FAST LIFTING FOR 3D HAND POSE ESTIMATION IN AR/VR APPLICATIONS

Onur G. Guleryuz and Christine Kaeser-Chen
Daydream, Google Inc., Mountain View, CA, USA

ABSTRACT

We introduce a simple model for the human hand skeleton that is geared toward estimating 3D hand poses from 2D keypoints. The estimation problem arises in AR/VR scenarios where low-cost cameras are used to generate 2D views through which rich interactions with the world are desired. Starting with a noisy set of 2D hand keypoints (camera-plane coordinates of detected joints of the hand), the proposed algorithm generates 3D keypoints that are (i) compliant with human hand skeleton constraints and (ii) perspective-project down to the given 2D keypoints. Our work considers the 2D to 3D lifting problem algebraically, identifies the parts of the hand that can be lifted accurately, points out the parts that may lead to ambiguities, and proposes remedies for ambiguous cases. Most importantly, we show that the finger-tip localization errors are a good proxy for the errors at other finger joints. This observation leads to a look-up-table-based formulation that instantaneously determines finger poses without solving constrained trigonometric problems. The result is a fast algorithm running super real-time on a single core. When hand bone-lengths are unknown our technique estimates these and allows smooth AR/VR sessions where a user's hand is automatically estimated in the beginning and the rest of the session seamlessly continued. Our work provides accurate 3D results that are competitive with the state-of-the-art without requiring any 3D training data.

Index Terms— Augmented-reality, virtual-reality, 3D hand pose, 3D keypoints, lifting, monocular estimation.

1. INTRODUCTION

Many mobile AR/VR applications rely on world views generated through low-cost 2D cameras [3, 4, 1, 2]. Through such systems information about the user and the surrounding environment is gathered, processed, and finally rendered in relation to the real/virtual world (see Figure 1). Keeping track of the users' hands can enable rich interactions that further extend the usability and benefits of AR/VR [20, 5]. For example, menu or gesture-based interactions allow the user to select options, input data, and point to interesting objects for further processing. The rendering step can take advantage of detected hands, enabling use cases with tactile feedback where menus are rendered in the palm of one hand while selections are made with the other. The hand can also be used as a secondary controller extending user interaction in cases where the primary controller is inadequate or awkward. Most of these applications benefit from knowing the 3D coordinates of users' hands, in particular the palm and finger tips, to acceptable accuracy levels.

Recent years have seen substantial progress in machine learning algorithms' ability to locate objects in monocular images with high precision [18, 7]. It is now a given that 2D keypoints of the hand (typically the labeled camera-plane coordinates of five finger joints on the palm, two joints per finger, five finger-tips, and a wrist center; see Figure 2 (a)) can be located accurately with moderately complex deep learning algorithms based on conv-nets [23, 21, 16]. Bounding boxes [10] that locate the hand (which can then be augmented with less accurate but noticeably quicker vision-based keypoint detectors) can also be computed reliably. 3D versions of these techniques, that is machine learning algorithms that provide 3D world coordinates of keypoints from a monocular image, are in active research [15, 14, 12, 9, 11]. 3D keypoint detectors still face an uphill climb since (i) one needs abundant data to learn the many constraints of the hand but 3D data is scarce, (ii) self occlusions of the hand widely observed in typical poses make learning and inference from monocular images difficult[1], (iii) the speed required in enabling rich interactions with naturally fast moving hands/fingers is hard to maintain with computationally complex deep-nets especially within the complexity/power envelopes of mobile AR/VR platforms.

