

Creatures great and SMAL: Recovering the shape and motion of animals from video*

Benjamin Biggs¹, Thomas Roddick¹, Andrew Fitzgibbon², and Roberto Cipolla¹

¹ University of Cambridge, Department of Engineering, Trumpington Street,
Cambridge, CB2 1PZ, UK {bjb56, tr346, rc10001}@cam.ac.uk

² Microsoft Research, 21 Station Road, Cambridge, CB1 2FB, UK awf@microsoft.com

Abstract. We present a system to recover the 3D shape and motion of a wide variety of quadrupeds from video. The system comprises a machine learning front-end which predicts candidate 2D joint positions, a discrete optimization which finds kinematically plausible joint correspondences, and an energy minimization stage which fits a detailed 3D model to the image. In order to overcome the limited availability of motion capture training data from animals, and the difficulty of generating realistic synthetic training images, the system is designed to work on silhouette data. The joint candidate predictor is trained on synthetically generated silhouette images, and at test time, deep learning methods or standard video segmentation tools are used to extract silhouettes from real data. The system is tested on animal videos from several species, and shows accurate reconstructions of 3D shape and pose.

1 Introduction

Animal welfare is an important concern for business and society, with an estimated 70 billion animals currently living under human care [1]. Monitoring and assessment of animal health can be assisted by obtaining accurate measurements of an individual’s shape, volume and movement. These measurements should be taken without interfering with the animal’s normal activity, and are needed around the clock, under a variety of lighting and weather conditions, perhaps at long range (e.g. in farm fields or wildlife parks). Therefore a very wide range of cameras and imaging modalities must be handled. For small animals in captivity, a depth camera might be possible, but techniques which can operate solely from intensity data will have a much wider range of applicability.

We address this problem using techniques from the recent human body and hand tracking literature, combining machine learning and 3D model fitting. A discriminative front-end uses a deep hourglass network to identify candidate 2D joint positions. These joint positions are then linked into coherent skeletons by solving an optimal joint assignment problem, and the resulting skeletons create an initial estimate for a generative model-fitting back-end to yield detailed shape and pose for each frame of the video.

Although superficially similar to human tracking, animal tracking (AT) has some interesting differences that make it worthy of study:

* The authors would like to thank GlaxoSmithKline for sponsoring this work.

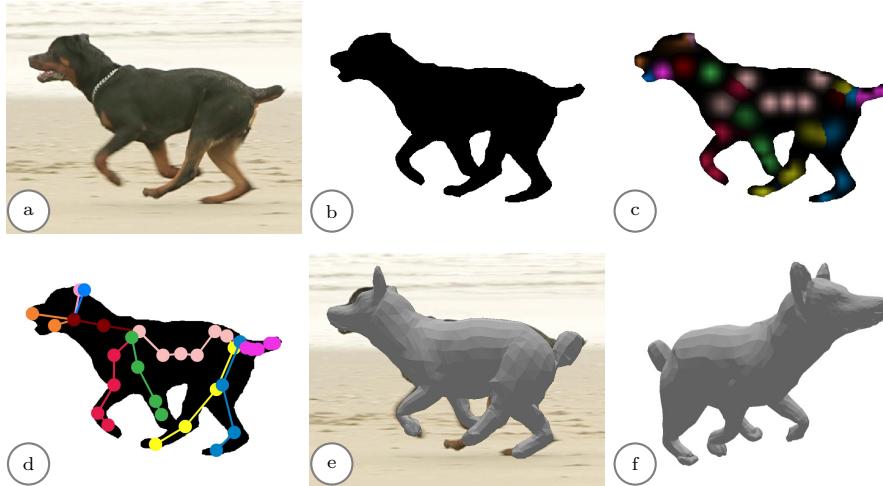


Fig. 1: System overview: input video (a) is automatically processed using DeepLabv3+ [2] to produce silhouettes (b), from which 2D joint predictions are regressed in the form of heatmaps (c). Optimal joint assignment (OJA) finds kinematically coherent 2D-to-3D correspondences (d), which initialize a 3D shape model, optimized to match the silhouette (e). Alternative view shown in (f).

Variability. In one sense, AT is simpler than human tracking as animals generally do not wear clothing. However, variations in surface texture are still considerable between individuals, and the variety of shape across and within species is considerably greater. If tracking is specialized to a particular species, then shape variation is smaller, but training data is even harder to obtain.

Training data. For human tracking, hand labelled sequences of 2D segmentations and joint positions have been collected from a wide variety of sources [3–5]. Of these two classes of labelling, animal *segmentation* data is available in datasets such as MSCOCO [4], PASCAL VOC [6] and DAVIS [7]. However this data is considerably sparser than human data, and must be “shared” across species, meaning the number of examples for a given animal shape class is considerably fewer than is available for an equivalent variation in human shape. While segmentation data can be supplied by non-specialist human labellers, it is more difficult to obtain *joint position* data. Some joints are easy to label, such as “tip of snout”, but others such as the analogue of “right elbow” require training of the operator to correctly identify across species.

Of more concern however, is 3D skeleton data. For humans, motion capture (mocap) can be used to obtain long sequences of skeleton parameters (joint positions and angles) from a wide variety of motions and activities. For animal tracking, this is considerably harder: animals behave differently on treadmills than in their quotidian environments, and although some animals such as horses

and dogs have been coaxed into motion capture studios [8], it remains impractical to consider mocap for a family of tigers at play.

