Structured Output Prediction and Learning for Deep Monocular 3D Human Pose Estimation

Stefan Kinauer*[1], Riza Alp Güler*[1], Siddhartha Chandra[1], and Iasonas Kokkinos[2]

[1] CentraleSupélec, INRIA Saclay
[2] Facebook AI Research

Abstract. In this work we address the problem of estimating 3D human pose from a single RGB image by blending a feed-forward CNN with a graphical model that couples the 3D positions of parts. The CNN populates a volumetric output space that represents the possible positions of 3D human joints, and also regresses the estimated displacements between pairs of parts. These constitute the 'unary' and 'pairwise' terms of the energy of a graphical model that resides in a 3D label space and delivers an optimal 3D pose configuration at its output. The CNN is trained on the 3D human pose dataset 3.6M, the graphical model is trained jointly with the CNN in an end-to-end manner, allowing us to exploit both the discriminative power of CNNs and the top-down information pertaining to human pose. We introduce (a) memory efficient methods for getting accurate voxel estimates for parts by blending quantization with regression (b) employ efficient structured prediction algorithms for 3D pose estimation using branch-and-bound and (c) develop a framework for qualitative and quantitative comparison of competing graphical models. We evaluate our work on the Human 3.6M dataset, demonstrating that exploiting the structure of the human pose in 3D yields systematic gains.

1 Introduction

Human pose estimation has made rapid progress thanks to deep learning, as witnessed by the improvements reported on large-scale benchmarks [1,2,3,4,5,6,7]. In this work we focus on the more challenging task of 3D human pose estimation from a single monocular image, which can have many applications in human-computer interaction, augmented reality, and can eventually lead to addressing generic 3D object pose estimation.

Recovering 3D information from a single 2D image is clearly ill-posed, given that different 3D scenes can project to the same 2D image. However, exploiting task-specific prior knowledge can increase the probability of the more plausible scenes. All leading approaches to 3D human pose estimation, such as [8,9,10,11], rely on incorporating prior knowledge about the structure of the 3D human body. Two-stage approaches, e.g. [9,10,12], firstly detect joint positions in 2D and subsequently lift joints into 3D by relying on prior knowledge about the 3D human pose. The advantage of such approaches is that they can exploit large datasets constructed for the prediction of 2D landmarks - the disadvantage is that errors in the 2D stage can propagate to the 3D predictions and can often not be recovered from. Inherently 3D approaches [13,14] discretize the depth

* Kinauer, S. and Guler, R.A. contributed equally to this work.

variable and train a CNN to score every possible combination of position and depth with respect to the presence of a joint - one can understand that the CNN learns to use the scale of the joint to guess its depth. This was recently shown in [13] to deliver results that are largely superior over previous 2-stage approaches. More recent works have delivered further improvements by fusing the 2D and 3D streams [15].

In directly regressing the pose from the input image, the aforementioned approaches do not explicitly impose constraints that exploit the dependencies between the human joints. In our understanding, what is missing from existing works is a method to exploit the structure of the human pose in a natively 3D setup. Authors in [16] acknowledge this deficiency of contemporary methods, and propose to use a stacked denoising auto-encoder to learn these dependencies implicitly. Other approaches to combining structured prediction with deep learning have recently been successfully pursued in 2D human pose estimation e.g. [17,18], while current approaches to incorporating structure in feedforward CNNs for pose estimation rely on cascading, or stacking the outputs of CNNs in 2D [5,6], which can become prohibitive when done in 3D, due to the increased memory and computation load. In this work we develop novel techniques that allow us to 'explicitly' capture the dependencies between human joints via an energy function that consists of unary and pairwise terms, and thereby pursue this direction in the arguably harder 3D setting.

Our contribution consists in showing that one can combine a volumetric representation with a structured model that imposes constraints between the relative positions of parts. Rather than relying exclusively on a feed-forward architecture, we show that one can append a structured prediction algorithm that propagates information on a graph that represents the part positions, and still remain computationally efficient. In particular, we train a CNN to not only populate the 'unary' 3D score maps for each part, but also to provide estimates of the relative positions of parts in 3D. This coupling of the part positions results in an optimization problem, that we treat as yet-another layer of a deep network, and thereby add functionality to the network.

