Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics
Study program: Open Informatics

Rekonstrukce trojdimenzionálních modelů zvířat

Reconstructing 3D Models of Animals

MASTER THESIS

Author: Bc. Valeriia Iegorova
Supervisor: Doc. Ing. Tomáš Pajdla, Ph.D.
Submitted: January 2022

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Iegorova Valeriia** Personal ID number: **453535**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Open Informatics**

Specialisation: **Computer Vision and Image Processing**

II. Master's thesis details

Master's thesis title in English:

Reconstructing 3D Models of Animals

Master's thesis title in Czech:

Rekonstrukce trojdimenzionálních modelů zvířat

Guidelines:

1) Study the approaches to 3D reconstruction from images and 3D data relevant for reconstructing non-static animals [1,2,3,4,5].
2) Experiment with available 3D reconstruction pipelines and evaluate their suitability for your task [1].
3) Choose the most promising approach and adjust it to you task.
4) Demonstrate the results of the adjusted approach on real data.

Bibliography / sources:

[1] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, Michael J. Black – 3D Menagerie: Modeling the 3D Shape and Pose of Animals - 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
[2] Richard A. Newcombe, D. Fox, S.Seitz - DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time - 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
[3] Silvia Zuffi, Angjoo Kanazawa, T. Berger-Wolf, Michael J. Black - Three-D Safari: Learning to Estimate Zebra Pose, Shape, and Texture From Images "In the Wild" - 2019 IEEE/CVF International Conference on Computer Vision (ICCV)
[4] Aljaž Božič, Michael Zollhöfer, Christian Theobalt, Matthias Nießner - DeepDeform: Learning Non-rigid RGB-D Reconstruction with Semi-supervised Data - 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)
[5] Richard Hartley, Andrew Zisserman – Multiple View Geometry in Computer Vision – 2003 Cambirdge Univerersity Press

Name and workplace of master's thesis supervisor:

doc. Ing. Tomáš Pajdla, Ph.D., Applied Algebra and Geometry, CIIRC

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **12.02.2021** Deadline for master's thesis submission: **04.01.2022**

Assignment valid until: **30.09.2022**

doc. Ing. Tomáš Pajdla, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

In Prague, January 2022

Bc. Valeriia Iegorova

Acknowledgements

I would like to thank my friends and colleagues Mgr. Juraj Páll and Ing. Jakub Jirůtka for their support and for review of this work. I would like to express special thanks to Ing. Tomáš Borovička for giving me access to computational resources and supporting this research, and to MSc. Jan Bím PhD for his valuable comments. Also, I thank Dr. Silvia Zuffi, the author of the *SMAL* animal model, and Benjamin Biggs, the author of the *SMBDL* model, for discussing their research with me.

I would like to thank my family for their support, and additionally thank my mother for proof-reading the text of this diploma. Also, I thank my partner for supporting me during late stages of this research.

Bc. Valeriia Iegorova

Název práce:
Rekonstrukce trojdimenzionálních modelů zvířat

Autor: Bc. Valeriia Iegorova

Studijní program: Computer Vision and Digital Image
Obor: Open Informatics
Druh práce: Master thesis

Vedoucí práce: Doc. Ing. Tomáš Pajdla, Ph.D.

Abstrakt: Tato diplomová práce se zabývá problematikou 3D rekonstrukce zvířat, která by mohla mít řadu praktických využití v zemědělství nebo počítačové animaci. Poskytuje ucelený přehled literatury a pojednává o existujících metodách 3D rekonstrukce a jejich vhodnosti pro danou úlohu. Následně demonstruje výsledky nejvhodnějších současných end-to-end rekonstrukčních pipeline a navrhuje novou metodu nazvanou *SMAL4V*, která tyto pipeline rozšiřuje pro videosekvence. K tomuto účelu také představuje novou datovou sadu, *AnimalKey*, na které lze 3D rekonstrukční modely trénovat nebo porovnávat, a zpřístupňuje ji výzkumné komunitě.

Klíčová slova: 3D rekonstrukce, neuronové sítě, strojové učení

Title:
Reconstructing 3D Models of Animals

Author: Bc. Valeriia Iegorova

Abstract: This diploma thesis explores the problem of 3D animal reconstruction, which could have multiple practical applications in agriculture or computer animation. It provides a comprehensive literature overview, discussing existing 3D reconstruction methods and their suitability for the given task. It then demonstrates the results of the most suitable state-of-the-art end-to-end reconstruction pipelines and proposes a new method, called *SMAL4V*, which extends those pipeline to video sequences. For that, it also introduces a new dataset, *AnimalKey*, on which 3D reconstruction models can be trained or compared, and makes it available to the research community.
Key words: 3D reconstruction, artificial neural networks, machine learning

Contents

List of Acronyms

ANNs *Artificial Neural Networks*

CNNs *Convolutional Neural Networks*

RNNs *Recurrent Neural Networks*

DIFs *Deep Implicit Functions*

RANSAC *RANdom SAmple Consensus*

ARAP *As-Rigid-As-Possible*

PCK *Percentage of Correct Keypoints*

IoU *Intersection-over-Union*

OSVOS *One-Shot Video Object Segmentation*

LSTM *Long Short-Term Memory*

GRU *Gated Recurrent Unit*

ToF *Time-of-Flight*

SfM *Structure-from-Motion*

NRSfM *Non-Rigid Structure-from-Motion*

SCAPE *Shape Completion and Animation for PEople*

SMPL *Skinned Multi-Person Linear*
SMAL with Refinement

SMAL *Skinned Multi-Animal Linear*

SMALR *SMAL with Refinement*

SMALST *SMAL with learned Shape and Texture*

SMBLD *Skinned Multi-Breed Linear Model for Dogs*

SMAL4V *SMAL for Videos*

CGAS *Creatures Great and SMAL*

WLDO *Who Left the Dogs Out*

BADJA *Benchmark Animal Dataset of Joint Annotations*

GLoSS *Global/Local Stitched Shape*

HMR *Human Mesh Recovery*

VIBE ***V****ideo* ***I****nference for* ***B****ody Pose and Shape* ***E****stimation*

List of Figures

Chapter 1

Introduction

This thesis deals with the problem of 3D animal reconstruction and describes the issues that arise in modern reconstruction pipelines when applied to the articulated moving objects. It provides a comprehensive overview of the modern and historical reconstruction methods and discusses their applicability to the given task. Following the successes of the neural networks in multiple areas of computer vision, it explores an end-to-end learning-based family of the reconstruction pipelines in more details. Results of several such pipelines are demonstrated and a new method, called *SMAL4V*, is proposed, extending the existing end-to-end methods to the video sequences. The proposed architecture leverages temporal information to predict smoother meshes throughout the video.

Figure 1.1: A typical example of the point clouds obtained from the reconstruction of some static objects. Image taken from [1].

1.1 Motivation

The research of 3D reconstruction has been a popular topic in computer vision for several decades. It has a wide range of possible applications, including medicine [2], agriculture [3][4], archaeology [5], ecology [6], robotics [7], autonomous vehicles [8], motion capture [9] and many others. A typical 3D reconstruction pipeline takes a set of images of an object or scene of interest and returns its 3D representation, such as a point cloud, mesh or voxel model. Optionally, the resulting representation can be colored or texturized if required by the application. Depending on

the approach used, different numbers of cameras and input images are used for the reconstruction, with different prior knowledge incorporated in the pipeline.

In the case of static scenes, many ready-to-use 3D reconstruction solutions are available, such as open-source OpenMVG [10], COLMAP [11][12], LSD-SLAM [13], ORB-SLAM [14], TheiaSfM [15] and AliceVision [16][17], or commercial products, such as RealityCapture [18] from Capturing Reality or Agisoft Metashape [19]. However, few scenes in the real world are static, and most of them contain non-rigid, articulated or moving objects. Once the effects of the objects' movement or deformation become too strong, the geometrical assumptions, on which the mentioned solutions are based, are not satisfied anymore. This causes such pipelines to output highly defective models, with multiple artifacts caused by the violated geometrical constraints.

Animals, which are the main topic of this thesis, are not static, and their bodies are highly articulated. This makes them extremely difficult to reconstruct with modern reconstruction methods. Moreover, unlike humans, most of the animals are not collaborative, and cannot be scanned with high-end 3D equipment in a lab. As a result, a majority of the available solutions output a 3D model of a quality insufficient for any practical use.

An example of such a poor-quality reconstruction, produced by the commercial RealityCapture photogrammetry pipeline [18], is shown in Figure 1.2. It should be noted that multiple images of the animal were used to create this reconstruction, and the animal was barely moving for the most of the capturing process. But, despite the care taken during the image capturing, there are still multiple artifacts, which prevent the model from being used in any practical situation.

Figure 1.2: An example of the 3D reconstruction of a cow. A photogrammetry pipeline from RealityCapture [18] was used to create this example.

While a challenging problem, the 3D reconstruction of animals has many promising applications in industry. In computer graphics, it could facilitate the creation of the 3D animal models for games or animated movies. Instead of manual creation of the animals' animation by professional artists, which is both expensive and time-consuming, a reconstruction pipeline could be used on the video recordings of real animals to create many plausibly moving 3D models [20].

In agriculture, the 3D reconstruction of the animal's body could be used for various types of animal monitoring. If the surface of the animal's body could be precisely reconstructed, it would allow the creation of the automatic growth and body condition monitoring tools. Or, if both poses and shapes could be reliably

reconstructed for a group of animals, advanced behavior monitoring tools could be created. By analyzing the postures and the points of interactions between different individuals, such tools could be used for the analysis of social interactions in different species [21] or detection of the anomalous postures, which could be a sign of a serious illness, such as lameness [22].

While the benefits from the proper 3D animal reconstruction are clear, there are currently no reconstruction methods that produce models of sufficient quality. This thesis thus presents an overview of the already-existing methods, and introduces a new *SMAL4V* method that expends them to the video sequence data.

1.2 Thesis structure

The rest of the thesis is structured in the following way:

- Chapter 2 introduces the geometry behind the 3D scene reconstruction. It explains which geometrical constraints enable the reconstruction of rigid objects, and how those constraints are violated by the articulated moving objects.
- Chapter 3 provides a detailed review of the current State of the Art in the area of 3D reconstruction. In Section 3.1, it starts with the overview of the general reconstruction methods, also discussing their applicability to the animal reconstruction problem. Then, in Section 3.2, it continues with the historical overview of the methods, specific to the animal reconstruction. The rest of the chapter introduces the most promising approaches in more details, as the method, proposed in this thesis, is based on them.
- Chapter 5 describes the datasets used in this thesis, including the newly-proposed *AnimalKey* dataset. Then, it introduces *SMAL4V*, an extension to the methods described in Chapter 3.
- Chapter 6 contains description and results of all performed experiments. Several preliminary experiments once again illustrate the need for the special animal reconstruction methods. The rest of the chapter compares the performance of *SMALST*, *SMBLD* and *SMAL4V* on the *AnimalKey* dataset.
- Finally, Chapter 7 briefly summarizes the thesis and outlines the possible directions for future work.

1.3 Contributions

The main contributions of this thesis are the following:

- It provides a highly detailed overview of the reconstruction methods, currently available for the animal modeling.
- It introduces a new *AnimalKey* dataset, which can be freely used by the researchers interested in the bounding box tracking, instance segmentation, 2D keypoints prediction or 3D reconstruction.
- Finally, it proposes a new *SMAL4V* animal reconstruction pipeline, which can be applied to video sequences with sparse annotations at test time.

Chapter 2

Background Knowledge

3D reconstruction is a geometrical problem of recovering the scene structure from the available 2D images. In the rigid case, its solution relies on the notion of epipolar geometry, which is explained in Section 2.1 for the two-view reconstruction. However, once the rigidity assumption does not hold anymore, the reconstruction problem becomes ill-posed, as is explained in Section 2.2. This explains why the reconstruction of the articulated moving objects, such as animals, is still an open research problem, and why special methods, discussed in the rest of this thesis, have to be designed to solve it.

2.1 Epipolar geometry

Epipolar geometry describes the relationship between different views of the same scene. It is concisely represented in a form of a *fundamental matrix* F, which depends only on the cameras' parameters and their relative poses, being completely independent of the reconstructed scene itself. The goal of a reconstruction pipeline is then to estimate F from a set of images, which is straightforward if the rigidity assumption holds. The rest of this discussion is based on texts from [23] and [24].

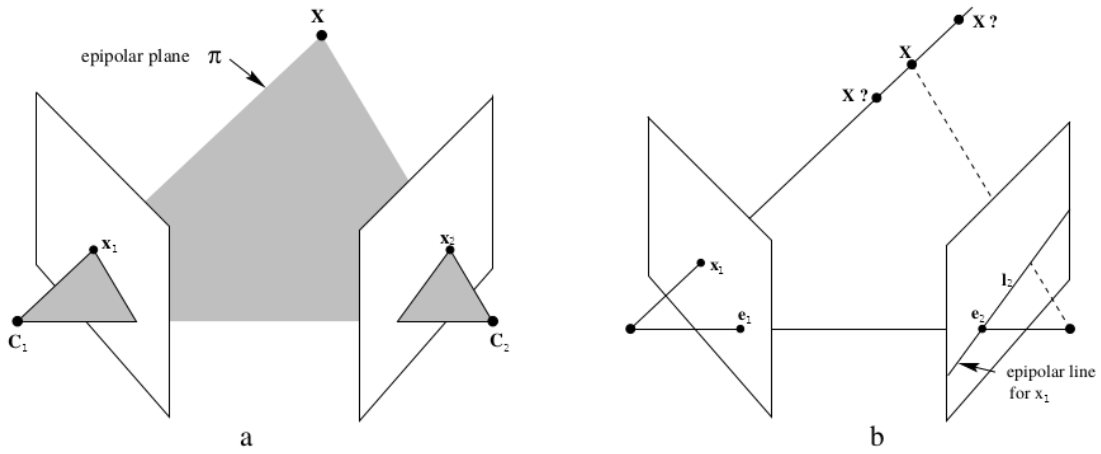

Figure 2.1: Geometry of the point correspondences. *(a)* A point in space X and its images x_1 and x_2, creating the epipolar plane σ. *(b)* A point in 2D image x_1, back-projected into 3D space. Its corresponding point as seen with the right camera would lie on the epipolar line l_2. Image taken from [23].

Having a point in the scene, which we denote as X, and two cameras with

centers at C_1 and C_2, we get two images of the point, x_1 and x_2, in the image planes π_1 and π_2, respectively. Those image points are created by projecting X with camera projection matrices P_1 and P_2:

$$x_i = P_i X;\ i \in \{1, 2\}.$$

The camera projection matrices themselves can be written as:

$$P_i = [K_i R_i | - K_i R_i C_i];\ i \in \{1, 2\}.$$

A rotation matrix $R \in SO(3)$ is a 3x3 orthogonal matrix (i.e. $R^T = R^{-1}$ and $\det R = \pm 1$), which, together with the camera projection center C, defines the camera pose. A calibration matrix K expresses the internal parameters of a camera, which do not depend on the recorded scene or on the camera pose. It is defined as a 3x3 upper-triangular matrix:

$$K = \begin{bmatrix} af & -af\cot\theta & u_0 \\ 0 & f/\sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix},$$

where a is the pixel aspect ratio (usually $a = 1$), θ is the angle between two image axes (usually $\theta = 90°$), (u_0, v_0) is the optical center (in pixels) and f is the focal length (i.e. distance between the camera and its projection plane in pixels).

A line connecting C_1 and C_2 is called a *baseline*. It intersects the image planes π_1 and π_2 at the *epipoles* e_1 and e_2, which means that the epipoles are projections of the camera centers:

$$e_1 = P_1 C_2 \text{ and } e_2 = P_2 C_1.$$

A plane comprising points X, x_1 and x_2 is then called an *epipolar plane* σ. The epipolar plane σ intersects π_i at the *epipolar line* l_i, which passes through the image point x_i and the epipole e_i, i.e. $l_i = e_i \times x_i$ for $i \in \{1, 2\}$.

If we look just at the image point x_1, a possible location of the back-projected space point X is ambiguous, and can lie anywhere on the ray $C_1 x_1$, as shown in Figure 2.1(b). In this case, X can be written as

$$X(\lambda) = P_1^+ x_1 + \lambda C_1, \tag{2.1}$$

where P^+ is the pseudo-inverse of P and $\lambda \in [0, \infty)$.

