クラスタリングを用いたニューラル BTF の圧縮

Neural BTF Compression Using Clustering

岡田 実生 †　岩崎慶 †
Mio Okada　Kei Iwasaki

概 要

BTF (Bidirectional Texture Function，双方向テクスチャ関数) は，現実世界の素材に様々な方向から光を当て，様々な方向から光の反射率を測定したテクスチャデータであり，布などの複雑な外観を表現する際に広く用いられている．BTF は，照明の入射方向 (2 次元)，カメラへの反射方向 (2 次元)，表面上の位置 (2 次元) の 6 次元関数として表現される．BTF を用いることで，布のように複雑な外観を表現することが可能であるが，BTF はデータ量が膨大であるという欠点がある．

この欠点を解決するために，オートエンコーダから着想を得たニューラルネットワークを用いて BTF を圧縮する手法が提案された．しかしこの方法は照明の入射方向の次元，カメラへの反射方向の次元の圧縮であり，BTF のテクスチャ画像におけるテクセル間の類似性について考慮しておらず，類似したテクセルを圧縮していない．

そこで本研究では，クラスタリングを用いたニューラル BTF を提案する．BTF を k-means++を用いてクラスタリングし，クラスタごとに代表値をとる．そしてニューラルネットワークにその代表値を入力する．ニューラルネットワークのみを用いた場合と比較して，BTF の画質は劣るものの圧縮率を 2 倍以上向上させた．

1. はじめに

コンピュータグラフィックス (CG) の分野において，現実世界の質感の再現は重要な研究課題の一つである．特に，BTF (Bidirectional Texture Function, 双方向テクスチャ関数) を用いた手法は，布のように複雑な外観の表現を可能にする．一方，BTF は，現実世界の素材に様々な方向から光を当て，様々な方向から光の反射率を測定したテクスチャデータであるため，膨大なデータ量と，離散的な光源，視点方向の補間が問題となる．

BTF を圧縮する研究として，PCA (主成分分析)[1] などによる行列分解が用いられてきた．この手法は比較的に単純な処理である．また，解析的方法における研究では，BTF 画像のピクセル毎の方向依存のモデル化のために多項式 [2] とラフォーチュンローブによるモデル化手法 [3] が用いられてきた．パラメトリックモデルは，物体表面の反射率 (アルベド) や物体表面の法線のような幾何学的特徴を編集しやすいが，実世界の材質の表現が十分ではない．近年の研究では，光の経路の近似にニューラルネットワークが使用されてきている．Maximov らは，Deep Appearance Map[4] と呼ばれるディープニューラルネットワークを用いて，モデルの外観を表現する手法を考案した．また，入力としての画像と，出力として推定された BRDF パラメータを取得する逆問題も近年の研究の焦点となっている [5]．オートエンコーダは初期の圧縮方法として注目されてきた [6]．オートエンコーダによる圧縮方法は行列分解法のように補間の機能を持たせることができる．これは潜在変数に，補間に関する情報を加えることで可能となる [7][8]．Chen らは，照明の当たる場所のデータを，位置とレイの方向をパラメータ化したニューラルネットワークとして保存した [9]．しかし，これらの方法はマテリアルの外観データセットの圧縮の方法を具体的に確立しておらず，その場合，潜在変数とニューラルネットワークの両方のサイズを考慮する必要があった．

Rainer ら [10] は，オートエンコーダを用いて BTF を圧縮するニューラル BTF を提案した．本研究では，Rainer らの提案したニューラル BTF を先行研究と呼称する．先行研究では，照明の入射方向の次元，カメラへの反射方向の次元の圧縮であり，BTF のテクスチャ画像におけるテクセル間の類似性について考慮しておらず，類似したテクセルを圧縮していない．

そこで本研究は，テクセル間の類似性に着目し，類似したテクセルをクラスタリングし，クラスタごとに代表値をとることで，より高い BTF の圧縮を可能にする．本論文は BTF と Rainer らの先行研究について述べたのち，本研究のクラスタリングを用いた BTF の圧縮の手法，実験とその結果について述べる．考察とまとめでは，結果を踏まえて圧縮方法の評価や復元した BTF との誤差，レンダリング結果の精度について述べる．