These concerns are of course alleviated if we have access to synthetic training data. Here, humans and animals share an advantage in the availability of parameterized 3D models of shape and pose. The recent publication of the Skinned Multi-Animal Linear (SMAL) model [9] can generate a wide range of quadruped species, although without surface texture maps. However, as with humans, it remains difficult to generate RGB images which are sufficiently realistic to train modern machine learning models. In the case of humans, this has been overcome by generating depth maps, but this then requires a depth camera at test time [10]. The alternative, used in this work, is to generate 2D silhouette images so that machine learning will predict joint heatmaps from silhouettes only.

Taking into account the above constraints, this work applies a novel strategy to animal tracking, which assumes a machine-learning approach to extraction of animal silhouettes from video, and then fits a parameterized 3D model to silhouette sequences. We make the following contributions:

- A machine-learned mapping from silhouette data of a large class of quadrupeds to generic 2D joint positions.
- A novel optimal joint assignment (OJA) algorithm extending the bipartite matching of Cao *et al.* [11] in two ways, one which can be cast as a quadratic program (QP), and an extension optimized using a genetic algorithm (GA).
- A procedure for optimization of a 3D deformable model to fit 2D silhouette data and 2D joint positions, while encouraging temporally coherent outputs.
- We introduce a new benchmark animal dataset of joint annotations (BADJA) which contains sparse keypoint labels and silhouette segmentations for eleven animal video sequences. Previous work in 3D animal reconstruction has relied on bespoke hand-clicked keypoints [9, 12] and little quantitative evaluation of performance could be given. The sequences exhibit a range of animals, are selected to capture a variety of animal movement and include some challenging visual scenarios such as occlusion and motion blur.

The system is outlined in Fig. 1. The remainder of the paper describes related literature before a detailed description of system components. Joint accuracy results at multiple stages of the pipeline are reported on the new BADJA dataset, which contains ground truths for real animal subjects. We also conduct experiments on synthetic animal videos to produce joint accuracy statistics and full 3D mesh comparisons. A qualitative comparison is given to recent work [9] on the related single-frame 3D shape and pose recovery problem. The paper concludes with an assessment of strengths and limitations of the work.

2 Related work

3D animal tracking is relatively new to the computer vision literature, but animal breed identification is a well studied problem [13]. Video tracking benchmarks often use animal sequences [14, 15], although the tracking output is typically limited to 2D affine transformations rather than the detailed 3D mesh that we propose. Although we believe our work is the first to demonstrate dense 3D tracking of animals in video without the need for user-provided keypoints, we do build on related work across computer vision:

Morphable shape models. Cashman and Fitzgibbon [16] obtained one of the first 3D morphable animal models, but their work was limited to small classes of objects (e.g. dolphins, pigeons), and did not incorporate a skeleton. Their work also showed the use of the 2D silhouette for fitting, which is key to our method. Reinert *et al.* [17] meanwhile construct 3D meshes by fitting generalized cylinders to hand-drawn skeletons. Combined skeletal and morphable models were used by Khamis *et al.* [18] for modelling the human hand, and Loper *et al.* [19] in the SMPL model which has been extensively used for human tracking.

The SMPL model was extended to animals by Zuffi *et al.* [9], where the lack of motion capture data for animal subjects is cleverly overcome by building the model from 41 3D scans of toy figurines from five quadruped families in arbitrary poses. Their paper demonstrates single-frame fits of their model to real-world animal data, showing that despite the model being built from “artists’ impressions” it remains an accurate model of real animals. This is borne out further by our work. Their paper did however depend on per-frame human annotated keypoint labels, which would be costly and challenging to obtain for large video sequences. This work was recently extended [12] with a refinement step that optimizes over model vertex positions. This can be considered independent to the initial SMAL model fit and would be trivial to add to our method.

Shape from silhouette. Silhouette images have been shown to contain sufficient shape information to enable their use in many 3D recovery pipelines. Chen *et al.* [20] demonstrate single-view shape reconstruction from such input for general object classes, by building a shape space model from 3D samples. More related to our work, Favreau *et al.* [21] apply PCA to silhouette images to extract animal gaits from video sequences. The task of predicting silhouette images from 2D input has been effectively used as a proxy for regressing 3D model parameters for humans [22, 23] and other 3D objects [24].

Joint position prediction. There is an extensive body of prior work related to joint position prediction for human subjects. Earlier work used graphical approaches such as pictorial structure models [25, 26, 5], which have since been replaced with deep learning-based methods [11, 27]. Few works predict animal joint positions directly owing to the lack of annotated data, although Mathis *et al.* [28] demonstrate the effectiveness of human pose estimation architectures for restricted animal domains. Our method instead trains on silhouette input, allowing the use of synthetic training imagery. The related task of animal part segmentation [29, 30] has seen some progress due to general object part datasets [31, 32].

2.1 Preliminaries

We are given a deformable 3D model such as SMAL [9] which parametrizes a 3D mesh as a function of *pose* parameters $\theta \in \mathbb{R}^P$ (e.g. joint angles) and *shape* parameters $\beta \in \mathbb{R}^B$. In detail, a 3D mesh is an array of vertices $\nu \in \mathbb{R}^{3 \times V}$ (the vertices are columns of a $3 \times V$ matrix) and a set of triangles represented as integer triples (i, j, k) , which are indices into the vertex array. A deformable model such as SMPL or SMAL may be viewed as supplying a set of triangles, and a function

$$\nu(\theta, \beta) : \mathbb{R}^P \times \mathbb{R}^B \mapsto \mathbb{R}^{3 \times V} \quad (1)$$

which generates the 3D model for a given pose and shape. The mesh topology (i.e. the triangle vertex indices) is provided by the deformable model, and is the same for all shapes and poses we consider, so in the sequel we shall consider a mesh to be defined only by the 3D positions of its vertices.

