

CEDEC2026

生成AIを用いたPBRマテリアル付き の3Dメッシュ生成手法

シリコンスタジオ株式会社

研究開発室

京田 文人

2026/7/24



- 画像生成AIの基礎知識
- 生成AIによる3Dモデル生成手法
- 生成AIによるテクスチャ生成手法
- 提案手法
- 生成結果と今後の課題
- まとめ

- 画像生成AIとは
- 画像生成AIの進化
- 画像生成AIの応用例

- 生成AI (Generative AI)

- 従来のAI研究では画像の**認識**がメインテーマだった

- 画像→「ネコ」



ネコ



イヌ



- 認識が十分にできるようになれば、「ネコ」らしい画像も作れる

- 認識できるということはネコの画像の持つ**本質的な特徴**を理解している
- 本質的な特徴から、それを持った画像を作る→**生成に応用可能！**

- **確率分布**を学習

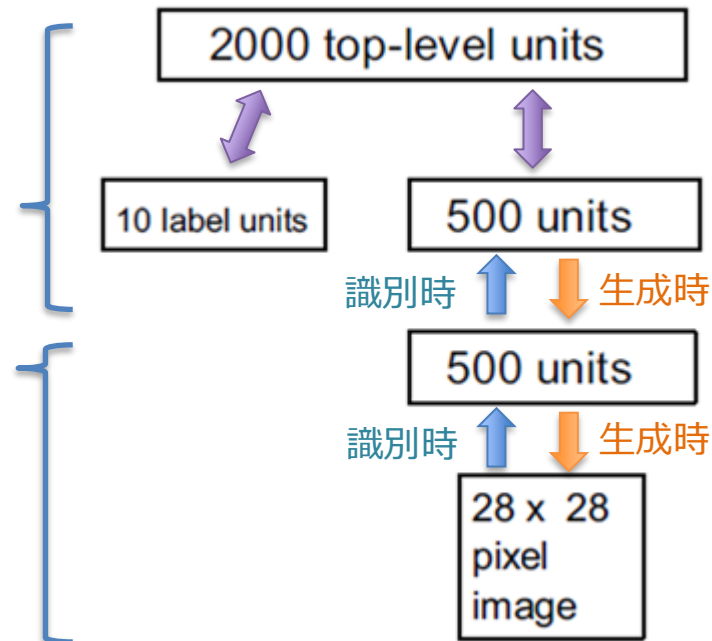
- 確率的にサンプリングするので、毎回異なるバリエーションを作れる
- 「特徴を捉えていてそれっぽい」けど、学習データの単なるコピーではない

- 近年では、画像だけでなく、動画や3Dモデルなど様々な生成が可能になって来ている

- Deep Belief Networks
- VAE
- GAN
- 拡散モデル(Diffusion Model)
- Large Multimodal Model

Deep Belief Networks^[1]

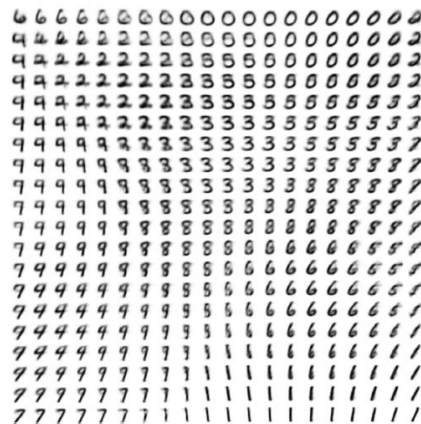
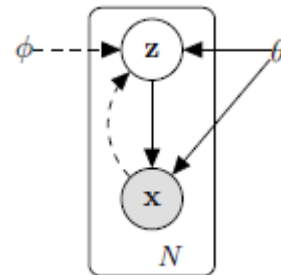
- 0～9の手書き数字の生成
 - 手書き数字の識別もできる
 - 上位2段で数字の抽象的な情報を保持
 - 下位2段で具体的な画像を識別/生成



[1] G. E. Hinton et al., "A fast learning algorithm for deep belief nets", 2006

VAE: Variational Auto-Encoder^[2]

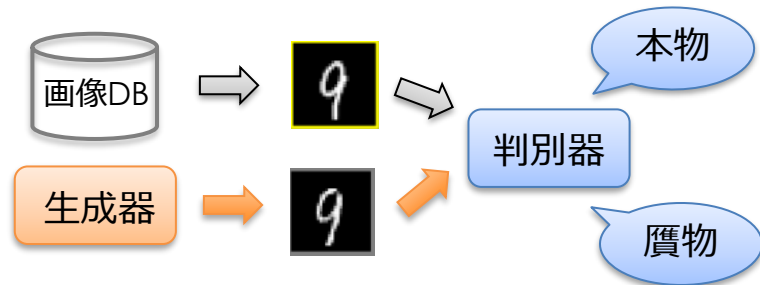
- 一般的な画像生成モデル
 - 手書き数字だけに限定されず、様々な画像を生成できる
- 潜在空間にエンコードして復元
 - 確率分布 p で潜在変数 z が決まる: $p_{\theta}(z)$
 - 潜在変数が z のときのデータ x が条件付き確率で決まる: $p_{\theta}(x|z)$
 - 確率分布 p を決める生成モデルのパラメータ θ を求めたい
 - データ x から潜在変数 z を推定する確率分布: $q_{\phi}(z|x)$
 - 識別モデルのパラメータ ϕ も一緒に学習
- 潜在空間内で連続的に補間できる
 - 細部の表現が苦手



[2] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes", 2013

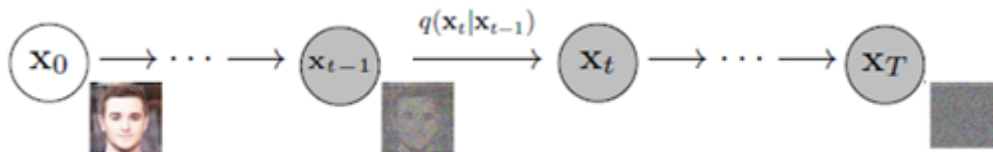
GAN: 敵対的生成ネットワーク[3]

- 2つの敵対的なAIをそれぞれ学習
 - 画像を生成する**生成器**
 - 画像が本物か生成されたものか識別する**判別器**
- ラベル付き教師データが内部で生成できる
 - 生成器が生成した画像を偽物、データセットの画像を本物とするので、判別器用の正解は明らか
 - 生成器は判別器の結果が教師データになる
- 判別器の学習がうまくいけば、相互の競争によってクオリティが上がり、細部まで人間には見分けのつかないほどの画像が生成される
 - 逆に、判別器の学習が変な方向に行ってしまった場合、判別器を騙すためだけの変な画像が生成されてしまう



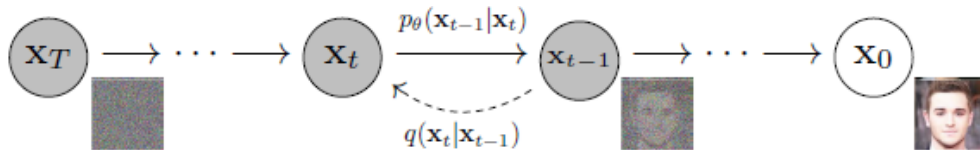
[3] I. J. Goodfellow et al., "Generative Adversarial Nets", 2014

- 画像にノイズを少しずつ付加してその過程(**拡散過程**)を学習
 - ノイズを加える前後を教師データにする
 - 入力と正解のペアを用意する必要がなく、どんな画像でも学習に使える



(論文[5]より引用して編集)

- ノイズを少しずつ除去して画像を生成
 - ノイズの加えられた後の画像から一つ前の画像を推定(**逆拡散過程**)
 - これを繰り返していくと、完全なノイズから高画質な画像が生成される



(論文[5]より引用)

[4] J. Sohl-Dickstein et al., "Deep Unsupervised Learning using Nonequilibrium Thermodynamics", 2015

[5] I. J. Goodfellow et al., "Denoising Diffusion Probabilistic Models", 2020

- 途中段階に**制約条件**を入れて、ガイドに従わせることができる
 - テキストでの指示
 - 深度値(depth)や法線(normal)を入れて、物体形状に沿わせる
- ステップ数が多いと生成に時間がかかる
- 近年はより効率化した手法が提案されている
 - Latent Diffusion^[6]
 - VAEで圧縮
 - Flow Matching^[7]
 - ノイズから画像への流れを学習し、進むべき方向を推定、微分方程式を解く
 - 1000ステップほど必要だったのが、数十ステップで可能に

[6] R. Rombach et al., "High-Resolution Image Synthesis with Latent Diffusion Models", 2022

[7] R. Rombach et al., "Flow Matching for Generative Modeling", 2023

- 大規模言語モデル(LLM)が画像・音声・動画なども扱えるようになった
 - 画像もトークン列にエンコードし、テキストのトークンと同様に扱う
 - 画像生成する際は画像生成AIにどんな画像にすべきかという条件を渡す
 - 世界の幅広い知識を持っている
 - 画像と言葉の対応関係を大量のデータから学習
 - 会話で長い文章による指示を理解できる