2. ニューラル BTF の圧縮と補間

本節では，BTF と Rainer ら [10] の先行研究であるニューラル BTF について述べる．

2.1 BTF

BTF は，布のような複雑なモデルのレンダリングを可能にする．BTF を構成するデータは，照明方向と視点方

† 和歌山大学, Wakayama University

向を様々に変えて布などのマテリアルを撮影した数千から数万のテクスチャ画像である．BTF は，テクスチャ座標 $\boldsymbol{p} = (x, y)$，照明方向 (光の入射角) $\boldsymbol{\omega}_i = (\theta_i, \phi_i)$，視点方向 (カメラの方向，光の反射角) $\boldsymbol{\omega}_o = (\theta_o, \phi_o)$ を変数とし，それらのテクスチャ座標，方向における反射率を返す 6 次元関数 $f(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$ として扱われる (図 1)．

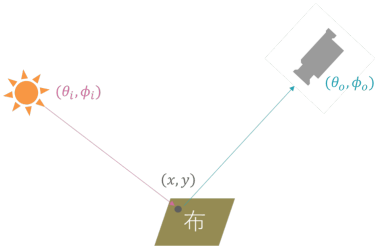

図 1: テクスチャ座標 $\boldsymbol{p} = (x, y)$，照明方向 $\boldsymbol{\omega}_i = (\theta_i, \phi_i)$，視点方向 $\boldsymbol{\omega}_o = (\theta_o, \phi_o)$ の関係．

BTF を用いて，輝度を計算したいシェーディング点での反射率を計算する手順は，は以下のとおりである．

1. シェーディング点における光の入射方向，反射方向，およびテクスチャ座標を得る (図 2)．
2. 数千から数万の画像データの中から，手順 1 の角度と近い入射，反射方向の画像を参照し，補間する (図 3)．
3. テクスチャ座標を基にその一点におけるピクセルを選ぶ (図 3)．
4. そのピクセルの値からモデル上のある一点における反射率を得る．

以上の操作を BTF を用いる全体のシェーディング点に対して行うことで，BTF を用いたレンダリングを行うことができる (図 4)．

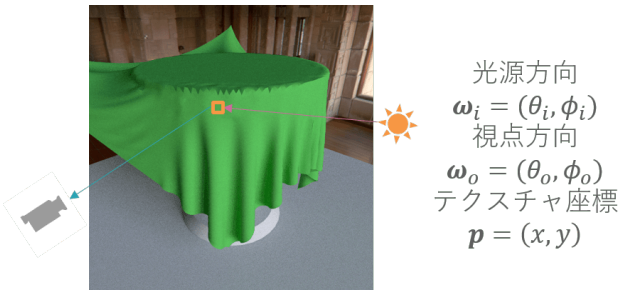

図 2: シェーディング点 (オレンジ色の枠) における光の入射方向，反射方向，およびテクスチャ座標の取得．

BTF の問題点は 2 つある．1 つ目は，BTF が数千から数万の画像から構成されるため，データ量が膨大になることである．2 つ目は，撮影されていない入射，反射方向の反射率を取得するためには，反射率の補間をする必要があることである．補間はレンダリング速度を下げるだけでなく，補間方法によってはレンダリングの正確さを失う．次項では，2 つの問題点の解決に使われるニューラルネットワークについて述べる．

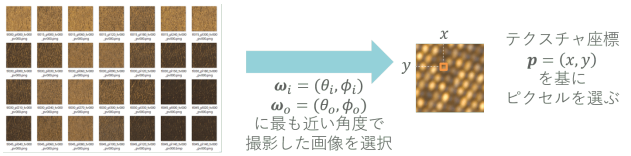

図 3: BTF を構成する画像列から反射率を取得．

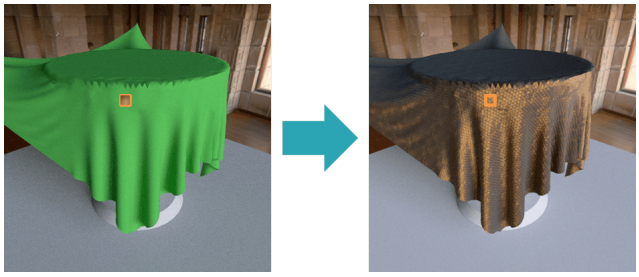

図 4: BTF を用いてレンダリングした例．

2.2 ニューラル BTF

先行研究 [10] は，オートエンコーダから着想を得たニューラルネットワークによって，BTF の膨大なデータ量を圧縮，および撮影されていない角度における反射率の補間を行った．通常のオートエンコーダはエンコーダとデコーダが対称の形になるが，ニューラル BTF と呼ばれる先行研究のニューラルネットワークは，エンコーダとデコーダが非対称の形である．これは，復元したい情報が任意の入射方向と出射方向に対する反射率であるためである．