In any given image, the model's 3D *position* (i.e. translation and orientation) is also unknown, and will be represented by a parametrization ϕ which may be for example translation as a 3-vector and rotation as a unit quaternion. Application of such a transformation to a $3 \times V$ matrix will be denoted by $*$, so that

$$\phi * \nu(\theta, \beta) \quad (2)$$

represents a 3D model of given pose and shape transformed to its 3D position.

We will also have occasion to talk about model *joints*. These appear naturally in models with an explicit skeleton, but more generally they can be defined as some function mapping from the model parameters to an array of 3D points analogous to the vertex transformation above. We consider the joints to be defined by post-multiplying by a $V \times J$ matrix K . The j^{th} column of K defines the 3D position of joint j as a linear combination of the vertices (this is quite general, as ν may include vertices not mentioned in the triangulation). A general camera model is described by a function $\pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$. This function incorporates details of the camera intrinsics such as focal length, which are assumed known. Thus

$$\kappa(\phi, \theta, \beta) := \pi(\phi * \nu(\theta, \beta) K) \quad (3)$$

is the $2 \times J$ matrix whose columns are 2D joint locations corresponding to a 3D model specified by (position, pose, shape) parameters (ϕ, θ, β) .

The model is also assumed to be supplied with a rendering function R which takes a vertex array in camera coordinates, and generates a 2D binary image of the model silhouette. That is,

$$R(\phi * \nu(\theta, \beta)) \in \mathbb{B}^{W \times H} \quad (4)$$

for an image resolution of $W \times H$. We use the differentiable renderer of Loper *et al.* [33] to allow derivatives to be propagated through R .

3 Method

The test-time problem to be solved is to take a sequence of input images $\mathcal{I} = [I_t]_{t=1}^T$ which are segmented to the silhouette of a single animal (i.e. a video with multiple animals is segmented multiple times), producing a sequence of binary silhouette images $\mathcal{S} = [S_t]_{t=1}^T$.

The computational task is to output for each image the shape, pose, and position parameters describing the animal’s motion.

As outlined above, the method has three parts. (1.) The discriminative front-end extracts silhouettes from video, and then uses the silhouettes to predict 2D joint positions, with multiple candidates per joint. (2.) Optimal joint assignment (OJA) corrects confused or missing skeletal predictions by finding an optimal assignment of joints from a set of network-predicted proposals. Finally, (3.) a generative deformable 3D model is fitted to the silhouettes and joint candidates as an energy minimization process.

3.1 Prediction of 2D joint locations using multimodal heatmaps

The goal of the first stage is to take, for each video frame, a $W \times H$ binary image representing the segmented animal, and to output a $W \times H \times J$ tensor of heatmaps. The network architecture is standard: a stacked hourglass network [34] using synthetically generated training data, but the training procedure is augmented using “multi-modal” heatmaps.

For standard unimodal heatmaps, training data comprises (S, κ) pairs, that is pairs of binary silhouette images, and the corresponding 2D joint locations as a $2 \times J$ matrix. To generate each image, a random shape vector β , pose parameters θ and camera position ϕ are drawn, and used to render a silhouette $R(\phi * \nu(\theta, \beta))$ and 2D joint locations $\kappa(\phi, \theta, \beta)$, which are encoded into a $W \times H \times J$ tensor of heatmaps, blurring with a Gaussian kernel of radius σ .

The random camera positions are generated as follows: the orientation of the camera relative to the animal is uniform in the range $[0, 2\pi]$, the distance from the animal is uniform in the range 1 to 20 meters and the camera height is in the range $[0, \frac{\pi}{2}]$. This smaller range is chosen to restrict unusual camera elevation. Finally, the camera “look” vector is towards a point uniformly in a 1m cube around the animal’s center, and the “up” vector is Gaussian around the model Y axis.

This training process generalizes well from synthetic to real images due to the use of the silhouette, but the lack of interior contours in silhouette input data often results in confusion between joint “aliases”: left and right or front and back legs. When these predictions are wrong and of high confidence, little probability mass is assigned to the area around the correct leg, meaning no available proposal is present after non-maximal suppression.

We overcome this by explicitly training the network to assign some probability mass to the “aliased” joints. For each joint, we define a list of potential aliases as weights $\lambda_{j,j'}$ and linearly blend the unimodal heatmaps G to give the final

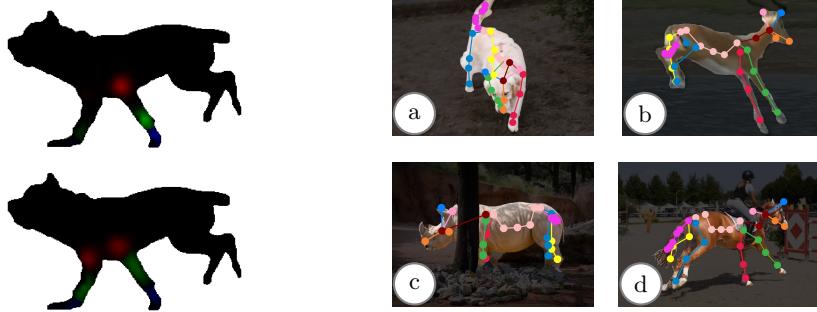


Fig. 2: Example predictions from a network trained on unimodal (top) and multi-modal (bottom) ground-truth for front-left leg joints.

