Lightweight Multi-View 3D Pose Estimation through Camera-Disentangled Representation

Edoardo Remelli[1] Shangchen Han[2] Sina Honari[1] Pascal Fua[1] Robert Wang[2]
[1]CVLab, EPFL, Lausanne, Switzerland
[2]Facebook Reality Labs, Redmond, USA

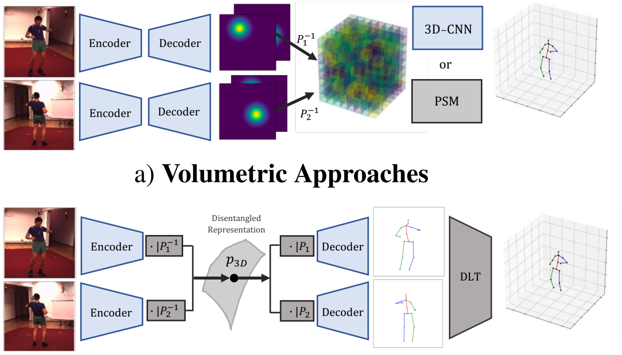
Abstract

We present a lightweight solution to recover 3D pose from multi-view images captured with spatially calibrated cameras. Building upon recent advances in interpretable representation learning, we exploit 3D geometry to fuse input images into a unified latent representation of pose, which is disentangled from camera view-points. This allows us to reason effectively about 3D pose across different views without using compute-intensive volumetric grids. Our architecture then conditions the learned representation on camera projection operators to produce accurate per-view 2d detections, that can be simply lifted to 3D via a differentiable Direct Linear Transform (DLT) layer. In order to do it efficiently, we propose a novel implementation of DLT that is orders of magnitude faster on GPU architectures than standard SVD-based triangulation methods. We evaluate our approach on two large-scale human pose datasets (H36M and Total Capture): our method outperforms or performs comparably to the state-of-the-art volumetric methods, while, unlike them, yielding real-time performance.

1. Introduction

Most recent works on human 3D pose capture has focused on monocular reconstruction, even though multi-view reconstruction is much easier, since multi-camera setups are perceived as being too cumbersome. The appearance of Virtual/Augmented Reality headsets with multiple integrated cameras challenges this perception and has the potential to bring back multi-camera techniques to the fore, but only if multi-view approaches can be made sufficiently lightweight to fit within the limits of low-compute headsets.

Unfortunately, the state-of-the-art multi-camera 3D pose estimation algorithms tend to be computationally expensive because they rely on deep networks that operate on volumetric grids [14], or volumetric Pictorial Structures [22, 21], to combine features coming from different views in accordance with epipolar geometry. Fig. 1(a) illustrates these approaches.



b) **Canonical Fusion (ours)**

Figure 1. Overview of 3D pose estimation from multi-view images. The state-of-the-art approaches project 2D detections to 3D grids and reason jointly across views through computationally intensive volumetric convolutional neural networks [14] or Pictorial Structures (PSM) [22, 21]. This yields accurate predictions but is computationally expensive. We design a lightweight architecture that predicts 2D joint locations from a learned camera-independent representation of 3D pose and then lifts them to 3D via an efficient formulation of differentiable triangulation (DLT). Our method achieves performance comparable to volumetric methods, while, unlike them, working in real-time.

In this paper, we demonstrate that the expense of using a 3D grid is not required. Fig. 1(b) depicts our approach. We encode each input image into latent representations, which are then efficiently transformed from image coordinates into world coordinates by conditioning on the appropriate camera transformation using feature transform layers [31]. This yields feature maps that live in a *canonical* frame of reference and are *disentangled* from the camera poses. The feature maps are fused using 1D convolutions into a unified latent representation, denoted as $p_{3\text{D}}$ in Fig. 1(b), which makes it possible to reason jointly about the extracted 2D poses across camera views. We then condition this latent code on the known camera transformation to decode it back to 2D image locations using a shallow 2D CNN. The proposed fusion technique, to which we will refer to as *Canonical Fusion*, enables us to drastically improve the accuracy of the 2D detection compared to the results obtained from each image independently, so much so, that we can lift these 2D detections to 3D reliably using the simple Direct Linear Transform (DLT) method [11]. Because standard DLT implementations that rely on Singular Value Decomposition (SVD) are rarely efficient on GPUs, we designed a faster alternative implementation based on the Shifted Iterations method [23].

In short, our contributions are: (1) a novel multi-camera fusion technique that exploits 3D geometry in latent space to efficiently and jointly reason about different views and drastically improve the accuracy of 2D detectors, (2) a new GPU-friendly implementation of the DLT method, which is hundreds of times faster than standard implementations.

We evaluate our approach on two large-scale multi-view datasets, Human3.6M [13] and TotalCapture [29]: we outperform the state-of-the-art methods when additional training data is not available, both in terms of speed and accuracy. When additional 2D annotations can be used [17, 2], our accuracy remains comparable to that of the state-of-the-art methods, while being faster. Finally, we demonstrate that our approach can handle viewpoints that were never seen during training. In short, we can achieve real-time performance without sacrificing prediction accuracy nor viewpoint flexibility, while other approaches cannot.

2. Related Work

Pose estimation is a long-standing problem in the computer vision community. In this section, we review in detail related multi-view pose estimation literature. We then focus on approaches lifting 2D detections to 3D via triangulation.

Pose estimation from multi-view input images. Early attempts [18, 9, 4, 3] tackled pose-estimation from multi-view inputs by optimizing simple parametric models of the human body to match hand-crafted image features in each view, achieving limited success outside of the controlled settings. With the advent of deep learning, the dominant paradigm has shifted towards estimating 2D poses from each view separately, through exploiting efficient monocular pose estimation architectures [20, 28, 30, 26], and then recovering the 3D pose from single view detections.