We can describe our contributions as (i) allowing for a high resolution in the estimation of the pose without increasing the computation/memory budget (Sec. 2.1) (ii) constructing the coupling term so as to make optimization tractable while working with a high resolution in the depth coordinates (Sec. 2.2) (iii) exploiting rich connectivity, i.e. allowing the edges in the underlying graph of the model to form loops, by using fast approximate inference that combines Branch-and-Bound with ADMM (Sec. 2.3) and (iv) using Deeply Supervised Network (DSN) [19] training to accelerate training, by forcing the 2D and 3D inputs to structured prediction to reflect part of the solution, while also leveraging on larger 2D datasets during training (Sec. 2.4).

We evaluate our approach on the Human3.6M dataset. Our results indicate that blending local information about the 3D pose of parts with information about the 3D displacements of parts yields systematic improvements over approaches that rely on either of these two cues alone, while incurring a marginal computational overhead, in the order of a fraction of a second.

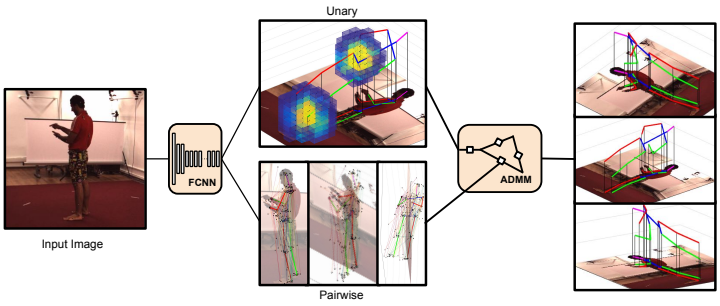

Fig. 1. We consider the task of 3D human pose estimation from a single RGB image. Our approach involves a fully convolutional neural network that provides a 'bottom-up' estimate of the 3D positions of parts and their relative displacements, and a structured prediction layer that combines them into a coherent estimate of the pose. The whole architecture is trained end-to-end, allowing us to optimize the CNN outputs with the respect to the subsequent pose estimation algorithm.

2 Method

We start by formulating our approach in terms of a structured prediction problem, and then provide the details about the individual components of our proposed approach. We represent the pose Φ in terms of the concatenation of the 3D coordinates of N individual parts ϕ_i

$$\Phi = \{\phi_1, \ldots, \phi_N\}. \tag{1}$$

Given an image I, we score a candidate pose in terms of a graphical model that considers individual properties of parts, as well as properties of some of their pairwise combinations:

$$S_I(\Phi) = \sum_{i=1}^{N} \mathcal{U}_i(\phi_i) + \sum_{i,j \in \mathcal{E}} \mathcal{P}_{i,j}(\phi_i, \phi_j), \tag{2}$$

where $\mathcal{U}$ stands for unary and $\mathcal{P}$ for pairwise potentials, and $\mathcal{E}$ is the set of edges used in our graphical model. The unary and pairwise terms are delivered by the CNN, while the structured prediction layer couples the parts through the optimization of Eq. 2. If we consider a generic cost function, this can be challenging even for simple cases, let alone for the 3D pose space we are working with. Our main technical contributions aim at making the construction and optimization of Eq. 2 tractable while still exploiting the structure of the output space.

2.1 Quantized Regression for Depth Estimation

One of the main challenges in constructing a volumetric CNN is that the amount of memory and computation scales linearly in the granularity of the depth quantization, requiring to tradeoff accuracy for speed/memory. The root of the problem is that the underlying quantity is continuous, but plain regression-based models may be neither

sufficiently accurate, nor expressive enough to capture the uncertainty and multimodality of the depth value caused by depth ambiguity, or occlusion.

Instead, we follow recent successful developments in object detection [20,21], dense correspondence estimation [22], and pose estimation [23] where a combination of classification and regression is used to attack the image-based regression problem. We use a first classification stage to associate a confidence value with a set of non-overlapping depth intervals, corresponding to a coarse quantization of the depth value. If we have N classes and a depth range of, say D units, the k-th class is associated with a quantized depth of $q_k = k\frac{D}{N}$. This however may be at a very coarse depth resolution. We refine this coarse estimate by combining it with the results af a regression layer that aims at recovering the residual between the ground-truth depth values and their quantized depth estimates.

As shown in Fig. 2 this strategy allows us to 'retarget' the voxels to 3D positions that lie closer to the actual part positions, without requiring the exhaustive sampling of the 3D space. In particular a voxel v lying at the k-th depth interval will become associated with a novel 3D position of part i, $p_i^v = k\frac{D}{N} + r_i(v)$, where $r_i(v)$ is the residual regressed by our network for the i-th part type at voxel v.