- 拡散モデルによる画像生成は様々な画像処理に応用可能

- スタイル変換

- アニメ風、水彩画風など

- Image editing

- 背景など一部だけ変更

- Inpainting

- 人物など一部の対象だけ消し、跡に残った空白を埋める

- Outpainting

- 画像を左右に拡張し、写っていない領域を埋める

- 超解像

- 低解像度の画像を高画質化



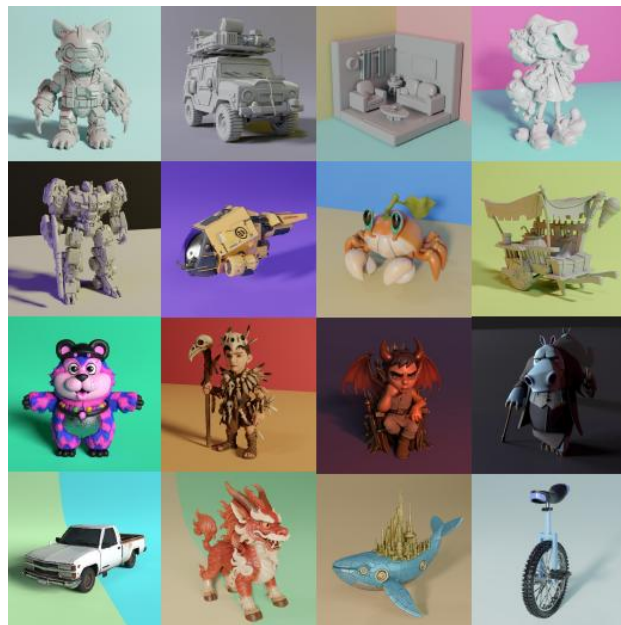
| 拡散モデルがもたらしたブレイクスルー

- 従来の画像処理
 - Inpaintingやスタイル変換などそれぞれのタスクで専用の手法が研究されてきた
 - 機械学習でも、各タスクに対応した教師データ(入力と正解画像のペア)を大量に集める必要があった
- 拡散モデル
 - **大量の画像だけ**あれば学習できる
 - 元画像にノイズを加えるだけで拡散過程の教師データになる
 - 世界中のあらゆる画像を使って非常に大規模な学習ができた
 - 画像の構造・質感・物体・背景などの**統計的な知識**を獲得
 - 生成時にテキスト、画像、マスクなどで条件を加えることができる
 - 画像処理のほとんどのタスクは**条件付き画像生成**に含まれる
- 結果、汎用的に**どんなタスクもこなせる**ようになった
 - 自然言語処理の分野で、次単語予測という、どんな文章でも学習できる手法を用いて大規模言語モデルを作ったら、どんなタスクもこなせるようになったのと同じ現象
 - このような大規模学習は、データ・モデル・計算量を増やすほど性能が向上する(**スケーリング則**)

|生成AIによる3Dモデル生成手法

- 3D生成AIとは
- 3Dモデルの構成
- 3D生成技術の進化
- 各3D生成サービスのクオリティ比較

- テキストや2D画像から3Dモデルを生成

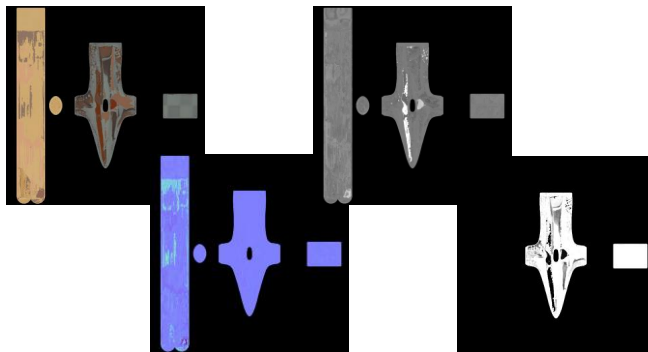
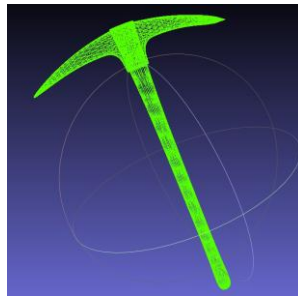
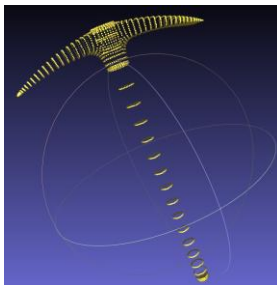


Hunyuan3D 2.5^[8]

[8] Z. Lai et al., "Hunyuan3D 2.5: Towards High-Fidelity 3D Assets Generation with Ultimate Details", 2025

|3Dモデルの構成

- 3Dモデルに必要な要素
 - 頂点
 - 面(三角形)
 - UV座標
 - テクスチャ
 - PBRマテリアル
 - ベースカラー
 - 法線
 - メタリック
 - ラフネス
 - AO
 - Height, Displacement
 - Opacity
 - リグ



- NeRF
- DreamFusion
- Gaussian Splatting
- Large Reconstruction Model
- Native 3D Generation

|NeRF: Neural Radiance Fields^[9]

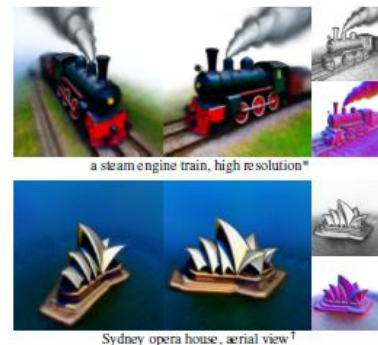
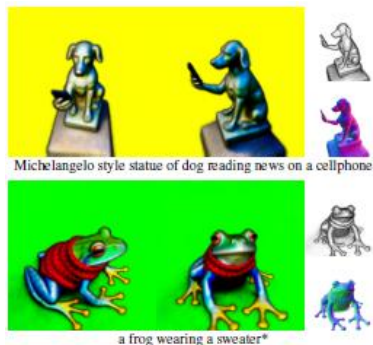
- 「空間全体」をニューラルネットワークで表現
 - 3D空間中の各点の位置と視線方向を入力すると、その点の色(radiance)と密度(density)を出力
 - 多数の写真から学習し、新しい視点を生成できる
- 長所
 - 見た目は写真のようにリアル
- 問題点
 - メッシュではないので編集が難しい
 - 描画が遅い



[9] B. Mildenhall et al., "High-Resolution Image Synthesis with Latent Diffusion Models", 2020

[nerfstudio](https://nerfstudio.org)

- 画像生成AI(Stable Diffusion)とNeRFを組み合わせた
 - テキストから画像を生成
 - どの角度から見ても同じ見た目になるようNeRFを最適化
- 長所
 - Text-to-3Dを実現
 - 既存の画像生成AIを利用でき、追加学習不要
- 問題点
 - 最適化に時間がかかる
 - メッシュ品質が低い
 - テクスチャがぼやける



[10] B. Poole et al., "DreamFusion: Text-to-3D using 2D diffusion", 2022

Gaussian Splatting^[11]

- 空間を3Dガウシアン（ぼかした点）の集合で表現
 - 画面上に投影し、GPUでラスタライズして描画
- 長所
 - レンダリングが高速でリアルタイム可能
 - 学習もNeRFより速い
- 問題点
 - メッシュではないので編集が難しい
 - 既存のゲームエンジンで扱いにくい



XGRIDSにより取得した3DGSデータ（福井県/一乗谷朝倉氏遺跡）福井コンピュータ株式会社

[11] B. Poole et al., "3D Gaussian Splatting for Real-Time Radiance Field Rendering", 2023

- 2D画像から3Dメッシュの再構成をニューラルネットワークで学習
 - InstantMesh^[12]
 - DreamFusionと異なり最適化は不要で1回の推論で生成できる
 - Pixel3D^[13]
 - 入力画像をピクセル単位で忠実に3D化
- 商用化されている
 - TripoSR
 - TRELIS
- 長所
 - 生成が高速(数秒～数分)
 - 高品質なメッシュ



[TripoSR : AI による 3D 再構成プラットフォーム](#)

[12] J. Xu et al., "InstantMesh: Efficient 3D Mesh Generation from a Single Image with Sparse-view Large Reconstruction Models", 2024

[13] D.-Y. Li et al., "Pixel3D: Pixel-Aligned 3D Generation from Images", 2026

Native 3D Generation

- テキスト→2D画像生成→3D再構成ではなく、直接3Dメッシュを生成する
 - 3Dをトークン化して学習
 - 最適化不要で1回の推論で生成
 - UVやマテリアルも同時生成
- 最新モデルの主流
 - Hunyuan3D^[8]
 - Meshy
 - Stable Fast 3D
 - Rodin
 - Tripo



[8] Z. Lai et al., "Hunyuan3D 2.5: Towards High-Fidelity 3D Assets Generation with Ultimate Details", 2025

(論文[13]より引用)

各3D生成サービスのクオリティ比較

- 2024年11月時点での弊社内での調査
- 評価は弊社エンジニア、テクニカルアーティストによる主観

	生成時間	メッシュのクオリティ	テクスチャ	テクスチャのクオリティ
Tripo AI	2分	高、左右非対称	PBR	△ノーマルマップにノイズ
Meshy	3.5~4分	高、左右対称で編集しやすいが、一部非多様体	カラーのみ	△ライティングが分離できていない
Rodin	2分	低、ほぼ左右対称で編集しやすい	PBR	△ノーマルマップにノイズ
Luma AI	10~15分	低、左右非対称	PBR	×ノイズ、似たような質感しか出ない
CSM	5~10分+1日以上(高精度化)	低(高精度化前)、座標軸が正面を向いていない	カラーのみ	△色ムラが多い
Masterpiece X	2分	高、ほぼ左右対称で編集しやすいが、非多様体を多く含む	カラーのみ	×塗分けが正しくない。ノイズ
sudo.ai	20秒	低、がたつきが多い	カラーのみ	△ライティングが分離できていない

3Dモデル生成の課題

- 生成AIの進化により、生成されるメッシュは非常に高品質になっている
- しかし、そのままでは実用的ではない部分もある
 - パーツ間のメッシュが分離されていない
 - または分離されているが用途に合わない
 - UVやテクスチャのクオリティが低い場合もある
 - PBRマテリアルではなく、カラーテクスチャのみのサービスもある



