撮影者に向けた視覚的フィードバックを介した多層シーン画像撮影システム

安永 綾花[*1]　　斎藤 英雄[*1]　　森 尚平[*2*1]

Abstract – 多層画像 (Multi-Plane Image; MPI) 表現による自由視点画像生成は高速な描画が可能であり，深層学習を利用することにより少数視点の入力画像からでも MPI を獲得できる．先行研究では，プレノプティック・サンプリング定理に基づいて適切撮影位置で画像を入力させるために拡張現実感技術を用いて視覚的な指針となる 3 次元物体を空間中に配置する方法が取られていた．本研究では，こうした撮影対象とは直接関係のない視覚的指針に従うタスクベースの撮影方法ではなく，撮影対象そのものに注目できる視覚的フィードバックを提案・検討する．本システムは，敢えて現在得られている MPI の撮影結果の誤差を可視化することでユーザに新しい視点を挿入するかを自身で判断できる方策を取る．こうすることで，プレノプティック・サンプリング定理に頼らずともユーザが満足できる結果が得られると期待される．本稿では，視覚的フィードバックを 3 種実装し，各利点・欠点を議論する．

Keywords : **Multi-plane image, user-in-the-loop photography, augmented reality**

1　はじめに

自由視点画像生成技術はかねてから活発に研究されているコンピュータビジョンの一分野であり，深層学習を用いた手法の発達により再度注目が集まている [1]．モバイル用途を考慮した場合，より少ない入力画像枚数で高速かつ高品質に 6 自由度のカメラをサポート可能な Multi-Plane Image (MPI) は人工現実感 (Virtual Reality; VR) や拡張現実感 (Augmented Reality; AR) への応用も期待できる．しかし，どのように入力画像を用意すればいいかに関しては未だ議論の余地がある．

MPI は，奥行方向に平行に並ぶ画像群であり，新しい視点を合成する目的で，単一視点 [2] または多視点画像 [3] から生成される．1 視点分の情報ではその近傍の視点しか再構成できないため，再構成したい光景の範囲に応じて複数視点分の MPI を生成しておく必要がある．再構成したい範囲が事前に分かっている前提の元，Mildenhall らは撮影すべきカメラ位置姿勢を示した 3 次元物体を AR 空間に配置し，その物体にカメラをあてがうことで事前計算されたカメラ位置姿勢での画像撮影を行う方法を提案した [3]．この方法は平面にのみ適用可能で，光景の撮影というタスクを別のデータ収集タスクに落とし込んでいる点に注目したい．同様のデータ収集タスクは複数の研究で見られ，多視点画像撮影において主流となっている [4, 5, 6, 7]．

我々は以下の 4 点において，この方法の問題点を指摘したい．1) 撮影者は撮影結果がどのようなものになるのか想像するほかない，2) AR タスクを実行しており，目の前の撮影対象に注意が向かない，3) この AR タスクは主観的作業負荷が高い可能性が指摘されている，4) 撮影位置の事前計算はプレノプティック・サンプリング定理に基づいており実質平面上での撮影に限られる．そこで，本稿では，AR デバイス（スマートフォン）で撮影した画像 1 枚から MPI を生成するシステムを構築し，即座に AR デバイス上で MPI の描画結果が確認できるシステムを設計・構築する．

提案システムでは，撮影者自身が満足する範囲での撮影が可能であり，その撮影位置は平面上に分布している必要はない．よって，再構成品質は撮影者に依存するものの，撮影者が満足するまで視点を追加することができるという利点がある．その上で，どのように MPI を撮影者に提示すれば次に必要となる視点を撮影するように促せるか，その可視化方法については議論の余地があり，本稿にて主に検討・議論するところである．本稿での内容をまとめると以下の通りである．

- プレノプティック・サンプリング定理とその事前計算に依存しない MPI 撮影システムを提案する．つまり，撮影された MPI を特定の可視化方法で実時間で撮影者に提示する視覚的フィードバックにより，撮影者が撮影対象に注目したまま MPI によるシーンの再構築を可能にする．
- 視覚的フィードバックを 3 種類実装した上でそれぞれが撮影者に与える影響について議論し，将来の実験計画とその展望を明らかにする．