The value of the associated unary terms, $U_i(p_i^v)$, is obtained in terms of the inner product between a joint-specific weight vector, w_i and a feature vector extracted from the CNN's output at the 2D position associated with voxel v.

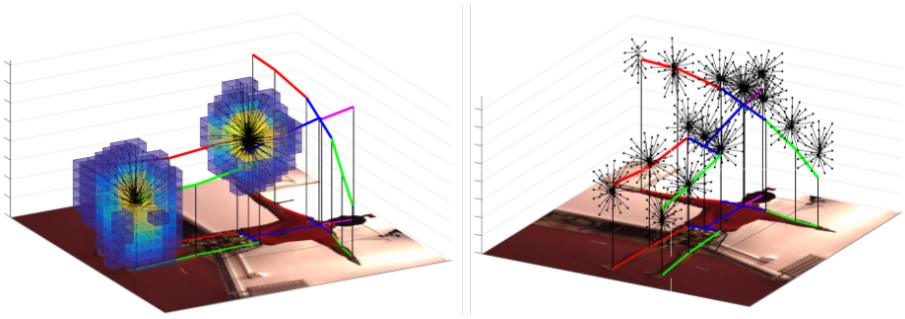

Fig. 2. Unary 3D coordinates via quantized regression. To efficiently regress the unary 3D coordinates, we use a divide and conquer strategy. We begin by quantizing the 3D space into voxels. We estimate the score of each joint belonging to each of these voxels using a classifier. Finally we regress a residual vector per voxel which indicates the offset between the center of the voxel and the continuous 3D position of each joint. *Left:* Sigmoid function on classified voxels and regressed residual vectors (in black) for two joints. *Right:* Regressed residual vectors for all joints.

2.2 Efficient Optimization with Quadratic Pairwise Terms

Having described how the unary terms are constructed in our model, we now turn to the pairwise terms and the resulting optimization problems. The expression for the pairwise term in Eq. 2, $\mathcal{P}_{i,j}(\phi_i, \phi_j, I)$ would suggest constructing a six-dimensional function.

Instead, as in the Deformable Part Model paradigm [24], we use a pairwise term that penalizes devations from a nominal displacement $\mu_{i,j}$:

$$\mathcal{P}_{i,j}(\phi_i, \phi_j, I) = -\sum_{d=1}^{3} c_d(\phi_{d,i} - \phi_{d,j} - \mu_{d,i,j})^2, \tag{3}$$

where the c_d parameters allow us to calibrate the importance of the different dimensions. These parameters are forced to be positive, while the expression in 3 corresponds to the log-probability under an axis-aligned Gaussian model, centered at the predicted part position. We note that as in [25,26], $\mu_{i,j}$ is image-dependent, and in our case is the output of a sub-network which is trained end-to-end. This enables us to capture dependencies between parts, where an estimate of their ideal displacement is combined with the local evidence provided by their unary terms.

One important advantage of the pairwise terms is that since they encode the relative position of parts they are often easier to model, since e.g. the distance between human joints is much more predictable than the actual positions of the joints. As such they can simplify the overall problem.

Another crucial advantage of the particular form of the pairwise term is that by virtue of being in the form of a quadratic cost function, it can easily be bounded from above and below using interval arithmetic - in particular, we rely on the 3D Branch-and-Bound algorithm introduced recently in [27] to efficiently search over optimal combinations of parts in 3D. A brute-force, dynamic programming-type algorithm for solving this task would require a quadratic number of operations, since it would need to compare pairs of points. Our implementation has a low-constant linear complexity for the construction of per-part KD-tree data structures, and logarithmic best-case complexity for the subsequent optimization. In practice optimization requires less than a tenth of a second on a CPU, while further accelerations could be obtained through GPU-based implementations.

2.3 Network Connectivity: from star-shaped to loopy graphs

The Branch-and-Bound (BB) algorithm we use for efficient inference only accommodates a star-shaped graph topology. This can be problematic if one wants to model human pose in terms of a tree-structured graph, or introduce loops to capture more constraints. For this we employ master-slave type approximate inference techniques that allow us to use BB for slave problems and coordinate them through a master. In particular we rely on the Alternating Direction Method of Multipliers (ADMM) [28,29,30] which matches the continuous nature of the pose estimation problem [30]. The approach to subdivide difficult problems into smaller and easier ones has before been seen in [31]. The authors introduced Dual Decomposition to optimize MRF-type energies, outperforming former state of the art of "tree-reweighted message passing" algorithms. Later works on ADMM like [30] borrow from developments outlined in [28] to reach convergence in a lower number of iterations. Loopy graphs are subdivided into easier to handle trees and coordinated via a master problem, which turns out to be updating the dual variables.