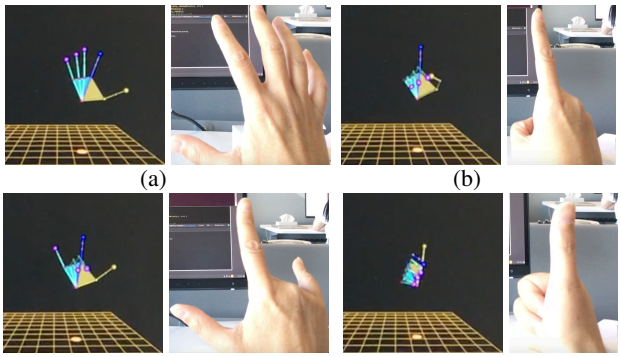


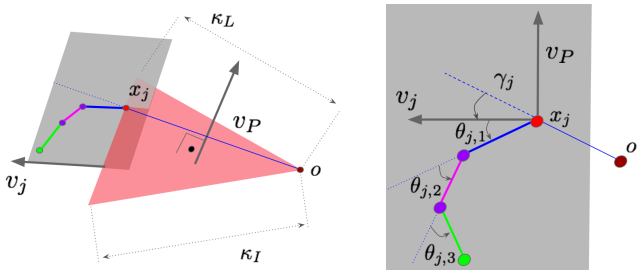
Fig. 1. Example poses in 3D estimated by the proposed work from a monocular camera. The hand is imaged from an egocentric view, 2D keypoints detected, then lifted to obtain 3D world coordinates. 3D poses are rendered from a perspective facing the camera for clarity.

Techniques that try to circumvent the difficulties of direct 3D inference concentrate on lifting 2D keypoints to 3D. This category of methods include nearest neighbor matchers [6], techniques that infer 3D pairwise distances between joints from 2D pairwise distances [13], and methods like [27] that apply deep learning to determine 3D keypoints based on 2D keypoint heat-maps. Researchers have also considered linear expansions of the 3D pose (concatenated into a vector of coordinates) by learning PCA basis and, as an improvement to PCA, sparse approximating dictionaries [22, 17, 26].

From a computational standpoint it can be seen that most available techniques are too complex for incorporation into mobile plat-

Fig. 2. Hand keypoints. (a) 20 keypoints marked on the hand. (b) A finger skeleton and the palm/thumb triangles used in our model.

[1]Self occlusions can cause severe problems even in stereo views necessitating multi-view setups in some applications [19].



(a) (b)

Fig. 3. Parts of the model used in this paper. (a) The palm triangle, $\mathcal{T}_P$. (b) The plane embedding of the j-th finger attached to $\mathcal{T}_P$.

forms. Beyond their main operation, many techniques require post-processing and further complexity since their output is not compliant with human hand skeleton constraints and bone-lengths. Furthermore much of existing work either ignores perspective projection constraints or, like the techniques using linear expansions, approximate them using weak perspective projection. Weak perspective projection is not an adequate placeholder for perspective projection especially for egocentric views in VR/AR, where the various hand joints' distance to the camera is often quite significantly disparate.

In this paper we propose a simple model for the human hand skeleton that is conveniently parametrized for quick estimation. We provide a lifting algorithm that predicts 3D world coordinates of input 2D camera-plane keypoints[2]. The 3D hand poses generated by our algorithm are compliant with human hand skeletal constraints and with perspective projection constraints. We show that ambiguities related to lifting [8] do not significantly impact the palm under typical poses and likewise affect the finger keypoints in application-wise less consequential poses. We further show that localization errors for finger-tips serve as a convenient proxy for localization errors for other joints. This allows us to further simplify our formulation to use look-up-tables that can instantaneously determine entire finger poses given only the locations of the palm and finger-tips. The end result is a fast algorithm that can be used for lifting as well as keypoint denoising (e.g., making the output of other algorithms skeletal and/or perspective projection compliant). When desired our work also incorporates bone-length variations by rapidly estimating skeleton-constrained bone-lengths with quadratic programming.

Section 2 introduces our model with sections 2.2 and 2.3 devoted to the lookup table formulation and bone-length estimation. Section 2.4 considers the impact of perspective projection on lifting and discusses ambiguous cases. Section 2.5 provides our lifting algorithm. Section 3 includes simulation results and concludes the paper.