[*1] 慶應義塾大学
[*2] グラーツ工科大学

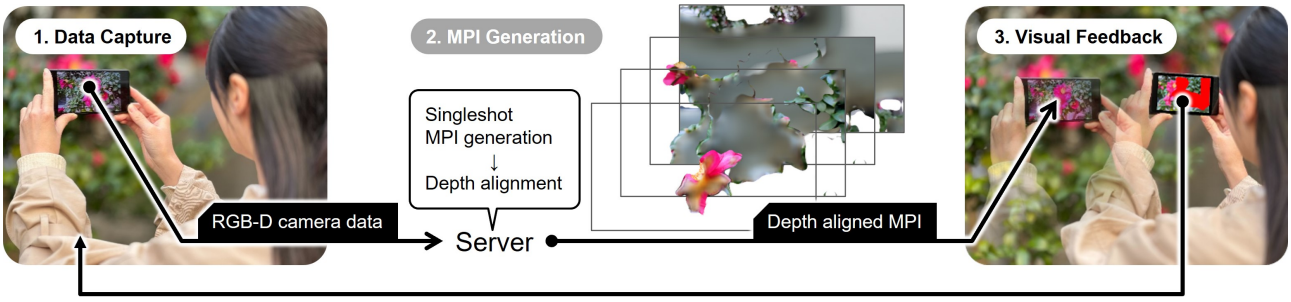

図 1　視覚的フィードバックを介した多層シーン画像撮影システム．
Fig. 1　Overview of our proposed system.

2　関連研究：MPI 生成とサンプリング定理

MPI は奥行方向に平行に並ぶ RGBα 画像群である．各画像平面を新規視点に合わせてホモグラフィ変換して重ね合わせてその視点での画像を生成する．MPI の生成には単一視点または多視点画像が必要である．本章では，単一視点あるいは多視点画像から MPI を生成する手法とそこで求められるプレノプティック・サンプリング定理の関連について述べる．

2.1　多視点画像による MPI 生成

Mildenhall ら [3] は，多視点画像を入力とした 3 次元 Convolutional Neural Network (CNN) により MPI を生成する手法，Local Light Field Fusion (LLFF) を提案した．彼らは D 層の MPI を用いることで，元の 4 次元ライトフィールドよりも $1/D$ 分の視点間隔で入力画像を撮影すればよいことを示した．その証明はプレノプティック・サンプリング定理に基づいて理論的・実験的に行われた．カメラパラメータの他，撮影対象に応じて決まる最小・最大奥行値と撮影者が撮影したい範囲から撮影間隔が導出されるため，これらのパラメータを事前に求めてシステムに入力する．求められた撮影間隔に応じて 1 章で述べた 3 次元物体を AR 空間に配置する．

Ishikawa ら [4] は，密に配置された多視点画像から MPI を生成する方法を提案した．この方法では AR 空間に配置された 3 次元平面上で無作為に撮影された画像群を用いればよいため，撮影者の主観的作業負荷が LLFF と比べて軽減される．一方で，MPI が生成される位置間隔は LLFF を踏襲している．

いずれの手法 [3][4] も撮影位置の事前計算は先のサンプリング定理に基づいており，撮影行為の AR タスク化や実質平面上での撮影に限られる欠点は避けられない．我々はこうしたサンプリング定理に基づいた議論から一旦離れ，撮影者が満足するまで視点を追加し MPI の精度を向上し続けられるシステムを目指す．

2.2　単一視点画像による MPI 生成

Tucker ら [2] は，単一視点画像からの MPI を生成する手法を最初に提案した．入力視点からは見えない遮蔽領域が視点変換後に露出するため，こうした領域は DispNet[8] を用いて埋めている．Li[9] らは，MPI と Neural Radiance Field (NeRF) の 2 つの長所を取り入れ，MINE という新しい 3 次元表現を提案した．MPI が離散的な層でしか空間を表現できないのに対し，MINE は連続的な奥行きに一般化した高密度な MPI の生成を可能にした．この 2 つの手法が MPI の層を奥行値の逆数で等間隔になるように配置するのに対し，Han ら [10] は Plane Adjustment Network を用いて層の配置をシーンに合わせて自動調整した．被遮蔽領域は Li[9] らと同様に何らかの画素値で埋める．

いずれの手法 [2][9][10] も自由視点画像生成が可能な有効範囲は 1 視点分に限られるため，更に範囲を増やすには多視点分の MPI 生成が必須である．この時，頼れるのは現在のところ LLFF によって示された視点間隔のみである．しかし，近年の研究結果で行われている被遮蔽領域を埋める操作や MPI の層の再配置により，この理論間隔を超える範囲でも自由視点画像生成が実質的に可能である．つまり，手法によって挿入するべき新規視点までの距離が異なるため，単純に MPI 用のプレノプティック・サンプリング定理を用いることが難しい．そこで，我々は，撮影者自身に新規視点挿入位置を決めさせる方法を取る．

