Point Cloud Completion for Mixed Reality

Arvid Berg
ETH Zurich
bergar@student.ethz.ch

Lukas Schüepp
ETH Zurich
lukaschu@student.ethz.ch

Simon Schläpfer
ETH Zurich
simschla@student.ethz.ch

Abstract

Mixed Reality applications, such as Dynamics 365 Guides, can help to improve productivity with holographic instructions. These instructions can be generated using point cloud data captured from an instructor's perspective while performing a specific task. However, this initial representation is partial and lacks completeness from alternative viewpoints. Therefore, in this project, we propose to leverage learning-based algorithms to deliver more complete representations. To achieve this, we train an encoder-decoder transformer-based model to predict completed point clouds constructed from multi-view given its corresponding point cloud from a single-view. We show that our approach can perform with an accuracy of 97% when evaluated on the Percentage of Correct Keypoints metric within a threshold radius of 0.1 normalized distance. In addition, we present a complete pipeline that anchors the location, records, segments, predicts, and integrates completed point clouds into the Mixed Reality environment, allowing for the replay of the recorded scene with the completed point cloud.

1. Introduction

The accurate representation of human actions is key for interactive experiences. The precise depiction of human actions, among these the movements of hands and hand actions, can highly improve instructional contexts [7]. Technologies like Microsoft HoloLens 2 [15] opened new avenues in this field, offering the capability to capture and render 3D information while users are interacting with objects.

The HoloLens 2, equipped with RGB and depth cameras, captures point cloud data. Point clouds, a 3D data format, are often used in 3D modeling due to their simple structure and the fact that they can be easily captured by sensors. Yet, these captures are constrained to single-view point clouds, leading to incomplete representations of the dynamic and complex nature of hands and hand actions.

Figure 1. Scene from an instructional video, where the processed point cloud generated through our pipeline is showcased.

This limitation hinders the potential of Mixed Reality applications in delivering fully realized instructional content since the single-view point clouds fail to provide a holistic view of hand actions from different perspectives. Consequently, reconstructing complete point clouds from incomplete data has become a crucial task. For example, when a teacher (user) has captured a sequence, it might be advantageous for students to observe the captured sequence from different angles.

In this project, we aim to develop a solution that generates complete hand point clouds in MR environments from the initial incomplete, non-holistic captures of the HoloLens 2. The essence of our approach lies in the development of a robust learning-based algorithm, capable of transforming the partial, single-view hand point clouds into complete, holistic representations. To achieve this, we propose the adaptation of a state-of-the-art encoder-decoder model AdaPoinTr [18], fine-tuned on a dataset of complete hand point cloud representations. Our proposed solution predicts 97% of the points correctly within a threshold radius of 0.1 normalized distance (hands are normalized to a length of 1) when evaluated on the Percentage of Correct Keypoints (PCK) metric.

Key to our development was the acquisition of training data to fine-tune the model. For that, our dataset should contain broad variations of point cloud data representing hand movements and hand poses from different perspectives. For that, we utilized two different datasets which we preprocessed to obtain both partial and complete point clouds of the corresponding hands. Appropriate data was instrumental in training our model, allowing it to learn from many diverse hand representations.

Furthermore, we implement an application which handles the complete pipeline including capturing a scene from an egocentric view with HoloLens 2 and is capable of replaying the processed sequence back on the same device. Initially, the point cloud of the entire scene is captured, containing partial information about the hands from a single perspective. These partial hand point clouds are then segmented from the entire scene using Meta's Segment Anything model [10] to isolate the points representing the hand from the remaining points. The segmented partial hand point clouds are then further processed by removing points using outlier thresholding based on the camera's pose and using Open3D's statistical outlier removal method [21]. Finally, the remaining points are centered, normalized, and passed as input to our neural network, which predicts completed hand point clouds. The predicted complete hand point clouds are then transformed back to their original pose. Lastly, we can replay the modified scene on HoloLens 2, ensuring it maintains its relevance to the instruction's location through the use of a QR code. Testing our applications with users, we received positive feedback. Users expressed admiration for the completed point cloud's quality and its accurate spatial positioning within the room.