ニューラル BTF (図 5) には ABRDF (Apparent Bidirectional Reflectance Distribution Function) を入力する．ABRDF とは，BTF $f(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$ のテクスチャ座標 $\boldsymbol{p}$ を固定した 4 次元の双方向反射率分布関数 $f_p(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$ である．また，ABRDF は，画像枚数分のデータを使い，ベクトルとして表現される．したがって，画像枚数を N とすると，ABRDF は N 次元ベクトルとして表現される．1 つの BTF から，BTF 画像の画素数だけ ABRDF は作成される．したがって，画素数を M とすると，ABRDF は M 個である．

ニューラル BTF (図 5) における活性化関数は ReLU，バッチサイズは 5，学習率は 0.05，オプティマイザは SGD (確率的勾配降下法)，誤差関数は MSE (平均二乗誤差)

が使用される．

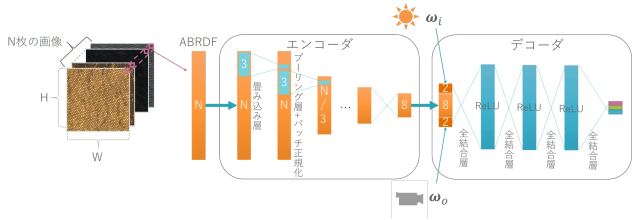

図 5: ニューラル BTF

2.3 入力データと出力データ

入力データ (ABRDF) は前処理として，対数変換を行った後，標準化する．標準化とは，平均が 0, 標準偏差が 1 になるようデータをスケーリングすることである．以下に入力データの前処理を表す式を示す．

$$\boldsymbol{x}_i^{(2)} = \log\left(\boldsymbol{x}_i^{(1)} + \boldsymbol{e}\right) \tag{1}$$

$$\boldsymbol{\mu} = \frac{1}{M}\sum_{i=1}^{M}\boldsymbol{x}_i^{(2)} \tag{2}$$

$$\boldsymbol{\sigma}^2 = \frac{1}{M}\sum_{i=1}^{M}\left(\boldsymbol{x}_i^{(2)} - \boldsymbol{\mu}\right)\otimes\left(\boldsymbol{x}_i^{(2)} - \boldsymbol{\mu}\right) \tag{3}$$

$$\boldsymbol{x}_i^{(3)} = \left(\boldsymbol{x}_i^{(2)} - \boldsymbol{\mu}\right)\oslash\boldsymbol{\sigma} \tag{4}$$

$\boldsymbol{x}_i^{(1)}\ (i \in \{1, ..., M\})$ は対数変換前の ABRDF である．ABRDF は前述の通り N 次元ベクトルとして表現される．$\boldsymbol{e}$ は $\boldsymbol{x}_i^{(1)}$ と同じ大きさで要素が全て 1 のベクトル，$\boldsymbol{x}_i^{(2)}$ は対数変換後の ABRDF，$\boldsymbol{x}_i^{(3)}$ は標準化後の ABRDF である．式 (3) における演算子 $\otimes$ はベクトルの要素ごとの乗算，式 (4) における演算子 $\oslash$ はベクトルの要素ごとの除算を表す．ニューラル BTF への入力は $\boldsymbol{x}_i^{(3)}$ である．

出力データは 1 ピクセル分の RGB 値であるため，3 次元ベクトルとして扱われる．正解データは，0 以上 1 以下に収まるように正規化を行う．正規化の式を以下に示す．

$$\alpha = \max\left\{\boldsymbol{y}_1, ..., \boldsymbol{y}_M\right\} \tag{5}$$

$$\left\{\frac{\boldsymbol{y}_1}{\alpha}, ..., \frac{\boldsymbol{y}_M}{\alpha}\right\} \tag{6}$$

$\boldsymbol{y}_i\ (i \in \{1, ..., M\})$ はニューラル BTF の出力データを表す．

2.4 エンコーダとデコーダ

エンコーダの構造は図 5 の中央部に示す通りである．ABRDF を十分小さいサイズに圧縮するために，畳み込みとマックスプーリング，バッチ正規化を順番に複数回行ったのち，全結合層を使って 8 次元の潜在変数に圧縮する．畳み込みの回数は ABRDF のベクトルの大きさ，すなわち BTF データの画像枚数に依存する．先行研究の実験において，22,801 の画像枚数を持つ BTF は 4 回の畳み込み，1,508 の画像枚数を持つ BTF は 1 回の畳み込みを行った．

