Monocular 3D Hand Pose Estimation with Implicit Camera Alignment

Christos Pantazopoulos[1*] Spyridon Thermos[2] Gerasimos Potamianos[1]
[1]Department of Electrical and Computer Engineering, University of Thessaly, Greece
[2]Moverse
cpantazop@uth.gr spiros@moverse.ai gpotam@ieee.org

Abstract

Estimating the 3D hand articulation from a single color image is an important problem with applications in Augmented Reality (AR), Virtual Reality (VR), Human-Computer Interaction (HCI), and robotics. Apart from the absence of depth information, occlusions, articulation complexity, and the need for camera parameters knowledge pose additional challenges. In this work, we propose an optimization pipeline for estimating the 3D hand articulation from 2D keypoint input, which includes a keypoint alignment step and a fingertip loss to overcome the need to know or estimate the camera parameters. We evaluate our approach on the EgoDexter and Dexter+Object benchmarks to showcase that it performs competitively with the state-of-the-art, while also demonstrating its robustness when processing "in-the-wild" images without any prior camera knowledge. Our quantitative analysis highlights the sensitivity of the 2D keypoint estimation accuracy, despite the use of hand priors. Code is available at the project page https://cpantazop.github.io/HandRepo/

1. Introduction

Reconstructing an articulated 3D hand from a single RGB image is a challenging problem in computer vision with a wide range of applications, including augmented and virtual reality (AR/VR), human-computer interaction (HCI), robotics, and the metaverse. However, it is a challenging task due to the lack of depth information, frequent object-related occlusions and self-occlusions, unknown camera intrinsics parameters, and the hand's complex articulation.

In this paper, we propose an alternative method for monocular 3D hand pose estimation that operates without prior camera parameters knowledge. Our approach leverages the robust 2D keypoint detection capabilities of MediaPipe [1], combined with a two-stage optimization pipeline that fits the MANO [26] hand model to the detected 2D keypoints. The first stage performs a rigid transformation to align the initial MANO hand model with the 2D detections, establishing a coarse 3D pose estimation. The second stage refines this estimate using a fingertip alignment loss and anatomical constraints to ensure physically plausible hand configurations.

By avoiding reliance on known camera parameters, our method is able to perform in-the-wild while maintaining accuracy. The anatomical constraints operate as a regularizer, preventing unrealistic hand poses, while the fingertip alignment loss improves precision in critical regions. We demonstrate that our approach achieves competitive performance compared to state-of-the-art (SotA) methods, even without knowing the camera intrinsic parameters, making it a practical solution for real-world applications.

Our key contributions include:

1. A camera-agnostic 3D hand pose estimation framework that leverages 2D keypoint detections.
2. A two-stage optimization pipeline combining rigid alignment and refinement with anatomical and fingertip constraints.
3. An extensive evaluation showcasing the method's robustness in-the-wild and its competitive performance on standard benchmarks.

The rest of the paper is organized as follows: a) the prior work in 2D and 3D hand pose estimation and the relevant parametric models are discussed in Sec. 2; a more detailed background of the leveraged modules is presented in Sec. 3; our method is detailed in Sec. 4; the experimental framework is presented in Sec. 5; and finally our conclusions are reported in Sec. 6

2. Prior Art

Hand pose estimation refers to predicting the position and orientation of the hand and fingers in relation to a set coordinate system using either RGB images, volumetric data

*This work is part of the author's diploma thesis

Figure 1. The 21 hand keypoints estimated by MediaPipe [1].

from depth cameras, or a combination of both. This paper focuses on implementations that exploit solely the color information of the human hand.

2D hand pose estimation. In 2D hand pose estimation, to estimate the pose of the hand we need to estimate the location of its keypoints. The human hand has 21 keypoints [10]: In Fig. 1 we depict the keypoints on a hand. In prior work on this topic, in [27] Simon *et al*. present an approach that uses a multi-camera system to train fine-grained detectors for keypoints that are prone to occlusion, such as the joints of a hand. This procedure is called multiview bootstrapping. It uses an initial keypoint detector to generate noisy labels across multiple views, triangulates valid detections in 3D, and reprojects them as new training data to iteratively improve the detector. This process yields a real-time RGB keypoint detector with accuracy comparable to depth-based methods.

3D Hand Pose Estimation. There have been numerous methods estimating 3D pose using depth or multi-view sensors. However, regressing pose from a single RGB image is challenging due to the fact that 3D pose requires some form of depth estimates, which are ambiguous given only an RGB image. This was later dealt with the use of parametric hand models like we do in our proposed method. Iqbal *et al*. [15] propose a new method for 3D hand pose estimation from a monocular image through a 2.5D pose representation. Zimmermann and Brox [34] also present an approach that estimates 3D hand pose from RGB images. To handle the "missing" depth data they propose a deep network that learns a network-implicit 3D articulation prior. Together with detected keypoints in the images, this network yields good estimates of the 3D pose. Additionally, Zimmermann *et al*. [35] introduce a large-scale 3D hand pose dataset based on synthetic hand models for training the involved networks.