2. Related Work

Deep learning methods have notably revolutionized the field of 3D computer vision, and alongside the problem of point cloud completion has seen significant improvements. Such methods mainly learn unordered representations by analyzing the inartistic 3D structure. A comprehensive review of deep learning-based 3D point cloud completion [5] points out that despite substantial progress, enhancing the quality of completed point clouds for practical use remains a challenge. This review analyzes various deep learning approaches, including view-based, convolution-based, graph-based, generative model-based, and transformer-based methods. Models like AdaPoinTr [18] have emerged, surpassing previous state-of-the-art methods like Point Completion Network (PCN) [20].

Many approaches for the reconstruction of hands are found in 3D Hand pose estimation and 3D Hand shape estimation [8] [3]. In 3D Hand pose estimation, recent work tried to estimate hand poses of an RGB image with additional modalities like depth [1]. Other approaches [6], [11] involve transforming depth maps to generate an incomplete point cloud from the perspective of the camera to help predict 3D pose. 3D hand shape estimation, usually represented as meshes, requires the prediction of both mesh topology and vertex positions. Many methods [9] [14] reconstruct the mesh by applying the parametric MANO model [13].

Utilizing recent advancements, our project adopts an approach using solely hand point cloud completion from 3D Lidar images, not requiring any hand pose or hand shape estimator.

3. Methods

3.1. Pipeline Overview

The objective of this study was to develop a pipeline capable of capturing depth images using a real-world device like the Hololens 2, extracting point clouds from these depth images, and then enhancing the set of point clouds. Finally, the augmented set of point clouds should be redeployed on the device.
To establish a comprehensive pipeline, several key steps must be taken into account. Initially, the depth images are not in the required format essential for the point cloud completion model. Furthermore, our focus shifts to point cloud completion for hands. Inherent to this, the recorded depth images need to undergo appropriate segmentation to ensure that the surrounding environment does not influence the point cloud. Moreover, normalization and centralization processes are employed, followed by a set of augmentations before feeding the point cloud into the completion model. The subsequent subsections will provide detailed insights into each specific step.

3.2. Training Datasets

We aim to capture and predict point clouds that represent 3D information in the form of $(x, y, z)^{\mathcal{W}}$, where $\mathcal{W}$ corresponds to a predetermined world coordinate frame. The MVHand dataset, which will be introduced in this section, and the Hololens 2 provide depth images, which encode information as $(p_x, p_y, z^{\mathcal{C}})$. Here, p_x and p_y are natural numbers corresponding to the pixel position, and z determines the depth with respect to a local camera frame denoted by $\mathcal{C}$. Therefore, accessing both intrinsic and extrinsic parameters becomes essential to transform the pixel position p_x or p_y and the depth $z^{\mathcal{C}}$ into the requested 3D point cloud information with respect to the desired world frame $\mathcal{W}$. Based on

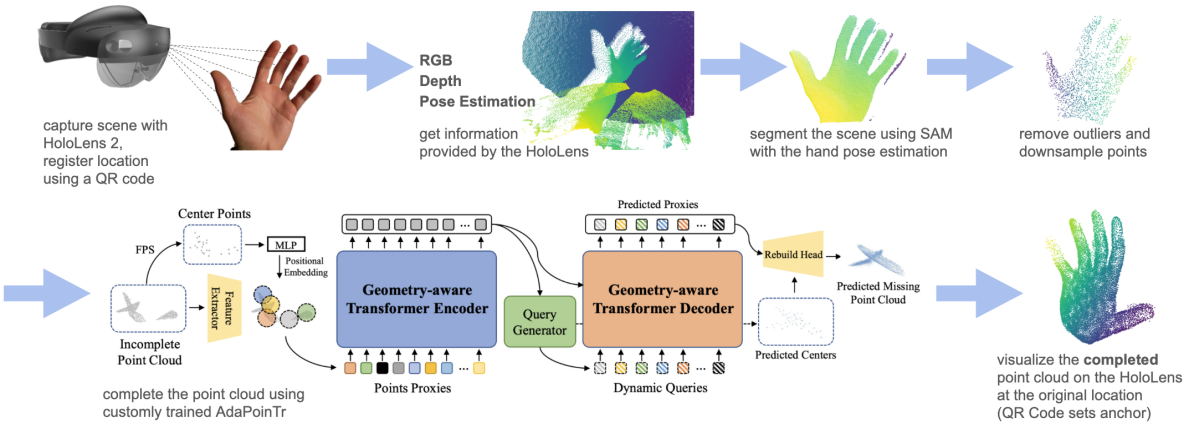

Figure 2. Overview of the complete pipeline, including a visualization of the AdaPoinTr model [18].

