Modeling vs. Learning Approaches for Monocular 3D Human Pose Estimation

Wenjuan Gong[†], Jürgen Brauer[‡], Michael Arens[‡], Jordi Gonzàlez[†]
[†]Computer Vision Center, Universitat Autònoma de Barcelona, Spain
[‡]Fraunhofer IOSB, Ettlingen, Germany
†{wenjuan,poal}@cvc.uab.es, ‡{juergen.brauer,michael.arens}@iosb.fraunhofer.de

Abstract

We tackle the problem of 3D human pose estimation based on monocular images from which 2D pose estimates are available. A large number of approaches have been proposed for this task. Some of them avoid to model the mapping from 2D poses to 3D poses explicitly but learn the mapping using training samples. In contrast, there also exist methods that try to use some knowledge about the connection between 2D and 3D poses to model the mapping from 2D to 3D explicitly.

Surprisingly, up to now there is no experimental comparison of these two classes of approaches that uses exactly the same data sources and thereby carves out the advantages and disadvantages of both methods. In this paper we present such a comparison for the most commonly used learning approach for 3D pose estimation – the Gaussian process regressor – with the most used modeling approach – the geometric reconstruction of 3D poses. The results show that the learning based approach outperforms the modeling approach when there are no big changes in viewpoint or action types compared to the training data. In contrast, modeling approaches show advantages over learning approaches when there are big differences between training and application data.

1. Introduction

Human pose estimation allows for a wide field of applications such as video search, visual surveillance and human computer interfaces used *e.g.* in video games. Full body 3D human pose estimation from monocular images is a difficult problem since the depth information is lost when projecting from 3D space to 2D image plane. For this, a huge set of approaches have been suggested to recover the 3D pose based on monocular images.

One class of approaches tries to map image features directly to 3D poses. For example, Agarwal and Triggs [1] use a grid of local gradient orientation histograms, *i.e.* a dense sampling of interest points, and learn a mapping to 3D poses using direct regression. Another class of approaches first tries to map image features to 2D poses and then maps 2D pose estimates to 3D poses. For example, Andriluka *et al.* [3] first identify consistent sequences of 2D poses (called 'tracklets') and formulate the 3D pose estimation problem within a Bayesian framework while the prior probability of 3D poses is modeled using a hierarchical Gaussian process latent variable model.

For the later class there exist two subclasses that differ in the way in which 2D poses are lifted to 3D poses. Learning approaches try to learn this mapping using training examples and adapt some mapping using *e.g.* support vector machine, relevance vector machine [2], or Gaussian process Regressors. Modeling approaches try to model this mapping from 2D to 3D poses explicitly by using knowledge about the inverse of the 3D to 2D mapping. Although the learning and modeling approaches are quite different by concept for the 2D to 3D lifting task it has not yet been investigated systematically how the two classes of approaches differ and what are the advantages of each class.

The main contribution of this paper is to close this gap. For this we present a systematic evaluation by choosing a typical representative method of each class and compare their 3D pose reconstruction performance directly using the same 2D input data. For the class of modeling approaches we choose a geometric reconstruction approach – originally presented for a restricted parallel projection camera model [13], used in several following works (*e.g.* [5], [7]), and recently extended to a realistic perspective projection camera model [4]. Refer to section 2 for detailed explanation of the geometric method. For the class of learning approaches we choose the Gaussian process regression since it is successfully used in many pose estimation works (*e.g.* [15]). Support vector machine and relevance vector machine [2] are more efficient in training as they are picking the most representative training samples for the model. But due to a better predicting accuracy, we choose Gaussian process regressor. We evaluate both methods on the TUM kitchen and the HumanEva dataset.

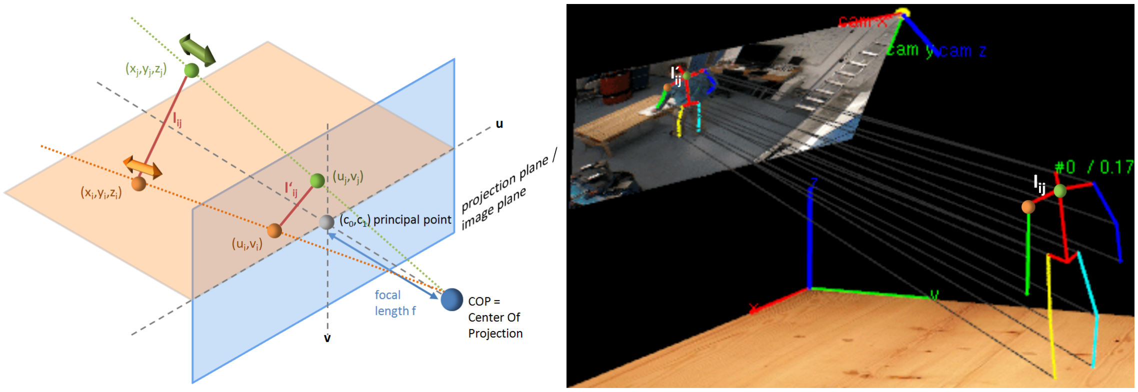

Figure 1. Geometric reconstruction of 3D poses. Using the foreshortening information of projected limb lengths l'_{ij} we can reconstruct the displacement in $\Delta_z = z_1 - z_2$ in z direction even for perspective camera models.