3D Hand Pose Estimation using Parametric Hand Models. A stepping stone to the evolution of the 3D hand pose estimation from a single RGB image without the use of depth information or other cameras/sensors has been the introduction of hand parametric models like MANO [26].

Boukhayma *et al*. [9] present the first end-to-end deep learning based method that predicts both 3D hand shape and pose from RGB images in the wild. This network consists of the concatenation of a deep convolutional encoder and a fixed model-based decoder. Panteleris *et al*. [24] present a method for the real-time estimation of the full 3D pose of one or more human hands using a single commodity RGB camera. More specifically, given an RGB image and the relevant camera calibration information, they employ a SotA detector to localize hands. Then, using a crop of a hand in the image, they run the pretrained network of OpenPose [27] for hands to estimate the 2D location of hand joints. Finally, non-linear least-squares minimization fits a 3D model of the hand, distinct from the MANO model, to the estimated 2D joint positions, recovering the 3D hand pose. Mueller *et al*. [20] address the problem of real-time 3D hand tracking based on a monocular RGB-only sequence. Their method combines a CNN with a kinematic 3D hand model. For training this CNN they generated a synthetic training dataset by using a neural network that translates synthetic images to "real" images, such that the so-generated images follow the same statistical distribution as real-world hand images. Ge *et al*. [12] propose a Graph CNN based method to reconstruct a full 3D mesh of hand surface that contains richer information of both 3D hand shape and pose. Baek *et al*. [8] also adopt the MANO parametric 3D hand model. To achieve the model fitting to RGB images they implement a hand mesh estimator by a neural network and a differentiable renderer, supervised by 2D segmentation masks and 3D skeletons.

Kulon *et al*. [16] introduce a simple and effective network architecture for monocular 3D hand pose estimation consisting of an image encoder followed by a mesh convolutional decoder that is trained through a direct 3D hand mesh reconstruction loss. They train the network by gathering a large-scale dataset of hand action in YouTube videos and use it as a source of weak supervision. Zhang *et al*. [32] present a Hand Mesh Recovery framework to tackle the problem of reconstructing the full 3D mesh of a human hand from a single RGB image. The mesh representation is achieved by parameterizing MANO. To this end, a differentiable re-projection loss is defined in terms of the derived MANO representations and the ground-truth labels, thus making this framework end-to-end trainable. Drosakis and Argyros [11] present a method for simultaneous 3D hand shape and pose estimation on a single RGB image frame. Specifically, their method fits the MANO 3D hand model to 2D hand keypoints, based on a 2D objective function that exploits anatomical joint limits, combined with shape regularization.

Lim *et al*. [17] present an approach for real-time estimation of 3D hand shape and pose from a single RGB image, using an efficient CNN named MobileNetV3-Small to extract key features from an input image. The extracted fea-

tures are then sent to an iterative 3D regression module to infer camera parameters, hand shapes, and joint angles for projecting and articulating a 3D hand model.

3. Background

MANO - A 3D Hand Parametric Model.

MANO is a parametric model commonly used in computer vision and graphics to encode the shape and pose variations of human hands. The MANO hand model takes as input 45 rotation parameters θ and 10 shape parameters β to produce a 3D hand mesh. Once the model - denoted as Φ - has been loaded properly, it can be initialized by defining the following variables:

$$\Phi\{\beta, \theta, r\} \quad \text{with} \quad \beta \in \mathbb{R}^{10}, \theta \in \mathbb{R}^{45}, r \in \mathbb{R}^{3}, \quad (1)$$

which correspond to the beta, pose, and global orientation parameters, respectively. The shape parameters, denoted as β, control the overall structure of the hand, such as the width of the fingers and palm. These parameters are derived through PCA on real hand scans, with only 10 parameters required to represent nearly all human hand shape variations. On the other hand, the pose parameters, denoted as θ, define the rotations of the hand's joints, allowing for various articulated poses. These rotations are represented in an axis-angle format for the 15 major joints of the hand, leading to a total of 45 values. Each model implementing MANO hand has different but similar functions available to obtain the joints, faces and vertices of the mesh in order to later visualize it.

Our implementation is based on MANOTorch [2], a differentiable PyTorch layer that deterministically maps MANO's pose and shape parameters to hand joints and vertices using PyTorch.