|生成AIによるテクスチャ生成手法

- Text2Tex
- PBRマテリアル生成
 - DreamMat
 - MaterialAnything
 - Make-it-Real

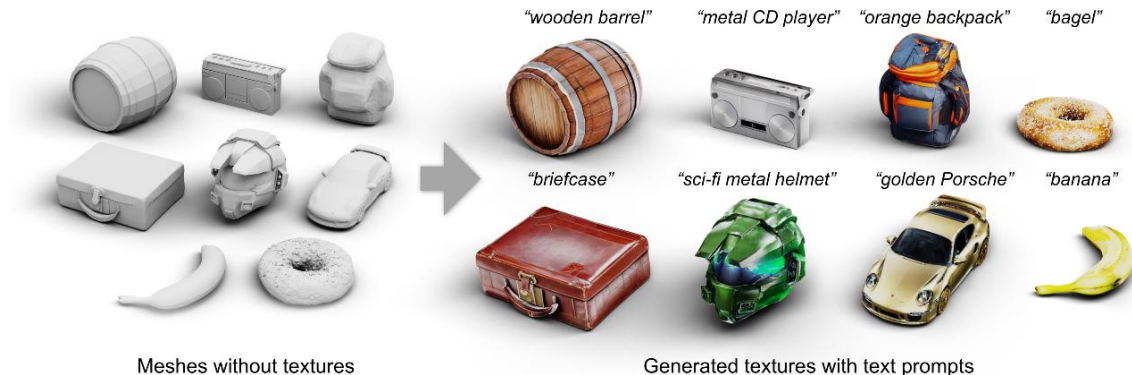
- 画像生成AI(拡散モデル)によって3Dモデルのテクスチャを生成

- 入力

- メッシュの頂点座標、面
- テキスト

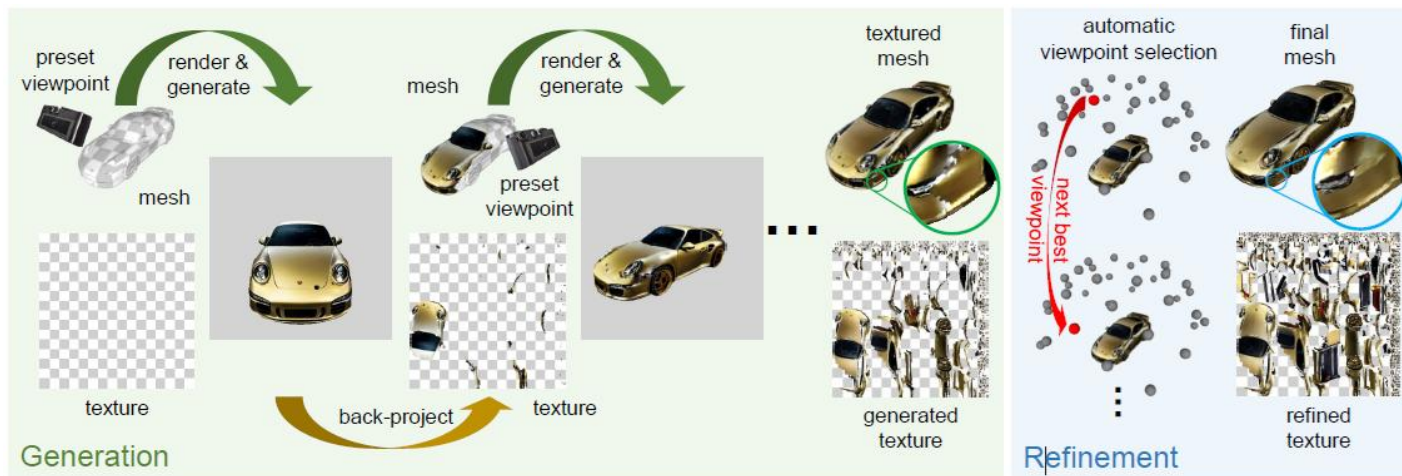
- 出力

- UV座標(xatlasで生成)
- カラーテクスチャ



[14] D. Z. Chen et al., "Text2Tex: Text-driven Texture Synthesis via Diffusion Models", 2023

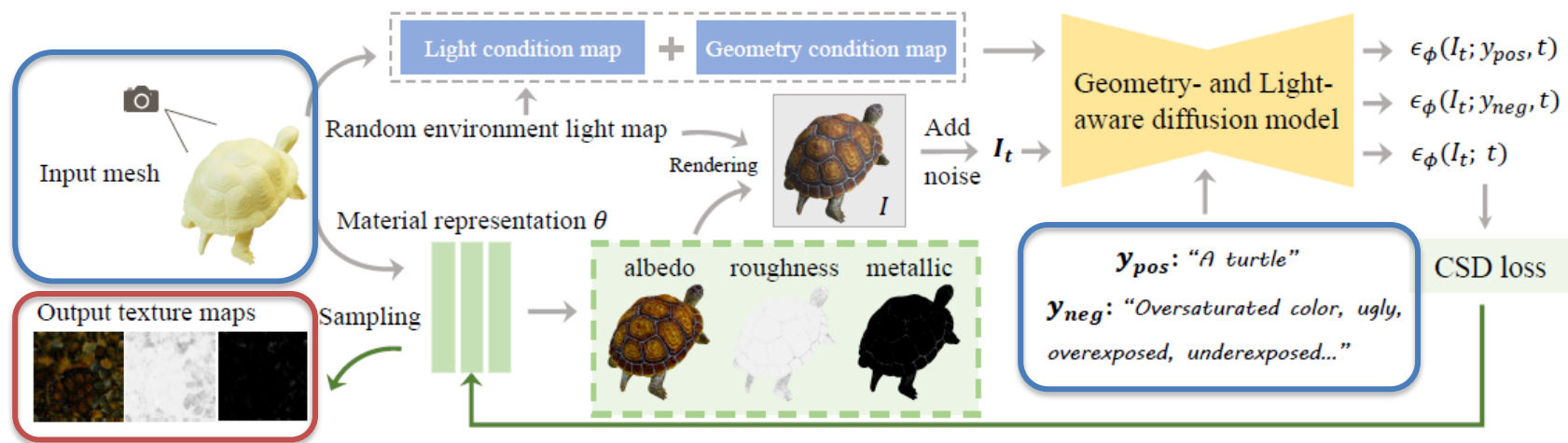
- 画像生成AI(拡散モデル)によって3Dモデルのテクスチャを生成
 - 深度値(depth)によって物体形状に合わせた画像を生成
 - UV空間のテクスチャに書き込む
 - 視点を変えてInpainting(穴埋め)画像生成を繰り返す
 - 生成後、継ぎ目の部分にフォーカスしてリファインメント



[14] D. Z. Chen et al., "Text2Tex: Text-driven Texture Synthesis via Diffusion Models", 2023

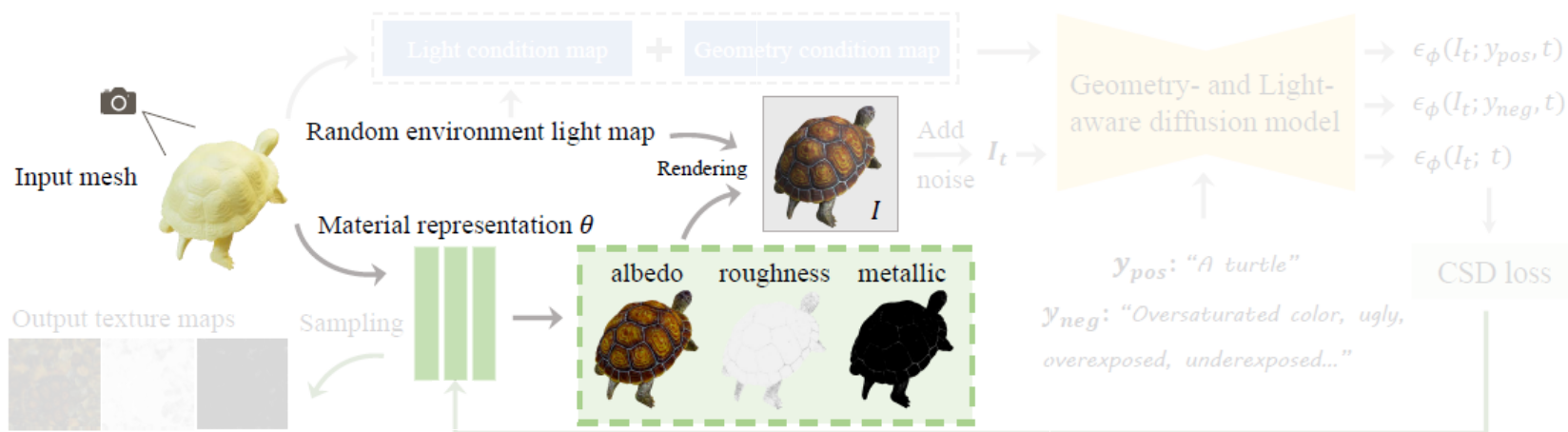
- メッシュから、PBRマテリアルのテクスチャを生成する
 - DreamMat
 - MaterialAnything
 - Make-it-Real

- メッシュから、PBRマテリアルのテクスチャを生成する
 - 入力: テクスチャのないメッシュ、テキストによる指示
 - 出力: ベースカラー、ラフネス、メタリックのテクスチャマップ



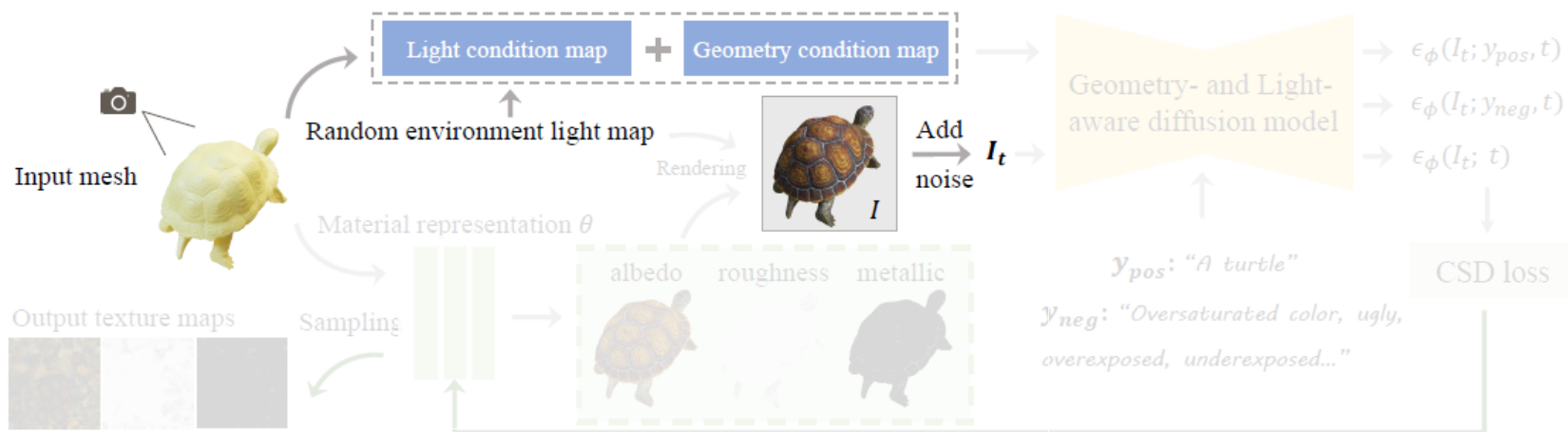
[15] Y. Zhang et al., "DreamMat: High-quality PBR Material Generation with Geometry- and Light-aware Diffusion Models", 2024