2. Geometric reconstruction of 3D Poses

A typical example of a modeling approach is the work presented by Taylor [13]. Assuming that the 3D to 2D image formation process can be modeled by a scaled orthographic projection, a 3D object point (x, y, z) is mapped to its corresponding 2D image point (u, v) by $u = s \cdot x, v = s \cdot y$. This corresponds to a parallel projection with a subsequent scaling with scaling factor s. If we know the 3D length of a limb l between two body marker points (x_1, y_1, z_1) and (x_2, y_2, z_2) and their corresponding projected points (u_1, v_1) and (u_2, v_2) we can reconstruct the displacement $\Delta_z := z_1 - z_2$ of the limb in z direction based on the measured length of the foreshortened limb in the 2D image. Since:

$$u_1 - u_2 = s \cdot (x_1 - x_2), \quad v_1 - v_2 = s \cdot (y_1 - y_2) \tag{1}$$

we can reformulate the Euclidian equation to get:

$$\begin{aligned} l^2 &= (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 && (2) \\ \Leftrightarrow (z1 - z2) &= \pm\sqrt{l^2 - (x_1 - x_2)^2 - (y_1 - y_2)^2} && (3) \\ \Leftrightarrow \Delta_z &= \pm\sqrt{l^2 - \frac{(u_1 - u_2)^2 + (v_1 - v_2)^2}{s^2}} && (4) \end{aligned}$$

Note that the displacement Δ_z can be reconstructed for one limb only up to a sign ($+$ or $-$) ambiguity in equation 4 since we cannot decide which of the limb endpoints (x_1, y_1, z_1), (x_2, y_2, z_2) is nearer to the camera. Having two reconstruction possibilities for one limb, for a body model with N limbs we have 2^N reconstruction possibilities for the whole body pose.

To solve this ambiguity, Taylor's original work assumed that a user labels which endpoint of each limb is nearer to the camera. Thus the original method was only a semi-automatic 3D pose reconstruction approach. To choose one of this 2^N solutions automatically different approaches have been suggested. Jiang [5] compares each of the candidate poses with over 4 million pose examples from the CMU motion capture database to assess the probability of each candidate pose. Mori and Malik [7] compare an unknown image with a sample database of images using shape context descriptor matching. The sample images are labeled with 2D body part locations and the information for each limb which of its endpoints is closer to the camera. Based on the shape context descriptor matches the 2D body part location and the information which limb endpoint is nearer to the camera is transferred to the unknown image. Wei and Chai [17] also tackle the problem of how to determine the unknown scale factor s. The set of limb projection constraints for all limbs of a body model in equation 4 are augmented by further constraints based on limb symmetries and fixed lengths on some rigid subparts of the human body such as the pelvis. Nevertheless, sometimes these additional constraints are not sufficient to solve the ambiguity. In such cases the pose reconstruction stops and the user has to solve the ambiguity manually.

Beside this ambiguity, Taylor's original method is more applicable in conditions that cameras are placed far from the captured objects. The model assumes that the projected size of a person or a limb does not depend on its distance to the camera which is not true when cameras are near. Note that the z coordinate has no influence on the resulting (u, v) coordinate. Parameswaran and Chellappa [9] therefore try to deal with a new camera model, *i.e.* perspective projections. Possible head orientations are reconstructed using a set of polynomial equations, epipolar geometry is recovered, and the rest of the body joint coordinates are computed using knowledge about the limb lengths in a recursive manner. But in their approach the authors have to make two strong assumptions: the torso twist has to be small – which is not true for many poses – and the locations of four markers on the head have to be given (*e.g.* forehead, chin, nose and left or right ear) – which is hard to be provided automatically since it would mean a very precise automatic localization of these markers on the head.

An approach that does not need to make such assumptions and nevertheless adopts a perspective projection camera model was presented recently [4]. In the perspective camera model a 3D point (x, y, z) is mapped to the 2D point (u, v) with $u = f\frac{x}{z} + c_0$, $v = f\frac{y}{z} + c_1$. f is called focal length, (c_0, c_1) is called principal point. For a known calibrated camera, we know the principal point (and the focal length), and can correct the 2D image coordinates for the principal point translation vector $(u' = u - c_0, v' = v - c_1)$ and therefore assume $(c_0, c_1) = (0, 0)$. Since

$$x_i = \frac{z_i u_i}{f}, \quad y_i = \frac{z_i v_i}{f} \tag{5}$$

we can reformulate the Euclidian equation into a quadratic equation for the z_i coordinate as following (refer to [4] for deduction details):

$$\begin{aligned} l_{ij}^2 &= (\frac{z_i u_i}{f} - \frac{z_j u_j}{f})^2 + (\frac{z_i v_i}{f} - \frac{z_j v_j}{f})^2 + (z_i - z_j)^2 && (6)\\ &\dots \\ \Leftrightarrow z_{i_{1/2}} &= -\frac{C z_j}{2A} \pm \sqrt{(\frac{C z_j}{2A})^2 - (\frac{B}{A} z_j^2 - \frac{f^2 l_{ij}^2}{A})} && (7) \end{aligned}$$

Equation 7 shows how to reconstruct the z coordinate of a child marker z_i given already reconstructed z_j coordinate of a parent marker. We further need to provide limb lengths l_{ij} (connecting marker i with j), and the 2D coordinates (u, v) of the markers within the image which are supposed to be provided by a 2D pose estimator.

