3D Animal Reconstruction with Deformable Template Models

Benjamin Biggs

Supervisor: Dr. Andrew Fitzgibbon
Prof. Roberto Cipolla

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Jesus College

May 2021

"Our perfect companions never have fewer than four feet."
Colette (1873 – 1954)

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Benjamin Biggs
May 2021

Acknowledgements

This work would not have been possible without the dedicated support from my PhD supervisors, Andrew Fitzgibbon and Roberto Cipolla. I would also like to thank the dedicated staff at GlaxoSmithKline, in particular the Pharma Supply Chain Tech Digital Innovation group, and particularly my line manager Patrick Hyett and project sponsor Julie Huxley-Jones.

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.

The authors would like to thank the GSK AI team for providing access to their GPU cluster, Michael Sutcliffe, Thomas Roddick, Matthew Allen and Peter Fisher for useful technical discussions, and the GSK TDI group for project sponsorship.

The authors would like to thank Richard Turner for useful technical discussions relating to normalizing flows, and Philippa Liggins, Thomas Roddick and Nicholas Biggs for proof reading. This work was entirely funded by Facebook AI Research.

Abstract

TODO. Across many sectors concerned with animal husbandry, there is growing support for a system able to continuously monitor captive animals. Within farmyards, zoos, veterinary centres, animal research facilities and many others, humans typically take responsibility for identifying signs of disease or distress within their animal populations. While this can be effective, a significant challenge is posed when a small number of humans are expected to care for large animal groups.

This report discusses the development of a system to track, monitor and react to signs of poor physiological and psychological health among captive animals. In this work, it is proposed that a useful component of such a system would be the recovery of a detailed per-frame 3D animal reconstruction from an input video sequence. This is achieved through an approach which combines discriminative machine learning with generative model fitting to recover strong shape and pose attributes.

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.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Motivation

Animal welfare is an important concern for business and society, with an estimated 70 billion animals currently living under human care [**?**]. Across multiple industries, monitoring and assessing animal health is achieved by measuring individuals' body shape 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).

Of course, employing humans to monitor large animal populations is costly and can lead to data bias. For example, prey animals such as rodents are known to alter their behaviour in the presense of perceived predators. To overcome this, a number of animal monitoring systems have been designed, especially for use in clinical work. Many systems are 'invasive', meaning they require animals to undergo a surgical operation (generally to implant a tracking chip) before monitoring can take place. Although these systems can offer detailed biometric data (such as blood pressure, ECG etc.), the implantation procedure is costly and can cause stress to the animals, leading to welfare concerns and complex behavioural effects.

Since even crudest system can ensure some basic health standard (e.g. to check that *some* activity occurs over a given time period), some systems further attempt to monitor the animals' physical activity.

To overcome this, a number of non-invasive systems are available. The most basic of these can still be surprisingly effective, for example checking that *some* motion has occured during a given time period can offer an important health standard. More advanced systems can also estimate energy and inquisitiveness levels of the target animals by analysing motion patterns over time. Most systems achieve this by either placing floor-level pressure pads [**?**], or by installing an overhead camera which performs simple visual blob detection via colour thresholding [**?**], [**?**]. One such open source system instructs the user to set a colour tolerance that masks all non-animal pixels and provide expected maximum and minimum animal sizes (in pixels) to help eliminate noise. While this is

effective at tracking multiple animals with distinctive colour when placed in an arena with a solid, fixed background, it does not work well in many scenarios, e.g. outdoors. The presence of changing light levels, casting of shadows across tracking targets or moving backgrounds (e.g. foliage) make such thresholds ineffective. Further, this system's ability to distinguish between multiple tracked subjects is hindered when animals cross one another, as two individual blobs temporarily become one, and from then on are difficult to resolve.

Some work has been done in automatic behavioural scoring for rodents, in which up to ten predefined behaviours can be visually recognized. These approaches typically employ machine learning algorithms, which are taught to recognize behaviours present in a video stream by analyzing a large set of pre-collected examples. For 'normal' behaviours (e.g. drinking, eating etc.) this can be a viable approach and by analyzing the changing frequency of such behaviours can indeed offer insights into underlying conditions. However, these systems cannot be readily extended to handle more serious conditions (e.g. animals experiencing a seizure), due to the ethical concerns associated with collecting sufficient examples for training.

In addition, many Of course, even if a behaviour detection system were built for a range of animal species this

Unfortunately, all these approaches suffer as they fail to reconstruct a full 3D mesh.

1.2 Approach

This thesis focuses on developing methods for recovering a 3D model of an animal oa tracking system to enable recovery of a per-frame 3D animal reconstruction from an image or video stream.

The system should apply to a wide range of animal species without significant customization. Success in this endeavour would enable real-time changes of a known skeletal structure to be programmatically analysed to completely model an animal's movements. These behaviour patterns could then be interpreted to form a profile for each animal in a batch, taking into account expected norms for their species as well as their individual personality traits. When animals are first brought into a facility, they are given some time to acclimatize to their new surroundings before a clinical study begins. The application could make use of this period to refine behaviour models to their particular characteristics without being influenced by external factors. The system would then begin monitoring the population, storing detailed analytics and reacting to any deviations to an animal's unique behaviour profile. As a simple example, should a typically lively and sociable dog suddenly begin exhibiting signs of withdrawal from the group, this would indicate a cause for concern and be stored in that animal's 'virtual log book'. In some cases, an animal may begin to exhibit signals that demand immediate attention, such as a dramatic and sudden energy drop that may indicate pain. The application could handle such events by sending an SMS text message to an on-call veterinary professional, to alert them of the specific problem and thereby enable a rapid response. These real-time diagnostics could then be aggregated and displayed on a dashboard screen, visible to all laboratory technicians. A concept drawing is shown in Figure **??**.

1.2.1 Problem definition

A major challenge of this work is to develop and adapt methods for resolving the inherent ambiguity associated with recovering a 3D model from 2D input data. This challenge can be overcome by augmenting the input video sequence (Figure **??**) with strong prior knowledge about the target species class (e.g. quadruped body measurements). This prior knowledge can be divided into two components: a *shape* prior that enforces topological (e.g. order of body parts) and measurement constraints (e.g. length of limbs), and a *pose* prior that defines likely limb configurations and can be used to rule out those which are anatomically impossible.

Fig. 1.1 An example input video sequence.

An example output showing the recovery of a 3D model from an input 2D monocular video is shown in Figure **??**:

Fig. 1.2 Sample output printed from Deformable Mesh Animation [**?**].

A distinction should be made between two common tracking techniques: (1) discriminative body part recognizers and joint position predictors, and (2) 3D reconstructions via generative model fitting. Discriminative predictors have become the dominant paradigm in human body tracking to facilitate common use-cases, such as gesture detection or controllerless gameplay. However, recovering 3D models from human subjects is a growing field. Applications are found in fashion to faciliate online 'try-ons' for virtual clothing [**?**], in animation and visual effects to generate virtual characters from live actor performances [**?**], and in healthcare for tracking patients' body weight over time [**?**]. It is hypothesized that recovering a full 3D animal reconstruction is necessary to enable the intended diagnostic purposes of this animal work. In particular, returning only joint positions or body parts

may be insufficient to estimate animal weight. If this can be realized, identifying behavioural changes from the reconstruction is expected to be a relatively straightforward machine learning problem.

A typical method for recovering 3D structure from tracking targets is using a *model fitting* approach, in which a 3D object representative of the target class is adapted to recreate the performance of the target. This method involves: (1) parameterizing a representative 3D *template mesh* with terms that represent shape and pose attributes and (2) defining an optimizer to adapt to these per-frame parameter settings to an input video sequence. An example of a template mesh is shown in Figure **??**.

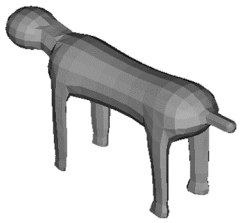


Fig. 1.3 An example prior, in this case a template mesh.

Shape attributes capture variation between different members of the target class and remain constant for a particular individual. For example, shape parameters may be adapted to vary a model's height and weight. However, pose attributes generally capture limb positions and joint angles, and therefore tend to vary considerably during a capture sequence. Figure **??** highlights the difference by keeping pose parameters fixed while shape attributes are varied between the three models [**?**].

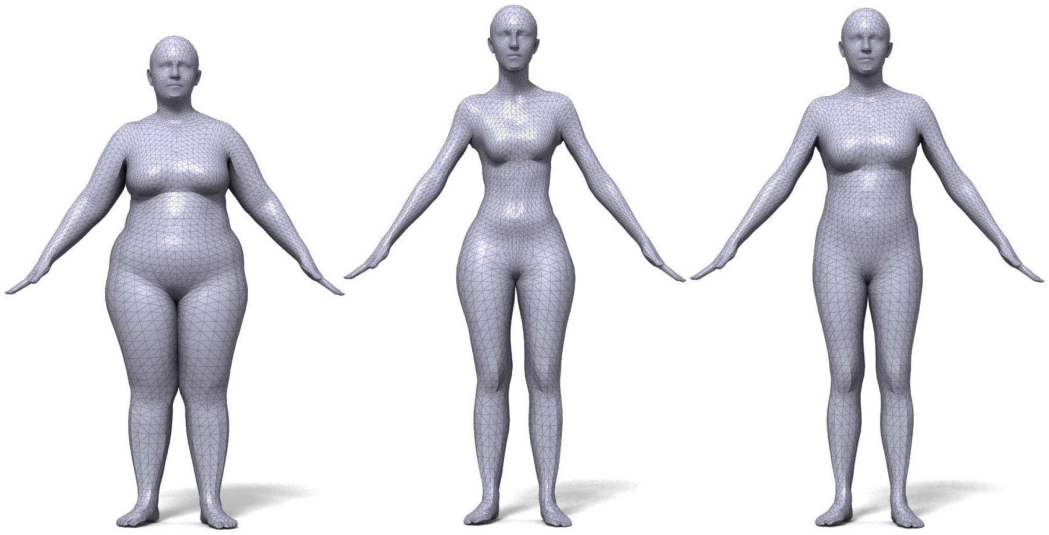


Fig. 1.4 Varying human shape parameters while pose remains fixed. Reprinted from [**?**].

Shape and pose parameters derived from a video sequence can be applied to the template mesh to generate a digital version of the same activity. If successful, the changing parameters should appear to adapt (or morph) the template mesh to faithfully reconstruct the performance given by the original live animal. In early experimentation, in which tracking targets are restricted to the same species,

the template can be chosen to be a close shape fit to the target animal, thereby largely reducing the problem to finding optimal per-frame pose parameters. However, tracking examples are eventually broadened to include a wide range of animal species.

1.2.2 Relation to human reconstruction

Given that human tracking is now an established computer vision subfield, and the growing interest in analysing human behaviour from CCTV camera tracking systems, it is natural to ask whether the techniques used in this work transfer to the animal case. As an emerging research field, animal tracking presents many challenges in common with human gait and pose tracking problems, particularly in accurately monitoring morphable objects which frequently self-occlude. However, notable additional challenges are posed by the large shape and texture variation between animal tracking candidates and also due to the lack of available training data which could otherwise be employed to train deep neural networks. An advantageous aspect of tracking animals over humans is the simple fact that animals tend not to wear clothing, which in humans causes significant shape and appearance variability.

The previous chapter discussed the primary objective of this work, which is to recover full 3D shape and pose from a live input video sequence exhibiting an animal subject. As explained, the major challenge common to all methods operating on monocular RGB input is to resolve the inherent depth ambiguity associated with recovering a 3D model from 2D input. Competitive methods achieve this by relying on strong motion cues [**?**] or (if available) by incorporating strong prior knowledge of the tracking target. Strong shape and pose priors (e.g. body part configuration, acceptable body part lengths, likely joint positions etc.) are available for this problem, so this report will focus on analysing these methods in the literature.

All solutions face an important design decision, which is to make a distinction between features of an input sequence the system should aim to model and to which it should remain invariant. For example, nearly all human systems aim to model the angle between a tracking target's upper and lower leg region, but nearly all will attempt to remain invariant to skin colour variation between candidates. The next two sections discuss examples of systems in which this decision has been made differently, generally according to the intended real-world application.

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:

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 [**?** **?** **?**]. Of these two classes of labelling, animal *segmentation* data is available in datasets such as MSCOCO [**?**], PASCAL VOC [**?**] and DAVIS [**?**]. 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 [**?**], it remains impractical to consider mocap for a family of tigers at play.

1.3 Contributions

In summary, the contributions of this thesis are as follows:

1. We demonstrate a robust framework for 3D animal reconstruction using deformable template models

1.4 Co-Authored Papers

Extracts from this thesis appear in the following co-authored publications and preprints. **??** contains work from:

> **Benjamin Biggs**, Thomas Roddick, Andrew Fitzgibbon and Roberto Cipolla. *Creatures Great and SMAL: Recovering the Shape and Motion of Animals from Video*. ACCV 2018, Oral Presentation.

?? contains work from:

> **Benjamin Biggs**, Oliver Boyne, James Charles, Andrew Fitzgibbon and Roberto Cipolla. *Who Left the Dogs Out? 3D Animal Reconstruction with Expectation Maximization In the Loop*. ECCV 2020.

?? contains work from:

> **Benjamin Biggs**, Sebastien Ehrhadt, Hanbyul Joo, Benjamin Graham, Andrea Vedaldi and David Novotny. *3D Multi-bodies: Fitting Sets of Plausible 3D Human Models to Ambiguous Image Data*. NeurIPS 2020, Spotlight Presentation.

1.5 Thesis Structure

The following thesis chapters discuss methods for deriving 3D animal reconstructions from monocular input images and video. **??** begins by covering the necessary background and motivation before an in-depth literature review in **??**. **??** discusses an approach for animal reconstruction learning only from synthetic training data. **??** describes an end-to-end and real-time technique applied to the challenging dog category. **??** introduces a method for handling input images with significant ambiguity. The final **??** summarizes the work and offers a discussion for worthy future endeavours in the field.

Chapter 2

Related Work

2.1 Introduction

This chapter discusses background and methods related to 3D shape and pose estimation for articulated subjects. We begin with techniques for correspondence prediction – a classical method which discusses work focussing on 3D shape and pose estimation for animals. 3D shape and pose estimation for articulated subjects – primarily animals and humans. We initially focus on skeletal prediction techniques, which output either sets 2D or 3D keypoints. Such techniques are of value to our objective of 3D surface reconstruction, as have been employed in multi-stage pipelines where keypoints are predicted first before a subsequent model fitting stage. The chapter continues with an evaluation of so-called 'model-free' techniques, which operate without an explicit template prior.

2.2 Predicting point correspondences

Before delving into methods for 3D reconstruction, it is first necessary to discuss techniques for identifying *point correspondences*. Point correspondences have a long history in computer vision for associating the same real-world location as it is represented by multiple camera views or on a 3D model surface. In a multi-image scenario, determining reliable correspondences between image pairs can be used to greatly reduce the ambiguity when reconstructing 3D scenes from 2D images. Even with only a single image available, correspondences can be predicted between the image and a representative 3D template mesh. This 3D-to-2D correspondence type is important for constraining the class of model fitting algorithms (discussed in depth later) which operate by aligning a 3D template mesh to a given 2D image. Of course, determining point correspondences is made more difficult in the presence of particular nuisance factors. In the case of animal imagery, we must associate points on a non-rigid object with independently moving parts (articulated), deal with frequent self-occlusion in which limbs overlap each other from the perspective of the camera, occlusion caused by environmental factors (e.g. trees, fences, humans etc.), varied and unknown backgrounds and a range of complex

lighting conditions (including shadows). Throughout this section, the methods highlighted will be appraised against their suitability in this complex setting.

2.2.1 Relating separate views of the same object/scene

The first class of techniques focuses on classical approaches for determining corresponding image points taken of precisely the same (and almost always rigid) object. Early techniques focused on stereo [**?**] or optical flow [**?**] imagery, and matched image points based on finding regions with similar pixel intensities. Due to the adverse effects caused by changeable environmental factors (e.g. lighting) would have on the appearance of the real-world location when captured in separate images, attention moved towards designing schemes with improved robustness. Improvements were achieved when matching points based on local *mid-level features* such as edges and corners, which have greater invariance to colour changes caused by lighting effects. Typical pipeliens would first identify *interest points* (typically corners [**? ? ?**] or blobs [**?**]), from which local image patches could be compared according to either Squared Sum of Intensity Differences (SSD) or a cross-correlation (CC) scheme. Steady improvements were then made through the design of ever-improving feature descriptors, which encode local image information around points and aim for invariance against common transformations (e.g. viewpoint, rotation and scaling). Progress in this field arguably reached maturity with the advent of SIFT [**?**], which encodes points according to local histograms of graident orientations and was later speed-up by SURF [**?**] and DAISY [**?**]. There have been modern attempts to learn sophisticated feature representations using convolutional neural network architectures [**? ?**], which are shown to offer still further improvement.

The primary aim of these systems is to derive point correspondences between multiple views of the same object, usually as depicted in stereo images or between successive frames of a video. Unfortunately, by matching points based on local geometric features learnt from few image examples, these techniques do not readily extend to identifying correspondences between different instances of the same category. For example, matching SIFT features is likely to result in poor quality correspondences if tested on two dogs of different breeds due to the differing appearance and body geometry. For similar reasons, this class of techniques tend to deteriorate when tested on articulated objects since the object's structure can change and cause self-occlusion between views. The techniques are also known to suffer in scenarios with significant viewpoint changes (e.g. image of the front/back of an animal), since there are few correpsonding points available for matching. Finally, these techniques do not directly offer a method for identifying correspondences between an image and a representative 3D mesh. Although some work exists that extends some of the aforementioned feature descriptors (e.g. 3D-SIFT [**?**]) to 3D, matching typically requires a photorealistic 3D scan of the 2D subject which we cannot assume as input for our problem.

2.2.2 Predicting semantically-meaningful keypoints

This section will explore an alternative class of methods for identifying point correspondences. So far, the approaches described do not detect correspondences with any semantic meaning; in other words, the returned points cannot truly be 'named' and there is no guarantee the same points (or even the same number of points) will be identified in different test images. Instead, this section will focus on techniques which predict a set of keypoint locations which are specified in a pre-defined list (for example: nose, tail tip, toe). In general, data-driven machine learning algorithms are used in order to learn an association between image appearance information and semantic keypoint labels. The techniques fall into two general categories: the former set of *supervised techniques* rely on large image datasets manually annotated with keypoint locations, and the latter set of *unsupervised techniques* learn the association through other means.

Supervised techniques

Early work in the supervised prediction of landmarks began through the refinement of object detection methods to predict fine-grained object part labels and eventually progressed to keypoint locations. Perhaps the earliest techniques in this category made use of face part annotations (referred to as fiducial points) to align target faces to improve the face recongition accuracy. Human detection and pose estimation methods progressed from simple bounding box representations [**?**], to object part prediction [**? ?**], poselets [**?**] and subsequently 2D keypoint localization [**? ?**]. Most commonly, methods aim to predict the location of important 2D human joints (such as the shoulders and wrists) in order to roughly approximate the subject's skeletal pose. For this reason, this task is commonly referred to as *2D human pose estimation*. The earliest techniques represented humans as a graph of parts [**?**] and fit shape primitives (e.g. cylinders [**?**]) to detected edges. Tree-based graphical models known as pictorial structures [**?**] were adopted and later made efficient [**?**]. Improvements were made with models capable of expressing complex relationships betwen joints, such as flexible part mixtures [**? ?**].

Before the popularization of modern deep learning architectures, various methods made use of features computed underneath predicted 2D landmark locations for fine-grained image classification tasks. For this reason, there are limited examples of keypoint datasets for animal categories such as dogs [**?**] and birds [**?**]. **??** of this thesis will discuss StanfordExtra, a new dataset complete with annotated keypoint locations and segmentation masks for 12,000 dog images, encompassing 120 different breeds. At the time of publication, StanfordExtra is the largest annotated animal dataset of its kind.

Recent works in 2D pose estimation typically employ convolutional neural networks (CNNs) due to the complex feature represenations that can be learnt for joints that, when applied discriminatively, enable accurate recongition. An early example [**?**] learnt a pose embedding space with a CNN, and employed a nearest neighbour search algorithm to regress a pose. Later, deeper CNN models were used to regress facial point [**?**] and full body [**?**] landmarks. More recent works improve robustness

by regressing keypoint confidence maps [**?**] rather than 2D keypoints directly, enabling spatial priors to be applied to remove outliers [**? ? ? ? ? ? ?**]. More recent methods are able to directly produce accurate confidence maps through a multi-stage pipeline [**?**]. Of particular note are hourglass [**?**] (relied upon in this thesis **??**) and multi-level [**? ?**] structures, which combine global reasoning of full-body attributes and of fine-grained details. A related class of methods [**? ?**] focus on *dense* human pose estimation, which relate all 2D image pixels to a representative 3D surface of the human body.

Modern techniques in 2D human pose estimation demonstrate impressive accuracy on in-the-wild datasets, and deal with with parsing multiple subjects in challenging poses and in the presence of various occluders. However, part of what enables these achievements is the prevalance of large 2D keypoint datasets which can be used for training. Further discussion of available 2D keypoint datasets has been left for **??**, in which they are considered in-depth. Further discussion on the history and advances in 2D human pose estimation are comprehensively reviewed in [**? ?**].

Unsupervised learning

As this thesis focuses on developing methods for animal reconstruction, it is useful to review techniques which operate without large 2D keypoint training datasets, which are scarce for animal subjects. Note that the methods in this section all describe approaches for determining point correspondences between different scenes. Under consideration are methods based on transfer learning, unsupervised learning and methods based on weak-supervision.

Early correspondence techniques include dense alignment methods including SIFT-flow [**?**] which employed optical flow methods to match image using SIFT features, and Bristow et al. [**?**] who demonstrate a method for learning per-pixel semantic correspondences using geometric priors. They also show examples on various animal categories. Recent unsupervised techniques learn *category-specific* semantic priors by employing deep networks on large image collections.

Zhou et al. [**?**] demonstrate a method for solving correspondences across an image collection by enforcing cycle consistency. Kanazawa et al. [**?**] introduce WarpNet which predicts a dense 2D deformation field for bird images by learning from synthetic thin-plate spline warps generated on extracted silhouettes. Thewlis et al. [**?**] apply a similar trick, by ensuring a consistent mapping of warped facial images to a spherical coordinate frame and show results on human and cats. Jakab et al. [**?**] show they can estimate 2D human pose without training data data by leverging that between two frames of a simple video sequence, human body shape and texture remains reasonably similar but the pose (including global rotation) varies. They therefore construct an architecture that, given a pair of frames (I, J) defines a network f that given frame I predicts a 2D location vector y. The system then combines this vector y with the second frame J and trains a secondary network g to reconstruct the original frame I. Due to the limited capacity of v, the fact that apart from the pose, most of the information necessary for reconstruction is already available in J, the network eventually learns to encode 2D pose coordinates using v.

Transfer learning describes a family of methods in which a machine learning model is first *pre-trained* to solve a related task (often making use of secondary dataset with may be larger in size) in order to accumulate knowledge which offers an advantage when solving the original task. DeepLabCut [**?**], LEAP [**?**] and DeepPoseKit [**?**] exemplify such techniques, in which existing architectures [**? ? ? ?**] are first trained to predict 2D human pose (making use of the large available datasets), and are then repurposed to predict 2D animal keypoints using few (generally 100s) training examples. Cao et al [**?**] demonstrate a cross-domain adaptation technique, which transfers knowledge gained from a modestly-sized animal dataset to unseen animal types. There are also dense estimation techniques, which extend DensePose [**?**] described above to proximal animal classes [**?**], such as chimpanzees, by aligning the geometry between the animal category to humans for which data is plentiful.

2.3 Modelling articulated subjects

The design of 3D morphable models (3DMMs) has a significant recent history in computer vision research. A 3DMM is a statistical model designed to represent the structure, deformation and apperance space of for a particular object category. Such a model can be constructed for any object category for which a dense point-to-point correspondence can be established between instances. For example, a 3DMM can be designed to represent medium-sized quadrupeds but perhaps not for general animal categories. How, for instance, would one sensibly determine correspondences between a dog and an octupus? 3DMMs have been used extensively as a strong 3D prior to aid various 3D reconstruction algorithms. They are, however, most influential for problems with the most ambiguity: particularly when dealing with articulated objects (e.g. animals or humans), when only a single monocular RGB image is available or when no paired 3D training data is available.

Blanz and Vetter [**?**] presented the first 3DMM, which expressed a low-dimensional face space space learnt by aligning various face scans. This work, presented over two decades ago, has been recognized with an impact paper award for the continued applications for the ideas presented. Indeed, the approach introduced has found applications far beyond faces [**? ? ?**], including for cars [**?**], other human body parts including the hands [**?**] and ears [**?**], the human body surface [**? ?**] and a restricted set of animal categories [**? ?**].

This section will cover methods for modelling articulated subjects, focussing primarily methods for human bodies and animals.

2.3.1 Constructing 3D morphable models

3D deformable models are typically represented by a polygon mesh. A polygon mesh $M = (V, T)$ is a collection of vertices, edges bound by vertex pairs, and polygons bound by sequences of edges and vertices [**?**]. Although other convex shapes are allowed, this thesis only has need to discuss

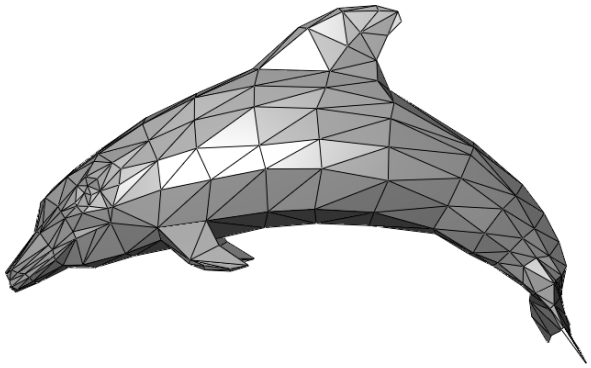


Fig. 2.1 A polygon mesh [**?**].

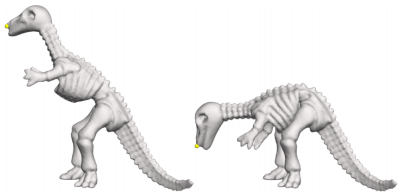


Fig. 2.2 Dinosaur mesh undergoing ARAP deformation, obtained by translating the highlighted yellow vertex. Reprinted from [**?**].

triangular mesh polygons, which henceforth will be referred to as *triangles*. An example mesh is shown in Figure **??**.

A 3D morphable model can then be constructed by deforming a template mesh M on n-vertices to a set of 3D training examples. Generally, this optimization will have a significant degrees of freedom so it is necesssary to employ techniques for regularization. One such regularizer is known as the *As Rigid as Possible* scheme:

Definition 2.3.1 (As Rigid as Possible)**.** As Rigid as Possible (ARAP) surface deformation [**?**] is a distance function that measures similarity between two meshes with corresponding vertices. For two vertex sets V and W, ARAP minimizes over $N = |V|$ rotation matrices. Note $j \sim i$ indicates vertex indices j adjacent to vertex index i:

$$D(V,W) = \min_{R_{1..N}} \sum_{i=1}^{N} \sum_{j \sim i} ||(V_i - V_j) - R_i(W_i - W_j)||^2 \tag{2.1}$$

This distance function can be incorporated into an energy-based optimizer as a regularization function. By considering how small vertex regions overlap, the function can be used to discourage 'unnatural movement', e.g. shearing effects, over mesh faces. ARAP regularizers are particularly useful in cases in which there is no prior knowledge of the mesh. Figure **??** shows an example of a dinosaur mesh undergoing ARAP deformation, obtained by translating the highlighted yellow vertex.