- ハッシュグリッドベースのマテリアル表現 θ を使用してレンダリング
 - マテリアルはランダムな初期値
 - 環境光はデータベースからランダムに選択



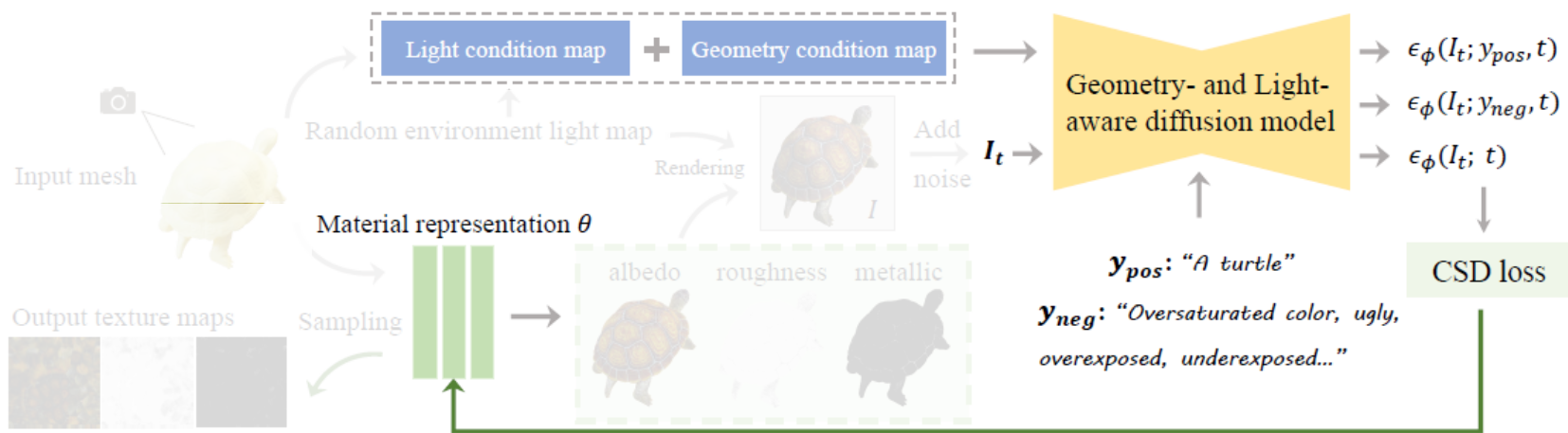
[15] Y. Zhang et al., "DreamMat: High-quality PBR Material Generation with Geometry- and Light-aware Diffusion Models", 2024

- マテリアル表現 θ を最適化
 - レンダリングされた画像にノイズを付加
 - ジオメトリ条件マップとライティング条件マップを作成
 - 拡散モデルを適用してノイズを除去
 - 蒸留損失(CSD loss)を計算してマテリアルパラメータ θ を更新



[15] Y. Zhang et al., "DreamMat: High-quality PBR Material Generation with Geometry- and Light-aware Diffusion Models", 2024

- マテリアル表現 θ を最適化
 - レンダリングされた画像にノイズを付加
 - ジオメトリ条件マップとライティング条件マップを作成
 - 拡散モデルを適用してノイズを除去
 - 蒸留損失(CSD loss)を計算してマテリアルパラメータ θ を更新

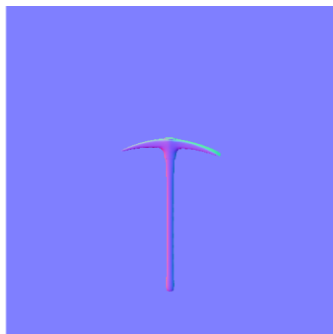


[15] Y. Zhang et al., "DreamMat: High-quality PBR Material Generation with Geometry- and Light-aware Diffusion Models", 2024

- 問題点
 - マテリアルパラメータの最適化ループの中に拡散モデルが組み込まれているため、時間がかかる
 - メタリックとラフネスが不正確
- つるはしのマテリアル生成結果
 - 頭部のメタリックが高くなっていない



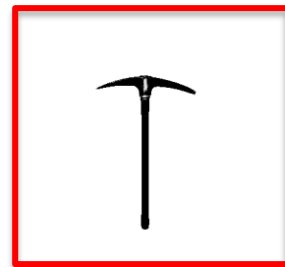
レンダリング結果



法線



アルベド



メタリック



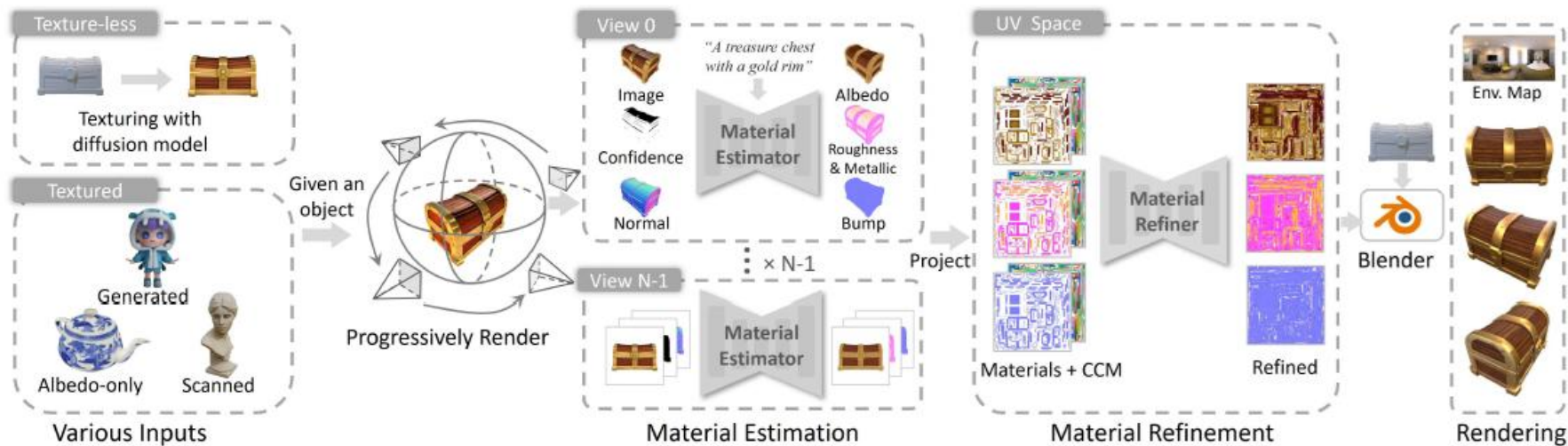
ラフネス

- テクスチャの有無に関わらず、PBRマテリアルを生成



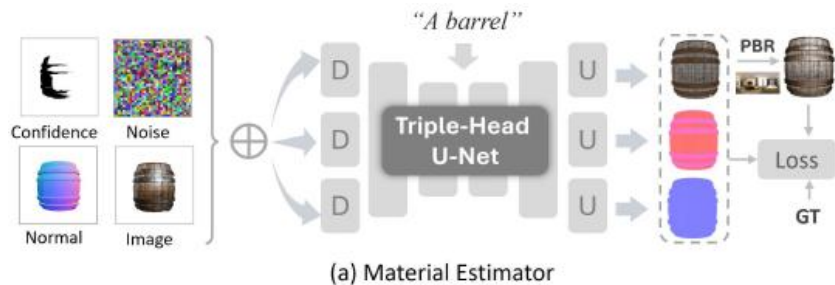
[15] X. Huang et al., "Material Anything: Generating Materials for Any 3D Object via Diffusion", 2024

- テクスチャのないメッシュの場合
 - まず拡散モデル(Text2Texなど)でテクスチャを生成する
- テクスチャはあるがライティングが不正確な場合
 - ライティングの信頼度を用いて、入力テクスチャを参照するかを制御

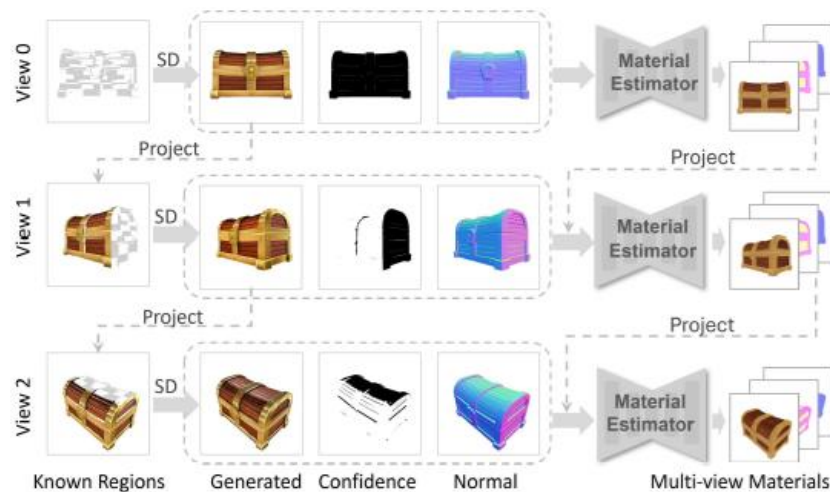
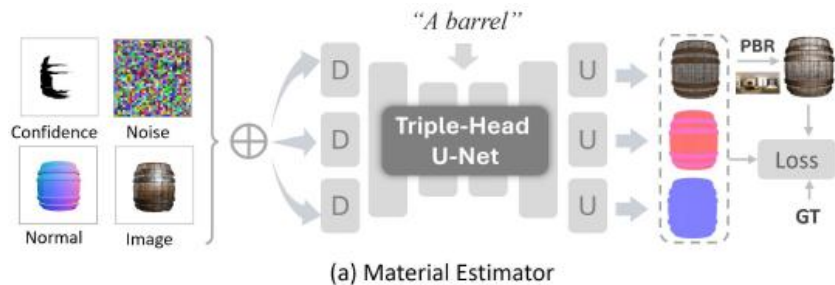


[15] X. Huang et al., "Material Anything: Generating Materials for Any 3D Object via Diffusion", 2024

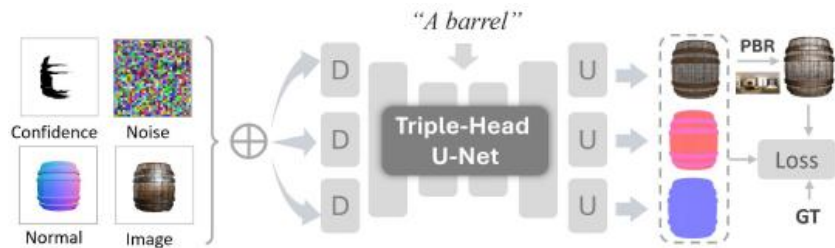
- U-NetベースのニューラルネットワークでPBRマテリアルを推定
- 視点ごとに繰り返してオブジェクト全体を生成
- Material RefinerによってUV空間でテクスチャの品質を上げる