The method we outline below uses approximate inference to obtain solutions in $\omega(T \log N)$ operations, where T is a low constant in the order of tens, N is the number of voxels, and $\log N$ is the cost of re-solving the slave subproblems. The ω (best-case)

notation relates to the (exact) Branch-and-Bound algorithm, which also empirically has typically this performance. Even though the ADMM-based results are now only approximately optimal, the cost function being optimized reflects more accurately the problem structure, which can positively affect accuracy.

We consider the case where the set of graph edges in Eq. 2 corresponds to a graph with loops. Denoting by $R \subset 1...K$ the subset of point indices belonging to more than one star graph, our optimization problem can equivalently be rewritten as follows:

$$\max S(\Phi) = \sum_{i=1}^{N} S_i(\Phi_i) \quad \text{s.t.} \quad \Phi_i(r) = u(r) \quad \forall r \in R, \tag{4}$$

where S_i is a set of loop-free subproblems, defined so that $S(\Phi) = \sum_{i=1}^{N} S_i(\Phi_i)$ for a common solution Φ. The consistency is enforced by the 'master', to whom the 'slave' subproblems S_i deliver their solutions Φ_i - obtained through Branch-and-Bound. In particular a relaxation to the constraints is updated and used to reset the problem solved by the slaves - at each step the relaxation becomes tighter and at convergence consistency is guaranteed. Dual Decomposition relaxes the constraints in Eq. 4 by introducing a Lagrange Multiplier $\lambda_i(r)$ for each agreement constraint. ADMM augments this with a quadratic constraint violation penalty resulting in an *augmented Lagrangian* function:

$$\mathcal{A}(\Phi, u, \lambda) = \sum_{i=1}^{N}(S_i(\Phi_i) + \sum_{r \in R}\langle \lambda_i(r), \Phi_i(r)\rangle) - \sum_{r \in R}((\sum_{i=1}^{N}\lambda_i(r))u(r) - \frac{\rho}{2}\sum_{i=1}^{N}(\Phi_i(r) - u(r))^2) \tag{5}$$

where ρ is a positive parameter that controls the intensity of the augmenting penalty. The quadratic term ensures rapid convergence by acting like a regularizer of the solutions found across different iterations. To maximize the augmented Lagrangian, ADMM iteratively performs the following steps:

$$\Phi_i^{t+1} = \text{argmax}_{\Phi_i}\mathcal{A}(\Phi_i, u^t, \lambda^t) \tag{6}$$

$$u^{t+1} = \text{argmax}_{u}\mathcal{A}(\{\Phi_i^{t+1}\}, u, \lambda^t) \tag{7}$$

$$\lambda_i^{t+1}(r) = \lambda_i^t(r) - \rho(\Phi_i^{t+1}(r) - u^{t+1}(r)) \tag{8}$$

In words, the slaves efficiently solve their sub-problems and update the master about Φ_i, then the master sets $u^{t+1}(r)$, and the current multipliers $\lambda_i^{t+1}(r)$, and communicates them back to the slaves for the next iteration. Unlike [30] who used dynamic programming to efficiently solve the slave problems, here we combine ADMM with the Branch-and-Bound algorithm. Interestingly, both of the additional terms contributed by the master problem to the slave problems, $\lambda_i(r)u(r)$, $(\Phi_i(r) - u(r))^2$ can be easily bounded using interval arithmetic, allowing for a straightforward incorporation into the original Branch-and-Bound method. With these changes we have observed similar convergence behavior as the one reported in [30]; In typically 15-20 (sometimes even less) ADMM iterations the slaves converge to a consistent pose estimate.

2.4 Deeply Supervised 2D- and 3D- Learning

We have observed substantial simplifications in the learning procedure by employing Deeply Supervised Network (DSN) [19] training. In particular we use loss functions that directly operate on the unary and pairwise terms, before these are integrated through structured prediction. We empirically observed that this substantially accelerates and robustifies learning, by helping the network come up with good 'proposals' to the subsequent combination stage.

As discussed in Sec.2.1, the unary coordinates are obtained by adding the quantization and regression signals. Rather than expect this result to be correctly obtained only by back-propagation from the last layer, we also associate a classification and regression problem with each 2D image position.

