# 曲線近傍を移動する視点から見えるステレオ自由視点画像の高速生成
# Fast Synthesis of Stereo Novel Views Observed from a Moving Range near Curve

小池 優太郎 *
Yutaro Koike
法政大学 情報科学部 ディジタルメディア学科
Email: yutaro.koike.9v@stu.hosei.ac.jp

***Abstract*—This paper presents a system for fast synthesis of stereo novel views seen from a viewpoint moving near a curve. To achieve this, we split the scene into tube-like segments and distributed data to multiple neural networks for training and synthesis. Previous studies have presented two notable methods. One can handle a wide area but generates images at slower than 0.2 fps, and the other is faster but does not handle a wide area. This paper introduces three methods: scene decomposition, post-rendering scaling, and foveation to generate stereo free-viewpoint images faster while extending the range of possible movement.**

## 1. はじめに

Research & Development Group, Hitachi, Ltd. が定める自由視点の定義は「三次元画像または映像において，画像または映像を撮影したカメラそのものが提供する視点ではなく，ユーザが自由に選択できる視点のこと」である [1]．自由視点画像は自由視点から見た画像である．自由視点画像を生成する技術は，ドローンが撮影するような映像を撮影現場に居なくとも生成できる技術である．

広範囲なシーン内を自由視点画像で見ることができれば，広大な公園内や巨大な建物内などをスムーズに散策する体験をディスプレイを通じて提供できる．しかし問題として，データサイズの増加により自由視点画像生成システム速度が低下することが予測される．そのため本研究は視点移動可能範囲を広げつつも，より高速に自由視点画像を生成するシステムを開発することを目指す．

## 2. 関連研究

Mildenhall らは少ない入力画像から複雑かつ連続的なシーンを合成し，高品質な自由視点画像を得る手法 NeRF を提案している [2]．彼らはラディアンスを持つ点が無数に存在する空間，Radiance Fields を関数 (1) で表現している．関数 (1) は，点の 3 次元座標 $\boldsymbol{x} = (x, y, z)$ と 2 次元視点方向 $\boldsymbol{d} = (\theta, \phi)$ を入力すると，その点におけるラディアンスの RGB 色 $\boldsymbol{c} = (r, g, b)$ とボリューム密度 $\sigma$ を出力する．彼らは関数 (1) を実現するために多層パーセプトロン (以下「MLP」という) を用いている．

$$F : (\boldsymbol{x}, \boldsymbol{d}) \rightarrow (\boldsymbol{c}, \sigma). \tag{1}$$

Mildenhall らが提示したデモ画像は小物や室内などいずれも狭い範囲を写したものである．他方で Tancik らは町通りの風景を自由視点画像で閲覧可能にする研究をおこなっている [3]．シーンを範囲ごとに分割し各範囲を別々の MLP に学習させることで，各 MLP が保持する平均データサイズを軽減している．しかしそこで提案された手法が自由視点画像を生成する速度は 0.2 fps 未満である．

Mildenhall らが提案した手法をステレオ画像に拡張し，ステレオ画像の性質を利用して高速化した研究もある．Deng らは NeRF を改良する形で，ステレオ自由視点画像の見え方を利用して高速化を試みている [4]．Deng らは解像度 1,440 [px] × 1,600 [px], 視野角 110 度の HMD 上に自由視点画像を 50 fps で生成する手法『FoV-NeRF』を提案した．両眼視差のずれを軽減したり，注視点から離れた部分の解像度を抑えたりなど，HMD 等のステレオディスプレイに特化した品質向上の工夫が施されている．一方でこの手法は固定された点 1 つを原点とする座標系を用いており，その点から離れた視点からの画像生成に非対応，すなわち歩くほどの距離移動ができない問題がある．

本研究は Tancik らの広範囲自由視点画像技術と FoV-NeRF の高速なステレオ自由視点画像技術を組み合わせ，実用的な速度で自由視点画像を生成するシステムを開発することを目指す．具体的には 10 m 以上伸びた曲線上を移動する視点から見たステレオ自由視点画像を 1 fps 以上で生成する．

## 3. 提案手法

本研究は 2 つの関連研究を参考に，曲線上の視点から見たステレオ自由視点画像を，高速に生成する手法を提案する．1 つ目の関連研究である Tancik らの研究は節 3.2 にて活用し，もう 1 つの関連研究 FoV-NeRF は節 3.5 にて活用する．本研究の特徴は視点移動範囲を「曲線上」に制限することである．視点位置を空間上に存在する任意の 3 次元点ではなく曲線上に制限する理由は 2 つある．