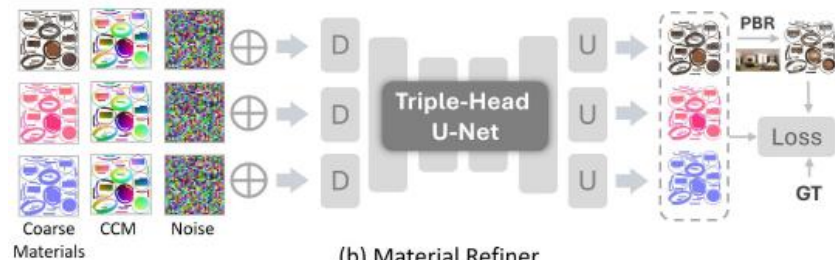
- U-NetベースのニューラルネットワークでPBRマテリアルを推定
- 視点ごとに繰り返してメッシュ全体でマテリアル生成
- Material RefinerによってUV空間でテクスチャの品質を上げる



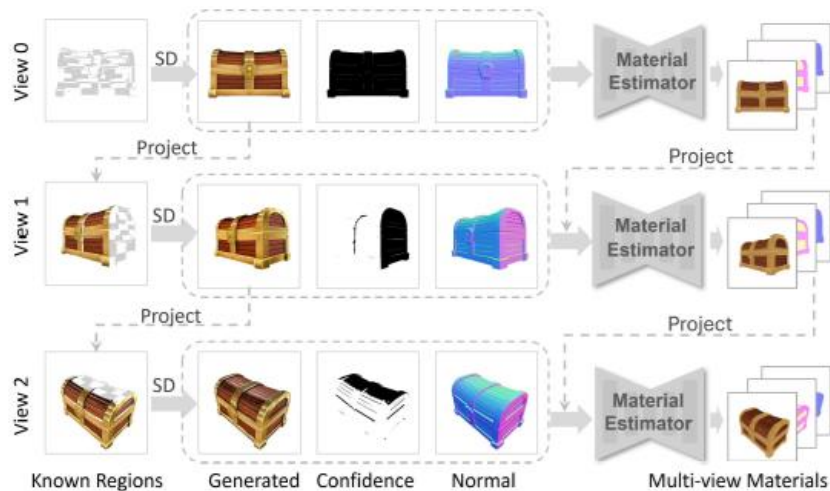
- U-NetベースのニューラルネットワークでPBRマテリアルを推定
- 視点ごとに繰り返してオブジェクト全体を生成
- Material RefinerによってUV空間でテクスチャの品質を上げる



(a) Material Estimator



(b) Material Refiner



[15] X. Huang et al., "Material Anything: Generating Materials for Any 3D Object via Diffusion", 2024

- 最適化ベースのDreamMatより格段に速い



[15] X. Huang et al., "Material Anything: Generating Materials for Any 3D Object via Diffusion", 2024

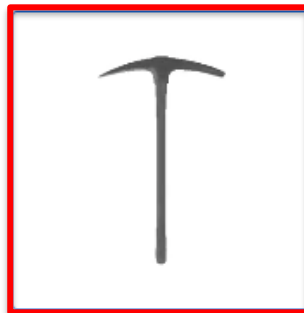
- つるはしに対してはDreamMatと同じく、メタリックが正しくならなかった



レンダリング結果



アルベド

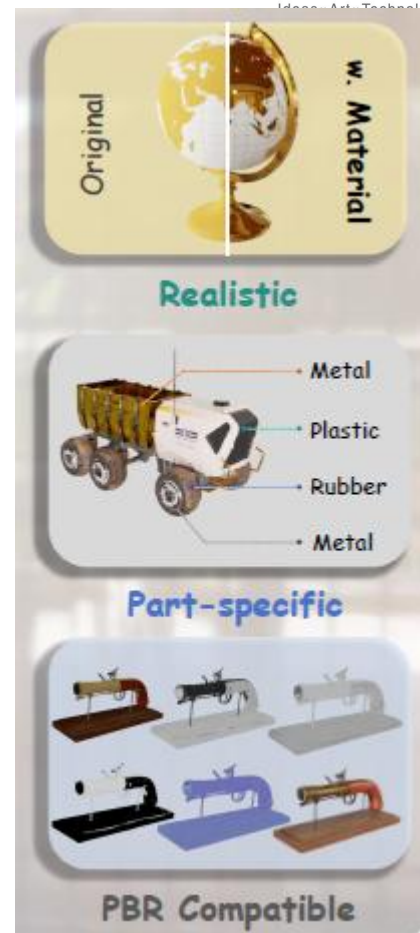
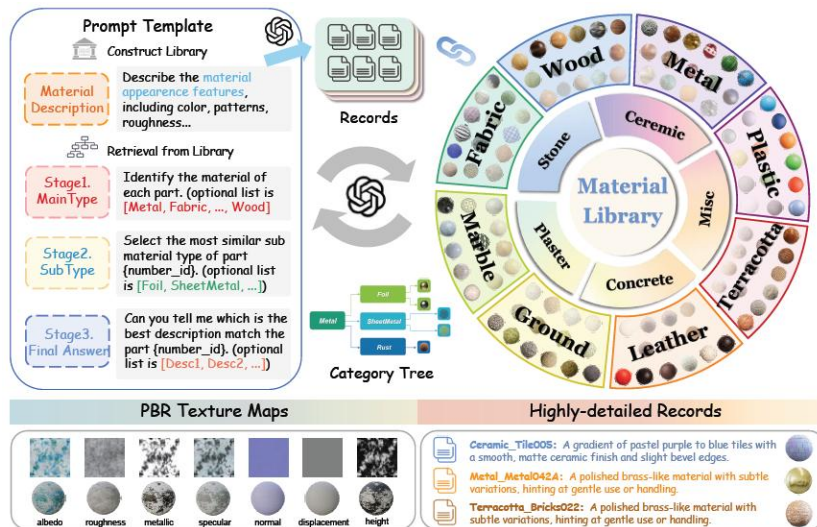


メタリック



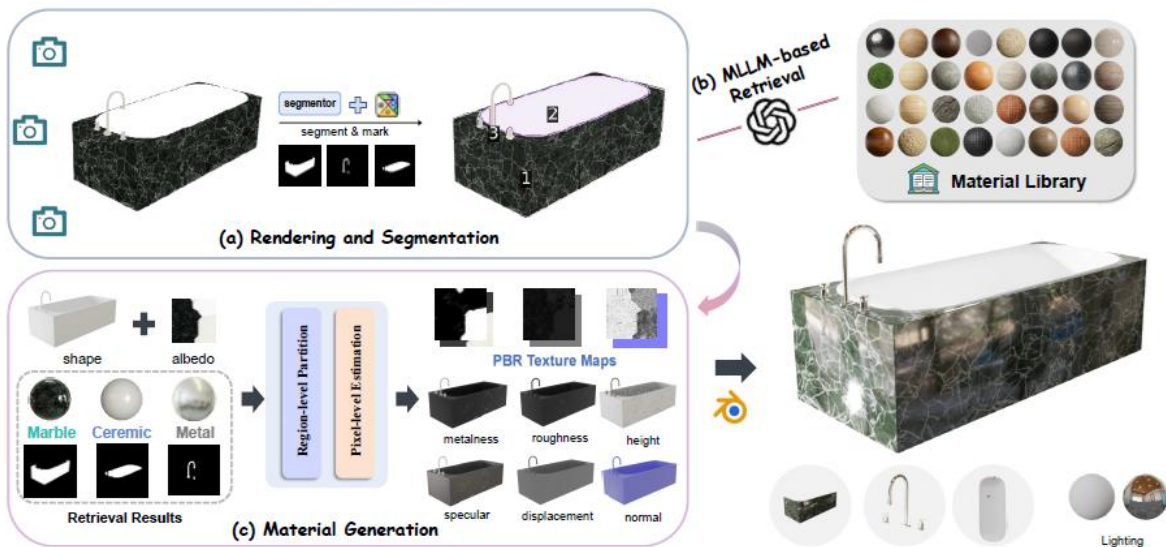
ラフネス

- LLM(大規模言語モデル)に問い合わせ、各パーツに適したマテリアルを選択させる
 - LLMが持っている現実の知識を活用
 - 金属と非金属をはっきり分けられる



[16] Y. Fang et al., "Make-it-Real: Unleashing Large Multimodal Model for Painting 3D Objects with Realistic Materials", 2024

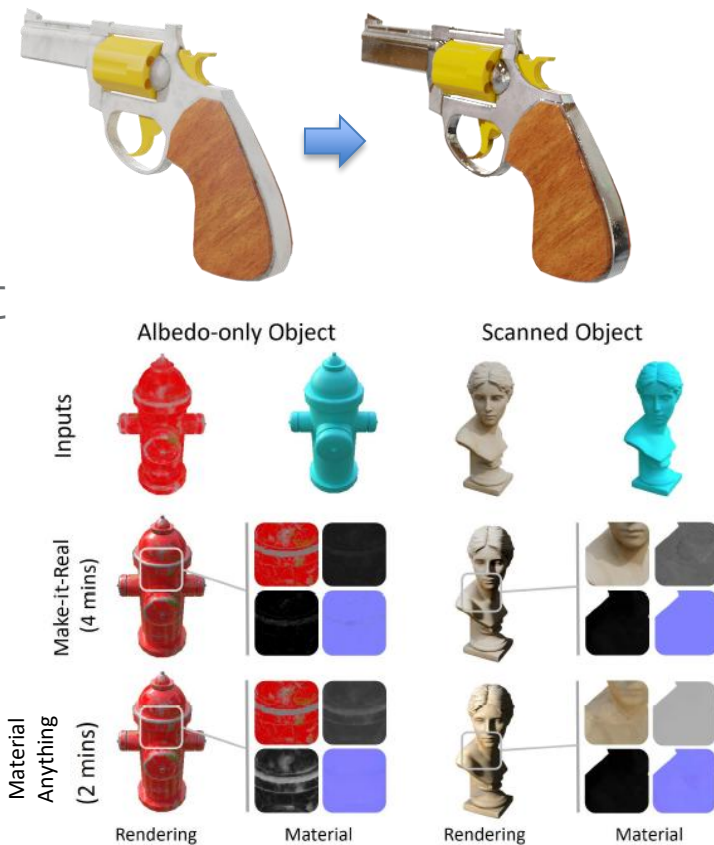
- 複数の視点からレンダリングしてパーツ分割
- 各パーツを構成するマテリアルをLLMに問い合わせ
- カラーテクスチャを参照してPBRテクスチャを生成



[16] Y. Fang et al., "Make-it-Real: Unleashing Large Multimodal Model for Painting 3D Objects with Realistic Materials", 2024

Make-it-Realの検証

- 金属のパーツと非金属のパーツにはっきり分かれた
- パーツ分割は2D画像で行っているため、視点によっては正しくない場合がある
- 同じパーツ内はすべて同じマテリアルで変化に乏しい



Make-it-RealとMaterial Anythingの比較(論文^[15]より)