Fig. 3: Example outputs from the joint prediction network, with maximum likelihood predictions linked into skeleton.

training heatmap H :

$$H_j(p) = \sum_{j'} \lambda_{j,j'} G(p; \kappa_{j'}, \sigma) \quad (5)$$

For non-aliased joints j (all but the legs), we simply set $\lambda_{j,j} = 1$ and $\lambda_{j,j'} = 0$, yielding the unimodal maps, and for legs, we use 0.75 for the joint, and 0.25 for the alias. We found this ratio sufficient to ensure opposite legs have enough probability mass to pass through a modest non-maximal suppression threshold without overly biasing the skeleton with maximal predicted confidence. An example of a heatmap predicted by a network trained on multimodal training samples is illustrated in Fig. 2.

3.2 Optimal joint assignment (OJA)

Since heatmaps generated by the joint predictor are multi-modal, the non-maximum suppression procedure yields multiple possible locations for each joint. We represent the set of joint proposals $X = \{x_{jp}\}$, where x_{jp} indicates the 2D position of proposal $p \in \{1, \dots, N_j\}$ associated with joint $j \in J$. Before applying the optimizer, we must select a subset of proposals $X^* \subseteq X$ which form a complete skeleton, i.e. precisely one proposal is selected for every joint. In this section we consider how to choose the optimal subset by formulating the problem as an extended optimal assignment problem.

In order to select a complete skeleton proposal from the set of joint proposals $\{x_{jp}\}$, we introduce a binary indicator vector $\bar{a}_j = \{a_{jp}\} \in \{0, 1\}^{N_j+1}$, where $a_{jp} = 1$ indicates that the p^{th} proposal for joint j is a correct assignment, and the $p = N_j + 1$ position corresponds to a *null proposal*, indicating that joint j has no match in this image. The null proposals are handled as described in each of the energy terms below. Let A be the jagged array $[\bar{a}_j]_{j=1}^J$ containing all assignment variables (for the current frame), and let $X^* = X(\bar{A})$ denote the subset of points

selected by the binary array A . Optimal assignment minimizes the function

$$L(A) = L_{\text{prior}}(A) + L_{\text{conf}}(A) + L_{\text{temp}}(A) + L_{\text{cov-sil}}(A) + L_{\text{cov-bone}}(A) \quad (6)$$

which balances agreement of the joint configuration with a learned *prior*, the network-supplied *confidences*, *temporal* coherence, and *coverage* terms which encourage the model to correctly project over the silhouette. Without the coverage terms, this can be optimized as a quadratic program, but we obtain better results by using the coverage terms, and using a genetic algorithm. In addition, the parameters A must satisfy the J constraints $\sum_{p=1}^{N_j+1} a_{jp} = 1$, that exactly one joint proposal (or the null proposal) must be selected for each joint.

L_{prior} : We begin by defining the prior probability of a particular skeletal configuration as a multivariate Gaussian distribution over selected joint positions.

The mean $\mu \in \mathbb{R}^{2J}$ and covariance $\Sigma \in \mathbb{R}^{2J \times 2J}$ terms are obtained from the training examples generated as above. The objective of OJA is to select a configuration X^* which maximizes the prior, which is equivalent to minimizing the Mahalanobis distance $(x^* - \mu)^T \Sigma^{-1} (x^* - \mu)$, which is given by the summation

$$L_{\text{prior}}(A) = \sum_j^J \sum_p^{N_j} \sum_k^J \sum_q^{N_k} a_{jp} a_{kq} (x_{jp} - \mu_j) \Sigma_{jk}^{-1} (x_{kq} - \mu_k) \quad (7)$$

This is a quadratic function of A , so $L_{\text{prior}}(A) = \text{vec}(A)^\top Q \text{vec}(A)$ for a fixed matrix Q , and can be formulated as a quadratic program (QP). Null proposals are simply excluded from the sum, equivalent to marginalizing over their position.

L_{conf} : The next energy term comes from the output of the joint prediction network, which provides a confidence score y_{jp} associated with each joint proposal x_{jp} . Then $L_{\text{conf}}(A) = \sum_j \sum_p -\lambda \log(y_{jp}) a_{jp}$ is a linear function of A , and λ_{conf} is a tunable parameter to control the relative contribution of the network confidences compared with that of the skeleton prior. Null proposals pay a fixed cost λ_{null} , effectively acting as a threshold whereby the null proposal will be selected if no other proposal is of sufficient likelihood.

L_{temp} : A common failure case of the joint prediction network is in situations where a joint position is highly ambiguous, for example between the left and right legs. In such cases, the algorithm will commonly alternate between two equally likely predictions. This leads to large displacements in joint positions between consecutive frames which are difficult for the later model fitting stage to recover from. This can be addressed by introducing a temporal term into the OJA. We impose a prior on the distance moved by each joint between frame t_0 and t_1 , which is given by a normal distribution with zero mean and variance $\sigma^2 = e^{\tau|t_1-t_0-1|}$. The parameter τ controls the strength of the interaction between distant frames. This results in an additional quadratic term in our objective function, which has the form $L_{\text{temp}} = a^\top T^{(t_0,t_1)} a$ for matrix $T^{(t_0,t_1)}$ given by

$$\left[T^{(t_0,t_1)} \right]_{jp,kq} = \begin{cases} e^{-\alpha|t_1-t_0-1|} \|x_{jp}^{(t_0)} - x_{kq}^{(t_1)}\|^2 & \text{if } j = k \\ 0 & \text{otherwise} \end{cases} \quad (8)$$



Fig. 4: Silhouette coverage loss. The error (shown in red) is the the distance between the median axis transform (right) and the nearest point on an approximate rendering (left).

Fig. 5: Bone coverage loss. One of the back-right leg joints is incorrectly assigned (left), leading to a large penalty since the lower leg bone crosses outside the dilated silhouette (right).