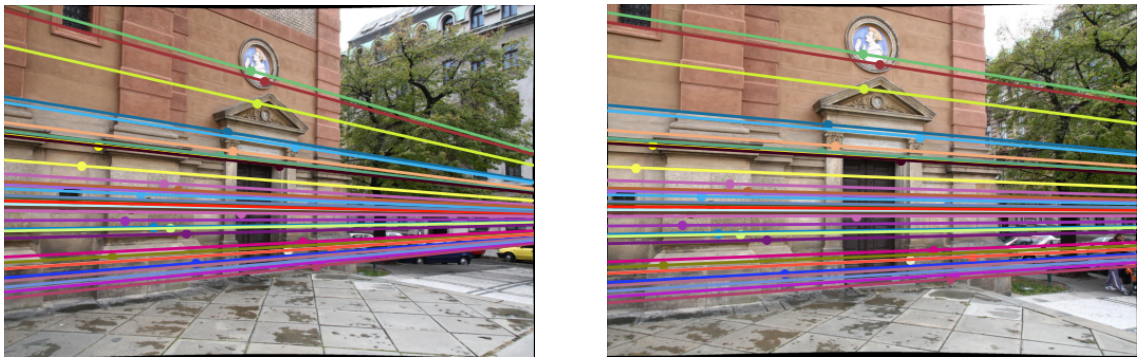

Figure 2.2: A pair of images with a subset of detected inlier image correspondences and their epipolar lines.

However, the position of X can be determined by finding the corresponding point x_2 in the second image plane. An example of such corresponding points along with the epipolar lines, on which they lie, is shown in Figure 2.2.

The *epipolar constraint* states that a corresponding point x_2 (in π_2) for the image point x_1 (in π_1) must always lie on the epipolar line l_2, thus creating a projective mapping $x_1 \rightarrow l_2$. A *fundamental matrix* F, mentioned above, defines this mapping based on the cameras' properties and poses. Since this mapping is clearly non-invertible, the matrix F is never full-rank.

To derive the exact expression for F, we can take two points on $C_1 x_1$:

$$X(\lambda = 0) = P_1^+ x_1 \text{ and } X(\lambda = \infty) = C_1.$$

Now, if we project those by the second camera's projection matrix, we get:

$$P_2 X(\lambda = 0) = P_2 P_1^+ x_1 \text{ and } P_2 X(\lambda = \infty) = P_2 C_1.$$

Since the epipolar line l_2 is a projection of the ray $C_1 x_1$, it can be written as

$$l_2 = (P_2 C_1) \times (P_2 P_1^+ x_1).$$

Also, since $P_2 C_1$ is a projection of the first camera center by the second camera, it is the same as the epipole in π_2 (i.e. e_2), which gives the following expression for l_2:

$$l_2 = [e_2]_\times (P_2 P_1^+) x_1.$$

The fundamental matrix is then defined as a 3x3 rank-2 homogeneous matrix F:

$$F \stackrel{\text{def}}{=} [e_2]_\times (P_2 P_1^+)$$

so that the epipolar lines l_1 and l_2 can be expressed as

$$l_1 = F^T x_2 \text{ and } l_2 = F x_1 \tag{2.2}$$

for $x_1 \neq e_1$ and $x_2 \neq e_2$.

Also, for $\forall x_1$ ($x_1 \neq e_1$ and $x_1 \in \pi_1$), there exists a corresponding epipolar line l_2 ($l_2 \in \pi_2$). By definition, l_2 contains the epipole e_2, i.e. $e_2^T l_2 = 0$. Substituting the result from Equation 2.2 for l_2, we get $e_2^T F x_1 = 0$. Since this has to hold for all image points $x_1 \in \pi_1$ ($x_1 \neq e_1$), e_2 is the left null-vector of F, i.e. $e_2^T F = 0$. Analogously, it can be shown that e_1 is the right null-vector of F, i.e. $Fe = 0$.

Finally, using Equation 2.2 and the fact that a point correspondence x_2 has to lie on the epipolar line l_2 (i.e. $x_2^T l_2 = 0$), we get

$$x_2^T F x_1 = 0 \tag{2.3}$$

for all corresponding points $x_1 \leftrightarrow x_2$.

This property of the fundamental matrix is crucial for the 3D reconstruction problem. It allows us to express F in terms of the image correspondences, ignoring the cameras' parameters, which are usually unknown in the real-world problems. Since F is a 3x3 homogeneous matrix, Equation 2.3 implies that we can compute F up to scale from 8 point correspondences. Additionally, since F is a rank-2 matrix, we can add a $\det F = 0$ constraint to estimate F from only 7 point correspondences.

Since the ground-truth image correspondences are never available (unless the synthetic data is used), more complicated algorithms have to be used to reliably

estimate F from a set of noisy tentative correspondences. An example output for one such algorithm, based on *RANdom SAmple Consensus* (*RANSAC*) [25], is shown in Figure 2.3. Each arrow shows the direction from the image point $x_i \in \pi_i$ ($i \in \{1, 2\}$) to its corresponding point in the other image. Note how the direction of arrows indicate how the camera pose changed between two frames.

Figure 2.3: Example of the automatically detected inlier point correspondences. The origin of each arrow corresponds an interest point in the given image, while the head of the error corresponds to the position of its corresponding point in the other image.

2.2 Non-rigidity and motion

When a rigid object starts moving, it creates ambiguity between its own motion and the camera motion. This can be, however, addressed by segmenting the image into static background and moving foreground, and estimating camera pose from the background only [26].

But as soon as we allow non-rigid deformations of the object of interest, as shown in Figure 2.4, the 3D reconstruction problem becomes ill-posed even if we assume a static camera. For a rigid object, position of any point on its surface at time t can be described by a single function $X(t) = R(t)X_0 + T(t)$, where $R(t)$ is a rotation matrix and $T(t)$ is a translation vector at time t.

Figure 2.4: Example frames showing a jellyfish moving and deforming at the same time. Image taken from [27].

For a deforming object, position of any point is instead obtained from the composition of a rigid motion with an unknown deformation function $h(\cdot, t)$, i.e. $X(t) = h(R(t)X_0 + T(t), t)$. However, for any $(h(\cdot, t), R(t), T(t))$, there exist infinitely many choices of $(\tilde{h}(\cdot, t), \tilde{R}(t), \tilde{T}(t))$ that produce the same $X(t)$ [27]:

$$X(t) = h(R(t)X_0 + T(t), t) = \tilde{h}(\tilde{R}(t)X_0 + \tilde{T}(t), t).$$

This ambiguity makes epipolar geometry undefined for the non-rigid case and motivates the creation of special reconstruction methods, which are the main focus of this diploma thesis.

2.3 Neural Networks

Since their introduction in 1940s as computation model for brain neurons [28], *Artificial Neural Networks* (*ANNs*) have become state-of-the-art solutions for multiple problems, including visual classification, speech recognition, machine translation and many others. Guided both by the successes of neural networks in computer vision tasks and by the findings of the neural scientist regarding 3D shape interpretation in visual cortex, this work explores the usage of *ANNs* for the reconstruction problem. This Section provides a brief introduction to the types of architectures used in this thesis, as well as to the models used as a black-box inside the proposed pipeline.

2.3.1 Convolutional Neural Networks

Considering the number of pixels in a typical image, the regular fully-connected *ANNs* would require too many connections to be trained. Moreover, they are not spatially-invariant, so the same object would be perceived differently depending on its position in the image. These two reasons lead to the creation of *Convolutional Neural Networks* (*CNNs*) [29][30][31], which have been state-of-the-art computer vision models for many years now. A comparison between a fully-connected and convolutional networks is shown in Figure 2.5.

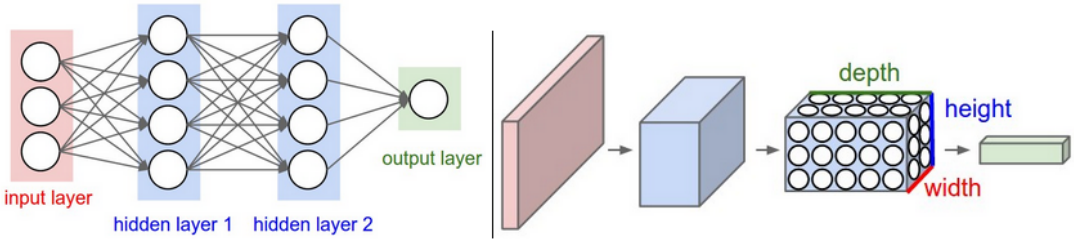

Figure 2.5: *Left:* a fully-connected neural network. *Right:* a convolutional neural network. Image taken from *https://cs231n.github.io/convolutional-networks/*.

Convolutional layers are the main building blocks of *CNNs*. They are based on the mathematical convolution operation, shown in Figure 2.6. In a convolutional layer, each output value corresponds to the convolution (i.e. sum of element-wise multiplication) between the 3D input receptive field and a 2D learnable filter. The filter size is a hyperparameter of a convolutional layer, which also defines the size of the receptive field.

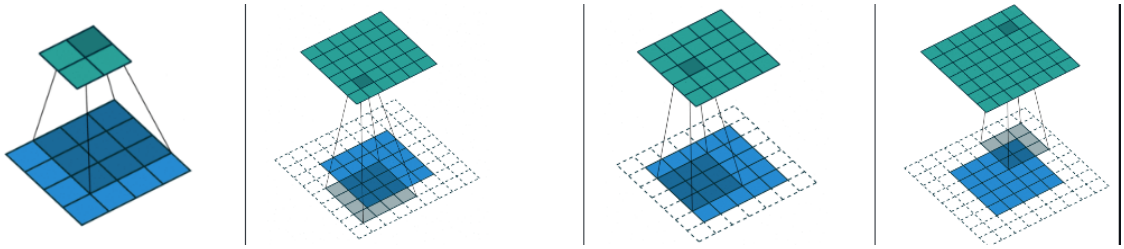

Figure 2.6: 2D convolutions with different padding. Image taken from [32].

Number of such filters, which is another hyperparameter, corresponds to the amount of depth channels that will be present in the output volume. Other hyperameters are stride, which defines the relative shift between the neighboring outputs' receptive fields, and the padding, which controls the padding of the whole input volume. For more details, including the animations of different convolution operations, please refer to [32] and *https://cs231n.github.io/convolutional-networks/*.

2.3.2 Recurrent Neural Networks

Recurrent Neural Networks (*RNNs*) are another modification of the classical fully-connected *RNNs*, created to allow processing of variable-length input sequences and predicting variable-length outputs. This is enabled by sharing the layer weights across time instead of learning separate weights matrices in each time step. Moreover, the recurrent architecture allows modeling of temporal dependencies by maintaining a hidden state vector which encodes the historical information. Special version of backpropagation, called backpropagation-through-time, is used train such models.

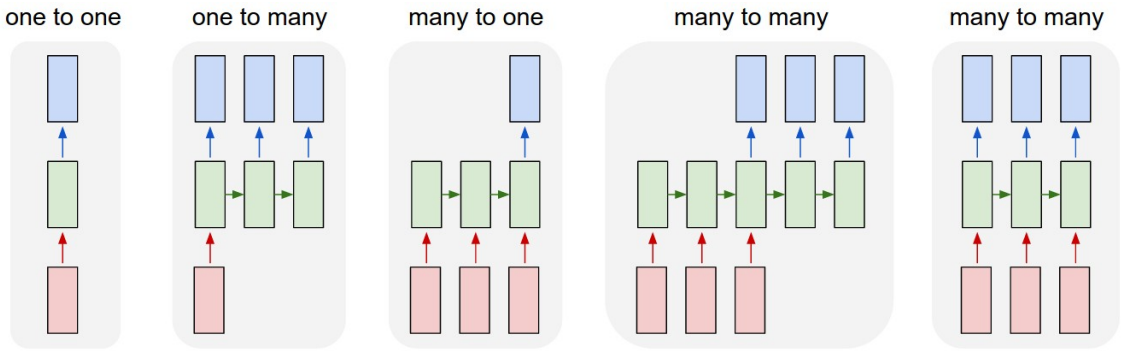

Figure 2.7: Neural networks with different input and output types (with the leftmost corresponding to a vanilla *ANN* and the rest corresponding to different versions of *RNNs*). Image taken from *http://karpathy.github.io/ 2015/05/21/rnn-effectiveness/*.

While compelling in theory, vanilla *RNNs* suffer from the vanishing gradient problem [33], preventing it from learning long-term dependencies. This issue has been addressed by the creation of more complicated hidden cells, such as *Long Short-Term Memory* (*LSTM*) [34] and *Gated Recurrent Unit* (*GRU*) [35], shown in Figure 2.8.

LSTM cells control the information flow with three sigmoid gates: input gate, forget gate and output gate. Additionally, they maintains a so-called cell state, which corresponds to the long-term memory. Input and forget gates control how this cell state is updated throughout the learning process, while output gate controls which parts of the cell state are sent to the cell's output. For more information on *LSTM*s, refer to [34] or *https://colah.github.io/posts/2015-08-Understanding-LSTMs/*.

GRU cells can be viewed as a simplified version of *LSTM*s. They do not maintain any cell state so that all the hidden state of a *GRU* cell is passed to its output. It then merges input and forget gates into a single update gate, which controls how the hidden state is modified during training. Thanks to these modification, *GRU* cells are faster to train, while providing comparable performance to *LSTM*s. For more information on *GRU*s, refer to [35] or *https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks*.

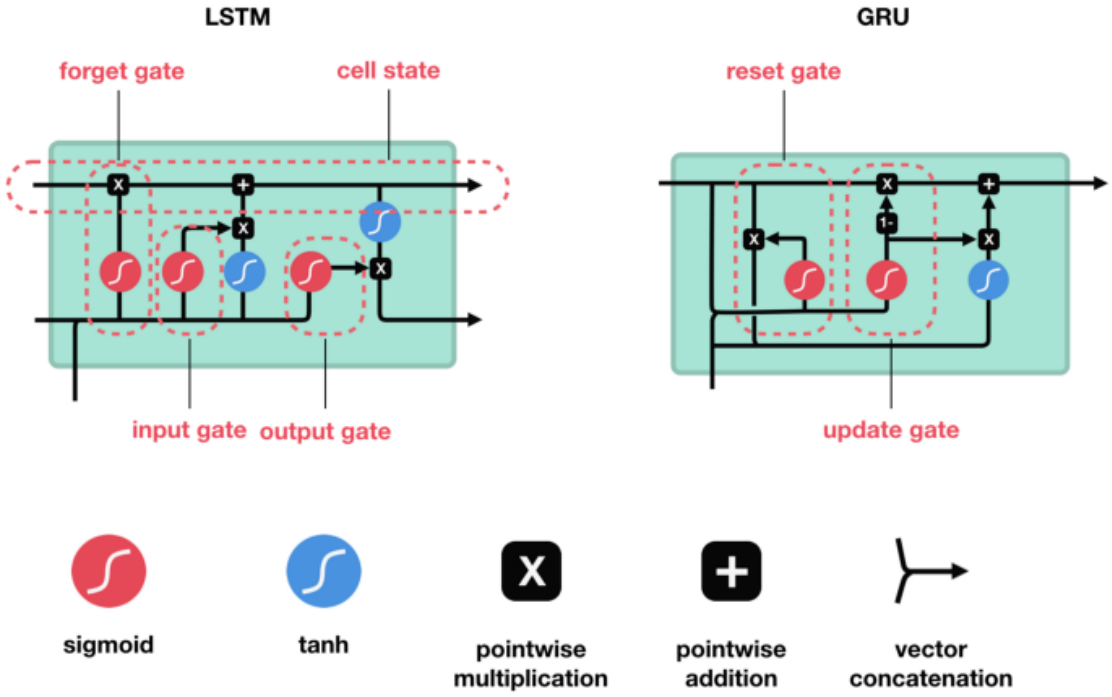

Figure 2.8: *Left*: *LSTM* hidden cell. *Right: GRU* hidden cell. Image taken from *https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21*.

RNNs with *LSTM* or *GRU* hidden cells have been especially successful in machine translation and sentiment analysis tasks, at least until the introduction of the *Transformer* model in [36]. They can be further modified, for example, by allowing the hidden state to access information from the future. This modification is called a *Bidirectional RNN* and can be used when the complete input sequence is available to the network at test time. Additionally, multiple recurrent layers can be stacked on top of each other to create deep *RNNs*.

Since this thesis proposes a method for the video sequence processing, it adds the recurrent layers to the proposed architecture so that the variable-length input could be efficiently processed. The *GRU* cells are used as they provide the trade-off between the simplicity and the ability to learn time dependencies.