Assuming a perspective projection camera model, we first have to start with an estimate for the z coordinate of the root marker of the kinematic tree, then we can apply equation 7 in a recursive manner: having computed the z coordinate for a parent marker, we can compute the two possible solutions for the z coordinate of the child marker and step down further in the kinematic tree. Since there are still two solutions for the z coordinate (either + and − in equation 7) we end up with a binary reconstruction tree with 2^N mathematically possible poses. To reduce the number of pose candidates already during the binary reconstruction tree traversal it was shown in [4] that it is possible to check for abnormal joint angles based on anatomical joint limits in the knees and elbows and prune branches of the reconstruction tree whenever we encounter anatomical violations.

To select a final 3D pose estimate from the remaining set of pose candidates, we can assign a probability

$$P(\vec{p}) = \prod P(\vec{j_i}) \tag{8}$$

for each pose candidate $\vec{p}$, where $P(\vec{j_i} = (\alpha, \beta, \gamma))$ is the probability to find a joint in a certain configuration $\vec{j_i} = (\alpha, \beta, \gamma)$ (the three Euler angles) which can be learned by observing motion capture sample data. The z coordinate of the root marker can be estimated by the distance of the person to the image plane. In [4] the proposed solution for the estimation of the person to camera distance was reconstructing all possible poses using different distance estimates and then choose the distance where the average pose probability takes on a maximum. This approach is successful for estimating the person ↔ camera distance since for distances different from the ground truth distance, the reconstructed poses have to be squeezed (distance too small) or pulled apart (distance estimate too big) into the perspectives rays bundle which in turn results in unlikely joint angles and small pose probabilities. For further details we refer the reader to [4].

3. Regression of 3D poses

The Gaussian process regressor is currently the most widespread representative for learning approaches in the pose estimation community since it has been proved to be an effective approach for the nonlinear 2D to 3D pose mapping problem [16]. The main idea of Gaussian process regression is to map unknown test data to a prediction by interpolating the training data weighted by the correlation between the training and test data. In our method, we take normalized 2D body part positions as input and output a 3D pose prediction – represented as a vector of direction cosines of limb orientations. In the following subsections, we will explain detailed representations and settings for the Gaussian process regressor used here.

3.1. Normalized 2D body part positions

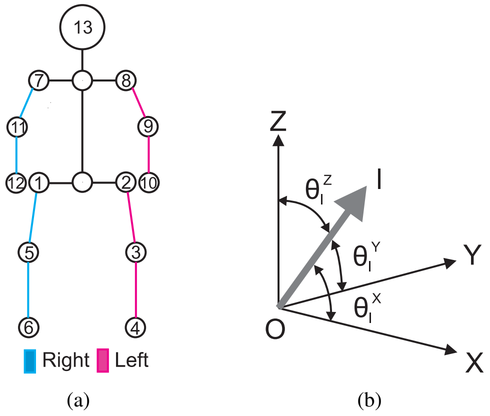

Figure 2. (a) The 3D stick figure model used for representing human pose with 2D body part indices. Thirteen body parts corresponding to the markers used in motion capture are used [12]. (b) The angles $(\theta_l^x, \theta_l^y, \theta_l^z)$ between the limb l and the axes [11].

From detected body part positions of a performer, we take 13 body parts. Correspondence of body parts and 3D stick figure model is shown in figure 2(a). The 2D body part positions are collected within a vector

$BP = [x_1, y_1, x_2, y_2, \ldots, x_i, y_i, \ldots, x_{12}, y_{12}, x_{13}, y_{13}]$ where (x_i, y_i) is the 2D position of the i-th body part. For representing the 2D pose independently of the persons's size and distance to the camera, we normalize this 2D pose vector:

$$BP_{norm} = (BP + M_{off}) * M_{scale} \tag{9}$$

where $*$ means element-wise multiplication and

$$\begin{aligned} M_{scale} &= [\frac{1}{y_{range}}, \frac{1}{y_{range}}, \ldots, \frac{1}{y_{range}}, \frac{1}{y_{range}}] && (10) \\ M_{off} &= [x_{off}, y_{off}, x_{off}, y_{off}, \ldots, x_{off}, y_{off}] && (11) \\ x_{off} &= -min(X) + (y_{range} - x_{range})/2 && (12) \\ y_{off} &= -min(Y) && (13) \end{aligned}$$

where X and Y are vectors of all x and y coordinate values of the 2D pose in the frame.

For upright standing persons, the range of y coordinate values is typically bigger compared to the range of x coordinate values. For this, we normalize both x and y coordinates by y range in each frame (M_{scale}). This makes sure, that we keep the aspect ratio of the performer and that normalized y coordinates range from 0 to 1. The normalized 2D part positions are the input of the regressor.