Most approaches use 3D volumes to aggregate 2D predictions. Pavlakos et al. [21] project 2D keypoint heatmaps to 3D grids and use Pictorial Structures aggregation to estimate 3D poses. Similarly, [22] proposes to use Recurrent Pictorial Structures to efficiently refine 3D pose estimations step by step. Improving upon these approaches, [14] projects 2D heatmaps to a 3D volume using a differentiable model and regresses the estimated root-centered 3D pose through a learnable 3D convolutional neural network. This allows them to train their system end-to-end by optimizing directly the 3D metric of interest through the predictions of the 2D pose estimator network. Despite recovering 3D poses reliably, volumetric approaches are computationally demanding, and simple triangulation of 2D detections is still the de-facto standard when seeking real-time performance [16, 5].

Few models have focused on developing lightweight solutions to reason about multi-view inputs. In particular, [15] proposes to concatenate together pre-computed 2D detections and pass them as input to a fully connected network to predict global 3D joint coordinates. Similarly, [22] refines 2D heatmap detections jointly by using a fully connected layer before aggregating them on 3D volumes. Although, similar to our proposed approach, these methods fuse information from different views without using volumetric grids, they do not leverage camera information and thus overfit to a specific camera setting. We will show that our approach can handle different cameras flexibly and even generalize to unseen ones.

Triangulating 2D detections. Computing the position of a point in 3D-space given its images in n views and the camera matrices of those views is one of the most studied computer vision problems. We refer the reader to [11] for an overview of existing methods. In our work, we use the Direct Linear Triangulation (DLT) method because it is simple and differentiable. We propose a novel GPU-friendly implementation of this method, which is up to two orders of magnitude faster than existing ones that are based on SVD factorization. We provide a more detailed overview about this algorithm in Section 3.4.

Several methods lift 2D detections efficiently to 3D by means of triangulation [1, 16, 10, 5]. More closely related to our work, [14] proposes to back-propagate through an SVD-based differentiable triangulation layer by lifting 2D detections to 3D keypoints. Unlike our approach, these methods do not perform any explicit reasoning about multi-view inputs and therefore struggle with large self-occlusions.

3. Method

We consider a setting in which n spatially calibrated and temporally synchronized cameras capture the performance of a single individual in the scene. We denote with $\{I_i\}_{i=1}^n$ the set of multi-view input images, each captured from a camera with known projection matrix P_i. Our goal is to estimate its 3D pose in the absolute world coordinates; we parameterize it as a fixed-size set of 3D point locations $\{\mathbf{x}^j\}_{j=1}^J$, which correspond to the joints.

Consider as an example the input images on the left of Figure 2. Although exhibiting different appearances, the

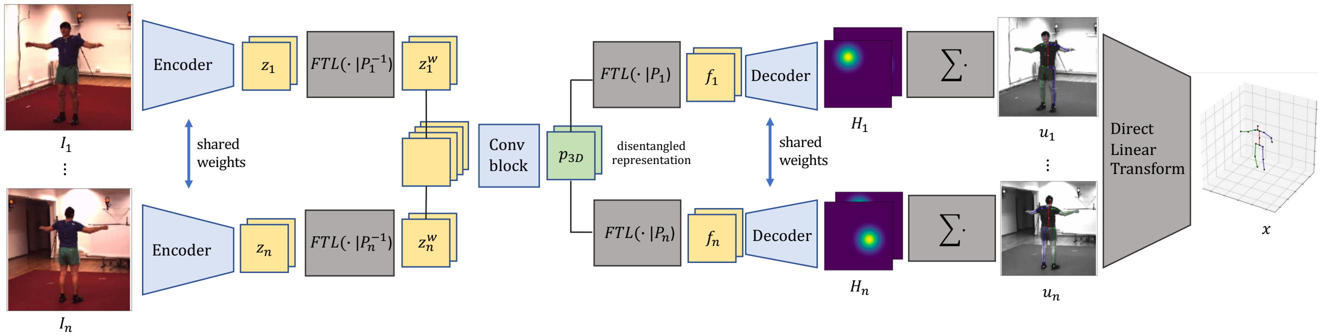


Figure 2. *Canonical Fusion.* The proposed architecture learns a unified view-independent representation of the 3D pose from multi-view inputs, allowing it to reason efficiently across multiple views. Feature Transform Layers (FTL) use camera projection matrices (P_i) to map features between this canonical representation, while Direct Linear Transform (DLT) efficiently lifts 2D keypoints into 3D. Blocks marked in gray are differentiable (supporting backpropagation) but not trainable.

frames share the same 3D pose information up to a perspective projection and view-dependent occlusions. Building on this observation, we design our architecture (depicted in Figure 2), which learns a unified *view-independent representation* of 3D pose from multi-view input images. This allows us to reason efficiently about occlusions to produce accurate 2D detections, that can be then simply lifted to 3D absolute coordinates by means of triangulation. Below, we first introduce baseline methods for pose estimation from multi-view inputs. We then describe our approach in detail and explain how we train our model.

3.1. Lightweight pose estimation from multi-view inputs

Given input images $\{I_i\}_{i=1}^n$, we use a convolutional neural network backbone to extract features $\{z_i\}_{i=1}^n$ from each input image separately. Denoting our encoder network as e, z_i is computed as

$$z_i = e(I_i). \tag{1}$$

Note that, at this stage, feature map z_i contains a representation of the 3D pose of the performer that is fully *entangled* with camera view-point, expressed by the camera projection operator P_i.