MediaPipe. The first step in our method is to detect and localize the 21 keypoints of the hand in the input image. This can be achieved using specialized frameworks that serve as keypoint encoders, such as MediaPipe [1], OpenPose [5], and MMPose [4]. We chose to use the MediaPipe Hand Landmarker [19] that takes image data as input and outputs hand landmarks in image coordinates, hand landmarks in world coordinates, and handedness (left/right hand) of multiple detected hands. The keypoint coordinates provided by MediaPipe are normalized and scaled between 0 and 1, so we denormalize them using the width and height of the image.

Optimization Methods. The BFGS (Broyden-Fletcher-Goldfarb-Shanno) algorithm is an iterative optimization method used to solve unconstrained optimization problems [23]. Limited-memory BFGS (L-BFGS) [22] is a Quasi-Newton optimization method that builds upon the BFGS algorithm while significantly reducing memory usage.

Loss Functions. The Mean Squared Error (MSE) [3] loss computes the average squared difference between the predicted values $\hat{y}_i$ and the true values y_i:

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

where n is the number of data points.

The Geman-McClure (GM) loss function is a robust alternative to the MSE loss, designed to mitigate the influence of outliers. It is defined as:

$$L_{\text{GM}}(r) = \frac{\rho^2 r^2}{r^2 + \rho^2}, \quad (3)$$

where r represents the residual error (*i.e.*, the difference between predicted and true values), and ρ is a parameter that controls the sensitivity of the function to large residuals. It was originally introduced in the context of tomographic image reconstruction [13], and has since been applied in various domains, including the 3D human pose estimation [25].

The Huber loss [14] combines the MSE and MAE behavior and is defined as:

$$L_\delta(a) = \begin{cases} \frac{1}{2}\ a^2 & \text{for } |a| \le \delta, \\ \delta\left(|a| - \frac{1}{2}\delta\right) & \text{otherwise,} \end{cases} \quad (4)$$

where $a = y - f(x)$.

Let us now introduce a more specific loss constraint that is directly related to hand articulation and our problem. The **anatomical joint limits error** E_{limits} is a penalty term applied to the 45-dimensional pose vector, where each joint has three DoF. This error function ensures that estimated hand poses remain within experimentally determined anatomical constraints, thereby enforcing plausible hand articulations. Similar to [11], this loss is implemented as a soft constraint using exponential functions that activate when joint angles exceed predefined limits:

$$E_{\text{limits}}(\boldsymbol{\theta}) = a_{\text{limits}} \sum_{i=0}^{m} \left(e^{l_i - \theta_i} + e^{\theta_i - u_i}\right), \quad (5)$$

where $[l_i, u_i]$ denote the lower and upper bounds for joint angle θ_i, and a_{limits} is an experimentally determined weight factor. The exponential terms enforce smooth constraints that discourage hand poses outside the allowable range. The concept of anatomical constraints for hand motion was first introduced by Lin *et al.* [18] and has since been adopted by various works, including [11], [30], and others.

Rigid Transformations. A rigid transformation [6] involves applying a rotation, translation, and optionally scaling to align two sets of points while preserving their internal geometric relationships. A rigid transformation consists of two main operations:

- Translation: A shift of an object in 3D space along the x, y, and z axes without altering its orientation.
- Rotation: A transformation that changes the orientation of an object while preserving its shape and size.

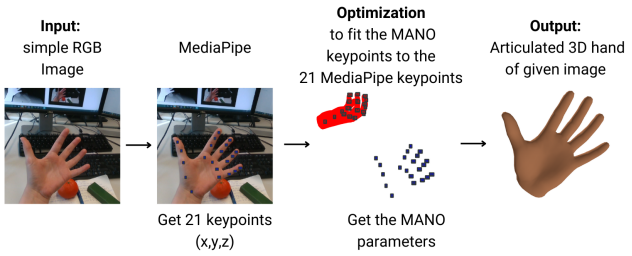


Figure 2. Our proposed pipeline.

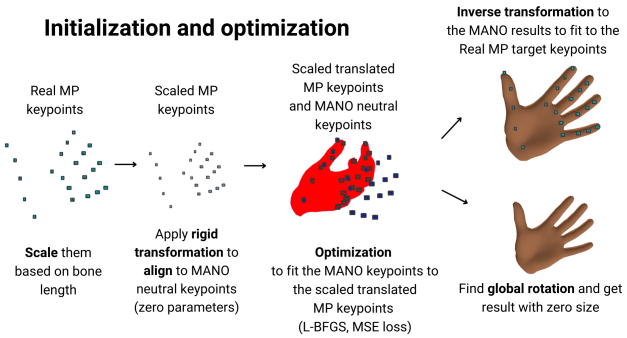


Figure 3. Optimization pipeline (MP stands for MediaPipe).