the triangulation principle, the position in the local camera frame can be determined

$$x^{\mathcal{C}} = z^{\mathcal{C}} \cdot \frac{p_x \cdot \alpha_x - c_x}{f_x} \tag{1}$$

$$y^{\mathcal{C}} = z^{\mathcal{C}} \cdot \frac{p_y \cdot \alpha_y - c_y}{f_y} \tag{2}$$

Here, f represents the focal length, α corresponds to the pixel size, and c determines the center position of the local camera frame with respect to the upper-left corner. Given the position in the local camera frame, it is common to employ a translation followed by a rotation, resulting in a homogeneous transformation, to obtain the 3D points in a world coordinate frame

$$\mathbf{r}^{\mathcal{W}} = T_{\mathcal{WC}} \cdot \mathbf{r}^{\mathcal{C}} \tag{3}$$

Here, $\mathbf{r} = (x, y, z, 1)$ represents the position, and $T_{\mathcal{WC}}$ is the homogeneous transformation from the coordinate frame $\mathcal{C}$ to the frame $\mathcal{W}$.

Another method for constructing data involves using a set of MANO parameters [13]. MANO operates by parametrizing a triangular mesh, where these parameters encapsulate the shape characteristics and pose estimations of the hand. Through this fitted mesh, we were able to derive a collection of 3D points that form the hand's surface.

3.2.1 MVHand

The MVHand dataset [19] comprises 21,200 frames, each featuring a single hand. Every frame includes four distinct depth images of the same scene, captured from four different perspectives. Segmentation masks for the hands are also provided. These masks can be overlaid onto the depth images in order to extract the hand-specific depth pixels. Applying the aforementioned transformations enables the alignment of the four segmented point clouds into the same world coordinate frame. Additionally, the dataset includes MANO parameters, allowing for the reconstruction of MANO hands [13].

For our approach, a dual strategy was adopted, utilizing the MANO-generated hand cloud as the ground truth and depth camera-derived point clouds from a single view as inputs. Choosing the noisy point cloud from the depth image is reasonable, considering that, in the use case, noisy depth images captured by the HoloLens 2 constitute the input data.

3.2.2 FreiHand

FreiHand [2] is the second hand dataset which was used. In comparison to MVHand, FreiHand provides no depth images and relies solely on MANO Parameters which we used to create partial point clouds by rendering the synthetic hand from four viewpoints and removing all hidden points.

In this manner, utilizing both datasets enabled the generation of 215k pairs of partial and complete point clouds.

3.2.3 Data Preparation and Augmentation

Given a set of points $(x_i, y_i, z_i)^{\mathcal{W}}_{i=1:n}$, a preliminary step involved normalizing and centralizing the data. Due to the geometric nature of the data, this step is crucial for generalizing between different point clouds. It is imperative to retain the centralization and normalization parameters such that, after the prediction step, the original scale and position can be accurately recovered.

In order to augment the dataset comprehensively and mitigate the risk of overfitting, five distinct data augmentation techniques were incorporated: scaling, flipping, rotation, shifting (complete point cloud), and shifting (individual points). This implementation ensures that the model can handle different input perspectives, right and left hands with various sizes.