3　提案システム

図 1 に提案システムの概要を示す．本システムでは，(1) ユーザのモバイルデバイスによる単一視点画像の撮影（3.1 節），(2) 単一視点画像からの MPI 生成（3.2 節），(3) 視覚的フィードバックの生成とユーザによる確認（3.3 節）の 3 ステップをユーザが満足のいく結果が得られるまで繰り返す．

3.1　単一視点画像の撮影

モバイルデバイスを用いて入力カラー画像 $\mathbf{I}_s$ を 1 枚だけ撮影する．この時，撮影した画像の外部・内部パラメータ，そして奥行画像 $\mathbf{D}_s$ も同時に保存する．モバイルデバイスには MPI を生成するのに用いられる CNN を動作させるだけの計算資源がないと想定し，

これらのデータはネットワークを介してより強力な計算が可能なサーバに送られる．尚，カメラの外部・内部パラメータは Unity[1]の AR Foundation[2]を，奥行画像は DepthLab[11] を用いて取得できる．

3.2 MPI の生成とスケール合わせ

ユーザが撮影した解像度 $C \times H \times W$ の単一視点画像 $\mathbf{I}_s$ と別途 AdaMPI の訓練に用いられた奥行画像生成アルゴリズム [12] から得られた奥行画像を AdaMPI[10] に入力し，D 層から成る MPI を生成する．$D \times C \times H \times W$ 画素の MPI を想定し，各層の RBG 値 $\mathbf{c}$ と密度 σ，各層を配置する奥行値 $\mathbf{z}$ を AdaMPI により出力する．

一般に，単一視点画像からの生成された MPI は $[0, 1]$ の奥行値に正規化されている．よって，現在ユーザが持つデバイスのトラッキング空間とのスケール差 s を求める必要がある．本システムでは，Tucker ら [2] のスケール普遍視点合成を参考にする（式 1）．

$$s = \exp\left[\frac{1}{|\mathbf{D}_s|}\sum_{(x,y,d)\in\mathbf{D}_s}\left(\ln\hat{\mathbf{D}}_s(x,y) - \ln(d^{-1})\right)\right]. \tag{1}$$

ここで，$\hat{\mathbf{D}}_s$ は AdaMPI から得られた MPI を奥行きチャンネルで描画することで得られる奥行画像である．

求めた奥行きのスケール差 s を用いて，ネットワークから出力された奥行値及び MPI の幅 w と高さ h をスケーリングできる．

$$z_i' = sz_i, \quad w_i' = z_i'\frac{w}{f}, \quad h_i' = z_i'\frac{h}{f}. \tag{2}$$

ここで，f は撮影に使用したモバイルデバイスのカメラの焦点距離を示し，$z_i \in \mathbf{z}$ とする．こうしてスケールを実空間と合わせた MPI，つまり，AdaMPI から得られた MPI と $\mathbf{z}'$ をユーザが持つデバイスへ転送する．モバイルデバイス上では，画像 $\mathbf{I}_s$ を撮影したカメラ位置姿勢を基準に生成した MPI をトラッキング空間に配置した上，各層は式 2 に基づいて拡大縮小する．

3.3 ユーザへの視覚的フィードバック

本稿では，モバイルデバイスにおいて以下の 3 つの視覚的フィードバックを実装する．

- **黒背景＋ MPI**：1 枚目の MPI が撮影された後，黒背景に切り替え，その上に視点に応じた MPI を重畳表示する．MPI を得るという目的のための最も単純な可視化方法と言える．MPI が見えなければ，または MPI 内に想定される画素が見えなければ追加の MPI を生成することが期待される．モバイルデバイスのトラッキング精度が低くとも大きな問題が見られないと考えられる．
- **実背景＋ MPI**：ビデオストリーミング上に視点に応じて MPI を重畳表示する．実背景と MPI との直接的な比較が可能となる．ただし，MPI が生成された視点付近ではその比較は難しくなると考えられる．
- **実背景＋誤差ハイライト**：ビデオストリーミングと視点に応じて描画された MPI との差分を計算し，一定以上の誤差が確認された画素を赤塗にする．MPI で再現しきれない画素に現れる誤差を可視化する．誤差は可視化されるものの，生成された MPI が求めるものとなっているかの判断は難しいと考えられる．