2.3.3 Neural Renderer

The goal of a mesh renderer is to generate a 2D image from a 3D mesh, which consists of vertices $\{v_i; i = 1..N\}$ and faces $\{f_i; i = 1..N\}$. A renderer first transforms the vertices into the image space from the scene space and then assigns the values to individual image pixels by *sampling*, as shown in Figure 2.9*(a)*. This latter step is called *rasterization* and is clearly discrete. It means that, if gradients are to be computed for this operation, they would be 0 for almost all pixels, as shown in Figure 2.9*(c)* for the 1D case. So, while the success of neural networks has made it attractive to try *CNNs* for the 3D reconstruction problem, it was not straightforward to create an end-to-end trainable pipeline for this task because of the broken gradient flow through the rasterization step.

In [37], this was finally solved by allowing the approximate gradient for the rasterization operation to be passed through the backpropagation. This allowed the creation of trainable 3D mesh predictors, such as one shown in Figure 2.10. A brief overview of the approximate gradients computation is provided below.

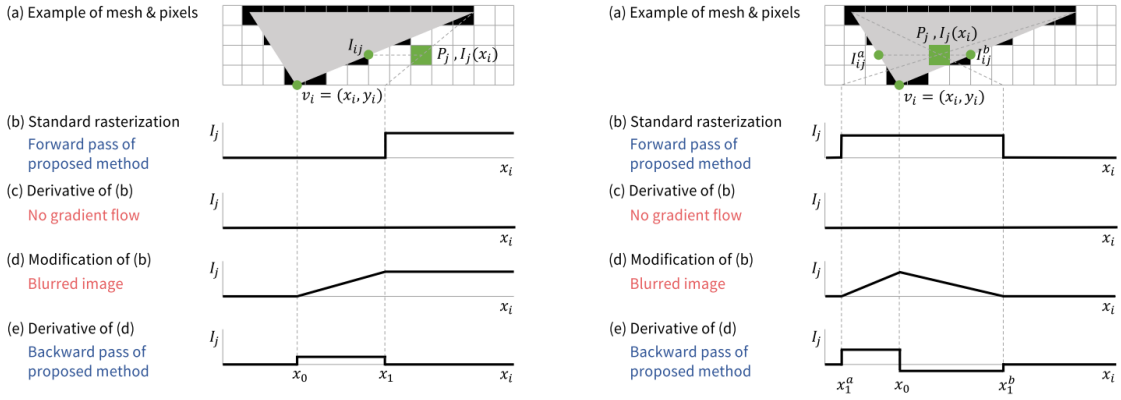

Figure 2.9: Calculation of approximate derivatives when: a pixel is outside of the rendered face *(left)*, pixel is inside of the rendered face *(right)*. Image taken from [37].

As already mentioned, an image pixel is assigned its intensity value based on the sampling operation. Since the goal is learning to predict the mesh, its vertices $v_i = (x_i, y_i)$ are changing throughout the training, and the gradient with respect to v_i has to be backpropagated through the network, as shown in Figure 2.10. As we change the x_i coordinate of v_i, the color $I_j(x_i)$ of an image pixel P_j is changing, as shown in Figure 2.9*(b)*. If the pixel of interest lies outside the face, defined by the current position of v_i, i.e. $v_i = (x_i = x_0, y_i)$, its intensity value is $I_j(x_0)$. However, if the vertex v_i is moved sufficient close to P_j (i.e. $v_i = (x_i = x_1, y_i)$ during training, its intensity value suddenly changes to the color of the corresponding mesh face, i.e. I_{ij}. The resulting $I_j(x_i)$ function from the standard rasterization process is thus a step function, which is non-differentiable at its discontinuity point ($x_i = x_1$), and with $\frac{\delta I_j(x_i)}{\delta x_i} = 0$ for the rest of $I_j(x_i)$'s domain.

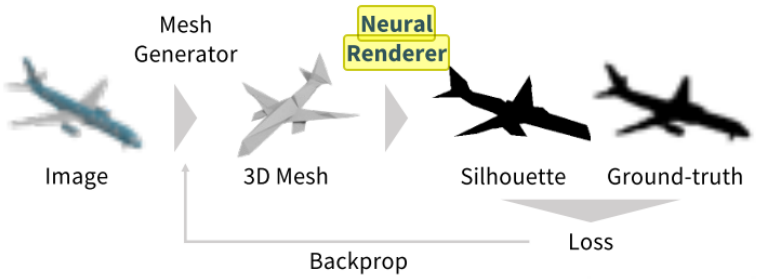

Figure 2.10: *NeuralRenderer* used to enable backpropagation through a simple 3D reconstruction pipeline. Image taken from [37].

To fix the 0 gradients problem, the original $I_j(x_i)$ function is smoothed by linear interpolation. Now, the gradients $\frac{\delta I_j(x_i)}{\delta x_i}$ are non-0 in the region (x_0, x_1). Same approach can applied for the pixel P_j inside of the mesh's face, or for calculation of $\frac{\delta I_j(x_i)}{\delta y_i}$ In this case, two non-0 derivatives region are created: (x_1^a, x_0) and (x_0, x_1^b), as shown in Figure 2.9 *(right)*.

2.3.4 One-Shot Video Object Segmentation

One-Shot Video Object Segmentation (*OSVOS*) has a two-fold contribution to this thesis. First, it provides an inspiration for the proposed *SMAL4V* reconstruction pipeline. Second, *OSVOS* itself is used inside that pipeline to create dense mask supervision signals from available sparse annotations. It is a semi-supervised method which requires only a few annotated frames to provide the segmentation masks for the rest of the video, as shown in Figure 2.11. Below follows a brief explanation of the *OSVOS* method. For more details, refer to the original research in [38].

Figure 2.11: Example segmentation results by *OSVOS*. Image taken from [38].

An overview of the *OSVOS* pipeline is shown in Figure 2.12. It consists of three stages, each of which outputs a single neural network, all with the same architecture but with different learned weights. The first stage outputs a *base network*, which is a *CNN* pre-trained for *ImageNet* task, i.e. not for the segmentation prediction. Despite being trained for image classification, it learns the convolutional filter weights that prove useful in the later stages.

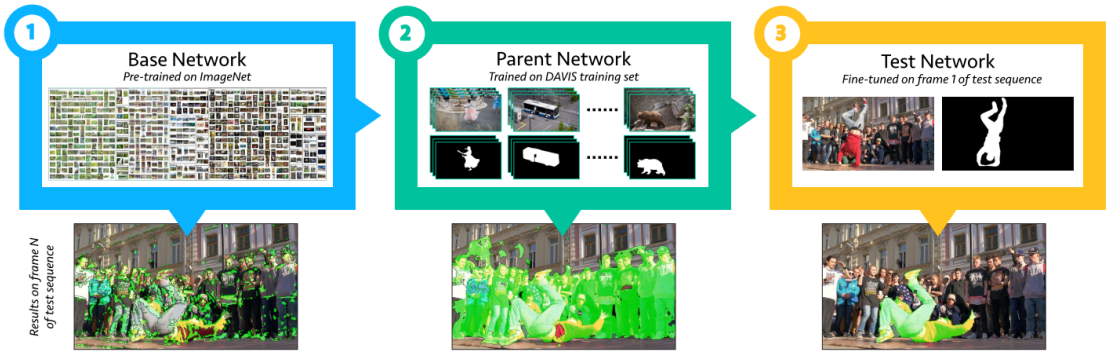

Figure 2.12: Overview of the *OSVOS* pipeline. Image taken from [38].

The second stage trains a so-called *parent network* on the *DAVIS* training set for the multi-instance segmentation task. It learns how to separate different objects from the image background, and comprises an important intermediate step between the first and last stages. The final stage then trains a *test network* for a single object in a single video sequence. The training is performed on just a handful of the annotated frames, often with just the first frame being annotated. The network is thus trying to overfit to the appearance of the object of interest so that it can efficiently predict it in the rest of the frames. No temporal information is used for the training. Yet, despite that, the test network manages to achieve temporally consistent results. Its relatively low training time and high frame-rate at test time make the *OSVOS* method widely used in the research community.

Chapter 3

Literature Overview

Recovery of the 3D structure from images or video sequences has been an area of active research for several decades [39][40][41][42][43][44][45]. Many reconstruction methods have been introduced, which can be classified depending on the required input, type of scenes or objects to which they are applicable, or prior knowledge incorporated into the reconstruction pipelines. This chapter tries to provide a comprehensive overview of those methods so that it is clear how the *SMAL4V* model, proposed in this thesis, is connected with different alternative approaches.

The rest of the chapter is structured as follows: Section 3.1 provides a general overview of the state-of-the-art 3D reconstruction methods. Then, Section 3.2 discusses the methods, created specifically for the reconstruction of animals, and their evolution throughout the recent years. Sections 3.3 and 3.4 describe a family of modern learning-based approaches that utilize the *SMAL* parametric model for the reconstruction of various animal species.

3.1 Overview of the Reconstruction Methods

Firstly, *active* and *passive* reconstruction techniques are distinguished. Active reconstruction interacts with the scene by the means of light projectors or time-of-flight sensors, often combined with the standard RGB input. Cameras that can gather such an input are usually called *RGB-D cameras* or *depth sensors*. Depth information provided by these cameras is usually more precise than the depth maps estimated from the RGB images alone. However, the required hardware for the reliable depth estimation is less widespread and significantly more expensive than the widely-available RGB cameras. Instead, passive reconstruction techniques use conventional RGB images, which are much easier to collect, with millions of images available online. Section 3.1.1 gives more information about the active reconstruction methods, while the rest of the chapter describes approaches that fall in the passive reconstruction category.

Likewise, we can distinguish between *rigid* and *non-rigid* reconstruction. As follows from the name, the former deals with the reconstruction of static rigid objects, while the latter tries to reconstruct the objects that are deformable or articulated. As already explained in Chapter 4, 3D reconstruction of static rigid objects is a well-posed problem that can be solved with the methods based on the epipolar constraints. Those methods, however, fail for the most of real-world scenarios, which usually involve dynamic scenes with a variety of non-rigid objects. On the other

hand, non-rigid reconstruction is an ill-posed problem [46], which has been an area of active research for several decades [42][47][41][48][46][49][45]. Many proposed solutions encourage the use of some prior knowledge about the reconstructed object, such as knowledge about its category (eg. human, animal, vehicle etc.)[50][51], or constraints on the space of the object's possible deformations [52][53].

The *template-based* methods, which can be used for both rigid and non-rigid reconstruction, incorporate such prior information by assuming a known 3D template [54]. To additionally simplify the problem, 3D-to-2D correspondences can be provided with the help of a reference image [55].

Finally, following the success of the machine learning in other computer vision problems, the *learning-based* reconstruction techniques emerged. Those learn automatically from the training data, so the image features, on which the predictions are then computed, do not have to be manually engineered. These approaches have been recently successful in many applications for the depth estimation or 3D reconstruction [56][57]. On the other hand, those also require big datasets and considerable computational resources, which is not always feasible in practice. Despite these disadvantages, learning-based approaches currently provide the most promising results for the difficult articulated objects (such as humans and animals), and serve as a basis for the *SMAL4V* method, proposed as part of this diploma thesis.

3.1.1 Active reconstruction

As briefly mentioned above, active reconstruction techniques directly interfere with the observed scene, for example, by influencing the illumination. This requires a special kind of camera, an RGB-D camera, with many different types currently available on the market.

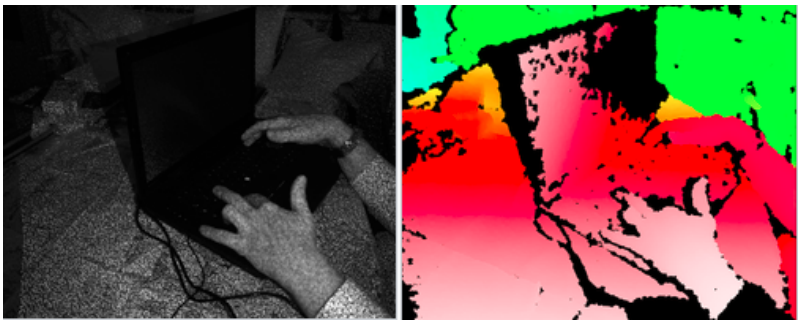

Figure 3.1: Example from the Microsoft Kinect camera. *Left*: image from the infrared camera, with structured light pattern visible in the IR spectrum. *Right*: the corresponding depth map with color map going from white (nearest) to blue (furthest). Images taken from *Wikipedia*.

Structured-light depth cameras combine a regular RGB camera with a laser projector, which projects an a-priori-known pattern into the scene [58]. Such pattern provides the reconstruction algorithm with a set of unique projection-to-camera correspondences, which can be used for the reliable depth estimation. An example of the projected pattern and a depth map, estimated from it, is shown in Figure 3.1. Examples of the structured-light cameras, among others, include ASUS Xtion Pro, Intel RealSense SR305, and Microsoft Kinect [59], which are shown in Figure 3.2.

Related is the *active stereo* approach, which also projects different light patterns into the scene. It, however, does not required complicated well-designed patterns, and projects, for example, random dots or stripes [60]. The depth estimation is then based on the passive stereo techniques, which use the projected points as additional camera-to-camera correspondences to improve the reconstruction quality in the untextured or ambiguous regions. Among cameras, utilizing this approach, are Intel RealSense D415 and Intel RealSense D435 [59], shown in Figure 3.3 *(left)*.

Figure 3.2: Examples of RGB-D cameras, based on the structured light technology. *Left*: ASUS Xtion Pro. *Middle*: Intel RealSense SR305. *Right*: Microsoft Kinect for Xbox 360.

Time-of-Flight (*ToF*) depth cameras interact with the reconstruction scene by sending the infrared light, which is then reflected and returned to the sensor. The depth estimation is performed by measuring the return time of the traveled light at each pixel [61]. *Pulsed-modulation ToF* cameras, such as Advanced Scientific Concepts' GSFL-4K/16K/16KS, send a single pulse and directly measure its return time to acquire the depth information from the scene, which requires very accurate sensors. *Radio-frequency-modulated ToF* devices, such MESA SwissRanger 4000, send a continuous signal and measure the depth by estimating the phase shift between the sent and returned signal. *Ranged-gated imagers*, such as Microsoft Kinect v2, contain a shutter, which blocks part of the returning light depending on the distance between the sensor and the object, thus allowing to determine the depth based on the brightness of the imaged point. The main challenge then lies in the depth calculation algorithm, as it has to properly model the light sources, the shutter closing times, different reflective properties of the objects etc. Those more complicated algorithms, however, allows to reduce the cost of the hardware itself. Some examples of the *ToF* cameras (Microsoft Kinect v2, MESA SwissRanger 4000, Creative Senz3D and SoftKinetic DS325 [59]) are shown in Figure 3.3 *(right)*.

Figure 3.3: Examples of RGB-D cameras. *Left*: active stereo cameras (Intel RealSense D415 and Intel RealSense D435). *Right*: *ToF* cameras (Microsoft Kinect v2, MESA SwissRanger 4000, Creative Senz3D, Soft Kinectic DS325).

Fusion Techniques

Since the raw scans, obtained from the depth cameras, usually suffer from a lot of noise and missing data, sophisticated algorithms are applied to improve the reconstruction quality, or properly merge the point clouds, resulting from different frames, into a single 3D model [62].

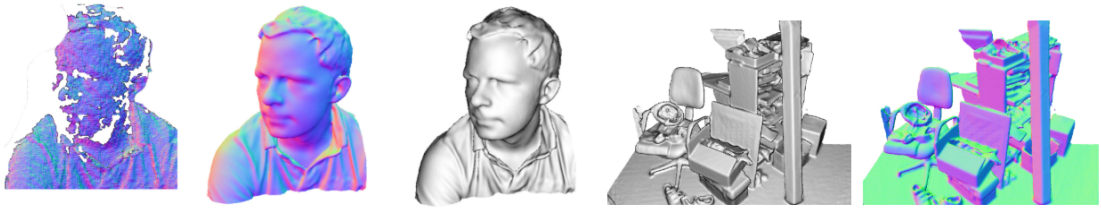

Figure 3.4: *Left:* the input *ToF* scan. *Color:* the normal maps after applying KinectFusion. *Grayscale:* Phong-shaded renderings after KinectFusion. Image taken from [63].

Proposed for the widely-used Kinect device, the *KinectFusion* approach is one of the most well-known techniques for the depth data fusion [63]. Though applicable only to the static scenes, it was the first approach that provided the real-time reconstruction possibility from the commonly accessible hardware.

