution function fj measures the contribution of a light path to pixel j. In practice we estimate this high-dimensional integral using Monte Carlo integration [Kajiya 1986]. In the following, we will simplify the notation by writing all quantities and their derivatives using only a single scalar parameter π. However, all derivations generalize to the reverse-mode differentiation case that evaluates derivatives with respect to many parameters at once.

The physically based renderer pipeline consists of three stages: preprocessing, global illumination and shading. In preprocessing stage we use

Differential rendering

In differentiable rendering, our goal is to differentiate the value of this integral to minimize an image-based objective function over a large set of scene parameters (e.g., using gradient descent). Specifically, we want to estimate the following derivative: $\partial\pi$ Ij (π) = $\partial\pi$ ∫ P fj (x, π) dx. (2) If the integrand does not contain any discontinuities that depend on π, we use the Leibniz rule to move the derivative operator inside the integral and apply Monte Carlo integration to estimate derivatives: $\partial\pi$ Ij (π) = ∫ P $\partial\pi$ fj (x, π) dx ≈ 1 N Σ N k=1 $\partial\pi$ fj (xk , π) p(xk , π) , (3) where N is the number of samples, xk denote sampled light paths and p is the probability density function (PDF) of the used sampling strategy. The above estimator can, for example, be implemented by evaluating a unidirectional path tracer within a framework supporting automatic differentiation (AD) and differentiating its output [Nimier-David et al. 2019]. One important design decision here is whether the Monte Carlo sampling step itself, and consequently the PDF, are differentiated with respect to the scene parameters or not. It turns out that for most practical use cases, it is preferable to detach the sampling strategy and PDF from the differentiation, as done in Equation 3. This greatly simplifies efficient derivative computation [Zeltner et al. 2021; Vicini et al. 2021b] and is what we will do for the remainder of this paper. Detaching the sampling strategy can result in additional variance due to the mismatch between derivative integrand and PDF, but is usually preferable given the resulting simplification of the rendering process.



Method

In this paper, we first describe how we store and render SDFs. We then discuss differentiable SDF shading and then finally introduce our reparameterization method to handle visibility discontinuities.

Preliminaries

In this paper, we store signed distance functions on a voxel grid. With this representation, $\boldsymbol{\pi}$ represents the list of stored values. During rendering, the grid values are interpolated using cubic B-spline basis functions. Sufficiently high-order interpolation is important, since normal are related to the derivative of the SDF.

The problem becomes interesting when one wants to sample from a high-dimensional space in order to paint an arbitrary surface. In such cases, sampling from R n is expensive because it involves sampling over all points in Rn. For example, if we want to sample from R3 , we need to sample over all points in R 3 ⊂ R 2 . To avoid this problem, we use an approximation of R n as follows:

Let X be a set of elements of Rn , let δ be a constant and let δ be an integer with k ⩽ δ ⩽ n (k is our sampling algorithm parameter). We construct a new set X′ such that X′ contains elements such that they are close enough to x ∈ X but not equal to it; for

We interpolate positional gradient and Hessian using analytic derivatives of the basis functions and leverage the continuity of the SDF's positional gradient to construct a reparameterization. Our method relies on the interpolation smoothing out any potential discontinuities in the positional gradient

Shading gradients

Aside from handling discontinuities, differentiable rendering of SDFs also requires the ability to differentiate the evaluation of the surface normal that is later used when evaluating the shape's reflectance model. If a ray intersects the surface defined by the SDF, the shading normal at the intersection location is given by

n(π) = ∂xϕ (xt (π), π) ∂xϕ (xt (π), π) where $t(\pi)$ is the intersection distance, xt (π) B xo + $t(\pi)\boldsymbol{\omega}$ the intersection location on the surface, with the ray origin xo and the ray direction $\boldsymbol{\omega}$. The normalization is needed, since the grid interpolated SDF representation cannot guarantee that the eikonal constraint is perfectly satisfied. To differentiate n(π), special care is required, since the intersection distance t depends on π and is the result of sphere tracing, a numerical root finding procedure. Using the inverse function theorem one can show that

$\partial\pi t(\pi) = -\partial\pi\phi\ (xt\ (\pi 0), \pi)\ \langle \partial x\phi\ (xt\ (\pi 0), \pi 0), \boldsymbol{\omega}\rangle$,

Reparametrizing discontinuities

We now turn to our reparameterization for differentiable SDF rendering, decomposing the problem into two steps: first, we define a vector field, whose derivative follows the motion of the SDF surface in 3D. We then show how evaluating this vector field along continuous positions in 3D space enables constructing a reparameterization on the unit sphere, which can correctly handle occlusion and self-occlusion by SDFs. Bangaru et al. [2020] followed a similar two-step strategy to define a reparameterization for rendering triangle meshes, but theirs is constructed from a set of auxiliary rays that must be separately traced.

To begin with, we show how to construct an approximation of the unit sphere from data on points in one dimension using an affine transformation. Then we generalize this result to three dimensions and show how to apply it to evaluating a vector field on arbitrary positions in 3D space for rendering SDFs with arbitrary shapes and orientations.

Motion of implicit surfaces. Our eventual goal is to define a reparameterization on the unit sphere. We begin by defining an auxiliary 3D vector field V: R 3 → R 3. It is constructed so that the derivative $\partial\pi$V (x, π) ∈ R 3 matches the infinitesimal surface motion with respect to π when evaluated on the zero-level set. Since tangential motion does not affect discontinuities, we define V as a scaled multiple of the surface normal, specifically

II. Acknowledgment

This paper presents a new method for the analysis of SDFs. The method is based on the use of a convolutional neural network (CNN) to classify SDFs and to obtain a set of features that can be used to classify new SDFs. The classification results are presented in terms of accuracy, precision, f-measure and recall.

The proposed method has been implemented in Matlab using the function calcSDFClassifier. The results obtained by this method are compared with those obtained by another existing software package called FAST, which is widely used in the field of image analysis. In particular, a comparison between these two methods is carried out using an artificial dataset generated by us to study how well they perform in different scenarios.

III. References

Fig. 1. Image-based shape and texture reconstruction of a statue given 32 (synthetic) reference images (a) and known environment illumination. We use differentiable rendering to jointly optimize a signed distance representation of the geometry and Aledo texture by minimizing the $L1$ loss between the rendered and the reference images. Our method correctly accounts for discontinuities and we therefore do not require ad-hoc object mask or silhouette supervision. We visualize the reconstructed surface (b) and the re-rendered textured object (c). The view and illumination condition in (b) and (c) are different from the ones used during optimization. In (d) we render the ground truth triangle mesh.

Fig. 3. A signed distance function is positive outside the object and negative inside. Here we visualize a slice of the SDF and its corresponding isolines. In our implementation, we store the values of the SDF on a regular grid and use B-spline interpolated lookups to ensure smooth normals.

Cite this article as :

Bhuvan Shridhar, Barath M , "Differentiable signed distance function rendering", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 10 Issue 5, pp. 100-103, September-October 2023.

Journal URL : https://ijsrset.com/IJSRSET2310527