以下，それぞれの実装方法を述べる．

まず，新規視点画像を MPI を用いて描画するため，MPI 撮影視点から新規視点に MPI の各層をホモグラフィ変換する．新規視点内のある画素における i 層の MPI の RGB 値と密度値を $\mathbf{c}_i$ と σ_i として，以下の様に画素値 $\hat{\mathbf{c}}$ と対応する透過度 $\hat{\alpha}$ を計算する．

$$\hat{\mathbf{c}} = \sum_{i=1}^{D}\left(\mathbf{c}_i\alpha_i\prod_{j=1}^{i-1}(1-\alpha_j)\right),$$
$$\hat{\alpha} = \sum_{i=1}^{D}\left(\alpha_i\prod_{j=1}^{i-1}(1-\alpha_j)\right).$$

ここで，$\alpha_i = \exp(-\delta_i\sigma_i)$，$\delta_i$ は i 層目と $i+1$ 層目間の距離とする．

黒背景＋ MPI:最終結果 $\mathbf{c}'$ は，背景色 $\mathbf{c}_{\text{blk}} = (0,0,0)^\intercal$ と $\mathbf{c}'$ とのアルファ合成結果となる．

$$\mathbf{c}' = \hat{\alpha}\hat{\mathbf{c}} + (1-\hat{\alpha})\mathbf{c}_{\text{blk}}. \tag{3}$$

実背景＋MPI：式 4 における $\mathbf{c}_{\text{blk}}$ を現在ストリーミングされている実背景の画素値 $\mathbf{c}_{\text{vid}}$ に置き換える．

$$\mathbf{c}' = \hat{\alpha}\hat{\mathbf{c}} + (1-\hat{\alpha})\mathbf{c}_{\text{vid}}. \tag{4}$$

実背景＋誤差ハイライト：実背景と MPI による描画結果が異なる画素に関して，事前に指定しておいたハイライト用の色を表示する．それ以外は現在ストリーミングされている実背景の画素とする．

$$\mathbf{c}' = \begin{cases}\mathbf{c}_{\text{err}}, & \text{if } L(\hat{\mathbf{c}}, \mathbf{c}_{\text{vid}}) > t\\ \mathbf{c}_{\text{vid}}, & \text{otherwise.}\end{cases} \tag{5}$$

ここで，$\mathbf{c}_{\text{err}}$ は事前に指定する誤差ハイライト用の色，$L(\cdot)$ は入力される画素値間の距離を計算する関数，t は関数 $L(\cdot)$ による閾値とする．

[1]Unity Technologies，Unity，https://unity.com/, アクセス日:2023 年 12 月 17 日

[2]Unity, Unity の AR Foundation フレームワーク,https://unity.com/ja/unity/features/arfoundation, アクセス日:2023 年 12 月 17 日