Later, the need for the reconstruction of real-world scenes lead to the generalization of *KinectFusion* to dynamic scenes. The first approach to do so was *DynamicFusion* [64], which estimates a dense 6D volumetric field and warps each reconstructed frame to the space of the first frame, thus enabling the use of the classical *KinectFusion* updates on the warped models. An example of the continuous improvement of the reconstruction quality for the dynamic scene is shown in Figure 3.5.

Figure 3.5: Real-time reconstruction of a dynamic scene with *DynamicFusion*. Image taken from [64].

Following the *DynamicFusion*, more approaches emerged to further improve the quality of the dynamic reconstruction with RGB-D cameras [65][66][57][67][68]. The comparison of some of them is shown in Figure 3.6.

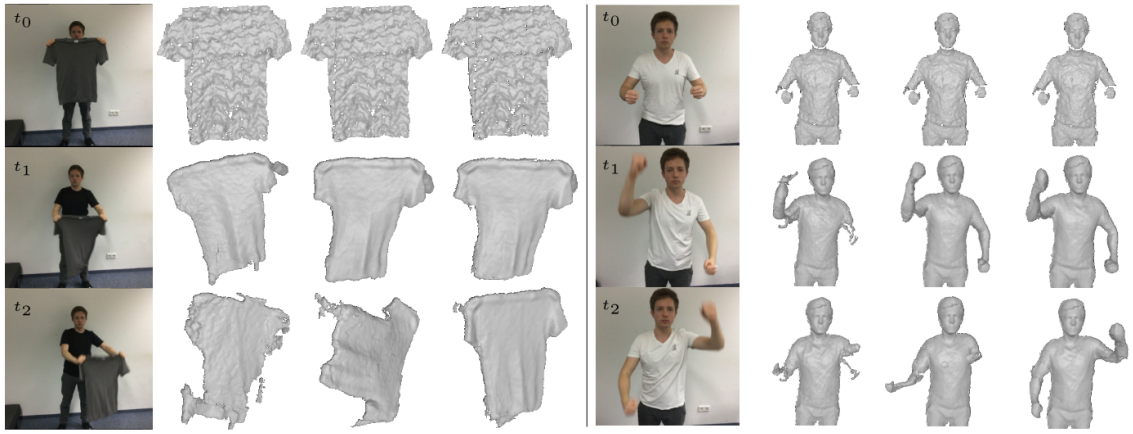

Figure 3.6: *From left to right*: input frame, reconstruction with *DynamicFusion* [64], reconstruction with *DeepDeform* [68], reconstruction with *Neural Non-Rigid Tracking* [67]. Image taken from [67].

While these methods present high-quality results for many dynamic scenes, their most obvious drawback is their reliance on the special hardware, which is less wide-spread and more expensive than the conventional RGB cameras. Moreover, most of them suffer from performance losses if used in the real-world scenes with uncontrolled illuminations due to the hardware limitations of the RGB-D cameras.

3.1.2 Structure-from-Motion

Structure-from-Motion (*SfM*) is a passive reconstruction technique used to estimate the 3D structure of an object from a set of images, taken by a moving camera. This approach is based on the epipolar geometry, described in Chapter 4. Thus, it strongly relies on the search of point-to-point correspondences between the individual image pairs, and experiences problems with reconstructing untextured objects or specular surfaces. Moreover, it requires a considerable out-of-plane rotation of a camera to create a good reconstruction, complicating the data collection, as shown in Figure 3.7. However, once proper input data is available, the scene geometry can be uniquely recovered by the factorization approach [69].

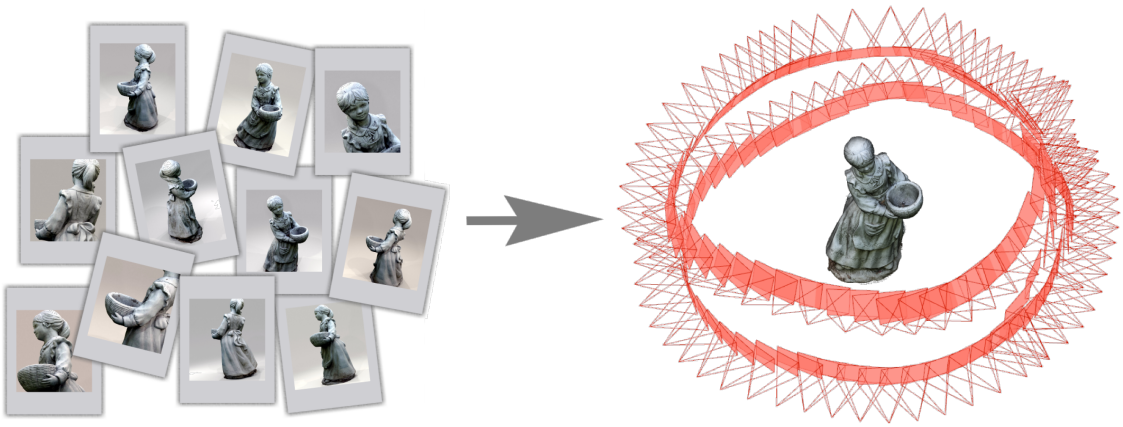

Figure 3.7: *Left*: examples of images of an object taken by a single moving RGB camera. *Right*: rigid object of interest and the camera positions (in red) from which the images were taken. Image taken from [70].

Since many real-world scenes are not in fact static, in recent years the *SfM* was extended to the dynamic scenes and non-rigid or articulated objects. In *Non-Rigid Structure-from-Motion* (*NRSfM*), effects from the pose variations of a camera are combined with the shape deformations of the observed object. However, if the dynamics of the object are not constrained, the problem becomes ill-posed [71] since many 3D deformable shapes and camera poses can result into the same 2D projections [72], as explained in Section 2.2. Thus, additional constraints in the form of different deformation priors have to be used.

Following the success of the factorization approach for rigid *SfM*, early *NRSfM* methods adapted it to the non-rigid objects by introducing a low-rank constraint on the object's deformations [49][73][46][72][45]. This low-rank constraint was first proposed in [49], along with a rank-3K constraint, similar to the rank-3 constraint for the rigid *SfM*. The non-rigid shapes are then assumed to lie on the linear space spanned by K unknown 3D basis shapes [45]. Its dual approach is to express the

evolving 3D shape in trajectory space by a linear combination of the basis trajectories [71]. Moreover, additional spatial or temporal [46][45] smoothness constraints can be used to get smoother reconstructed surfaces.

In [74], the inherent ambiguity of the orthonormality constraints was discussed, and the additional basis constraints were proposed, claiming to provide a closed-form solution to non-rigid shape and motion recovery. This work lead to the creation of many approaches which utilize additional priors on top of the orthonormality constraints, such as Gaussian prior on the shape coefficients [46], ordering prior on the basis [75], or assumptions of the articulated motion [76].

In [77], the necessity of the basis constraints was challenged. The additional rank-3 constraint, overlooked in [74] was applied to prove that the orthonormal constraints (together with the rank-3 constraint) are in fact sufficient for the unambiguous shape recovery, and that the actual difficulty of the non-rigid reconstruction instead lies in the difficult and highly non-linear shape of the cost function used for the optimization.

While often-used and well-performing for the smoothly deforming objects, low-rank constraints are often not enough for handling strong local deformations. Apart from that, they require the selection of the number of shape bases, which is usually different for each object category. So, many recent approaches use physical models for the object's deformation instead. The examples include modeling isometry [78] or elastic deformations [53], or modeling the non-rigid objects as ensemble of particles, whose interactions are described by the Newton's second law of motion [79].

While the methods, mentioned above, perform the global object-wise reconstruction, it is also possible to perform the reconstruction separately in the piecewise [42][80] or point-wise fashion [81]. In that case, after the *NRSfM* techniques are applied separately on each region of the reconstructed surface, the stitching of the intermediate results is performed to get the final 3D model.

3.1.3 Template-based methods

It often happens that the quality or the amount of the available data is not satisfactory for creation of a high-quality reconstruction with *NRSfM* methods, especially if the object of interest has a complicated shape or significant non-rigid deformations. In such situations, additional prior knowledge can be incorporated into the reconstruction pipeline.

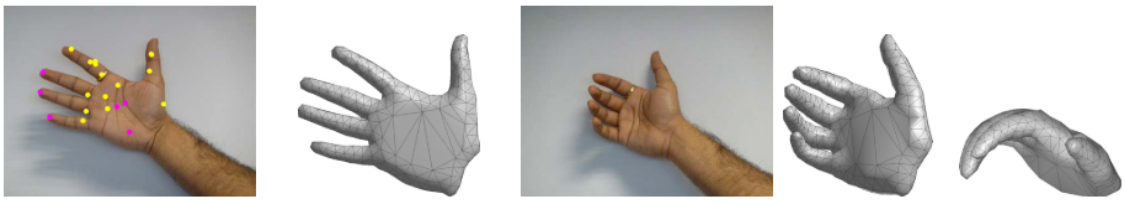

Figure 3.8: Example of the template-based reconstruction. *From left to right:* (a) reference image with 2D-to-3D correspondences marked, (b) input template, (c) input RGB image, (d) resulting 3D reconstruction. Image taken from [55].

If the category of the reconstructed object is known, a strong a-priori assumption about the object's appearance can be expressed by utilizing a 3D shape template. A 3D scan of a similar object or an artist-defined 3D model can be used

as such template. Then, only the relative deformations of the template have to be estimated from the input data, while the general topology of a 3D model remains unchanged. Additionally, a 3D template can be connected to a reference image, providing ground-truth 3D-to-2D correspondences. An example of reconstruction, obtained from the input RGB image with the help of a 3D template and its corresponding reference image is shown in Figure 3.8.

Since all the topological information is provided by the template itself, it is often sufficient to use only one image of the object of interest to perform the 3D reconstruction. Even those parts of the object, which are not visible in the input image, can still be plausibly reconstructed, providing a clear advantage in the situations with lacking data.

Because regular 3D templates contain no a-priori information about the space of possible deformations, additional geometric constraints (such as inextensibility [82], isometry [83] or conformity [84]) can be used to limit the output deformations, same as in *NRSfM* [55]. Many early template-based methods focus on the isometry or inextensibility constraints, but demonstrate the results only for a very limited set of flat bounded shapes, such as papers or T-shirts. More recent approaches try to widen the applicability of the template-based reconstruction, applying it, among others, to the animal reconstruction, as will be discussed in Section 3.2.

3.1.4 Model-based methods

Model-based methods can be, in some sense, viewed as an extension of the template-based methods, combining those with machine learning techniques. They aim at representing the resulting reconstruction by a short vector of model parameters, where the space of parameters is learned from a large set of ground-truth 3D scans. This family of methods is especially successful for the human [85][86][87] and animal [88] reconstruction.

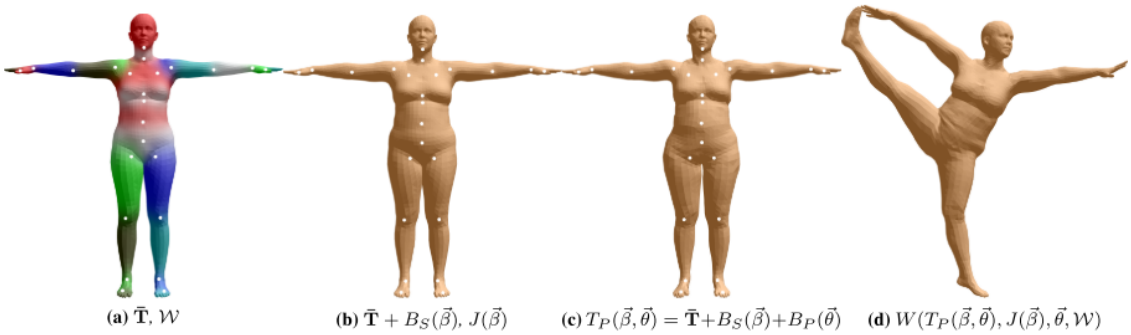

Figure 3.9: Example of a human *SMPL* model. *From left to right*: (a) human template, (b) template with the custom shape deformations, (c) template with the additional pose-dependent deformations, (d) posed and deformed template. Image taken from [86].

In [85], the *Shape Completion and Animation for PEople* (*SCAPE*) model is introduced to address the human reconstruction problem. It utilizes a pre-defined human template, similar to the approaches described in Section 3.1.3, whose deformations are factored into *shape-dependent* deformations (due to a person's appearance) and *pose-dependent* deformations (due to the articulated motion). In [86], a similar idea is used to create the *Skinned Multi-Person Linear* (*SMPL*) model,

example of which is shown in Figure 3.9. Unlike *SCAPE*, which multiplies triangle deformations, *SMPL* is additive in vertex space, allowing to easily fit it to 3D data by the vertex error minimization [86]. Additionally, the availability of the explicit 3D joints in *SMPL* model allows it to be trained on the 2D labelled keypoints by minimizing the reprojection error [89].

SMPL model has become very popular in both research and industry. Multiple extensions to *SMPL* were proposed, allowing the continuous reconstruction from consecutive frames [90] or from video sequences [91][92]. Blender, Unreal Engine and Unity have *SMPL* plugins, which allow simplified creation of realistic human models for game development, computer animation etc. Recently, an improvement to *SMPL* model, called *STAR*, was proposed in [87]. It generalizes better to the previously unseen bodies, while also reducing the number of model parameters.

Also, *Deep Implicit Functions* (*DIFs*) were proposed as an alternative 3D shape representation [93][94]. A template implicit function is used instead of a *SMPL*-like mesh-based deformable template to represent the mean shape of some object category. A conditional spatial warping function is then used to deform the template and personalize the resulting 3D shape [94]. The main advantage of such templates is that they can be automatically learned from the training data for different object categories, as shown in Figure 3.10*(top)*, without the need to create a separate template for each object type like with mesh-based templates.

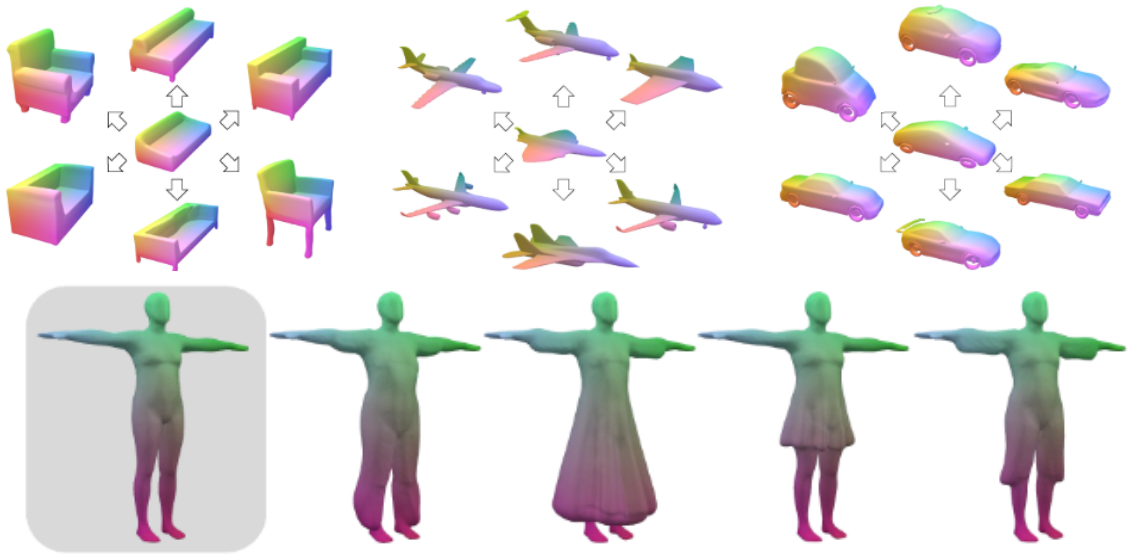

Figure 3.10: *Top:* example of the learned category-specific templates. Image taken from [94]. *Bottom:* example of a *SMPL* model, clothed with the use of *deep implicit functions*. Image taken from [94].

Moreover, the mesh-based and functions-based approaches can be combined in a single reconstruction pipeline to leverage the advantages of both model types. For example, in the case of human reconstruction, a *SMPL* model can be used to represent a person's body, while the person's clothing can be expressed by *DIFs*, as shown in Figure 3.10*(bottom)*. Such a combination removes the need to have a separate mesh-based template for each type of clothing. At the same time, it does not discard the high-quality *SMPL* body model, which was learned on thousands of 3D scans.