We associate every pixel with a set of discrete labels corresponding to quantized depth values. For each joint we learn a different classification function; we consider a voxel as being positive if the respective joint is within certain proximity to the center of the 3D volume. We train this classifier using the cross-entropy loss. We also regress residual vectors between voxel centers and groundtruth joints using an L1 loss which is only active when a voxel is close enough to 3D landmarks.

For the pairwise terms, we regress vectors that point from each 3d joint to others. Similar to the unary coordinates, we regress these quantities in a fully-convolutional manner. The smooth L1 loss for the pairwise offsets between a specific joint and the rest of the joints is only active on pixels within certain proximity to the specific joint.

2.5 Training with a Structured Loss Function

Having outlined our cost function and our optimization algorithm, we now turn to parameter estimation. Our graphical model is defined in Eq. 2, and the pairwise terms are described in Eq. 3. As outlined in the preceding sub-sections, our network generates the unary terms $U_i(p_i^v)$, the nominal displacements $\mu_{i,j}$ and the 3D coordinates ϕ_i. In this section we describe training of all these parameters, as well as the calibration parameters c in Eq. 3, using a structured loss function [32,33,30] that reflects the geometric nature of the problem we want to address. Once our loss function is defined, back-propagation can be used to update all of the underlying network parameters.

While authors in [34] use an Intersection-over-Union (IoU) based structured loss for the task of detection, given that in this setup we have access to continuous ground truth values that naturally capture the underlying geometry of the problem, we opt for simplicity and use a more straightforward structured loss function.

Given that Φ denotes the 3D coordinates for a candidate configuration of parts (Eq. 1), and $\hat{\Phi}$ denotes the groundtruth 3D coordinates, we use the Mean Euclidean Distance, $\Delta(\hat{\Phi}, \Phi) = \frac{1}{P}\sum_{p=1}^{P} \|\phi_p - \hat{\phi}_p\|_2$ as a loss for our learning task, penalizing the 3D displacement of our estimated landmarks from their ground truth positions. As in standard structured output prediction, we use this loss to induce a set of constraints in pose space:

$$S(\hat{\Phi}) > S(\Phi) + \Delta(\hat{\Phi}, \Phi) \;\; \forall \Phi, \tag{9}$$

requiring that the score of the ground truth configuration should be greater than the score of any other configuration by a margin depending on how far the particular configuration

is from the ground truth.
Since this cannot hold in general, we introduce slack variables ξ: $\xi(\Phi) = max(S(\Phi) + \Delta(\hat{\Phi}, \Phi) - S(\hat{\Phi}), 0)$. Thus, the slack variables represent the violations of the constraints in Eq. 9, and our goal here is to learn the model parameters that minimize the slack variables.
Standard training of structural SVMs [32,33,30] typically finds the most violated configuration given by $\Phi^* = \text{argmax}_{\Phi}(S(\Phi) + \Delta(\hat{\Phi}, \Phi) - S(\hat{\Phi}))$ and tries to reduce the violation of this configuration by updating the model parameters appropriately via the cutting-planes or Franke-Wolfe algorithm. In this work we use the standard stochastic gradient algorithm to minimize these slack variables. We do so by first finding K most violated configurations for each input sample (K is a hyper-parameter which affects the convergence speed; we set $K = 20$ based on experiments on a validation set). We then compute the sub-gradients of the model parameters with respect to each of these violated constraints and back-propagate them through the network.

3 Experimental Evaluation

3.1 Network Architecture

In our experiments we use a fully-convolutional 151 layer ResNet, with weights initialised from a model pre-trained on MPII for 2D body pose estimation [3]. Both that 3D and 2D branches of our network are implemented as single-level convolution layers branching from the last layer of the ResNet. The input images to the system are cropped and rescaled to a fixed size of 320x320; the downsampling factor of our network is 16, leading to a cube of 20x20x20 dimensions for 3D unary detection and residual regression branches and 20x20 spatial dimensions for the 2D branches.

3.2 Dataset

We use the largest available 3D human pose dataset Human3.6M (29) to train and evaluate our approach. The dataset consists of 3.6 million video frames of daily life activities performed by actors whose 3D joint locations are recorded by motion capture systems. Following the recent works in the literature, we have used frames from subjects S1, S5, S6, S7 and S8 for training and S9 and S11 for testing. We have used frames from all 4 cameras and all 15 actions in our training and testing in an action-agnostic manner. We have sub-sampled the videos at 10 frames per second. Several videos that suffer from *drift* of the groundtruth joints are removed from the dataset.