デコーダの構造は図 5 の右側に示す通りである．デコーダはニューロン数が 106 である 4 層の全結合層を用いる．デコーダの入力は，エンコーダから出力された 8 次元の潜在変数，入射，反射方向を連結した 12 次元のベクトルである．

入射，反射方向は通常，球面座標 (θ, ϕ) で得られるが，ニューラル BTF に入力する際はステレオ投影によって変形した平面座標 (x, y) を用いる．ステレオ投影とは球面座標を平面座標で表現する方法の 1 つである．ステレオ投影は文献によって様々な定義式があるが，先行研究においてはステレオ投影に関する詳しい記載はなかったため，次節に本研究で用いたステレオ投影の式を示す．

デコーダの出力は，通常のオートエンコーダであれば ABRDF であるが，ニューラル BTF のデコーダは通常のオートエンコーダと異なるため，ニューラル BTF の出力は必要な 1 ピクセル分の RGB 値のみを補間して出力できる．

3. クラスタリングを用いたニューラル BTF の圧縮

3.1 ABRDF のクラスタリング

提案法では，類似した ABRDF をクラスタリングすることで，先行研究 [10] のニューラル BTF をさらに圧縮する．

類似した ABRDF を k-means++[11] で分類し，ABRDF のクラスタの ID を格納したリストを作成する．また，クラスタごとに代表値として中央値をとる．ニューラル BTF には，作成したリストを参照して，ABRDF の代わりに ABRDF の代表値を入力する．

ABRDF の次元が大きいため，k-means++でクラスリングを行う前に，主成分分析で ABRDF の次元削減を行う．後に保存する潜在変数は，BTF の画素数分の個数からクラスタ数 k に削減できるため，先行研究以上の圧縮が可能になる．図 6 に，ABRDF のクラスタリングの流れを示す．

本研究で利用するニューラル BTF について述べる．本研究のエンコーダは畳み込み層，プーリング層，バッチ正規化を 3 回繰り返した後，全結合層を通して 8 次元に圧縮する．畳み込み層のストライドは 1，パディングは 1，フィルタサイズは 3 である．RGB の 3 チャンネルで入力するため，1 層目の畳み込み層は 3 チャンネルであ

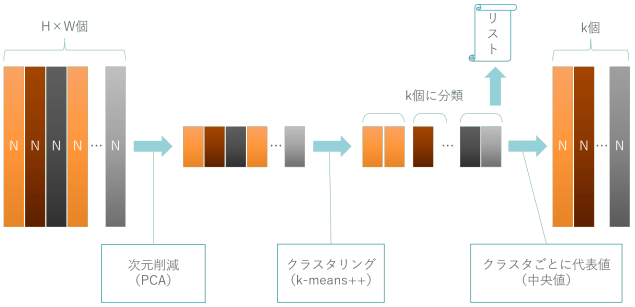

図 6: ABRDF のクラスタリング

る．プーリング層のカーネルサイズは 3，ストライドは 3 である．学習率は 0.08 とする．

ニューラル BTF に入力する際に用いる，球面座標 (θ, ϕ) から平面座標 (x, y) へのステレオ投影の式を以下に示す．

$$(R, \Theta) = \left(\frac{\sin(\pi - \theta)}{1 - \cos(\pi - \theta)}, \phi \right) \quad (7)$$

$$x = R \cos \Theta \quad (8)$$

$$y = R \sin \Theta \quad (9)$$

他のニューラル BTF の設定は，先行研究と同様に，全結合層の活性化関数が ReLU，バッチサイズが 5，オプティマイザが SGD，誤差関数が MSE である．

3.2 k-means++

k-means[11] はデータ点同士の平均二乗距離の最小化を利用するクラスタリング手法である．以下に k-means のアルゴリズムを示す．

1. それぞれの $i \in \{1, ..., k\}$ について，任意に k 個の最初の中心 $\mathcal{C} = \{c_1, ..., c_k\}$ を選ぶ．
2. データ点の集合 χ の全てのデータ点において，最も近い c_i $(i \in \{1, ...k\})$ を計算し，c_i をクラスタ中心とする集合 C_i に割り振る．
3. それぞれの $i \in \{1, ..., k\}$ について，クラスタ中心 c_i を C_i の全ての点の中心になるようにとりなおす．
4. $\mathcal{C}$ の更新が無くなるまで手順 2 と 3 を繰り返す．

k-means++ は k-means アルゴリズムにおける中心 $\mathcal{C}$ の任意選択 (手順 1) を改良したアルゴリズムである．以下に k-means++ のアルゴリズムを示す．