3.2 Reconstruction Methods in Animal Modeling

Due to their abundance in the real-world scenes, animals were one of popular targets for the 3D reconstruction methods as soon as the first non-rigid techniques were introduced. The seminal work of Bregler [49], already mentioned in the Section 3.1.2, used the reconstruction of a giraffe from several RGB frames as an example for its *NRSfM* factorization technique. This early work was a good demonstration of the complexity of 3D reconstruction for non-rigid articulated bodies, as the proposed method was able to produce only a very crude partial reconstruction of the giraffe's neck, as shown in Figure 3.11.

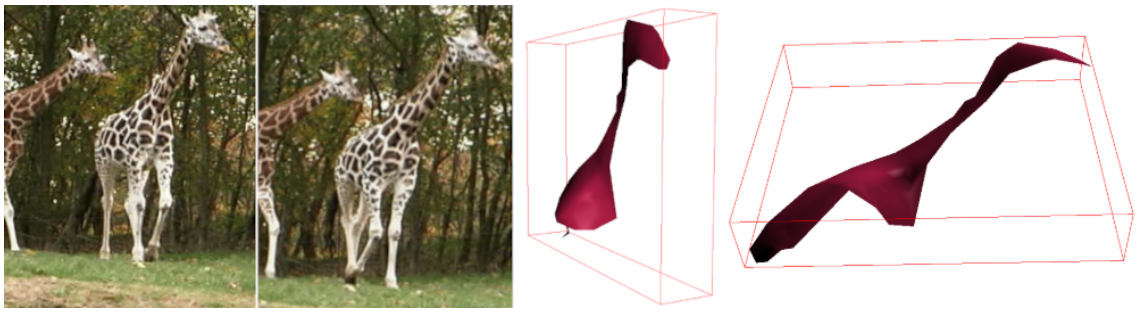

Figure 3.11: *Left*: input RGB frames. *Right*: reconstruction of the giraffe's body surface. Image taken from [49].

After the research on the non-rigid reconstruction began, a lot of works focused on the task of the human reconstruction due to its wider applicability and relative ease of the data collection [91][90][95]. The animal reconstruction drew a lot of inspiration from the methods created for humans, though many adjustments had to be considered due to the wider variability between the animal shapes, and the inability to collect high-quality input data, especially for the dynamic animal poses.

One of such early works proposed the application of morphable models for the lower-dimensional parametrization of the animal body [96]. It utilized multiple frames of the same animal class (but not of the same individual) to modify the input rigid template. The silhouettes and arbitrary user-clicked keypoints were used to constrain the energy-minimization procedure.

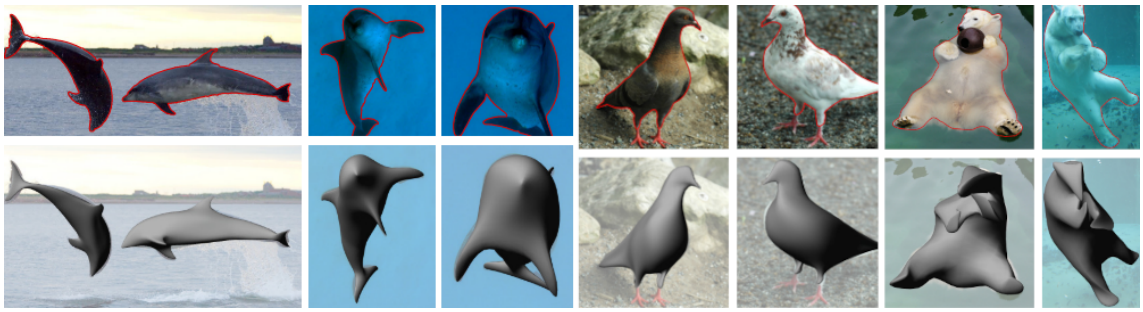

Figure 3.12: *Top*: input RGB frames and corresponding animal contours. *Bottom*: resulting 3D reconstructions. Image taken from [96].

The additional regularization terms prevented the resulting mesh from containing extreme deformations. This work, however, focused mostly on dolphins and suffered from inability to reconstruct different animal classes with higher joints articulation, as shown in Figure 3.12. The resulting 3D shape was often "spilled" outside

of the input silhouette, once projected back to the image, as such type of errors was not explicitly penalized in the proposed energy-minimization formulation.

Despite the relatively poor results, as compared to the modern State of the Art, the early works proposed many elements, which proved to be useful for the animal reconstruction. For example, user-clicked keypoints, pre-defined deformable templates and the per-frame silhouettes are still used as input to many modern animal reconstruction pipelines. Also, many studies started to focused on the 3D reconstruction of animals fr om a single image only, as such data was easier to obtain.

One of such studies, introduced in [55], proposed a template-based approach, relying on the reference-to-image point correspondences, silhouette constraints and area constraints. Same as some methods for general 3D reconstruction, many of which were applied to much simpler bounded planar objects, the inextensibility prior was assumed to constrain the deformation space. The results of this approach are shown in Figure 3.13.

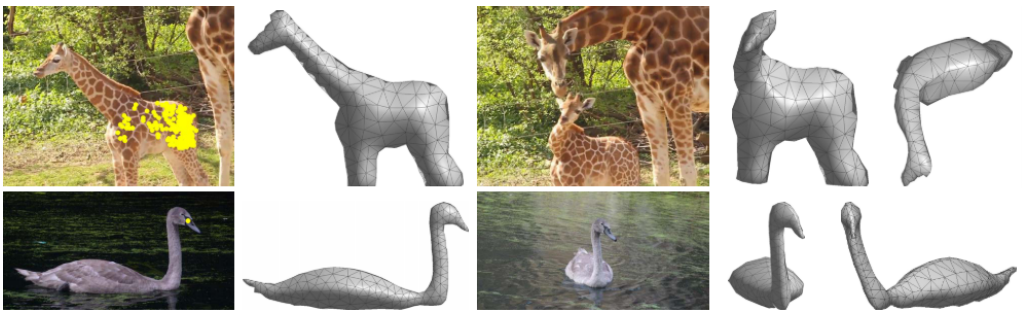

Figure 3.13: *From left to right:* (a) reference image with 2D-to-3D correspondences marked, (b) input template, (c) input RGB image, (d) resulting 3D reconstruction. Image taken from [55].

One more template-based approach, described in [54], once again relies on the input 3D template and user-clicked 2D keypoints. Additionally, a local stiffness concept is introduced to express the fact that some parts of the animal's body are less likely to deform than others. As can be seen from the results, shown in Figure 3.14, this method already manages to reconstruct animals in more complicated poses with plausible pose-related shape deformations. Still, the resulting 3D models are very general, representing the class-level animal template in the image-defined pose rather than an accurate reconstruction of a single individual.

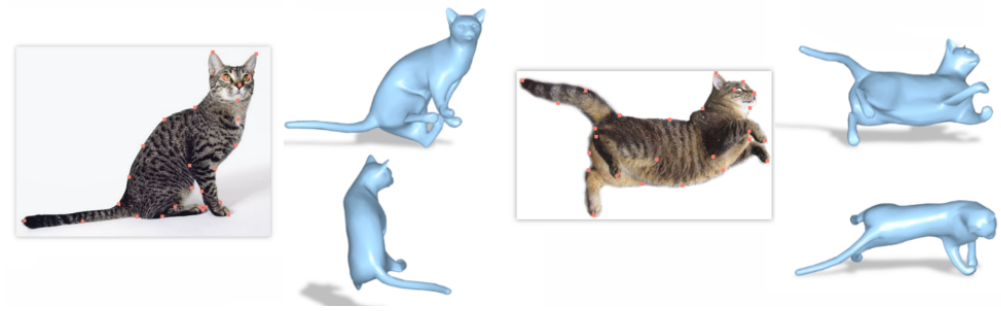

Figure 3.14: *Each left*: input RGB image with user-clicked keypoints. *Each right*: the resulting 3D reconstruction. Image taken from [54].

Another approach, presented in [76], proposes to simplify the reconstruction of an articulated object by subdividing it into simpler components, which are then reconstructed separately. Same as [96], this method does not attempt to create an accurate reconstruction of a single individual. Instead, it uses images of different animals to create a general but plausible class-level reconstruction in a reference pose. The example results are shown in Figure 3.15.

Figure 3.15: *Left*: input RGB images with corresponding contours of individual predefined body parts. *Center*: the reconstruction of individual parts. *Right*: the resulting 3D model. Image taken from [76].

Recently, many model-based approaches, similar to those in Section 3.1.4, were introduced [88][97][98][99][21][100], heavily inspired by the successes in human reconstruction task [86]. In [88], lack of the dataset with 3D animal scans is mitigated by scanning animal toy models. The energy minimization is then used to register these toys' scans to an artist-defined 3D template. This allows to define a parametric animal model, called *Skinned Multi-Animal Linear* (*SMAL*), whose parameter space represents the variation in animals' shapes and poses, with separate shape parameter clusters often corresponding to different biological species. The *SMAL* model is described in more details in Section 3.3 as it is used in the *SMAL4V* method, proposed in this thesis. An example reconstruction with *SMAL* fitting pipeline is shown in Figure 3.16*(left)*.

In [97], the *SMAL* is modified to allow for the multi-frame input, which reduces the shape ambiguity by providing multiple views of a single animal. Additionally, small per-vertex displacements are used to make the mesh projections better fit the input silhouettes, thus recovering finer details, like horns or antlers. The resulting method, called *SMAL with Refinement* (*SMALR*), significantly improves the reconstruction quality, as shown in Figure 3.16*(right)*. But, while creating good 3D models, both *SMAL* and *SMALR* pipelines require the ground-truth segmentations and 2D keypoints as input, which makes it very time-consuming to use these methods, for example, for video reconstruction.

Finally, the recently increased interest in deep learning, as well as multiple successes of the *Convolutional Neural Networks* (*CNNs*) [29][30][31] in different computer vision problems, has lead to creation of the first learning-based approaches for animal reconstruction. See Section 2.3 for some background knowledge required to understand more technical details of the deep-learning-based methods.

In [101], a *Creatures Great and SMAL* (*CGAS*) method is proposed, where *CNNs* are used independently of the rendering pipeline to remove the need for the user-provided keypoints and silhouettes. A state-of-the-art *DeepLabv3+* network [102] is used to predict the silhouettes from RGB images, and a stacked hour-

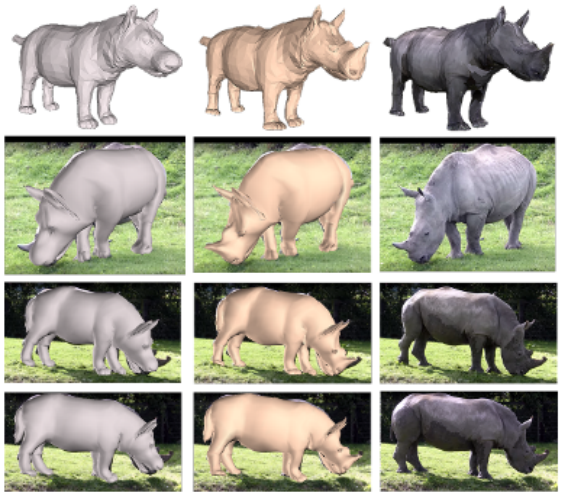

Figure 3.16: Comparison between the results of *SMAL (left)* and *SMALR (middle)* image fitting. Image taken from [97].

glass network [103] is trained on a synthetic dataset to predict 2D keypoints from those silhouettes. Moreover, since this method is created for video reconstruction, temporal information is used both for the correction of predicted keypoints and for model fitting. This allows to resolve the pose ambiguities and enforce smoother movements of the resulting 3D model. Still, the proposed *BADJA* dataset contains only 11 annotated videos, which is not sufficient for training of any deep model.

Figure 3.17: *From left to right:* input images, predicted silhouettes, predicted keypoints, post-processed keypoints, predicted mesh and the same mesh from another view, produced by *CGAS* pipeline. Image taken from [101].

SMALST method, introduced in [98], proposes to train a CNN on a synthetic zebras dataset so that the network learns to predict *SMAL* model parameters directly from RGB images. The end-to-end training of such a CNN is enabled by the *Neural Renderer*, which enables the gradient flow through the mesh rendering pipeline (for more details, see Section 2.3.3). In [99], the *SMAL* model is extended with limbs scaling terms, thus creating a new *Skinned Multi-Breed Linear Model for Dogs* (*SMBLD*) model. A new real-world dogs dataset is used to train (with 2D supervision only) a new *WLDO* pipeline to predict the *SMBLD* parameters. Both *SMALST* and *WLDO* methods are described in more details in Section 3.4.

Since *SMAL* can be used only for quadrupeds, another mesh-based model is introduced in [21] for 3D reconstruction of birds. With no 3D birds scans available, the shape and pose distributions have to be learned from a multi-view cowbirds dataset with extensive 2D annotations. Those distributions are then used to create 14,000 synthetic examples, on which a neural network is trained to predict the model parameters from keypoints and silhouettes. A separate network is trained to predict the keypoints and silhouettes from the raw RGB inputs. A full pipeline from [21] is shown in Figure 3.18.

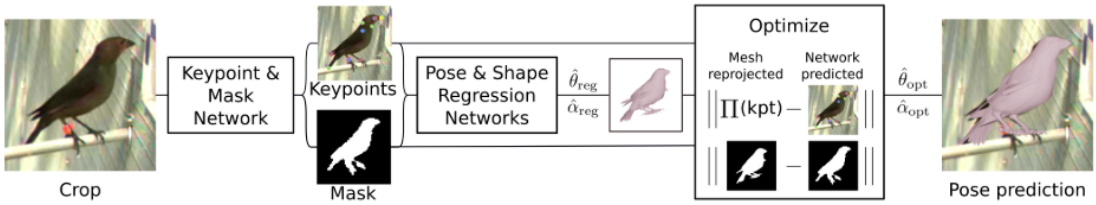

Figure 3.18: A 3D reconstruction pipeline for birds 3D reconstruction from a single-view image RGB input. Image taken from [21].

In [100], a model from [21] is extended with two scaling parameters and is then used as a 3D template for the energy-minimization-based fitting procedure, shown in Figure 3.19. There, the bird shape is modeled hierarchically: first as a difference between the template and the mean species shape, and then as a difference between the mean shape and the individual shape. By performing this model fitting on the whole *CUB-200* birds dataset, a new multi-species birds model, called *AVES*, is defined.

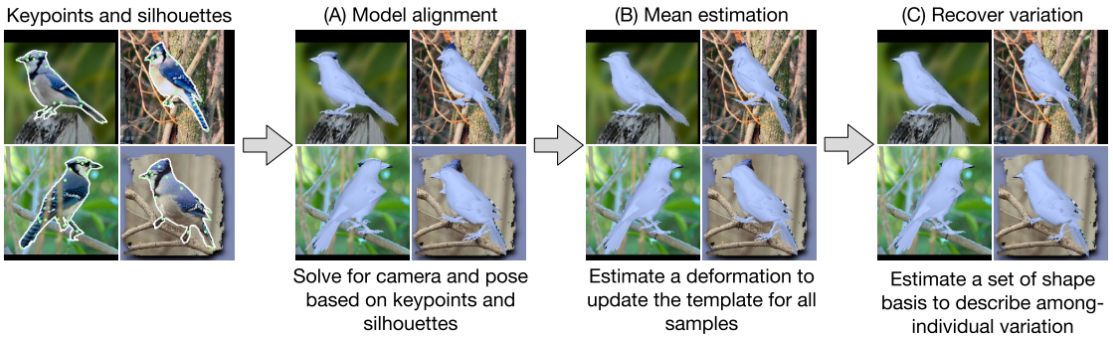

Figure 3.19: An overview of the hierarchical fitting procedure for birds reconstruction. Image taken from [100].

Some other deep-learning-based approaches instead focus on reducing the reliance on the 3D or multi-view data. In [51], the *CMR* method is trained on the *CUB-200* birds dataset to teach a neural network (with 2D-only supervision) to predict textured meshes from RGB images without having any 3D priors available. In [104], the *U-CMR* method is proposed, removing the need for 2D keypoint annotations but requiring an input 3D template. In [105], only silhouette supervision is used for training a multi-species reconstruction method. While these methods greatly reduce the requirements on the training set, their results are often poor and not suitable for real-world applications, as shown in Figure 3.20.

Figure 3.20: Example results for a multi-species reconstruction network trained only with silhouettes supervision. Image taken from [105].