3.2. 3D human pose representation

The output of our regressor is 3D human poses. According to our experiments, with the dimension of the output from regressor increases, the regressor will take more time for training parameters. We use the same representations for human posture as in [11], considering this representation is concise and unambiguous. We model a human pose using twelve rigid body parts: hip, torso, shoulder, neck, two thighs, two lower legs, two upper arms and two forearms. These parts are connected by a total of ten inner joints, as shown in figure 2(a). For defining a local coordinate system in the hip, we use the direction of the torso for the y axis and the direction vector pointing from the left hip to the right hip as z axis. The x axis is then given by cross product of y axis and z axis.

The pose of an actor in an image frame is represented as a vector of direction cosines, i.e. the cosines of the angles between the limb direction vectors and the three coordinate axes of the root coordinate system. That is, limb orientation is modeled using three parameters, without modeling self rotation of limbs around its axes, as shown in figure 2(b). The overall posture of the subject for a frame is represented using a vector of direction cosines measured on twelve limbs. This results in a 36-dimensional representation of the pose:

$$\psi = [\cos\theta_1^x, \cos\theta_1^y, \cos\theta_1^z, \ldots, \cos\theta_{12}^x, \cos\theta_{12}^y, \cos\theta_{12}^z], \tag{14}$$

where θ_l^x, θ_l^y and θ_l^z are the angles between the limb l and the axes of the root coordinate system in the hip as shown in figure 2(b).

3.3. Gaussian process regression

Gaussian processes yield a method for specifying a probability distribution over functions by specifying a mean and a covariance function for the function values $f(x)$. By training a Gaussian process with sample data $\{x, f(x)\}$ the variance of the Gaussian process becomes small for function values $f(x)$ at supporting points x included in the training data, which corresponds to an increased certainty about the function values at these points, while at other points x' the variance of the Gaussian process remains high which corresponds to a high uncertainty about the function values $f(x')$ at such points.

More formally, a Gaussian process is completely specified by a mean function and covariance function. If we denote the mean function as $m(\mathbf{x})$ and the covariance function as $Cov[\mathbf{f}(\mathbf{x_1}), \mathbf{f}(\mathbf{x_2})] = k(\mathbf{x_1}, \mathbf{x_2})$, a Gaussian process is denoted as $\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x_1}, \mathbf{x_2}))$, where

$$\begin{aligned} m(\mathbf{x}) &= E[\mathbf{f}(\mathbf{x})] && (15) \\ k(\mathbf{x_1}, \mathbf{x_2}) &= E[(\mathbf{f}(\mathbf{x_1}) - m(\mathbf{x_1}))(\mathbf{f}(\mathbf{x_2}) - m(\mathbf{x_2}))] && (16) \end{aligned}$$

The covariance function specifies how two function values $f(x_1)$ and $f(x_2)$ (the function values are considered as random variables) can change, given two arguments x_1,x_2. Since we want the Gaussian process to interpolate continuously between supporting points, a continuous covariance function is used as well. A typical covariance function that is used for a Gaussian process is the squared exponential:

$$k(\mathbf{x_1}, \mathbf{x_2}) = \theta_1^2 \exp(-\frac{(\mathbf{x_1} - \mathbf{x_2})^2}{2\theta_2^2}) \tag{17}$$

where θ_1, θ_2 are called the amplitude and lengthscale hyperparameters respectively. This covariance function makes sure that the covariance of two function values $f(x_1)$ and $f(x_2)$ of nearby x_1, x_2 is high (which will result in a smooth function), while the covariance of $f(x_1)$ and $f(x_2)$ is low, if x_1, x_2 are far away.

Given a 2D pose estimate which is represented as the 26 dimensional vector BP_{norm} we train one Gaussian process to predict each of the 36 dimensions of the 3D pose vector ψ separately. For the Gaussian process training and prediction we used a reference implementation[1].

4. Experiments

In this section, we describe the settings for the experiments and how we measure the error of an estimated pose both for the regression and the geometric reconstruction

[1] http://www.gaussianprocess.org/gpml/code/matlab/doc/

method. Based on a comparison of these errors, we analyze and conclude advantages and disadvantages of each method.

4.1. Training and Test Data Composition

We chose the public available HumanEva [12] and the TUM kitchen dataset [14] for an exhaustive evaluation and comparison of both methods since both datasets provide 3D motion capture ground truth data which allows to compute an error for each estimated pose. Furthermore, intrinsic and extrinsic camera parameters are provided as well which allows to project the 3D poses into the image and thereby provide 2D ground truth poses as well. Both datasets contain sequences where different subjects (4 for HumanEva, 4 for TUM kitchen) perform different actions (walking, boxing, laying a kitchen table, etc.) recorded from different viewpoints (7 for HumanEva, 4 for TUM kitchen). These variations of performers, action types and viewpoints between training and test data, allow to define a set of experiments in which different capabilities of both methods can be tested. We use 4 categories of experiments:

1. *train* on a sequence recorded from one camera view $\rightarrow$ *test* on a sequence recorded from another view (1a/1b/1c/1d). The change of viewpoint can be weak (1a/1b) or strong (1c/1d).