2. HAND SKELETON MODEL

2.1. Two-Triangle Model for the Hand

The proposed model represents the palm and the portion of the thumb that includes the thumb's metacarpal bone using two triangles $\mathcal{T}_P$ and $\mathcal{T}_T$ (see Figure 2 (b)). All sides and angles of $\mathcal{T}_P$ are user specific with its pose (location and orientation in 3D space) to be determined at each time instant. For $\mathcal{T}_T$ only two side-lengths (toward the index finger and the thumb knuckles) are fixed with the remaining length and angles varying with its pose, all determined at each time instant. The thumb is considered attached to $\mathcal{T}_T$ and the remaining fingers to $\mathcal{T}_P$. Note that the position of the index finger couples the poses of the triangles. In addition to the finger-tips and the finger-related joints on the triangles, the thumb has a single joint and the other fingers have two joints.

3D finger keypoints embedded in two dimensions: The main motivation of the triangle formulation is to enable easily-parametrized surfaces that constrain the relevant joints on each triangle and, as importantly, to constrain the relative positions of the fingers. For the latter purpose each triangle is equipped with a unit vector (v_P for $\mathcal{T}_P$ and v_T for $\mathcal{T}_T$) that constrains the movement of the attached fingers. (The relative poses of v_P/v_T to $\mathcal{T}_P$/$\mathcal{T}_T$ are fixed as hyper-parameters but can also be estimated per-user if desired.) As illustrated in Figure 3 (a) for $\mathcal{T}_P$, we assume all 3D keypoints of each finger lie in a plane, i.e., each 3D keypoint of the finger can be parametrized using two coordinates given the triangles and the finger related joints on the triangles. For finger j the associated plane passes through the finger-joint, x_j, on the respective triangle, with the 2D plane subspace spanned by v_P/v_T and a finger-specific direction vector v_j (estimated at each time instant). Referring to Figure 3, the k-th 3D keypoint on finger-j, $f_{j,k}$ (3×1), is then located at,

$$f_{j,k} = x_j + v_j \sum_{l=1}^{k} \delta_{j,l} cos(\phi_{j,l}) + v_P \sum_{l=1}^{k} \delta_{j,l} sin(\phi_{j,l}), \qquad (1)$$

where $\delta_{j,l}$ are the finger bone-lengths and $\phi_{j,l} = \sum_{m=1}^{l} \theta_{j,m}$. Note that (1) holds with $k = 1, \ldots, 3$ for the index through little finger, and with $k = 1, 2$ and v_P replaced by v_T for the thumb.

Skeletal constraints: In addition to constraints imposed by the triangles and plane embeddings, our model has the following constraints. ***(i)*** *Middle and ring fingers are between index and little fingers, i.e.,*

$$\begin{aligned} x_3 &= \lambda x_2 + (1-\lambda) x_5, \\ x_4 &= \rho x_2 + (1-\rho) x_5, \end{aligned} \qquad (2)$$

where $1 > \lambda > \rho > 0$ are parameters specific to a user's hand. Let U (3×3) contain unit vectors in the direction of the three edges emanating from o so that the first column of U points toward the metacarpal bone of the thumb, second and third columns to the index and little finger joints on the palm respectively. We have,

$$U = R_{hand} \left[\; R_{rel}\, u_T \;\vdots\; u_I \;\vdots\; u_L \; \right] \qquad (3)$$

where u_T, u_I, and u_L (3×1) correspond to a fixed, coplanar pose of the two triangles, the rotation R_{rel} (3×3) poses $\mathcal{T}_T$ relative to $\mathcal{T}_P$, and R_{hand} determines the overall orientation. This leads to,

$$X^t = \begin{bmatrix} \kappa_T & 0 & 0 \\ 0 & \kappa_I & 0 \\ 0 & \lambda\kappa_I & (1-\lambda)\kappa_L \\ 0 & \rho\kappa_I & (1-\rho)\kappa_L \\ 0 & 0 & \kappa_L \end{bmatrix} U^t + \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} o^t, \qquad (4)$$

where X has x_j, $j = 1, ..., 5$, in its columns, $\kappa_T, \kappa_I, \kappa_L$ are the side lengths of the triangles, o is the wrist center, and $(.)^t$ is transpose.
(ii) *Finger joints have bending limits, i.e.,*