3.3 Joint Training with 2D Pose

Our network is initialized with ResNet parameters obtained by training for 2D joint localization on the MPII dataset, but we observe that including samples from MPII as training samples increases performance - apparently not doing so results in the network forgetting about 2D joint localization. As in [35] we modify the labelled joints of the Human3.6m dataset in order to be able to utilize the 2D data. In particular we include

a joint of "thorax" between shoulders that is connected to the "neck" and discarding "chin" and "abdomen" joints. The resulting skeleton structure is identical to the one of MPII. We have verified that two identical networks trained with baseline and MPII-type label structures lead to equivalent evaluation scores, thus it is fair to compare to existing methods. The active losses for an MPII sample are 2D detection and X and Y pairwise offset values, while the 3D position estimates are ignored.

3.4 Results

Since our groundtruth comes in the form of projected coordinates, we can obtain the 3D pose only up-to a similarity transform. We report "reconstruction error", which is measured as the mean euclidean distance to the ground truth, after applying Procrustes analysis.

	Directions	Discussion	Eating	Greeting	Phoning	Photo	Posing	Purchases
UNARY alone	49.69	49.45	47.77	50.69	54.80	57.35	43.76	44.11
center star	49.41	49.26	47.35	49.93	50.97	56.12	43.62	**43.43**
stick figure	49.13	49.19	47.15	49.70	50.50	55.57	43.53	43.59
extended stick figure	49.16	49.07	47.35	49.82	50.67	55.45	43.60	43.57
2-hop	**48.89**	**48.75**	**47.07**	**49.40**	**49.82**	**55.31**	**43.30**	43.47
	Sitting	Sit. Down	Smoking	Waiting	Walk Dog	Walking	Walk Tog.	Average
UNARY alone	65.39	95.76	53.53	46.27	51.53	41.59	49.52	53.48
center star	61.50	**78.09**	52.51	45.88	50.63	41.08	49.41	51.42
stick figure	60.14	79.46	51.52	45.74	50.59	40.73	49.33	51.12
extended stick figure	**59.94**	78.51	**51.42**	46.01	50.39	40.89	49.32	51.08
2-hop	60.48	78.20	51.69	**45.63**	**50.16**	**40.74**	**49.17**	**50.87**

Table 1. Comparison of average reconstruction errors for different graph topologies.

We experimented with a number of graph topologies and notice that performance depends on the graph structure: **center star** describes the graph topology where all joints are connected to one central root node at the human's torso. It performs better than "unary only", indicating that the body center "knows" something about the other body parts.

stick figure is a graph that directly corresponds to the human skeleton, i.e. the wrist is connected to the elbow, the elbow is connected to the shoulder, and so on. Clearly the shoulder knows better where the elbow has to be than the root node in the torso. This structure clearly performs better than "center star".

extended stick figure is an extension to "stick figure", containing all its edges plus additional connections between the elbows of left and right arm, left and right knee, head to shoulders and torso to knees. This shows that additional loops boost performance, stabilizing against outliers or false evidence.

2-hop follows the human skeleton like "stick figure" and adds connections from every joint to its indirect (2-)neighbours in the skeleton. This connects, for example, hand with shoulder and ankle to hips and left to right knee. "2-hop" performs best, helping to resolve occlusions and improving accuracy.

	Average error
Yasin et al. [36]	108.3
Rogez et al. [37]	88.1
Tome et al. [8]	70.7
Pavlakos et al. [13] [3]	53.2
(Ours)Unary	53.48
(Ours)ADMM	50.87

Table 2. A comparison of our approach to methods that report reconstruction error in literature.

	cam1	**cam2**	**cam3**	**cam4**	**Average**
S-9	55.62	51.24	56.10	55.22	54.54
S-11	51.14	42.86	47.83	41.90	45.91
Average	53.72	47.64	52.59	49.57	

Table 3. Reconstruction errors for videos for specific cameras and test subjects in the Human 3.6M dataset.

Our experiments, reported in Table 1 clearly indicate that the 2-hop graph topology outperforms all of the other structures that we experimented with. This indeed justifies using approximate inference (ADMM), since these results require employing a loopy graph.

In Table.2 we compare the performance of our method to existing methods. Our results indicate that (a) our quantizatoin + regression-based unary network already delivers excellent results, at the level of the current state of the art. (b) Structured prediction yields an additional, quite substantial boost.