4. Method

Overview. To tackle the problem of fitting an articulated 3D hand from a single RGB image, we design the pipeline shown in Fig. 2. The input is a standard RGB image containing a human hand, and the output is a 3D hand in the exact same pose and orientation. The first step in the pipeline involves passing the input image through the MediaPipe Hand model estimator [1] to extract the 21 hand keypoints along with handedness information (whether the hand is left or right). This process results in a list of keypoints corresponding to pixel locations in the input image, which we later use as ground truth parameters for fitting the MANO hand model [26]. For the optimization step, our goal is to fit the MANO model's keypoints to the extracted "ground truth" keypoints from MediaPipe. Given the pose and shape parameters of MANO, we can obtain the 21 hand joint locations through linear interpolation. MANO takes 45 pose parameters, 10 shape parameters, and 3 global rotation parameters as input to generate a 3D mesh that represents a unique hand configuration. The fitting process begins with a neutral "zero" pose and shape. Through a series of transformations and iterative optimization, we adjust the MANO parameters to align its 21 keypoints with the estimated MediaPipe keypoints by minimizing a loss function. A key challenge was when the input hand was in a different global rotation than MANO's default pose. The root of the problem was the initialization: MANO starts in a neutral pose, shape, and rotation, since we have no prior information about the input hand's orientation, and initialized every parameter with zeros. However, if the hand in the input image had a significantly different rotation, for example, in a "handshake" position, the optimization process failed, resulting in completely implausible hand meshes.

Optimization Pipeline. To address this challenge, we compute a rigid transformation aligning the neutral MANO keypoints to MediaPipe keypoints. Additionally, we applied scaling to ensure both sets of keypoints were properly aligned before beginning the optimization process. The complete optimization pipeline is illustrated in Fig. 3. The rigid transformation was computed using six stable palm joints, namely keypoints [0,1,5,9,13,17] as illustrated in Fig. 1 to minimize the influence of finger articulation. To compute the transformation, we implemented a custom function that returns a 4×4 transformation matrix, where the top-left 3×3 block represents the rotation, the top-right 3×1 column is the translation vector, and the bottom row is used for homogenous coordinates. For scaling, we computed a scale factor using the distance between keypoints 0 and 5 (wrist to index MetaCarpoPhalangeal joint) in both the target and MANO keypoints, ensuring anatomically proportional alignment. After the initial alignment, we used scipy.minimize [7] to optimize the MANO parameters, specifically leveraging either the BFGS, or the L-BFGS method. We experimented with three loss functions: MSE, GM, and Huber loss. To improve the accuracy of fingertip alignment, we explored weighted loss functions, as we observed that most keypoints were densely concentrated around the palm. Due to the nature of the loss functions, the optimization process primarily focused on minimizing the error in these denser regions, often leading to less precise alignment of the fingertips. However, in real-world hand movement, fingertips play a critical role in defining hand gestures and poses, making their accurate positioning essential. To address this, we applied weighted loss functions across all three variations (MSE, GM, and Huber), giving higher importance to fingertip keypoints to ensure their proper alignment. Additionally, we attempted to integrate anatomical joint constraints into the loss function to enforce physically plausible hand poses. However, this approach proved ineffective, as it restricted the optimization process too severely. Despite experimenting with various weighting schemes, the results did not improve. Instead, we adopted a two-stage optimization strategy:

- Stage 1: A standard MSE loss function was used to obtain an initial estimate of the hand pose.
- Stage 2: The output of Stage 1 was refined using anatomical loss constraints, combined with an MSE loss applied only to 2D keypoints. This ensured that the estimated hand remained within realistic anatomi-

Table 1. The description of what combination of optimizer and loss function each experiment uses.

ID	OPTIMIZERS		LOSSES						
	LBFGS	BFGS	MSE	MSE fingertips	Geman-McClure	Geman-McClure fingertips	Huber	Huber fingertips	Anatomical **(2 stages)**
A	✓		✓	✓					
B	✓		✓						
C	✓				✓	✓			
D	✓				✓				
E	✓						✓	✓	
F	✓						✓		
G		✓	✓	✓					
H	✓		✓						✓

cal bounds.

After optimization, we reversed the rigid transformation to recover the MANO keypoints and mesh in the original scale, orientation, and position. The rotation matrix, translation vector, and scaling factor used for initialization were inverted to map the optimized MANO results back to the target's coordinate system so that we can inspect our results visually. To determine the root pose, we computed the axis-angle representation of the rotation matrix derived from the rigid transformation and incorporated it into the MANO model parameters. This ensured that the global rotation was accurately represented in the final output. This means that we now also have a 3D hand with the correct global orientation and pose but with the zero shape that is not really a problem.