$$\Theta_{j,min} < \theta_{j,k} < \Theta_{j,max}, \qquad (5)$$

where the bounds are hyper-parameters set as 0 and $\pi/2$ respectively.
(iii) *Finger embedding planes have directional bounds, i.e.,*

$$\Gamma_{j,min} < \gamma_j < \Gamma_{j,max}, \qquad (6)$$

where γ_j is as shown in Figure 3 (b) and the bounds are again hyper-parameters, set as $-\pi/15$ and $\pi/15$ respectively.
(iv) *Triangles' relative pose cannot be arbitrary, i.e.,*

$$\tau_{min} \leq \angle(\perp_{\mathcal{T}_P}, \perp_{\mathcal{T}_T}) \leq \tau_{max}, \qquad (7)$$

where $\angle(\perp_{\mathcal{T}_P}, \perp_{\mathcal{T}_T})$ is the angle between the surface normals of the two triangles (measured starting from $\perp_{\mathcal{T}_P}$) and the hyper-parameters τ_{min} and τ_{max} set to 0 and $\pi/2$ respectively.

[2]The predictions of the algorithm are up to an overall scale parameter. Such a scale parameter is needed by all monocular 3D estimation methods since an object at depth z from the camera and a similar but twice as large object at $2z$ will yield similar 2D keypoints under similar poses.

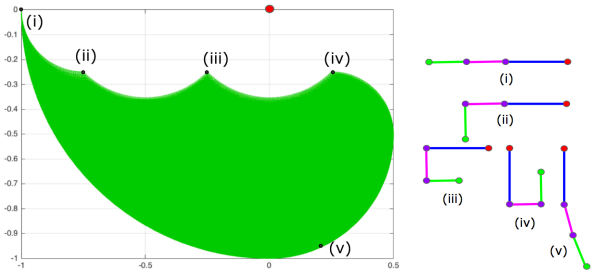


Fig. 4. Possible locations of the finger-tip assuming a unit length finger-j on the right hand palm that is pointing left ($\delta_{j,1} = .50$, $\delta_{j,2} = .25$, $\delta_{j,3} = .25$). (a) Finger-tip locations and example points. (b) Finger poses that correspond to the example points.

(v) *Embedding vectors enforce realistic finger movement, i.e.,* $v_P = \perp_{\mathcal{T}_P}$, $\angle(\perp_{\mathcal{T}_P}, v_T) = \pi/8$ with the incremental correction for v_T enabling more natural thumb movement.

Fixed and time varying parameters: The user-specific parameters are the triangle sides κ_T, κ_I, and κ_L, the multipliers λ and ρ (see (4)), the $\mathcal{T}_P$ angle $\angle(u_I, u_L)$ (see (3)), and the finger bone lengths $\delta_{j,l}$ (see (1)). These parameters are set/estimated per user. The remaining parameters (other than the hyper parameters as mentioned above) are estimated at each time instant to produce the 3d pose.

Fitting the model: Let F_n be the matrix that contains the finger keypoints in (1) at time instant n and let F_n^i contain the input keypoints. The model is fit to minimize the squared sum of Frobenius norms,

$$\mathcal{M}_n = ||F_n^i - F_n||_2^2 + ||X_n^i - X_n||_2^2 + ||o_n^i - o_n||_2^2. \tag{8}$$

Time varying parameters are determined for every n and, when desired, the user-specific parameters for the sum, $\sum_n \mathcal{M}_n$.

2.2. Loci of Finger Tips and the Finger Pose Lookup Table

The 3D locations of finger keypoints in (1) looks complicated especially under joint angle constraints. Nevertheless, there is considerable structure on the possible locations of the finger-tip with respect to that finger's joint location on the associated triangle. Our method takes advantage of this structure which is illustrated in Figure 4 for the j-th finger on the palm of the right hand. All locations inside the green zone in Figure 4 (a) are locations for the finger-tip through skeletal-compliant poses of the finger. As illustrated in Figure 4 (b), at location (i) the finger is straight, pointing left. At (ii) the last segment of the finger is maximally bent. At (iii) the last two segments, and finally at (iv) all segments of the finger are maximally bent. (v) illustrates an almost straight finger maximally bent at the palm joint[3].