1-a. 最初の中心 c_1 を χ からランダムに選択する．

1-b. データ点と既に定められた最近傍中心 c_i の距離 $D(x)$ を計算し，重み付き確率分布 $\frac{D(x')^2}{\sum_{x \in \chi} D(x)^2}$ にしたがって，次のクラスタ中心 $c_i = x' \in \chi$ を選択する．

1-c. 決められたクラスタ数 k に至るまでステップ 2 を繰り返す．

この後，k-means アルゴリズムのステップ 2 - 4 を行う．

4. 結果

提案法の結果を，圧縮率と ABRDF の補間の性能によって評価する．使用した BTF は，シャンタン絹と似た布を撮影して作成された UTIA BTF Database の fabric02 [12]，ニットを撮影して作成された UBO2003 Datasets の WOOL [13]，黒板消しなどに利用される布を撮影して作成された UBO2003 Datasets の CORDUROY [13]，羊毛ニットを撮影して作成された UBO2003 Datasets の PULLI [13] の 4 種類である．これらの BTF は 6561 枚の画像からなる．すなわち，ABRDF の次元 $N = 6561$ である．また，BTF 画像を 100×100 にトリミングして実験を行った．すなわち，ABRDF の個数 $M = 10000$ である．実行環境には，CPU が Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz, メモリが 16GB, GPU が NVIDIA GeForce GTX 970 の PC を用いた．

クラスタリングに用いるクラスタ数のパラメータを k で表す．クラスタ数 k が小さいほど高い圧縮が可能となるが，クラスリングによる圧縮をすると誤差も大きくなる．ニューラル BTF の学習時間は 2 時間から 3 時間である．

4.1 圧縮

この節では本研究の圧縮性能について述べる．BTF 圧縮の結果を表 1 に示す．同じ画像枚数，画像サイズの BTF を使用したため BTF ごとの圧縮結果は同じである．BTF の各ピクセルの RGB データは float 型で表現されている．

先行研究における圧縮後データ量とは，デコーダの重みと，BTF を圧縮したデータである潜在変数のデータ量の和を表す．本研究における圧縮後データ量とは，デコーダの重みと，潜在変数と，作成したリストのデータ量の和である．デコーダの重みは，表 1 において「デコーダ」と表現している．

4.2 補間

この節では本研究の補間性能について述べる．誤差を評価する指標には，RMSE (Root Mean Square Error, 二乗平均平方根誤差) を用いる．RMSE は式 (10) で表される．

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \quad (10)$$

表 1: BTF データの圧縮前後のデータ量と圧縮比 (圧縮後データ量：圧縮前データ量)

	ニューラル BTF	提案法		
		$k = 500$	$k = 1000$	$k = 2000$
元データ	751MB	751MB	751MB	751MB
デコーダ	95.2KB	95.2KB	95.2KB	95.2KB
潜在変数	313KB	15.6KB	31.3KB	62.5KB
リスト	-	39.1KB	39.1KB	39.1KB
合計	408KB	150KB	166KB	197KB
圧縮比	1：1885	1：5127	1：4633	1：3904

y_i,$\hat{y}_i\,(i \in \{1,..,n\})$ はそれぞれ出力データと真値を表す．本研究における出力データは BTF の画像データの復元画像，真値とは BTF の画像データである．n は真値の個数である．本項においては BTF 画像の画素数と BTF テクスチャデータの枚数の積であり，次項においては復元画像の画素数である．

真値との誤差を表 2 に示す．表 2 より，いずれの BTF も先行研究と比べて誤差の値が大きくなった．また，実験前は k を小さくするほど大きな圧縮が可能となり，真値や先行研究の出力データとの誤差の値が大きくなると考えたが，表 2 より，k と誤差の関連性は大きくない．この原因は，ニューラル BTF の誤差の範囲内と考える．

表 2: 真値との誤差 (RMSE)

	ニューラル BTF	提案法		
		$k = 500$	$k = 1000$	$k = 2000$
fabric02	0.0203	0.0212	0.0200	0.0203
WOOL	0.0264	0.0275	0.0274	0.0274
CORDUROY	0.0285	0.0289	0.0282	0.0288
PULLI	0.0337	0.0346	0.0340	0.0340