5. Experiments

For the quantitative evaluation of our method, we follow the evaluation pipeline proposed in [11] and [33]. This ensures direct comparability between our approach and both their methods, as well as with other SotA techniques they benchmarked. Although [11] employs an optimization-based approach similar to ours, while [33] follows a learning-based approach, we evaluate our method against both SotA deep learning methods to provide a comprehensive performance assessment.

Implementation Details. For optimization, we employed the BFGS and L-BFGS algorithms from scipy.minimize [7]. Regarding loss functions, we utilized PyTorch's built-in MSELoss for Mean Squared Error computation. However, for the Geman-McClure and Huber loss functions, we implemented custom versions to ensure proper integration within our framework. Additionally, anatomical constraints are inherently implemented in the Manotorch [2] framework, which was a key factor in our decision to adopt it.

Using Images in the Wild. With the improvements in initialization and optimization, the model is now robust enough to handle hand keypoints extracted from images captured "in the wild," such as casual or uncontrolled environments. Moreover, our current implementation does not use the camera parameters so every 2D image containing a hand can be used as an input.

Table 2. Ablation Study: End-Point Error (mm) (↓) and AUC of PCK (↑) results of EgoDexter and Dexter+Object.

ID	EgoDexter		Dexter+Object	
	EPE (mm) (↓)	AUC of PCK (↑)	EPE (mm) (↓)	AUC of PCK (↑)
A	**17.724**	**0.883**	13.985	**0.946**
B	19.642	0.859	14.859	0.943
C	17.729	**0.883**	13.980	**0.946**
D	19.648	0.859	14.878	0.942
E	17.864	0.882	13.978	**0.946**
F	19.963	0.855	15.279	0.939
G	17.787	**0.883**	**13.975**	**0.946**
H	42.833	0.492	24.574	0.784

Metrics. To ensure a direct and fair comparison with SotA methods, we evaluate our approach using the same metrics as [11] and [33]. These metrics assess the accuracy of 3D hand pose estimation by measuring the deviation of predicted keypoints from the ground truth.

We compute the Root Mean Square Error (RMSE), also referred to as End-Point Error (EPE), for all 3D hand joints. This metric quantifies the absolute difference between the estimated keypoints and the ground truth in millimeters. A lower RMSE value indicates a more precise reconstruction of the hand pose.

The Percentage of Correct Keypoints (PCK) evaluates the proportion of keypoints that are correctly estimated within a given threshold distance from the ground truth. We compute PCK for thresholds ranging from 20mm to 50mm to analyze accuracy at different tolerance levels. To summarize the PCK performance across different thresholds, we also compute the Area Under the Curve (AUC) for this range. A higher AUC value indicates better overall accuracy, with an ideal score reaching 1.0, representing a perfect

fit. This metric is particularly useful for comparing methods holistically, as it accounts for performance across multiple threshold levels rather than relying on a single fixed distance.

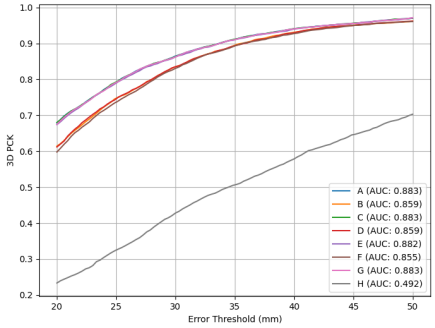
Datasets. Since our method does not involve a training phase, we exclusively use datasets for evaluation purposes.

EgoDexter [21] is an RGB-D dataset designed for evaluating hand-tracking algorithms in cluttered environments with significant occlusions. It consists of four video sequences featuring four different actors (two female) interacting with various objects in diverse settings. Ground truth annotations include manually labeled 3D fingertip positions on depth data. However, only fingertips are annotated, and due to occlusions, not all frames contain annotations for all five fingertips.

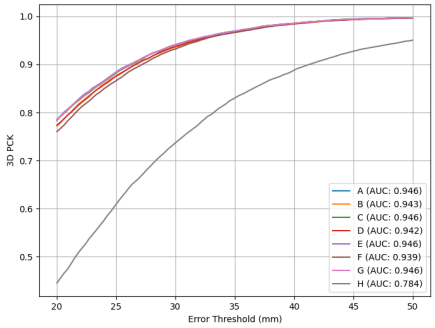
Dexter+Object [29] is an RGB-D dataset designed for evaluating algorithms that track both hands and objects simultaneously. It comprises six video sequences featuring two actors (one female) interacting with a simple cuboid-shaped object. Ground truth annotations include manually labeled 3D fingertip positions and three cuboid corners on depth data. However, for our evaluation, we only utilize the five-fingertip positions. Although all frames contain annotations, occlusions are present, particularly in sequences in which the cuboid obstructs parts of the hand. Both datasets provide manually annotated 3D hand keypoints in millimeters in camera coordinates. However, our method, which is based on MediaPipe's predictions, outputs 21 keypoints in pixel space in (x, y, z) format, including depth values. Thus, we must transform them to camera coordinates to ensure compatibility with the dataset's ground truth values.