A small neighborhood of a given finger-tip location will contain different poses that lead to close-by finger-tips. Suppose x_j, i.e., the intercept of the finger on $\mathcal{T}_P$, is known. Assume that one knows the location of the j-th finger-tip, $f_{j,3}$, approximately via $\hat{f}_{j,3}$ such that,

$$||f_{j,3} - \hat{f}_{j,3}||_2 < \epsilon_{j,3}. \tag{9}$$

It is then interesting to see what can be said about the errors of the remaining joints, $\epsilon_{j,1}$ and $\epsilon_{j,2}$, conditioned on (9) . Suppose the joint approximations $\hat{f}_{j,1}$ and $\hat{f}_{j,2}$ are taken as the means of the respective neighborhoods that result from (9) and skeletal constraints. Figure 5 shows $\epsilon_{j,1}$ and $\epsilon_{j,2}$ calculated for the finger in Figure 4 over all possible finger-tip locations. Note first that when the finger-tip location is very finely known joint location errors are small but relatively significant. This is because multiple close-by poses can result in similar

[3]The thumb-tip loci is likewise structured albeit with a different shape.

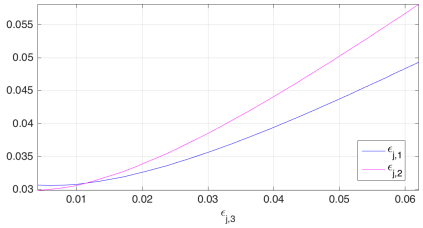


Fig. 5. Errors $\epsilon_{j,1}$ and $\epsilon_{j,2}$ for the finger joint locations as a function of the finger-tip error $\epsilon_{j,3}$ using the unit-length finger of Figure 4. $\epsilon_{j,3}$ serves as a convenient proxy for the remaining errors.

finger-tip locations. Regardless, when the finger-tip location incurs more realistic errors the joint errors closely track the finger-tip error. For example when $\epsilon_{j,3} = 0.04$ ($4mm$ assuming a $10cm$ finger) both joints' errors are within 0.045 ($4.5mm$).

We use this observation to construct finger-specific lookup tables (LUT). After projecting a noisy finger tip estimate on its embedding plane, the relevant LUT is consulted for (i) the location of the closest skeletal-compliant finger-tip and (ii) the most probable locations of finger joints that result in that finger-tip location. The LUT can be considered as Figure 4 discretized on a rectangular grid with each cell storing three angles that can be used in (1) to recover the finger poses. Cells that correspond to non-skeletal-compliant points refer to the skeletal-compliant pose that results in the closest location in Euclidean distance (i.e., cells that correspond to the white regions in the figure refer to the closest cell in the green region.) Within each cell we assumed the pose that minimized the ℓ_1 norm, $\sum_{l=1}^{k} |\theta_{j,l}|$, to be the most probable (see Figure 3 (b)).

2.3. Estimating Lengths

Using (1) one can see that the finger keypoint locations vary linearly with respect to the finger bone lengths. Hence, given the other length and pose-related parameters and using (8), $\sum_n \mathcal{M}_n$ leads to a quadratic program for the bone-lengths of each finger with positivity and relative size constraints incorporated. Equation (4) indicates linear variation with triangle side length κ_T, but a nonlinear coupling of κ_I, κ_L with the multipliers λ and ρ. Given the narrow ranges for the multipliers, a joint solution can be accomplished by sweeping the λ, ρ pair and solving a simple quadratic program for the side lengths for each potential pair. During length estimation, we minimized $\sum_n \mathcal{M}_n$ by iterating between solving for the lengths and lifting the pose (see Algorithm 1).