1 つ目は自由視点画像生成用の素材を撮影することが容易になるためである．視点位置を空間上に存在する任意の 3 次元点にすると，様々な角度から撮った写真が必要である．一方で視点位置を曲線上に制限すると，その曲線付近からの撮影で十分であり，より少ない写真からでも高品質な画像を生成できると考える．

2 つ目はより自然な使用用途に合っているためである．人間は歩行や自動車などで移動するため，多くのツアーやテーマパークライドなどのように，一定のコースに沿って視点が変化することが自然である．したがって，視点移動を曲線上の移動に制限しても自然な使用用途には問題ないと考えるためである．

### 3.1. ニューラルボリュームレンダリング

ボリュームレンダリングは物体を密度で表現することで，ポリゴンでは表されない雲や粒子システム等をレ

* Supervisor: Prof. Takafumi Koike

ンダリングする技術である [5]．シーンの表現には 3 次元点座標を入力するとその地点の色とボリューム密度を返す関数を用いる．その関数は全結合な MLP で実装する．提案手法はその色とボリューム密度を基にボリュームレンダリングをおこなうことで新規視点画像を得る．

ボリューム密度 $\sigma(\boldsymbol{x})$, 色情報 $\boldsymbol{c}(\boldsymbol{x})$ が既知の時，レイ $\boldsymbol{r}(t) = \boldsymbol{O} + t\boldsymbol{d}$ に対応する色 $\boldsymbol{C}(\boldsymbol{r})$ は式 (2) で求められる．なお $t_n, t_f (t_n \leq t_f)$ は限界を表す実数であり，本研究では $t_n = 0, t_f = 1$ とした．コンピュータ上では式 (2) を離散的に求めており，その計算に用いるサンプリング点の個数 (式 (2) の $N$) をレイサンプリング数と呼ぶ．一般にレイサンプリング数が高いほど PSNR 値が高くなる．なお式 (2) の $T(\boldsymbol{r}, t)$ はレイが $[t_n, t]$ 区間内で通過するボリューム密度が総じて低いという確率である．レイが $[t_n, t]$ 区間内で高ボリューム密度点を通過すると，$T(\boldsymbol{r}, t)$ が増加し， $\boldsymbol{C}(\boldsymbol{r})$ が色濃くなるという仕組みである．

$$
\begin{aligned}
t_i &= \frac{i}{N-1}(t_f - t_n) + t_n, \\
\boldsymbol{C}(\boldsymbol{r}) &= \sum_{i=0}^{N-1} T(\boldsymbol{r}, i)\sigma(\boldsymbol{r}(t_i))\boldsymbol{c}(\boldsymbol{r}(t_i))dt, \\
T(\boldsymbol{r}, i) &= \exp\left(-\sum_{j=0}^{i} \sigma(\boldsymbol{r}(t_j))ds\right).
\end{aligned}
\tag{2}
$$

本研究で用いる MLP は Tiny CUDA Neural Network Framework (以下「TinyCudaNN」という) である．TinyCudaNN は CUDA GPU 内のレジスタや Shared Memory などの高速なメモリを活用することで， Tensorflow より高速に動作する Neural Network を提供する [6].

### 3.2. シーンデコンポジション

提案手法はシーンを図 1 に示されるようなチューブ状の範囲に区切る．この工夫をシーンデコンポジションと呼ぶ．チューブ状の範囲をセグメントと呼び，セグメントを定める軸となる曲線をセグメント基軸と呼ぶ．1 セグメントごとに 1 つの多層パーセプトロンを対応させることで，並列学習を可能にする．$u$ を 0 以上 1 未満の実数，曲線を $\boldsymbol{f}(u)$ としたとき，ある 3 次元座標 $\boldsymbol{x}$ がどのセグメントに属しているかは $||\boldsymbol{x} - \boldsymbol{f}(u)||$ を最小にする $u$ を求めることで判定可能である．本研究では軸となる曲線は非一様 2 次の B スプライン曲線を採用する．