5.1. Ablation Study

Tab. 1 presents the different experimental configurations we tested to evaluate our method on two datasets. Each experiment is identified by a letter (ID) and corresponds to a specific combination of optimizers and loss functions. While we tested all loss function combinations using the L-BFGS optimizer, we selectively tested what we considered the most promising loss function setup with the BFGS optimizer. Tab. 2 presents the quantitative results of our experiments, reporting the End-Point Error (in mm) and the AUC of PCK for both datasets. Our findings indicate that most experiments perform consistently well, with results comparable to each other. This consistency is expected, as the different loss functions are all variations of the MSE loss, designed to improve robustness against outliers. The best-performing results for each metric are highlighted in bold, although in many cases, multiple configurations achieve similar results. Our observations suggest that weighted fingertips loss functions tend to yield slightly better performance across metrics. However, the standard loss functions also achieve competitive results, while the better results of



(a) 3D PCK results on the EgoDexter dataset.



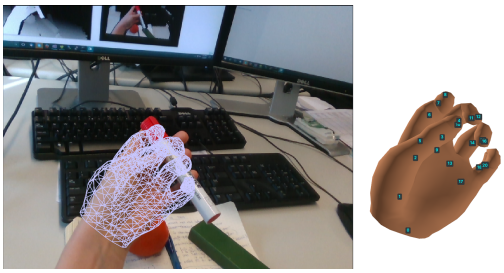
(b) 3D PCK results on the Dexter+Object dataset.

Figure 4. 3D PCK evaluation results.

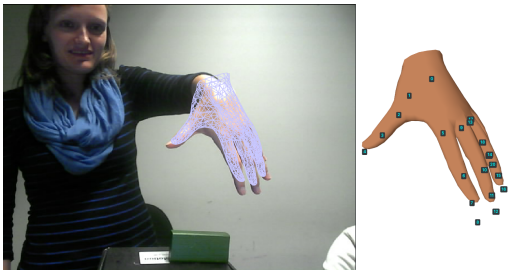
the weighted fingertips loss may be attributed to the specific evaluation datasets focusing primarily on fingertip keypoints. Additionally, we can observe that in experiment H, this two-stage approach integrating anatomical constraints resulted in worse overall performance although it successfully corrected depth-related errors caused by MediaPipe's 3D predictions in specific cases.

Overall, our results suggest that the most effective optimizer-loss combinations were Experiment A and Experiment G. This outcome aligns with expectations, as L-BFGS and BFGS are closely related, with L-BFGS being a memory-efficient variant. If computational efficiency is considered alongside accuracy, Experiment A, using L-BFGS with a weighted combination of MSE losses, emerges as the optimal choice. Figs. 4a and 4b compare the performance of the methods discussed above on the EgoDexter and Dexter+Object datasets, respectively. In both cases, the method incorporating anatomical constraints performs the worst. For the EgoDexter dataset, we observe that methods utilizing weighted fingertip loss functions consistently outperform those with standard loss functions, forming a distinct cluster of higher-performing models. However, on the Dexter+Object dataset, this distinction is less pronounced.

(a) Simple hand pose from EgoDexter.



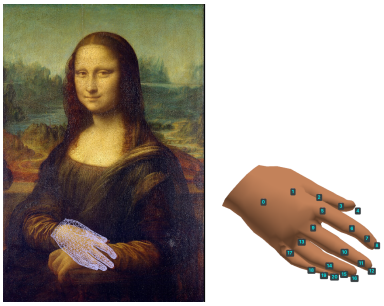
(b) Challenging hand pose with occlusion from EgoDexter.



(c) Dexter+Object result with MSE loss. Fingertip misalignment.

(d) Same image with weighted fingertips MSE loss. Fingertip alignment improves.

(e) Failure case due to severe occlusion.



(f) In-the-wild test on the Mona Lisa painting. Method generalizes well.

Figure 5. Qualitative results on both EgoDexter and Dexter+Object datasets as well as images "in the wild." Examples include successful and failure cases, demonstrating both robustness and limitations of the method.