We first propose a baseline approach, similar to [16, 10], to estimate the 3D pose from multi-view inputs. Here, we simply decode latent codes z_i to 2D detections, and lift 2D detections to 3D by means of triangulation. We refer to this approach as *Baseline*. Although efficient, we argue that this approach is limited because it processes each view independently and therefore cannot handle self-occlusions.

An intuitive way to jointly reason across different views is to use a learnable neural network to share information across embeddings $\{z_i\}_{i=1}^n$, by concatenating features from different views and processing them through convolutional layers into view-dependent features, similar in spirit to the recent models [15, 22]. In Section 4 we refer to this general approach as *Fusion*. Although computationally lightweight and effective, we argue that this approach is limited for two reasons: (1) it does not make use of known camera information, relying on the network to learn the spatial configuration of the multi-view setting from the data itself, and (2) it cannot generalize to different camera settings by design. We will provide evidence for this in Section 4 .

3.2. Learning a view-independent representation

To alleviate the aforementioned limitations, we propose a method to jointly reason across views, leveraging the observation that the 3D pose information contained in feature maps $\{z_i\}_{i=1}^n$ is the same across all n views up to camera projective transforms and occlusions, as discussed above. We will refer to this approach as *Canonical Fusion*.

To achieve this goal, we leverage *feature transform layers* (FTL) [31], which was originally proposed as a technique to condition latent embeddings on a target transformation so that to learn interpretable representations. Internally, a FTL has no learnable parameter and is computationally efficient. It simply reshapes the input feature map to a point-set, applies the target transformation, and then reshapes the point-set back to its original dimension. This technique forces the learned latent feature space to preserve the structure of the transformation, resulting in practice in a disentanglement between the learned representation and the transformation. In order to make this paper more self-contained, we review FTL in detail in the Supplementary Section.

Several approaches have used FTL for novel view synthesis to map the latent representation of images or poses from one view to another [25, 24, 7, 6]. In this work, we leverage FTL to map images from multiple views to a unified latent representation of 3D pose. In particular, we use FTL to project feature maps z_i to a common canonical representation by explicitly conditioning them on the camera projection matrix P_i^{-1} that maps image coordinates to the

```
Algorithm 1: DLT-SII($\{\mathbf{u}_i, P_i\}_{i=1}^N, T = 2$)
  $A \leftarrow A(\{\mathbf{u}_i, P_i\}_{i=1}^N)$;
  $B \leftarrow (A^T A + \sigma I)^{-1}$;
  $\sigma \leftarrow 0.001$ (see Theorem 1);
  $\mathbf{x} \leftarrow \mathrm{rand}(4, 1)$;
  for $i = 1 : T$ do
      $\mathbf{x} \leftarrow B\mathbf{x}$;
      $\mathbf{x} \leftarrow \mathbf{x}/\|\mathbf{x}\|$;
  end
  return $\mathbf{y} \leftarrow \mathbf{x}(0:3)/\mathbf{x}(4)$;
```

world coordinates

$$z_i^w = \mathrm{FTL}(z_i | P_i^{-1}). \tag{2}$$

Now that feature maps have been mapped to the same canonical representation, they can simply be concatenated and fused into a *unified representation of 3D pose* via a shallow 1D convolutional neural network f, i.e.

$$p_{\mathrm{3D}} = f(\mathrm{concatenate}(\{z_i^w\}_{i=1}^n)). \tag{3}$$

We now force the learned representation to be disentangled from camera view-point by transforming the shared p_{3D} features to view-specific representations f_i by

$$f_i = \mathrm{FTL}(p_{\mathrm{3D}} | P_i). \tag{4}$$

In Section 4 we show both qualitatively and quantitatively that the representation of 3D pose we learn is effectively disentangled from the camera-view point.

Unlike the *Fusion* baseline, *Canonical Fusion* makes explicit use of camera projection operators to simplify the task of jointly reasoning about views. The convolutional block, in fact, now does not have to figure out the geometrical disposition of the multi-camera setting and can solely focus on reasoning about occlusion. Moreover, as we will show, *Canonical Fusion* can handle different cameras flexibly, and even generalize to unseen ones.

3.3. Decoding latent codes to 2D detections

This component of our architecture proceeds as a monocular pose estimation model that maps view-specific representations f_i to 2D Heatmaps H_i via a shallow convolutional decoder d, i.e.

$$H_i^j = d(f_i), \tag{5}$$

where H_i^j is the heatmap prediction for joint j in Image i. Finally, we compute the 2D location u_i^j of each joint j by simply integrating heatmaps across spatial axes

$$\mathbf{u}_i^j = \Big(\sum_{x,y} x H_i^j, \sum_{x,y} y H_i^j\Big) / \sum_{x,y} H_i^j. \tag{6}$$

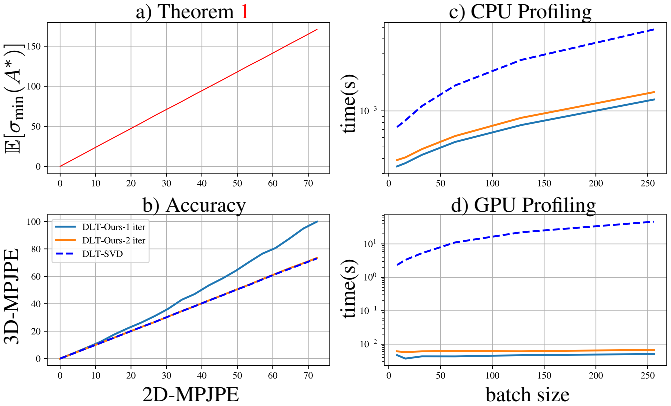


Figure 3. Evaluation of DLT. We validate the findings of Theorem 1 in (a). We then compare our proposed DLT implementation to the SVD of [14], both in terms of accuracy (b) and performance (c),(d). Exploiting Theorem 1, we can choose a suitable approximation for $\sigma_{\min}(A^*)$, and make DLT-SII converge to the desired solution in only two iterations.