2.4. Perspective Projection and Lifting

Assume the camera's pose is known and adjustments to coordinates in terms of camera parameters are done so that the world coordinate q projects to the camera-plane coordinate $p = \mathcal{P}(q)$ via,

$$p = \mathcal{P}(q) = (q_1/q_3, q_2/q_3, 1)^t, \tag{10}$$

so that the vanishing point is at $(0, 0, 0)$ and the camera is at $(0, 0, 1)$ oriented toward the positive z-direction.

Let the matrix Q ($3 \times N$) contain N world coordinates in its columns. Suppose Q is rotated using R (3×3), translated with w (3×1), and finally perspective-projected to obtain the matrix P ($3 \times N$). Let s ($N \times 1$) be the vector of all ones. We have,

$$RQ + ws^t = PD, \tag{11}$$

where D is diagonal, $D(i, i) = d_i$, for some depth vector d ($N \times 1$).

Definition 2.1 *Q can be unambiguously lifted from P if*

$$R^*, w^* = \arg\min_{R', w'} \left\{ \min_{D' \geq 0} ||R'Q + w's^t - PD'||_2 \right\}, \tag{12}$$

results in a unique pose, $R^*Q + w^*s^t$ of Q.

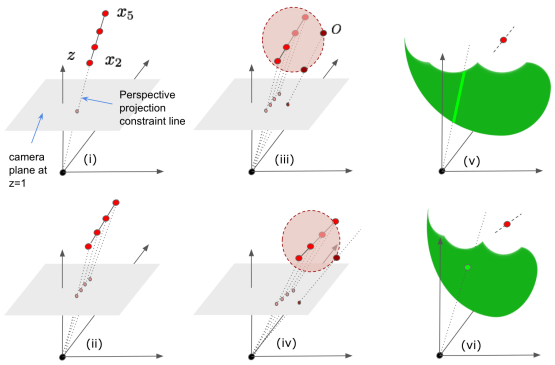


Fig. 6. Ambiguous (top row) and unambiguous (bottom row) poses. In (i) all camera-plane keypoints coalesce at one point resulting in an ambiguity in the locations of the finger joints on $\mathcal{T}_P$. This is an extreme, singular pose for the hand. Typical poses as shown in (ii) can be unambiguously lifted. Assuming known $x_2, \ldots, x_5$, (iii) shows the circle that must contain o when the projection constraint line for o is within the plane of the circle. (iv) illustrates the typical case where the line intercepts the circle's plane at a single point. (v) is similar to (iii) in that the ambiguity is caused by the constraint line being within the finger embedding plane. (vi) is the more typical case of the line intercepting the plane at a single point.

Proposition 2.2 *Three or more points on a line can be unambiguously lifted whenever the line does not project to a single point*[4].

With known hand lengths, the finger joint locations on $\mathcal{T}_P$ can be unambiguously lifted except when they project to a single point. One hence expects correct lifting of $x_2, ..., x_5$ for most poses. Figure 6 shows ambiguous and unambiguous poses for $\mathcal{T}_P$[5], ignoring noise for clarity. Observe that detecting ambiguous poses is straightforward under noise. One can, for example, calculate deviation of $\mathcal{P}(x_j),\ j = 2, \ldots, 5$ for (i), obtain the depicted constraint circle and calculate an appropriate distance to the line passing through the vanishing point and $\mathcal{P}(o)$ for (iii), and check the plane embedding finger-j against the projection constraint passing through the fingertip. With such measures ambiguous cases can be detected and countermeasures taken (e.g., by resolving using temporally nearby poses, giving feedback to the user, using image features, etc.) .

2.5. Algorithm

Algorithm 1 outlines our lifting algorithm which iteratively applies skeleton fitting and perspective projection constraints. When tracking a single hand, an unoptimized implementation exceeds 300 fps on a single desktop CPU core with minimal memory/cache impact (an optimized version is expected to be significantly faster).