QP solution. Thus far, all terms in $L(A)$ are quadratic or linear. To optimize over an entire sequence of frames, we construct the block diagonal matrix \hat{Q} whose diagonal elements are the prior matrices $Q^{(t)}$ and the block symmetric matrix \hat{T} whose off-diagonal elements are the temporal matrices $T^{(t_0, t_1)}$. The solution vector for the sequence \hat{A} is constructed by stacking the corresponding vectors for each frame. The quadratic program is specified using the open source CVXPY library [35] and solved using the “*Suggest-and-Improve*” framework proposed by Park and Boyd [36]. It is initialized by choosing the proposal with the highest confidence for each joint. Appropriate values for the free parameters $\lambda_{\text{conf,temp}}$ and α were chosen empirically via grid search.

$L_{\text{cov-}\{\text{sil,bone}\}}$: The above quadratic formulation is sufficient to correct many errors in the raw output (which we later demonstrate in the experimental section), but suffers from an ‘overcounting’ problem, in which leg joint predictions both cover the same silhouette leg region, leaving another leg empty. We therefore extend the definition of $L(A)$ to include two additional terms.

$L_{\text{cov-sil}}$: penalizes large silhouette areas with no nearby selected joint. This term requires a precomputed set of silhouette sample points $Z \subseteq \mathbb{R}^2$, which we aim to “cover” as best as possible with the set of selected joints. Intuitively, the silhouette is considered well-covered if all sample points are close to *some* selected joint proposal. The set Z is generated from the medial axis transform (MAT)[37] of the silhouette, $Z^t = \text{MAT}(S^t)$ with a cubed loss strongly penalizing projection outside the silhouette:

$$L_{\text{cov-sil}}(A^t; X^t, Z^t) = \sum_i \min_j \|Z_i^t - \hat{X}_j^t\|^3 \quad (9)$$

$L_{\text{cov-bone}}$: is used to prevent bones crossing the background. The joint hierarchy is stored in a kinematic tree structure $K = \{\{j, k\} \text{ if joints } j, k \text{ are connected by a bone}\}$.

$$L_{\text{cov-bone}}(A^t; X^t, S^t, K) = \sum_{\{j, k\} \in K} \left(1 - \min_{\lambda \in [0:0.1:1]} S^t(\hat{X}_j^t + \lambda(\hat{X}_j^t - \hat{X}_k^t)) \right) \quad (10)$$

GA Solution. We minimize this more complex objective using a genetic algorithm (GA)[38], which requires defining a fitness function, “genes”, an initial population, crossover procedure, and mutation procedure. The *fitness function* is precisely the energy $L(A)$ given above, and the *genes* are vectors of J integers, rather than one-hot encodings. We begin with a population size of 128 genes, in which the first 32 are set equal to the max confidence solutions given by the network in order to speed up convergence. The remaining 96 are generated by selecting a random proposal for each joint. *Crossover* is conducted as standard by slicing genes in two parts, and pairing first and second parts from different parents to yield the next generation. In each generation, each gene has some probability of undergoing a *mutation*, in which between 1 and 4 joints have new proposals randomly assigned. Weights were set empirically and we run for 1000 generations. Examples of errors corrected by these two energy terms are shown in Fig. 4 and Fig. 5.

3.3 Generative model optimization

The generative model optimization stage refines model parameters to better match the silhouette sequence \mathcal{S} , by minimizing an energy which sums 4 terms:

Silhouette energy. The silhouette energy E_{sil} compares the rendered model to a given silhouette image, given simply by the L2 difference between the OpenDR rendered image and the given silhouette:

$$E_{\text{sil}}(\phi, \theta, \beta; S) = \|S - R(\phi * \nu(\theta, \beta))\| \quad (11)$$

Prior energy. The prior term E_{prior} encourages the optimization to remain near realistic shapes and poses. It can constrain pose only weakly because of the lack of training data from real poses. We adopt three prior energy terms from the SMAL model. The Mahalanobis distance is used to encourage the model to remain close to: (1) a distribution over shape coefficients given by the mean and covariance of SMAL training samples of the relevant animal family, (2) a distribution of pose parameters built over a walking sequence. The final term ensures the pose parameters remain within set limits.

$$E_{\text{lim}}(\theta) = \max\{\theta - \theta_{\text{max}}, 0\} + \max\{\theta_{\text{min}} - \theta, 0\}. \quad (12)$$

Joints energy. The joints energy E_{joints} compares the rendered model joints to the OJA predictions, and therefore must account for missing and incorrect joints. It is used primarily to stabilize the nonlinear optimization in the initial iterations, and its importance is scaled down as the silhouette term begins to enter its convergence basin.

$$E_{\text{joints}}(\phi, \theta, \beta; X^*) = \|X^* - \phi * \nu(\theta, \beta)\mathbf{K}(:, j)\| \quad (13)$$

Temporal energy. The optimizer for each frame is initialized to the result of that previous. In addition, a simple temporal smoothness term is introduced to penalize large inter-frame variation:

$$E_{\text{temp}}(\phi, \theta, \beta; X^*) = (\phi_t - \phi_{t+1})^2 + (\beta_t - \beta_{t+1})^2 \quad (14)$$

The optimization is via a second order dogleg method [39].

4 Experiments

Datasets. In order to quantify our experiments, we introduce a new benchmark animal dataset of joint annotations (BADJA) comprising several video sequences with 2D joint labels and segmentation masks. These sequences were derived from the DAVIS video segmentation dataset [7], as well as additional online stock footage for which segmentations were obtained using Adobe’s UltraKey tool [40]. A set of twenty joints on the 3D SMAL mesh were labeled, illustrated in Fig. 6. These joints were chosen on the basis of being informative to the skeleton and being simple for a human annotator to localize. To make manual annotation feasible and to ensure a diverse set of data, annotations are provided for every fifth frame.