4.2.1 復元

復元結果を図 7 に示す．参照画像は 100×100 にトリミングした BTF の画像を使用する．BTF それぞれの上段は視点方向 $\boldsymbol{\omega}_i = (15°, 60°)$，光源方向 $\boldsymbol{\omega}_o = (30°, 60°)$ を，下段は視点方向 $\boldsymbol{\omega}_i = (75°, 345°)$，光源方向 $\boldsymbol{\omega}_o = (345°, 0°)$ を参照および復元した．図 7 は，左の列から参照画像，先行研究の手法によって復元した画像，本研究の手法によって復元したパラメータ k の異なる画像である．

先行研究の手法で復元した画像は参照画像と比べぼやけているが，本研究における手法はぼやけている上に，粗さが見られる．

4.2.2 レンダリング結果

先行研究の手法で取得したデコーダの重み，潜在変数を使用してレンダリングを行った．また，本研究の手法で取得したデコーダの重み，潜在変数，ABRDF のクラスタの ID を格納したリストを使用し，$k = 500, 1000, 2000$ にパラメータを変更してレンダリングを行った．レンダリングのサンプル取得回数は 128 回である．レンダリング時間は手法によって変わらず，それぞれ 1 分ほどである．レンダリング結果およびその暗部と明部のトリミング画像を図 8 に示す．また，トリミング画像の先行研究と本研究の RMSE を表 3，4 に示す．

表 3: 暗部の誤差 (RMSE)

	提案法		
	$k = 500$	$k = 1000$	$k = 2000$
fabric02	0.0182	0.0173	0.0198
WOOL	0.0276	0.0225	0.0236
CORDUROY	0.0288	0.0355	0.0286
PULLI	0.0257	0.0275	0.0215

表 4: 明部の誤差 (RMSE)

	提案法		
	$k = 500$	$k = 1000$	$k = 2000$
fabric02	0.0244	0.0242	0.0242
WOOL	0.0275	0.0274	0.0274
CORDUROY	0.0408	0.0340	0.0239
PULLI	0.0404	0.0486	0.0295

5. 考察とまとめ

本研究では，ABRDF をクラスタリングした後，クラスタごとに代表値を取り，それをニューラルネットワークで学習させることで，BTF の更なる圧縮に取り組んだ．元の BTF 画像との誤差は最大 4.4%の誤差が，$k = 500$ のときの fablic02 において増加したが，先行研究と比べて 2 倍以上の圧縮が可能となった．

圧縮と誤差の関係について述べる．実験前は大きく圧縮するほど，すなわち k を小さくするほど画像の誤差が大きくなると考えていた．しかし，今回の実験では k の値と誤差の関連性は低かった．原因はニューラルネットワークの学習の誤差と考えるが，検証するには k の値をより多く変更することや，同じ値で複数回の実験を行う必要がある．今後はクラスタ数 k の調整やニューラルネットワークの適用など他の圧縮方法を採用することにより，より高い圧縮率や，精度を維持したままの圧縮を実現できる可能性がある．

復元画像について述べる．先行研究に関しては参照画像よりもぼやけており，本研究に関してはぼやけた上に粗さを確認できる．粗さの原因は，同じクラスタのピクセルが本来異なる輝度であるにも関わらず，同じ輝度にされるためである．画像の粗さをなくすためには，それぞれの ABRDF の平均輝度のような，ABRDF 固有のパラメータを乗算するなど，工夫をする必要がある．

レンダリング結果について述べる．復元画像と比べ，

粗さは確認されなかった．これは，レンダリング結果画像の 1 つのピクセルにおいて 100 回以上のサンプル採取をしたことにより，類似した隣接ピクセルとの輝度に差が生まれ，滑らかになったためと考える．レンダリング結果の RMSE の値に大きな差は見られなかったが，見た目に関しては，$k = 500$ の暗部のトリミング画像の上部が特に総じて精細さを欠いていた．

先行研究を追実験した過程で判明したことは，ニューラルネットワークにとって再現しやすい BTF とそうでない BTF があることである．これらの違いはニューラルネットワークにとって規則性のある BTF とそうでない BTF の差であると考えるが，明確な理由は定かにならなかった．先行研究と本研究のニューラルネットワークの精度の向上および汎用性のためには，原因を究明し，手法を改良する必要がある．

参考文献

[1] Michael Weinmann, Juergen Gall, and Reinhard Klein. Material classification based on training data synthesized using a btf database. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III*, pp. 156–171. Springer International Publishing, 2014.