Note that this operation is differentiable with respect to heatmap H_i^j, allowing us to back-propagate through it. In the next section, we explain in detail how we proceed to lift multi-view 2D detections to 3D.

3.4. Efficient Direct Linear Transformation

In this section we focus on finding the position $\mathbf{x}^j = [x^j, y^j, z^j]^T$ of a 3D point in space given a set of n 2d detections $\{\mathbf{u}_i^j\}_{i=1}^n$. To ease the notation, we will drop apex j as the derivations that follow are carried independently for each landmark.

Assuming a pinhole camera model, we can write $d_i \mathbf{u}_i = P_i \mathbf{x}$, where d_i is an unknown scale factor. Note that here, with a slight abuse of notation, we express both 2d detections $\mathbf{u}_i$ and 3d landmarks $\mathbf{x}$ in homogeneous coordinates. Expanding on the components we get

$$d_i u_i = p_i^{1T} \mathbf{x}, \; d_i v_i = p_i^{2T} \mathbf{x}, \; d_i = p_i^{3T} \mathbf{x}, \tag{7}$$

where p_i^{kT} denotes the k-th row of i-th camera projection matrix. Eliminating d_i using the third relation in (7), we obtain

$$(u_i p_i^{3T} - p_i^{1T})\mathbf{x} = 0 \tag{8}$$

$$(v_i p_i^{3T} - p_i^{2T})\mathbf{x} = 0. \tag{9}$$

Finally, accumulating over all available n views yields a total of $2n$ linear equations in the unknown 3D position $\mathbf{x}$, which we write compactly as

$$A\mathbf{x} = \mathbf{0}, \quad \text{where } A = A(\{u_i, v_i, P_i\}_{i=1}^N). \tag{10}$$

Note that $A \in \mathbb{R}^{2n \times 4}$ is a function of $\{u_i, v_i, P_i\}_{i=1}^N$, as specified in Equations (8) and (9). We refer to A as the

DLT matrix. These equations define $\mathbf{x}$ up to a scale factor, and we seek a non-zero solution. In the absence of noise, Equation (10) admits a unique non-trivial solution, corresponding to the 3D intersection of the camera rays passing by each 2D observation $\mathbf{u}_i$ (i.e. matrix A does not have full rank). However, considering noisy 2D point observations such as the ones predicted by a neural network, Equation (10) does not admit solutions, thus we have to seek for an approximate one. A common choice, known as the *Direct Linear Transform* (DLT) method [11], proposes the following relaxed version of Equation (10):

$$\min_{\mathbf{x}} \|A\mathbf{x}\|, \text{subject to} \|\mathbf{x}\| = 1. \tag{11}$$

Clearly, the solution to the above optimization problem is the eigenvector of $A^T A$ associated to its smallest eigenvalue $\lambda_{\text{min}}(A^T A)$. In practice, the eigenvector is computed by means of Singular Value Decomposition (SVD) [11]. We argue that this approach is suboptimal, as we in fact only care about *one* of the eigenvectors of $A^T A$.

Inspired by the observation above that the smallest eigenvalue of $A^T A$ is zero for non-noisy observations, we derive a bound for the smallest eigenvalue of matrix $A^T A$ in the presence of Gaussian noise. We prove this estimate in the Supplementary Section.

Theorem 1 *Let A be the DLT matrix associated to the non-perturbed case, i.e. $\sigma_{min}(A) = 0$. Let us assume i.i.d Gaussian noise $\varepsilon = (\varepsilon_u, \varepsilon_v) \sim \mathcal{N}(0, s^2 I)$ in our 2d observations, i.e. $(u^*, v^*) = (u + \varepsilon_u, v + \varepsilon_v)$, and let us denote as A^* the DLT matrix associated to the perturbed system. Then, it follows that:*

$$0 \leq \mathbb{E}[\sigma_{min}(A^*)] \leq Cs, \;\; \text{where } C = C(\{u_i, P_i\}_{i=1}^N) \tag{12}$$

In Figure 3(a) we reproduce these setting by considering Gaussian perturbations of 2D observations, and find an experimental confirmation that by having a greater 2D joint measurement error, specified by 2D-MPJPE (see Equation 13 for its formal definition), the expected smallest singular value $\sigma_{\text{min}}(A^*)$ increases linearly.

The bound above, in practice, allows us to compute the smallest singular vector of A^* reliably by means of *Shifted Inverse Iterations* (SII) [23]: we can estimate $\sigma_{\text{min}}(A^*)$ with a small constant and know that the iterations will converge to the correct eigenvector. For more insight on why this is the case, we refer the reader to the Supplementary Section.

SII can be implemented extremely efficiently on GPUs. As outlined in Algorithm 1, it consists of one inversion of a 4×4 matrix and several matrix multiplication and vector normalizations, operations that can be trivially parallelized. In Figure 3(b) we compare our SII based implementation of DLT (estimating the smallest singular value of A with $\sigma = 0.001$) to an SVD based one, such as the one proposed in [14]. For 2D observation errors up to 70 pixels (which is a reasonable range in 256 pixel images), our formulation requires as little as two iterations to achieve the same accuracy as a full SVD factorization, while being respectively $10/100$ times faster on CPU/GPU than its counterpart, as evidenced by our profiling in Figures 3(c,d).