2. *train* on a sequence comprising a subject $S_i \rightarrow$ *test* on a sequence comprising another subject S_j [2] (2a/2b),

3. *train* on a sequence showing one action class $A_1 \rightarrow$ *test* on a sequence showing another action class A_2 (3a/3b),

4. *train* on a sequence from HumanEva (TUM kitchen) dataset $\rightarrow$ *test* on a sequence from the other dataset, i.e. TUM kitchen (HumanEva) (4a/4b)

Both approaches, the regression and the geometric reconstruction method, map 2D input poses to 3D pose estimates. In experiments 1a-4b we test on ground truth 2D input poses and estimated 2D poses as well (see table 1). The estimated 2D poses stem from a Implicit Shape Model based 2D pose estimator, that learns the spatial relation between SIFT features and 15 body parts and uses this learned relationship to vote for the location of each body part. The method and the quality of these estimated 2D poses is described in detail in [8].

The quality of estimated 2D input poses depends on the performance of the 2D pose estimator. To be independent from this quality of a 2D pose estimator we also added two further experiments 5a/5b (compare table 1 and table 2) in which we put more and more noise onto ground truth 2D poses and evaluate the capability of both the regression and geometric reconstruction method to estimated 3D poses with such 2D poses of different noise levels. Here, a noise level of n% means that we added a random translation vector (Δ_x, Δ_y) to each marker position where the length of this random vector is in the range of 0-n% of the person's height (measured in pixels) in the current frame.

In the learning phase the regression method uses the (2D ground truth pose, 3D ground truth pose) pairs of the training data to train the Gaussian processes and fix the hyper-parameters. In contrast, the geometric reconstruction method uses only the 3D ground truth poses of the training data to learn joint angle probabilities for all joints.

4.2. Error Measurements

Since we use the same input 2D poses (ground truth / estimated / noisy) for both experiments this allows us to compare both approaches - the regression and the geometric reconstruction - based on their 3D pose estimation performance which is measured by the average angular error and average absolute marker position error of the estimated 3D poses compared to the ground truth 3D poses. Suppose predicted limb angles $\hat{\Theta}$ and ground truth limb angles Θ are denoted as

$$\hat{\Theta} = [\hat{\theta}^x_{l_1}, \hat{\theta}^y_{l_1}, \hat{\theta}^z_{l_1}, \dots, \hat{\theta}^x_{l_{14}}, \hat{\theta}^y_{l_{14}}, \hat{\theta}^z_{l_{14}}] \quad (18)$$

$$\Theta = [\theta^x_{l_1}, \theta^y_{l_1}, \theta^z_{l_1}, \dots, \theta^x_{l_{14}}, \theta^y_{l_{14}}, \theta^z_{l_{14}}] \quad (19)$$

then the angular error is defined as:

$$Err_{Ang} = \frac{\sum_{i=1}^{J} |\Theta_i - \hat{\Theta}_i| \bmod 180^\circ}{J} \quad (20)$$

where $J = 3 \cdot 14$ (3 Euler angles, 14 limbs) and "mod" is to deal with angle singularity problem.

An angular error in a joint at a high level of the kinematic tree (*e.g.* shoulder joint) will have a bigger impact on the resulting pose than an angular error in a joint at a low level of the kinematic tree (*e.g.* wrist joint). For this, we do not only compute the angular error Err_{Ang} but also compare the absolute marker positions of the estimated poses with the ground truth pose.

Since the geometric reconstruction method first reconstructs 3D marker locations and then computes joint angles based on the reconstructed marker positions this comparison is straightforward. In contrast, the regression method maps a normalized 2D body part location vector BP_{norm} to a 3D joint angle vector ψ such that there are at first no estimated 3D marker locations at all. To compute 3D marker location estimates, we assume a person of average U.S. size [6] and use pre-computed relative limb length ratios to compute absolute limb length estimates. Based on the estimated joint angles and these estimated limb lengths

[2] Person S_i within the TUM kitchen dataset is different from the person S_i within the HumanEva dataset

Exp.	training	testing	change of	**Geometric reconstruction**				**Regression**			
2D input				Ground truth		Estimation		Ground truth		Estimation	
				[°]	[mm]	[°]	[mm]	[°]	[mm]	[°]	[mm]
1a	TUM, 0-0-cam3, S1	TUM-0-0-cam2, S1	viewpoint (weak)	6.12	143.51	13.44	230.72	**0.12**	**1.82**	**7.358**	**146.83**
1b	HE, walk-cam1, S1	HE, walk-cam2, S1	viewpoint (weak)	7.47	155.77	10.74	187.57	**1.39**	**22.27**	**3.79**	**78.80**
1c	TUM, 0-0-cam1, S1	TUM-0-2-cam3, S1	viewpoint (strong)	5.24	135.14	10.93	194.49	**5.48**	**96.69**	**5.85**	**102.71**
1d	HE, box-cam1, S1	HE, box-cam2, S1	viewpoint (strong)	8.15	159.14	11.59	189.33	**4.49**	**85.02**	**5.05**	**95.56**
2a	TUM, 0-0-cam3, S1	TUM-0-3-cam3, S2	person	6.53	156.43	12.20	197.92	**5.52**	**89.09**	**7.92**	**140.93**
2b	HE, walk-cam1, S1	HE, walk-cam1, S2	person	8.61	158.41	11.71	194.97	**3.87**	**64.29**	**5.12**	**95.18**
3a	HE, walk-cam1, S2	HE, box-cam1, S2	action	16.65	210.78	17.18	202.41	**11.48**	**197.23**	**11.53**	**192.57**
3b	HE, box-cam1, S2	HE, walk-cam1, S2	action	**9.10**	**153.64**	12.02	197.35	9.34	166.02	**9.13**	**160.09**
4a	HE, walk-cam2, S1	TUM, 0-2-cam3, S2	dataset	**7.57**	155.15	13.48	214.47	8.40	**137.11**	**8.47**	**139.70**
4b	TUM, 0-2-cam3, S2	HE, walk-cam2, S1	dataset	8.07	160.06	10.98	188.34	**7.08**	**123.78**	**7.52**	**131.10**