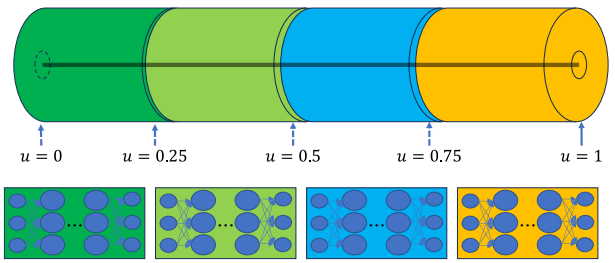


図 1. セグメント基軸と 4 つのセグメント

$M$ 個の実視点位置の 3 次元直交座標を $\boldsymbol{V}_0, \boldsymbol{V}_1, ..., \boldsymbol{V}_{M-1}$ と置き，全点の間を通過する曲線を求める．$\boldsymbol{V}_0, \boldsymbol{V}_1, ..., \boldsymbol{V}_{M-1}$ から 3 次元制御点 $\boldsymbol{C}_0, \boldsymbol{C}_1, ..., \boldsymbol{C}_{N-1}$ を得るために Jing らの論文に掲載された「最小二乗法を用いた B スプライン曲線フィッティング」の方法を導入する [7]．この方法は $\sum_{i=0}^{M-1} ||\boldsymbol{V}_i - \boldsymbol{f}(t_i)||$ が最小になるような $\boldsymbol{V}_i$ を得る方法である．$t_i$ は $\boldsymbol{V}_i$ の地点を示す値で，当アルゴリズムは $\boldsymbol{f}(u_i) = \boldsymbol{V}_i$ に近づくようフィッティングする．今回は単純化のため $u_i = \frac{i}{M-1}$ とする．このとき制御点は方程式 (4) を解くことで得られる．方程式 (4) にある行列 $A$ は式 (3) により定義される．なおノット列を始める最初の 3 値が 0，最後の 3 値が 1 と揃っているため，$\boldsymbol{C}_0 = \boldsymbol{V}_0, \boldsymbol{C}_{N-1} = \boldsymbol{V}_{M-1}$ である．また数式の記述範囲を節約するため $\boldsymbol{C}_i$ は行ベクトルとし，$\boldsymbol{q}_i = \boldsymbol{C}_i - \boldsymbol{C}_0 B_0^2\left(\frac{i}{M-1}\right) - \boldsymbol{C}_{N-1} B_{N-1}^2\left(\frac{i}{M-1}\right)$ と定義する．

$$
A = \begin{pmatrix}
B_1^2\left(\frac{1}{M-1}\right) & B_2^2\left(\frac{1}{M-1}\right) & \cdots & B_{N-2}^2\left(\frac{1}{M-1}\right) \\
B_1^2\left(\frac{2}{M-1}\right) & B_2^2\left(\frac{2}{M-1}\right) & \cdots & B_{N-2}^2\left(\frac{2}{M-1}\right) \\
\vdots & & & \\
B_1^2\left(\frac{M-2}{M-1}\right) & B_2^2\left(\frac{M-2}{M-1}\right) & \cdots & B_{N-2}^2\left(\frac{M-2}{M-1}\right)
\end{pmatrix}.
\tag{3}
$$

$$
A^T A \begin{pmatrix} \boldsymbol{C}_1 \\ \boldsymbol{C}_2 \\ \vdots \\ \boldsymbol{C}_{N-1} \end{pmatrix} = \begin{pmatrix}
\sum_{i=1}^{N-2} B_1^2\left(\frac{1}{M-1}\right)\boldsymbol{q}_1 \\
\sum_{i=1}^{N-2} B_2^2\left(\frac{2}{M-1}\right)\boldsymbol{q}_2 \\
\vdots \\
\sum_{i=1}^{N-2} B_{N-1}^2\left(\frac{M-2}{M-1}\right)\boldsymbol{q}_{N-1}
\end{pmatrix}.
\tag{4}
$$

### 3.3. 高速なセグメント特定

ある 3 次元座標の点がどのセグメントに属するかを特定することをセグメント特定と呼ぶ．本研究では 1 枚の自由視点画像を生成するために新規視点原点 $(x, y, z)$ に関してセグメント特定する．高速なセグメント特定を実現するためにキャッシングと kd 木を用いたアルゴリズムを導入し，$(x, y, z)$ から $\hat{t}$ を求める処理を高速化する．このアルゴリズムは以下のステップに沿って $\boldsymbol{x} = (x, y, z)$ から $\hat{t}$ を求める．このアルゴリズムは並列化可能であるため，並列分散処理による更なる高速化も可能である．