One of the latest approaches, introduced in [106] shortly before the submission of this diploma thesis, combines several successful ideas into a single solution. It uses a state-of-the-art optical flow network [107] to find the dense correspondences, which are used as a supervisory signal instead of user-clicked keypoints. A pre-trained segmentation network [108] is used to provide the supervision for silhouettes. The perceptual similarity [109] is used to supervise the texture prediction. Finally, the bundle-adjustment-inspired optimization algorithm is proposed to refine the reconstruction results. Those steps unify the learning-based approach with the knowledge about animals' motion and geometry, producing high-quality results, as shown in Figure 3.21. Still, the resulting 3D models often suffer from unrealistic bending around the joints due to the lack of high-quality 3D pose priors.

Figure 3.21: Reconstruction examples for the differential deformation model. Image taken from [106].

An alternative approach simultaneously predicts camera parameters, 3D template's articulation and *Canonical Surface Mapping* (i.e. mapping between the 2D pixels and corresponding points on a 3D template), as shown in Figure 3.22. The enforced consistency between these predictions together with the (possibly-noisy) silhouettes serve as supervision for training of several deep networks [110].

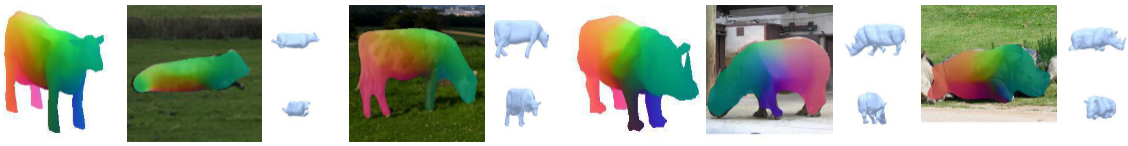

Figure 3.22: *From left to right:* a 3D template with colored surface points, predicted *CSM* and predicted mesh articulation from two views. Image taken from [110].

3.3 Skinned Multi-Animal Linear Model

Skinned Multi-Animal Linear (*SMAL*) [88] method proposes a solution to the lack of ground-truth 3D data by learning a parametric animal model on the 3D scans of the artificial animal figurines. An example of these toy 3D scans is shown in Figure 3.23. The *SMAL* model proposes a significant improvement over the earlier solutions, overview of which was provided in Section 3.2. It is heavily inspired by the successful *SMPL* human model. In the rest of this Section follows the brief summary of how the *SMAL* model is learned. Please, refer to the original *SMAL* paper for more details.

Figure 3.23: Example of 3D scans, obtained from the animal figurines, which are then used to train the *SMAL* model. Image taken from [88].

3.3.1 Global/Local Stitched Shape Model

The initial registration is done with the *Global/Local Stitched Shape* (*GLoSS*) model. It is a part-based globally differentiable model, which is fit to a 3D scan by the minimization of the following energy function:

$$E(\Pi) = E_m(\mathbf{d},\, \mathbf{s}) + E_{stitch}(\Pi) + E_{curv}(\Pi) + E_{data}(\Pi) + E_{pose}(\mathbf{r}). \tag{3.1}$$

$\Pi = \{\mathbf{l},\, \mathbf{r},\, \mathbf{s},\, \mathbf{d}\}$ is the parametrization of the *GLoSS* model, defined for the manually selected 33 parts of the animal body. It contains the $\mathbf{l}$ parameters for the individual parts' location, $\mathbf{r}$ parameters for the parts' *absolute* 3D rotation, $\mathbf{s}$ parameters for the intrinsic shape, and $\mathbf{d}$ parameters for the pose-dependent deformation.

E_m is a model term, encoding the symmetry constraints, the regularization of the pose deformation variables, and the distance from the shape distribution. E_{stitch} is a stitching term, based on the distance between the corresponding points of the neighboring body parts. E_{curv} term is a curvature point, enforcing the relationship between body parts to be similar to those in the template. E_{data} term contains the distances between the corresponding 3D keypoints, together with the model-to-scan and scan-to-model distances. Finally, E_{pose} is the pose prior for the tail.

The vertices of the *GLoSS* model are further refined by combining the E_{data} term with the *As-Rigid-As-Possible* (*ARAP*) regularization [111], and optimizing the mesh vertices in a model-free way by minimizing the following energy function:

$$E(\mathbf{v}) = E_{data}(\mathbf{v}) + E_{arap}(\mathbf{v}). \tag{3.2}$$

3.3.2 Fitting to 3D scans

Given the initial 3D fitting with the *GLoSS* model, all the meshes are brought to the same neutral pose. The *SMAL* model is then defined as the function $M(\beta, \theta, \gamma)$, where β is a vector of shape coefficients, θ is the *relative* rotation for the 33 joints,

and γ is the global translation of the root of the kinematic tree. The *SMAL* model is fitted to the 3D scans by minimizing the following energy function:

$$E(\beta, \theta, \gamma) = E_{pose}(\theta) + E_s(\beta) + E_{data}(\beta, \theta, \gamma), \tag{3.3}$$

where E_{pose} is the distance from the pose prior distribution, E_s is the distance from the shape prior distribution, and E_{data} is the same as in Section 3.3.1, redefined for the *SMAL* model parametrization.

The *SMAL* model is then again refined with a model-free optimization step, minimizing the energy function from Equation 3.2 together with a coupling co-registration term E_{coup} from [112]:

$$E(\mathbf{v}) = E_{data}(\mathbf{v}) + E_{arap}(\mathbf{v}) + E_{coup}(\mathbf{v}). \tag{3.4}$$

3.3.3 Fitting to images

The *SMAL* model parameters can then be inferred from the RGB data, as shown in Figure 3.24. Since the fitting is now done with respect to the 2D data, the projection of the 3D mesh into a 2D image plane is required. For that, the perspective camera model $\Pi(\mathbf{v_i}; f)$ is assumed, where f is the focal length, projecting the vertex $\mathbf{v_i}$ onto the image plane. Because the focal length is an unknown parameter, it is added to the *SMAL*'s parameter space, which is now denoted as $\Theta = \{\beta, \theta, \gamma, f\}$.

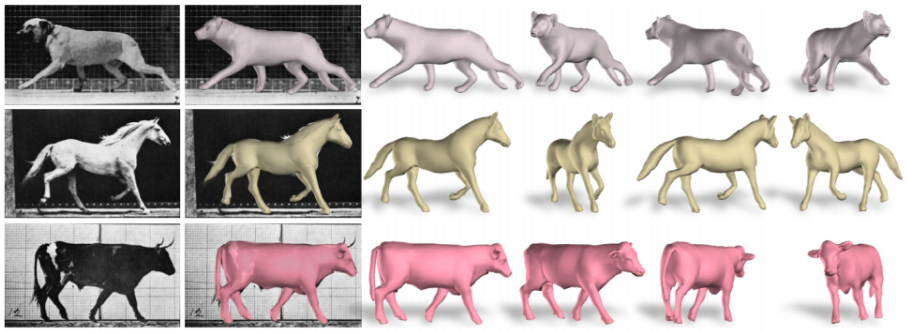

Figure 3.24: Example of the reconstructed animal shapes, fitted to the corresponding images. Image taken from [88].

To fit the mesh projection to an RGB image, the ground-truth 2D silhouettes and keypoints are required, which are manually prepared for each input image. The fitting is then done by the minimization of the following objective function:

$$E(\Theta) = E_{kp}(\Theta; x) + E_{silh}(\Theta; S) + E_\beta(\beta) + E_\theta(\theta) + E_{lim}(\theta), \tag{3.5}$$

where S is the ground-truth silhouette, E_{kp} is the keypoints reprojection error, E_{silh} is the silhouette reprojection error, E_β is the distance from a shape prior, E_θ is the distance from a pose prior, and E_{lim} is an additional constraint, which forces the predicted pose to stay within the predefined bounds.

3.4 End-to-End Learning Pipelines

Even though *SMAL* pipeline (as well as its extension, called *SMALR* [97]) provides high-quality animal reconstruction, it suffers from a serious drawback: it still

requires users to manually segment the image and annotate the 2D keypoints. The labelling process can be very time-consuming, especially if more frames per animal are considered. As mentioned in Section 3.2, several solutions were proposed to resolve this problem, such as the automatic segmentation with joints predictor [101] or the end-to-end learning approaches [98][99][105][110].

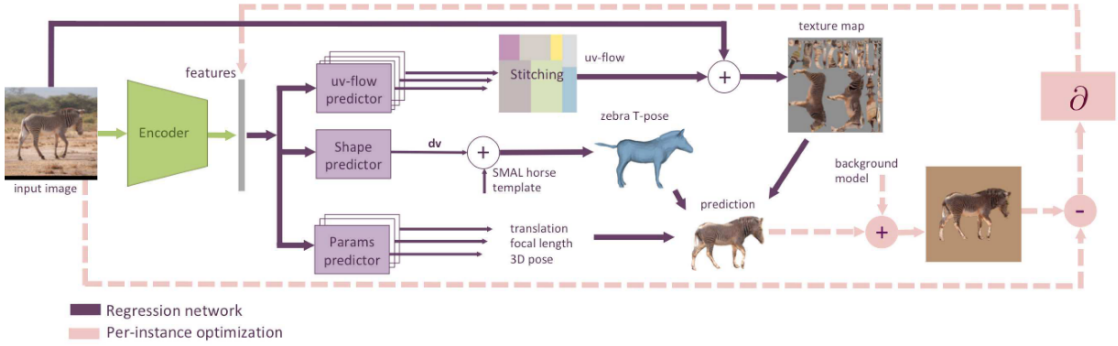

Figure 3.25: The *SMALST* modeling pipeline. Image taken from [98].

This thesis follows the ideas from the end-to-end methods: namely, from *SMAL with learned Shape and Texture* (*SMALST*), shown in Figure 3.25, and *Who Left the Dogs Out* (*WLDO*), shown in Figure 3.26. As can be seen from the figures, both create a 3D animal model from a single RGB image, without the need of providing any additional manual annotations at test time.

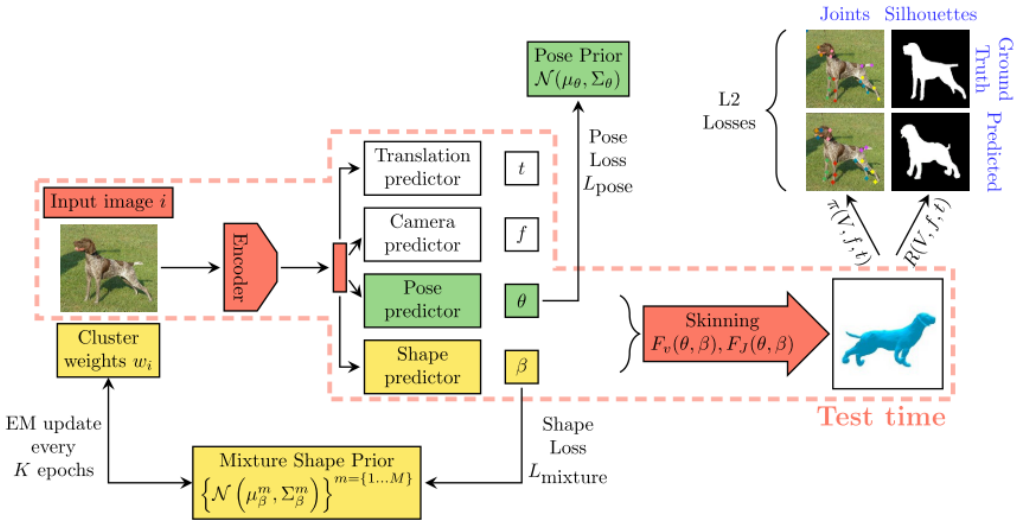

Figure 3.26: The *WLDO* modeling pipeline. Image taken from [99].

Both pipelines follow a similar design, inspired by the *Human Mesh Recovery* (*HMR*) architecture from [90]. The input image is first processed by a deep *CNN* encoder, extracting the features that are then used by separate multi-layer perceptrons to predict the camera (i.e. focal length), translation, shape and pose parameters. While most of predictors work similarly in both these methods, their shape predictors have important differences. In *SMALST*, the shape predictor outputs a long vector of vertex offsets **dv**, while in *WLDO* it outputs 24 blend-shape parameters (20 shape parameters from the original *SMAL* model, and 4 additional

limb scaling parameters). Also, *SMALST* contains a uv-flow predictor, which enables the output of the textured meshes, while *WLDO* completely drops the texture prediction. More importantly, there are significant differences in the datasets, on which these methods are trained.

SMALST uses a synthetic training set, where 12,850 zebra images are generated by projecting (with random camera parameters) different *SMAL* meshes on backgrounds from *COCO* dataset [113], as shown in Figure 3.27*(top)*. Since the dataset is synthetic, it provides the ground-truth shape, pose and camera parameters, allowing *SMALST* pipeline to be trained with 3D supervision. The supervisory vertex offsets $\mathbf{dv}_{gt}$ are derived from the ground-truth shape parameters: $\mathbf{dv}_{gt} = B_s\beta_{gt} + \mathbf{dv}_{gt}^{SMALR}$. The training procedure then minimizes the following loss function:

$$\begin{aligned} L_{train} = L_{silh}(S_{gt}, S) + L_{kp2D}(K_{2D,gt}, K_{2D}) + L_{cam}(f_{gt}, f) + \\ L_{img}(I_{input}, I, S_{gt}) + L_{pose}(\theta_{gt}, \theta) + L_{trans}(\theta_{gt}, \theta) + L_{shape}(\mathbf{dv}_{gt}, \mathbf{dv}) + \\ L_{uv}(\mathbf{uv}_{gt}, \mathbf{uv}) + L_{tex}(T_{gt}, T) + L_{dt}(\mathbf{uv}, S_{gt}), \end{aligned} \tag{3.6}$$

where the individual components are the silhouette loss L_{silh}, the 2D keypoint loss L_{kp2D}, the camera loss L_{cam}, the image loss L_{img}, the 3D pose loss L_{pose}, the translation loss L_{trans}, the 3D shape loss L_{shape}, the uv-flow loss L_{uv}, the texture map loss L_{tex}, and the texture loss L_{dt} from [51]. For more information about the individual loss components or overall training procedure, please consult the original *SMALST* paper [98].

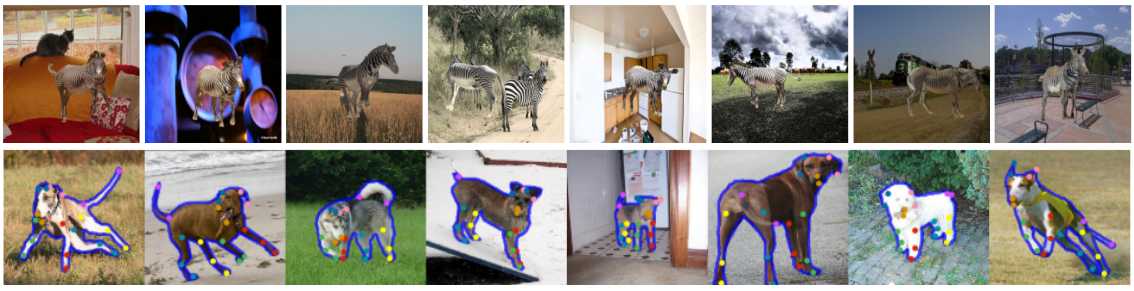

Figure 3.27: *Top:* synthetic zebra images, generated for the *SMALST* model training. Image taken from [98]. *Bottom:* real-world dog images from *StanfordExtra*, annotated for the *SMBLD* model training. Image taken from [99].

In the meantime, *WLDO* sticks to the real-world images and introduces the *StanfordExtra* dataset, which provides the 2D keypoint and silhouette annotations for the dogs' images from the *Stanford Dog Dataset* [114]. Total of 8,476 annotated images are available, with some examples shown in Figure 3.27*(bottom)*. While featuring different breeds, shapes and poses of dogs in complicated real-world scenarios, *StanfordExtra* does not provide any 3D supervision, forcing the training to be weakly-supervised. The training procedure is then implemented as minimization of the following loss function:

$$\begin{aligned} L_{train} = L_{silh}(S_{gt}, S) + L_{kp2D}(K_{2D,gt}, K_{2D}) + L_{pose}(\theta; \mu_\theta, \Sigma_\theta) + \\ L_{shape}(\beta; \mu_\beta, \Sigma_\beta) + L_{mixture}(\beta_i; \mu_\beta, \Sigma_\beta, \Pi_\beta), \end{aligned} \tag{3.7}$$

where the individual components are the silhouette loss L_{silh}, the joints reprojection loss L_{kp2D}, the distance from the pose prior L_{pose}, the distance from the shape prior L_{shape}, and the mixture shape loss $L_{mixture}$. The latter improve the quality of dogs reconstruction, implicitly providing distinct shape priors for different breeds [99].