Table 1. Experiments definition and 3D pose reconstructions errors for both the geometric reconstruction and regression approach. For each experiment we present the average angular and average marker position error of the estimated poses resulting from the geometric reconstruction and the regression approach compared to the ground truth poses.

Exp.	training	testing	**Geometric reconstruction**									
			2%		4%		6%		8%		10%	
			[°]	[mm]	[°]	[mm]	[°]	[mm]	[°]	[mm]	[°]	[mm]
5a	TUM, 0-0-cam3, S1	TUM-0-0-cam2	6.69	149.19	7.83	160.43	9.14	175.17	10.51	191.87	11.75	205.82
5b	HE, walk-cam1, S1	HE, walk-cam2, S1	8.00	159.03	8.97	168.07	10.04	177.59	11.42	189.02	12.53	198.49

Table 2. 3D pose reconstruction errors for geometric reconstruction method with noisy 2D input poses and different noise levels.

Exp.	training	testing	**Regression**									
			2%		4%		6%		8%		10%	
			[°]	[mm]	[°]	[mm]	[°]	[mm]	[°]	[mm]	[°]	[mm]
5a	TUM, 0-0-cam3, S1	TUM-0-0-cam2	5.72	112.01	5.77	113.11	5.85	115.01	5.94	117.08	6.03	119.21
5b	HE, walk-cam1, S1	HE, walk-cam2, S1	1.55	24.70	1.90	31.34	2.27	39.53	2.59	47.53	2.85	54.73

Table 3. 3D pose reconstruction errors for regression method with noisy 2D input poses and different noise levels.

we can then reconstruct 3D marker locations as well. If we denote these estimated marker positions $\hat{\mathbf{P}}$ and ground marker positions $\mathbf{P}$:

$$\hat{\mathbf{P}} = [\hat{x}_1, \hat{y}_1, \hat{z}_1, \ldots, \hat{x}_{15}, \hat{y}_{15}, \hat{z}_{15}] \quad (21)$$

$$\mathbf{P} = [x_1, y_1, z_1, \ldots, x_{15}, y_{15}, z_{15}], \quad (22)$$

then the average marker position error is defined as

$$Err_{pos} = \frac{\sum_{i=1}^{M} |\mathbf{P}_i - \hat{\mathbf{P}}_i|}{M}. \quad (23)$$

where $M = 3 \cdot 15$ (x/y/z coordinates, 15 markers). Err_{Ang} is specified in degrees, Err_{pos} in mm.

4.3. Results

The 3D pose estimation error results of all experiments are shown in table 1, table 2, and table 3.

For the case of estimated 2D input poses (see table 1) and noisy 2D input poses (see table 2 and table 3) the results are obvious: the regression method outperforms the geometric reconstruction method in all scenarios. This shows that the Gaussian process learning based approach is able to use the training data samples sufficiently to interpolate to new data. The estimated 3D poses generated from the regression method are substantially better than for the 3D pose estimates obtained from the geometric reconstruction method. This shows that the geometric reconstruction approach presented in its puristic form here is not able to deal with noisy 2D input poses. As the 2D input poses get more and more noisy, errors from regression method increase slower compared to errors of the geometric reconstruction method (compare table 2 with table 3). The reason is that the wrong 2D body part locations will lead to wrong 2D limb lengths which in turn will lead to wrong displacement values (see equation 4 and equation 7). The working principle – using the foreshortening information of limbs to reconstruct the limb displacement in z direction – continuously loses its basis with increasing noise in the 2D input poses (see table 2). This underlines the need to augment modeling based approaches for 2D to 3D pose estimation with some explicit handling of noise while it is not necessary for the regression / learning based approaches.