前準備 1 : 曲線 $\boldsymbol{f}(u)$ 上にある $N$ 個の点をサンプリングしキャッシュする．$\boldsymbol{P}_i = \boldsymbol{f}(\frac{i}{N-1})(0 \leq i \leq N-1)$

前準備 2 : kd 木に $\boldsymbol{P}_i (i = 0, 1, ..., N-1)$ を登録する．

1) kd 木にクエリを投げ，$\boldsymbol{P}_i$ 中から $||\boldsymbol{x} - \boldsymbol{P}_i||$ が最も小さくなる 5 点 $\boldsymbol{P}_a, \boldsymbol{P}_b, ..., \boldsymbol{P}_e$ を求める．なお，$a, b, c, d, e$ は互いに異なる．
2) $u_a = a + \frac{1}{N}\frac{(\boldsymbol{P_{a+1}} - \boldsymbol{P_a})\cdot\boldsymbol{x}}{||\boldsymbol{P_{a+1}} - \boldsymbol{P_a}||^2}$ とする．
3) 定義されている $\boldsymbol{f}(u_a), \boldsymbol{f}(u_b), \boldsymbol{f}(u_c), \boldsymbol{f}(u_d), \boldsymbol{f}(u_e)$ の内，$\boldsymbol{x}$ に最も近い点の引数を $u'$ と置く．例えば $\boldsymbol{f}(u_a)$ が定義されていて $(\boldsymbol{x}, \boldsymbol{f}(u_a))$ 間の距離が最も近い場合は $\hat{u} = u_a$ とする．
4) $\boldsymbol{x}$ とユークリッド距離的に最も近い $\boldsymbol{f}(u)$ 上にある点の座標は $\boldsymbol{f}(\hat{u})$ である．

### 3.4. レンダリング後拡大

MLP は膨大な時間計算量を必要とする可能性があるため，MLP への依存度を軽減する高速化手法「レンダリング後拡大」を導入する．レンダリング後拡大は目標となる画像を縮小し生成する．次にバイリニア補間付き拡大をおこない生成画像を得る手法である．本研究では縦に 1/2 倍，横に 1/2 倍まで縮小することで，ニューラルネットワークへの入力を 1/4 倍まで縮小する．

### 3.5. ステレオ映像の生成

自由視点画像を生成するために用いている画像データには，撮影したカメラのポーズ (原点の位置情報と回転情報) が含まれている．一つの画像には一つのポーズが付随しており，ポーズは原点を表す 3 次元縦ベクトル $\boldsymbol{O}$ と回転を表す 3 次正方行列 $R$ で表される．ここでカメラの原点からその画像の画素 $(i, j)$ へ延びるレイ $\boldsymbol{r}$ は式 (5) で表される．

$$\boldsymbol{r} = \boldsymbol{O} + t\left(j - \frac{W}{2},\quad -\left(i - \frac{H}{2}\right),\quad -1\right) R. \qquad (5)$$

FoV-NeRF を参考に注視点から離れた部分のレイサンプリング数を抑えて，高速化を図る．FoV-NeRF では画像中心を注視点とし，視野角 20 度以内，45 度以内とその他の範囲で区切っていた．画像中心から画像端までの最短距離を $r$ [px] とおく．本研究では生成画像の中心から半径 $r\tan 20^\circ = 0.364r$ [px] の円の中に入る画素に高いレイサンプリング数を設定する．同様に生成画像の中心から半径 $r\tan 45^\circ = 0.5r$ [px] の円の中に入る画素に中程度のを，その他の画素には低いレイサンプリング数を設定する．この注視点から離れた部分のレイサンプリング数を抑える手法を foveation と呼ぶ．

## 4. 実験

本手法が有効であることを示すために実験を 3 つおこなった．1 つ目の実験ではセグメント配置が動作することを確認した．2 つ目の実験では 3 つの手法 (セグメンテーション，レンダリング後拡大，foveation) が生成速度と品質に与える影響を測定した．3 つ目はステレオ自由視点画像を生成できることを確認した．用いたデータは Tancik らの研究にて使われた Mission Bay データセットである．このデータセットにはカリフォルニア州ミッションベイで撮影された約 12,000 枚の写真とカメラ情報 (位置・向き・内部パラメータなど) が含まれている．