The video sequences were selected to comprise a range of different quadrupeds undergoing various movement typical of their species. Although the dataset is perhaps insufficient in size to train deep neural networks, the variety in animal shape and pose renders it suitable for evaluating quadruped joint prediction methods.

4.1 Joint prediction

For the joint predictor ρ we train a stacked hourglass network [34]. Following state-of-the-art performance on related human 2D pose estimation datasets ([3], [4]), we construct a network consisting of 8 stacks, 256 features and 1 block. As input we provide synthetically-generated silhouette images of size 256×256 , which are obtained by randomly sampling shape and pose parameters from the SMAL model. The corresponding training targets are ground truth heatmaps produced by smoothing the 2D projected joint locations with a Gaussian kernel. Since we are working with synthetic data, we are able to generate training samples on the fly, resulting in an effectively infinite training set. A small adaptation was required to prevent the network degenerating to an unfavourable solution on silhouette input: foreground masks were applied to both ground truth silhouette and predicted heatmaps to prevent the network degenerating to an all-zero heatmap, which produces a reasonably good loss and prevents the network training successfully. The network was trained using the RMSProp optimizer for 40k iterations with a batch size of 18 and learning rate of 2.5×10^{-4} . The learning rate was decayed by 5% every 10k iterations. Training until convergence took 24 hours on a Nvidia Titan X GPU.

Joint accuracy is evaluated with the Probability of Correct Keypoint (PCK) metric defined by Yang and Ramanan [41]. The PCK is the percentage of predicted keypoints which are within a threshold distance d from the ground truth keypoint location. The threshold distance is given by $d = \alpha \sqrt{|S|}$ where $|S|$ is the area

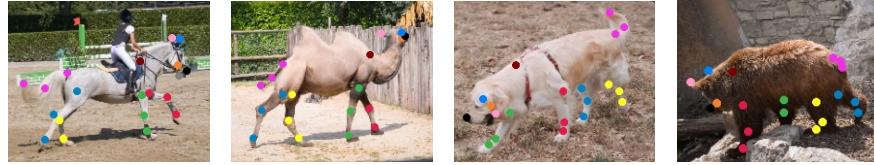


Fig. 6: Example joint annotations from the BADJA dataset. A total of 11 video sequences are in the dataset, annotated every 5 frames with 20 joint positions and visibility indicators.

of the silhouette and α is a constant factor which we set to $\alpha = 0.2$ for these experiments.

Fig. 3 shows a selection of maximum likelihood joint predictions on real world images. Note that despite being trained only on synthetic data, the network generalizes extremely well to animals in the wild. The performance extends even to species which were not present in the SMAL model, such as the impala and rhino. The network is also robust to challenging poses (3b), occlusions (3c) and distraction objects such as the human rider in (3d). It is however susceptible to situations where the silhouette image is ambiguous, for example if the animal is facing directly towards or away from the camera. Figure 11 contains examples of failure modes.

4.2 Optimal joint assignment

Following non-maximum suppression of the joint heatmaps obtained in Section 4.1, we apply OJA to select an optimal set of joints with which to initialize the final optimization stage. It can be seen that the OJA step is able to address many of the failure cases introduced by the joint prediction network, for example by eliminating physically implausible joint configurations (Fig. 7, row 1) or by resolving the ambiguity between the left and right legs (Fig. 7, row 2). Table 1 summarizes the performance of both the raw network predictions and results of the two OJA methods. Over most of the sequences in the BADJA dataset it can be seen that the use of coverage terms (employed by the OJA-GA model) improves skeleton accuracy. In particular, the bear, camel and rs_dog sequences show substantial improvements. The method does however struggle on the horsejump_high sequence, in which part of the silhouette is occluded by the human rider which adversely affects the silhouette coverage term. Across all sequences the selected OJA-GA method improves joint prediction accuracy by 7% compared to the raw network output.

4.3 Model fitting

The predicted joint positions and silhouette are input to the optimization phase, which proceeds in four stages. The first stage solves for the model’s global rotation and translation parameters, which positions the camera. We follow SMPLify [42] by solving this camera stage for torso points only, which remain

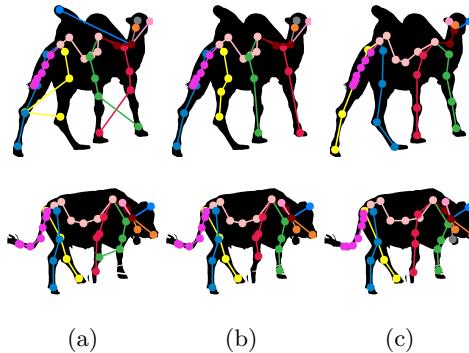


Fig. 7: Example skeletons from raw predictions (a), processed with OJA-QP (b), and OJA-GA (c).

	Raw	QP	GA
bear	83.1	83.7	88.9
camel	73.3	74.1	87.1
cat	58.5	60.1	58.4
cows	89.2	88.4	94.7
dog	66.9	66.6	66.9
horsejump-high	26.5	27.7	24.4
horsejump-low	26.9	27.0	31.9
tiger	76.5	88.8	92.3
rs_dog	64.2	63.4	81.2
Average	62.8	64.4	69.5

Table 1: Accuracy of OJA on BADJA test sequences.