For the case of using 2D ground truth input poses, the average 3D pose error is for the regression method 5.7° (averaged over all experiments 1a-4b) compared to 8.3° for the geometric reconstruction approach. This shows that the regression method yields 3D pose estimates with an er-

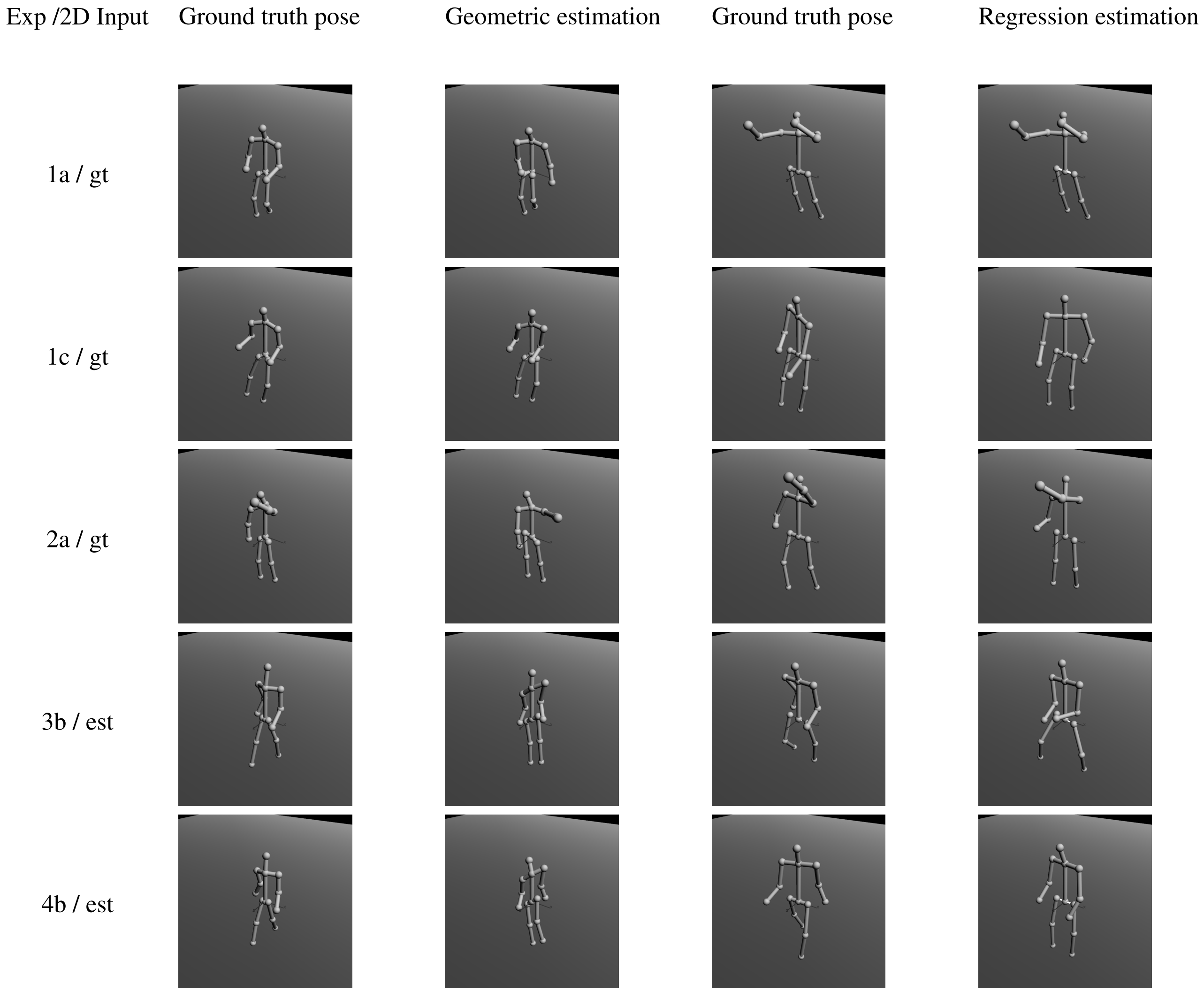

Figure 3. Qualitative 3D pose estimation samples. Column "Exp /2D Input" shows the experiment number and 2D pose input type. There are two types of 2D input poses. "gt" means ground truth 2D pose and "est" means estimated 2D pose. 1st+2nd column: ground truth 3D pose and corresponding estimated 3D pose by the geometric reconstruction approach. 3rd+4th column: ground truth 3D pose and corresponding estimated 3D pose by the Gaussian process approach.

ror of about 2.6° lower than the modeling approach used here. Especially due to the fact that the Gaussian process regressor yields better 3D pose estimates in average, it is interesting that nevertheless in some cases, the geometric reconstruction method could outperform the regression method slightly (1c/3b/4a). We trace this back to the fact, that learning based approaches run into problems if the test data is substantially different from the training data. This is the case in experiment 1c where we find a big viewpoint change, in 3b where we find a change of action, and in experiment 4a where we switched from one dataset to another. We expect an even bigger difference if we compute the joint angle probabilities on a bigger variety of motion capture data compared to the situation here in which we use just one dataset to estimate the joint angle probabilities. In scenarios, in which there are new actions, viewpoints or other changes compared to the training data we expect the model based approaches to be the better choice since then interpolation capabilities of learning based approaches will not be sufficient to generalize to the new data.