実験 1 では Mission Bay データセットに含まれていた撮影地点を全てプロットし，全点に沿うようなセグメント基軸を描画した．セグメント基軸は節 3.2 にて述べたアルゴリズムで作成した．結果を図 2 と図 3 に示した．図 2 ではセグメント基軸が全体的にカメラ位置に沿っていることが確認できる．図 2 の一部を拡大した図 3 では曲がりうねった箇所にも沿っていることが確認できる．実験 2,3 で用いるデータはこのうち 0.4 から 0.5 まで延びるセグメント内 (図 2 における上から 3 番目の赤線周辺) にある実視点画像・位置・向きである．

実験 2 では 3 つの手法 (セグメンテーション，レンダリング後拡大，foveation) の効果を測定した．3 つの手法を組み合わせて，生成速度と結果画像品質がどのように変化するかを計測した (表 1)．簡単のため，表 1 ではそれぞれの手法を Se, Sc, Fov と略す．結果画像品質は PSNR, SSIM, LPIPS で正解画像との画像類似度を求めることで計測した [8].

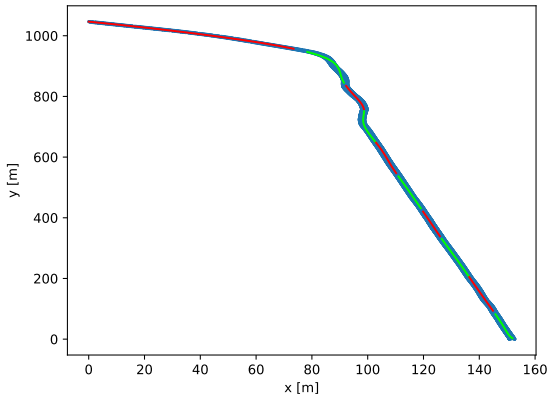


図 2. カメラ位置 (青点) とセグメント基軸

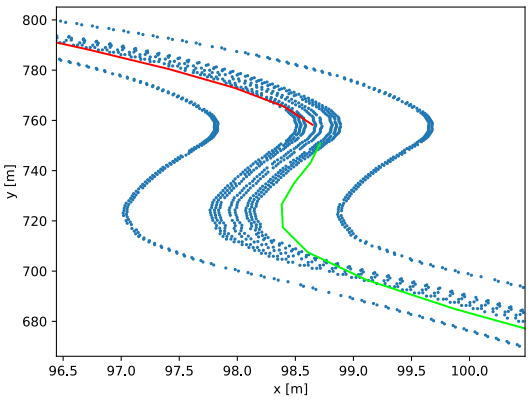


図 3. [拡大版] カメラ位置 (青点) とセグメント基軸

セグメンテーション無し実験では 1,125 枚分の実視点画像・位置・向きのデータを 1 つの MLP に与えた．学習イテレーション数は 10,000 である．対してセグメンテーション付き実験では 1,125 枚分のデータを 10 つの セグメントに分割し各セグメントに対応した MLP に与えた．すなわち 10 つの MLP に分散して学習させた．学習イテレーション数は 1 つの MLP につき 1,000 である．

Foveation 付き生成結果画像 (図 7) では生成画像の部分ごとにレイサンプリング数を変えた．画像中心から半径 226 [px] 内は 512，半径 311 [px] 内は 256，その他の箇所は 128 にレイサンプリング数を設定した．

実験 3 ではステレオ自由視点画像を生成した．図 8 はインタラクティブなステレオ自由視点画像生成システムの動作風景である．左右間隔 5 [m] のステレオカメ

表 1. 生成速度と結果画像品質

| 適応手法 | None(図 5) | Se(図 6) | Se+Sc(図 6) | Se+Fov(図 7) |
|---|---|---|---|---|
| fps | 0.422 | 0.422 | 1.608 | 0.981 |
| PSNR | 18.92 | 21.07 | 21.04 | 20.94 |
| SSIM | 0.9296 | 0.9577 | 0.9573 | 0.9565 |
| LPIPS | 0.131 | 0.126 | 0.127 | 0.127 |
| PSNR 毎秒 | 7.984 | 8.891 | 33.832 | 20.542 |

ラを配置した．両画像とも解像度は 512 [px] 四方であり，両画像の生成には合計で 0.724 秒かかった．またセグメント間を移動する際に，セグメントごとでシーンの美麗さが変化することがあった．すなわちセグメント間で自由視点画像の品質が不安定であった．

図 4. 正解画像 (1096 [px] x 622 [px])