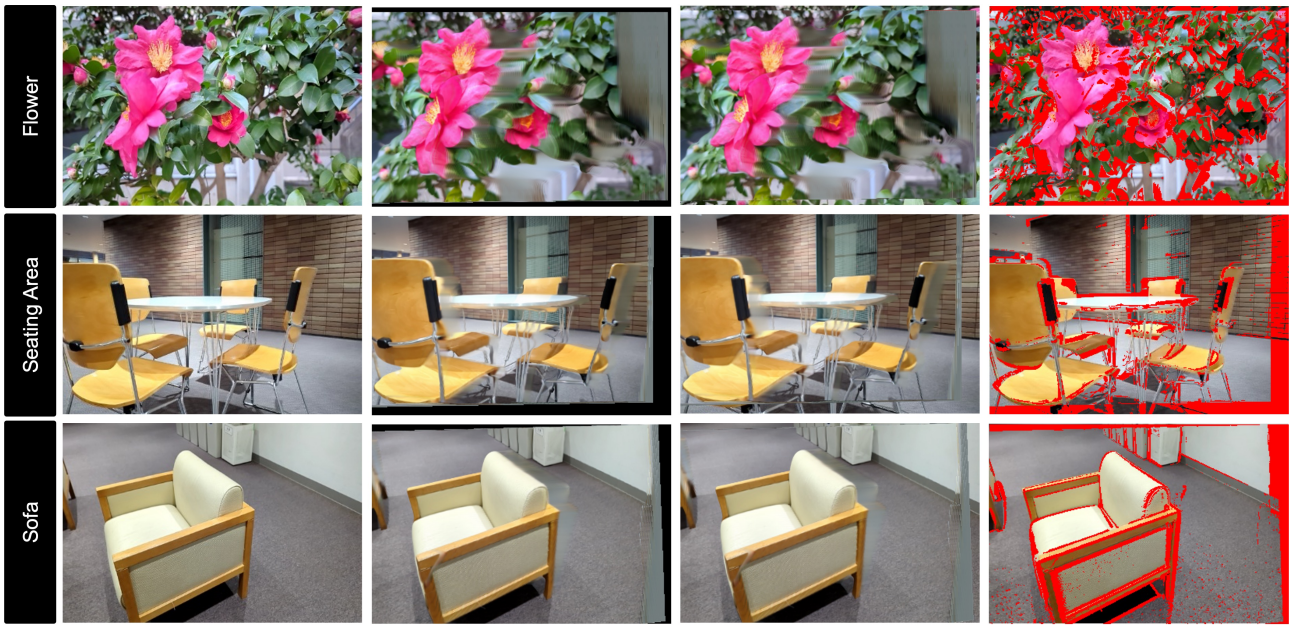

図 2　可視化方法の定性的比較.
Fig. 2　Qualitative comparison of visualization method.

4　システムの実装と試行実験

単一視点での多層シーン画像生成における不確実性を撮影者に伝えるため，ある MPI の視点を動かすと，どのようなアーティファクトが見えるか確認する．また，次の視点位置を効果的に伝えるための可視化方法について議論する．

4.1　システム実装の詳細

モバイルデバイスには Google Pixel 6 を用いた．ソフトウェアは Unity 及び C#と Shader Lab で実装し，デバイスのトラッキングには AR Foundation を用いた．サーバには Intel Core i7 3.3 GHz CPU，32 GB RAM，NVIDIA GeForce RTX 2080 GPU 11GB を搭載したデスクトップ PC を用いた．

両者は File Transfer Protocol (FTP) を用いてデータをやり取りする．つまり，モバイルデバイスで撮影された 1 視点分のデータはファイルとしてサーバの特定のフォルダにアップロードされ，サーバはそれを監視する．サーバは，データのアップロードが完了した際に MPI を生成する（3.2 節）．別のフォルダに生成された MPI が保存されるため，クライアントであるモバイルデバイスはそのフォルダに MPI が生成されるのを確認してダウンロードする．

MPI 生成には AdaMPI[3]を用い，スケール合わせを行う処理は別途 Python にて実装した．AdaMPI には事前学習された重みを用いた．そのため，カメラは 384 × 256 画素で水平画角 81.78679° とした．

[3]https://github.com/yxuhan/AdaMPI

4.2　試験用データ収集

左右 2 視点分の情報を 1 組にした実画像対を 3 シーンで撮影した（図 2a）．保存した 2 視点の内 1 視点を選択し，その視点の MPI を生成した．生成した MPI をもう一方を視点にて 3.3 節で議論した 3 つの可視化方法で表示した．

4.3　結果と考察

図 2 に，生成した MPI をもう一方の視点にて描画した結果を各可視化方法について示す．

黒背景+MPI（図 2b）は，MPI が生成されていない画素は黒塗されている．そのため，撮影者に再構成されていない範囲を伝え，MPI の視点追加を促すことができる．一方で，第 2 視点において現れる被遮蔽領域の画素は MPI 生成時に埋められた色で表示される．こうした被遮蔽領域はぼやけた色で表示され，これらを埋めるために明示的に新たな視点を挿入することを促すのは難しいと考えられる．

実背景+MPI（図 2c）では，視点に応じて実背景に MPI が重畳表示され，実背景と MPI の直接的な比較が可能となる．ただし，MPI が生成された視点付近では実背景と MPI の重畳度が高く，撮影者に比較の差異を提示することは難しい．

実背景+誤差ハイライト（図 2d）では，実背景と MPI の間に一定以上の誤差が確認された画素は赤塗され，撮影者に MPI で再現しきれなかった範囲を効果的に伝えることができる．ただし，黒背景+MPI（図 2b Flower の中心）で引き伸ばされたアーティファクトが発生している範囲に対し，実背景+誤差ハイライト

（図 2d Flower の中心）では適切にハイライトがされていない．故に，撮影者に生成された MPI が求めるものになっているかの判断を促すことは難しい．また，被遮蔽領域と初期視点での画角外の領域とを区別するすべがなく，被遮蔽領域が納得いく結果であったとしても視点の追加を促してしまう可能性がある．従って，MPI に内在する非遮蔽領域の画素値を特定し，実背景と MPI の誤差をより明瞭に可視化して撮影者に提示することも考えられる．