[2] Tom Malzbender, Dan Gelb, and Hans Wolters. Polynomial texture maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pp. 51–528. Association for Computing Machinery. 2001.

[3] David K. McAllister, Anselmo Lastra, and Wolfgang Heidrich. Efficient rendering of spatial bi-directional reflectance distribution functions. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, HWWS '02, pp. 79–88. Eurographics Association. 2002.

[4] Maxim Maximov, Laura Leal-Taixe, Mario Fritz, and Tobias Ritschel. Deep appearance maps. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8728–8737, 2019.

[5] Xiao Li, Yue Dong, Pieter Peers, and Xin Tong. Modeling surface appearance from a single photograph using self-augmented convolutional neural networks. *ACM Transactions on Graphics*, Vol. 36, No. 4, 2017.

[6] Geoffrey E. Hinton and Ruslan Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, Vol. 313, pp. 504–507, 2006.

[7] Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. Deep image-based relighting from optimal sparse samples. *ACM Transactions on Graphics*, Vol. 37, No. 4, p. 126, 2018.

[8] Stephen Lombardi, Jason Saragih, Tomas Simon, and Yaser Sheikh. Deep appearance models for face rendering. Vol. 37. Association for Computing Machinery, 2018.

[9] Anpei Chen, Minye Wu, Yingliang Zhang, Nianyi Li, Jie Lu, Shenghua Gao, and Jingyi Yu. Deep surface light fields. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, Vol. 1, No. 1, 2018.

[10] Gilles Rainer, Wenzel Jakob, Abhijeet Ghosh, and Tim Weyrich. Neural btf compression and interpolation. *Computer Graphics Forum*, Vol. 38, pp. 235–244, 2019.

[11] David Arthur and Sergei Vassilvitskii. K-means++: the advantages of careful seeding. In *In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007.

[12] Vávra R. Haindl M., Filip J. Digital material appearance: the curse of tera-bytes. *ERCIM News*, No. 90, pp. 49–50, 2012.

[13] Mirko Sattler, Ralf Sarlette, and Reinhard Klein. Efficient and realistic visualization of cloth. In *Proceedings of the 14th Eurographics Workshop on Rendering*, EGRW '03, pp. 16–177. Eurographics Association. 2003.