5.2. Quantitative Evaluation

Tab. 3 presents a comparison of our proposed method against SotA approaches, reporting the AUC of PCK on both the Dexter+Object and EgoDexter datasets. The table includes both optimization-based and learning-based methods. Our method achieves the highest AUC on the EgoDexter dataset, outperforming all other approaches. For the Dexter+Object dataset, our method is nearly on par with the best-performing approach, with only a marginal difference of 0.002 compared to Zhou *et al*. [33]. Notably, our method not only surpasses other optimization-based methods such as Drosakis and Argyros [11] and Boukhayma *et al*. [9], but it also outperforms several learning-based methods. This is particularly significant since learning-based methods typically require substantial computational resources for training. Furthermore, by comparing Tab. 3 with Tab. 2, we observe that not only does our best-performing configuration achieve SotA results, but even several of our alternative se-

Table 3. AUC of PCK (↑) comparison with state-of-the-art methods. We use "*" to note the methods that work in real-time, which is a more challenging task.

Method	**Dexter+ Object**	**EgoDexter**
Ours	0.946	**0.883**
Zhou *et al*. [33]	**0.948***	0.811*
Zhang *et al*. [32]	0.825	-
Baek *et al*. [8]	0.650	-
Xiang *et al*. [31]	0.912	-
Boukhayma *et al*. [9]	0.763	0.674
Iqbal *et al*. [15]	0.672	0.543
Spurr *et al*. [28]	0.511	-
Mueller *et al*. [20]	0.482*	-
Zimmermann and Brox [34]	0.573	-
Drosakis and Argyros [11]	0.764	0.563

tups remain competitive with the top methods in the field.

Finally, it is important to emphasize that the datasets used for evaluation were not included in the training phase of any learning-based methods, ensuring a fair comparison with SotA approaches.

5.3. Qualitative Evaluation

To further assess the performance of our method, we present qualitative results across various scenarios in Figs. 5a–5f. Figs. 5a and 5b showcase examples from the EgoDexter dataset. In the simpler case (Fig. 5a), our method accurately predicts the hand keypoints. However, in the more challenging scenario (Fig. 5b), where occlusion from object interaction occurs, the method still performs reasonably well, though minor inaccuracies appear. Figs. 5c and 5d compare the effect of different loss functions on the Dexter+Object dataset. The simple MSE loss (Fig. 5c) results in less accurate predictions, particularly in fingertip locations. By contrast, using a weighted MSE loss with fingertip emphasis (Fig. 5d) improves the prediction. In Fig. 5e, we present a failure case caused by extreme occlusion. The EgoDexter dataset does not provide ground-truth keypoints for such cases, making evaluation difficult. Additionally, MediaPipe fails to detect a hand in this scenario, which directly impacts our method, as we rely on its initial keypoint predictions rather than ground-truth annotations from a dataset. However, in less challenging cases where MediaPipe successfully detects a hand, our approach remains effective.

Finally, in Fig. 5f, we demonstrate that our method generalizes beyond structured datasets by estimating hand poses from an image of the Mona Lisa painting. This showcases that our approach does not rely on camera parameters and can function on in-the-wild RGB images, making it applicable in diverse real-world scenarios.

6. Conclusion

In this paper, we proposed an optimization-based solution for estimating the 3D articulation of a human hand from a single RGB image, without knowledge of the camera intrinsic parameters. Our method leveraged the MediaPipe keypoint detector to obtain an initial estimation of the hand joints in the 2D space, and it performed a fitting stage using the MANO parametric model to obtain the 3D joint rotations. For the fitting stage we incorporated a fingertip alignment loss coupled with anatomical constraints. Our extensive evaluation demonstrated that our approach can robustly operate in-the-wild without the need for prior camera parameter information, while being competitive when compared to SotA data-driven models.

References

[1] Hand landmarks detection guide. https://developers.google.com/mediapipe/solutions/vision/hand_landmarker [Accessed: (10/6/2025)]. 1, 2, 3, 4

[2] Manotorch: framework implementing MANO in PyTorch. https://github.com/lixiny/manotorch [Accessed: (10/6/2025)]. 3, 5

[3] Mean squared error (MSE), probabilitycourse.com. https://www.probabilitycourse.com/chapter9/9_1_5_mean_squared_error_MSE.php [Accessed: (9/2/2025)]. 3

[4] MMPose hand keypoint estimation. https://mmpose.readthedocs.io/en/latest/demos.html [Accessed: (10/6/2025)]. 3

[5] OpenPose. https://github.com/CMU-Perceptual-Computing-Lab/openpose [Accessed: (10/6/2025)]. 3

[6] Rigid transformation in 3D space: Translation and rotation. https://medium.com/@parkie0517/rigid-transformation-in-3d-space-translation-and-rotation-d701d8859ba8 [Accessed: (10/6/2025)]. 3

[7] scipy.optimize. minimize. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html [Accessed: (10/6/2025)]. 4, 5

[8] Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim. Pushing the envelope for RGB-based dense 3D hand pose estimation via neural rendering. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1067–1076, 2019. 2, 7