- 提案手法のアプローチ
- 提案手法のワークフロー
 - テキストから3Dメッシュ生成
 - カラーテクスチャの生成
 - PBRマテリアルの生成

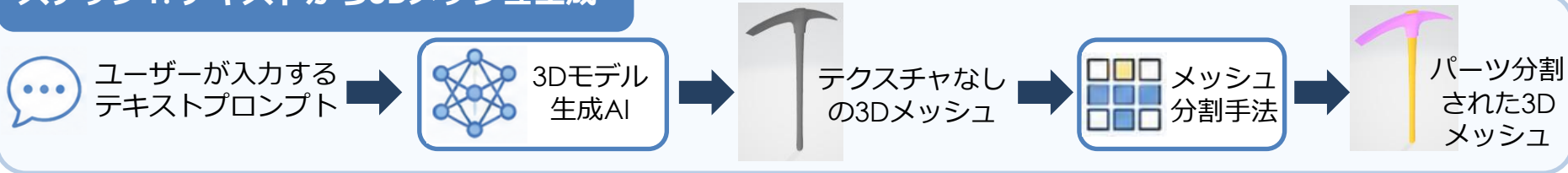
目的

ユーザの意図を反映させて扱いやすく

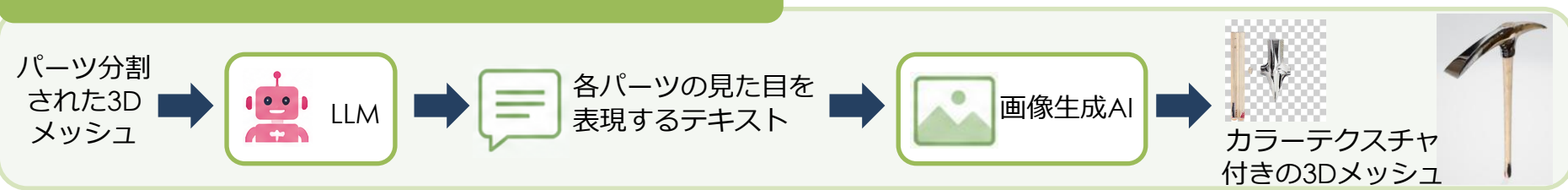
- AI生成された3Dモデルに対し、まずパーツ分割を行う
 - 各パーツについて、LLMに問い合わせ、常識的な知識を適用
 - パーツ分割が意図通りになっていない場合、ユーザがGUIで修正できる
- (テクスチャがない場合のみ)、Text2Texでテクスチャを生成
 - 各パーツについて異なるテキストプロンプトで生成し、統合する
- Make-it-RealでPBRマテリアルを生成
 - パーツに割り当てるマテリアルが意図通りになっていない場合、ユーザがGUIで修正できる

提案手法のワークフロー

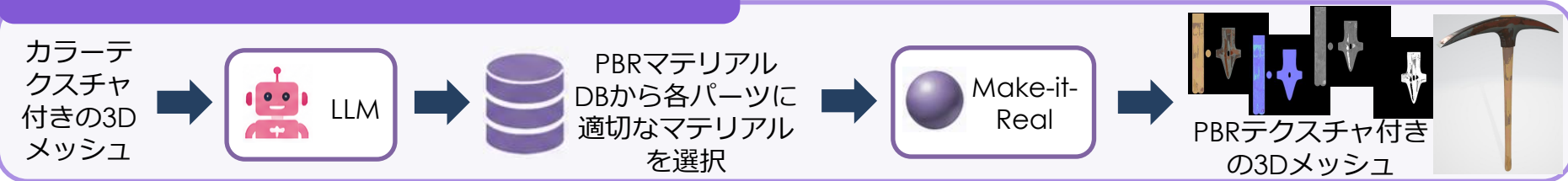
ステップ1: テキストから3Dメッシュ生成



ステップ2: カラーテクスチャ付きの3Dメッシュ生成



ステップ3: PBRマテリアル付きの3Dメッシュ生成



- メッシュ生成
 - 任意の3Dメッシュ生成サービスを選択できる
- 前処理
 - 面数削減(pymeshlab)
 - 多すぎると処理に時間がかかるため
 - 最終的に生成されたテクスチャを元のメッシュに適用すれば問題ない
 - 回転
 - +Yが上、+Zが前になるように回転
 - 物体の向きがおかしいと、AIの認識や画像生成に影響する
 - UV生成
 - UVは元のメッシュと共通にしておく

- パーツ分割に基づいてUV生成
 - BlenderやxatlasでUV展開するよりもチャートの数が少なく、歪みが少なく、パーツ単位でまとまっているUVが得られる



[17] Z. Wang et al., "PartUV: Part-Based UV Unwrapping of 3D Meshes", 2026

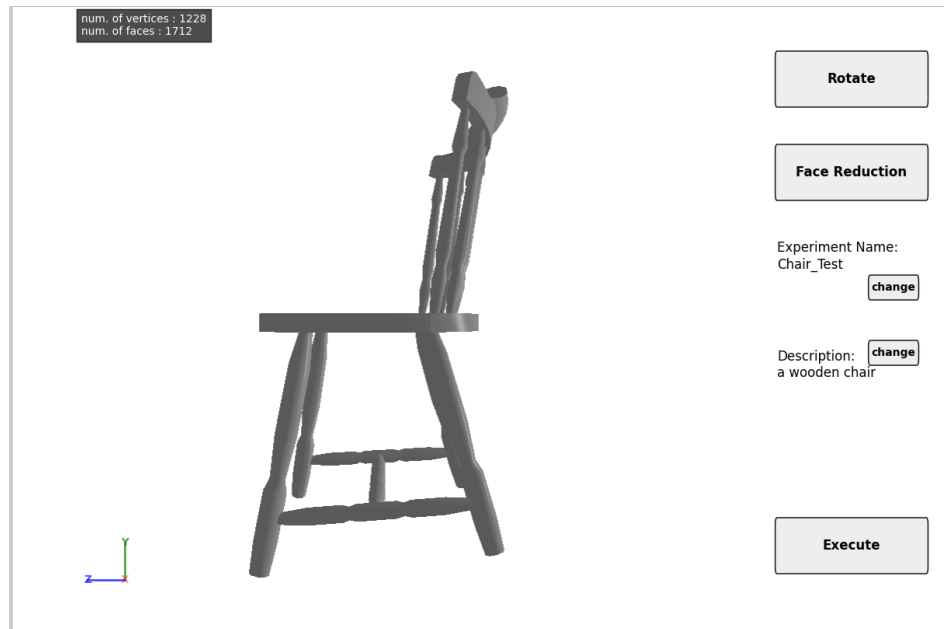
1. PartFieldの結果に基づいて2分木を作成
2. 木をルートから辿って、単一の平面に展開したときの歪みを測定
3. 歪みが閾値以上ならさらに細かく分割



[17] Z. Wang et al., "PartUV: Part-Based UV Unwrapping of 3D Meshes", 2026

初期設定GUI

- メッシュを表示
- 回転と面数削減
- 結果を保存するフォルダ名
- テキストプロンプトの指定
- パーツ分割処理の実行



- 3Dメッシュを意味のあるパーツに分割する
 - 完全自動化は難しいため、ユーザが修正できるGUIも用意する
- 試した手法
 - PartSLIP
 - PartField
 - Segment Any Mesh

- 点群データを単語リストに対応するパーツに分割
- 単語に対応する点が1つも見つけれなかったり、どの単語にも割り当てられない点がありえる



[18] M. Liu et al., "PartSLIP: Low-Shot Part Segmentation for 3D Point Clouds via Pretrained Image-Language Models", 2023

SAMesh: Segment Any Mesh^[19]

- 各面をセグメンテーションする
 - 必ずすべての面がそれぞれ、いずれか1つのセグメントに属する
 - この形式の方が扱いやすいため、こちらを採用



[19] G. Tang et al., "Segment Any Mesh", 2025

- PartUVの内部で使用されている
- 論文中の比較ではPartSLIP、SAMeshより高速かつ高精度
 - 実際に使うと、あまり意味的な分割にならない場合もあった
- 2分木で階層的に分割でき、処理時間も短いため、後述するGUIツールで活用



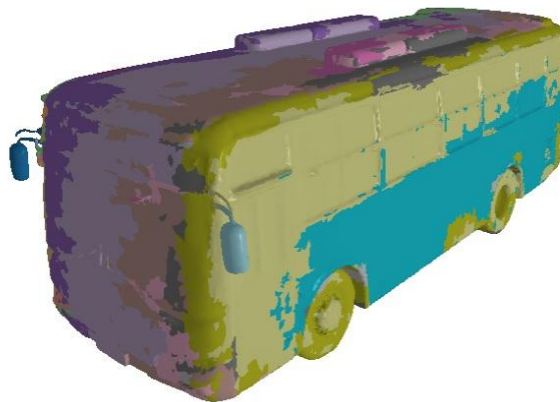
[20] M. Liu et al., "PARTFIELD: Learning 3D Feature Fields for Part Segmentation and Beyond", 2025

SAMeshとPartFieldの比較

- SAMesh

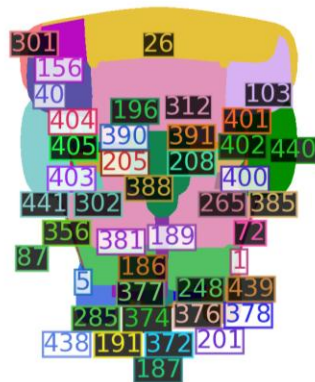
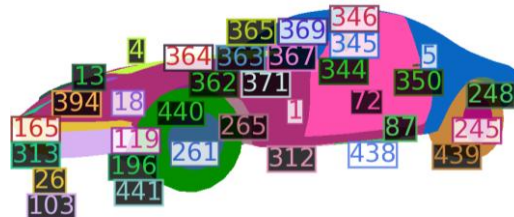
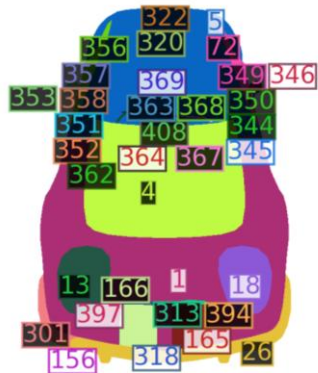


- PartField



|セグメントのマージ

- 分割されたセグメントが細かすぎる場合がある
 - セグメントの数だけLLMに問い合わせるため、コストがかかる
 - 可能な限りセグメントの数を減らす必要がある