We note that there are some methods that only use a single camera or only S-11 frames as test samples and the rest of the videos for training. In order to compare our approach to such works, we present our results per camera and per subject in Tab.3. Our results show that we are also outperforming the very recent work of [35], who uses only S-11 as test set and obtains 48.3, which is inferior with respect to our S-11 result(45.91), even though we have not used S-9 for training.

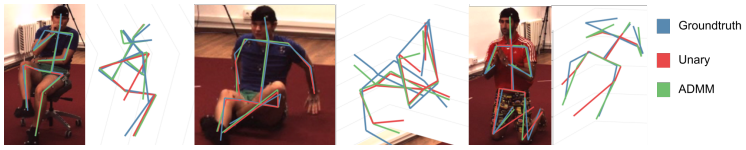

Fig. 3. Examplar pose estimates by ADMM inference: Blue indicates the ground truth pose, whereas red and green is the solution obtained from "unaries alone" and ADMM respectively.

We provide qualitative results in Fig.3, demonstrating cases where the ADMM inference clearly increases the pose estimation performance. Figure 4 shows some example images from the LSP dataset in the left column, augmented with the inferred body skeleton. The other three columns illustrate the plausible 3D structure as inferred by our approach.

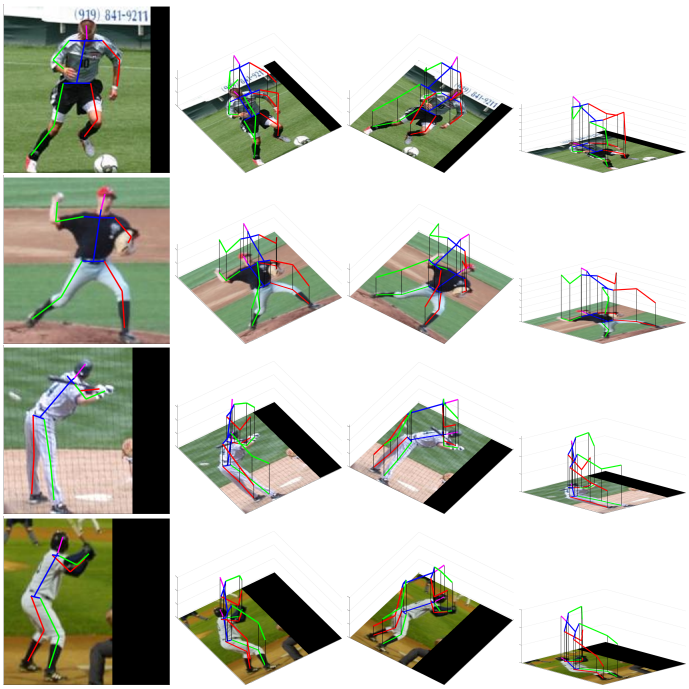

Fig. 4. Monocular 3D pose estimation results on LSP dataset.

4 Conclusion

In this work we have introduced an efficient method for 3D human pose estimation from 2D images. To this end, we augment the functionality of existing deep learning networks by adding a final layer that optimizes an energy function with variables in three dimensions. We have shown our method to deliver state-of-the-art 3D human pose estimation results, and intend to explore ways of extending our method to other tasks, such as general 3D object pose estimation. Another avenue of investigation is to further expand our optimization approach to handle even more general pairwise potentials. Furthermore we intend to make our optimization module software publicly available where it can be easily adopted for other applications in the computer vision community.