3. SIMULATION RESULTS AND CONCLUSION

We used Algorithm 1 to lift 2D keypoints to 3D using the dataset provided in [24]. Following the observations of Section 2.2 we restricted the algorithm input to 11 keypoints (the keypoints on the triangles and finger tips) but obtained output and compared results on all 20 keypoints. This disadvantages our technique over others that utilize the full keypoint set, yet validates Section 2.2, and showcases the efficacy of our LUT formulation. As we are not proposing a 2D keypoint detector we emulated detector errors by adding Gaussian noise of standard deviation σ to the input pixel domain keypoints[6].

[4]We skip the proof to preserve space.

[5]$\mathcal{T}_T$ is similar except that the keypoint at the metacarpal bone of the thumb is approximately constrained to lie on a disc with respect to $\mathcal{T}_P$.

[6]Note that for a palm depicted within $\sim 80 \times 80$ pixel regions in input RGB, especially $\sigma = 5$ emulates a less than stellar 2D key-point detector.

Algorithm 1 FL - Fast Lifting

$m \leftarrow 1, \tau \leftarrow \infty, \varepsilon^0 \leftarrow 0, [X^1, F^1, o^1] \leftarrow [X^i, F^i, o^i]$.
Initialize $o, R_h, R_r, v_j, \theta_{j,k}$ `//j = 1, ...5, k = 1, ..., 2 or 3`
while $\tau > tol$ **do**
  `//fit skeleton`
  $[R_h, R_r] \leftarrow \arg\min_{R_h, R_r} ||F^m - F||_2^2 + ||X^m - X||_2^2$
  `update`$(X,\ F)$ `//Eq. (3), (4), (1)`
  $o \leftarrow \arg\min_o ||F^m - F||_2^2 + ||X^m - X||_2^2 + ||o^m - o||_2^2$
  `update`$(X,\ F)$
  $[v_1, ..., v_5] \leftarrow \arg\min_{v_1,\ldots,v_5} ||F^m - F||_2^2$
  $\theta_{j,k} \leftarrow$ `consult_LUT`$(f_j^m - x_j, v_j, v_{P/T})$
  `update`(F)
  `//perspective projection constraint`
  $[X^{m+1}, F^{m+1}, o^{m+1}] \leftarrow$
    `apply_pp_constraint`(X, F, o, X^i, F^i, o^i)
  $\varepsilon^m = ||F^{m+1} - F^m||_2^2 + ||X^{m+1} - X^m||_2^2 + ||o^{m+1} - o^m||_2^2$
  $\tau \leftarrow \varepsilon^m - \varepsilon^{m-1}, m \leftarrow m + 1$
return $X^{m+1}, F^{m+1}, o^{m+1}, R_h, R_r, v_j, \theta_{j,k}$

[27]	[24]	[25]	FL^a	$FL^{a,e}$	$FL^a_{\sigma=3}$	$FL^a_{\sigma=5}$	FL
0.95	0.84	0.77	0.96	0.94	0.92	0.88	0.93

Table 1. AUC for different methods. Our aligned results under FL^a columns are competitive with, or exceed, others even under significant camera-plane keypoint noise. Our completely unaligned result under the FL column is likewise competitive and shows that the method can predict depth and lift accurately.

We report results using PCK (percentage of correct keypoints) over thresholds from $20mm$ to $50mm$ errors, in $2mm$ increments. In Table 1 we compare AUC results (area under curve - evaluated using the PCK curves) to [27] and to [24, 25] (as tabulated in [27]). The work in [27] translates its output to match the ground-truth 3D location by aligning to a root 3D key-point. For FL^a we likewise aligned to the root keypoint to enable comparisons. $FL^{a,e}$ corresponds to results when our algorithm also estimates lengths (using only the 2D projected data with known κ_I providing the overall scale). FL results are not aligned and showcase true lifting performance. As shown in Figure 7, more than 80% of keypoints have under $20mm$ error for our unaligned FL results. Our technique, which requires no 3D data to train or learn from, is clearly accurate in guessing 3D poses over a large majority of the time and over a myriad of hand poses.