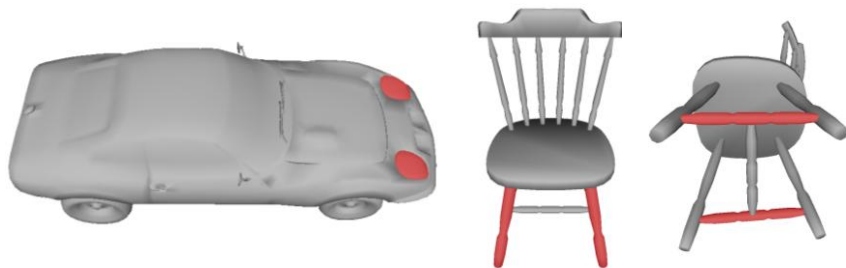
- 基準1: 左右対称に配置されたパーツ

- 世の中には左右対称の物体が多い

1. 物体全体の対称軸を検出
2. 各セグメントのバウンディングボックスから位置が左右対称な(左右反転させて重なる)組み合わせを検出
3. 片方を反転させた形状類似度を見て近ければマージ

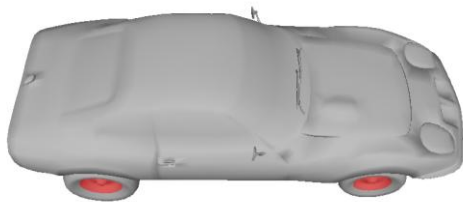
- 形状類似度

1. 点群をサンプリング
2. 点群を入力とする形状識別用のニューラルネットワーク(PointNet++^[21])に通す
3. 潜在空間ベクトルを正規化して内積を計算

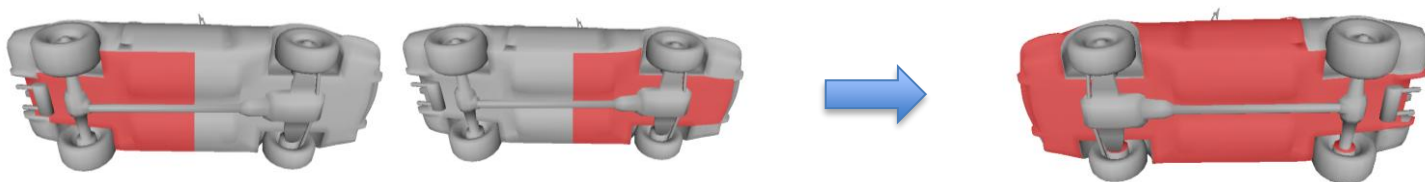
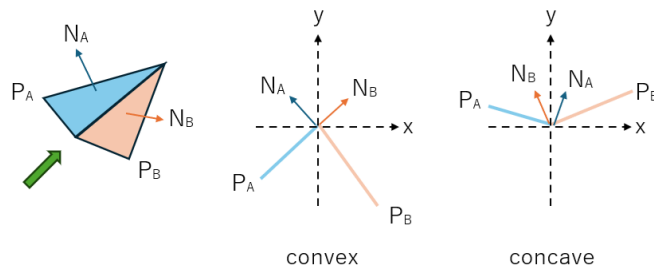


[21] C. R. Qi et al., "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space", 2017

- 基準2: 形状の似ているパーツ
 - バウンディングの大きさの比が閾値以下
 - 法線の分布をチェック
 - 平面的なセグメントは形状が似ているため、形状類似度の判定閾値を厳しくする
 - 形状類似度が閾値以上ならマージ



- 基準3: 隣接していて明確な境界のないパーツ
 - セグメント間の境界（共有辺）の左右での法線の変化から判定
 - 凹凸チェック
 - 凹面は凸面より境界になりやすいため、法線変化の判定閾値を緩くする



- 全体の構造をLLMに問い合わせ
 - 入力: 3Dメッシュを複数の視点からレンダリングした画像
 - 出力:
 - 対象物がいくつのパーツからできているか
 - 各パーツの名称と構成するマテリアルの説明



- 1. Metal Head** - A polished, shiny metallic surface with a sleek design, featuring a pointed end and a broad curved area, ideal for breaking hard surfaces.
- 2. Wooden Shaft** - A sturdy, cylindrical wooden handle with a natural, unrefined texture, providing grip and support for the user, tapering slightly towards the end.

パーツリストの作成

- 分割された各セグメントごとにLLMに問い合わせ
 - 入力: 1つのセグメントのみが色付けされた画像
 - 出力: パーツリストのうちどのパーツに属しているか
- 見えている部分が小さすぎたり、判別困難なセグメントは「不明」とする
 - テクスチャ/マテリアル生成において、「不明」の部分は通常(パーツ分けしていない状態)の結果を適用

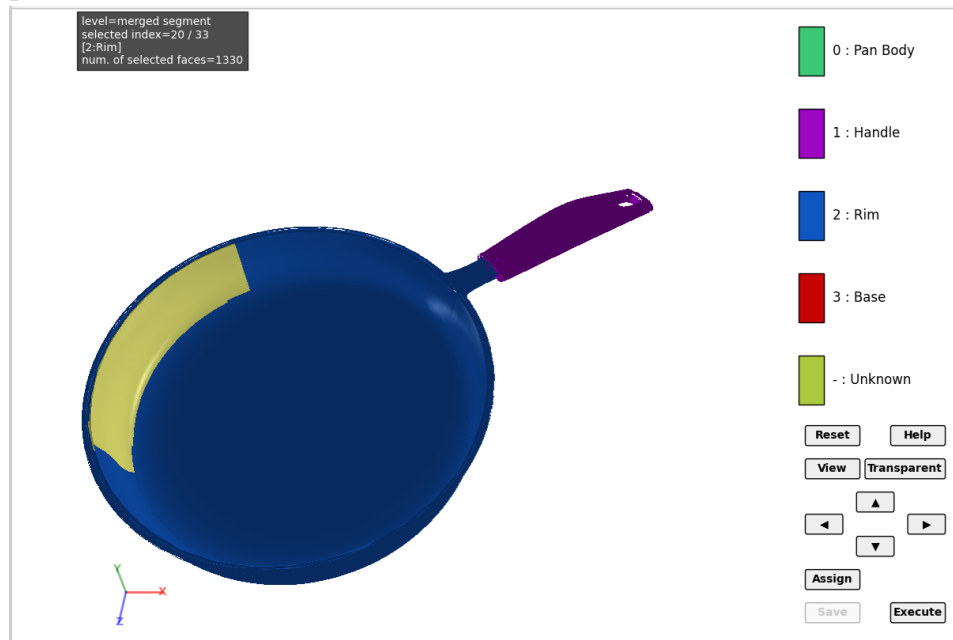


1. Metal head



2. Wooden shaft

- パーツリストはテキストファイルで保存
- クリックして選択し、パーツ番号を割り当て
- 1クリックで選択する単位を調整可能
 - パーツ
 - マージ処理後のセグメント
 - マージ処理前のセグメント
 - 選択済みの面と共通祖先ノードを持つ面
(PartFieldの2分木を活用)
 - 選択済みの面の周囲の面
 - 単一の面



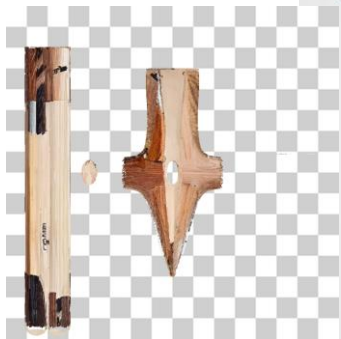
- 生成されたメッシュにカラーテクスチャがない場合のみ
- Text2Texを利用
 - 今となっては古いモデルなので、あまりクオリティは高くない
 - 他の手法に置き換え可能
- Text2Texはあまり長い文章で説明しても反映されない
 - CLIPによる単語埋め込みと対応させているだけ
 - そこで各視点において、パーツごとに画像を生成して統合

・ パーツ分けなし



途中段階における
拡散モデルの生成画像

the front view of a pickaxe
with a **metallic head** and a
wooden shaft.



最終的な生成テクスチャ

・ パーツ分け有り



the front view of a part of a
pickaxe. The part is **Metal
Head**, A polished, shiny
metallic surface with a sleek
design, featuring a pointed
end and a broad curved area,
ideal for breaking hard
surfaces.



最終的な生成テクスチャ

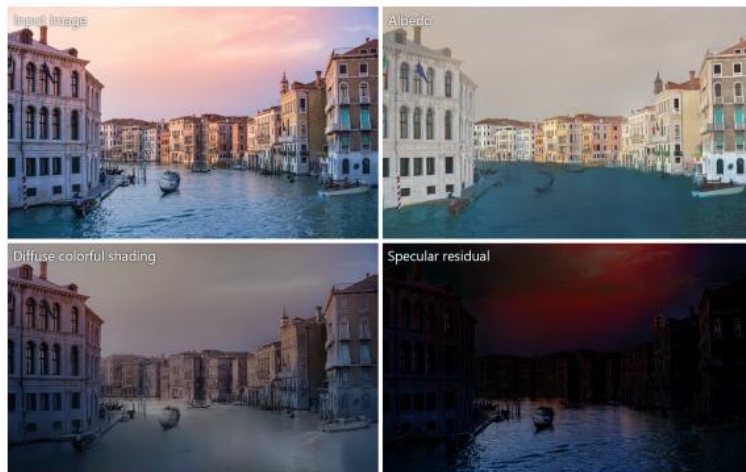


途中段階における
拡散モデルの生成画像

the front view of a part of a
pickaxe. The part is **Wooden
Shaft**, A sturdy, cylindrical
wooden handle with a natural,
unrefined texture, providing
grip and support for the user,
tapering slightly towards the
end.