3.5. Loss function

In this section, we explain how to train our model. Since our DLT implementation is differentiable with respect to 2D joint locations $\mathbf{u}_i$, we can let gradients with respect to 3D landmarks $\mathbf{x}$ flow all the way back to the input images $\{I_i\}_{i=1}^n$, making our approach trainable end-to-end. However, in practice, to make training more stable in its early stages, we found it helpful to first train our model by minimizing a 2D Mean Per Joint Position Error (MPJPE) of the form

$$L_{\text{2D-MPJPE}} = \sum_{i=1}^{n} \frac{1}{J} \sum_{j=1}^{J} \|\mathbf{u}_i^j - \hat{\mathbf{u}}_i^j\|_2, \tag{13}$$

where $\hat{\mathbf{u}}_j^i$ denotes the ground truth 2D position of j-th joint in the i-th image. In our experiments, we pre-train our models by minimizing $L_{\text{2D-MPJPE}}$ for 20 epochs. Then, we finetune our model by minimizing 3D MPJPE, which is also our test metric, by

$$L_{\text{3D-MPJPE}} = \frac{1}{J} \sum_{j=1}^{J} \|\mathbf{x}_j - \hat{\mathbf{x}}_j\|_2, \tag{14}$$

where $\hat{\mathbf{x}}_j$ denotes the ground truth 3D position of j-th joint in the world coordinate. We evaluate the benefits of finetuning using $L_{\text{3D-MPJPE}}$ in the Section 4.

4. Experiments

We conduct our evaluation on two available large-scale multi-view datasets, TotalCapture [29] and Human3.6M [13]. We crop each input image around the performer, using ground truth bounding boxes provided by each dataset. Input crops are undistorted, re-sampled so that virtual cameras are pointing at the center of the crop and normalized to 256×256. We augment our train set by performing random rotation(±30 degrees, note that image rotations correspond to camera rotations along the z-axis) and standard color augmentation. In our experiments, we use a ResNet152 [12] pre-trained on ImageNet [8] as the backbone architecture for our encoder. Our fusion block consists of two 1×1 convolutional layers. Our decoder consists of 4 transposed convolutional layers, followed by a 1×1 convolution to produce heatmaps. More details on our architecture are provided in the Supplementary section. The networks are trained for 50 epochs, using a Stochastic Gradient Descent optimizer where we set learning rate to 2.5×10^{-2}.

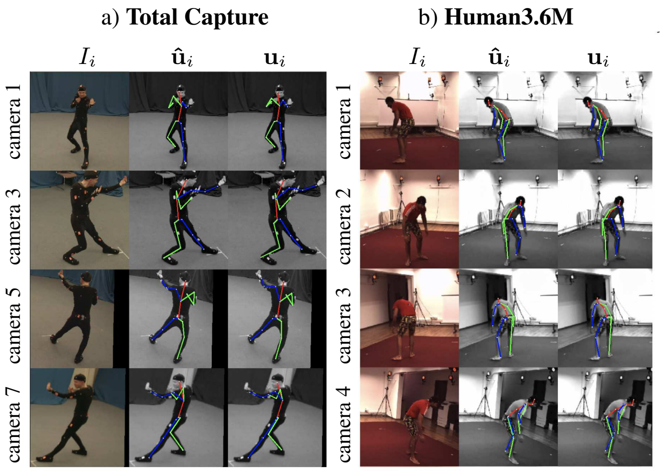


Figure 4. We visualize randomly picked samples from the test set of TotalCapture and Human3.6M. To stress that the pose representation learned by our network is effectively *disentangled* from the camera view-point, we intentionally show predictions *before* triangulating them, rather than re-projecting triangulated keypoints to the image space. Predictions are best seen in supplementary videos.

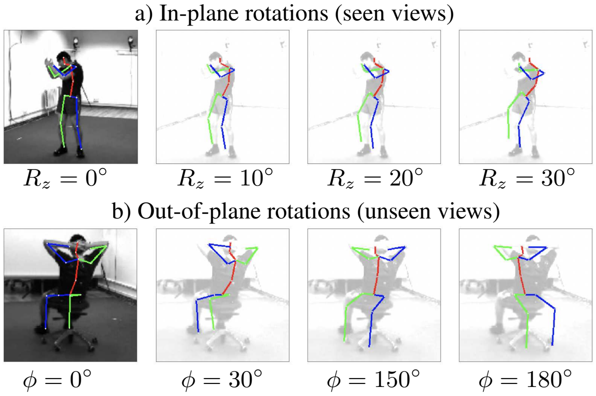


Figure 5. In the top row, we synthesize 2D poses after rotating cameras with respect to z-axis. In the bottom row, we rotate camera around the plane going through two consecutive camera views by angle ϕ, presenting the network with *unseen* camera projection matrices. Note that after decoding $p_{3\text{D}}$ to a novel view, it no longer corresponds to the encoded view. 2D Skeletons are overlaid on one of the original view in order to provide a reference. These images show that the 3D pose embedding $p_{3\text{D}}$ is *disentangled* from the camera view-point. Best seen in supplementary videos.