Once aligned, a d-dimensional *shape space* can then be defined (with $d << n$), where each $w \in \mathbb{R}^d$ gives rise to a vertex configuration in $\mathbb{R}^3$ (with unchanged triangulation). In this way, every plausible

3D example has a parameter vector $w \in \mathbb{R}^d$ that generates it. This construction can then be interpreted as a *generative* model. However, very few selections of $w \in \mathbb{R}^d$ will generate a plausible-looking 3D mesh. This can then be interpreted probabilistically, by defining a density function $f(w)$ that defines the likelihood that a realistic 3D example would be represented by w in shape space.

2.3.2 Modelling shapes (e.g. faces)

The concepts raised above were first introduced in the seminal work of Blanz and Vetter [**?**]. They define a linear generator function based on principle component analysis (PCA) in order to map d-dimensional parameter vectors to the set of n vertex coordinates. In particular they use the mapping:

$$g(\alpha) = \bar{c} + E\alpha \tag{2.2}$$

where $g : \mathbb{R}^d \mapsto \mathbb{R}^{3n}$ is the generator function, $\bar{c} \in \mathbb{R}^{3n}$ is the mean 3D face in the training dataset and $E \in \mathbb{R}^{3n \times d}$ is a matrix containing the d most dominant eigenvectors computed over shape residuals $\{c_i - \bar{c}_i\}$.

This construction assumes 3D faces in this d-dimensional parameter space follow a multivariate normal distribution (a design decision explored further in this thesis **??**). In addition, the function $f(w)$ which defines the likelihood shape space vector α represents a plausible face, is therefore given by the Mahalanbois distance of α to the origin.

Note that this formulation additionally enables the definition of facial expressions. For example, Blanz and Vetter defined an expression (e.g. surprise) according to the difference in shape space between a expressive and neural face of the same subject. This then enabled the formulation above to be factored into identity and expression components:

$$g(\alpha_{idt}, \alpha_{exp}) = \bar{c} + E_{idt}\alpha_{idt} + E_{exp}\alpha_{exp} \tag{2.3}$$

where E_{idt}, E_{exp} are the basis vectors of the identity and expression space and $\alpha_{idt}, \alpha_{exp}$ are the coefficients. As noted by Lewis et al. [**?**], the basis vectors of the expression space above can be interpreted as a data-driven *blendshape model*: a standard approach in the animation industry for representing facial expressions as a linear combination of target faces. This concept will later reemerge in a section discussing corrective blendshapes uses in SMPL [**?**].

As identified by Blanz and Vetter, improved modelling of finer details (particularly around the eye or nose regions) can be obtained through local modelling. Various authors [**?**] began manually segmentating the face into parts and learning individual PCA representations for them. Later, segmentations were automatic and learned based on displacement patterns found in the training dataset. Next, approaches were adopted based on hierarchical, multi-scale frameworks [**? ?**]. Possibly the closest to later sections which require a focus on *pose deformation* is the work of Wu et al [**?**]. who combine a a local shape space model with an anatomical bone structure that helps regularize deformation.

A standard challenge in face modelling is towards reconstructing appearance, typically incorporating albedo and illumination (although frequently these are not factored, in which case apperance is generally referred to as *texture*). Early work modelled shape and texture independently [**?** **?**], although recent techniques show solving for these factors jointly enable constraints to be applied due to correlations present. Perhaps most interesting are the recent techniques among these [**?** **?**], who propose methods based on deep convolutional models to jointly model shape and texture.

2.3.3 Modelling articulation (e.g. hands)

3D morphable models have also influenced work in 3D hand tracking and modelling. Human hands servce multiple purposes in everyday life, serving a mechanism to handle tools/objects, expressing emotion and aiding (or even as the primary tool for) communication. As a result, hands (and particularly fingers) exhibit complex articulation patterns which are best characterized as 3D *rotations*. Compared to the previous section, in which face shape variation could be represented as as an abstract linear basis learnt from scans, an advantage to modelling hands is the modes of articulation can be defined in advance.

In particular, human hand motion is controlled by a hierarchical bone structure referred to as a *skeleton*. The point at which two bones meet is referred to as a *joint* and can be used to define acceptable centers of articulation. The direction and magnitude of the articulation can then be neatly expressed as a 3D rotation.

This formulation helps provide insight into why the abstract linear basis (shape space) introduced in the previous section would be poor choice for modelling hands. Deformation is here characterized in terms of 3D rotations, and 3D rotations are non-linear with respect to the input angle. This is easily shown:

Definition 2.3.2 (3D Rotations)**.** The simplest kind of 3D rotation is an *elementary rotation* and involves a rotation around a single axis of a coordinate system. For example, the following matrix represents a rotation by an angle γ around the x axis:

$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{bmatrix} \tag{2.4}$$

One can then apply this matrix to an input point $p \in \mathbb{R}^3$ to compute the new position p' after rotating γ around the x axis:

$$p' = R_x(\theta)p \tag{2.5}$$

This formulation can then be extended to represent any 3D rotation as the composition of elementary rotations. For example, a 3D rotation can be decomposed into a γ rotation around the

x-axis (pitch), followed by a β rotation around the y-axis (yaw) and finally by an α rotation around the z-axis (roll).

$$R = R_z(\alpha)R_y(\beta)R_x(\gamma) \tag{2.6}$$

One can see immediately that affecting a 3D rotation (i.e. computing the new position of the points) is a non-linear function of the input angle. It is necessary, therefore to describe an alternative technique for low-dimensional and efficient mesh deformation, which relies on *rigging* and *linear blend skinning*.

Definition 2.3.3 (Skeletal Rigging and Linear Blend Skinning (LBS))**.** In cases that the articulated object is known in advance, it is common to augment a representative 3D mesh model with an internal skeleton that approximates the biological counterpart. This is achieved through a process known as *rigging*.

Formally, a skinned mesh consists of a set of rigged vertices $V \subseteq \mathbb{R}^3 \times \mathbb{R}^{|J|}$, a set of faces $F \subseteq V^3$ and joint transformation matrices $J \subseteq \mathbb{R}^{3\times 4}$. Each vertex $v = (x,s) \in V$ consists of positional coordinate $x \in \mathbb{R}^3$ and a weight vector $s \in \mathbb{R}^{|J|}$ which describes the level of influence each joint $j \in J$ has over its movement. Many approaches exist for assigning weights, but perhaps the simplest is to build a vector with entries corresponding to the distance from the vertex to each joint centre. Skinning weight vectors are normalized such that their entries sum to one, and for computational reasons, the number of non-zero elements is typically limited to 2 or 4. The weakness of such models is that artifacts and other unrealistic deformations can occur around the model joints, particularly for meshes that model non-linear structures such as humans. However, the technique is frequently used in computer graphics and game design when a character's shape is known ahead of time.

Once a mesh has been suitably rigged, Linear Blend Skinning (LBS) can be used to apply the mesh deformation. Typically, a user assigns a transformation (e.g. rotation and translation) to each 'joint' and LBS computes the an updated position of vertex $v = (x,s)$ where x is the original location and $s = \{s_j\}$ are the skinning weight influence of joints j. The matrix U_j (occasionally referred to as a the *reference pose*) is the mapping from the bone's "default" coordinate system to world coordinates. As a result, it need only be computed (and inverted) once since it is invariant to changing joint angles. In general, a user will supply the matrices D_j to define the mapping from the bone's deformed coordinate system to world coordinates.

The updated position of an input point x can then be calculated by LBS:

$$LBS(D,x;U,s) = \sum_j s_j D_j U_j^{-1} x \tag{2.7}$$

Note this formulation is made slightly more complicated in the case of kinematic trees, since the

Similarly to the approach mentioned above, this formulation can again be seen as a generative 3D model. In particular, an output vertex positions v' can be computed from a vector of 3D joint rotations θ and input vertex position v as follows:

$$v' = LBS(R(\theta), v) \tag{2.8}$$

With the above formulation, it is now possible to give an overview of recent hand tracking literature. In many cases, 3D morphable hands models are built to reflect the 27 biological hand bones (occasionally except the capal bones which join the fingers to the wrists). Of course, while the major source of hand deformation is due to 3D pose, some sources of variation are present due to hand size and finger propotions.

Allen et al [**?**] handle this variation adapting a 3D surface with displacement maps with various constraints designed to avoid self-intersections before adopting the linear blend skinning formulation defined above. Rhee et al [**?**] learn a shape deformation space (with a similar technique to that of faces) and user-specific skinning weights for LBS. Albrecht undergo a laborious process to create extremely detailed hand models through laser scanninng plaster casts [**?**]. A significant advance was presented by Taylor et al [**?**]. who presented a method to learn a personalized hand model (although not a shape basis) given an input video sequence (with depth) taken of a user slowly articulating their fingers. Ballan et al. [**?**] follow a similar process by making use of multi-view input.

2.3.4 Modelling the human body surface

However, of all human categories, the works of most relevance to this thesis are those which represent the entire human body surface. It is first important to characterize these two deformation modes which must be overcome with modelling algorithms. Firstly, there is considerable variation in the *shape* characteristics between different human subjects. Humans not only vary in their heights and weights, but also in their body part proportions, muscle density, fat etc. Secondly, humans exhibit significant *pose* variation, characterized by the range of motion of body parts (e.g. arms and legs). In general, pose is likely to change for a individual subject over a sequence.

The earliest deformable 3D models of the human body was presented by Allen at al [**?**] (although [**?**] came soon after with similar ideas). Allen et al. learnt a PCA shape space model from 250 registered body scans cound in the CAESAR dataset. The model was articulated through a set of pose parameters, which use linear blend skinning to interpolate rotation matrices assigned to the joint to transform model veertices. Unfortunately, this approach suffers from artefacts around joint locations, due to a loss of volume. For this reason, it is important to note that pose and shape are not entirely independent; in fact, body shape does indeed change due to pose variation. Imagine for example, how a fatty stomach region would deform during a walking sequence. SCAPE [**?**] improved over this by introducing a model equipped with both body shape variation and pose-dependent shape changes, expressed in terms of triangle deformations (rather than vertex displacements, see [**?**] for a comprehensive overview). An important advance was made by Hasler et al. [**?**], who learn two linear blend rigs: one for pose and one for body shape. In this model, shape change was controlled through the introduction of abstract bones that further deform the vertices.

Perhaps the most significant advance however, was the introduction of the Skinned Multi-Person Linear (SMPL) model of Loper et al. [**?**]. SMPL follows a similar design philosophy to SCAPE by decomposing shape into identity-dependent and pose-dependent components. However, unlike scape, SMPL adopts a vertex-based skinning approach based on corrective blend shapes. The model's shape space is first taught how human beings deform through pose changes using 1786 high-resolution 3D scans of different subjects in a wide variety of poses. Following alignment to a template mesh, a linear model for each biological gender is created from the CAESAR dataset [**?**] using principal component analysis (PCA). SMPL can then be viewed as a function, which makes use of a shape basis and linear blend skinning to map a set of pose and shape parameters to a set of vertex locations. Precisely, *pose* is given as a set of 3D rotations (per-joint and global) in axis-angle form $\theta \in \mathbb{R}^{24}3$. *Shape* is then given as coefficients for a learned shape basis $\beta \in \mathbb{R}^{10}$. The SMPL function can then be viewed as:

$$v = \text{SMPL}(\theta, \beta) + t \tag{2.9}$$

where $v \in \mathbb{R}^{6890\times 3}$ and $t \in \mathbb{R}^3$ is a global translation parameter. Further details on SMPL have been left to **??** of this thesis, which makes use of the model to examine uncertainty when deriving 3D reconstructions of ambiguous input imagery.

More recently, SMPL has been combined with face and hand models to add expressive capabilities [**? ? ?**]. CAPE [**?**] also shows how to add a clothing parameter to effectively model humans in clothing, a challenge solved by learning a shape prior over freeform vertex deformations Techniques have also been developed to model human clothing – a common challenge generally handled by allowing SMPL model vertices to vary independently to the provided blend shapes. SMPL has also been recently improved with STAR [**?**] which constructs a part-based shape space (closely related to the local PCA space discussed in the earlier shape section of this literature review). They show this new parameterization is much more efficient (uses approximately 20% of the model parameters of SMPL) and avoids capturing spurious long-range correlations present in the training dataset. They also show a method for learning shape-dependent pose-corrective blendshapes, that better model how individuals with different body shapes deform with motion. Tangential work of Xu et al. [**?**] train an end-to-end network and learn 3D human body model parameters (including faces and hands) for an input artist model using variational auto-encoders and normalizing flows. This work will be further explored **??** in which these generative models will be fully examined.

2.3.5 Modelling animals

There is still relevatively little work specifically focusing on the 3D scanning [**?**] and modelling of animal categories. The variation in animal shape and sizes combined with the practical challenges associated with scanning live animal subjects (particularly in attaching traditional motion capture equipment) make scanning a difficult task. As a result, there is a significant lack of real 3D animal training data available in the public domain which could otherwise have been employed to build 3D deformable models. As with humans, animals deformations can again be factored into shape

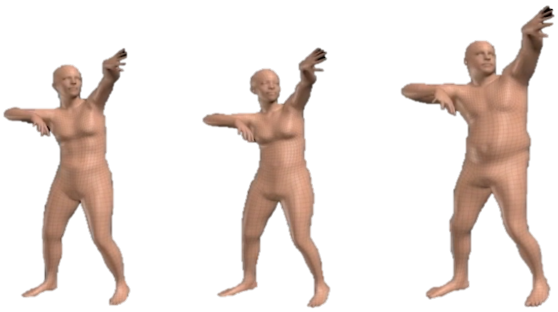


Fig. 2.3 SMPL model showing pose-invariant shape changes, reprinted from [**?**].

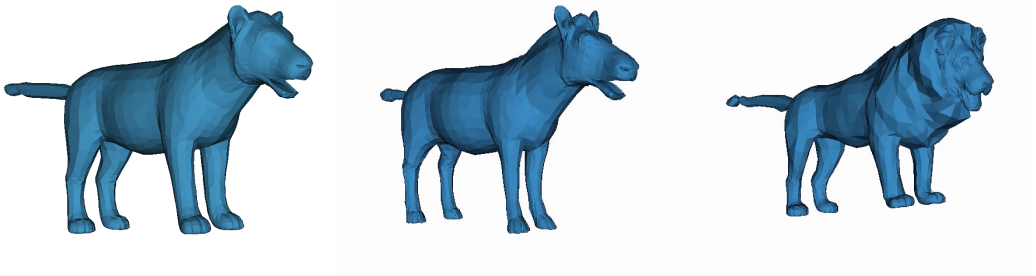
(e.g. variation mostly due to identity) and pose (variation due to articulated motion). However, the enormous diversity among animal species and even between individual breeds results in a much more complex shape space.

Some early work by Favreau et al [**?**] describe a method for animating an artist-designed rigged 3D model, by tracking a 2D sequence. Chen et al. [**?**] learn a shape space by registering 11 3D shark models downloaded from the Internet. Cashman et al. [**?**] learn a morphable model of dolphin shapes by adapting a representative 3D model to 2D images. Ntouskos et al. [**?**] fit geometric primitives to manually-segemented animal parts generated from an input collection. Reinert et al. [**?**] demonstrates an effective method for fitting generalized cylinders to an input video sequence supplied with sketched limb tracks. They demonstrate reconstructed results with 3D texture on a few quadruped sequences. So far, none of these techniques for animal reconstruction explicitly factor shape and pose.

SMAL

A similar technique to that used to build the SMPL model has been recently used to build a Skinned Multi-Animal Linear Model (SMAL) [**?**], a generative animal model exhibiting realistic 3D shape (see Figure **??**) and pose (see Figure **??**). Due to the lack of available motion capture data for animal subjects, the SMAL model is learnt from a set of 41 3D scans of toy figurines in arbitrary poses. The figurines span five quadruped families, and included examples of lions, cats, tigers, dogs, horses, any many more, although notably for this work no rodent toys were included. The paper introduces a new technique to accurately align each toy scan to a common template, allowing the shape space to be learnt.

From the paper, SMAL is defined as a function $\text{SMAL}(\theta, \beta)$ parameterized by pose-invariant shape $\beta \in \mathbb{R}^{41}$ (again, coefficients of a low-dimensional shape space) and pose $\theta \in \mathbb{R}^{32\times 3}$ (including global rotation). There are three pose parameters for each of the 32 body joints and an additional three to express the global rotation. Global translation γ is expressed by a further three parameters. The SMAL function returns a triangulated surface comprising 6890×3 vertex locations. **??** and **??** of this thesis make use of the SMAL model in order to reconstruct various quadruped categories.



(a) Default SMAL mesh. (b) SMAL in horse shape. (c) SMAL in lion shape.

Fig. 2.4 SMAL with varying shape parameters.

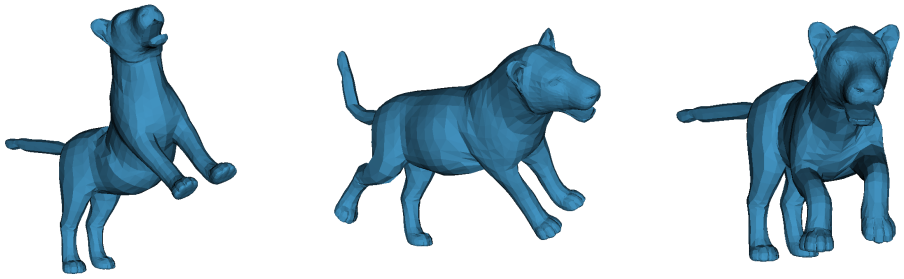


Fig. 2.5 SMAL with varying pose parameters.

2.4 Methods for monocular reconstruction of articulated subjects

Having discussed methods for modelling articulated subjects, this section will discuss approaches for reconstructing the 3D shape and pose of a subject from a monocular image or video. It is important to note that this task is challenging and fundamentally ill-posed. In common with other challenging 3D reconstruction tasks, input images will typically exhibit variation in camera view, lighting and environmental occlusion. However, 3D reconstruction pipelines for articulated subjects must also deal with variation due to body shape, body pose, clothing and self-occlusion (body parts obscuring other parts). In addition, the challenge of reconstructing 3D models from 2D images is also inherently ambiguous. As explained by Toshev and Szegedy [**?**], even if 2D structure can be determined (for example, using 2D keypoint prediction), the subsequent 'lifting' step to recover 3D remains ill-posed, as the space of consistent 3D poses for given 2D landmark locations is infinite. It is for this reason that the history of monocular 3D reconstruction makes extensive use of 3D morphable models (or other geometric, temporal, structural priors), as they provide necessary optimization contraints.

The theme of this section is therefore to discuss how 3D morphable models (3DMMs) can be incoporated into 3D reconstruction pipelines. In general, algorithms take as input an image or video and predict a set of 3D model parameters α (often factored into shape β and pose θ). Once determined, the output parameters are then supplied to the morphable model's generator function $g : \alpha \mapsto \mathbb{R}^3$ (e.g. SMPL : $(\beta, \theta) \mapsto \mathbb{R}^3$ or SMAL : $(\beta, \theta) \mapsto \mathbb{R}^3$) to produce vertex positions $V \subseteq \mathbb{R}^3$ for the model $M = (V, T)$ with fixed triangulation T. For completeness and comparison, this section will make some

mention of the small class of 3D reconstruction methods for articulated subjets which operate without an explicit 3D morphable model. Although currently a developing area, current work in this category typically requires either paired 3D training data, employ alternative (and arguably more restrictive) shape priors (e.g. symmetry constraints) or produce results of significantly lower fidelity.

Methods which align a parametric 3D model to monocular input date back as far as 1963, in a seminal paper by Roberts [**?**]. Roberts presents a method which optimizes parameters for viewpoint and cuboidal shape primitives to reconstruct a 2D line image. Model-based methods have also been applied to understand object structure, starting with fitting of geometric primitives [**?**] and later with Active Shape Models [**?**] which learn deformation priors from a provided training set. Perhaps due to the numerous commercial applications, the majority of recent work in 3D shape and pose recovery focuses particuarly on *humans* as a special case. The first example of such an approach is the seminal work of Blanz and Vetter [**?**] who built the first 3D morphable face model by aligning 3D scans and optimized the parameters to provide a fit to a single image. Since then, the research community has collected a multitude of open source human datasets which provide strong supervisory signals for training deep neural networks. These include include accurate 3D deformable template models [**?**] generated from real human scans, 3D motion capture datasets [**? ?**] and large 2D datasets [**? ? ?**] which provide keypoint and silhouette annotations. The combination of these publically available datasets and their incoporation into deep learning pipelines have led to impressive reconstruction results when tested on in-the-wild human images and videos. Unfortunately, the diversity among animal subjects and the practical challenges associated with data capture have resulted in few datasets being made available. Despite appearing superficially similar to human tracking, these factors result in specific challenges to animal tracking which must be carefully handled. Perhaps for this reason, the body of related literature for animal tracking is considerably sparser. The remainder of this section will focus on methods for 3D pose estimation, followed by 3D shape and pose reconstruction of human and animal bodies. Further discussion on techniques for body part (e.g. face, hands) reconstruction are deferred to the following survey papers [**? ?**].

2.4.1 3D Pose Estimation

Techniques for 3D pose estimation output a set of 3D keypoint locations which can be combined to form a skeletal outline. Apart from basic limb measurements, no other shape detail (e.g. surface definition, object density etc.) is obtained. However, it should be noted that this output form is often perfectly satisfactory depending on the intended application. In particular, this family of techniques have found numerous applications in controllerless gaming (e.g. Microsoft Kinect [**?**]), motion capture (e.g. for digital character generation [**?**]), gait analysis (e.g. identifying lameness in cattle [**?**]) and many more.

The general approach is to recover a 3D skeleton such that the 3D joints project to known or estimated 2D joints subject to anatomical priors. Early approaches in this category fit human stick figures with various constraints, including assumptions of fixed limb lengths [**?**], length ratios [**?**] or

that limb lengths are isometric across individuals and vary only in global scaling [**?**]. More advanced techniques built statistical models of shape variation using anthropometric tables or learnt them from motion capture data [**?**].

A broad category of appraoches for this are methods for *non-rigid structure from motion* [**?**]. The general formulation is to express a 3D skeleton $S \in \mathbb{R}^{3\times P}$ on P points as a linear combination of basis shapes $S_1, \ldots, S_k$ where $S_i \in \mathbb{R}^{3\times P}$. Precisely:

$$S = \sum_{i=1}^{K} l_i \cdot S_i \quad S, S_i \in \mathbb{R}^{3\times P} \quad l_i \in \mathbb{R} \tag{2.10}$$

Assuming scaled orthographic projection, the following expression represents the projection of P points of S into 2D image coordinates (u_i, v_i):

$$\begin{bmatrix} u_1 & u_2 & \ldots & u_P \\ v_1 & v_2 & \ldots & v_P \end{bmatrix} = R \cdot (\sum_{i=1}^{K} l_i \cdot S_i) + T \tag{2.11}$$

or equivalently:

$$\begin{bmatrix} u_1 & u_2 & \ldots & u_P \\ v_1 & v_2 & \ldots & v_P \end{bmatrix} = \begin{bmatrix} l_1 R & \ldots & l_K R \end{bmatrix} \cdot \begin{bmatrix} S_1 \\ S_2 \\ \ldots \\ S_K \end{bmatrix} \tag{2.12}$$

This can then be extended to handle multiple views of the subject taken over a monocular video sequence. Let $(u_i^{(t)}, v_i^{(t)})$ denote the tracked 2D point at timestep t. This gives rise to the following system, taken over N timesteps:

$$\underbrace{\begin{bmatrix} u_i^{(1)} & \ldots & u_P^{(1)} \\ v_i^{(1)} & \ldots & v_P^{(1)} \\ u_i^{(2)} & \ldots & u_P^{(2)} \\ v_i^{(2)} & \ldots & v_P^{(2)} \\ & \ldots & \\ u_i^{(N)} & \ldots & u_P^{(N)} \\ v_i^{(N)} & \ldots & v_P^{(N)} \end{bmatrix}}_{W} = \underbrace{\begin{bmatrix} l_1^{(1)} R^{(1)} & \ldots & l_K^{(1)} R^{(1)} \\ l_1^{(2)} R^{(2)} & \ldots & l_K^{(2)} R^{(2)} \\ & \ldots & \\ l_1^{(N)} R^{(N)} & \ldots & l_K^{(2)} R^{(N)} \end{bmatrix}}_{Q} \cdot \underbrace{\begin{bmatrix} S_1 \\ S_2 \\ \ldots \\ S_K \end{bmatrix}}_{B} \tag{2.13}$$

This shows the tracking matrix W can be factored into 2 matrices: Q which contains the camera pose $R^{(t)}$ and configuration weights $l_1^{(t)}, \ldots, l_K^{(t)}$ per frame t. B encodes the K basis shapes S_i. This system can be factored with singular value decomposition to yield the shape basis S_i, per-frame camera rotations R and per-frame configuration weights l. A number of techniques follow this formulation [**? ? ?**], but start with a shape basis learnt from available motion capture datasets (e.g. CMU [**?**]).

More recent approaches were designed to be fully automatic. Shotton et al. [**?**] designed a commerically-available system for 3D human skeletal tracking which required a depth sensor. A generative 3D body model was used to synthesize a large training dataset of depth images with corresponding body part labels. Density estimators for each body part are then used in combination to localize body joints with a calculated confidence value. Taylor et al. [**?**] predict dense correpsondences between image pixels (again, with depth so in $\mathbb{R}^3$) and a representative 3D human body model, again by training on synthetic depth images. **??** of this thesis demonstrates a technique for predicting keypoints by training on synthetic *silhouette* data, rendered from an animal deformable body model, which overcomes the need for depth imagery at test time.

Automatic monocular approaches often take advantage of 2D keypoint or body part detectors when reasoning about 3D skeletons. Simo-Serra et al. [**? ?**] form a probabilistic model that models both 3D pose and 2D keypoints jointly, overcoming noise among 2D body parts. Other approaches [**? ?**] employ a two-stage pipeline; they begin by localizing 2D joint positions on an input image before running a subsequent optimization step that 'lifts' these to a 3D pose. Tangential work [**?**] takes uses detected 2D joints to perform a nearest neighbour search in a 3D mocap dataset. The most recent two-stage pipelines rely on deep convolutional networks to predict keypoints. Examples of such systems include DeepPose [**?**], an approach which employs a CNN to reason jointly about 2D landmark detection and 3D pose estimation from single RGB images. Pishchulin et al. [**?**] later introduced DeepCut which extends DeepPose to the multi-person case.

State-of-the-art techniques now operate as a direct regression to a 3D pose. Most often, paired 3D training data (such as is available from datasets such as Human3.6M [**?**]) is required which is generally expensive to obtain, particularly for animal categories. One branch of approaches [**?**] predicts body configuration in terms of angles. Other approaches include Pavlakos et al. [**?**], who use a 2D joint predictor [**?**] followed by a deep architecture to regress 3D heatmaps. Moreno-Noguer [**?**] learn a pairwise distance matrix from 2D-to-3D space in order to allow unlikely 3D predictions to be ruled out with a suitable prior. These techniques were designed under the assumption that neural networks would struggle to learn a 'lifting' function from 2D to 3D pose. This assumption was corrected by Martinez et al. [**?**] who demonstrate the effectiveness of a simple architecture at regressing accurate 3D keypoints from 2D predictions. This technique was later interpreted probabilistically by The technique was interpreted probabilistically by Li et al. [**?**], who handled ambiguity in the 2D-to-3D lifting problem with a mixture density network. Related work that predicts a depth segmentation (so not strictly 3D keypoints) is SURREAL [**?**] who train their network with data generated synthetically with a 3D human body model.