3.3. Model - AdaPoinTr

For point cloud completion, we utilized the AdaPoinTr model [18], an enhancement of the previously introduced Pointr model [17]. AdaPoinTr takes an innovative approach to point cloud completion by framing it as a translation task from one set to another. It employs a transformer encoder-decoder architecture to generate comprehensive point clouds. The encoder incorporates multi-head self-attention layers to refine proxy features by combining both long-range and short-range information. Meanwhile, the decoder uses self-attention and cross-attention mechanisms to gain structural insights. Additionally, AdaPoinTr integrates a geometry-aware block to leverage the inherent 3D geometric properties of point clouds. The last step takes the decoder's output, the predicted proxies, and uses a fully connected layer to predict the final output. We incorporate some additional ideas that were introduced in [19], substituting the last fully connected layer with a folding net [16] or a hand template-specific folding net [19], respectively. Essentially the model takes a set of normalized and centralized points $(\bar{x}_i, \bar{y}_i, \bar{z}_i)_{i=1:n}$ and predicts a set of points $(\bar{x}_j, \bar{y}_j, \bar{z}_j)_{j=1:k}$ that represent the completed point cloud. We set n to 2048 and k to 16384.

3.3.1 FoldingNet

A further enhancement includes substituting a folding net introduced in [16], instead of a fully connected layer. The underlying idea is that several objects are based on flat surfaces. Accordingly, one could use a set of transformations on the surface area of the desired objects, to recover a flat surface. Based on this idea, a set of fixed 2D grid points is concatenated onto the decoder output before proceeding with fully connected layers in a similar manner as before.

3.3.2 Hand Template

While using a folding net has shown to be useful for abstract figures such as tables and chairs [18], this idea could be rather difficult to generalize for complex and diverse structures such as hands. To address this challenge for hands, we considered using a hand template instead of a set of fixed 2D grid points, as introduced in [19]. Again, a grid is concatenated onto the decoder's output, followed by a set of fully connected layers.

3.4. Deployment on HoloLens 2

We used the Hololens 2 device in a user case scenario to record data. During training, segmentation masks were provided within the data sets to extract depth values corresponding to the hand. Hololens 2 does not provide such masks, necessitating the offline segmentation of hands. Further processing steps were implemented to handle outliers post-segmentation. An application was developed on the Hololens 2 device for recording and replaying point clouds.

3.4.1 Segmentation and Further Processing

Accurate segmentation is crucial for the model's performance. When processing recorded data offline, a fast and automated workflow for segmenting hands from each frame is desirable. In our case, we opted to use the well-established segmentation model SAM [10]. While SAM is a powerful tool, it requires a set of points to identify the regions of interest in the image. Fortunately, many devices, such as the Hololens 2, provide hand pose estimation. These hand pose estimations can serve as input, along with the corresponding RGB image, to the SAM model.

Given a set of pixel positions assumed to represent relevant points on the hand, one could easily use these as identifiers for the parts to be segmented. After constructing masks that determine the hand's position in the RGB image, we were able to overlay the mask onto the depth image, resulting in a set of points $(p_x, p_y, z^{\mathcal{C}})_{i=1:n}$, assumed to be the points of interest.

After post-hand segmentation, numerous of outliers can occur. Such outliers may arise from inaccuracies in the mask provided by SAM or directly from the HoloLens depth sensor. To address this issue, a two-step outlier removal process is implemented. Initially, we calculate the distance of the hand relative to the camera and eliminate outliers falling outside the ± 15 cm range. Subsequently, Open3D's [21] outlier removal function is applied with parameters `nb_neighbors=20` and `std_ratio=0.8`. The resulting point cloud is then downsampled to the specified number of points, which, in our case, is 2048.

3.4.2 Application

For deployment, we built an application in Unity, which in a first step is able to save the recorded depth-, RGB-images and hand pose estimation. The latter ones are then used in an offline post-processing step to segment the hands, while the depth image is then used for accurate 3D point cloud completion. In a second step, the application is able to replay the completed point cloud. In many cases, one wishes to replay the point cloud at a specified location. As an example, one can record a video of using a printer. When a different person wants to replay the completed point cloud, the point cloud should at best be replayed at the same location where it was recorded with respect to the printer. To achieve this, it was decided to make use of a QR code which can serve as an anchor.