Seq.	Family	PCK (%)		Mesh Seq.	Family	PCK (%)		Mesh
		Raw	OJA-GA			Raw	OJA-GA	
01	Felidae	91.8	91.9	38.2	06	Equidae	84.4	84.8
02	Felidae	94.7	95.0	42.4	07	Bovidae	94.6	95.0
03	Canidae	87.7	88.0	27.3	08	Bovidae	85.2	85.8
04	Canidae	87.1	87.4	22.9	09	Hippopotamidae	90.5	90.6
05	Equidae	88.9	89.8	51.6	10	Hippopotamidae	93.7	93.9

Table 2: Quantitative evaluation on synthetic test sequences. We evaluate the performance of the raw network outputs and quadratic program post-processing using the probability of correct keypoint (PCK) metric (see sec. 4.1). We evaluate mesh fitting accuracy by computing the mean distance between the predicted and ground truth vertices.

largely fixed through shape and pose variation. We then solve for all shape, pose and translation parameters and gradually decrease the emphasis of the priors. The silhouette term is introduced in the penultimate stage, as otherwise we find this can lead to the optimizer finding unsatisfactory local minima.

The final outputs of our optimization pipeline are shown in Fig. 10. In each of the cases illustrated the optimizer is able to successfully find a set of pose and shape parameters which, when rendered, closely resembles the input image. The final row of Fig. 10 demonstrates the generalizability of the proposed method: the algorithm is able to find a reasonable pose despite no camel figurines being included in the original SMAL model.

Comparison to other work. We compare our approach visually to that given by Zuffi *et al.* [9]. Recall that their results require hand-clicked keypoints whereas ours fits to points predicted automatically by the hourglass network, which was trained on synthetic animal images. Further, their work is optimized for single

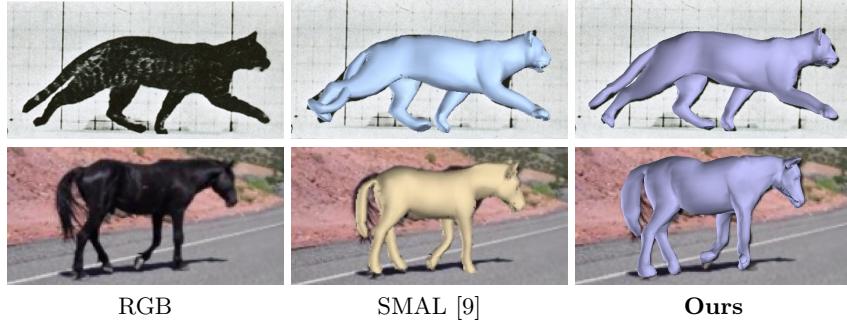


Fig. 8: Our results are comparable in quality to SMAL [9], but note that we do not require hand-clicked keypoints.



Fig. 9: Evaluating synthetic data. Green models: ground truth, Orange models: predicted. Frames 5, 10 and 15 of sequence 4 shown. Error on this sequence 22.9.

frame fitting and is tested on animals in simple poses, whereas we instead focus on the more challenging task of tracking animals in video. Fig. 8 shows the application of our model to a number of single frame examples from the SMAL result data [9].

Quantitative experiments. There is no existing ground truth dataset for comparing reconstructed 3D animal meshes, but an estimate of quantitative error is obtained by testing on synthetic sequences for a range of quadruped species. These are generated by randomly deforming the model and varying the camera position to animate animal motion, see Figure 9. Table 2 shows results on these sequences.

4.4 Automatic silhouette prediction

While not the main focus of our work, we are able to perform the full 3D reconstruction process from an input image with no user intervention. We achieve this by using the DeepLabv3+ network [2] as a front-end segmentation engine to automatically generate animal silhouettes. This network was trained on the PASCAL VOC 2012 dataset, which includes a variety of animal quadruped classes. An example result generated using the fully automatic pipeline is shown in Fig. 1.



Fig. 10: Example results on various animals. From left to right: RGB input, extracted silhouette, network-predicted heatmaps, OJA-processed joints, overlay 3D fit and alternative view.



Fig. 11: Failure modes of the proposed system. Left: Missing interior contours prevent the optimizer from identifying which way the dog is facing. Middle: The model has never seen an elephant, so assumes the trunk is the tail. Right: Heavy occlusion. The model interprets the tree as background and hence the silhouette term tries to minimize coverage over this region.

5 Conclusions

In this work we have introduced a technique for 3D animal reconstruction from video using a quadruped model parameterized in shape and pose. By incorporating automatic segmentation tools, we demonstrated that this can be achieved with no human intervention or prior knowledge of the species of animal being considered. Our method performs well on examples encountered in the real world, generalizes to unseen animal species and is robust to challenging input conditions.

6 Acknowledgements

The authors would like to thank Ignas Budvytis and James Charles for technical discussions, Peter Grandi and Raf Czlonka for their impassioned IT support, the Biggs' family for the excellent title pun and Philippa Liggins for proof reading.