2.4.2 Model-based human shape and pose

This section will discuss methods for reconstructing a full 3D *dense* human from a monocular image or video sequence. Early work in this category fit shape primitives combined into a kinematic tree to silhouettes extracted from the input [**? ? ?**]. The introduction of the 3D deformable human body

model known as SCAPE [**?**] enabled various fitting approaches. Sigal et al [**?**] compute shape features from manually extracted silhouettes and use a mixture of experts formulation for predicting SCAPE model parameters. Later, Guan et al. [**?**] fit the SCAPE model to provided keypoints, extracted silhouettes, edges and shading cues. They also define an interpenetration term that penalizes self-intersecting body parts, although this does not lead to easy optimization. Hasler et al. [**?**], Zhou et al. [**?**] and Chen et al. [**?**] present a similar approach, although show optimization only to input keypoints and manually or semi-manually (e.g. GraphCut [**?**]) extracted silhouettes.

A significant advance was made by the introduction of SMPLify [**?**], the first fully-automatic method for monocular 3D human pose and shape reconstruction. Many of the concepts presented by SMPLify are used throughout this thesis, making it worthy of study.

Fitting a 3D model to 2D keypoints

SMPLify works by fitting the SMPL [**?**] model to a set of 2D image locations predicted by DeepCut [**?**], a deep convolutional neural network. For an input image I, DeepCut predicts a set of image keypoint locations $J_{\text{est}} \in \mathbb{R}^{23\times 2}$ which correspond to locations on the 3D SMPL mesh J_{SMPL}. Precisely $J_{\text{SMPL}} = R_\theta(J(\beta))$ where $J(\beta)$ computes 3D skeleton positions from SMPL shape parameters β, and R_θ is the global rigid transformation effected by SMPL pose parameters θ. A model fitting approach is then used to align the SMPL model to the predicted keypoint positions. This is achieved through optimizing the SMPL parameters (β,θ), global translation t and camera parameters K, subject to priors over pose, shape and limb interpenetration priors.

The key energy term used in the optimization (and indeed throughout this thesis) is given by $E_{\text{J}}(\beta,\theta;K,J_{\text{est}})$ and measures the weighted 2D distance between estimated keypoints J_{est} and the corresponding SMPL joints J_{SMPL}.

$$E_J(\beta,\theta,t;K,J_{\text{est}}) = \sum_{\text{joint},i} w_i \rho(\Pi_K(J_{\text{SMPL,i}} - J_{\text{est,i}})) \tag{2.14}$$

The weighted 2D distance is implemented using the Geman-McClure [**?**] penalty function ρ which helps deal with noisy DeepCut estimates. SMPLify implements Π_K perspective camera model with known (or roughly initialized) focal length although others opt for orthographic projection. The following definition provides a quick primer for this:

Definition 2.4.1 (Primer on camera geometry)**.** Perspective projection is a function which maps a 3D structure to blah blah.

$$2X = Y \tag{2.15}$$

Orthographic projection is the following:

$$3X = Z \tag{2.16}$$

The full energy formulation is then given as:

$$E(\beta,\theta) = E_J(\beta,\theta;K,J_{\text{est}}) + \lambda_\theta E_\theta(\theta) + \lambda_\alpha E_\alpha(\theta) + \lambda_{\text{sp}} E_{\text{sp}}(\theta;\beta) + \lambda_\beta E_\beta(\beta) \tag{2.17}$$

where the following energy terms are employed, balanced according to the λ scalar weights:

- $E_\theta(\theta)$ is referred to as a *pose prior* which favours more likely poses by assigning large punishment to those that deviate from known poses collected from a large dataset.
- $E_\beta(\beta)$ is referred to as a *shape prior* which favours more likely pose-invariant shape configurations by assigning large punishment to those that deviate from known shapes collected from a large dataset.
- $E_\alpha(\theta)$ is a *joint limit* prior which ensures particular joints remain within acceptable angle limits. For example, a knee joint in a human model should be prohibited from bending more than 5 degrees upwards.
- $E_{sp}(\theta;\beta)$ is an *interpenetration* term, which can only be defined in such shape modelling approaches. Using both shape and pose from the model, it is possible to determine if any limbs are self-intersecting, or intersect other parts of the body and assign appropriate penalty.

SMPLify has recently undergone subsequent variations, including Huang et al. [Huang Multiview] who fit SMPL to multi-view images and Pavlakos et al. [SMPL-X fitting] who extend SMPL with hand and facial expression parameters and follow a similar fitting procedure. **??** of this thesis will introduce a *self-supervised* version of SMPLify that uses synthetic data for training, thereby overcoming the need for a large 2D dataset with manually-labelled keypoints.

An example result can be seen in Figure **??**:

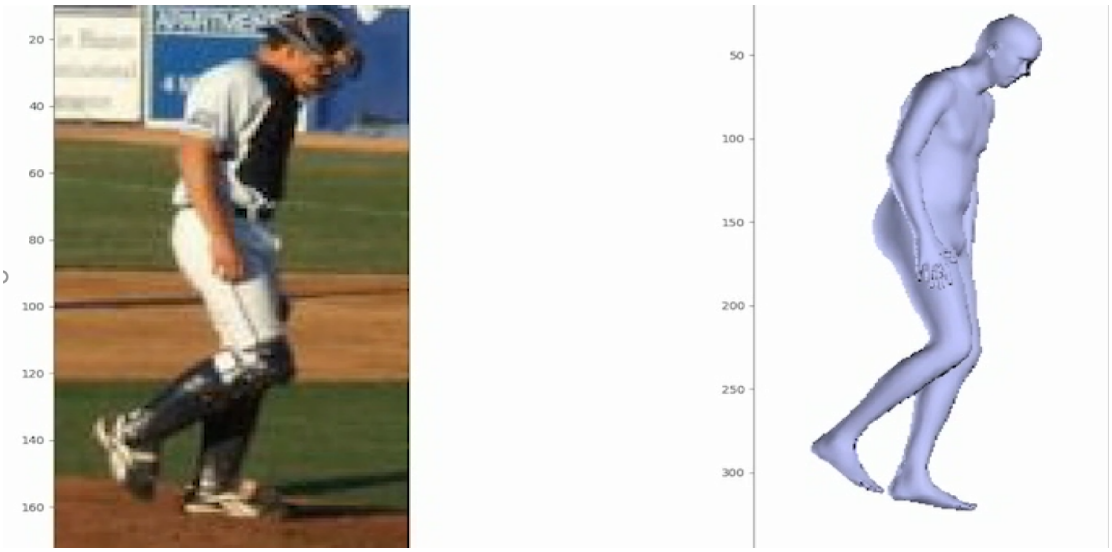


Fig. 2.6 SMPLify: Fitting the SMPL model to the Leeds Sports Dataset.

Direct regression

The most recent, and state-of-the-art approaches employ deep learning techniques to solve the entire optimization problem by directly regressing shape and pose parameters of the template model. An early approach by Tan et al. [**?**] used deep neural networks to learn an encoding $f : \mathbb{R}^{H\times W} \mapsto (\theta, \beta, t, K)$ of input images to SMPL pose and shape, translation and camera parameters. Their method makes use of a *silhouette renderer* $R : (V,T) \mapsto \{0,1\}^{H\times W}$ (learnt using synthetic data) capable of producing a binary silhouette from a predicted SMPL mesh. In this way, R allows the network's SMPL predictions to be supervised to ensure generated silhouettes match ground-truth annotations. Improvements were realised by incoporating an abudance of available human data into the training pipeline. Apart from 3D morphable models (e.g. SMPL [**?**]), methods use paired 3D motion capture data (e.g. Human3.6M [**?** **?**], 3DPW [**?**]) to help relate 2D appearance and the underlying 3D structure, unpaired 3D motion capture data (e.g. CMU [CMU]) for learning detailed priors over the distribution of human shapes and poses, and large 2D keypoint datasets (e.g. MSCOCO [COCO], LSP [LSP], MPI [MPI]) which help promote generalization to 'in-the-wild' scenarios. A notable work in this category is Human Mesh Recovery (HMR) of Kanazawa et al. [**?**], although multiple concurrent works exist[Hsiao-Yu Fish Tung], [pavlakos], [neural body fitting] and [DenseRaC]. **??** of this thesis will explore a method for modelling uncertainty in 3D reconstruction, by exploring extensions to the aforementioned architectures.

BodyNet [bodynet] follow a similar pipeline with the inclusion of a texture prediction module supervised by 2D body part segementations [dataset]. Silhouette data has also been shown to assist in accurate reconstruction of clothes, hair and other appearance detail [**?**] [SiCloPe] [ARCH]. Methods typically represent 3D clothes as 'freeform' vertex deformations; in other words, deformations beyond the standard SMPL blend shapes which represent unclothed bodies. This presents significantly more

degrees of freedom in the optimization, which must be controlled; either by large training datasets with limited variation or mesh-based deformation priors such as ARAP.

While the dominant paradigm in human reconstruction is now end-to-end deep learning methods, SPIN [**?**] show impressive improvement by incorporating an energy minimization process within their training loop to further minimize a 2D reprojection loss subject to fixed pose & shape priors. This idea inspired the work presented in **??** of this thesis, in which a 3D dog shape prior is learnt during the training loop via expectation maximization.

There have also been a few recent works that reconstruct 3D humans without an explicit template prior. The general idea is to interpret a neural network as an implicit representation of the 3D surface. In particular, the network f is trained to map sampled 3D locations $(x, y, z) \in \mathbb{R}^3$ to an occupancy value $\{0, 1\}$ (and optionally a texture value $(r, g, b) \in \mathbb{R}^3$). This results in a memory efficient representation of the 3D surface as the space used to embed the surface does not need to be explicitly stored. Saito et al. [**?** **?**] present an example of such an architecture. Their network maps 3D locations with ResNet [Resnet] features sampled at coordinates projected with a known weak-perpective camera to occupancy and texture values. Li et al.[MonoPort] later sped up reconstruction and rendering to allow real-time inference. Neural Radiance Fields (NeRF) capture high resolution details of static scenes by mapping 3D coordinates (x, y, z) and viewing directions (θ, ϕ) to volume density and view-dependent emitted radiances. The work was adapted to articulated structures [ANerf] by learning new skin for the SMPL model, using SPIN [**?**] as a basic 3D skeleton predictor. A current downside of these techniques is the need to train a separate model per scene/subject and the slow inference time. However, these works do demonstrate high quality results and are a worthy direction for future research.

2.4.3 Model-based animals

Having discussed reconstruction methods for other articulated subjects, this section will move on work related to the main topic of this thesis: 3D reconstruction of animal subjects. As summarized above, there are multiple challenges associated with animals which are not present with humans. Firstly, the general class of animal subjects is significantly more diverse in appearance and structure than the human category. This holds (although to a lesser extent) with various important animal subcategorizations, such as the class of medium-to-large quadrupeds, dog breeds, or birds. This diversity leads to challenges when designing 3D morphable models, since extreme deformations must allowed while penalizing even subtle adapations deemed unnatural. Also of concern are the significant variations in animal motion patterns (causing complex self-occlusion), body textures including fur (which vary even within breeds) and difficult environmental conditions. An interesting advantageous aspect to reconstructing animals is that they are less frequently clothed; a common nusiance factor for human reconstruction. These factors combined with practical challenges associated with capturing and annotating images has also led to a lack of open-source datasets which depict animal subjects. Of particular concern to this thesis is the lack of 3D training data captured from real animal subjects

Dataset	Modality	Annotation	Subject	Num. Examples
MSCOCO [**?**]	Image (RGB)	S2, BB2	DCaHoZSECoBG	23,467
PascalVOC [**?**]	Image (RGB)	S2, BB2	xxx	xxx
DAVIS [**?**]	Video (RGB)	S2	xxx	xxx
StanfordDogs	Image (RGB)	BB2	D (120 breeds)	20,580
StanfordExtra	Image (RGB)	J2, S2	D (120 breeds)	12,000
BADJA	Video (RGB)	J2, S2	xxx	xxx
AnimalPose	Image (RGB)	J2	xxx	xxx
SMAL	N/A	MM3	DCaHoZCoTLHi	47

Table 2.1 Dataset summary: analysis of available datasets. Bold rows are datasets introduced in this thesis. S2: 2D Silhouettes, BB2: 2D bounding boxes, J2: 2D Joints, MM3: 3D Morphable Model. P3: 3D Priors. M3: 3D Model. D: dog, Ca: cat, Ho: horse, Z: zebra, S: sheep, E: elephant, Co: cow, B: bear, G: giraffe, T: tiger, L: lion: Ch: cheetah, Hi: hippo

(typically using motion capture or static scans). Equivalent resources made available for humans have been used as a basis for building 3D morphable models (e.g. SMPL [**?**], FAUST [FAUST]), for learning priors over human shape and pose [learning priors], and to provide per-image 3D supervisation when training deep neural networks [**?**]. Thankfully, online animal imagery which could serve as the basis for future datasets is plentiful, as are datasets with 2D silhouette data [**? ? ?**]. A formal comparison is given in **??**. While animals are often featured in computer vision literature, there are still relatively few works that focus on accurate 3D animal reconstruction. However, the creative approaches used in these methods makes a formal review worthwhile. A summary of recent approaches is tabulated in **??**.

Learning animal shape from 2D image collections

Early work in animals focuses on learning 3D animal shape spaces by registering an input 3D model to image collections. Examples include Chen et al.[chensharks] who learn a shark model and Cashman et al. [**?**] who recover a parameterized, morphable 3D model from unrelated 2D images depicting examples of the target class. Their method requires user-supplied 2D object outlines and point constraints for each image, and a single rigid mesh for the entire object class. The authors demonstrate recovering an 8-parameter morphable dolphin model from 32 images sourced from the Internet. To reduce required user activity, it is reasonable to assume that given sufficient labelled training data, it would be simple to manipulate a convolutional network architecture able to perform foreground / background segmentation and identify key points (say, joints) for the desired object class. The system achieves impressive results when optimizing over both pose and shape parameters across a range of object classes, but suffers from an overly smooth shape representation which causes trouble for strongly articulated classes such as polar bears. Related is the work of Ntouskos et al.[ntouskos] who use an optimization scheme to combine geometric primitives fit to segmented animal parts.

Paper	Animal Class	Training requirements	Template Model	Video required	Test Time Annotation	Model Fitting	Test Size
This paper	Dogs	J2, S2, T3, P3	SMAL	No	None	No	1703
3D-Safari [**?**]	Zebras, horses	M3 (albeit synthetic), J2, S2, P3	SMAL	3-7 frames / animal	None	Yes	200
Lions, Tigers and Bears (SMALR) [**?**]	MLQ	Not trained	SMAL	3-7 frames / animal	J2, S2	Yes	14
3D Menagerie (SMAL) [**?**]	MLQ	Not trained	SMAL	No	J2, S2	Yes	48
Creatures Great and SMAL [**?**]	MLQ	Not trained	SMAL	Yes	S2 (for best results shown)	Yes	9
Category Specific Mesh Reconstructions [**?**]	Birds	J2, S2	Bird convex hull	No	None	No	2850
What Shape are Dolphins [**?**]	Dolphins, Pigeons	Not trained	Dolphin Template	25 frames / category	J2, S2	Yes	25
Animated 3D Creatures [**?**]	MLQ	Not trained	Generalized Cylinders	Yes	J2, S2	Yes	15

Table 2.2 Literature summary: Our paper extends large-scale "in-the-wild" reconstruction to the difficult class of diverse breeds of dogs. MLQ: Medium-to-large quadrupeds. J2: 2D Joints. S2: 2D Silhouettes. T3: 3D Template. P3: 3D Priors. M3: 3D Model.

Reconstructing animals with unknown skeleton

The approach was later extended to articulated classes in work by Stebbing et al. [**?**], who optimize a 3D template model to animal video sequences. In this work, rather than defining an internal skeleton, the template model is rigged with virtual markers, which assign each mesh vertex v_i to one of M groups that share a set of basis rotations B_m. User interaction is again reuqired to segement the animal from the background and to provide mesh-to-image keypoint correspondences.

Through reasonably accurate pose fitting and by allowing some pose-invariant shape deformation, this work produces smooth meshes which match the input video. Moreover, experimentation demonstrates that ARAP is a useful prior for reconstructing articulated, non-rigid motion in instances that an internal skeleton is a priori unknown. However, the shape attributes for the reconstructed model are not particularly accurate, which results in frequent errors appearing at internal occluding contours. In addition, the large non-convex optimization algorithm is an expensive operation, taking around 1 minute per video frame on a standard Linux workstation. Results are shown on 11 sequences.

Results showing this work fitting a crude dog template mesh to a sample video obtained from YouTube are shown previously in Figure **??**. Figure **??** shows another example, which operates on a template impala mesh. Similar is the technique of Favreau et al. [favreau], who fitting geometric primitives to a 2D video sequence with manually sketched body part labels.

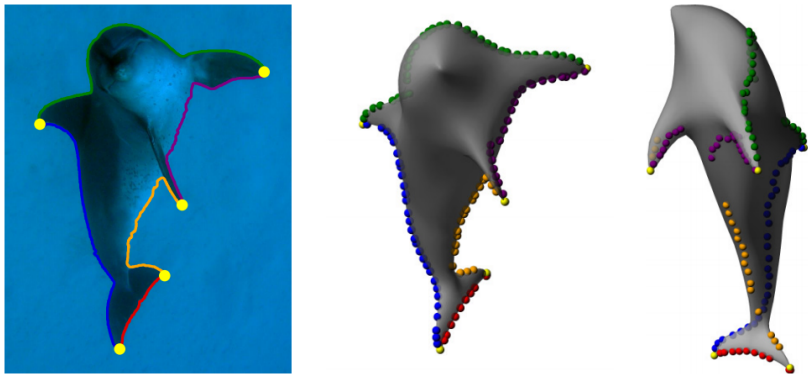
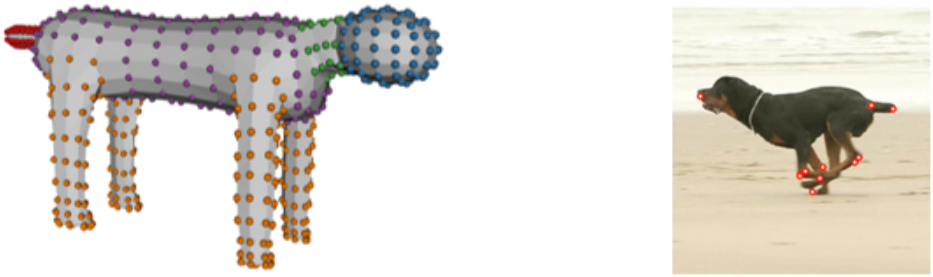


Fig. 2.7 8-parameter dolphin model with annotated contour (left) and contour generators (middle and right).



(a) Template mesh with joint movement constraints.

(b) Example of user supplied point tracks.

Fig. 2.8 User input required for the deformable mesh animation algorithm, reprinted from [**?**].

Fitting a morphable animal mesh to images

Unfortunately, none of the techniques above are suitable for reconstructing an animal category with significant shape diversity. Zuffi et al. [**?**] made a significant contribution by releasing SMAL, a deformable 3D quadruped model (analagous to SMPL [**?**] for human reconstruction) build from 41 scans of artist-designed toy figurines. The authors also released shape and pose priors generated from artist data. The authors also discuss a small modification to the SMPLify [smplify] approach in order to fit the SMAL model to RGB animal input images. However, an example result showing the result of the optimizer fitting the SMAL mesh to an RGB image of a fox can be seen in Figure **??**. The whole optimization process takes around 1 minute per frame.

The terms of the optimization largely mirror the work of SMALify, although the interpenetration term is ommited and joints are provided manually rather than by a CNN. Importantly, the optimization incoporates an additional *silhouette* term which aligns the 3D model shape to a binary silhouette image extracted by a user. **??** of this thesis shows silhouette terms are of particular importance when reconstructing accurate animal shape. The following definition describes the process for differentiably rendering a 3D model to form a silhouette image which can then be compared to the provided 2D silhouette:

Fig. 2.9 Example of an dog template being fit to input video sequence, reprinted from [**?**]

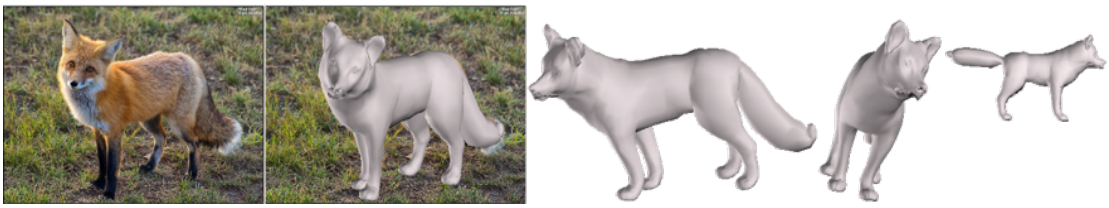


Fig. 2.10 Fitting SMAL to a hand segmented animal, reprinted from [**?**].

Definition 2.4.2 (Differentiable Rendering)**.** The process of generating a 2D image from a 3D polygon mesh is known as rendering and can be achieved through a process known as raytracing. Raytracing is a rendering technique able to generate photorealistic 2D images from the scene. It can be considered the opposite process by which the human eye perceives the world, as this method involves lines being cast outwards, beginning at a point known as the *camera origin*. Figure **??** shows a typical set up, in which rays are cast from the camera origin through each pixel on the image plane. The colour for the pixel is obtained by following the ray through the scene until a light source or non-reflective surface is reached, taking into account any reflections or non-opaque scene items. Due to the considerable comptuation required, the operation is often parallelized and assigned to the GPU. However, the technique is typically considered unsuitable for real-time rendering of complex scenes (due to complex ray paths) or when high resolution images (many rays required) are needed. However, for this work, scenes are typically made up of a single non-reflective, solid mesh surface and contain no complex elements (e.g. shadows, non-constant lighting.

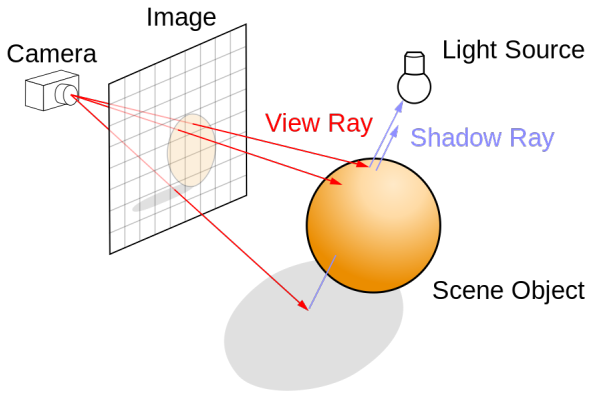


Fig. 2.11 Diagram showing raycast rendering. [**?**].

It is also worth noting that the standard method for raycasting is not differentiable, causing problems for differentiable optimizers (including neural networks). However, alternative rendering methods [**?**] are available for these purposes.

The SMAL authors [**?**] demonstrate their system by fitting their deformable 3D model to quadruped species. Subsequent work includes SMALR [**?**], who fit to multiple images of an animal and incoporate similar freeform deformation term used to model clothing in 3D human pipelines. They show their technique, which takes advantage of multi-view constraints afforded by fitting to video sequences, allows high-quality reconstructions of more exotic animal categories with details not covered in the original SMAL 3D model training set.

A common property of techniques discussed so far is the reliance on manual annotations (commonly 2D silhouettes and keypoint correspondences) at test time. 3D-Safari [**?**] overcome this by training a deep network on synthetic data (generated by applying SMALR [**?**] on a large input dataset) to recover detailed zebra shapes in the wild. **??** of this thesis demonstrates a technique which overcomes the need for a large input dataset of joint annotations by training on synthetic data generated from the graphics model alone.

A further drawback of the approaches is their reliance on a test-time energy-based optimization procedure, which is susceptible to failure with poor quality keypoint/silhouette predictions and increases the computational burden. **??** of this method presents an automatic reconstruction method that overcomes the need for additional energy-based refinement, and is trained purely from single in-the-wild images.

Tangential to these approaches is the work of Kulkarni et al.[Canonical Surface Mapping] who reconstruct pose (although not shape) parameters for an rigged mesh using a deep network, learning from geometry cycle consistency terms rather than annotated keypoints. However, the shape is fixed to the input mesh and pose is also often confused, particularly determining the difference between the legs. The results appear of significantly poorer quality than the results in **??** and **??** of this thesis. Also of consideration is the end-to-end network of Kanazawa et al. [**?**] and subsequent paper which overcomes the need for keypoints by capturing uncertainty in camera viewpoints. In addition, their

network does not require an input 3D morphable model, although they do initialize a sphere model to a mean bird. In addition, the bird category exhibits more limited articulation than our dog category.

Chapter 3

Learning from Synthetic Data; Bridging the Domain Gap

3.1 Introduction

This chapter introduces a system for recovering the 3D shape and motion of a wide variety of quadrupedial animals from video. Inspired by techniques from the recent human body and hand tracking literature (most notably SMPLify [**?**]), the system comprises a machine learning front-end for predicting 2D joint positions followed by an energy minimization stage which fits a detailed 3D model to the image. However, this chapter will discuss a number of key challenges which arise when reconstructing animal subjects compared with humans and methods to overcome them.

3.1.1 Limited animal training data

For human tracking, hand labelled sequences of 2D segmentations and joint positions have been collected from a wide variety of sources [**?** **?** **?**]. Of these two classes of labelling, animal *segmentation* data is available in datasets such as MSCOCO [**?**], PASCAL VOC [**?**] and DAVIS [**?**]. 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. In the case of SMPLify, the network is trained on large-scale human keypoint datasets, including MPII [**?**] (approx. 40,000 people) and the Leeds Sport Dataset [**?**]. Unfortunately, there is no keypoint dataset for animals that cover even a small fraction of the quadruped types that we aim to reconstruct.

Of greater 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

Paper	Release Year	Animal Class	Training requirements	Template Model	Video required	Test Time Annotation	Model Fitting	Test Size
What Shape are Dolphins [**?**]	2013	Dolphins, Pigeons	Not trained	Dolphin Template	25 frames / category	J2, S2	Yes	25
Animated 3D Creatures [**?**]	2016	MLQ	Not trained	Generalized Cylinders	Yes	J2, S2	Yes	15
3D Menagerie (SMAL) [**?**]	2017	MLQ	Not trained	SMAL	No	J2, S2	Yes	48
Category Specific Mesh Reconstructions [**?**]	2018	Birds	J2, S2	Bird convex hull	No	None	No	2850
Lions, Tigers and Bears (SMALR) [**?**]	2018	MLQ	Not trained	SMAL	3-7 frames / animal	J2, S2	Yes	14
This paper	2018	MLQ	Not trained	SMAL	Yes	S2 (for best results shown)	Yes	9
Who Left the Dogs Out?	2018	Dogs	J2, S2, T3, P3	SMAL	No	None	No	1703
3D-Safari [**?**]	2019	Zebras, horses	M3 (albeit synthetic), J2, S2, P3	SMAL	3-7 frames / animal	None	Yes	200
U-CMR [**?**]	2020	Birds	M3 (albeit synthetic), J2, S2, P3	Bird convex hull	3-7 frames / animal	None	Yes	200

Table 3.1 Literature summary: Analysis of required input annotations. MLQ: Medium-to-large quadrupeds. J2: 2D Joints. S2: 2D Silhouettes. T3: 3D Template. P3: 3D Priors. M3: 3D Model.

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 [**?**], it remains impractical to consider mocap for a family of tigers at play.

These dataset challenges have, at least in part, resulted in the lack of *automatic* approaches for 3D animal reconstruction. Precisely, prior to the design of the system discussed in this chapter, 3D animal reconstruction techniques relied on at least one of manually provided silhouettes, keypoint locations or limb tracks. The techniques are summarized in **??** and discussed in depth in **??**.