Model	Chamfer Distance L1
PCN	42.49
AdaPoinTr - FC	**33.80**
AdaPoinTr - FoldingNet	38.42
AdaPoinTr - Hand Template	38.29

Table 1. Performance comparison of different models based on Chamfer Distance L1 (scaled with a factor of 1000). The following models were used: PCN [20] and AdaPoinTr with various reconstruction heads (fully connected layer (FC), FoldingNet, Hand Template).

4. Results

The results section is structured in two parts. First, we examine the point cloud completion algorithm's performance on the test set and real-world data. Following this, the second section highlights the presentation of the final HoloLens application.

4.1. Point Cloud Completion

4.1.1 Training

The proposed models of Sec. 3.3 were trained from scratch on the dataset presented in Sec. 3.2 with the training method provided by the AdaPoinTr framework [18]. The training was done on GPUs of the ETH Euler cluster and took around 7 hours until full convergence. As a loss, the Chamfer Distance L1 was used [4].

4.1.2 Performance on Test Set

To evaluate the performance of the trained model we used a data set split of the MVHand data set [19] as a test set.

Two metrics were employed for performance evaluation. The Chamfer Distance [4] and the Percentage of Correct Keypoints [12] , wherein all metrics and distances underwent normalization, as the assessments were conducted on the normalized dataset. The proximity of predicted points to ground truth points was quantified using the Chamfer Distance metric [4].

While the Chamfer Distance serves as a useful metric for overall prediction quality, it solely represents the mean distance error, lacking insight into the percentage of correct points. To address this limitation, we adopted the Percentage of Correct Keypoints (PCK) metric, commonly employed in pose estimation scenarios [12]. Given the absence of joint predictions in our case, we assessed each predicted point's proximity to a ground truth point within a specified threshold. An illustration of points within (blue) and outside (red) the threshold is depicted in Fig. 3.

Table 1 shows the Chamfer Distance on the test data from the different models introduced in Sec. 3.3. Additionally, the results of using the PCN model [20], one of the first promising deep learning-based point cloud completion models, are also displayed. Evaluating the best-performing model (AdaPoinTr-FC, Tab. 1) across various thresholds (normalized radius) revealed that at a radius of 0.1, 97% of predicted points fall within the defined threshold, as demonstrated in Fig. 4.

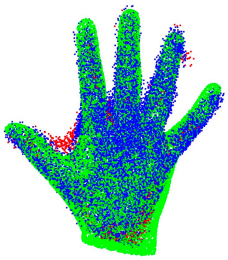

Figure 3. Visualization of PCK@0.05. Green: ground-truth points, Blue: predicted points within the threshold, Red: predicted points outside of the threshold.

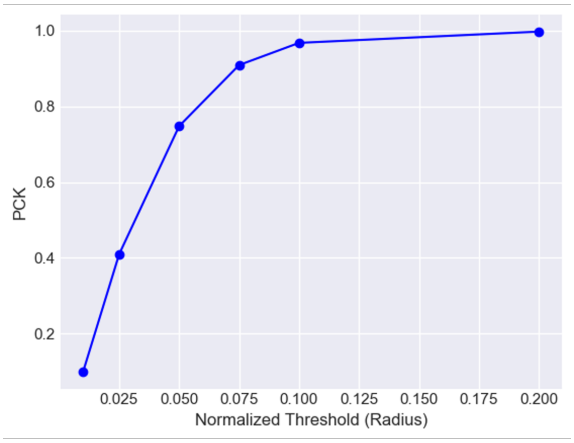

Figure 4. AdaPoinTr-FC PCK at different thresholds.

4.1.3 Performance on Real Data

The significant domain gap between the synthetic training/test data and real-world data poses a challenge, particularly due to the prevalence of synthetic hand data in a major portion of our dataset. To address this, we implemented extensive data augmentation strategies, as discussed in Sec. 3.2.3, aiming to enhance generalization. In the data preprocessing stage detailed in Sec. 3.4.1 for the Hololens-captured information , the data underwent segmentation using SAM [10], outlier thresholding based on camera pose, and the removal of statistical outliers through Open3D's method [21].

Given the absence of ground-truth data for Hololens-captured information, traditional metrics like Chamfer Distance or PCK were impractical. Consequently, the evaluation of predictions was solely conducted visually, with examples presented in Fig. 5.