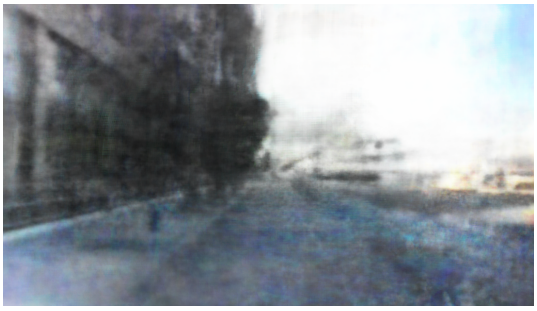

図 5. 生成結果画像 (セグメンテーションなし)

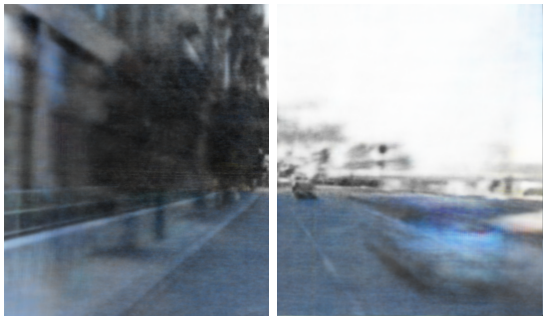

図 6. セグメンテーションあり生成結果画像 (左半分: レンダリング後拡大なし，レンダリング後拡大あり)

図 7. Foveation 付き生成結果画像

実験 4 では学習イテレーション数を増やすことが品質にどう影響するかを測定する．セグメンテーション付き実験の学習イテレーション数を 1 つの MLP あたり 5,000 まで増やし学習させた．次に学習させたモデルとセグメンテーション，レンダリング後拡大を用いて品質を評価した．結果として生成 fps は 1.815， PSNR は 21.790，SSIM は 0.9642，LPIPS は 0.113 を得た．


図 8. ステレオ自由視点画像

### 4.1. 考察

レンダリング後拡大は fps を 4 倍近く向上させた上に， PSNR の減少も 0.003 以内に収まった．速度が約 4 倍向上した原因は， MLP が処理するレイの個数が 1/4 倍になったためと考える．品質がさほど落ちなかった原因は，生成画像に元々高周波成分が入っていないぼやけた風景画であるためだと考える．

ステレオ自由視点画像生成システムを動作確認中に，セグメント間を移動すると品質が不安定になることがあった．考えられる理由の 1 つはセグメントごとに異なる実画像を学習しているためである．品質を安定させるためにはセグメント間に重なる地点を設ける，すなわちオーバーラップを設けることが有効だと考える．実際に Tancik らもオーバーラップはアーティファクトを防ぐためには重要であると述べている [3].

## 5. 結論

本研究ではシーンデコンポジションをおこなうためにセグメント基軸による区分をおこなった．加えて kd 木やレンダリング後拡大， TinyCudaNN などの高速化手法を適応したことで 1 fps を上回る生成速度を得た．

## 参考文献

[1] “Research & Development Group, Hitachi, Ltd. 用語集: 自由視点,” https://www.hitachi.co.jp/rd/glossary/jp_shi/ziyuushiten.html, (2024 年 02 月 09 日確認).

[2] B. Mildenhall, P. P. Srinivasan *et al.*, “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” *Communications of the ACM*, pp. 99–106, 2021.

[3] M. Tancik, V. Casser *et al.*, “Block-NeRF: Scalable Large Scene Neural View Synthesis,” *Proceedings of the IEEE/CVF CCVPR*, pp. 8248–8258, 2022.

[4] N. Deng, Z. He *et al.*, “FoV-NeRF: Foveated Neural Radiance Fields for Virtual Reality,” *IEEE Trans. VCG.*, pp. 3854–3864, 2022.

[5] J. T. Kajiya and B. P. Von Herzen, “Ray Tracing Volume Densities,” *Proceedings of the 11th Annual Conference on CGIT*, pp. 165—-174, 1984.

[6] T. Müller, F. Rousselle *et al.*, “Real-time Neural Radiance Caching for Path Tracing,” *ACM Trans. Graph.*, pp. 1–16, 2021.

[7] L. Jing and L. Sun, “Fitting B-Spline Curves by Least Squares Support Vector Machines,” *2005 International Conference on Neural Networks and Brain*, pp. 905–909, 2005.

[8] R. Zhang, P. Isola *et al.*, “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric,” *IEEE/CVF Conference CVPR*, pp. 586–595, 2018.