4.1. Datasets specifications

TotalCapture: The TotalCapture dataset [29] has been recently introduced to the community. It consists of 1.9 million frames, captured from 8 calibrated full HD video cameras recording at 60Hz. It features 4 male and 1 female subjects, each performing five diverse performances repeated 3 times: ROM, Walking, Acting, Running, and Freestyle. Accurate 3D human joint locations are obtained from a marker-based motion capture system. Following previous work [29], the training set consists of “ROM1,2,3”, “Walking1,3”, “Freestyle1,2”, “Acting1,2”, “Running1” on subjects 1,2 and 3. The testing set consists of “Walking2 (W2)”, “Freestyle3 (FS3)”, and “Acting3 (A3)” on subjects 1, 2, 3, 4, and 5. The number following each action indicates the video of that action being used, for example Freestyle has three videos of the same action of which 1 and 2 are used for training and 3 for testing. This setup allows for testing on unseen and seen subjects but always unseen performances. Following [22], we use the data of four cameras (1,3,5,7) to train and test our models. However, to illustrate the generalization ability of our approach to new camera settings, we propose an experiment were we train on cameras (1,3,5,7) and test on *unseen* cameras (2,4,6,8).

Human 3.6M: The Human3.6M dataset [13] is the largest publicly available 3D human pose estimation benchmark. It consists of 3.6 million frames, captured from 4 synchronized 50Hz digital cameras. Accurate 3D human joint locations are obtained from a marker-based motion capture system utilizing 10 additional IR sensors. It contains a total of 11 subjects (5 females and 6 males) performing 15 different activities. For evaluation, we follow the most popular protocol, by training on subjects 1, 5, 6, 7, 8 and using unseen subjects 9, 11 for testing. Similar to other methods [19, 21, 27, 15, 22], we use all available views during training and inference.

4.2. Qualitative evaluation of disentanglement

We evaluate the quality of our latent representation by showing that 3D pose information is effectively disentangled from the camera view-point. Recall from Section 3 that our encoder e encodes input images to latent codes z_i, which are transformed from camera coordinates to the world coordinates and latter fused into a unified representation $p_{3\text{D}}$ which is meant to be disentangled from the camera view-point. To verify this is indeed the case, we propose to decode our representation to different 2D poses by using different camera transformations P, in order to produce views of the same pose from novel camera view-points. We refer the reader to Figure 5 for a visualization of the synthesized poses. In the top row, we rotate one of the cameras with respect to the z-axis, presenting the network with projection operators that have been seen at train time. In the bottom row we consider a more challenging scenario, where we synthesize novel views by rotating the camera around the plane going through consecutive cameras. Despite presenting the network with unseen projection operators, our decoder is still able to synthesize correct 2D poses. This experiment shows our approach has effectively learned a representation of the 3D pose that is disentangled from camera view-point. We evaluate it quantitatively in Section 4.4.

Methods	Seen Subjects (S1,S2,S3)			Unseen Subjects (S4,S5)			Mean
	Walking	Freestyle	Acting	Walking	Freestyle	Acting	
Qui *et al*. [22] Baseline + RPSM	28	42	30	45	74	46	41
Qui *et al*. [22] Fusion + RPSM	19	**28**	21	32	**54**	**33**	29
Ours, Baseline	31.8	36.4	24.0	43.0	75.7	43.0	39.3
Ours, Fusion	14.6	35.3	20.7	28.8	71.8	37.3	31.8
Ours, Canonical Fusion(no DLT)	10.9	32.2	16.7	27.6	67.9	35.1	28.6
Ours, Canonical Fusion	**10.6**	30.4	**16.3**	**27.0**	65.0	34.2	**27.5**

Table 1. 3D pose estimation error MPJPE (mm) on the TotalCapture dataset. The results reported for our methods are obtained without rigid alignment or further offline post-processing.

Methods	Seen Subjects (S1,S2,S3)			Unseen Subjects (S4,S5)			Mean
	Walking	Freestyle	Acting	Walking	Freestyle	Acting	
Ours, Baseline	28.9	53.7	42.4	46.7	75.9	51.3	48.2
Ours, Fusion	73.9	71.5	71.5	72.0	108.4	58.4	78.9
Ours, Canonical Fusion	**22.4**	**47.1**	**27.8**	**39.1**	**75.7**	**43.1**	**38.2**

Table 2. Testing the generalization capabilities of our approach on unseen views. We take the networks of Section 4.3, trained on cameras (1,3,5,7) of the TotalCapture training set, and test on the unseen views captured with cameras (2,4,6,8). We report 3D pose estimation error MPJPE (mm).

4.3. Quantitative evaluation on TotalCapture

We begin by evaluating the different components of our approach and comparing to the state-of-the-art volumetric method of [22] on the TotalCapture dataset. We report our results in Table 1. We observe that by using the feature fusion technique (*Fusion*) we get a significant 19% improvement over our *Baseline*, showing that, although simple, this fusion technique is effective. Our more sophisticated *Canonical Fusion (no DLT)* achieves further 10% improvement, showcasing that our method can effectively use camera projection operators to better reason about views. Finally, training our architecture by back-propagating through the triangulation layer (*Canonical Fusion*) allows to further improve our accuracy by 3%. This is not surprising as we optimize directly for the target metric when training our network. Our best performing model outperforms the state-of-the-art volumetric model of [22] by $\sim$ 5%. Note that their method lifts 2D detections to 3D using Recurrent Pictorial Structures (RPSM), which uses a pre-defined skeleton, as a strong prior, to lift 2D heatmaps to 3D detections. Our method doesn't use any priors, and still outperform theirs. Moreover, our approach is orders of magnitude faster than theirs, as we will show in Section 4.6. We show some uncurated test samples from our model in Figure 4(a).

4.4. Generalization to unseen cameras

To assess the flexibility of our approach, we evaluate its performance on images captured from unseen views. To do so, we take the trained network of Section 4.3 and test it on cameras (2,4,6,8). Note that this setting is particularly challenging not only because of the novel camera views, but also because the performer is often out of field of view in camera 2. For this reason, we discard frames where the performer is out of field of view when evaluating our *Baseline*. We report the results in Table 2. We observe that *Fusion* fails at generalizing to novel views (accuracy drops by 47.1mm when the network is presented with new views). This is not surprising as this fusion technique over-fits by design to the camera setting. On the other hand the accuracy drop of *Canonical Fusion* is similar to the one of *Baseline* ($\sim$ 10mm). Note that our comparison favors *Baseline* by discarding frames when object is occluded. This experiments validates that our model is able to cope effectively with challenging unseen views.