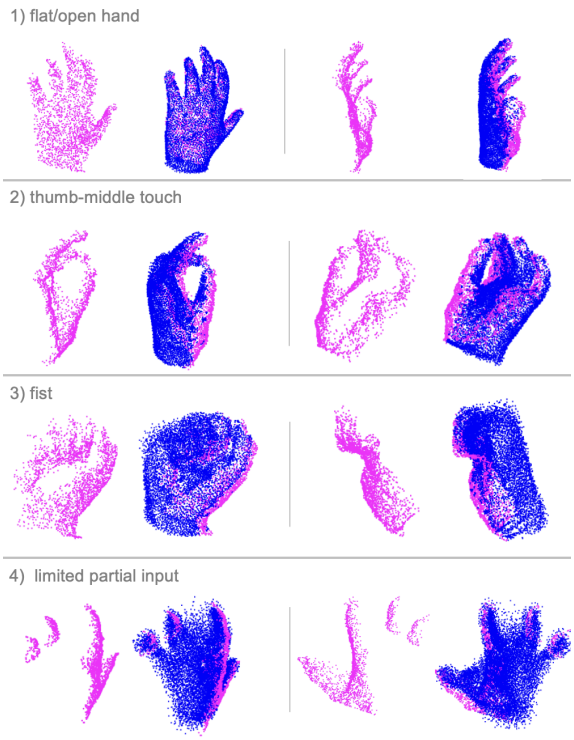

Figure 5. Visual results of real-world data captured by the HoloLens, rendered from two different views. Pink: input (partial point cloud), Blue: prediction (completed point cloud).

4.2. Results on HoloLens

The results were tested on Hololens 2. The application features two main buttons in a holographic interface responsible for either recording new data or playing a processed and completed point cloud, as shown in Fig. 6. The recorded data is downloaded manually and processed. After the point cloud completion steps, the resulting point cloud can be uploaded onto the device again. A QR code acts as an anchor, capturing the pose of the object. Upon detecting a QR code, the play button displays the corresponding point cloud aligned with the QR code's position and orientation. Fig. 1 illustrates a typical scene where the completed point cloud interacts with a printer.

5. Discussion

In general, the model not only showed satisfactory predictive outcomes on the test dataset, as shown in Fig. 4 by the PCK, but also proved to handle various hand poses of real-world inputs obtained from HoloLens recordings (Fig. 5). This observation showcases that our data set and proposed data augmentation is able to generalize across different data sources and show robustness across various input scenarios.

It's worth noting that certain predictions are noisy. The emergence of noise, particularly pronounced in scenarios

Figure 6. The application's interface. Below is the button which allows someone to record new data. The above panel can be used for replaying a processed point cloud. As soon as a QR code is detected, one can press next or previous to choose from a set of completed point clouds each associated with a title and a description.

where partial point clouds exhibit substantial missing parts (like in Sample 4 of Fig. 5), introduces ambiguity due to the lack of informative data in these regions.

However, investigating why AdaPoinTr, when coupled with a FoldingNet or the Hand Template as a reconstruction head demonstrated inferior performance compared to its counterpart with a fully connected layer, presents an interesting area for further exploration. The intricate and diverse nature of hand poses and shapes may contribute to the observed differences.

5.1. Extensions

One possible extension involves expanding the scope to accommodate scenes with two hands interacting with objects. This adjustment would contribute to a more realistic simulation of real-world interactions, allowing the model to adapt and generalize better to diverse hand-object scenarios.

An additional avenue for improvement lies in leveraging RGB data, which is available alongside point cloud information. By integrating color information into the point cloud, the instructional content would look more realistic.

Furthermore, instead of using the QR Code method to set the anchor, the Azure Object Anchor (AOA) could be used. This would make the application even more seamless as there is no need for a distinct QR Code on each object.

5.2. User Study

Upon testing our applications with users, we received positive feedback, with users expressing admiration for the completed point cloud's quality and its accurate spatial positioning within the room. Additionally, users highlighted

that 4D tutorials featuring completed instead of partial hands would help them understand and utilize specific machines.