3.1.2 Synthetic image generation

In the absence of real-world motion capture data, the recent publication of the Skinned Multi-Animal Linear (SMAL) model [**?**] offers a method for *synthetic* data generation. Analogous in design to the popular SMPL [**?**] parametric human model, SMAL can generate a wide range of quadruped species. As discussed in **??**, SMAL overcomes the lack of 3D animal training data by instead learning from 41 scans of toy figurines. This results in SMAL being of considerably lower fidelity than the SMPL, which was constructed from 1786 3D human scans. However, the model's PCA shape space offers a mechanism for sampling potentially infinite quadruped animals of satisfactory realism which can form a valuable training dataset.

The use of synthetic images to train 3D reconstruction systems has been attempted in recent computer vision literature. The general approach is to capture training images by capturing multiple images of a representative 3D model (or scene) by randomly sampling camera and environmental parameters. A machine learning model is then trained on this dataset D_{synth} of synthetic images and later tested on a dataset D_{real} of real-world examples. This procedure offers an important practical advantage; the training images can be automatically generated with correponding annotations. Depending on the task, this can overcome a lengthy and complex process of sourcing manual annotations from users. However, designing a data pipeline which can generate synthetic images with sufficient realism to be representative of real test images D_{real} is challenging. A common pitfall of such approaches is the tendancy to train machine learning models that perform well on unseen synthetic images but poorly on real world examples. This phenomenon is often caused by systematic differences between the D_{synth} and D_{real}, resulting in the model becoming overreliant on features present only in D_{synth} or unfamiliar with features present only in D_{real}. The differences between datasets is often described as the *domain gap* which must be bridged in order to achieve a successful predictor.

Fortunately, there are a number of options for tackling this problem. A popular strategy is to ensure the generation pipeline is of extremely high quality. SURREAL[SURREAL] is a method which learns from synthetic human images rendered using the popular SMPL [**?**] parametric model. In this work, the authors go to significant effort to source realistic UV texture maps for SMPL and apply a high quality rendering engine complete with a lighting and reflectance model to generate their training images. A similar approach is followed by SMALST [SMALST], who follow a multi-view optimization pipeline to construct a synthetic zebra training dataset complete with realistic textures. Related also are autonomous driving systems that train (or pre-train) on synthetic 'virtual worlds' [Virtual Worlds]. An alternative option is to process the test image dataset to find a representation which is easier to synthesise. For 3D human skeleton prediction, Shotton et al. [**?**] synthetise a set of depth images used to train a keypoint regressor. An unfortunate byproduct of this training style however is that it necessitates a depth sensor at test time.

A significant advance has been made recently with the introduction of generative adversarial networks (GANs). The idea of a GAN is to train two 'competing' networks: the *generator* is tasked with producing synthetic images and the *discriminator* determines if a given input image originated in a real-world dataset or was instead synthesized by the generator. These two networks are given opposite goals: the generator is penalized if it fails to 'fool' the discriminator, and the discriminator is penalized if it fails to recognize a synthesized image. By training these networks jointly, the generator eventually learns to produce synthetic images of extremely high quality, since the discriminator provides feedback on even subtle aspects of the synthetic images which are unreaslistic. Despite the remarkable quality of 'GANerated' images, the challenge of conditioning GANs to generate *structured* images is still an evolving field. For example, a sensible challenge might be to train a GAN to generate UV texture maps for the SMAL model.

of this is that there should be an analoug

The system presented in this section adopts an alternative (and much simpler) approach for synthetic image generation. The idea builds on the realisation that a 2D silhouette image encodes a significant amount of the object's shape features, and by virtue of being textureless and having no background detail is much simpler to synthesise. As previously discussed, there are also a great number of 'off-the-shelf' image segmentation engines which can produce accurate animal silhouettes of a high quality.

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.

3.1.3 Handling silhouette ambiguity

An unfortunate drawback of using silhouette images (for example, when compared to RGB or depth counterparts) is that missing interior contours leads to a source of reconstruction ambiguity. An example of this is shown in Figure 6 in which two distinct 3D reconstructions are shown to yield similar 2D silhouette reprojections. Clearly, a naive joint predictor which regreses a single set of 2D coordinates from an input silhouette would perform suboptimally, since the silhouette alone contains insufficient information to resolve such cases.

However, the fact that the system is designed to accept a video sequence as input (rather than a single frame) provides additional information which can be used to resolve per-frame ambiguities. This section describes a method for incoporating temporal knowledge in order to obtain a reliable set of predicted 2D joint coordinates that later form the basis for the 3D model fitting stage. The key insight is to modify a standard joint heatmap predictor [hourglassnet] in order to encourage multi-modal outputs, both when trained on synthetic data and later tested on real world frames. A novel discrete optimization stage is then used to recover the skeleton trajectory of maximal likelihood from the sequence of multi-modal predictions. This section will explore two methods for achieving this: the first frames the problem as a quadratic program and the later provides achieves a significant efficiency improvement by using a genetic algorithm.

3.1.4 Contributions

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. The system exhibits the following contributions:

- A machine-learned mapping from silhouette data of a large class of quadrupeds to generic 2D joint positions.

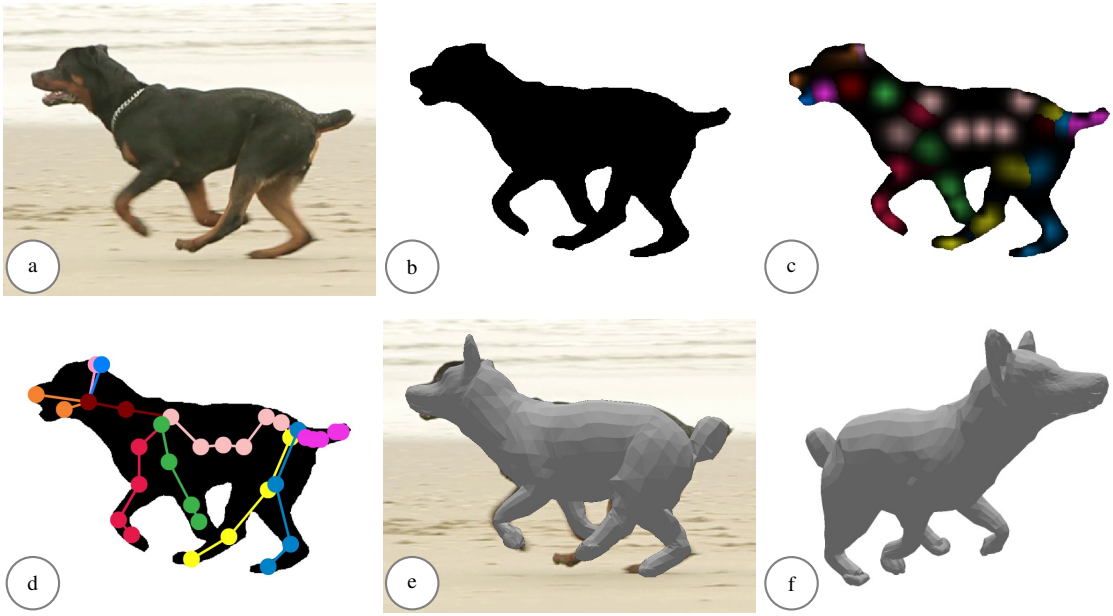


Fig. 3.1 **System overview**: input video (a) is automatically processed using DeepLabv3+ [**?**] to produce silhouettes (b), from which 2D joint predictions are regressed in the form of heatmaps (c). Optimal joint assignment (OJA) finds kinematically coherent 2D-to-3D correspondences (d), which initialize a 3D shape model, optimized to match the silhouette (e). Alternative view shown in (f).

- A novel optimal joint assigment (OJA) algorithm extending the bipartite matching of Cao *et al.* [**?**] 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 [**? ?**] 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. **??**. The remainder of the paper provides 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. Experiments are also conducted on synthetic animal videos to produce 3D joint accuracy statistics and full mesh comparisons. A qualitative comparison is given to recent work [**?**] on the related single-frame 3D shape and pose recovery problem. The paper concludes with an assessment of strengths and limitations of the work.

3.2 Preliminaries

3.2.1 Deformable 3D quadruped model

This section provides a formal definition for the deformable 3D model that is used to generate synthetic training data and in the model fitting stage to obtain the final mesh. Our system assumes a deformable 3D model such as SMAL [**?**] which parametrizes a 3D mesh as a function of *pose* parameters $\theta \in \mathbb{R}^P$ (e.g. joint angles) and *shape* parameters $\beta \in \mathbb{R}^B$. As discussed in **??**, a 3D mesh is an array of vertices $v \in \mathbb{R}^{3\times V}$ (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 SMAL may be viewed as supplying a set of triangles, and a function

$$v(\theta, \beta) : \mathbb{R}^P \times \mathbb{R}^B \mapsto \mathbb{R}^{3\times V} \tag{3.1}$$

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 a mesh will 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 in axis angle form. Application of such a transformation to a $3 \times V$ matrix will be denoted by $*$, so that

$$\phi * v(\theta, \beta) \tag{3.2}$$

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

It is also necessary to define a model's *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. Note that even in the case of rigged models, this provides a mechanism to add additional joints beyond the ones required to drive model deformation. In any case, joints are 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 v may include vertices not mentioned in the triangulation).

3.2.2 Camera model, joint reprojection and silhouette rendering

For both synthetic image generation and the later model fitting stage, it is necessary to be able to *render* the 3D model. 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

$$\kappa(\phi, \theta, \beta) := \pi(\phi * v(\theta, \beta)\mathtt{K}) \tag{3.3}$$

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,

$$R\big(\phi * v(\theta, \beta)\big) \in \mathbb{B}^{W \times H} \tag{3.4}$$

for an image resolution of $W \times H$. In order to allow derivatives to be propagated through R (essential for the silhouette term in the model fitting stage), the *differentiable renderer* of Loper et al. [**?**] is used. Please see the relevant section in **??** for further details.

3.2.3 System overview

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 input image the shape, pose, and position parameters describing the animal's motion. Inspired by recent work in human 3D reconstruction, this objective can be broken down into a multiple stage pipeline.

The core components are as follows:

1. The discriminative front-end extracts silhouettes from video, and then uses the silhouettes to predict multi-modal heatmaps, from which 2D joint positions are obtained 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.

3. Generative deformable 3D model is fitted to the silhouettes and joint candidates as an energy minimization process.

These stages are described in detail over the following three sections.

3.3 Predicting 2D joint candidate locations

The goal of the first stage is to take, for each video frame, an image reperesenting the animal and to output a $W \times H \times J$ tensor of heatmaps. To achieve this, we train the stacked hourlgass network [**?**] of Newell et al. to a large dataset of synthetically generated quadruped images.

3.3.1 Generating synthetic quadruped images

In order to render a synthetic quadruped image, a set of pose θ, shape β and position ϕ parameters are required. With these parameters, a (textureless) training image can be generated by applying **??** and corresponding ground truth 2D joints locations can be generated via **??**. What remains is to ensure a model trained on the synthetic images generalizes to the real-world test images. To achieve this, it is important that the training dataset captures the modes of variation by appropriately sampling the model parameters.

Sampling shape and pose parameters

Given that the real-world test images exhibit multiple quadruped species, a primary mode of variation is the animal's *shape* characterisitics. Naively, generating a dataset large enough to capture possible test animal shapes would be a challenging task, since it would require access to multiple real or artist-generated 3D scans. Fortunately, the SMAL model defines a linear shape space which allows repeated sampling of different (and realistic) quadruped shapes. Of course, having been built from only toy figurines, the variation is still somewhat limited, but as the later experimental section will show, the model is fit for purpose. Synthetic shape parameters β are obtained by sampling from a Gaussian prior:

$$X = Y \tag{3.5}$$

Another mode of variation in test images is the various animal limb positions. These can again be synthetised by sampling from a gaussian prior constructed from a dataset of animal poses. Since no such dataset exists for real animals, a prior is instead built from a small set of artist-generated poses, originally provided by the SMAL model authors. The sampling strategy is therefore given by:

$$X = y \tag{3.6}$$

Sampling position (camera) parameters

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.

What about textures and backgrounds?

At this point, the synthetic test images have plausible shape, pose and positional parameters but are lacking in realistic texture. Furthermore, it is not clear how to render plausible background scenes, nor how to convincingly place the 3D animal in such a scene. This challenge allows for a primary contribution of this work: using the silhouette domain as a representation much easier to synthetise (since surface texture maps and background textures are lost) and can be obtained from real-world test images. Another important characteristic of the mapping between real-world images and silhouette counterparts is that the shape and pose information is mostly preserved, apart from a few ambiguities (e.g. limb ordering due to lacking interior contours).

To summarise, 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. See an example in To generate each image, a random shape vector β, pose parameters θ and camera position ϕ are drawn, and used to render a silhouette $R\big(\phi * v(\theta, \beta)\big)$ and 2D joint locations $\kappa(\phi, \theta, \beta)$.

3.3.2 Prediction of 2D joint locations using multimodal heatmaps

A pose estimation network can now be trained on a dataset of binary silhouette images S with corresponding 2D joint locations $\kappa(\phi, \theta, \beta)$. The core network architecture used for this task is the stacked hourglass network [**?**] of Newell et al., with an adaptation to produce multi-modal outputs. The stacked hourglass network is now briefly described:

Stacked hourglass network

A stacked hourglass network is a convolutional neural network specifically designed for the task of pose estimation. Hourglass allows inference to take place across multiple scales; an important advantage allowing the network to reason about global information (such as the full body) and local information (such as face features). This is achieved using a repeated bottom-up, top-down modules which each produce a $W \times H \times J$ heatmap tensor. Supervision is applied to each of these heatmaps, which are then passed to ths subsequent block, allowing high level features to be reevaluated for higher order spatial relationships (generally only present at low resolutions). The network’s architecture can be seen in detail in Figure XXX. Hourglass uses a mean squared error loss against a ground truth heatmap tensor, which is applied to the network’s final output and all intermediatary hetamaps. Therefore, ground truth joint coordinates $\kappa(\phi, \theta, \beta)$ must first be encoded into a $W \times H \times J$ tensor of heatmaps. The original authors find that the network has better convergence properties if the ground truth heatmaps are blurred slightly, since a gradient is then provided to predicted “near misses”. This is achieved by blurring ground truth heatmaps with a Gaussian kernel of radius σ. The final 2D joint positions are then obtained using non-maximum suppression on the output heatmaps

$$A = B \tag{3.7}$$

Adaptations for training on synthetic data

This setup is mostly suitable for training on synthetic quadruped data, subject to a couple of adaptations. Firstly, the silhouette training images differ considerably in appearance to the RGB images used by the original Hourglass authors. A common difficulty for training pose estimation networks on full RGB images is in trying to distinguish between objects and the background class. With binary silhouette images, this distinction is made trivial as background and foreground are assigned values 0 or 1 respectively. Unfortunately, this causes a problem when naively training HourglassNet on the generated synthetic images as the network can greatly minimize the loss by simply predicting 'background' for every image pixel (since background pixels tend to outnumber foreground pixels). The fact that 'background' is a much simpler class to predict than the other joint classes causes a training instability which is challenging to overcome by adjusting the learning rate. Instead, the loss function is replaced with a *weighted* version of the mean squared error loss. This modification encourages the network to devote it's attention to all classes equally. The precise formulation for this is given as follows:

$$A = B \tag{3.8}$$

Adapting stacked hourglass network for multi-output learning

As detailed in the later experimental section, the network trained using this process generalizes well from synthetic to real images due to the use of the silhouette, and produces accurate predictions for most joints. However, the predictor performs poorly on some joints due to ambiguities which result from the lack of interior contours in the silhouette input data. These missing details cause often result in confusion between joint "aliases": left and right or front and back legs. When these predictions are wrong and are represented by high confidence heatmap regions, little probability mass is assigned to the area around the correct leg, meaning no available proposal is present after non-maximal suppression.

This is handled with a further technical contribution; handling the prediction uncertainty by adapting the stacked hourglass network to produce multiple outputs. This is achieved by explicitly training the network to assign some probability mass to the "aliased" joints. For each joint, a list of potential aliases are defined as weights $\lambda_{j,j'}$ and linearly blend the unimodal heatmaps G to give the final training heatmap H:

$$H_j(p) = \sum_{j'} \lambda_{j,j'} G(p; \kappa_{j'}, \sigma) \tag{3.9}$$

Fig. 3.2 Example predictions from a network trained on unimodal (top) and multi-modal (bottom) ground-truth for front-left leg joints.

Fig. 3.3 Example outputs from the joint prediction network, with maximum likelihood predictions linked into skeleton.

For non-aliased joints j (all but the legs), $\lambda_{j,j} = 1$ and $\lambda_{j,j'} = 0$, yielding the unimodal maps. For aliased joints, the joint is assigned a weight $\lambda_{j,j} = 0.75$ and aliases are assigned a weight $\lambda_{j,j'} = 0.25$. This ratio is shown to ensure opposite legs have sufficient 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. **??**. Note that the construction of at most bi-modal ground truth heatmaps sets a practical constraint on the number of output modes. In other words, the loss is minimized if the network produces one output mode for non-aliased joints and two output modes for aliased joints.

3.4 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. The set of joint proposals is represented as $X = \{x_{jp}\}$, where x_{jp} indicates the 2D position of proposal $p \in \{1,...,N_j\}$ associated with joint $j \in J$. Before applying the optimizer, a subset of proposals $X^* \subseteq X$ should be selected in order to form a complete skeleton, i.e. precisely *one proposal is selected for every joint*. This section will 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 $\{x_{jp}\}$, a binary indicator vector $\bar{a}_j = \{a_{jp}\} \in \{0,1\}^{N_j+1}$ is introduced, where $a_{jp} = 1$ indicates that the p^{th} proposal

Joint j	Proposal p	x_{jp}	a_{jp}
Nose	0	$(0,2)$	1
	1	$(4,2)$	0
	NULL	—	0
Upper Leg	NULL	—	1
Paw	0	$(4,2)$	0
	1	$(8,10)$	1
	NULL	—	0
Tail	0	$(4,2)$	0
	1	$(8,10)$	0
	2	$(4,2)$	1
	NULL	—	0

Table 3.2 Example inputs and output to the joint assignment problem. Non maximum suppression applied to predicted heatmap tensors yields a set of proposals p for each each skeleton joint j at 2D location x_{jp}. a_{jp} is an illustrative assignment vector which is predicted by the OJA algorithm.

Joint j	Proposal p			
Nose	1	0	0	
Upper Leg	1			
Paw	0	1	0	
Tail	0	0	1	0

Table 3.3 Assignment variables $A = \bar{a}_j = \{a_{jp}\} \in \{0,1\}^{N_j+1}$ for the current frame stored as a jagged array.

Joint j	$X^* = X(A)$
Nose	$(0,2)$
Upper Leg	—
Paw	$(8,10)$
Tail	$(4,2)$

Table 3.4 2D joint locations $X^* = X(A)$ selected by the assignment variable A

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 $X^* = X(A)$ denote the subset of points selected by the binary array A.

Optimal assignment minimizes the function

$$L(A) = L_{\text{conf}}(A) + L_{\text{null}}(A) + L_{\text{prior}}(A) + L_{\text{temp}}(A) + L_{\text{cov-sil}}(A) + L_{\text{cov-bone}}(A) \tag{3.10}$$

which balances agreement of the joint configuration with the network-supplied *confidences*, a learned *prior*, *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 better results are obtained by including 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.

3.4.1 Basic formulation

Network confidences: $L_{\text{conf}}(A)$

The first energy term $L_{\text{conf}}(A)$ comes from the output of the joint prediction network, which provides a confidence score y_{jp} associated with each joint proposal x_{jp}. Then $L_{\text{conf}}(A) = \sum_j \sum_p -\lambda_{\text{conf}} \log(y_{jp}) A_{jp}$ 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. Note that if only this term is included, the OJA would simply produce the result of the standard non-maximal suppresion algorithm, selecting the heatmap location with highest network confidence. Note that this function can be rewritten as

$$L_{\text{conf}}(A) = \lambda_{\text{conf}} \log(y^T) \text{vec}(A) \tag{3.11}$$

Null proposals: $L_{\textbf{null}}(A)$

Under some circumstances, for example when a body part is heavily occluded or ambiguous, *all* available proposals for a given joint may be of poor quality. In such a case, including any of these options in a skeleton configuration may have a detrimental impact on the later model fitting stage. Under these circumstances, it may be preferable to exclude the joint in question from the optimization entirely. 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. Precisely, a jagged array D is defined

$$D_{jp} = \begin{cases} \lambda_{\text{null}} & \text{if } p \text{ is null proposal} \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

and resulting energy becomes

$$L_{\text{null}}(A) = \text{vec}(D)\text{vec}(A) \tag{3.13}$$

Skeleton Prior: $L_{\textbf{prior}}(A)$

The next energy term is used to discourage anatomically implausible skeletal configurations from being selected. The prior probability of a skeletal assignment A is represented as a multivariate Gaussian distribution over the selected joint positions $X^* = X(A)$

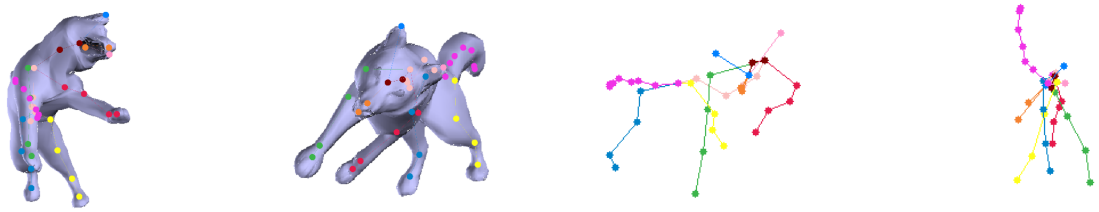
$$P_{\text{prior}}(A) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp\left(-\frac{1}{2}(x^* - \mu)^T \Sigma^{-1} (x^* - \mu)\right) \tag{3.14}$$

The mean $\mu \in \mathbb{R}^{2J}$ and covariance $\Sigma \in \mathbb{R}^{2J \times 2J}$ terms are obtained from synthetic training examples generated earlier. The prior is shown in **??**. The objective of the OJA is to find the assignment vector A which maximizes this prior, which is equivalent to minimizing the negative log prior

$$L_{\text{prior}}(A) = \frac{k}{2}\log(2\pi) + \frac{1}{2}|\Sigma| + \frac{1}{2}(x^* - \mu)^T \Sigma^{-1} (x^* - \mu) \tag{3.15}$$

This formulation is reducible to a minimization over the Mahalanobis distance, which is given by the summation

$$L_{\text{prior}}(A) = \sum_{j}^{J} \sum_{p}^{N_j} \sum_{k}^{J} \sum_{q}^{N_k} a_{jp} a_{kq} (x_{jp} - \mu_j) \Sigma_{jk}^{-1} (x_{kq} - \mu_k) \tag{3.16}$$



(a) (b) (c) (d)

Fig. 3.4 Skeleton Prior: Synthetic quadruped training data examples (rendered with texture to show 3D) generated by sampling pose, shape and position parameters and applying to SMAL model (a), (b). The set of 2D skeletal positions are used to create a vector of means $\mu \in \mathbb{R}^{2J}$ and covariance matrix $\Sigma \in \mathbb{R}^{2J \times 2J}$. The Gaussian distribution constructed can be sampled to create new skeletons, such as those shown in (c), (d).

Notice 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. Precisely, the elements of the matrix Q are precisely the Mahalanobis distance between each individual pair of joint proposals

$$[Q]_{jp,kq} = (x_{jp} - \mu_j)\Sigma^{-1}_{jk}(x_{kq} - \mu_k) \tag{3.17}$$

Null proposals are simply excluded from the sum, equivalent to marginalizing over their position.

Temporal Prior: $L_{\text{temp}}(A)$

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. A prior is imposed 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 $L_{temp} = a^\top T^{(t_0,t_1)} a$ for matrix $T^{(t_0,t_1)}$ given by

$$\left[T^{(t_0,t_1)}\right]_{jp,kq} = \begin{cases} e^{-\alpha|t_1 - t_0 - 1|}||x^{(t_0)}_{jp} - x^{(t_1)}_{kq}||^2 & \text{if } j = k \\ 0 & \text{otherwise} \end{cases} \tag{3.18}$$

3.4.2 QP solution.

A general quadratic program is made up of a quadratic objective function and linear equality and inequality constraints. Thus far, all terms in $L(A)$ are quadratic or linear.

To optimize over a sequence of frames, we construct the block diagonal matrix $\hat{Q}$ whose diagonal elements are the prior matrices $Q^{(t)}$ and off-diagonal elements are the temporal matrices $T^{(t_0,t_1)}$. The confidence term **??** and null penalty term **??** are combined and the vector $\hat{c}$ is obtained by stacking across each frame. The solution vector for the sequence $\hat{a}$ is similarly constructed by stacking the vectorized assignment matrices A across timesteps. The jagged array B is used to formalize the constraint that only one proposal should be selected per joint. Precisely

$$[B]_{jp,kq} = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{otherwise} \end{cases} \tag{3.19}$$

and (similarly to A) is vectorized and stacked across timesteps to form the constraint $\hat{B}\hat{a} = 1$. Finally, the constraint $\hat{a}(1-\hat{a}) = 0$ is applied to ensure binary values of $\hat{a}$. The resulting quadratic program formulation is then given in **??**.

$$\begin{aligned} \underset{\hat{a}}{\text{minimize}} \quad & \hat{a}^T \hat{Q} \hat{a} + \hat{c}^T \hat{a} \\ \text{subject to} \quad & \hat{B}\hat{a} = 1, \\ & \hat{a}(1-\hat{a}) = 0 \end{aligned} \tag{3.20}$$

The quadratic program is specified using the open source CVXPY library [**?**] and solved using the "*Suggest-and-Improve*" framework proposed by Park and Boyd [**?**]. It is initialized by choosing the proposal with the highest confidence for each joint. Appropriate values for the free parameters $\lambda_{\text{conf,temp,null}}$ and α were chosen empirically via grid search.

3.4.3 Incoporating coverage priors

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.

Silhouette coverage: $L_{\text{cov-sil}}$

The silhouette coverage term is designed to penalize 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)[**?**] of the silhouette, $Z^t = \text{MAT}(S^t)$ with a cubed loss strongly penalizing projection outside the silhouette:

$$L_{\text{cov-sil}}(A^t; X^t, Z^t) = \sum_i \min_j \|Z_i^t - \hat{X}_j^t\|^3 \tag{3.21}$$

Fig. 3.5 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).

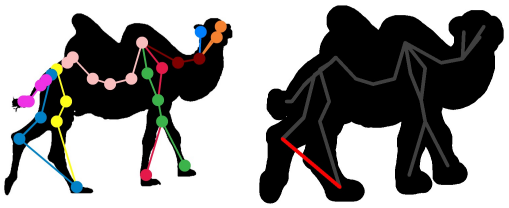


Fig. 3.6 Bone coverage loss. One of the back-right leg joints is incorrectly assigned (left), leading to a large penalty since the lower leg bone crosses outside the dilated silhouette (right).

Bone coverage: $L_{\text{cov-bone}}$

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

$$L_{\text{cov-bone}}(A^t; X^t, S^t, K) = \sum_{\{j,k\} \in K} \left(1 - \min_{\lambda \in [0:0.1:1]} S^t(\hat{X}_j^t + \lambda(\hat{X}_j^t - \hat{X}_k^t)) \right) \tag{3.22}$$

3.4.4 Formulation as a genetic algorithm

We minimize this more complex objective using a genetic algorithm (GA)[**?**]. A genetic algorithm is a method for solving optimization problems using a natural selection process that mimics biological evolution. Unlike the quadratic program described previously, the GA optimization procedure relies on crossover and mutation, rather than by calculating derivatives. In some cases (and as shown in this work), this can lead to much faster convergence properties. The following components are required:

Genes.