- 拡散モデルによる生成画像にはライティングによる反射が含まれている可能性がある
 - Intrinsic Image Decomposition^[22]でアルベドのみを抽出
 - 見た目を維持したい場合もあるためオプションで選択可能



[22] C. Careaga and Y. Aksoy, "Colorful Diffuse Intrinsic Image Decomposition in the Wild", 2024

- Make-it-Realをベースに改造

- パーツ分割

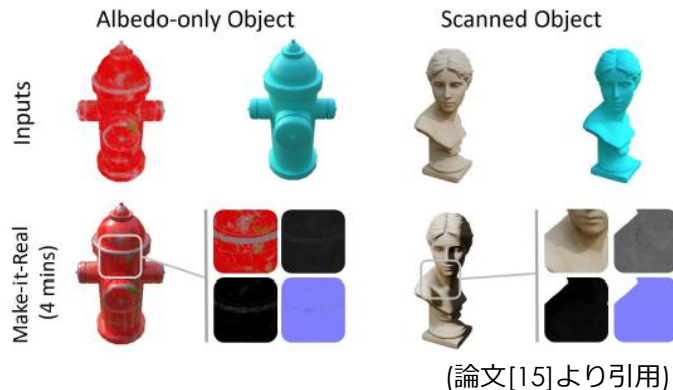
- オリジナル:レンダリング画像を2Dでセグメンテーション
 - 提案手法: あらかじめ3Dメッシュでセグメンテーションしマスク画像をレンダリング

→視点に依らず一貫したセグメンテーション

- マテリアル割り当て

- オリジナル: 1つのパーツに1マテリアル
 - 提案手法: 1つのパーツに複数のマテリアル候補

→同じパーツ内でも複数のマテリアルが混在し、複雑な表現が可能



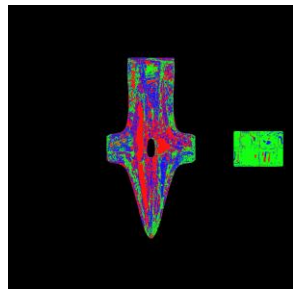
材料割り当て

- LLMへの問い合わせ時
 - 3つまで材料名を答えさせる
 - 同じパーツが異なる視点では別の材料名で答えられる場合もあるので、最終的な候補はそれ以上になる
- テクスチャの生成時
 - 候補の材料から、カラーテクスチャを参照して最も近い材料を適用

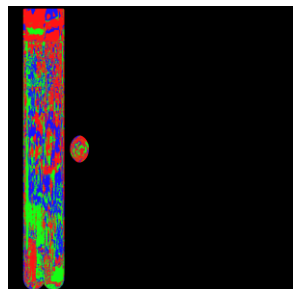


|マテリアル割り当ての詳細

- ピクセル単位で独立してマテリアル選択を行うと、1ピクセルずつバラバラになってしまう



Metal head



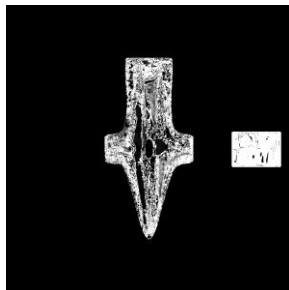
Wooden shaft



マテリアル選択結果



ベースカラー



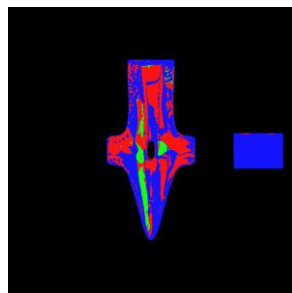
メタリック



|マテリアル割り当ての詳細

- 最初にカラーテクスチャをK-meansでクラスタリングし、各クラスタの平均色を用いてマテリアルを選択

→ある程度のまとまりになる



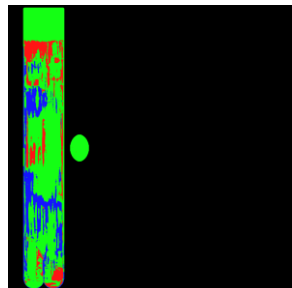
Metal head



R

G

B



Wooden shaft

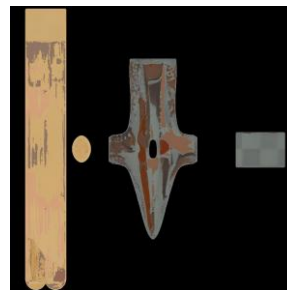


R

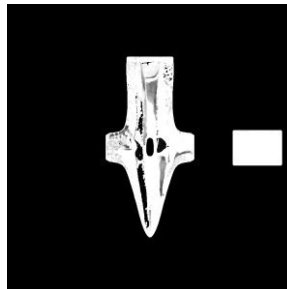
G

B

マテリアル選択結果



ベースカラー

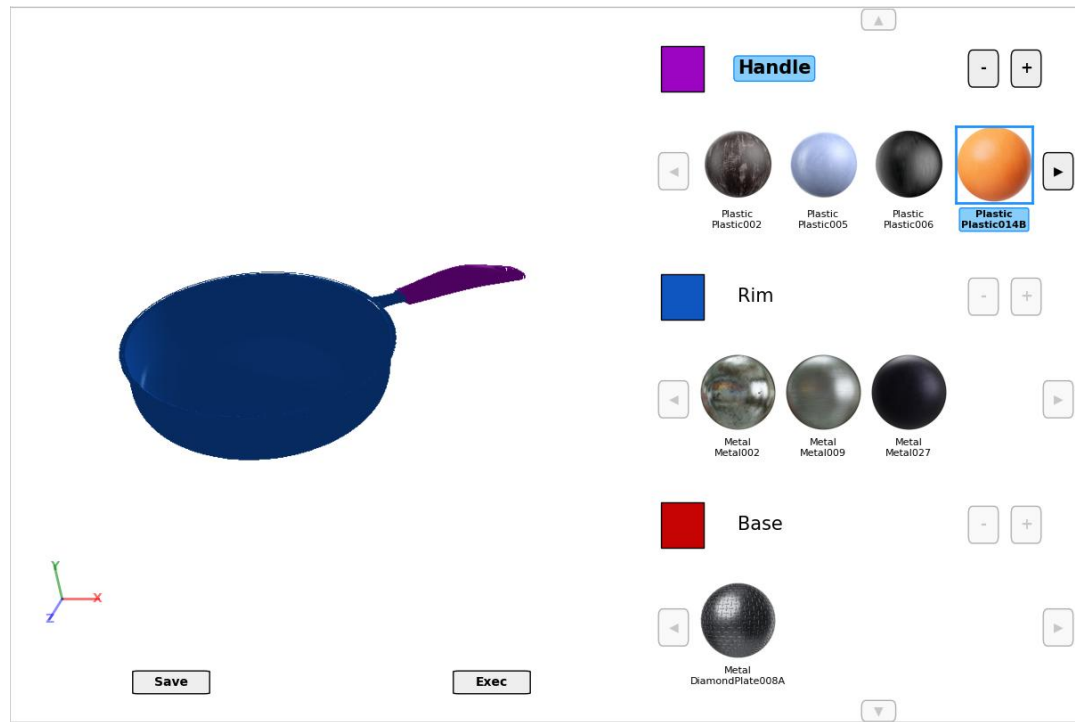


メタリック



|マテリアル割り当てGUI

- LLMによるマテリアルの選択は、問い合わせのたびに結果が変わり、ユーザの意図通りになるとは限らない
- 各パーツに割り当てるマテリアルを変更/追加/削除するツール
- Text2Texのテクスチャとは全く異なる色にする場合、最終出力のベースカラーにはマテリアルのベースカラーも選択可能



| 生成結果と今後の課題

- 生成結果
- 今後の課題

- フライパン
 - マテリアルのベースカラー



生成結果

- つるはし
 - Text2Texのテクスチャ



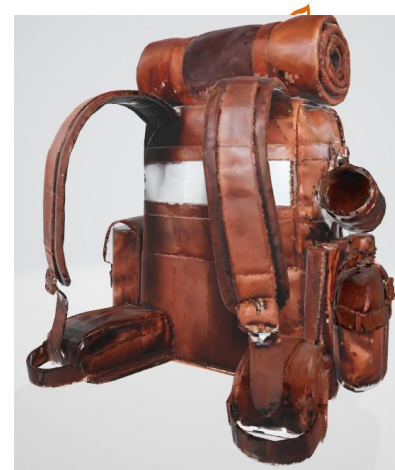
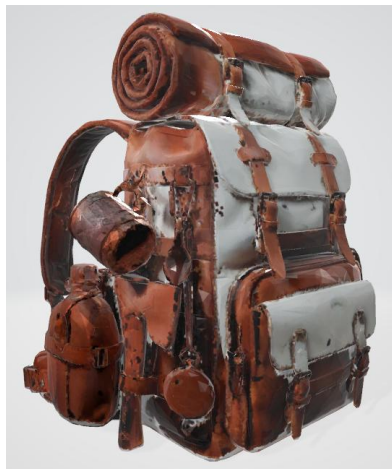
Silicon Studio
Ideas × Art × Technology

- マテリアルのベースカラー



生成結果

- バックパック
– Text2Tex



on Studio
eas × Art × Technology

- マテリアルのベースカラー



- ヘルメット
 - カラーテクスチャはメッシュと一緒に3D生成AIで出力されたもの



- トラック
 - カラーテクスチャはメッシュと一緒に3D生成AIで出力されたもの



- ガラスなど、シェーダーの異なるマテリアルへの対応
 - Make-it-RealのPBRマテリアルライブラリの追加方法
 - サムネイル画像を作成
 - マテリアルリストに名称と説明文を追加
 - LLMにサムネイル画像を見せてマテリアルの説明文を出力させる
 - 実行時には、説明文を元にLLMに選択させる
 - つまり、マテリアル選択にはシェーダーが異なっても問題ないので、最終出力テクスチャにシェーダー切り替え情報を付加するだけ
- 面数が多い場合への対応
 - 面数が多いと、PartUVなどが動作しなかったり、非常に時間がかかったりする
 - 面数を削減しすぎると、ディテールが失われ、パーツの境界が検出されなくなる
- カラーテクスチャ生成の改善
 - Text2Texは時間がかかる上に品質が低い
 - 他の生成モデルに置き換えを検討
- テクスチャパターンの反映
 - Make-it-Realではピクセル単位でマッチングするため、マテリアルライブラリのテクスチャパターンは反映されない

- 3D生成AIで生成されたメッシュを使いやすくするワークフローを提案した
 - パーツ分割
 - カラーテクスチャ生成
 - PBRマテリアル生成
- パーツ分割やマテリアル選択において、ユーザーの意図通りにならなかった場合でも、GUIから簡単に修正できる
 - ノンストップでメッシュと一緒にPBRマテリアルまで生成されるよりも意図通りの結果を得やすい
- LLMや拡散モデルは特定のモデルに依存していないため、最新モデルの性能向上とともに品質も上がっていくことが期待できる

Ideas × Art × Technology

技術力・表現力・発想力を兼ね備えたCGソリューションプロバイダー

私たちシリコンスタジオは、自社開発による数々のミドルウェアを有し、CGの黎明期から今日に至るまでCG関連事業に取り組み、技術力、表現力、発想力の研鑽を積み重ねてきました。それら3つの力を高い次元で融合させ、CGが持つ可能性を最大限に発揮させられること、それが私たちの強みです。

Ideas × Art × Technology

私たちはCG業界をリードするソリューションプロバイダーとして、お客さまの課題解決はもちろん、付加価値のあるアウトプットの提供をお約束いたします。





Ideas × Art × Technology

技術力・表現力・発想力を兼ね備えたCGソリューションプロバイダー