However, some users expressed a desire for a higher frame rate to enhance the overall experience. Additionally, there was interest in a direct comparison with partial point clouds, allowing users to observe how our model completes the point cloud. Another noteworthy suggestion involved providing a denser point cloud for better visibility, yet this was constrained by the computational limitations of the HoloLens.

6. Conclusion

This project introduces a pipeline for capturing and playing back instructional content in Mixed Reality. To ensure an authentic user experience, the captured point cloud of the interacting hand is completed to form a complete 3D hand shape. This highlights the capability of advanced learning-based models like AdaPoinTr to complete point clouds even of complex shapes, such as hands, when trained on appropriate data.

Expanding our framework to handle not only hands but also objects that are interacted with has the potential to enhance the realism of instructional content even further. This project establishes a solid foundation for future works to complete point clouds for instructional content on mixed reality devices such as the HoloLens 2.

References

[1] Yujun Cai, Liuhao Ge, Jianfei Cai, and Junsong Yuan. Weakly-supervised 3d hand pose estimation from monocular rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[2] Jimei Yang Bryan Russel Max Argus Christian Zimmermann, Duygu Ceylan and Thomas Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 3

[3] Martin de La Gorce, David J. Fleet, and Nikos Paragios. Model-based 3d hand pose estimation from monocular video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1793–1805, 2011. 2

[4] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. *CoRR*, abs/1612.00603, 2016. 5

[5] Ben Fei, Weidong Yang, Wen-Ming Chen, Zhijun Li, Yikang Li, Tao Ma, Xing Hu, and Lipeng Ma. Comprehensive review of deep learning-based 3d point cloud completion processing and analysis. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):22862–22883, 2022. 2

[6] Liuhao Ge, Zhou Ren, and Junsong Yuan. Point-to-point regression pointnet for 3d hand pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[7] C.E. Hughes, C.B. Stapleton, D.E. Hughes, and E.M. Smith. Mixed reality in education, entertainment, and training. *IEEE Computer Graphics and Applications*, 25(6):24–30, 2005. 1

[8] Leyla Khaleghi, Alireza Sepas-Moghaddam, Joshua Marshall, and Ali Etemad. Multiview video-based 3-d hand pose estimation. *IEEE Transactions on Artificial Intelligence*, 4 (4):896–909, 2023. 2

[9] Sameh Khamis, Jonathan Taylor, Jamie Shotton, Cem Keskin, Shahram Izadi, and Andrew Fitzgibbon. Learning an efficient model of hand shape variation from depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[10] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 2, 4, 5

[11] Shile Li and Dongheui Lee. Point-to-pose voting based hand pose estimation using residual permutation equivariant layer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[12] Dushyant Mehta, Helge Rhodin, Dan Casas, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation using transfer learning and improved CNN supervision. *CoRR*, abs/1611.09813, 2016. 5

[13] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), 2017. 2, 3

[14] David Joseph Tan, Thomas Cashman, Jonathan Taylor, Andrew Fitzgibbon, Daniel Tarlow, Sameh Khamis, Shahram Izadi, and Jamie Shotton. Fits like a glove: Rapid and reliable hand shape personalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[15] Dorin Ungureanu, Federica Bogo, Silvano Galliani, Pooja Sama, Xin Duan, Casey Meekhof, Jan Stühmer, Thomas J. Cashman, Bugra Tekin, Johannes L. Schönberger, Pawel Olszta, and Marc Pollefeys. Hololens 2 research mode as a tool for computer vision research, 2020. 1

[16] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation, 2018. 4

[17] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers, 2021. 4

[18] Xumin Yu, Yongming Rao, Ziyi Wang, Jiwen Lu, and Jie Zhou. Adapointr: Diverse point cloud completion with adaptive geometry-aware transformers, 2023. 1, 2, 3, 4, 5

[19] Ziwei Yu, Linlin Yang, Shicheng Chen, and Angela Yao. Local and global point cloud reconstruction for 3d hand pose estimation, 2021. 3, 4, 5

[20] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network, 2019. 2, 5

[21] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 2, 4, 5