Candidate solutions to the optimization problem are referred to as a set of 'genes'. In this case, a gene is the asssignment array A although is represented as a vector of J integers (indicating the proposal ID), rather than one-hot encodings.

Initial population.

Genes are constructed subject to the constraints given in **??**. The genetic algorithm is initialized with a population size of 128 genes. Of these, the first 32 are set equal to the max confidence solutions given by the network, which speeds up convergence. The remaining 96 genes are generated by selecting a random proposal for each joint.

Fitness function.

A fitness function is used to evaluate the quality of a gene. In this setting, the fitness function is precisely the energy $L(A)$ given above which minimizes to yield an optimal skeleton configuration.

Crossover.

Crossover is a fundemental operation common to evolutionary methods which defines the process of combining the information of two 'parent' genes to generate an offspring gene. *Single-point* crossover operates by slicing parent genes each into two parts, and combining first and second parts from different parents to yield the next generation. Following standard practice, the crossover point is randomly selected for each combination.

Mutation.

Analogous to biological mutation and used to maintain diversity among genes, each gene is assigned some probability of undergoing a *mutation*. If a gene is selected for mutation, between 1 and 4 joints have new proposals randomly assigned.

The genetic algorithm described has weights set empirically and is run for 1000 generations. Examples of errors corrected by the additional coverage energy terms are shown in Fig. **??** and Fig. **??**.

3.5 3D model fitting

The model optimization stage refines model parameters to match the silhouette sequence $\mathcal{S}$. The procedure defined here is inspired by the human reconstruction method of SMALify [**?**] (defined in detail in **??**) and the non-automatic quadruped fitting method presented in 3D Menagerie (3DM) [**?**]. The technique presented in this section can be viewed as an extension of these approaches to input video sequences.

A naive 3D model fitting implementation would be to simple apply 3DM independently to each of the N video frame to yield a set of pose parameters $[\theta_t]_{t=1}^N$, shape parameters $[\beta_t]_{t=1}^N$ and position parameters $[\phi_t]_{t=1}^N$. However, fitting to a video sequence rather than single frames offers additional opportunities to constrain the challenging monocular 3D reconstruction task. For example, video sequences typically offer multiple views of the animal subject, although the animal's limb positions often change between frames. However, the animal's *shape* characteristics (such as height, body proportions etc.) can be relied upon to remain largely consistent between frames. This fact is exploited through an extension that learns a single set of *global* shape parameters β and is assigned to across all frames $\beta_t := \beta$. Another benefit offered by video is the opportunity to constrain inter-frame subject motion. Assuming a reasonable framerate, it is expected that the animal's pose and position parameters should vary only slightly between successive frames. This intuition is characterized in **??**.

The following section defines the 4 energy terms used in the optimization:

Silhouette energy.

The silhouette energy E_{sil} compares the 3D animal model to the silhouette image according to the L2 distance between the OpenDR rendered binary image and the input silhouette:

$$E_{\text{sil}}(\phi_t, \theta_t, \beta; S_t) = \|S_t - R\big(\phi_t * v(\theta_t, \beta)\big)\| \tag{3.23}$$

Unimodal Prior energy.

The prior term E_{prior} encourages the regressed shape and pose parameters to remain close to a those in the combined artist traininthose in our set of artist 3D dog meshes.

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.

$$E_{\text{lim}}(\theta_t) = \max\{\theta_t - \theta_{\max}, 0\} + \max\{\theta_{\min} - \theta_t, 0\}. \tag{3.24}$$

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.

$$E_{\text{joints}}(\phi_t, \theta_t, \beta; X^*) = \|X^* - \phi_t * v(\theta_t, \beta)\mathrm{K}_t(:, j)\| \tag{3.25}$$

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:

$$E_{\text{temp}}(\phi_t, \theta_t) = (\phi_t - \phi_{t+1})^2 + (\theta_t - \theta_{t+1})^2 \tag{3.26}$$

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

3.6 Experiments

This section details both quantitative and qualitative evaluation of the system described in the previous sections. Due to the lack of animal keypoint datasets in the public domain, it is necessary to construct a new Benchmark Animal Dataset of Joint Annotations (BADJA). BADJA is a dataset comprising several video sequences with hand-clicked 2D joint labels and segmentation masks. Qualitative results are provided on BADJA and also on images used in the evaluation of 3D Menagerie (3DM). However, it should be noted this comparison is not entirely fair, since 3DM requires hand-clicked keypoints as input and system introduced in this chapter is optimized for video input.

3.6.1 BADJA Dataset

The BADJA dataset contains 9 videos, with manually collected keypoint and silhouette annotations. 7 video sequences were obtained from the DAVIS video segmentation dataset [**?**] and came with existing silhouettes, and the remainder were sourced from online stock footage. Segementation masks for these were generated using Adobe's UltraKey tool [**?**]. For all sequences, a set of 20 joints were labelled as illustrated in Fig. **??**. The first 16 joints are located on legs, neck and tail, and defined by the rigged 3D SMAL skeleton. The final joints (nose tip, chin and ears) do not directly relate to 3D SMAL joints, but are instead defined by particular SMAL model vertices. A similar trick is used in SMALify [**?**] to indicate the position of the human nose which has no corresponding SMPL joint. The collection of joints were chosen on the basis of being of informative to the skeleton and being simple for a non-expert human annotator to localize. To make manual annotation feasible and to ensure a diverse set of data, annotations are provided for every fifth video frame and were collected using the excellent LabelMe tool [LabelMe].

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.

3.6.2 Joint prediction

For the joint predictor ρ the stacked hourglass network [**?**] modified for multi-modal output is trained on synthetic animal silhouette data. Following state-of-the-art performance on related human 2D pose estimation datasets ([**?** **?**]), the network consists of 8 stacks, 256 features and 1 block. Synthetic silhouette images of size 256×256 are provided as input, 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. As a benefit of working with synthetic data, training samples can be generated on the fly, which effectively results in a training set of infinite size. A small adaptation was required to prevent the network degenerating to an unfavourable solution on silhouette input: foreground masks were

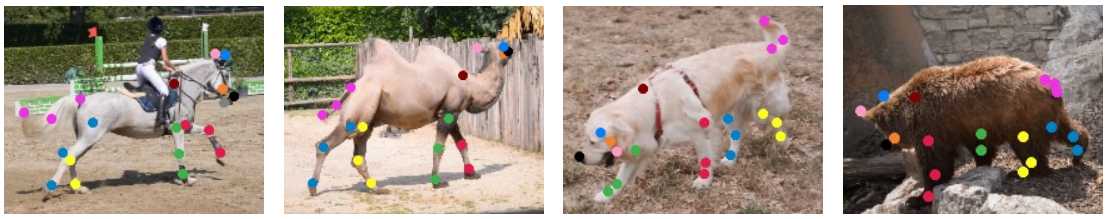


Fig. 3.7 Example joint annotations from the BADJA dataset. A total of 11 video sequences are in the dataset, annotated every 5 frames with 20 joint positions and visibility indicators.

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

Fig. **??** 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 (**??**b), occlusions (**??**c) and distraction objects such as the human rider in (**??**d). 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 **??** contains examples of failure modes.

3.6.3 Optimal joint assignment

Following non-maximum suppression of the joint heatmaps obtained in Section **??**, OJA is applied to select an optimal set of joints which initialize the final 3D model fitting 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. **??**, row 1) or by resolving the ambiguity between the left and right legs (Fig. **??**, row 2). Table **??** 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

(a) (b) (c)

Fig. 3.8 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 3.5 Accuracy of OJA on BADJA test sequences.

term. Across all sequences the selected OJA-GA method improves joint prediction accuracy by 7% compared to the raw network output.

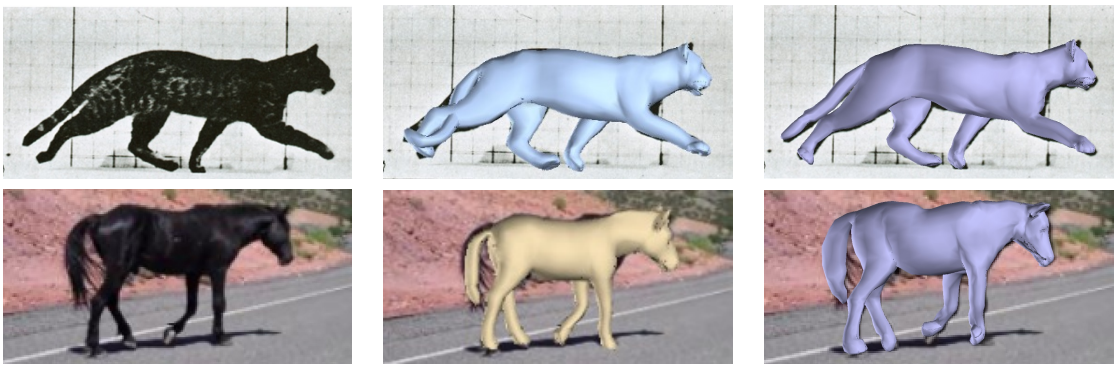
Seq.	Family	PCK (%)		Mesh	Seq.	Family	PCK (%)		Mesh
		Raw	OJA-GA				Raw	OJA-GA	
01	Felidae	91.8	91.9	38.2	06	Equidae	84.4	84.8	19.2
02	Felidae	94.7	95.0	42.4	07	Bovidae	94.6	95.0	40.6
03	Canidae	87.7	88.0	27.3	08	Bovidae	85.2	85.8	41.5
04	Canidae	87.1	87.4	22.9	09	Hippopotamidae	90.5	90.6	11.8
05	Equidae	88.9	89.8	51.6	10	Hippopotamidae	93.7	93.9	23.8

Table 3.6 Quantitative evaluation on synthetic test sequences. The performance of the raw network outputs and OJA methods are evaluated using the probability of correct keypoint (PCK) metric. Mesh fitting accuracy is evaluated by computing the mean distance between the predicted and ground truth vertices.

3.6.4 Model fitting

The predicted joint positions and silhouette are input to the 3D model fitting optimization phase, which proceeds in four stages. The first stage solves for the model's global rotation and translation parameters, which positions the camera. Following SMPLify [**?**], this camera stage is solved for torso points only, which remain largely fixed through shape and pose variation. The remaining stages solve for all shape, pose and translation parameters the emphasis of the priors gradually decreased. The silhouette term is introduced in the penultimate stage, as including this too early can lead to the optimizer finding unsatisfactory local minima.

The final outputs of our optimization pipeline are shown in Fig. **??**. 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. **??** demonstrates the generalizability of the



RGB SMAL [**?**] **Ours**

Fig. 3.9 Results are comparable in quality to SMAL [**?**], but note that we do not require hand-clicked keypoints.

Fig. 3.10 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.

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.

Our approach is compared visually to that given by Zuffi *et al.* [**?**]. Recall that their results require hand-clicked keypoints rather than fitting to points predicted automatically by the hourglass network, which was trained on synthetic animal images. Further, their work is optimized for single frame fitting and is tested on animals in simple poses, whereas the focus of this work is on the more challenging task of tracking animals in video. Fig. **??** shows the application of our model to a number of single frame examples from the SMAL result data [**?**].

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 **??**. Table **??** shows results on these sequences.

3.6.5 Automatic silhouette prediction

While not the main focus of our work, the system presented in this chapter is able to perform the full 3D reconstruction process from an input image with no user intervention. This is achieved by incorporating the DeepLabv3+ network [**?**] as a front-end segmentation engine which automatically generates 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. **??**.

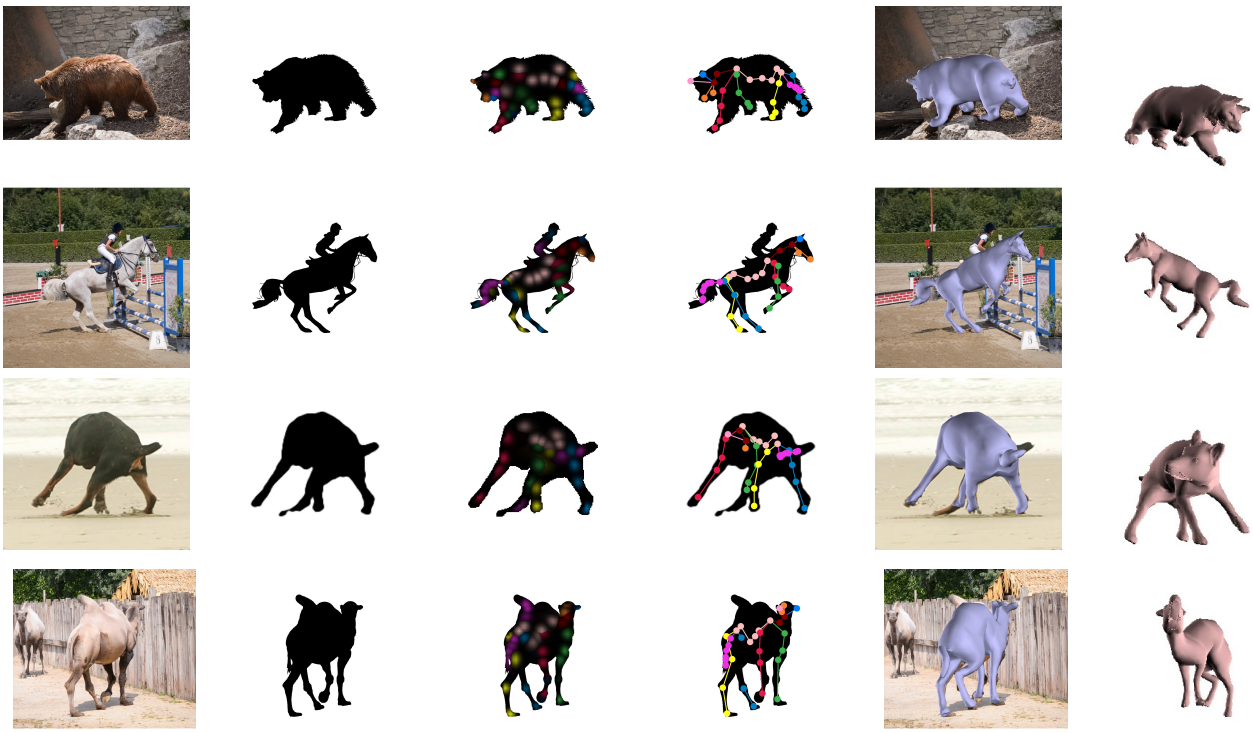


Fig. 3.11 Example results on various animals. From left to right: RGB input, extracted silhouette, network-predicted heatmaps, OJA-processed joints, overlay 3D fit and alternative view.

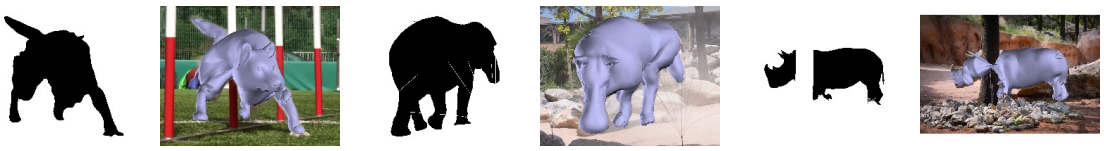


Fig. 3.12 Failure modes of the proposed system. *Left*: Missing interior contours prevent the optimizer from identifying which way the dog is facing. *Middle*: The model has never seen an elephant, so assumes the trunk is the tail. *Right*: Heavy occlusion. The model interprets the tree as background and hence the silhouette term tries to minimize coverage over this region.

3.7 Conclusions

The chapter introduces a technique for reconstructing 3D quadrupeds from video by using a quadruped model parameterized in shape and pose. By incorporating automatic segmentation tools, the pipeline can be deployed without requiring human intervention or even precise knowledge of the species of animal being considered. The method performs well on examples encountered in the real world, generalizes to unseen animal species and is robust to challenging shapes and poses.

As a direction for future work, it would be worthwhile to look at methods for synthetic image generation that preserve important edge information lost during silhouette extraction. In particular, the lost interior contours cause ambiguities which necessitate the OJA method described here. An alternative is to extend the recent work of SMALST [SMALST] (which requires hand-clicked training images) by instead synthetizing realistic textures using generative adverserial networks. Of course, it is important to ensure the texture generation process preserves the sampled pose parameters. A naive method is to rendering texture as a UV map on top of the SMAL model, but this could also be framed as an image-to-image translation problem, beginning with a low detail SMAL render and mapping to a photorealistic image with a carefully designed pose-preserving generator. Other components of the system which could be improved would be building a more sophisticated motion prior, able to represent likely animal trajectories. In addition, robustness to environmental factors as shown in **??** could be improved by rendering synthetic causes of occlusion (perhaps even as rectangles).

Chapter 4

End-to-end Dog Shape Recovery with a Learned Shape Prior

4.1 Introduction

This chapter introduces WLDO, an automatic and end-to-end neural network which recovers the 3D pose and shape of dogs from monocular internet images. The large variation in shape between dog breeds, significant occlusion and low quality of internet images makes this a challenging problem. In addition, the dog cateogry is poorly represented by existing 3D morphable models. Despite the ubiquity of dogs in society (there are more than 63 million pet dogs in the US alone [**?**]), these factors have perhaps contributed to the lack of effective 3D reconstruction methods. This chapter demonstrates that the natural variation present in a 2D dataset is sufficient to learn a detailed 3D dog prior, which helps regularize parameter estimation. WLDO achieves this by adding additional limb scaling parameters to the morphable model, and including expectation maximization (EM) update steps into the network training loop. These contributions are shown experimentally to improve the quality of reconstruction. Presently, state of the art animal 3D reconstruction techniques incoporate per-image (or per-sequence as in **??**) test time energy minimization procedures that prevent real-time application. Inspired by recent methods for human reconstruction [**?**], WLDO comprises an end-to-end convolutional neural network which directly regresses morphable model parameters with no subsequent energy minimization phase, achieving inference at approximately 10 frames per second. This is important for downstream tasks, such as animal monitoring systems, which rely on live data in order to alert experts to immediate causes for concern. The experimentation section is based on StanfordExtra, a new 'in the wild' dataset of dog images, containing 120 breeds. The method presented achieves state of the art performance on this dataset, shows strong generalization characteristics to a new dataset, and outperforms model fitting approaches, even when they are given access to ground truth annotations at test time.

4.1.1 Adding local parameters to the PCA shape space

As discussed in **??**, there is a long history of 3D reconstruction approaches which use morphable models to represent articulated subjects. Recently, the dominant paradigm is to factor the deformation space into a parameterization based on pose (which governs articulated limb positions) and a global PCA shape space governing body proportions. While efficient and differentiable, even as early as the seminal paper of Blanz and Vetter [,] it was noted that globally defined shape representations are poor at representing fine details (in their case, features such as the eyes and nose). Blanz and Vetter proposed tackling this by by manually segmenting the face into separate regions and learning separate PCA models per regions. While this does achieve higher fidelity modelling, it comes at the cost of a less compact shape space representation and a need to manually partition the object. Alterantively, and most closely related to the work in this chapter, is the recent work of [STAR] which serves as a drop in replacement for the original SMPL [SMPL] human model. The authors note that the use of SMPL's global blend parameters result in the need for dense pose-corrective offsets in order to relate every mesh vertex with every kinematic tree joint. This causes an unwanted effect of capturing suprious long-range correlations between seemingly unrelated parts of the mesh. STAR offers an improvement with a local formulation, learning a set of mesh vertices which are influenced by each joint's movement. Another limitation of globally defined PCA spaces (which is of particular relevance to **??**) is that it is difficult to relate uncertanties in reconstructed shape to specific parts of the mesh. For example, since each body part is governed by multiple shape parameters, so are the uncertanties, making it challenging to reason about occluded parts. These challenges are compounded in a low training data setting. The SMAL animal model is built from scans of 41 toy figurines, which results in a PCA shape space that captures unwanted correlations coincidentally present in the training corpus. Examples of this are shown in Figure XXX. This chapter demonstrates an approach for adding a few extra local shape parameters to the SMAL model, allowing the body part to scale independently to the rest of the mesh. These parameters help the model generalize to dogs outside of the 3D training samples, are an inexpensive addition to the generator function and produce better shape reconstructions for WLDO and a previous approach based on energy minimization.

4.1.2 Automatic and real-time 3D dog reconstruction

Early 3D reconstruction approaches in humans [examples] use an energy minimization framework which iteratively optimizes a 3D morphable model to match an input image or video sequence. These methods typically design an energy function that balances *data terms* to encourage strong alignment between the 3D model and input image, and *prior terms* which ensure realistic predictions. However, more recent works frame the reconstruction task as a direct regression from the input image to 3D model parameters. These are typically implemented using convolutional neural networks which are trained on large and varied datasets and now achieve state-of-the-art performance on a number of benchmark datasets. Alongside fast test-time performance, deep learning methods learn accurate priors over the object category which lead to improved performance on images with occlusion and

they do not fall into failure cases common to optimization techniques, generally caused by poor model initialization. However, a downside of deep learning methods is their reliance on large datasets. Apart from the input 3D morphable model (e.g. SMPL [SMPL]), typically required are large *paired 3D datasets* and *unpaired 2D datasets* containing images and 2D annotations. Paired 3D datasets help models learn an association between input image features and 3D shape and pose parameters. However, such datasets require specialized equipment to collect (e.g. motion capture) resulting in limited variety in captured scenes. To overcome this, unpaired 2D datasets typically containing 2D keypoint and/or silhouette annotations are used to help models generalize to "in-the-wild" scenarios. Occasionally, these methods are also evaluated in an 'unpaired' setting, in which the paired datasets are ommited. Under these conditions, methods must overcome fundamental ambiguities to learn the 2D-to-3D mapping, or risk predicting 3D bodies with impossible joint angles or have implausible body weight distributions. Explicit 3D priors are often learned during training to ensure the predicted models lie on the manifold of plausible bodies. The "unpaired" training mode is of most relevance to this chapter's task of reconstructing 3D dogs, since paired 3D animal data is severely limited. Of concern, then, is how to design a suitable prior for the dog category. This is explored in depth in the following sections.

Existing 3D animal reconstruction techniques are either entirely optimization based or incoporate optimization procedures as a part of the prediction pipeline. Notable exceptions include deep networks used to reconstruct unarticulated categories such as birds (e.g. CMU, UCMU) or the technique of Kulkarni et al. [] which operates on articulated categories but does not recover shape characteristics. SMALST is perhaps the closest attempt to an end-to-end technique for articulated shape and pose recovery, although video sequences are required for training and a test-time optimization strategy is used to refine initial regressed parameters, preventing real time operation. Further, it can be argued that the zebra species tackled in this paper is more limited than the 120 dog breeds examined in this chapter. Inspired by these approaches and the recent end-to-end work in human mesh reconstruction, WLDO comprises a deep neural network that directly regresses 3D dog model parameters *with no subsquent optimization phase* to enable real-time inference.

4.1.3 Learning a 3D animal prior

While detailed 3D morphable models and associated priors are already available for humans [,] the limited 3D training data for animals has made designing equivalent resources more difficult. The SMAL paper proposed an approach for building a morphable quadruped models from a few toy figurines rather than scans of real subjects (such as used for SMPL [SMPL]. Shape and pose priors were similarly collected by fitting the SMAL model to 2-3 artist animation sequences. Consequently, the SMAL model and priors are of relatively low fidelity and poorly represent some species, particularly dogs. Alongside the prevalence of challenging poses, occlusion and difficult environmental factors, overly restrictive animal priors have likely contributed to the lack of effective 3D reconstruction approaches for the dog category. **??** has already demonstrated an example of this, in which synthetically

data generated according to the SMAL prior was used to train a 2D joint predictor. While high quality results are shown on categories better represented by SMAL, the experimental section of this chapter shows the method generalizes poorly to some out-of-domain dog breeds.

Some recent deep learning methods overcome this by forgoing data-driven 3D priors for the regression task altogether. SMALST is one such example; they supervise a deep network using 3D synthetic models generated using video sequences. Other techniques [] instead use smoothness terms, deformation constraints and symmetry constraints, although these are only effective when modelling unarticulated categories such as birds and dolphins. Of course, the quality of 3D morphable models, animal priors and potentially reconstruction results could be improved by collecting larger datasets of detailed 3D scans. However, collecting such a dataset would be expensive and time-consuming, either relying on scans of real animal subjects or contracting talented 3D graphics artists to build hundreds (or thousands) of accurate models. This chapter proposes an alternative approach, arguing that even a low-quality, unimodal 3D shape prior can act as a useful initialization for a novel *refinement process*, which learns a more expressive, multimodal prior by learning from an annotated 2D dataset.

4.1.4 Refinement steps "in the loop"

A key insight relied upon in this chapter is the potential for collaboration between a deep neural network which processes input images to regress 3D model parameters and an optimization process that tunes a 3D shape prior. The technique used for this was inspired by the SPIN network of Kolotouros et al.[] who introduced a 3D human reconstruction approach based on a similar hybrid. In their approach, the HMR backbone [HMR] is used to yield a set of SMPL [SMPL] human body shape and pose parameters Θ_{reg} from an input image. However, during SPIN's training phase the initial fit Θ_{reg} is processed by the SMPLify [SMPLify] model fitting procedure, which refines the initial fit based on the ground truth joints to yield a new estimate Θ_{opt}. The difference between these two predictions is expressed as a loss $||\Theta_{reg} - \Theta_{opt}||$ which is backpropagated to incrementally improve the predictions of the regression network. At test time, only the regression network is used meaning the inference speed is unaffected.

A related approach is used in this chapter in order to incrementally improve the representational power of a 3D shape prior. In particular, WLDO's training phase incoporates a optimization strategy based on expectation maximization which regularly updates the prior based on weights learned from an annotated 2D image dataset. Similarly to SPIN, this creates a collaborative effect: the representational power of the 3D shape prior gradually improves by learning from better predictions produced by the 3D reconstruction network, and the 3D reconstruction network learns from the improving shape prior to produce accurate fits. Experimentally, it is shown that WLDO's reconstructed dog models are of good enough quality to avoid a time consuming test time optimization process. Further details introducing expectation maximization and it's use in WLDO's training loop are deferred to the method section below.

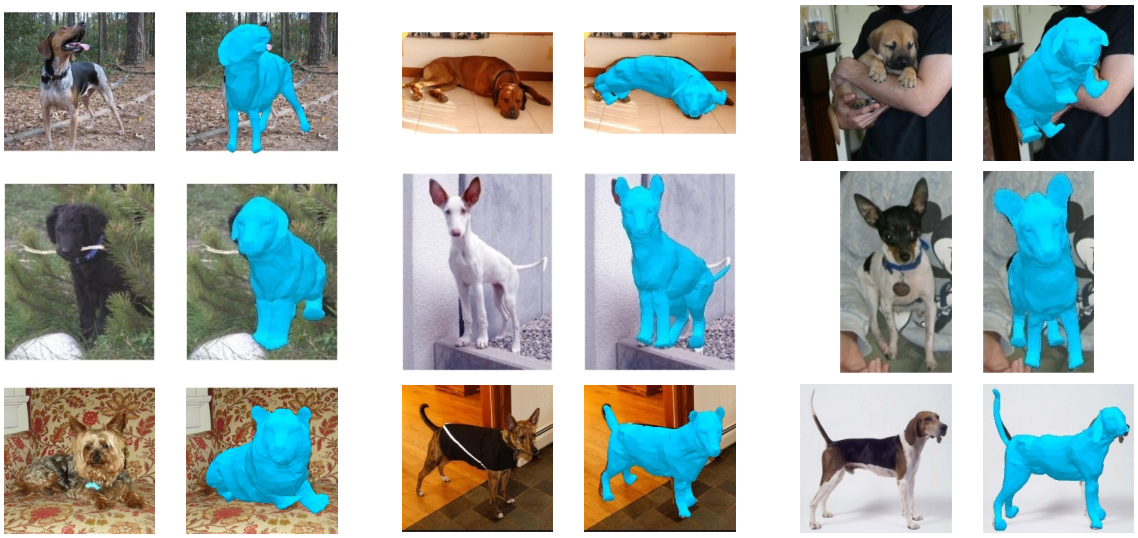


Fig. 4.1 **End-to-end Dog Shape Recovery with a Learned Shape Prior.** We propose a novel method that, given a monocular image of a dog can predict a set of parameters for our SMBLD 3D dog model which is consistent with the input. We regularize learning using a multi-modal shape prior, which is tuned during training with an expectation maximization scheme.