Chapter 4

Problem Statement

As described in Chapter 3, there are several main problems with modern animal reconstruction methods. The approaches that utilize RGB-D cameras, described in Section 3.1.1, are often limited by the illumination conditions in the scene, filming distance, price of the hardware etc. From methods in Section 3.2, the end-to-end learning-based approaches seem like the most promising path as they do not require any additional annotations at test time, can be easily applied to any RGB images or videos, and produce plausible output meshes. Still, they suffer from multiple issues:

- Many of the presented approaches, including *SMALST* and *SMBLD*, use only a single-frame data for the reconstruction [98][99][105][110], which usually keeps many parts of the animal's body occluded, and often results into pose ambiguity. Both these issues could be addressed by utilizing the multi-frame [97] or video data [101][106], as demonstrated in *SMALR* or *CGAS* pipelines.

- *SMALST* [98], described in Section 3.4, is trained on a relatively homogeneous synthetic dataset, so the resulting network has problems generalizing to different animal species, complicated poses, different illuminations etc. Creating a more diverse synthetic dataset to re-train the network would be extremely challenging, especially for the video data, and would require a lot of work from experienced 3D animators.

- Real-world animal videos with annotated masks and keypoints are very scarce, with only a few datasets available, like *TigDog* [115] for tigers and horses (keypoints and approximate masks) or *BADJA* [101] for several animal species (only keypoints, just 11 videos in total). Without large-scale public video datasets, it is not possible to properly train and evaluate deep end-to-end reconstruction models.

The *SMAL4V* approach, proposed in this thesis, tries to address all these issues. A new dataset of cow videos, sparsely annotated with 2D keypoints and silhouettes, is introduced. It features different cow breeds, animals in complicated poses, scenes with occlusions, diverse weather and illumination conditions etc. A *SMALST* and *SMBLD* end-to-end pipelines are then extended to account for the motion information provided by the video data, resolving shape and pose ambiguities, while enforcing smooth motion between frames.

Chapter 5

Methodology

Despite the inherent ambiguity of the non-rigid 3D reconstruction, humans and animals are able to interpret rather complex 3D scenes. However, the ability for the scene understanding is not innate. Multiple experiments in neuroscience, which analyze the brain activity in the visual cortex, prove that vision is learned in the early stages of the animal development. The individuals, deprived of this learning possibility, are then functionally blind or experience serious problems with vision, despite the absence of any apparent physical damage to their eyes [116][117].

This suggests that the learning-based methods, utilizing *Artificial Neural Networks* (*ANNs*), could be a suitable approach to the 3D scene understanding and reconstruction. In such a scenario, training of an *ANN* roughly corresponds to the learning period in a living organism, and gradually incorporates prior knowledge about our world into the model. Even though this sounds straightforward to implement, modern solutions struggle from different issues, as discussed in Chapter 4.

This thesis tries to address those issues with the newly proposed *SMAL4V* method, closely related to the *SMALST* and *SMBLD* pipelines, described in Sections 3.3 and 3.4. The rest of this chapter is structured as follows: Section 5.1 introduces datasets used for training and evaluation of the proposed method, Section 5.2 describes the architecture of the *SMAL4V* pipeline, while Section 5.3 discusses how to train it with scarce 2D supervision.

5.1 Datasets

As already mentioned, the scarcity of annotated datasets is one of the main issues preventing the creation of high-quality animal reconstruction pipelines. At the time of writing, there are no public datasets of 3D animal scans available, making it impossible to train any deep model on the real-world data with strong supervision. While *SMALST* avoids this issue by generating synthetic training samples from *SMALR* mesh projections, it is hardly possible to generate plausible synthetic videos that would feature realistic motion patterns and complex environments. Moreover, *SMALST* has relatively low generalization abilities due to the simplicity of the synthetic data it was trained on. These issues thus call for the creation of new real-world large-scale datasets with 2D supervision.

5.1.1 StanfordExtra

The *StanfordExtra* dataset was originally proposed in [99] for weakly-supervised end-to-end 3D reconstruction of dogs. This dataset, based on the images from *Stanford Dog Dataset* [114], provides 2D keypoints and silhouettes annotations, examples of which are shown in Figure 5.1. In each image, 20 keypoints are labelled: 3 per leg, 2 per ear, 2 per tail and 2 per face. All labels are obtained from multiple annotators on *Amazon Mechanical Turk*. From original 20,580 dog images, only 8,476 are left after filtering out unreliable annotations or images with excessive occlusion. Also, this dataset features only single-frame data for each animal, making it impossible to infer any motion information. Still, this is the first large-scale dataset suitable for the end-to-end training of a 3D reconstruction pipeline.

Figure 5.1: Examples of images and their corresponding annotations from the *StanfordExtra* dataset. Image taken from [99].

5.1.2 TigDog

The *TigDog* dataset was proposed in [115] (and then updated in [118]) for the animal behavior analysis. Horses and tigers video sequences are used to infer animals' activity from their motion data. For that, 16,000 low-resolution horse frames are labeled, as well as 17,000 high-resolution tiger frames, with 19 keypoints annotated in each frame. Examples of annotated 2D keypoints are shown in Figure 5.2. Additionally, the approximate silhouette annotations are provided by applying a segmentation method from [119]. Considering the dataset size, it can be used for both training and evaluation of the deep models.

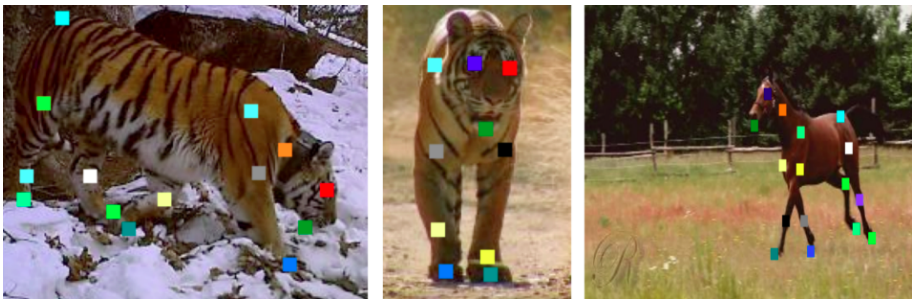

Figure 5.2: Examples of images and their corresponding annotations from the *TigDog* dataset. Image taken from [115].

5.1.3 Benchmark Animal Dataset of Joint Annotations

Benchmark Animal Dataset of Joint Annotations (*BADJA*) dataset was released together with *CGAS* pipeline in [101]. Its main motivation is providing a benchmark for qualitative comparison of animal joints prediction methods. It consists of 11 videos where 20 keypoints are annotated every 5-th frame (except for one video where every frame is annotated). The annotations examples are shown in Figure 5.3. However, while providing high-quality labels, this dataset is too small to train any 3D reconstruction pipeline. It can be, however, used for qualitative evaluation on videos.

Figure 5.3: Examples of images and their corresponding annotations from the *BADJA* dataset. Image taken from [101].

5.1.4 AnimalKey

Due to the scarcity of the suitable training data, a new *AnimalKey* dataset is proposed as part of this thesis. It is based on hundreds of videos of cows, which are downloaded from several free sources, such as *Pixabay*, *Videvo* and *Pexels*. Since not enough free videos were available on these free resources, *Our Wyoming Life*, *The Funky Farmer* and *Soil Mates of Georgia* YouTube projects are used as the additional video sources (with permissions from the video owners).

A set of manually selected frames is annotated with silhouettes and 30 keypoints, with example frames shown in Figure 5.4. At the time of submission, 900 frames are annotated with keypoints, from which 700 also have annotated silhouettes. In total, 519 short video sequences are prepared, with each having at least one frame fully annotated. Keypoint trackers, such as *DeepLabCut* [120], or video-based segmentation networks, such as *OSVOS* [38] can then be used to extend those sequences with approximate annotations for other frames. The nature of the dataset thus allows it to be used for multiple computer vision tasks.

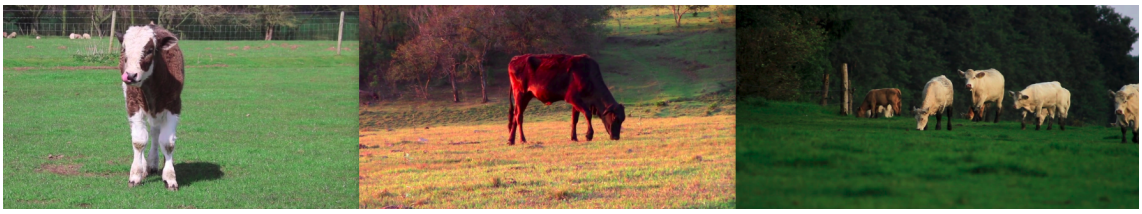

Figure 5.4: Examples of images from the *AnimalKey* dataset.

To facilitate the labeling process, a simple web-page, whose interface is shown in Figure 5.5, was prepared. Due to financial limitations, all the annotations were performed by the author of this thesis. Source code for the labeling page is made

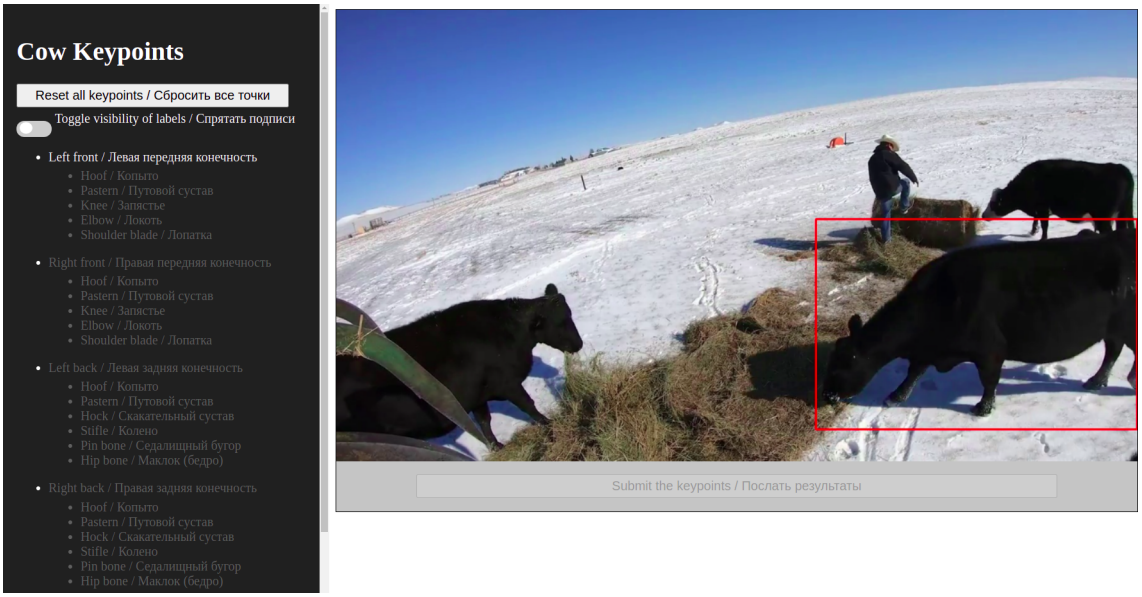

Figure 5.5: A screenshot of the labeling interface, prepared for the keypoints collection.

available at *https://github.com/iegorval/keypointsLabellingTool*. The segmentations and bounding boxes are annotated with *Supervisely* and *LabelMe* tools, respectively.

The resulting dataset is released to the research community. The download links and further information are available at *https://github.com/iegorval/AnimalKey*. It is planned to regularly update the dataset after the submission of this thesis, especially if the further research is financed. Several hundreds of unlabeled short sequences, featuring cows, are already prepared to be labeled. There are also plans to add different animal species to the dataset so that more researchers can benefit from it.

5.2 Proposed Pipeline

This thesis proposes the *SMAL for Videos* (*SMAL4V*) method for 3D animal reconstruction from video sequences to address the issues outlined in Chapter 4. It falls in the *SMAL* methods family, together with *SMALST*, *SMBLD* and *SMALR*. The proposed approach is also heavily inspired by the *One-Shot Video Object Segmentation* (*OSVOS*) segmentation method [38], and, though created independently, is somewhat similar to *VIBE* human reconstruction pipeline [92]. An overview of the proposed *SMAL4V* pipeline is shown in Figure 5.6.

Same as *OSVOS*, it is separated into three stages, each of which outputs a single mesh predicting network. The first stage outputs a so-called *base network*, which corresponds to the *WLDO* mesh predictor from [99], trained on *StanfordExtra*. Unlike the *OSVOS*'s base network, it is trained on the same task as the other two networks, though on different animal species.

Then, the *parent network* keeps the same mesh predictor from *WLDO* but fine-tunes it on *AnimalKeys*, i.e. on cows. This is done on the independent frames from *AnimalKey*, without utilizing any temporal information since *WLDO*'s mesh predictor is not designed to work with video sequences.

The availability of the pre-trained base network significantly reduces the training data and time requirements, which was critical condition for this thesis. While

Figure 5.6: Overview of the proposed *SMAL4V* pipeline. Created with *draw.io* tool. Similarity with diagram design from [38] is intended to highlight the analogies between two approaches.

WLDO's mesh predictor had to be trained for 96 hours on approximately 6700 labeled images, *SMAL4V*'s parent network is fine-tuned for a few hours on XXX images. In both cases, the training is performed on a single P100 GPU.

Finally, a *test network* is proposed as a main contribution of this thesis. Its architecture is shown in Figure 5.7 and is described below. It expands the *WLDO*'s mesh predictor to enable temporal data processing. Many design decisions are heavily influenced by the need to reduce the computational time and annotation effort required for training.

5.2.1 Architecture of the test network

The *AnimalKey* dataset, as described in Section 5.1.4, contains short video sequences (stored as collections of frames) with sparse keypoint and silhouette annotations. A single sequence is selected, and its frames are sent to the pre-trained encoder from the parent network. The last feed-forward layers of the *WLDO*'s encoder are removed, and a pre-trained fully-convolutional encoder is left. Same as in *VIBE*, it processes each frame individually to create a sequence of encoded features (here, of length 18,432).

Then, this sequence of features is split into the windows (here, of size 8), and is fed to the *temporal encoder*, implemented with *Recurrent Neural Networks* (*RNNs*). In this thesis, a single-layer bidirectional *RNN* with gated-recurrent units of hidden size 1024 is used for the visualization of results. The temporal encoder produces a single output for each frame, i.e. a new feature vector for each frame (now, of size 1024). Thanks to the recurrent architecture, a sequence of any length can be processed in this way.