References

1. Food and Agriculture Organization of the United Nations: FAOSTAT statistics database (2016) [Online; data retrieved from FAOSTAT on 21-November-2017].
2. Chen, L., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. (2018)
3. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: New benchmark and state of the art analysis. In: Proc. CVPR. (2014)
4. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollar, P., Zitnick, L.: Microsoft COCO: Common objects in context. In: Proc. ECCV. (2014)
5. Johnson, S., Everingham, M.: Clustered pose and nonlinear appearance models for human pose estimation. In: Proc. BMVC. (2010) 12.1–12.11
6. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results (2012)
7. Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., Sorkine-Hornung, A.: A benchmark dataset and evaluation methodology for video object segmentation. In: Proc. CVPR. (2016)
8. Wilhelm, N., Vögele, A., Zsoldos, R., Licka, T., Krüger, B., Bernard, J.: Furyexplorer: visual-interactive exploration of horse motion capture data. In: Visualization and Data Analysis 2015. Volume 9397. (2015) 93970F
9. Zuffi, S., Kanazawa, A., Jacobs, D., Black, M.J.: 3D menagerie: Modeling the 3D shape and pose of animals. In: Proc. CVPR, IEEE (2017) 5524–5532
10. Shotton, J., Fitzgibbon, A., Blake, A., Kipman, A., Finocchio, M., Moore, B., Sharp, T.: Real-time human pose recognition in parts from a single depth image. In: Proc. CVPR, IEEE (2011)
11. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: Proc. CVPR. Volume 1. (2017) 7
12. Zuffi, S., Kanazawa, A., Black, M.J.: Lions and tigers and bears: Capturing non-rigid, 3D, articulated shape from images. In: Proc. CVPR. (2018)
13. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: Proc. CVPR. (2009)
14. Li, X., Qi, Y., Wang, Z., Chen, K., Liu, Z., Shi, J., Luo, P., Loy, C.C., Tang, X.: Video object segmentation with re-identification. The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops (2017)
15. Khoreva, A., Benenson, R., Ilg, E., Brox, T., Schiele, B.: Lucid data dreaming for object tracking. The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops (2017)
16. Cashman, T.J., Fitzgibbon, A.W.: What shape are dolphins? Building 3D morphable models from 2D images. IEEE TPAMI **35** (2013) 232–244
17. Reinert, B., Ritschel, T., Seidel, H.P.: Animated 3D creatures from single-view video by skeletal sketching. In: Graphics Interface. (2016) 133–141
18. Khamis, S., Taylor, J., Shotton, J., Keskin, C., Izadi, S., Fitzgibbon, A.: Learning an efficient model of hand shape variation from depth images. In: Proc. CVPR, IEEE (2015)
19. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: SMPL: A skinned multi-person linear model. ACM Trans. Graphics (Proc. SIGGRAPH Asia) **34** (2015) 248:1–248:16
20. Chen, Y., Kim, T.K., Cipolla, R.: Inferring 3D shapes and deformations from single views. In: Proc. ECCV, Springer (2010) 300–313
21. Favreau, L., Reveret, L., Depraz, C., Cani, M.P.: Animal gaits from video. In: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation. (2004) 277–286

22. Tan, V., Budvytis, I., Cipolla, R.: Indirect deep structured learning for 3D human body shape and pose prediction. In: Proc. BMVC. (2017)
23. Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. In: Computer Vision and Pattern Recognition (CVPR). (2018)
24. Wiles, O., Zisserman, A.: Silnet : Single- and multi-view reconstruction by learning from silhouettes. In: Proc. BMVC. (2017)
25. Andriluka, M., Roth, S., Schiele, B.: Monocular 3D pose estimation and tracking by detection. In: Proc. CVPR, IEEE (2010) 623–630
26. Pishchulin, L., Andriluka, M., Gehler, P., Schiele, B.: Poselet conditioned pictorial structures. In: Proc. CVPR, IEEE (2013) 588–595
27. Bulat, A., Tzimiropoulos, G.: Human pose estimation via convolutional part heatmap regression. In: Proc. ECCV, Springer (2016) 717–732
28. Mathis, A., Mamidanna, P., Cury, K.M., Abe, T., Murthy, V.N., Mathis, M.W., Bethge, M.: DeepLabCut: Markerless pose estimation of user-defined body parts with deep learning. Technical report, Nature Publishing Group (2018)
29. Wang, P., Shen, X., Lin, Z., Cohen, S., Price, B., Yuille, A.L.: Joint object and part segmentation using deep learned potentials. In: Proc. ICCV. (2015) 1573–1581
30. Wang, J., Yuille, A.L.: Semantic part segmentation using compositional model combining shape and appearance. In: Proc. CVPR. (2015) 1788–1797
31. Chen, X., Mottaghi, R., Liu, X., Fidler, S., Urtasun, R., Yuille, A.: Detect what you can: Detecting and representing objects using holistic models and body parts. In: Proc. CVPR. (2014)
32. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ADE20K dataset. In: Proc. CVPR. (2017)
33. Loper, M.M., Black, M.J.: OpenDR: An approximate differentiable renderer. In: Proc. ECCV, Springer (2014) 154–169
34. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Proc. ECCV, Springer (2016) 483–499
35. Diamond, S., Boyd, S.: CVXPY: A Python-embedded modeling language for convex optimization. Journal of Machine Learning Research **17** (2016) 1–5
36. Park, J., Boyd, S.: General heuristics for nonconvex quadratically constrained quadratic programming. (2017)
37. Blum, H.: A transformation for extracting new descriptors of shape. Models for Perception of Speech and Visual Forms, 1967 (1967) 362–380
38. Holland, J.H.: Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press (1992)
39. Lourakis, M., Argyros, A.A.: Is Levenberg-Marquardt the most efficient optimization algorithm for implementing bundle adjustment? In: Proc. ICCV. (2005) 1526–1531
40. Adobe Systems Inc.: Creating a green screen key using ultra key. (<https://helpx.adobe.com/premiere-pro/atv/cs5-cs55-video-tutorials/creating-a-green-screen-key-using-ultra-key.html>) Accessed: 2018-03-14.
41. Yang, Y., Ramanan, D.: Articulated human detection with flexible mixtures of parts. TPAMI **35** (2013) 2878–2890
42. 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: Proc. ECCV, Springer (2016) 561–578