4.1.5 Contributions

This chapter proposes a number of contributions which extend the state of the art in 3D animal reconstruction in several ways. While each contribution is inspired by recent literature, the WLDO approach is the first to exhibit the combination, leading to a new state of the art state of the art in terms of scale and object diversity.

1. Reconstructing pose and shape on a test set of 1703 low-quality internet images of a complex 3D object class (dogs).
2. Direct regression to object pose and shape parameters from a single image without a model fitting stage.
3. Use of easily obtained 2D annotations in training, and none at test time.
4. Learning of a new multi-modal prior in the training phase (via EM update steps), rather than fitting it to 3D data as in previous work.
5. Introducing new degrees of freedom to the SMAL model, allowing explicit scaling of subparts.

4.2 Building SMBLD: a new parametric dog model

At the heart of the method is a parametric mesh representation of a 3D animal, which is based on the Skinned Multi-Animal Linear (SMAL) model proposed by Zuffi et al. [**?**]. As discussed in **??**,

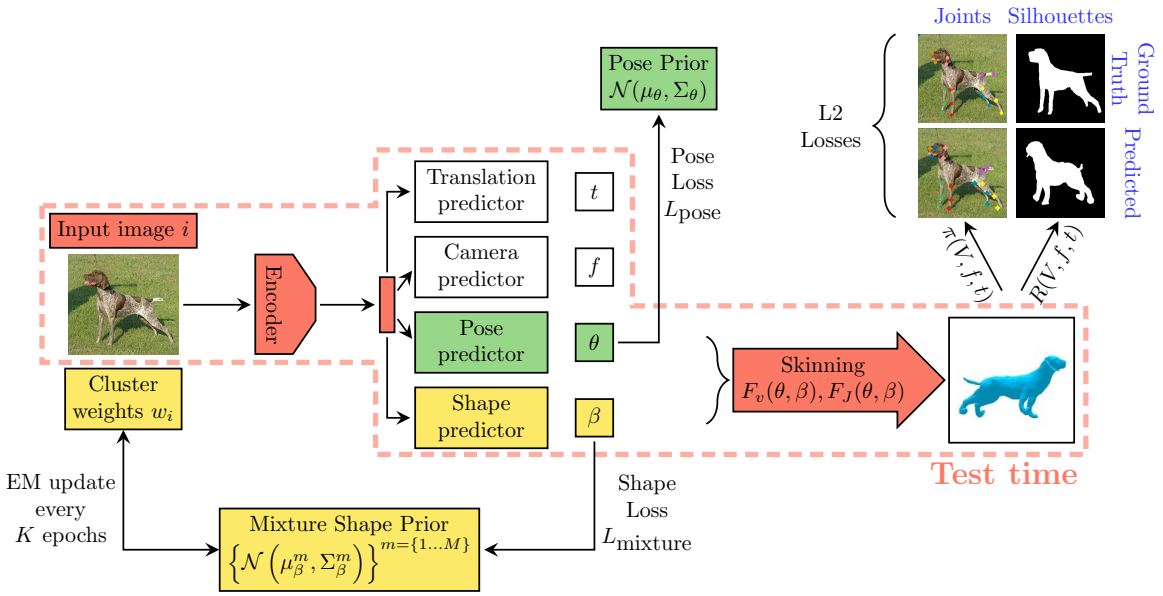


Fig. 4.2 Our method consists of (1) a deep CNN encoder which condenses the input image into a feature vector (2) a set of prediction heads which generate SMBLD parameters for shape β, pose θ, camera focal length f and translation t (3) skinning functions F_v and F_J which construct the mesh from a set of parameters, and (4) loss functions which minimise the error between projected and ground truth joints and silhouettes. Finally, we incorporate a mixture shape prior (5) which regularises the predicted 3D shape and is iteratively updated during training using expectation maximisation. At test time, our system (1) condenses the input image, (2) generates the SMBLD parameters and (3) constructs the mesh.

SMAL is a deformable 3D quadruped mesh parameterized by shape and pose. The *shape* $\beta \in \mathbb{R}^B$ parameters are PCA coefficients of an undeformed template mesh with limbs in default position. The *pose* $\theta \in \mathbb{R}^P$ parameters meanwhile govern the joint angle rotations (35×3 Rodrigues parameters) which effect the articulated limb movement.

First, it is important to give a detailed description of the SMAL generator function, which is later augmented with scale parameters to improve model flexibility.

$$v(\theta,\beta) = W(T_P(\theta,\beta), J(\beta), \theta, \mathcal{W}) \tag{4.1}$$

$$T_P(\theta,\beta) = \bar{T} + B_S(\beta) + B_P(\theta) \tag{4.2}$$

Here, SMAL relies on a standard linear blend skinning (LBS) formulation to return a set of posed vertices from SMAL template vertices in rest pose $\bar{T}$, SMAL 3D joint locations J, a pose vector θ and blend weights $\mathcal{W}$

$$W(\bar{T}, J, \theta, \mathcal{W}) : \mathbb{R}^{3N\times 3K\times|\theta|\times|\mathcal{W}|} \mapsto \mathbb{R}^{3N} \tag{4.3}$$

The rest pose mesh $T_P(\theta,\beta) \in \mathbb{R}^{3889\times 3}$ is constructed from the SMAL template vertices $\bar{T}$ and additive vertex offsets $B_S(\beta)$ and $B_P(\theta) \in \mathbb{R}^{3889\times 3}$. These are referred to as shape and pose blend

shapes respectively. Note that despite being a function of pose, $B_P(\theta)$ still represents a shape modification. This is often referred to as a *corrective* offset, fixing artefacts common to linear blend skinning, often by fat. The pose vector $\theta = \left[\omega_0^T, \ldots, \omega_J^T\right]^T$ contains 3 parameters for each joint. Each element $\omega_j \in \mathbb{R}^3$ denotes the axis-angle representation of the relative rotation of each part j with respect to its parent in the kinematic tree. The local 3×3 rotation matrix corresponding to a joint j is denoted $\exp(\omega_j)$ and is obtained from axis angle representation using the *Rodrigues formula*

$$\exp(\omega_j) = \mathcal{I} + \hat{\bar{\omega}}_j \sin(||\omega_j||) + \hat{\bar{\omega}}_j^2 \cos(||\omega_j||) \tag{4.4}$$

where $\hat{\bar{\omega}}_j$ is the skew symmetric matrix of the 3-vector $\bar{\omega} = \frac{\omega}{||\omega||}$ representing the unit norm axis of rotation. $\mathcal{I}$ is the 3×3 identity matrix.

The linear blend skinning function (LBS) is then used to articulated the rest pose mesh $T_P(\theta, \beta)$. Let $t_i = \bar{t}_i + b_{S,i}(\beta) + b_{P,i}(\theta) \in T_P(\theta, \beta)$ be a *shaped* vertex in rest pose where $b_{S,i}(\beta)$ and $b_{P,i}(\theta)$ denotes the same vertex in $B_S(\beta)$ and $B_P(\theta)$ respectively. The transformation to obtain a *posed* vertex t_i' is then

$$t_i' = \sum_{k=1}^{K} w_{k,i} G_k'(\theta, J) t_i \tag{4.5}$$

$$G_k'(\theta, J) = G_k(\theta, J) G_k(\theta^*, J)^{-1} \tag{4.6}$$

$$G_k(\theta, J) = \prod_{j \in A(k)} \left(\begin{array}{c|c} \exp(\omega_j) & j_j \\ \hline \mathbf{0} & \mathbf{1} \end{array}\right) \tag{4.7}$$

Here, $w_{k,i}$ is an element of the blend weight matrix $\mathcal{W}$, representing how much the rotation of part k effects the vertex i, $\exp(\omega_j)$ is the local 3×3 rotation matrix corresponding to joint j, $G(\theta, J)$ is the world transformation of joint k, and $G'(\theta, J)$ is the same transformation after removing the transformation due to the rest pose θ^*. Each 3-element vector in J corresponding to a single joint center j, is denoted j_j . Finally, $A(k)$ denotes the ordered set of joint ancestors of joint k.

Shape blend shapes

Shape blend shapes are applied to the SMAL rest model to smoothly modify the mesh's body proportions. This has the effect of appearing to alter the animal category. The body shapes are globally represented by a linear function B_S

$$B_S(\beta; \mathcal{S}) = \sum_{n=1}^{|\beta|} \beta_n S_n \tag{4.8}$$

where $\beta \in \mathbb{R}^{41}$ is a vector of linear shape coefficients and S_n represents the orthnormal principle components of shape displacements, learned from the registered training scans.

Pose blend shapes

Let $R : \mathbb{R}^{|\theta|} \mapsto \mathbb{R}^{9K}$ be a function which maps the pose vector θ to a vector of concatenated part relative rotation matrices $\exp(\theta)$. Given the SMAL model contains 33 joints, $R(\theta)$ is a vector of length $33 \times 9 = 297$. Since elements of $R(\theta)$ comprise sine and cosine functions of the input pose vector, $R(\theta)$ is therefore non-linear with respect to θ.

The pose blend shapes are defined to be linear in $R^*(\theta) = (R(\theta) - R(\theta^*))$ where θ^* denotes the rest pose. Letting $R_n(\theta)$ denote the n^{th} element of $R(\theta)$, the vertex deviations from the rest template are

$$B_P(\theta;\mathcal{P}) = \sum_{n=1}^{9K} (R_n(\theta) - R_n(\theta^*))P_n \tag{4.9}$$

An important observation is that by subtracting the rest pose $R(\theta^*)$, the contribution of the pose blend shape is zero when the model is in the rest pose.

Joint locations

Differing animal body shapes have an effect on the 3D joint locations.

$$J(\beta;J,\bar{T},\mathcal{S}) = \mathcal{J}(\bar{T} + B_S(\beta;\mathcal{S})) \tag{4.10}$$

where $\mathcal{J}$ is a matrix that transforms rest vertices into rest joints.

Complete formulation

The complete formulation is therefore given as

$$v(\theta,\beta;\bar{T},\mathcal{W},\mathcal{S},\mathcal{J},\mathcal{P}) = W\left(T_P(\theta,\beta;\bar{T},\mathcal{S},\mathcal{P}),J(\beta;\mathcal{J},\bar{T},\mathcal{S}),\theta,\mathcal{W}\right) \tag{4.11}$$

4.2.1 Introducing scale parameters

While SMAL has been shown to adequately represent a variety of quadruped types, the modes of dog shape variation are poorly captured by the current model. This is unsurprising, since SMAL used only four dogs in its construction. This limitation is overcome using a simple but effective method that improves the model's representational power over this particularly diverse animal category. The set of shape parameters β are augmented with an additional set κ which independently scale parts of the mesh. For each model joint, parameters $\kappa_x, \kappa_y, \kappa_z$ are defined which apply a local scaling of the mesh along the local coordinate x, y, z axes, before pose is applied.

A simple method for adding scaling parameters is to apply a transformation to the rotation matrices for each joint. **??** then becomes

$$G_k(\theta,J) = \prod_{j \in A(k)} \left(\begin{array}{c|c} Q_j \exp(\omega_j) & j_j \\ \hline \mathbf{0} & \mathbf{1} \end{array}\right) \tag{4.12}$$

where the matrix

$$Q_j = \begin{pmatrix} \kappa_{j,x} & 0 & 0 \\ 0 & \kappa_{j,y} & 0 \\ 0 & 0 & \kappa_{j,z} \end{pmatrix} \tag{4.13}$$

Unfortunately, this formulation causes challenges should a user wish to scale a mesh subpart along the kinematic tree. While it is natural for joint *rotations* to be compositionally applied along the kinematic tree (mirroring the effect of skeletal limb motion) this does not follow for scaling parameters. For example, it is common for a user to apply scaling to a subpart which does not necessarily affect subsequent parts. With the formulation at present, it would be the responsibility of the user to "undo" scaling for the remaining tree, requiring them to be aware of the kinematic tree index ordering. This is seen as suboptimal behaviour, so the following formulation is preferred, defining *absolute* scale parameters that do not propagate their effect. Taking advantage of the hierarchical ordering of the kinematic tree for joint k, $A(k) = [A(k)_0, A(k)_1, \ldots, k]$ are joints along the tree. Therefore, the absolute scaling matrix is given by

$$G_k(\theta, J) = \left(\begin{array}{c|c} \exp(\omega_{A(k,0)})Q_{A(k,0)} & j_{A(k,0)} \\ \hline \mathbf{0} & \mathbf{1} \end{array}\right) \prod_{j \in [A(k)_1, \ldots, k]} \left(\begin{array}{c|c} Q_{j-1}^{-1} \exp(\omega_j) Q_j & j_j \\ \hline \mathbf{0} & \mathbf{1} \end{array}\right) \tag{4.14}$$

Note that Q_{j-1} refers to the parent scaling matrix of joint j. Note that the scale parameters in this formulation determine only the scale for the particular subpart by undoing that of the parent.

4.2.2 Sharing scale parameters

Allowing each joint to scale entirely independently can however lead to unrealistic deformations. To overcome this, scale parameters are shared between multiple joints which are physiologically related and would naturally scale together. For example, the y-scaling for the four legs (which governs the length) is controlled by only a single parameter. The new Skinned Multi-Breed Linear Model for Dogs (SMBLD) is therefore adapted from SMAL by adding 6 scale parameters to the existing set of shape parameters. Figure **??** shows how introducing scale parameters increases the flexibility of the SMAL model. The next task is to learn an prior (with which to initialize the later EM process) that covers these new scale parameters.

4.2.3 Initializing a dog shape/scale prior

The SMBLD shape prior is initialized by fitting the model to a set of 13 artist-designed 3D dog meshes, downloaded from the internet and are similar in style to the SMAL toys. While the collection does offer a few more examples of dogs than used in the original SMAL set, the models still lack realism and are very constrained when compared to the 120 different breeds represented in the later test set. An energy minimization process is used to align the SMBLD vertices to each scan, in order to obtain a set of shape β, pose θ and κ parameters.

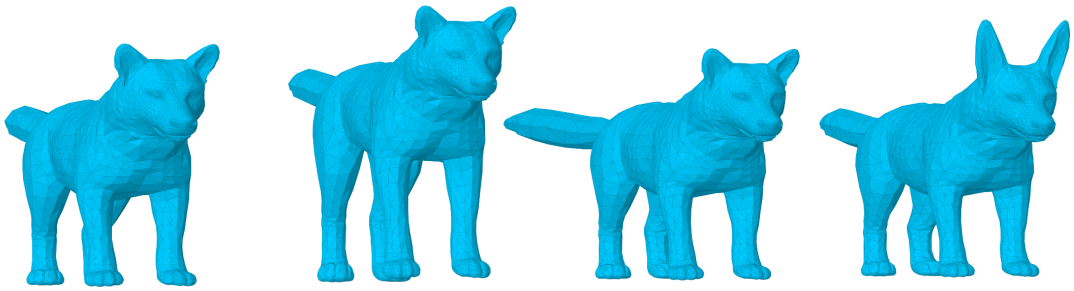


Fig. 4.3 **Effect of varying SMBLD scale parameters**. *From left to right*: Mean SMBLD model, 25% leg elongation, 50% tail elongation, 50% ear elongation.

Fitting SMBLD to 3D scans

Recall that SMBLD's shape β and θ parameters are exactly as defined in SMAL, meaning there is no need to relearn (or adapt) the blend shape data $\mathcal{P},\mathcal{S}$ or joint selectors $\mathtt{K}$. The scale κ parameters, however, are new to SMBLD. The goal of this section is to learn a new 3D prior over both shape and scale parameters to initialize the prior refined during the WLDO training loop.

Note that fitting SMBLD to 3D scans emits much simpler optimization in comparison to 2D images, since the complete 3D information of the target mesh is available. In addition, the target meshes are not particularly detailed and are already aligned in the canonical T-pose, so we avoid need for a complex alignment technique as discussed in **??**.

An energy minimization process is used to align the SMBLD mesh $v(\theta,\beta)$ to each of the 3D scans $[(V,T)_i]_{i=1}^N$ subject to smoothing regularizers. The following energy formulation is minimized using stochastic gradient descent (SGD) with learning rate $1.0e^{-4}$ for 1000 iterations

$$E_{\text{opt}} = E_{\text{chamfer}} + E_{\text{laplacian}} + E_{\text{edge}} + E_{\text{normal}} \tag{4.15}$$

where each of these terms has a scalar weight λ. Here, $\lambda_{\text{chamfer}} = \lambda_{\text{edge}} = 1.0$, $\lambda_{\text{normal}} = 0.01$ and $\lambda_{\text{laplacian}} = 0.1$. Further details on the specific energy terms are now provided.

Chamfer energy.

Chamfer energy measures the average distance between vertices of the SMBLD mesh $V = v(\theta,\beta)$ and the target mesh vertices V'. Let $V_P = [v_i]_{i=1}^p$ and $V'_P = [v'_i]_{i=1}^p$ be random p-subsets of V and V' respectively. The chamfer energy is then

$$E_{\text{chamfer}}(V_P, V'_P) = \frac{1}{p}\sum_{i=1}^{p} \min_j \left|v_i - v'_j\right| \tag{4.16}$$

Uniform laplacian energy.

The SMAL mesh can be viewed as $M = (v(\theta, \beta), T)$ where the vertices are given by the generator function above and the triangulation is fixed. The uniform Laplacian matrix $L \in \mathbb{R}^{N \times N}$ is defined as $LV = w_i \sum_j (v_j - v_i)$ where $w_{ij} = 1/|S_i|$ and S_i is the set of neighbouring vertices to i. Note that LV_i points to the centroid of the neighbouring vertices. The energy formulation is then

$$E_{\text{laplacian}} = \sum_i w_i \sum_j (v_j - v_i) \tag{4.17}$$

Edge energy.

This energy is equal to the average edge length across the mesh, and is used to encourage uniform distribution of vertices. Let $E = \{ (i, j) \mid v_i, v_j \in V \text{ are adjacent vertices} \}$, then

$$E_{\text{edge}} = \frac{1}{|E|} \sum_{(i,j) \in E} ||v_j - v_i|| \tag{4.18}$$

Normal energy.

Normal energy is a measure of the average normal consistency between adjacent triangles. For two triangles $t_i, t_j \in T$ in SMAL's fixed triangle set, let n_i, n_j be the respective normals. Recall that $n = (v_1 - v_0) \cdot (v_2 - v_0)$. Then normal consistency is defined by

$$E_{\text{normal}} = 1 -$$

4.3 End-to-end dog reconstruction from monocular images

This section considers the task of reconstructing a 3D dog mesh from a monocular image. To achieve this, an end-to-end convolutional neural network is trained to predict a set of parameters for the SMBLD model and also for a perspective camera. In particular, the network is trained to predict pose θ and shape-scale $\beta - space$ SMBLD parameters together with translation t and focal length f for a perspective camera. A complete overview of the proposed system is shown in Figure **??**.

4.3.1 Model architecture

The network architecture for WLDO is inspired by the model of 3D-Safari [**?**]. Given an input image cropped to (224, 224), a Resnet-50 [**?**] backbone network is applied in order to encode a 1024-dimensional feature map. These features are passed through various linear prediction heads to produce the required parameters. The pose, translation and camera prediction modules follow the design of 3D-Safari, but we describe the differences in our shape module.

Pose, translation and camera prediction.

These modules are independent multi-layer perceptrons which map the above features to the various parameter types. As with 3D-Safari we use two linear layers to map to a set of 35×3 3D pose parameters (three parameters for each joint in the SMBLD kinematic tree) given in Rodrigues form. We use independent heads to predict camera frame translation $t_{x,y}$ and depth t_z independently. We also predict the focal length of the perspective camera similarly to 3D-Safari.

Shape and scale prediction.

Unlike 3D-Safari, we design our network to predict the set of shape parameters (including scale) rather than vertex offsets. We observe improvement by handling the standard 20 blend-shape parameters and our new scale parameters in separate linear prediction heads. We retrieve the scale parameters by $\kappa = \exp x$ where x are the network predictions, as we find predicting log scale helps stabilise early training.

4.3.2 Training losses

A typical approach for training such an end-to-end system would be to supervise the prediction of (θ, β, t, f) with 3D ground truth annotations [**?** **?** **?**]. However, building a suitable 3D annotation dataset would require an experienced graphics artist to design an accurate ground truth mesh for each of 20,520 StanfordExtra dog images, a prohibitive expense.

We instead develop a method that instead relies on *weak 2D supervision* to guide network training. In particular, we rely on only 2D keypoints and silhouette segmentations, are significantly cheaper to obtain.

The rest of this section describes the set of losses used to supervise the network at train time.

Joint reprojection.

The most important loss to promote accurate limb positioning is the joint reprojection loss L_{joints} which compares the projected model joints $\pi(F_J(\theta, \beta), t, f)$ to the ground truth annotations $\hat{X}$. Given the parameters predicted by the network, we apply the SMBLD model to transform the pose and shape parameters into a set of 3D joint positions $J \in \mathbb{R}^{35\times 3}$, and project them to the image plane using translation and camera parameters. The joint loss L_{joints} is given by the ℓ_2 error between the ground truth and projected joints:

$$L_{\text{joints}}(\theta, \beta, t, f; \hat{X}) = \|\hat{X} - \pi(F_J(\theta, \beta), t, f)\|_2 \tag{4.20}$$

Note that many of our training images exhibit significant occlusion, so $\hat{X}$ contains many invisible joints. We handle this by masking L_{joints} to prevent invisible joints contributing to the loss.

Silhouette loss.

The silhouette loss L_{sil} is used to promote shape alignment between the SMBLD dog mesh and the input dog. In order to compute the silhouette loss, we define a rendering function $R : (v,t,f) \mapsto S$ which projects the SMBLD mesh to produce a binary segmentation mask. In order to allow derivatives to be propagated through R, we implement R using the differentiable Neural Mesh Renderer [**?**]. The loss is computed as the ℓ_2 difference between a projected silhouette and the ground truth mask $\hat{S}$:

$$L_{\text{sil}}(\theta,\beta,t,f;\hat{S}) = \|\hat{S} - R\big(F_V(\theta,\beta),t,f\big)\|_2 \tag{4.21}$$

Priors.

In the absence of 3D ground truth training data, we rely on priors obtained from artist graphics models to encourage realism in the network predictions. We model both pose and shape using a multivariate Gaussian prior, consisting of means μ_θ, μ_β and covariance matrices $\Sigma_\theta, \Sigma_\beta$. The loss is given as the log likelihood of a given shape or pose vector under these distributions, which corresponds to the Mahalanobis distance between the predicted parameters and their corresponding means:

$$L_{\text{pose}}(\theta;\mu_\theta,\Sigma_\theta) = (\theta-\mu_\theta)^T\Sigma_\theta^{-1}(\theta-\mu_\theta) \tag{4.22}$$

$$L_{\text{shape}}(\beta;\mu_\beta,\Sigma_\beta) = (\beta-\mu_\beta)^T\Sigma_\beta^{-1}(\beta-\mu_\beta) \tag{4.23}$$

Unlike previous work, we find there is no need to use a loss to penalize pose parameters if they exceed manually specified joint angle limits. We suspect our network learns this regularization naturally because of our large dataset.

4.4 Expectation Maximization in the Loop

As previously discussed, our initial shape prior is obtained from artist data which we find is unrepresentative of the diverse shapes present in our real dog dataset.

Each training image i is assigned a set of latent variables $\{{}^1_i, \ldots, {}^M_i\}$ encoding the probability of the dog shape in image i being generated by component m.

We address this by proposing to recover the latent variables w_i^m and parameters (μ_β^m, Σ_β^m and Π_β^m) of our 3D shape prior by learning from monocular images of in-the-wild dogs and their 2D training labels in our training dataset.

We achieve this using Expectation Maximization (EM), which regularly updates the means and variances for each mixture component and per-image mixture weights based on the observed shapes in the training set. While training our 3D reconstruction network, we progressively update our shape mixture model with an alternating 'E' step and 'M' step described below:

The 'E' Step.

The 'E' step computes the expected value of the latent variables w_i^m assuming fixed $(\mu_\beta^m, \Sigma_\beta^m, \Pi_\beta^m)$ for all $i \in \{1,\ldots,N\}, m \in \{1,\ldots,M\}$.

The update equation for an image i with latest shape prediction β_i and cluster m with parameters $(\mu_\beta^m, \Sigma_\beta^m, \Pi_\beta^m)$ is given as:

$$w_i^m := \frac{\mathcal{N}(\beta_i|\mu_\beta^m, \Sigma_\beta^m)\Pi_\beta^m}{\sum_{m'}^{M} \mathcal{N}(\beta_i|\mu_\beta^{m'}, \Sigma_\beta^{m'})\Pi_\beta^{m'}} \tag{4.24}$$

The 'M' Step.

The 'M' step computes new values for $(\mu_\beta^m, \Sigma_\beta^m, \Pi_\beta^m)$, assuming fixed w_i^m for all $i \in \{1,\ldots,N\}, m \in \{1,\ldots,M\}$.

The update equations are given as follows:

$$\mu_\beta^m := \frac{\sum_i w_i^m \beta_i}{\sum_i w_i^m} \quad \Sigma_\beta^m := \frac{\sum_i w_i^m (\beta_i - \Sigma_\beta^m)(\beta_i - \Sigma_\beta^m)^T}{\sum_i w_i^m} \quad \Pi_\beta^m := \frac{1}{N}\sum_i w_i^m \tag{4.25}$$

4.5 Building StanfordExtra: a new large-scale dog keypoint dataset

Fig. 4.4 **StanfordExtra example images**. *Left*: outlined segmentations and labelled keypoints for 24 representative images. *Right*: heatmap of deviation of worker submitted results from mean for each submission.

In order to evaluate our method, we introduce *StanfordExtra*: a new large-scale dataset with annotated 2D keypoints and binary segmentation masks for dogs. We opted to take source images from the existing Stanford Dog Dataset [**?**], which consists of 20,580 dog images taken "in the wild" and covers 120 dog breeds. The dataset contains vast shape and pose variation between dogs, as well as nuisance factors such as self/environmental occlusion, interaction with humans/other animals and partial views. Figure **??** (left) shows samples from the new dataset.

We used Amazon Mechanical Turk to collect a binary silhouette mask and 20 keypoints per image: 3 per leg (knee, ankle, toe), 2 per ear (base, tip), 2 per tail (base, tip), 2 per face (nose and jaw). We can approximate the difficulty of the dataset by analysing the variance between 3 annotators at both the joint labelling and silhouette task. Figure **??** (right) illustrates typical per-joint variance in joint labelling. Further details of the data curation procedure are left to the supplementary materials.