References

1. Toshev, A., Szegedy, C.: Deeppose: Human pose estimation via deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014) 1653–1660
2. Pishchulin, L., Insafutdinov, E., Tang, S., Andres, B., Andriluka, M., Gehler, P.V., Schiele, B.: Deepcut: Joint subset partition and labeling for multi person pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 4929–4937
3. Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., Schiele, B.: Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In: European Conference on Computer Vision, Springer (2016) 34–50
4. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. arXiv preprint arXiv:1611.08050 (2016)
5. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: European Conference on Computer Vision, Springer (2016) 483–499
6. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 4724–4732
7. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. arXiv preprint arXiv:1703.06870 (2017)
8. Tome, D., Russell, C., Agapito, L.: Lifting from the deep: Convolutional 3d pose estimation from a single image. arXiv preprint arXiv:1701.00295 (2017)
9. Chen, C.H., Ramanan, D.: 3d human pose estimation= 2d pose estimation+ matching. arXiv preprint arXiv:1612.06524 (2016)
10. Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., Black, M.J.: Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In: European Conference on Computer Vision, Springer (2016) 561–578
11. Mehta, D., Rhodin, H., Casas, D., Sotnychenko, O., Xu, W., Theobalt, C.: Monocular 3d human pose estimation in the wild using improved cnn supervision. arXiv preprint arXiv:1611.09813v3 (2017)
12. Guler, A., Chandra, S., Kokkinos, I.: Human joint angle estimation and gesture recognition for assistive robotic vision. In: ACVR, ECCV Workshop. (2016)
13. Pavlakos, G., Zhou, X., Derpanis, K.G., Daniilidis, K.: Coarse-to-fine volumetric prediction for single-image 3d human pose. arXiv preprint arXiv:1611.07828 (2016)
14. Burenius, M., Sullivan, J., Carlsson, S.: 3d pictorial structures for multiple view articulated pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2013) 3618–3625
15. Tekin, B., Márquez-Neila, P., Salzmann, M., Fua, P.: Fusing 2d uncertainty and 3d cues for monocular body pose estimation. arXiv preprint arXiv:1611.05708 (2016)
16. Tekin, B., Katircioglu, I., Salzmann, M., Lepetit, V., Fua, P.: Structured prediction of 3d human pose with deep neural networks. CoRR **abs/1605.05180** (2016)
17. Tompson, J.J., Jain, A., LeCun, Y., Bregler, C.: Joint training of a convolutional network and a graphical model for human pose estimation. In: NIPS. (2014)
18. Yang, W., Ouyang, W., Li, H., Wang, X.: End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 3073–3082
19. Lee, C., Xie, S., Gallagher, P.W., Zhang, Z., Tu, Z.: Deeply-supervised nets. In: AISTATS. (2015)
20. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1440–1448

21. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. (2015) 91–99
22. Guler, R.A., Trigeorgis, G., Antonakos, E., Snape, P., Zafeiriou, S., Kokkinos, I.: Densereg: Fully convolutional dense shape regression in-the-wild. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (July 2017)
23. Papandreou, G., Zhu, T., Kanazawa, N., Toshev, A., Tompson, J., Bregler, C., Murphy, K.P.: Towards accurate multi-person pose estimation in the wild. CoRR **abs/1701.01779** (2017)
24. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. International Journal of Computer Vision **61**(1) (2005) 55–79
25. Chen, X., Yuille, A.L.: Articulated pose estimation by a graphical model with image dependent pairwise relations. In: Advances in Neural Information Processing Systems. (2014) 1736–1744
26. Sapp, B., Toshev, A., Taskar, B.: Cascaded models for articulated pose estimation. Computer Vision–ECCV 2010 (2010) 406–420
27. Kinauer, S., Berman, M., Kokkinos, I.: Monocular surface reconstruction using 3d deformable part models. In: Computer Vision–ECCV 2016 Workshops, Springer (2016) 296–308
28. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine Learning **3**(1) (2011) 1–122
29. Martins, A.F., Smith, N.A., Aguiar, P.M., Figueiredo, M.A.: Dual decomposition with many overlapping components. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (2011) 238–249
30. Boussaid, H., Kokkinos, I.: Fast and exact: Admm-based discriminative shape segmentation with loopy part models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014) 4058–4065
31. Komodakis, N., Paragios, N., Tziritas, G.: Mrf optimization via dual decomposition: Message-passing revisited. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE (2007) 1–8
32. Joachims, T., Finley, T., Yu, C.N.J.: Cutting-plane training of structural svms. Machine Learning **77**(1) (2009) 27–59
33. Pepik, B., Stark, M., Gehler, P.V., Schiele, B.: Multi-view and 3d deformable part models. IEEE Trans. Pattern Anal. Mach. Intell. **37**(11) (2015) 2232–2245
34. Zhang, Y., Sohn, K., Villegas, R., Pan, G., Lee, H.: Improving object detection with deep convolutional networks via bayesian optimization and structured prediction. (2015) 249–258
35. Sun, X., Shang, J., Liang, S., Wei, Y.: Compositional human pose regression. arXiv preprint arXiv:1704.00159 (2017)
36. Yasin, H., Iqbal, U., Kruger, B., Weber, A., Gall, J.: A dual-source approach for 3d pose estimation from a single image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 4948–4956
37. Rogez, G., Schmid, C.: Mocap-guided data augmentation for 3d pose estimation in the wild. In: Advances in Neural Information Processing Systems. (2016) 3108–3116