4.5. Quantitative evaluation on Human 3.6M

We now turn to the Human36M dataset, where we first evaluate the different components of our approach, and then compare to the state-of-the-art multi-view methods. Note that here we consider a setting where no additional data is used to train our models. We report the results in Table 3. Considering the ablation study, we obtain results that are consistent with what we observed on the TotalCapture dataset: performing simple feature fusion (*Fusion*) yields a 18% improvement over the monocular baseline. A further $\sim$ 10% improvement can be reached by using *Canonical Fusion (no DLT)*. Finally, training our architecture by back-propagating through the triangulation layer (*Canonical Fusion*) allows to further improve our accuracy by 7%. We show some uncurated test samples from our model in Figure 4(b).

We then compare our model to the state-of-the-art methods. Here we can compare our method to the one of [22] just by comparing fusion techniques (see *Canonical Fusion*

Methods	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Mean
Martinez *et al*. [19]	46.5	48.6	54.0	51.5	67.5	70.7	48.5	49.1	69.8	79.4	57.8	53.1	56.7	42.2	45.4	57.0
Pavlakos *et al*. [21]	41.2	49.2	42.8	43.4	55.6	46.9	40.3	63.7	97.6	119.0	52.1	42.7	51.9	41.8	39.4	56.9
Tome *et al*. [27]	43.3	49.6	42.0	48.8	51.1	64.3	40.3	43.3	66.0	95.2	50.2	52.2	51.1	43.9	45.3	52.8
Kadkhodamohammadi *et al*. [15]	39.4	46.9	41.0	42.7	53.6	54.8	41.4	50.0	59.9	78.8	49.8	46.2	51.1	40.5	41.0	49.1
Qiu *et al*. [22]	34.8	35.8	32.7	33.5	34.5	38.2	29.7	60.7	53.1	35.2	41.0	41.6	31.9	31.4	34.6	38.3
Qui *et al*. [22] + RPSM	28.9	32.5	26.6	28.1	**28.3**	**29.3**	**28.0**	36.8	41.0	**30.5**	35.6	30.0	28.3	**30.0**	30.5	31.2
Ours, Baseline	39.1	46.5	31.6	40.9	39.3	45.5	47.3	44.6	45.6	37.1	42.4	46.7	34.5	45.2	64.8	43.2
Ours, Fusion	31.3	37.3	29.4	29.5	34.6	46.5	30.2	43.5	44.2	32.4	35.7	33.4	31.0	38.3	32.4	35.4
Ours, Canonical Fusion (no DLT)	31.0	35.1	28.6	29.2	32.2	34.8	33.4	32.1	35.8	34.8	33.3	32.2	29.9	35.1	34.8	32.5
Ours, Canonical Fusion	**27.3**	**32.1**	**25.0**	**26.5**	29.3	35.4	28.8	**31.6**	**36.4**	31.7	**31.2**	**29.9**	**26.9**	33.7	**30.4**	**30.2**

Table 3. No additional training data setup. We compare the 3D pose estimation error (reported in MPJPE (mm)) of our method to the state-of-the-art approaches on the Human3.6M dataset. The reported results for our methods are obtained without rigid alignment or further offline post-processing steps.

(no DLT) vs Qui *et al*. [22] (no RPSM) in Table 3). We see that our methods outperform theirs by $\sim 15\%$, which is significant and indicates the superiority of our fusion technique. Similar to what observed in Section 4.3, our best performing method is even superior to the off-line volumetric of [22], which uses a strong bone-length prior (Qui *et al*. [22] Fusion + RPSM). Our method outperforms all other multi-view approaches by a large margin. Note that in this setting we cannot compare to [14], as they do not report results without using additional data.

4.6. Exploiting additional data

Methods	Model size	Inference Time	MPJPE
Qui *et al*. [22] Fusion + RPSM	2.1GB	8.4s	26.2
Iskakov *et al*. [14] Algebraic	320MB	2.00s	22.6
Iskakov *et al*. [14] Volumetric	643MB	2.30s	**20.8**
Ours, Baseline	**244MB**	**0.04s**	34.2
Ours, Canonical Fusion	251MB	**0.04s**	21.0

Table 4. Additional training data setup. We compare our method to the state-of-the-art approaches in terms of performance, inference time, and model size on the Human3.6M dataset.

To compare to the concurrent model in [14], we consider a setting in which we exploit additional training data. We adopt the same pre-training strategy as [14], that is we pretrain a monocular pose estimation network on the COCO dataset [17], and fine-tune jointly on Human3.6M and MPII [2] datasets. We then simply use these pre-trained weights to initialize our network. We also report results for [22], which trains its detector jointly on MPII and Human3.6M. The results are reported in Table 4.

First of all, we observe that *Canonical Fusion* outperforms our monocular baseline by a large margin ($\sim 39\%$). Similar to what was remarked in the previous section, our method also outperforms [22]. The gap, however, is somewhat larger in this case ($\sim 20\%$). Our approach also outperforms the triangulation baseline of (Iskakov *et al*. [14] Algebraic), indicating that our fusion technique if effective in reasoning about multi-view input images. Finally, we observe that our method reaches accuracy comparable to the volumetric approach of (Iskakov *et al*. [14] Volumetric).