In this section, we describe our process for obtaining keypoint and segmentation annotations for the Stanford Dog Dataset [**?**]. We submit the entire set of 20,580 dog images to the Amazon Mechnical Turk crowdsourcing platform to obtain a set of 20 keypoint and segmentation masks. We overlay 1 bounding box, provided with the original dataset, on the submitted images to identify the specific dog for the annotators to label. Each image was sent to 3 independent annotators for collecting keypoints and segmentation masks.

4.5.1 Keypoints.

To identify keypoints, workers were given a list of 20 keypoints to click: 2 per tail, 3 per leg, 2 per ear, nose and jaw. They were additionally asked to provide a visibility flag per point.

For each keypoint, we process the three clicks to yield a reliable coordinate. From the 3 clicks, we discard clicks that are further than a set tolerance from the mean. If at least 2 clicks remain, we take the mean coordinate as the accepted keypoint position. Otherwise, the point has not been reliably identified between workers, so we set the keypoint as invisible. As described in the main paper, we remove images from train and test splits which have fewer than 8 visible keypoints.

4.5.2 Segmentation.

For each image, each worker $w \in \{w_1, w_2, w_3\}$ submits a binary segmentation mask $\mathbf{A}^w \in \mathbb{R}^{H \times W}$. We request a re-labelling for any submissions which fail simple criteria, such as if the highlighted area is below a threshold number of pixels.

For each image, we generate the most likely segmentation by comparing submissions across workers. For any two workers w, w' we compute a correlation coefficient:

$$c_{w,w'} = \frac{\sum_i \sum_j \left[\mathbf{A}^w \odot \mathbf{A}^{w'} \right]_{i,j}}{\max_{p=\{w,w'\}} \sum_i \sum_j \mathbf{A}^p_{i,j}} \tag{4.26}$$

Where $\odot$ denotes the element-wise product of the matrices. We remove a worker's segmentation A^w if all correlation coefficients $c_{w,w'}$ are below a set threshold. The final binary mask is computed from the remaining submissions:

$$\hat{A}_{i,j} = \begin{cases} 1, & \text{if } \sum_w A^w_{i,j} > 1 \\ 0, & \text{otherwise} \end{cases} \tag{4.27}$$

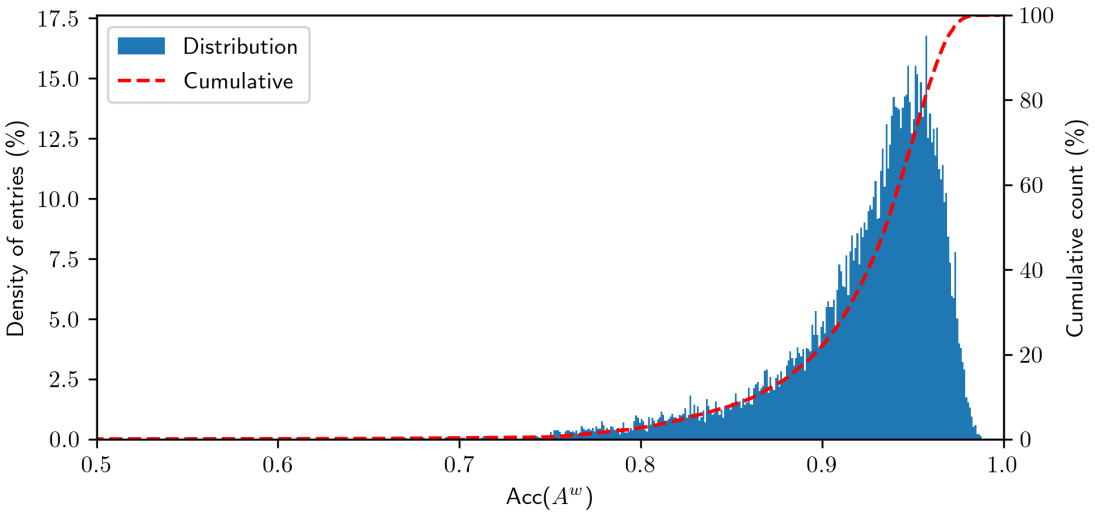


Fig. 4.5 Accuracy distribution of all submitted dog segmentations across the entire Stanford Dog Dataset.

We can also define the accuracy of a worker's segmentation, as the largest of their correlation coefficients: $\text{Acc}(A^w) = \max_{w' \neq w}\{c_{w,w'}\}$. Figure **??** shows the set of segmentation annotation accuracies over the entire labelled dataset.

4.6 Experiments

In this section we compare our method to competitive baselines. We begin by describing our new large-scale dataset of annotated dog images, followed by a quantitative and qualitative evaluation.

4.6.1 Evaluation protocol

Our evaluation is based on our new StanfordExtra dataset. In line with other methods which tackle "in-the-wild" 3D reconstruction of articulated subjects [**?** **?**], we filter images from the original dataset of 20,580 for which the majority of dog keypoints are invisible. We consider these images unsuitable for our full-body dog reconstruction task. We also remove images for which the consistency in keypoint/silhouette segmentations between the 3 annotators is below a set threshold. This leaves us with 8,476 images which we divide per-breed into an 80%/20% train and test split.

We consider two primary evaluation metrics. IoU is the intersection-over-union of the projected model silhouette compared to the ground truth annotation and indicates the quality of the reconstructed 3D shape. Percentage of Correct Keypoints (PCK) computes the percentage of joints which are within a normalized distance (based on square root of 2D silhouette area) to the ground truth locations, and evaluates the quality of reconstructed 3D pose. We also produce PCK results on various joint groups (legs, tail, ears, face) to compare the reconstruction accuracy for different parts of the dog model.

4.6.2 Training procedure

We train our model in two stages. The first omits the silhouette loss which we find can lead the network to unsatisfactory local minima if applied too early. With the silhouette loss turned off, we find it satisfactory to use the simple unimodal prior (and without EM) for this preliminary stage since there is no loss to specifically encourage a strong shape alignment. After this, we introduce the silhouette loss, the mixture prior and begin applying the expectation maximization updates over $M = 10$ clusters. We train the first stage for 250 epochs, the second stage for 150 and apply the EM step every 50 epochs. The entire training procedure takes 96 hours on a single P100 GPU.

Recall that the training objective for our end-to-end system for predicting SMBLD parameters consistent with a monocular dog input image is given by:

$$L_{\text{opt}} = L_{\text{joints}} + L_{\text{sil}} + L_{\text{pose}} + L_{\text{shape}} + L_{\text{mixture}} \tag{4.28}$$

Each loss term is weighted with a scalar λ and we train our method in two stages:

Stage 1.

We set $\lambda_{\text{joints}} = 10.0, \lambda_{\text{pose}} = 1.0, \lambda_{\text{shape}} = 1.0, \lambda_{\text{sil}} = 0.0, \lambda_{\text{mixture}} = 0.0$. We train this stage for 250 epochs, using the Adam optimizer, with learning rate set to 10^{-4}.

Stage 2.

In this stage, we introduce the silhouette loss to encourage a shape alignment between the projected model silhouette and the ground truth annotation. We set $\lambda_{\text{joints}} = 10.0, \lambda_{\text{pose}} = 0.5, \lambda_{\text{shape}} = 0.0, \lambda_{\text{sil}} = 100.0, \lambda_{\text{mixture}} = 0.1$. We train this stage for 150 epochs and run the described EM update step every $K = 15$ epochs. We selected to use $M = 10$ clusters based on a grid search over $M = 1, 5, 10, 25$ and comparing IoU. We again use the Adam optimizer, and set the learning rate to 10^{-5}.

4.6.3 Comparison to baselines

We first compare our method to various baseline methods. SMAL [**?**] is an approach which fits the 3D SMAL model using per-image energy minimization. Creatures Great and SMAL (CGAS) [**?**] is a three-stage method, which employs a joint predictor on silhouette renderings from synthetic 3D dogs, applies a genetic algorithm to clean predictions, and finally applies the SMAL optimizer to produce the 3D mesh.

At test-time both SMAL and CGAS rely on manually-provided segementation masks, and SMAL also relies on hand-clicked keypoints. In order to produce a fair comparison, we produce a set of *predicted* keypoints for StanfordExtra by training the Stacked Hourglass Network [**?**] with 8 stacks and 1 block, and *predicted* segmentation masks using DeepLab v3+ [**?**]. The Stacked Hourglass Network achieves 71.4% PCK score, DeepLab v3+ achieves 83.4% IoU score and the CGAS joint predictor achieves 41.8% PCK score.

Method	Kps	Seg	IoU	PCK				
				Avg	Legs	Tail	Ears	Face
SMAL [**?**]	Pred	Pred	67.9	67.1	65.7	79.5	54.9	87.4
SMAL	GT	GT	69.2	72.6	69.9	**92.0**	58.6	**96.9**
SMAL	GT	Pred	68.6	72.6	70.2	91.5	58.1	**96.9**
SMAL	Pred	GT	68.5	67.4	66.0	79.9	55.0	88.2
CGAS [**?**]	CGAS	Pred	62.4	43.7	46.5	64.1	36.5	21.4
CGAS	CGAS	GT	63.1	43.6	46.3	64.2	36.3	21.6
SMAL + scaling	Pred	Pred	69.3	69.6	69.4	79.3	56.5	87.6
SMAL + scaling + new prior	Pred	Pred	70.7	71.6	71.5	80.7	59.3	88.0
Ours	—	—	**73.6**	**75.7**	**75.0**	77.6	**69.9**	90.0

Table 4.1 **Baseline comparisons.** Both PCK and silhouette IOU scores are shown for SOTA methods under varying conditions. A combination of both ground truth (GT) and predicted (Pred) keypoints/segmentations using hourglass network and deeplab respectively. For the CGAS method we also test using their keypoint predictor (CGAS). The addition of scaling and new prior are shown to improve the original SMAL method.

Table **??** and Figure **??** show the comparison against competitive methods. For full examination, we additionally provide results for SMAL and CGAS in the scenario that ground-truth keypoints and/or segmentations are available at test time.

The results show our end-to-end method outperforms the competitors when they are provided with predicted keypoints/segmentations (white rows). Our method therefore achieves a new state of the art on this 3D reconstruction task. In addition, we show our method achieves improved average IoU/PCK scores than competitive methods, even when they are provided ground truth annotations at test time (grey rows). We also demonstrate wider applicability of two contributions from our work (scale parameters and improved prior) by showing improved performance of the SMAL method when these are incorporated. Finally, our model's test-time speed is significantly faster than the competitors as it does not require an optimizer.

4.6.4 Generalization to unseen dataset

Table **??** shows an experiment to compare how well our model generalizes to a new data domain. We test our model against the SMAL [**?**] method (using predicted keypoints and segmentations as above for fairness) on the recent Animal Pose dataset [**?**]. The data preparation process is the same as for StanfordExtra and no fine-tuning was used for either method. We achieve good results in this unseen domain and still improve over the SMAL optimizer.

4.6.5 Ablation study

We also produce a study in which we ablate individual components of our method and examine the effect on the PCK/IoU performance. We evaluate three variants: (1) **Ours w/o EM** that omits EM

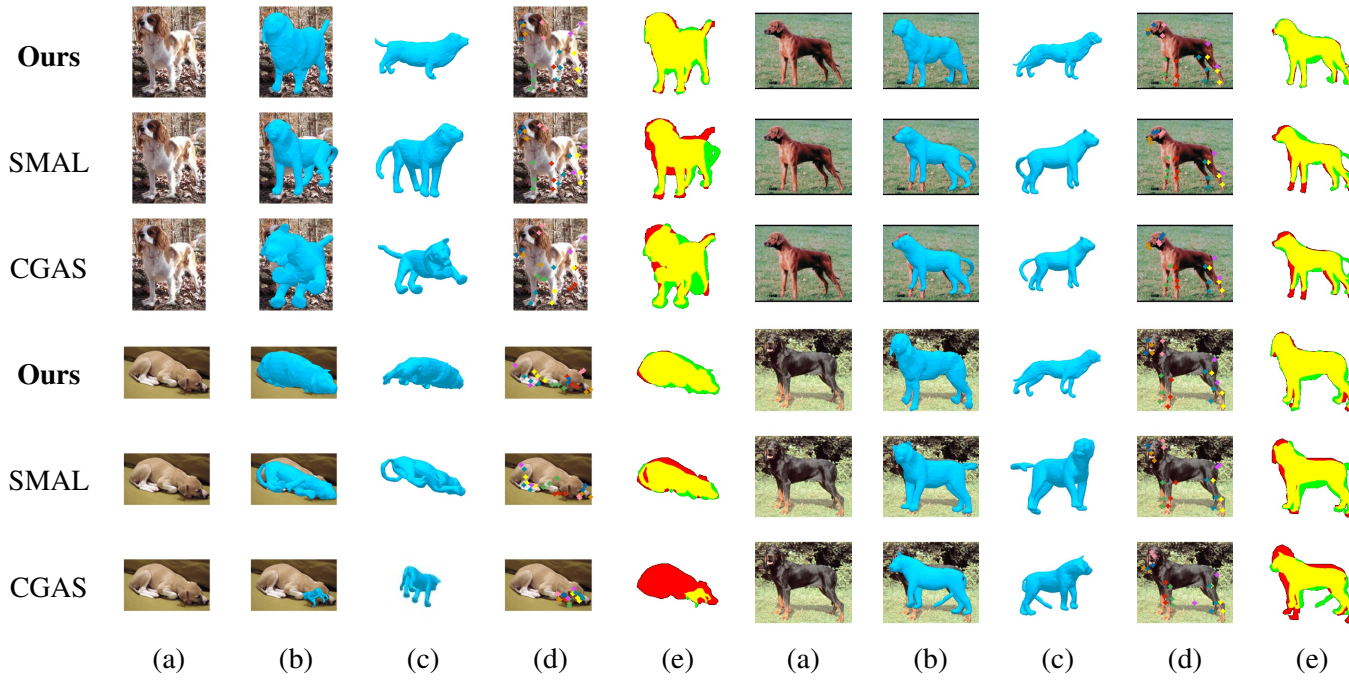


Fig. 4.6 **Qualitiative comparison to SOTA.** Row 1: **Ours**, Row 2: SMAL [**?**], Row 3: CGAS [**?**]. (a) input image, (b) predicted 3D mesh, (c) canonical view 3D mesh, (d) reprojected model joints and (e) silhouette reprojection error.

updates, (2) **Ours w/o MoG** which replaces our mixture shape prior with a unimodal prior, (3) **Ours w/o Scale** which removes the scale parameters.

The results in Table **??** indicate that each individual component has a positive impact on the overall method performance. In particular, it can be seen that the inclusion of the EM and Mixture of Gaussians prior leads to an improvement in IoU, suggesting that the shape prior refinements steps help the model accurately fit the exact dog shape. Interestingly, we notice that adding the Mixture of Gaussians prior but omitting EM steps slightly hinders performance, perhaps due to an sub-optimal initialization for the M clusters. However, we find adding EM updates to the Mixture of Gaussian model improves all metrics except the ear keypoint accuracy. We observe the error here is caused

Method	IoU	PCK				
		Avg	Legs	Tail	Ears	Face
SMAL [**?**]	63.6	69.1	60.9	83.5	75.0	93.0
Ours	**66.9**	**73.8**	**65.1**	**85.6**	**84.0**	**93.6**

Table 4.2 **Animal Pose dataset [?]**. Evaluation on recent Animal Pose dataset with no fine-tuning to our method nor joint/silhouette predictors used for SMAL.

Method	IoU	PCK				
		Avg	Legs	Tail	Ears	Face
Ours	**73.6**	**75.7**	**75.0**	**77.6**	69.9	90.0
−EM	67.7	74.6	72.9	75.2	**72.5**	88.3
−MoG	68.0	74.9	74.3	73.3	70.0	**90.2**
−Scale	67.3	72.6	72.9	75.3	62.3	89.1

Table 4.3 **Ablation study.** Evaluation with the following contributions removed: (a) EM updates, (b) Mixture Shape Prior, (c) SMBLD scale parameters.

by the our shape prior learning slightly imprecise shapes for dogs with extremely "floppy" ears. Although there is good silhouette coverage for these regions, the fact our model has only a single articulation point per ear causes a lack of flexibility that results in occasionally misplaced ear tips for these instances. This could be improved in future work by adding additional model joints to the ear. Finally, we find the increased model flexibility afforded by the SMBLD scale parameters have a positive effect on IoU/PCK scores.

4.6.6 Qualitative evaluation

Figure **??** shows a range of example system outputs when tested on range of StanfordExtra and Animal Pose [**?**] dogs with varying pose and shape and in challenging conditions. Note that only StanfordExtra is used for training.

4.7 Conclusions

This paper presents an end-to-end method for automatic, monocular 3D dog reconstruction. We achieve this using only weak 2D supervision, provided by our novel StanfordExtra dataset. Further, we show we can learn a more detailed shape prior by tuning a gaussian mixture during model training and this leads to improved reconstructions. We also show our method improves over competitive baselines, even when they are given access to ground truth data at test time.

Future work should involve tackling some failure cases of our system, for example handling multiple overlapping dogs or dealing with heavy motion blur. Other areas for research include extending our EM formulation to handle video input to take advantage of multi-view shape constraints, and transferring knowledge accumulated through training on StanfordExtra dogs to other species.

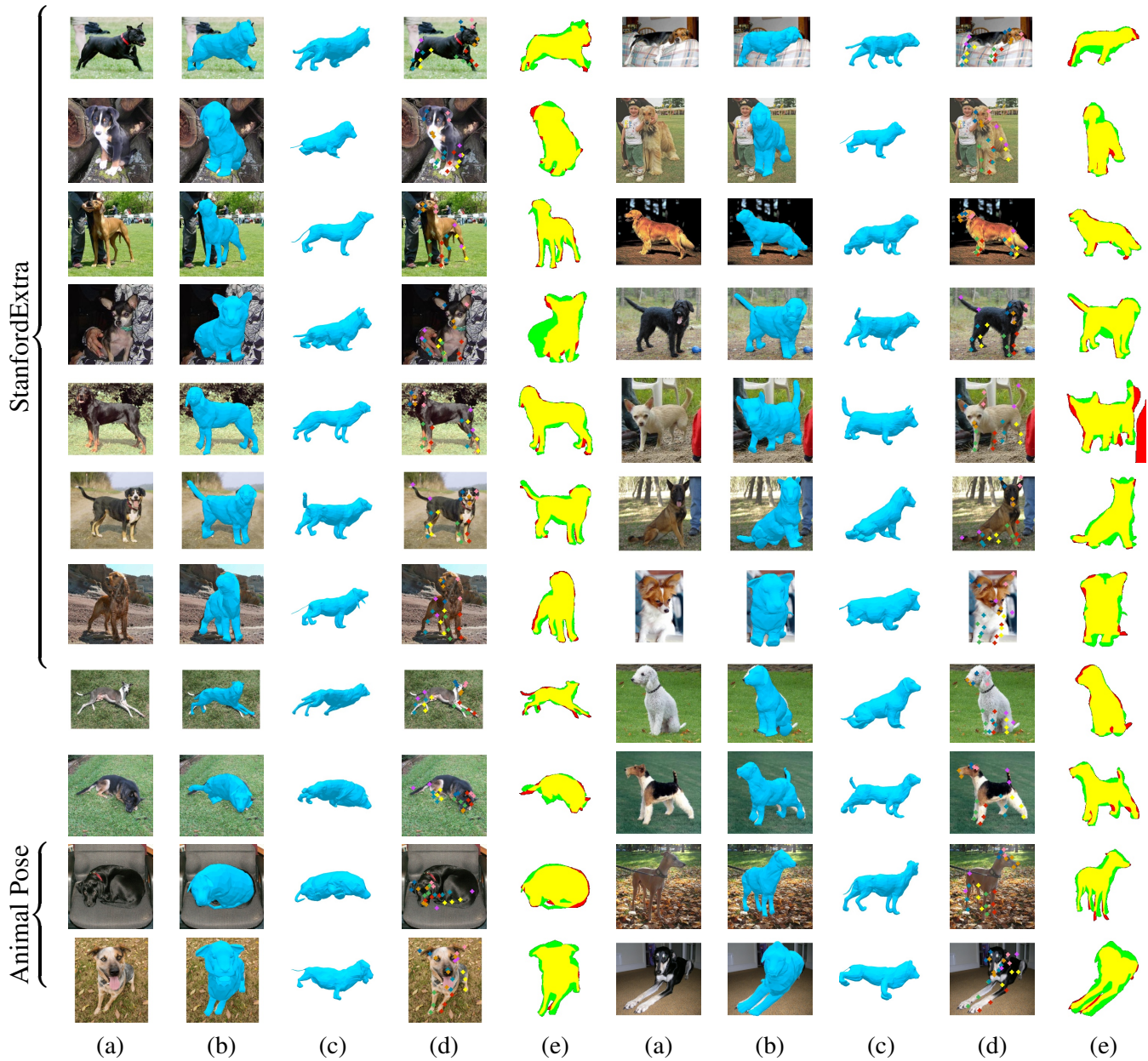


Fig. 4.7 **Qualitative results on StanfordExtra and Animal Pose [?].** For each sample we show: (a) input image, (b) predicted 3D mesh, (c) canonical view 3D mesh, (d) reprojected model joints and (e) silhouette reprojection error.

Chapter 5

Handling Ambiguous Input with Multi-Output Learning

5.1 Introduction

We consider the problem of obtaining dense 3D reconstructions of humans from single and partially occluded views. In such cases, the visual evidence is usually insufficient to identify a 3D reconstruction uniquely, so we aim at recovering several plausible reconstructions compatible with the input data. We suggest that ambiguities can be modelled more effectively by parametrizing the possible body shapes and poses via a suitable 3D model, such as SMPL for humans. We propose to learn a multi-hypothesis neural network regressor using a best-of-M loss, where each of the M hypotheses is constrained to lie on a manifold of plausible human poses by means of a generative model. We show that our method outperforms alternative approaches in ambiguous pose recovery on standard benchmarks for 3D humans, and in heavily occluded versions of these benchmarks.

We are interested in reconstructing 3D human pose from the observation of single 2D images. As humans, we have no problem in predicting, at least approximately, the 3D structure of most scenes, including the pose and shape of other people, even from a single view. However, 2D images notoriously [**?**] do not contain sufficient geometric information to allow recovery of the third dimension. Hence, single-view reconstruction is only possible in a probabilistic sense and the goal is to make the posterior distribution as sharp as possible, by learning a strong prior on the space of possible solutions.

Recent progress in single-view 3D pose reconstruction has been impressive. Methods such as HMR [**?**], GraphCMR [**?**] and SPIN [**?**] formulate this task as learning a deep neural network that maps 2D images to the parameters of a 3D model of the human body, usually SMPL [**?**]. These methods work well in general, but not always (**??**). Their main weakness is processing *heavily occluded images* of the object. When a large part of the object is missing, say the lower body of a sitting human, they output reconstructions that are often implausible. Since they can produce only one hypothesis as output, they very likely learn to approximate the mean of the posterior distribution,

Fig. 5.1 **Human mesh recovery in an ambiguous setting.** We propose a novel method that, given an occluded input image of a person, outputs the set of meshes which constitute plausible human bodies that are consistent with the partial view. The ambiguous poses are predicted using a novel *n*-quantized-best-of-*M* method.

which may not correspond to any plausible pose. Unfortunately, this failure modality is rather common in applications due to scene clutter and crowds.

In this paper, we propose a solution to this issue. Specifically, we consider the challenge of recovering 3D mesh reconstructions of complex articulated objects such as humans from highly ambiguous image data, often containing significant occlusions of the object. Clearly, it is generally impossible to reconstruct the object uniquely if too much evidence is missing; however, we can still predict a *set* containing all possible reconstructions (see **??**), making this set as small as possible. While ambiguous pose reconstruction has been previously investigated, as far as we know, this is the first paper that looks specifically at a deep learning approach for ambiguous reconstructions of the *full human mesh*.

Our primary contribution is to introduce a principled multi-hypothesis framework to model the ambiguities in monocular pose recovery. In the literature, such multiple-hypotheses networks are often trained with a so-called *best-of-M* loss — namely, during training, the loss is incurred only by the best of the M hypothesis, back-propagating gradients from that alone [**?**]. In this work we opt for the *best-of-M* approach since it has been show to outperform alternatives (such as variational auto-encoders or mixture density networks) in tasks that are similar to our 3D human pose recovery, and which have constrained output spaces [**?**].

A major drawback of the *best-of-M* approach is that it only guarantees that *one* of the hypotheses lies close to the correct solution; however, it says nothing about the plausibility, or lack thereof, of the *other* $M-1$ hypotheses, which can be arbitrarily 'bad'.[1] Not only does this mean that most of the hypotheses may be uninformative, but in an application we are also unable to tell *which* hypothesis

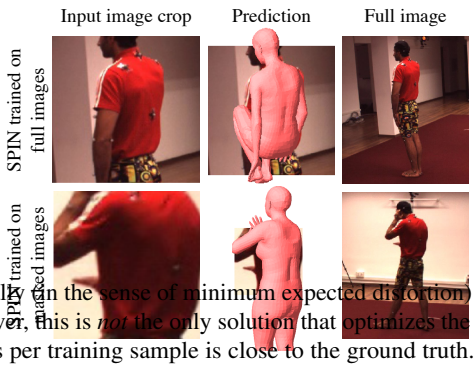


Fig. 5.2 **Top**: Pretrained SPIN model tested on an ambiguous example, **Bottom**: SPIN model after fine-tuning to ambiguous examples. Note the network tends to regress to the mean over plausible poses, shown by predicting the missing legs vertically downward — arguably the average position over the training dataset.

[1] Theoretically, best-of-M can minimize its loss by quantizing optimally (in the sense of minimum expected distortion) the posterior distribution, which would be desirable for coverage. However, this is *not* the only solution that optimizes the best-of-M training loss, as in the end it is sufficient that *one* hypothesis per training sample is close to the ground truth. In fact, this is exactly what happens; for instance, during training hypotheses in best-of-M are known to easily become degenerate and 'die off', a clear symptom of this problem.

should be used, and we might very well pick a 'bad' one. This has also a detrimental effect during learning because it makes gradients sparse as prediction errors are back-propagated only through one of the M hypotheses for each training image.

In order to address these issues, our first contribution is a *hypothesis reprojection loss* that forces each member of the multi-hypothesis set to correctly reproject to 2D image keypoint annotations. The main benefit is to constrain the *whole* predicted set of meshes to be consistent with the observed image, not just the best hypothesis, also addressing gradient sparsity.

Next, we observe that another drawback of the best-of-M pipelines is to be tied to a particular value of M, whereas in applications we are often interested in tuning the number of hypothesis considered. Furthermore, minimizing the reprojection loss makes hypotheses geometrically consistent with the observation, but not necessarily likely. Our second contribution is thus to improve the flexibility of best-of-M models by allowing them to output any smaller number $n < M$ of hypotheses while at the same time making these hypotheses *more representative of likely* poses. The new method, which we call n-quantized-best-of-M, does so by quantizing the best-of-M model to output weighed by a *explicit pose prior*, learned by means of normalizing flows.

To summarise, our key contributions are as follows. First, we deal with the challenge of 3D mesh reconstruction for articulated objects such as humans in *ambiguous* scenarios. Second, we introduce a *n-quantized-best-of-M* mechanism to allow best-of-M models to generate an arbitrary number of $n < M$ predictions. Third, we introduce a mode-wise re-projection loss for multi-hypothesis prediction, to ensure that predicted hypotheses are *all* consistent with the input.

Empirically, we achieve state-of-the-art monocular mesh recovery accuracy on Human36M, its more challenging version augmented with heavy occlusions, and the 3DPW datasets. Our ablation study validates each of our modelling choices, demonstrating their positive effect.