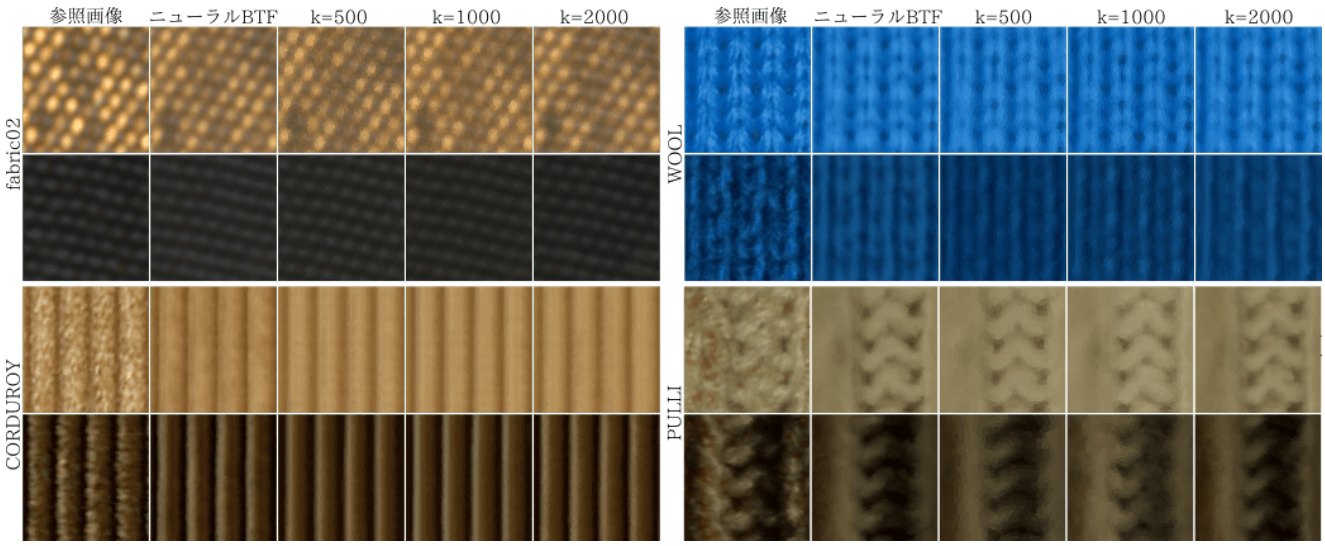

図 7: BTF の復元画像：上の段から 4 種類の BTF を，2 種類の入射，反射方向で参照および復元した．

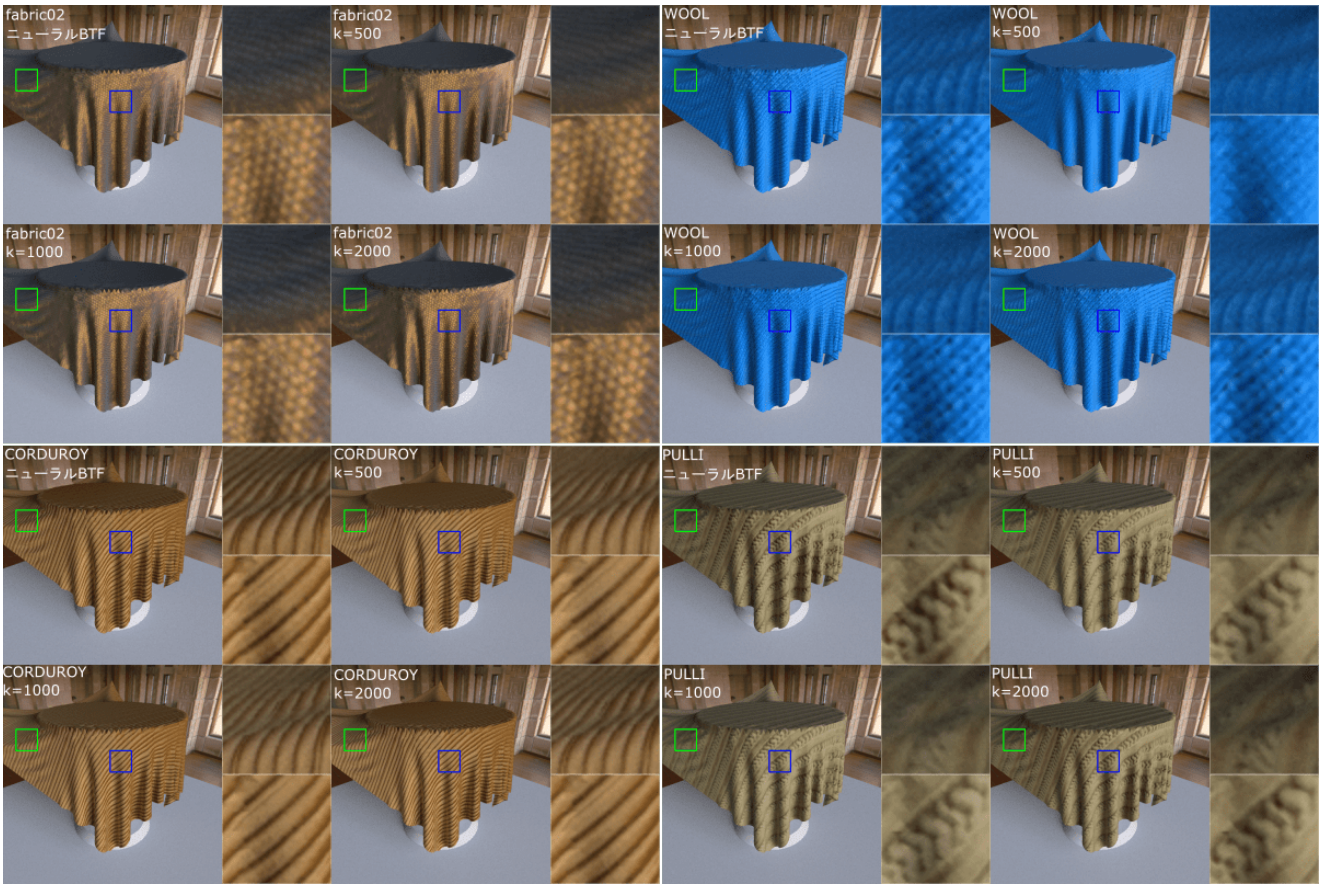

図 8: レンダリング結果：先行研究と本研究の手法におけるレンダリング結果画像と，その一部拡大画像である．