To give insight on the computational efficiency of our method, in Table 4 we report the size of the trained models in memory, and also measure their inference time (we consider a set of 4 images and measure the time of a forward pass on a Pascal TITAN X GPU and report the average over 100 forward passes). Comparing model size, *Canonical Fusion* is much smaller than other models and introduces only a negligible computational overhead compared to our monocular *Baseline*. Comparing the inference time, both our models yield a real-time performance ($\sim 25 fps$) in their un-optimized version, which is much faster than other methods. In particular, it is about 50 times faster than (Iskakov *et al*. [14] Algebraic) due to our efficient implementation of DLT and about 57 times faster than (Iskakov *et al*. [14] Volumetric) due to using DLT plus 2D CNNs instead of a 3D volumetric approach.

5. Conclusions

We propose a new multi-view fusion technique for 3D pose estimation that is capable of reasoning across multiview geometry effectively, while introducing negligible computational overhead with respect to monocular methods. Combined with our novel formulation of DLT transformation, this results in a real-time approach to 3D pose estimation from multiple cameras. We report the state-of-the-art performance on standard benchmarks when using no additional data, flexibility to unseen camera settings, and accuracy comparable to far-more computationally intensive volumetric methods when allowing for additional 2D annotations.

6. Acknowledgments

We would like to thank Giacomo Garegnani for the numerous and insightful discussions on singular value decomposition. This work was completed during an internship at Facebook Reality Labs, and supported in part by the Swiss National Science Foundation.

References

[1] Sikandar Amin, Mykhaylo Andriluka, Marcus Rohrbach, and Bernt Schiele. Multi-view pictorial structures for 3d human pose estimation. In *Bmvc*, volume 2, page 7. Citeseer, 2013. 2

[2] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 2, 8

[3] Vasileios Belagiannis, Sikandar Amin, Mykhaylo Andriluka, Bernt Schiele, Nassir Navab, and Slobodan Ilic. 3d pictorial structures for multiple human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1669–1676, 2014. 2

[4] Magnus Burenius, Josephine Sullivan, and Stefan Carlsson. 3d pictorial structures for multiple view articulated pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3618–3625, 2013. 2

[5] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017. 2

[6] Xipeng Chen, Kwan-Yee Lin, Wentao Liu, Chen Qian, and Liang Lin. Weakly-supervised discovery of geometry-aware representation for 3d human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10895–10904, 2019. 3

[7] Xu Chen, Jie Song, and Otmar Hilliges. Monocular neural image based rendering with continuous view control. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4090–4100, 2019. 3

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 5

[9] Juergen Gall, Bodo Rosenhahn, Thomas Brox, and Hans-Peter Seidel. Optimization and filtering for human motion capture. *International journal of computer vision*, 87(1-2):75, 2010. 2

[10] Semih Günel, Helge Rhodin, Daniel Morales, João Campagnolo, Pavan Ramdya, and Pascal Fua. Deepfly3d: A deep learning-based approach for 3d limb and appendage tracking in tethered, adult drosophila. *bioRxiv*, page 640375, 2019. 2, 3

[11] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2, 5

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[13] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014. 2, 5, 6

[14] Karim Iskakov, Egor Burkov, Victor Lempitsky, and Yury Malkov. Learnable triangulation of human pose. *arXiv preprint arXiv:1905.05754*, 2019. 1, 2, 4, 5, 8

[15] Abdolrahim Kadkhodamohammadi and Nicolas Padoy. A generalizable approach for multi-view 3d human pose regression. *ArXiv*, abs/1804.10462, 2018. 2, 3, 6, 8

[16] Xiu Li, Zhen Fan, Yebin Liu, Yipeng Li, and Qionghai Dai. 3d pose detection of closely interactive humans using multiview cameras. *Sensors*, 19(12):2831, 2019. 2, 3

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2, 8

[18] Yebin Liu, Carsten Stoll, Juergen Gall, Hans-Peter Seidel, and Christian Theobalt. Markerless motion capture of interacting characters using multi-view image segmentation. In *CVPR 2011*, pages 1249–1256. IEEE, 2011. 2

[19] Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little. A simple yet effective baseline for 3d human pose estimation. In *ICCV*, 2017. 6, 8

[20] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016. 2

[21] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Harvesting multiple views for marker-less 3D human pose annotations. In *Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 6, 8

[22] Haibo Qiu, Chunyu Wang, Jingdong Wang, Naiyan Wang, and Wenjun Zeng. Cross view fusion for 3d human pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1, 2, 3, 6, 7, 8

[23] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical mathematics*, volume 37. Springer Science & Business Media, 2010. 2, 5

[24] Helge Rhodin, Victor Constantin, Isinsu Katircioglu, Mathieu Salzmann, and Pascal Fua. Neural scene decomposition for multi-person motion capture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7703–7713, 2019. 3

[25] Helge Rhodin, Mathieu Salzmann, and Pascal Fua. Unsupervised geometry-aware representation for 3d human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 750–767, 2018. 3

[26] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *arXiv preprint arXiv:1902.09212*, 2019. 2

[27] Denis Tome, Matteo Toso, Lourdes Agapito, and Chris Russell. Rethinking pose in 3d: Multi-stage refinement and recovery for markerless motion capture. In *2018 International Conference on 3D Vision (3DV)*, pages 474–483. IEEE, 2018. 6, 8

[28] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015. 2

[29] Matthew Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton, and John Collomosse. Total capture: 3d human pose estimation fusing video and inertial sensors. In *BMVC*, volume 2, page 3, 2017. 2, 5, 6

[30] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016. 2

[31] Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Interpretable transformations with encoder-decoder networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5726–5735, 2017. 1, 3