5 技術的限界

本稿では，撮影者に向けた視覚的フィードバックを介した多層シーン画像撮影システムを紹介した．本研究は初期段階であり，現行のシステムには改善すべき点が残っている．

モバイルデバイスの描画性能：本提案システムでは Shader を用いて MPI の描画を実装しており，MPI の描画速度が主にモバイルデバイスの GPU 性能に依存する．ボリューム描画を行う AdaMPI の MPI を用いる我々の場合，現状 8 Hz 程度で動作している．

モバイルデバイスのトラッキング性能:モバイルデバイスのトラッキング性能には限界がある．アプリケーション起動時に頻繁に発生するカメラの絶対的な位置姿勢の更新により撮影を続けることが困難になる場合がある．主にバンドル調整による全体最適化によるものと考えられるが，事前によくデバイスを動かしてシーンを再構築しておくか，LiDAR センサといったレンジファインダに頼る方法が考えられる．

安定したトラッキング下でも精度が十分とは言えず，画像上では MPI がずれて表示されることがある．実背景＋誤差ハイライトはトラッキング精度の影響を多大に受ける．MPI に内在する非遮蔽領域の画素値を特定してハイライト表示する場合，現在のビデオストリーミングとは無関係に誤差表示が実現できる．

6 むすび

本稿では，多層画像によるシーン再構成過程を撮影者に伝える視覚的フィードバックを提示するシステムを提案した．従来の多視点画像による MPI 生成はプレノプティック・サンプリング定理に基づくことから平面上での撮影に限られ，単一視点画像による MPI 生成は有効範囲が位置視点分に限られるという課題があった．そこで，我々は，現在得られている MPI の撮影結果の誤差を可視化し，ユーザに新しい視点を挿入するかを自身で判断させる方策に基づいたシステムを目指し試作した．本稿では，視覚的フィードバックを 3 種類実装し，MPI を生成する初期視点とその描画先である第 2 視点を組にして 3 シーン分のデータを用いて試行した．

次の段階として，今回得られた知見を基に視覚的フィードバックを改善するとともにインタラクティブ・システムを実装する．完成したシステムを用いて被験者実験を行い，実際にどの可視化方法が好まれるか確認したい．

謝辞

本研究は Austrian Science Fund FWF (No. P33634) の助成を受けたものである．

参考文献

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, and R. Ramamoorthi, and R. Ng, “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” Proc. ECCV, 2020.

[2] R. Tucker and N. Snavely, “Single-view View Synthesis with Multiplane Images,” Proc. CVPR, 2020.

[3] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng and A. Kar, “Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines,” ACM TOG, 2019.

[4] R. Ishikawa, H. Saito, D. Kalkofen and S. Mori, “Multi-Layer Scene Representation from Composed Focal Stacks,” IEEE TVCG, Vol. 29, No. 11, pp. 4719–4729, 2023.

[5] P. Mohr, S. Mori, T. Langlotz, B. H. Thomas, D. Schmalstieg, and D. Kalkofen, “Mixed Reality Light Fields for Interactive Remote Assistance,” Proc. ACM CHI, 2020.

[6] A. Davis, M. Levoy, and F. Durand, “Unstructured Light Fields,” Proc. Eurographics, 2012.

[7] C. Birklbauer and O. Bimber, “Active Guidance for Light-Field Photography on Smartphones,” Computers & Graphics, Vol. 53, Part B, pp. 127–135, 2015.

[8] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, “A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation,” Proc. CVPR, 2016.

[9] J. Li, Z. Feng, Q. She, H. Ding, C. Wang, and G. H. Lee, “MINE: Towards Continuous Depth MPI with NeRF for Novel View Synthesis,” Proc. IEEE/CVF ICCV, 2021.

[10] Y. Han, R. Wang, and J. Yang, “Single-view View Synthesis in the Wild with Learned Adaptive Multiplane Images,” Proc. ACM SIGGRAPH, 2022.

[11] R. Du, E. Turner, M. Dzitsiuk, L. Prasso, I. Duarte, J. Dourgarian, J. Afonso, J. Pascoal, J. Gladstone, N. Cruces, S. Izadi, A. Kowdle, K. Tsotsos, and D. Kim, “DepthLab: Real-time 3D Interaction with Depth Maps for Mobile Augmented Reality,” Proc. ACM UIST, 2020.

[12] R. Ranftl, A. Bochkovskiy，and V. Koltun, “Vision Transformers for Dense Prediction,” Proc. IEEE/CVF ICCV, pp. 12159–12168, 2021.