[9] Adnane Boukhayma, Rodrigo de Bem, and Philip H. S. Torr. 3D hand shape and pose from images in the wild. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10835–10844, 2019. 2, 7

[10] Olga Chernytska. Gentle introduction to 2D hand pose estimation: Approach explained, 2021. https://towardsdatascience.com/gentle-introduction-to-2d-hand-pose-estimation-approach-explained-4348d6d79b11 [Accessed: (10/6/2025)]. 2

[11] Drosakis Drosakis and Antonis A. Argyros. 3D hand shape and pose estimation based on 2D hand keypoints. *Proceedings of the 16th International Conference on PErvasive Technologies Related to Assistive Environments*, 2023. 2, 3, 5, 7

[12] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3D hand shape and pose estimation from a single RGB image. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10825–10834, 2019. 2

[13] Stuart Geman and Donald E. McClure. Statistical methods for tomographic image reconstruction. *Bulletin of the International Statistical Institute*, 52(4):5–21, 1987. 3

[14] Peter J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35:492–518, 1964. 3

[15] Umar Iqbal, Pavlo Molchanov, Thomas M. Breuel, Juergen Gall, and Jan Kautz. Hand pose estimation via latent 2.5D heatmap regression. In *European Conference on Computer Vision*, 2018. 2, 7

[16] Dominik Kulon, Riza Alp Güler, Iasonas Kokkinos, Michael M. Bronstein, and Stefanos Zafeiriou. Weakly-supervised mesh-convolutional hand reconstruction in the wild. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4989–4999, 2020. 2

[17] Guan Ming Lim, Prayook Jatesiktat, and Wei Tech Ang. MobileHand: Real-time 3D hand shape and pose estimation from color image. In *International Conference on Neural Information Processing*, 2020. 2

[18] John Lin, Ying Wu, and T.S. Huang. Modeling the constraints of human hand motion. In *Proceedings Workshop on Human Motion*, pages 121–126, 2000. 3

[19] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. MediaPipe: A framework for building perception pipelines. *ArXiv*, abs/1906.08172, 2019. 3

[20] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. Ganerated hands for real-time 3D hand tracking from monocular RGB. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 7

[21] Franziska Mueller, Dushyant Mehta, Oleksandr Sotnychenko, Srinath Sridhar, Dan Casas, and Christian Theobalt. Real-time hand tracking under occlusion from an egocentric RGB-D sensor. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2017. 6

[22] Jorge Nocedal. Updating Quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35:773–782, 1980. 3

[23] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer, 2006. 3

[24] Paschalis Panteleris, Iasonas Oikonomidis, and Antonis A. Argyros. Using a single RGB frame for real time 3D hand pose estimation in the wild. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 436–445, 2017. 2

[25] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10967–10977, 2019. 3

[26] 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), Nov. 2017. 1, 2, 4

[27] Tomas Simon, Hanbyul Joo, I. Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4645–4653, 2017. 2

[28] Adrian Spurr, Jie Song, Seonwook Park, and Otmar Hilliges. Cross-modal deep variational hand pose estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 89–98, 2018. 7

[29] Srinath Sridhar, Franziska Mueller, Michael Zollhoefer, Dan Casas, Antti Oulasvirta, and Christian Theobalt. Real-time joint tracking of a hand manipulating an object from RGB-D input. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 6

[30] Dimitrios Tzionas, Luca Ballan, Abhilash Srikantha, Pablo Aponte, Marc Pollefeys, and Juergen Gall. Capturing hands in action using discriminative salient points and physics simulation. *International Journal of Computer Vision*, 118(2):172–193, Mar. 2016. 3

[31] Donglai Xiang, Hanbyul Joo, and Yaser Sheikh. Monocular total capture: Posing face, body, and hands in the wild. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10957–10966, 2018. 7

[32] Xiong Zhang, Qiang Li, Wenbo Zhang, and Wen Zheng. End-to-end hand mesh recovery from a monocular RGB image. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2354–2364, 2019. 2, 7

[33] Yuxiao Zhou, Marc Habermann, Weipeng Xu, Ikhsanul Habibie, Christian Theobalt, and Feng Xu. Monocular real-time hand shape and motion capture using multi-modal data. In *Proceedings of the IEEE International Conference on Computer Vision*, 2020. 5, 7

[34] Christiane Zimmermann and Thomas Brox. Learning to estimate 3D hand pose from single RGB images. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4913–4921, 2017. 2, 7

[35] Christiane Zimmermann, Duygu Ceylan, Jimei Yang, Bryan C. Russell, Max Argus, and Thomas Brox. FreiHAND: A dataset for markerless capture of hand pose and shape from single RGB images. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 813–822, 2019. 2