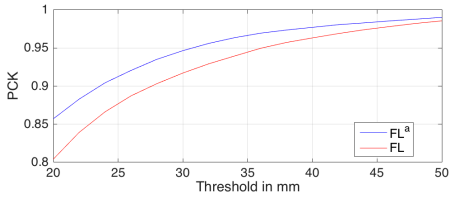


Fig. 7. PCK curves for FL^a and FL.

We proposed a simple model for the hand skeleton and an associated 3D fitting algorithm that can be used in lifting and denoising. Our lookup-table-based fast algorithm can lift most poses accurately without requiring any 3D data even while estimating bone-lengths. We showed that lifting can be accomplished accurately except for singular poses which can be detected and remedied. As indicated by the FL results, in addition to lifting in mobile AR/VR scenarios, our work can also be used to lift the bulk of the poses in 2D datasets to 3D in order to generate 3D datasets. Our work can be improved by adding finer details to the model and taking advantage of finger joints during fitting. We leave such extensions to another article.

4. REFERENCES

[1] Arcore: http://developers.google.com/ar/.

[2] Arkit: http://developer.apple.com/arkit/.

[3] Glass: https://www.x.company/glass/.

[4] Hololens: http://www.microsoft.com/enus/hololens/hardware.

[5] M. C. Cabral, C. H. Morimoto, and M. K. Zuffo. On the usability of gesture interfaces in virtual reality environments. In *Proceedings of the 2005 Latin American Conference on Human-computer Interaction*, CLIHC '05, pages 100–108, New York, NY, USA, 2005. ACM.

[6] C. H. Chen and D. Ramanan. 3d human pose estimation = 2d pose estimation + matching. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5759–5767, July 2017.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[8] B. K. P. Horn. *Robot Vision*. MIT Press, Cambridge, MA, USA, 1986.

[9] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. *arXiv*, 2017.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. 2016. To appear.

[11] J. Martinez, R. Hossain, J. Romero, and J. J. Little. A simple yet effective baseline for 3d human pose estimation. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2659–2668, 2017.

[12] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Trans. Graph.*, 36(4):44:1–44:14, July 2017.

[13] F. Moreno-Noguer. 3d human pose estimation from a single image via distance matrix regression. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[14] F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt. GANerated Hands for Real-time 3D Hand Tracking from Monocular RGB. *ArXiv e-prints*, Dec. 2017.

[15] M. Oberweger and V. Lepetit. Deepprior++: Improving fast and accurate 3d hand pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[16] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy. Towards accurate multi-person pose estimation in the wild. 2017.

[17] V. Ramakrishna, T. Kanade, and Y. A. Sheikh. Reconstructing 3d human pose from 2d image landmarks. In *European Conference on Computer Vision, 2012*, Pittsburgh, PA, October 2012.

[18] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.

[19] T. Simon, H. Joo, I. A. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4645–4653, 2017.

[20] T. Starner, B. Leibe, D. Minnen, T. Westyn, A. Hurst, and J. Weeks. The perceptive workbench: Computer-vision-based gesture tracking, object tracking, and 3d reconstruction for augmented desks. *Machine Vision and Applications*, 14(1):59–71, Apr 2003.

[21] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Trans. Graph.*, 33(5):169:1–169:10, Sept. 2014.

[22] C. Wang, Y. Wang, Z. Lin, A. L. Yuille, and W. Gao. Robust estimation of 3d human poses from a single image. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2369–2376, June 2014.

[23] S. E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4732, June 2016.

[24] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang. 3d hand pose tracking and estimation using stereo matching. *CoRR*, abs/1610.07214, 2016.

[25] R. Zhao, Y. Wang, and A. M. Martínez. A simple, fast and highly-accurate algorithm to recover 3d shape from 2d landmarks on a single image. *CoRR*, abs/1609.09058, 2016.

[26] X. Zhou, M. Zhu, S. Leonardos, and K. Daniilidis. Sparse representation for 3d shape estimation: A convex relaxation approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(8):1648–1661, Aug 2017.

[27] C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single rgb images. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4913–4921, Oct 2017.