When there is a big variance regarding the action type, the regression method has problems predicting the 3D poses correctly, because no similar poses are learned in training. If variances are only present for certain limbs, for example upper body limbs or lower body limbs, the regression method can correctly predict the body parts that have similar orientation as in training data, *e.g*. experiment 4b with estimated 2D body part input in figure 3. This is due to the fact that every limb orientation in the regression method is estimated separately from others, in contrast to the modeling approach. Thus, for the regression method errors from the root of the kinematic tree structure will not transmit to leave nodes. Another problem occurs when there are ambiguities in mapping, *e.g*. in experiment 1c with ground truth 2D body part input in figure 3, the predict pose is left-right flipped compared with ground truth 3D pose. This is due to lack of depth information in 2D images, the same 2D pose might correspond to more than one 3D poses.

5. Conclusion and Future Work

In this paper, we compared the most commonly used learning approach – the Gaussian process regressor – with the most used modeling approach – the geometric reconstruction of 3D poses based on Taylor's method. We test on two public datasets: TUM kitchen and HumanEva data. We experimented on ground truth 2D body part data, noisy 2D ground truth body part data and real detected 2D body parts. Our conclusion is that regression methods perform better in conditions with minor viewpoint changes and minor action variances between training and test data. In contrast, the geometric method is more suitable in cases where there are major viewpoint and action type changes.

For future work, one obvious extension of the geometric reconstruction approach will deal with uncertainty and noise from the 2D pose estimator. Also, we would like to compare with other solutions. Peng *et al*. [10] propose an interesting geometric solution. While they need one frame with specific limb configuration, it is nevertheless mathematically formulated. Also Zhao *et al*. [18] solve posture reconstruction problem using energy minimization. By incorporating these methods, we can get a border view of pose reconstruction problem.

Acknowledgements

This work is supported by the Spanish projects TIN2009-14501-C02-02, Consolider MIPRCV (CSD200700018), and Avanza I+D ViCoMo (TSI-020400-2009-133).

References

[1] A. Agarwal and B. Triggs. A local basis representation for estimating human pose from cluttered images. In *Proc. of Asian Conf. on Computer Vision*, pages 50–59, 2006. 1

[2] A. Agarwal and B. Triggs. Recovering 3d human pose from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:44–58, 2006. 1

[3] M. Andriluka, S. Roth, and B. Schiele. Monocular 3d pose estimation and tracking by detection. In *Proc. of CVPR 2010*, USA, 2010. 1

[4] J. Brauer and M. Arens. Reconstructing the missing dimension: From 2d to 3d human pose estimation. In *Proc. of REACTS workshop, in conj. with Int. Conf. of Computer Analysis of Images and Patterns (CAIP2011)*, Spain, Málaga, 2011. to appear in. 1, 3

[5] H. Jiang. 3d human pose reconstruction using millions of exemplars. In *Proc. of 20th ICPR*, pages 1674–1677, 2010. 1, 2

[6] M. A. McDowell, C. D. Fryar, C. L. Ogden, and K. M. Flegal. Anthropometric reference data for children and adults: United states, 2003-2006. *National Health Statistics Reports*, (10):1–45, 2008. 5

[7] G. Mori and J. Malik. Recovering 3d human body configurations using shape contexts. *PAMI*, 28(7):1052–1062, 2006. 1, 2

[8] J. Müller and M. Arens. Human pose estimation with implicit shape models. In *Proc. of the first ACM international workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Streams*, ARTEMIS '10, pages 9–14, New York, NY, USA, 2010. ACM. 5

[9] V. Parameswaran and R. Chellappa. View independent human body pose estimation from a single perspective image. In *In Proc. of CVPR 2004*, volume 2, pages II–16 – II–22 Vol.2, june-2 july 2004. 2

[10] X. Peng, B. Zou, S. Chen, and P. Luo. Reconstruction of 3d human motion pose from uncalibrated monocular video sequences. *International Journal Of Information And Systems Sciences*, 5(3–4):503–515, 2009. 8

[11] I. Rius, J. Gonzàlez, J. Varona, and F. X. Roca. Action-specific motion prior for efficient bayesian 3d human body tracking. *Pattern Recognition*, 42(11):2907–2921, 2009. 3, 4

[12] L. Sigal, A. O. Balan, and M. J. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *IJCV*, 87:1–24, 2010. 3, 5

[13] C. J. Taylor. Reconstruction of articulated objects from point correspondences in a single uncalibrated image. *Computer Vision and Image Understanding*, 80:349–363, 2000. 1, 2

[14] M. Tenorth, J. Bandouch, and M. Beetz. The TUM Kitchen Data Set of Everyday Manipulation Activities for Motion Tracking and Action Recognition. In *THEMIS workshop. In conj. with ICCV 2009*. 5

[15] R. Urtasun, D. J. Fleet, A. Hertzmann, and P. Fua. Priors for people tracking from small training sets. *IEEE Int. Conf. on Computer Vision*, 1:403–410, 2005. 1

[16] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *PAMI*, 30(2):283–298, 2008. 3

[17] X. K. Wei and J. Chai. Modeling 3d human poses from uncalibrated monocular images. In *IEEE 12th Intern. Conf. on Computer Vision*, pages 1873–1880, October 2009. 2

[18] J. Zhao, L. Li, and K. C. Keong. 3D posture reconstruction and human animation from 2D feature points. *Computer Graphics Forum*, 24(4):759–771, 2005. 8