5.2 Related work

There is ample literature on recovering the pose of 3D models from images. We break this into five categories: methods that reconstruct 3D points directly, methods that reconstruct the parameters of a 3D model of the object via optimization, methods that do the latter via learning-based regression, hybrid methods and methods which deal with uncertainty in 3D human reconstruction.

5.2.1 Reconstructing 3D body points without a model.

Several papers have focused on the problem of estimating 3D body points from 2D observations [**?** **?** **?** **?** **?**]. Of these, **?**] introduced a particularly simple pipeline based on a shallow neural network. In this work, we aim at recovering the full 3D surface of a human body, rather than only lifting sparse keypoints.

5.2.2 Fitting 3D models via direct optimization.

Several methods *fit* the parameters of a 3D model such as SMPL [**?**] or SCAPE [**?**] to 2D observations using an optimization algorithm to iteratively improve the fitting quality. While early approaches such as [**?** **?**] required some manual intervention, the SMPLify method of **?**] was perhaps the first to fit SMPL to 2D keypoints fully automatically. SMPL was then extended to use silhouette, multiple views, and multiple people in [**?** **?** **?**]. Recent optimization methods such as [**?** **?** **?**] have significantly increased the scale of the models and data that can be handled.

5.2.3 Fitting 3D models via learning-based regression.

More recently, methods have focused on regressing the parameters of the 3D models directly, *in a feed-forward manner*, generally by learning a deep neural network [**?** **?** **?** **?** **?**]. Due to the scarcity of 3D ground truth data for humans in the wild, most of these methods train a deep regressor using a mix of datasets with 3D and 2D annotations in form of 3D MoCap markers, 2D keypoints and silhouettes. Among those, HMR of **?**] and GraphCMR of **?**] stand out as particularly effective.

5.2.4 Hybrid methods.

Other authors have also combined optimization and learning-based regression methods. In most cases, the integration is done by using a deep regressor to initialize the optimization algorithm [**?** **?** **?** **?** **?**]. However, recently **?**] has shown strong results by integrating the optimization loop in learning the deep neural network that performs the regression, thereby exploiting the weak cues available in 2D keypoints.

5.2.5 Modelling ambiguities in 3D human reconstruction.

Several previous papers have looked at the problem of modelling ambiguous 3D human pose reconstructions. Early work includes **?**], **?**] and **?**].

More recently, **?**] learn a prior over human skeleton joint angles (but not directly a prior on the SMPL parameters) from a MoCap dataset. **?**] use the Mixture Density Networks model of [**?**] to capture ambiguous 3D reconstructions of sparse human body keypoints directly in physical space. **?**] learn a conditional variational auto-encoder to model ambiguous reconstructions as a posterior distribution; they also propose two scoring methods to extract a single 3D reconstruction from the distribution.

?] tackle the problem of video 3D reconstruction in the presence of occlusions, and show that temporal cues can be used to disambiguate the solution. While our method is similar in the goal of correctly handling the prediction uncertainty, we differ by applying our method to predicting *full mesh* of the human body. This is arguably a more challenging scenario due to the increased complexity of the desired 3D shape.

Finally, some recent concurrent works also consider building priors over 3D human pose using normalizing flows. **?**] release a prior for their new GHUM/GHUML model, and **?**] build a prior on SMPL joint angles to constrain their weakly-supervised network. Our method differs as we learn our prior on 3D SMPL joints.

5.3 Preliminaries

Before discussing our method, we describe the necessary background, starting from SMPL.

5.3.1 SMPL.

SMPL is a model of the human body parameterized by axis-angle rotations $\theta \in \mathbb{R}^{69}$ of 23 body joints, the shape coefficients $\beta \in \mathbb{R}^{10}$ modelling shape variations, and a global rotation $\gamma \in \mathbb{R}^3$. SMPL defines a *skinning function* $S : (\theta, \beta, \gamma) \mapsto V$ that maps the body parameters to the vertices $V \in \mathbb{R}^{6890 \times 3}$ of a 3D mesh.

5.3.2 Predicting the SMPL parameters from a single image.

Given an image $\mathbf{I}$ containing a person, the goal is to recover the SMPL parameters (θ, β, γ) that provide the best 3D reconstruction of it. Existing algorithms [**?**] cast this as learning a deep network $G(I) = (\theta, \beta, \gamma, t)$ that predicts the SMPL parameters as well as the translation $t \in \mathbb{R}^3$ of the perspective camera observing the person. We assume a fixed set of camera parameters. During training, the camera is used to constrain the reconstructed 3D mesh and the annotated 2D keypoints to be consistent. Since most datasets only contain annotations for a small set of keypoints ([**?**] is an exception), and since these keypoints do not correspond directly to any of the SMPL mesh vertices, we need a mechanism to translate between them. This mechanism is a fixed linear regressor $J : V \mapsto X$ that maps the SMPL mesh vertices $V = S(G(I))$ to the 3D locations $X = J(V) = J(S(G(I)))$ of the K joints. Then, the projections $\pi_t(X)$ of the 3D joint positions into image $\mathbf{I}$ can be compared to the available 2D annotations.

5.3.3 Normalizing flows.

The idea of normalizing flows (NF) is to represent a complex distribution $p(X)$ on a random variable X as a much simpler distribution $p(z)$ on a transformed version $z = f(X)$ of X. The transformation f is learned so that $p(z)$ has a fixed shape, usually a Normal $p(z) \sim \mathcal{N}(0, 1)$. Furthermore, f itself must

be *invertible* and *smooth*. In this paper, we utilize a particular version of NF dubbed RealNVP [**?**]. A more detailed explanation of NF and RealNVP has been deferred to the supplementary.

5.3.4 Method

We start from a neural network architecture that implements the function $G(I) = (\theta, \beta, \gamma, t)$ described above. As shown in SPIN [**?**], the HMR [**?**] architecture attains state-of-the-art results for this task, so we use it here. However, the resulting regressor $G(I)$, given an input image I, can only produce a single unique solution. In general, and in particular for cases with a high degree of reconstruction ambiguity, we are interested in predicting *set* of plausible 3D poses rather than a single one. We thus extend our model to explicitly produce a set of M different hypotheses $G_m(I) = (\theta_m, \beta_m, \gamma_m, t_m)$, $m = 1, \ldots, M$. This is easily achieved by modifying the HMR's final output layer to produce a tensor M times larger, effectively stacking the hypotheses. In what follows, we describe the learning scheme that drives the monocular predictor G to achieve an optimal coverage of the plausible poses consistent with the input image. Our method is summarized in **??**.

5.3.5 Learning with multiple hypotheses

For learning the model, we assume to have a training set of N images $\{I_i\}_{i=1,\ldots,N}$, each cropped around a person. Furthermore, for each training image I_i we assume to know (1) the 2D location Y_i of the body joints (2) their 3D location X_i, and (3) the ground truth SMPL fit $(\theta_i, \beta_i, \gamma_i)$. Depending on the set up, some of these quantities can be inferred from the others (e.g. we can use the function J to convert the SMPL parameters to the 3D joints X_i and then the camera projection to obtain Y_i).

5.3.6 Best-of-M loss.

Given a single input image, our network predicts a set of poses, where at least one should be similar to the ground truth annotation X_i. This is captured by the best-of-M loss [**?**]:

$$\mathcal{L}_{\text{best}}(J, G; m^*) = \frac{1}{N} \sum_{i=1}^{N} \left\| X_i - \hat{X}^{m_i^*}(I_i) \right\|, \;\; m_i^* = \underset{m=1,\ldots,M}{\operatorname{argmin}} \left\| X_i - \hat{X}^{m}(I_i) \right\|, \tag{5.1}$$

where $\hat{X}^m(I_i) = J(G_m(V(I_i)))$ are the 3D joints estimated by the m-th SMPL predictor $G_m(I_i)$ applied to image I_i. In this way, only the best hypothesis is steered to match the ground truth, leaving the other hypotheses free to sample the space of ambiguous solutions. During the computation of this loss, we also extract the best index m_i^* for each training example.

5.3.7 Limitations of best-of-M.

As noted in **??**, best-of-M only guarantees that one of the M hypotheses is a good solution, but says nothing about the other ones. Furthermore, in applications we are often interested in modulating the

number of hypotheses generated, but the best-of-M regressor $G(I)$ only produces a fixed number of output hypothesis M, and changing M would require retraining from scratch, which is intractable.

We first address these issues by introducing a method that allows us to train a best-of-M model for a large M once and leverage it later to generate an arbitrary number of $n < M$ hypotheses without the need of retraining, while ensuring that these are good representatives of likely body poses.

5.3.8 n-quantized-best-of-M

Formally, given a set of M predictions $\hat{\mathcal{X}}^M(I) = \{\hat{X}^1(I), ..., \hat{X}^M(I)\}$ we seek to generate a smaller n-sized set $\bar{\mathcal{X}}^n(I) = \{\bar{X}^1(I), ..., \bar{X}^n(I)\}$ which preserves the information contained in $\hat{\mathcal{X}}^M$. In other words, $\bar{\mathcal{X}}^n$ *optimally quantizes* $\hat{\mathcal{X}}^M$. To this end, we interpret the output of the best-of-M model as a set of choices $\hat{\mathcal{X}}^M(I)$ for the possible pose. These poses are of course not all equally likely, but it is difficult to infer their probability from (**??**). We thus work with the following approximation. We consider the prior $p(X)$ on possible poses (defined in the next section), and set:

$$p(X|I) = p(X|\hat{\mathcal{X}}^M(I)) = \sum_{i=1}^{M} \delta(X - \hat{X}^i(I)) \frac{p(\hat{X}^i(I))}{\sum_{k=1}^{M} p(\hat{X}^k(I))}. \tag{5.2}$$

This amounts to using the best-of-M output as a conditioning *set* (i.e. an unweighted selection of plausible poses) and then use the prior $p(x)$ to weight the samples in this set. With the weighted samples, we can then run K-means [**?**] to further quantize the best-of-M output while minimizing the quantization energy E:

$$E(\bar{\mathcal{X}}|\hat{\mathcal{X}}) = \mathbb{E}_{p(X|I)} \left[\min_{\{\bar{X}^1,...,\bar{X}^n\}} \|X - \bar{X}^j\|^2 \right] = \sum_{i=1}^{M} \frac{p(\hat{X}^i(I))}{\sum_{k=1}^{M} p(\hat{X}^k(I))} \min_{\{\bar{X}^1,...,\bar{X}^n\}} \|\hat{X}^i(I) - \bar{X}^j\|^2. \tag{5.3}$$

This can be done efficiently on GPU — for our problem, K-Means consumes less than 20% of the execution time of the entire forward pass of our method.

5.3.9 Learning the pose prior with normalizing flows.

In order to obtain $p(X)$, we propose to learn a normalizing flow model in form of the RealNVP network f described in **??** and the supplementary. RealNVP optimizes the log likelihood $\mathcal{L}_{\text{nf}}(f)$ of training ground truth 3D skeletons $\{X_1, ...X_N\}$ annotated in their corresponding images $\{I_1, ..., I_N\}$:

$$\mathcal{L}_{\text{nf}}(f) = -\frac{1}{N} \sum_{i=1}^{N} \log p(X_i) = -\frac{1}{N} \sum_{i=1}^{N} \left(\log \mathcal{N}(f(X_i)) - \sum_{l=1}^{L} \log \left| \frac{df_l(X_{li})}{dX_{li}} \right| \right). \tag{5.4}$$

5.3.10 2D re-projection loss.

Since the best-of-M loss optimizes a single prediction at a time, often some members of the ensemble $\hat{\mathcal{X}}(I)$ drift away from the manifold of plausible human body shapes, ultimately becoming 'dead'

predictions that are never selected as the best hypothesis m^*. In order to prevent this, we further utilize a re-projection loss that acts across all hypotheses for a given image. More specifically, we constrain the set of 3D reconstructions to lie on projection rays passing through the 2D input keypoints with the following *hypothesis re-projection loss*:

$$\mathcal{L}_{\text{ri}}(J,G) = \frac{1}{N}\sum_{i=1}^{N}\sum_{m=1}^{M}\left\|Y_i - \pi_{t_i}(\hat{X}^m(I))\right\|. \tag{5.5}$$

Note that many of our training images exhibit significant occlusion, so Y may contain invisible or missing points. We handle this by masking $\mathcal{L}_{\text{ri}}$ to prevent these points contributing to the loss.

SMPL loss. The final loss terms, introduced by prior work [**? ? ?**], penalize deviations between the predicted and ground truth SMPL parameters. For our method, these are only applied to the best hypothesis m_i^* found above:

$$\mathcal{L}_\theta(G;m^*) = \frac{1}{N}\sum_{i=1}^{N}\|\theta_i - G_{\theta,m_i^*}(I_i)\|;\ \mathcal{L}_V(G;m^*) = \frac{1}{N}\sum_{i=1}^{N}\|S(\theta_i,\beta_i,\gamma_i) - S(G_{(\theta,\beta,\gamma),m_i^*}(I_i))\| \tag{5.6}$$

$$\mathcal{L}_\beta(G;m^*) = \frac{1}{N}\sum_{i=1}^{N}\|\beta_i - G_{\beta,m_i^*}(I_i)\|;\ \mathcal{L}_{\text{rb}}(G;m^*) = \frac{1}{N}\sum_{i=1}^{N}\|Y_i - \pi_{t_i}(\hat{X}^{m_i^*}(I_i))\| \tag{5.7}$$

Note here we use $\mathcal{L}_{\text{rb}}$ to refer to a 2D re-projection error between the best hypothesis and ground truth 2D points Y_i. This differs from the earlier loss $\mathcal{L}_{\text{ri}}$, which is applied across all modes to enforce consistency to the visible *input* points. Note that we could have used **????** to select the best hypothesis m_i^*, but it would entail an unmanageable memory footprint due to the requirement of SMPL-meshing for every hypothesis before the best-of-M selection.

5.3.11 Overall loss.

The model is thus trained to minimize:

$$\begin{aligned}\mathcal{L}(J,G) = {} & \lambda_{\text{ri}}\mathcal{L}_{\text{ri}}(J,G) + \lambda_{\text{best}}\mathcal{L}_{\text{best}}(J,G;m^*) + \lambda_\theta\mathcal{L}_\theta(J,G;m^*) \\ & + \lambda_\beta\mathcal{L}_\beta(J,G;m^*) + \lambda_{\text{V}}\mathcal{L}_V(J,G;m^*) + \lambda_{\text{rb}}\mathcal{L}_{\text{rb}}(J,G;m^*)\end{aligned} \tag{5.8}$$

where m^* is given in **??** and $\lambda_{\text{ri}}, \lambda_{\text{best}}, \lambda_\theta, \lambda_\beta, \lambda_{\text{V}}, \lambda_{\text{rb}}$ are weighing factors. We use a consistent set of SMPL loss weights across all experiments $\lambda_{\text{best}} = 25.0, \lambda_\theta = 1.0, \lambda_\beta = 0.001, \lambda_{\text{V}} = 1.0$, and set $\lambda_{\text{ri}} = 1.0$. Since the training of the normalizing flow f is independent of the rest of the model, we train f separately by optimizing $\mathcal{L}_{\text{nf}}$ with the weight of $\lambda_{\text{nf}} = 1.0$. Samples from our trained normalizing flow are shown in **??**

Fig. 5.3 **Example samples from the normalizing flow** $f : X \mapsto z$; $p(z) \sim \mathcal{N}(0,1)$, trained on a dataset of ground truth 3D SMPL control skeletons $\{X_1, ..., X_N\}$.

Dataset	Quantization n	1		5		10		25	
	Metric	MPJPE	RE	MPJPE	RE	MPJPE	RE	MPJPE	RE
H36M	HMR [**?**]	—	56.8	—	—	—	—	—	—
	GraphCMR [**?**]	71.9	50.1	—	—	—	—	—	—
	SPIN [**?**]	62.2	41.8	—	—	—	—	—	—
	SMPL-MDN	64.4	44.8	61.8	43.3	61.3	43.0	61.1	42.7
	SMPL-CVAE	70.1	46.7	68.9	46.4	68.6	46.3	68.1	46.2
	Ours	**61.5**	**41.6**	**59.8**	**42.0**	**59.2**	**42.2**	**58.2**	**42.2**
3DPW	HMR [**?**]	—	81.3	—	—	—	—	—	—
	GraphCMR [**?**]	—	70.2	—	—	—	—	—	—
	SPIN [**?**]	96.9	**59.3**	—	—	—	—	—	—
	SMPL-MDN	105.8	64.7	96.9	61.2	95.9	60.7	94.9	60.1
	SMPL-CVAE	96.3	61.4	93.7	60.7	92.9	60.5	92.0	60.3
	Ours	**93.8**	59.9	**82.2**	**57.1**	**79.4**	**56.6**	**75.8**	**55.6**
AH36M	SMPL-MDN	113.9	74.7	98.0	70.8	95.1	69.9	91.5	69.5
	SMPL-CVAE	114.5	76.5	111.5	75.7	110.6	75.4	109.7	75.1
	Ours	**103.6**	**67.8**	**96.4**	**67.1**	**93.5**	**66.0**	**90.0**	**64.2**
A3DPW	SMPL-MDN	159.7	82.8	154.6	83.0	149.6	80.7	122.1	76.6
	SMPL-CVAE	156.6	80.2	154.5	79.9	153.9	79.8	153.1	79.8
	Ours	**149.6**	**78.5**	**125.6**	**74.4**	**116.7**	**73.7**	**107.8**	**72.1**

Table 5.1 **Monocular multi-hypothesis human mesh recovery** comparing our approach to two multi-hypothesis baselines (SMPL-CVAE, SMPL-MDN) and state-of-the-art single mode evaluation models [**? ? ?**] on Human3.6m (H36M), its ambiguous version AH36M, on 3DPW and its ambiguous version A3DPW.

5.4 Experiments

In this section we compare our method to several strong baselines. We start by describing the datasets and the baselines, followed by a quantitative and a qualitative evaluation.

5.4.1 Datasets

We begin by testing on a recent animal dataset of dogs. Since this dataset is limited, we additionally run an evaluation on a Human3.6m dataset which is considerably more varied and challenging.

RGBD-Dog Dataset

Our evaluation begins with the recent RGBD-Dog dataset.

Quantization n		5		10		25	
Mode reproj.	Flow weight	MPJPE	RE	MPJPE	RE	MPJPE	RE
		86.4	57.9	84.0	57.5	79.0	56.3
	✓	84.1	**57.0**	81.9	56.7	77.8	55.8
✓		82.7	57.5	79.9	57.0	76.2	55.9
✓	✓	**82.2**	57.1	**79.4**	**56.6**	**75.8**	**55.6**

Table 5.2 **Ablation study on 3DPW** removing either the normalizing flow or the mode re-projection losses and reporting the change in performance.

Our evaluation focuses on the Human3.6m (**H36M**) [**?** **?**] and **3DPW** datasets [**?**].

Human3.6M

H36M is one of the largest datasets of humans annotated with 3D pose using MoCap sensors.

As common practice, we train on subjects S1, S5, S6, S7 and S8, and test on S9 and S11.

3D People in the Wild

3DPW is only used for evaluation and, following [**?**], we evaluate on its test set.

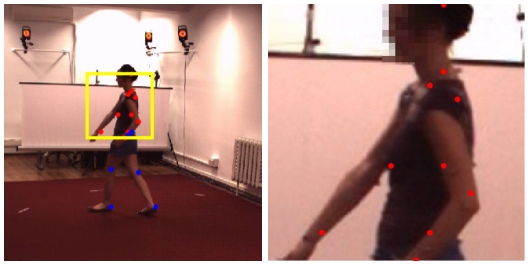


Fig. 5.4 Example image and corresponding annotation from the ambiguous H36M dataset **AH36M**. Best viewed in colour.

5.4.2 Evaluation Protocol

Our evaluation is consistent with [**?** **?**] - we report two metrics that compare the lifted dense 3D SMPL shape to the ground truth mesh: Mean Per Joint Position Error (**MPJPE**), Reconstruction Error (**RE**). For H36M, all errors are computed using an evaluation scheme known as "Protocol #2". Please refer to supplementary for a detailed explanation of MPJPE and RE.

5.4.3 Multipose metrics.

MPJPE and RE are traditional metrics that assume a single correct ground truth prediction for a given 2D observation. As mentioned above, such an assumption is rarely correct due to the inherent ambiguity of the monocular 3D shape estimation task. We thus also report MPJPE-n/RE-n an extension of MPJPE RE used in [**?**], that enables an evaluation of n different shape hypotheses. In more detail, to evaluate an algorithm, we allow it to output n possible predictions and, out of this set, we select the one that minimizes the MPJPE/RE metric. We report results for $n \in \{1, 5, 10, 25\}$.

5.4.4 Ambiguous H36M/3DPW (AH36M/A3DPW).

Since H36M is captured in a controlled environment, it rarely depicts challenging real-world scenarios such as body occlusions that are the main source of ambiguity in the single-view 3D shape estimation problem.

Hence, we construct an adapted version of H36M with synthetically-generated occlusions (**??**) by randomly hiding a subset of the 2D keypoints and re-computing an image crop around the remaining visible joints. Please refer to the supplementary for details of the occlusion generation process.

While 3DPW does contain real scenes, for completeness, we also evaluate on a noisy, and thus more challenging version (A3DPW) generated according to the aforementioned strategy.

5.4.5 Baselines

Our method is compared to two multi-pose prediction baselines. For fairness, both baselines extend the same (state-of-the-art) trunk architecture as we use, and all methods have access to the same training data.

SMPL-MDN follows [**?**] and outputs parameters of a mixture density model over the set of SMPL log-rotation pose parameters. Since a naïve implementation of the MDN model leads to poor performance ($\approx$ 200mm MPJPE-$n = 5$ on H36M), we introduced several improvements that allow optimization of the total loss **??**. **SMPL-CVAE**, the second baseline, is a conditional variational autoencoder [**?**] combined with our trunk network. SMPL-CVAE consists of an encoding network that maps a ground truth SMPL mesh V to a gaussian vector z which is fed together with an encoding of the image to generate a mesh V' such that $V' \approx V$. At test time, we sample n plausible human meshes by drawing $z \sim \mathcal{N}(0,1)$ to evaluate with MPJPE-*n*/RE-*n*. More details of both SMPL-CVAE and SMPL-MDN have been deferred to the supplementary material.

For completeness, we also compare to three more baselines that tackle the standard single-mesh prediction problem: HMR [**?**], GraphCMR [**?**], and SPIN [**?**], where the latter currently attain state-of-the-art performance on H36M/3DPW. All methods were trained on H36M [**?**], MPI-INF-3DHP [**?**], LSP [**?**], MPII [**?**] and COCO [**?**].

5.4.6 Results

?? contains a comprehensive summary of the results on all 3 benchmarks. Our method outperforms the SMPL-CVAE and SMPL-MDN in all metrics on all datasets. For SMPL-CVAE, we found that the encoding network often "cheats" during training by transporting all information about the ground truth, instead of only encoding the modes of ambiguity. The reason for a lower performance of SMPL-MDN is probably the representation of the probability in the space of log-rotations, rather in the space of vertices. Modelling the MDN in the space of model vertices would be more convenient due to being more relevant to the final evaluation metric that aggregates per-vertex errors, however, fitting such high-dimensional (dim=6890×3) Gaussian mixture is prohibitively costly.

Furthermore, it is very encouraging to observe that our method is also able to outperform the single-mode baselines [**?** **?** **?**] on the single mode MPJPE on both H36M and 3DPW. This comes as a surprise since our method has not been optimized for this mode of operation. The difference is more significant for 3DPW which probably happens because 3DPW is not used for training and, hence, the normalizing flow prior acts as an effective filter of predicted outlier poses. Qualitiative results are shown in **??**.

Ablation study.

We further conduct an ablative study on 3DPW that removes components of our method and measures the incurred change in performance. More specifically, we: 1) ablate the hypothesis reprojection loss; 2) set $p(X|I) = \text{Uniform}$ in **??**, effectively removing the normalizing flow component and executing unweighted K-Means in n-quantized-best-of-M. **??** demonstrates that removing both contributions decreases performance, validating our design choices.

5.5 Conclusions

In this work, we have explored a seldom visited problem of representing the set of plausible 3D meshes corresponding to a single ambiguous input image of a human. To this end, we have proposed a novel method that trains a single multi-hypothesis best-of-M model and, using a novel n-quantized-best-of-M strategy, allows to sample an arbitrary number $n < M$ of hypotheses.

Importantly, this proposed quantization technique leverages a normalizing flow model, that effectively filters out the predicted hypotheses that are unnatural. Empirical evaluation reveals performance superior to several strong probabilistic baselines on Human36M, its challenging ambiguous version, and on 3DPW. Our method encounters occasional failure cases, such as when tested on individuals with unusual shape (e.g. obese people), since we have very few of these examples in the training set. Tackling such cases would make for interesting and worthwhile future work.

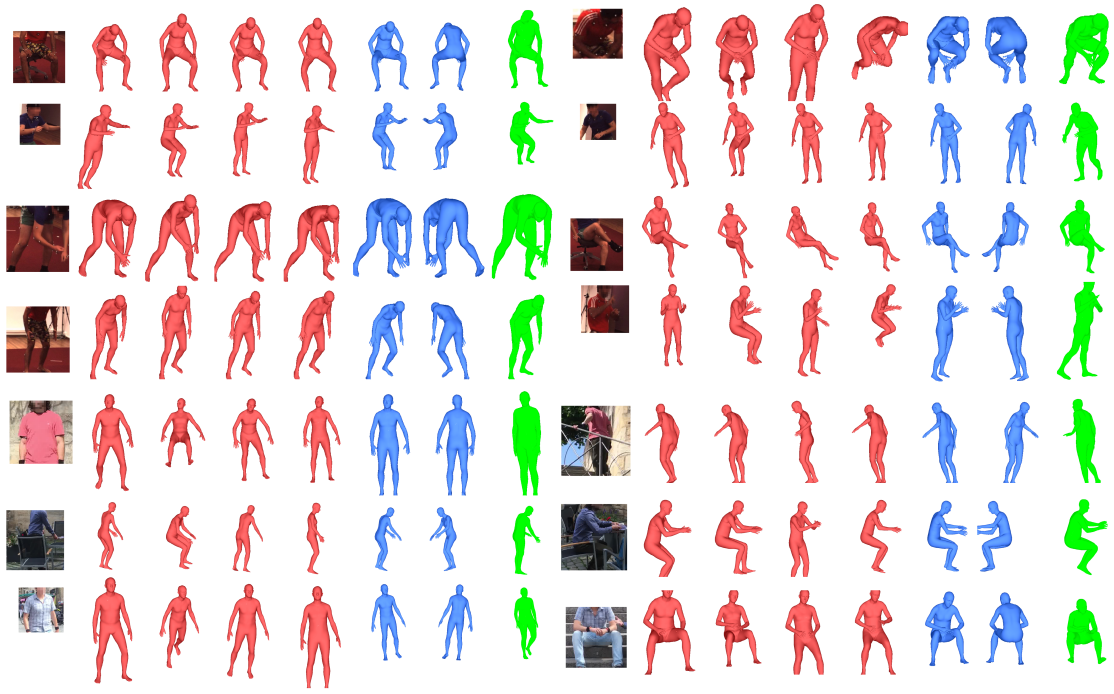


Fig. 5.5 **Qualitative results from $n = 5$ quantization on monocular mesh recovery on AH36m and A3DPW.** From left to right, each group of figures depicts the input ambiguous image, five network hypotheses with the closest to the ground truth in blue, and the ground truth pose in green.

Chapter 6

Conclusions

6.1 Discussion and Limitations

In this section I will conclude and discuss limitations

6.1.1 Discussion

Talk about meshes, radiance fields etc.

6.1.2 Applications in Animal Tracking

Discussion as to what GSK have been doing.

6.1.3 Future Work

What needs to happen etc.