Now, the encoded frames are fed to the parent network's pre-trained pose, camera, translation and shape predictors. They stick to the *WLDO*'s architecture, mostly due to the inability to train a different set of predictors with the currently-available dataset. Since *WLDO*'s pose, camera, and translation predictors require input of size 100, a linear layer is inserted between them and the temporal encoder,

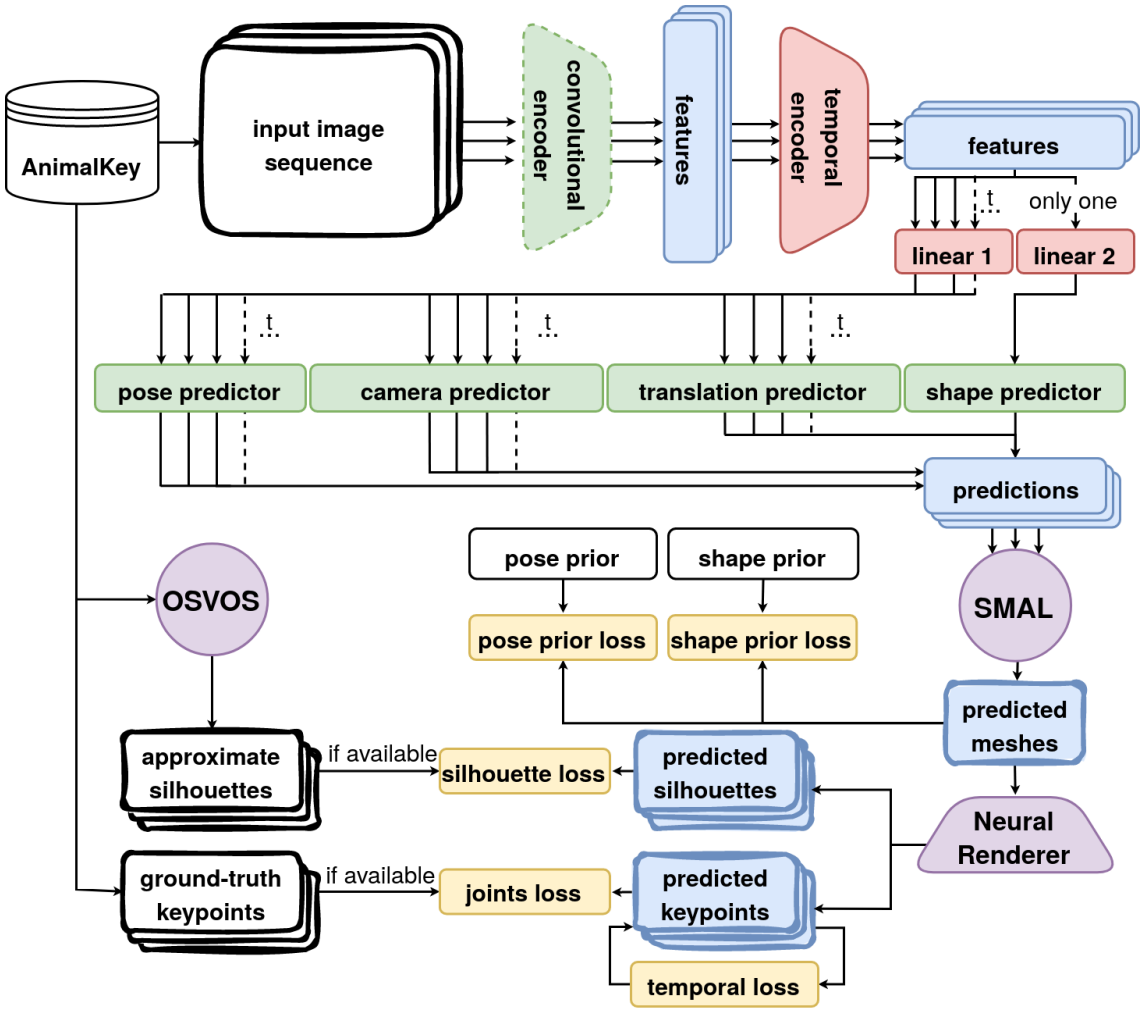

Figure 5.7: Proposed architecture of the *SMAL4V*'s test network. Created with *draw.io* tool. Best viewed in color. *White*: available input, *green*: pre-trained from the parent network (*dotted border* indicates frozen weights), *red*: layers specific to the test network, *blue*: intermediate and final network outputs, *yellow*: individual loss terms, *violet*: models that are used as black-boxes in the proposed pipeline.

reducing the feature length from 1024 to 100. Similarly, because *WLDO*'s shape predictor requires input of size 18,432, a linear layer with input size of 1024 and output size of 18,432 is inserted between the recurrent layer and the predictor.

The temporally-encoded sequence of features are thus independently sent to the predictors through the corresponding linear layers. Also, to account the fact that a cow's shape is unchanged throughout the video sequence, only the last element of this sequence is sent to the shape predictor. In this way, a single animal shape is outputted for all the input frames.

All predictions are then sent to the *SMAL* model, which outputs a posed mesh for each input frame. The *Neural Renderer* is then used to get the predicted keypoints and silhouettes, in the same manner as it is done in both *WLDO* and *SMALST* pipelines.

5.3 Training Procedure

The base network is downloaded from *https://github.com/benjiebob/WLDO* and kept as is. Then, the parent network is trained on independent frames from *AnimalKey* with almost the same procedure as proposed in [99] (for more details, see Section 3.4). Because cows do not have such a strong variety in shapes as dogs, the *SMBLD* limb scaling parameters are omitted, and the standard *SMAL* shape prior is used instead of the multi-model shape prior from *WLDO*.

As mentioned in Section 5.2, the test network is fine-tuned to a single video sequence only, inspired by *OSVOS*. This choice is mostly influenced by the insufficient annotations available for the multi-sequence temporal training. With the current dataset size, the model would definitely overfit to the training set. Still, the test network architecture is designed in a way that allows training on multiple video sequences, same as in *VIBE*, once the *AnimalKey* dataset is expanded. Thus, the multi-sequence training is kept for the future work.

The single-sequence fine-tuning of the test network is done by minimizing the following loss function:

$$L_{train} = \Sigma_{t=0}^{T} L_{silh}(S_{osvos}, S) + \Sigma_{t \in Fanno} L_{kp2D}(K_{2D,gt}, K_{2D})) + \Sigma_{t \notin Fanno} L_{silh}(S_{osvos}, S) + \Sigma_{t=0}^{T} L_{pose}(\theta_t; \mu_\theta, \Sigma_\theta) + L_{shape}(\beta; \mu_\beta, \Sigma_\beta), \tag{5.1}$$

where F_{anno} is a subset of frames with available ground-truth annotations and F_{osvos} is a subset of frames annotated by *OSVOS*. The meanings of individual loss terms are explained below.

L_{silh} is the L1 loss between the predicted and *OSVOS*-generated silhouettes, averaged over all pixels. L_{kp2D} is the L1 loss between the predicted and ground-truth 2D keypoints, computed only for those keypoints that are marked as visible. L_{pose} is the pose prior loss, computed as a Mahalanobis distance between the predicted pose vector and the pose prior distribution from *SMAL*. L_{shape} is the shape prior loss, again computed as a Mahalanobis distance between the prediction and the prior *SMAL* distribution. Note that, unlike the other terms, L_{shape} is not summed along the input sequence, since a single shape is produced by the *SMAL4V* pipeline.

Since such online training of the test network corresponds to its test time, it implies that *SMAL4V* requires at least sparse annotations at test time. Though not ideal, this provides a compromise between *SMALST*/*SMBLD* and *SMALR* approaches. While the former do not require keypoints and silhouette annotations at test time, they also do not utilize any multi-frame or temporal information. From the other side, *SMALR* would require annotations at every frame, making it impossible to use it on any longer video sequences. Also, note that all the mentioned approaches additionally require bounding boxes at test time, same as the related approaches for the human reconstruction [92].

Chapter 6

Experimental Results

This Chapter presents all the main experiments that were performed as part of this thesis. Section 6.1 once again discusses the need for the specially-designed animal reconstruction models by showing how the classical pipelines are insufficient for the given task. Then, Section 6.2 demonstrates the results of the *SMAL* family of animal reconstruction methods (for more details, see Sections 3.3 and 3.4), inluding newly proposed *SMAL4V*.

6.1 Preliminary Experiments

As described in Chapter 3, there exist dozens of reconstruction methods with different data requirements and results quality. Probably the most obvious to try are off-the-shelf photogrammetry pipelines, which are successfully used in industry. Another seemingly promising direction is RGB-D sensors, which are constantly growing in popularity, with many cheaper models available nowadays (and some of them even installed in the modern smartphones). Devices with such sensors can increase the reconstruction quality by capturing real depth information from the scene. This Section demonstrates results from experimenting with both these directions. The described experiments were performed in parallel with studying State of the Art for animal 3D reconstruction.

6.1.1 Reconstruction with rigid techniques

As explained in Chapter 2, 3D reconstruction of the rigid scenes is a well-constrained geometrical problem, which let to multiple off-the-shelf solutions, both commercial and open-source, being available. The rigid techniques rely on finding the image correspondences and, based on their positions in the image, finding the original 3D scene points by re-projection. Assuming that the motion of a scene is nearly-static, it is sometimes possible to apply those methods to real-world scenarios despite the presence of some non-rigid objects. The purpose of the first experiment is thus to try the rigid solutions on some example scenes of interest and report the observed performance.

The data for the experiments in this section include several video sequences of different cows and horses. While all of the collected data cannot be shown due to the space limitation, some examples are shown in Figures 6.1, 6.2, and 6.4. Multiple issues with such animal data are directly obvious. First, the animal has to be observed from all sides to be properly reconstructed by general-purpose pipeline, which

do not incorporate any prior knowledge about the animal species being processed. This typically requires around one minute of recording, during which most animals are unable to keep static.

Figure 6.1: Example frames from a horse video, collected with an Apple iPhone 12 Pro.

Figure 6.2: Example frames from a horse video, collected with a Samsung S10.

While horses, which are usually well-trained, can be made to kept almost static by a human assistant, this cannot be done for most of animals, even for domesticated ones. For example, cows, which are heavily used in agriculture industry, have to be filmed while in feeding fences or milking robots, which provide additional occlusion. Also, since the untrained animals cannot be made to go to such locations, it also means that often they cannot be properly filmed at all.

Figure 6.3: Output of a keypoint detector on a frame from a horse video.

Second, many animals are mostly untextured, which is clearly seen in Figures 6.1 and 6.2. This confuses many keypoints detectors, making it complicated to reconstruct animals even if it is kept static. Example output for one of such keypoints

detectors is shown in Figure 6.3. It demonstrates once again why classical pipelines, based on epipolar geometry, can struggle with animal reconstruction even if the animal is kept static during the capturing process.

Finally, illumination is often an issue, as shown in Figure 6.4. Many barns do not have good artificial illumination, making it hard to collect high-quality video sequences. Often, some sides of the animal stay almost invisible due to insufficient illumination.

Figure 6.4: Example frames from a cow video.

The *Reality Capture* photogrammetry pipeline is tested on the collected data. In Figure 6.5, the reconstruction results for two horse video sequences are shown. In the top row, the animal was kept as static as possible, while in the bottom row it was allowed to slightly move. Notice how most of the animal body cannot be reconstructed anymore even though the movement between frames was relative small. It clearly shows that, if we are to reconstruct freely moving animals, more specialized methods have to be created.

Figure 6.5: *Top:* 3D model obtained from Reality Capture on a nearly-static animal. *Bottom:* significantly decreased reconstruction quality once the animal slightly changed its pose between the frames.

In Figure 6.6, example reconstructions for two cow sequences are shown. Again, in the top row animal was nearly-static, as it was filmed during the feeding process. While a relative good reconstruction quality is achieved in that case, it is often impossible to catch an untrained animal somewhere where it voluntarily does not move. In the bottom row, a reconstruction for a freely-moving cow is shown. While 25 input images were used, only 3 cameras were properly reconstructed due to the several pose and background changes between the individual frames.

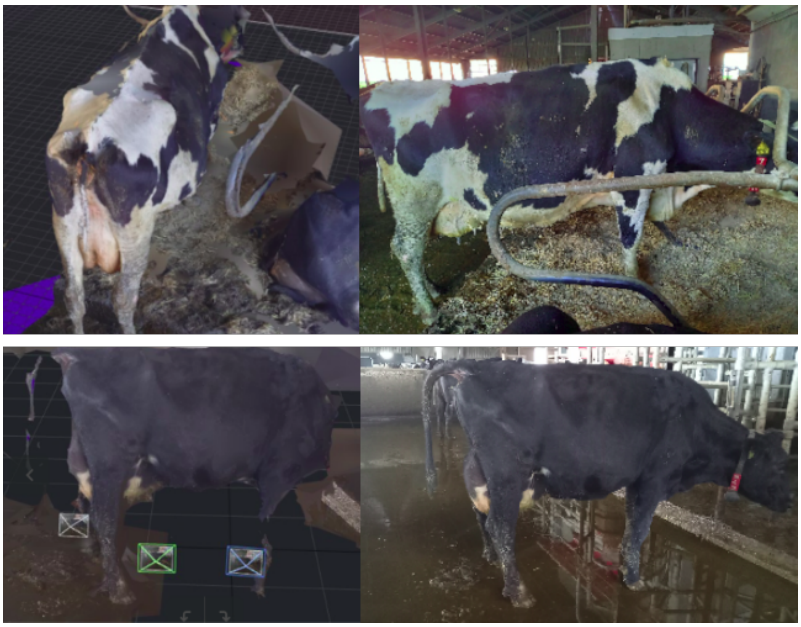

Figure 6.6: 3D model obtained from Reality Capture on: nearly-static animal *(top)*, freely-moving animal *(bottom)*.

Based on these preliminary RGB experiments, it was confirmed that other approaches to animal reconstruction have to be researched and tested.

6.1.2 Reconstruction with RGB-D cameras

At the stage of the preliminary experiments, reconstruction with RGB-D cameras was also considered as one of the possible directions for this thesis. The depth sensors provide valuable information to the reconstruction pipeline but suffer from several drawbacks. Most of the cheaper depth sensors do not work under natural illumination (for more details, see Section 3.1.1), severely limiting the environments where animals can be recorded. Also, quality of a single raw scan is usually not sufficient so that a target has to be recorded for longer periods of time. Same as with RGB-only data, this is difficult to achieve with animals who are rarely collaborating with the data acquisition process.

Intel RealSense D415, Intel RealSense D435 and ASUS Xtion were tried in the field for the data collection. Unfortunately, their quality was insufficient to record outdoor sequences of moving animals in sufficient quality. A mixed-reality headset was briefly tried on a single animal in the outdoor environment, with the result shown in Figure 6.7. While seemingly providing high-quality 3D information, it was only possible to record such data on a completely static animal.

With the idea of simplifying the RGB-D data acquisition process, the newly-released Apple iPhone 12 Pro was tried for data collection. It features a built-in

Figure 6.7: 3D data obtained with a *HoloLens* mixed-reality headset.

LiDAR sensor, allowing for the depth-based reconstruction to be performed directly on device. A small iOS app was written to extract the raw depth maps from the ARCore SDK. Also, an off-the-shelf *Scaniverse* mobile application, which uses the LiDAR data in its pipeline, was tried for the 3D reconstruction. Example reconstructions from this app are shown in Figure 6.8. Despite the depth information available, the pipeline still fails for a moving animal, once again indicating the need for designing special reconstruction methods for this task.

Figure 6.8: 3D model obtained from *Scaniverse* on: nearly-static animal *(top)*, freely-moving animal *(bottom)*.

So, based on the results of preliminary RGB-D experiments, it was decided to continue with RGB-only data due to its abundance. Even though a large 3D dataset would be very useful for the research community, it would most likely require a bigger team of data collectors and more finances for the required equipment. The research on the 3D reconstruction with RGB-D cameras is thus left to future work.

6.2 Experiments with Parametric Models

The *SMAL* [88] and *SMBLD* [99] parametric models were described in Sections 3.3 and 3.4. Learned on a small set of 3D toy scans or of artist-defined models, correspondingly, they provide strong shape and pose priors that allow 3D reconstruction even from a single RGB image. The original *SMAL* pipeline, as well as its *SMALR* extension [97], provide plausible reconstructions but require ground-truth silhouettes and keypoints for every frame. This makes them unusable for any longer video sequences. It is unfortunate as video data not only provides a way to resolve shape and pose ambiguities but can also provide valuable information for behavior analysis or for generation of plausible computer animations.

So, learning-based *SMALST* and *WLDO* pipelines are chosen as the basis for the rest of experiments in this Section as they do not require dense annotations at test time (except for the bounding boxes). The datasets, described in Section 5.1 are used for the qualitative and quantitative comparison of tested methods.

For the quantitative comparison, *Percentage of Correct Keypoints* (*PCK*) and *Intersection-over-Union* (*IoU*) metrics are reported. *PCK*, as follows from the name, is percentage of correctly predicted keypoints, where a keypoint is considered correctly predicted if its distance from the ground truth, normalized by the object's silhouette area, is within some threshold α. In that case, *PCK* is written as $PCK@\alpha$, eg. $PCK@0.1$. The correctness of the rendered silhouettes can be measured by the *IoU* metric, calculated as $IOU = \frac{S_{gt} \cap S_{pred}}{S_{gt} \cup S_{pred}}$.

6.2.1 Experiments with the SMALST pipeline

Code for *SMALST* was downloaded from *https://github.com/silviazuffi/smalst*. However, due to the out-of-dated Python packages used, it was not possible to directly use the provided code. Those packages had unsolvable problems with the modern CUDA versions, and older CUDA drivers were not supported by the used hardware. Thus, with the permission from the *SMALST*'s author, most of the code base was translated into Python 3, with as modern package versions as possible. This translated code is stored as a subset of *SMAL4V* repository so that these models can be effortlessly used in a single environment. Parts of the code that are guarded by the Max Planck Institute's licence are not publicly re-distributed and are left unmodified. Instead, the download links to them is provided in the *SMAL4V* repository at *https://github.com/iegorval/smal4v*.

A pre-trained *SMALST* model is downloaded and used for the experiments performed in this thesis. However, to verify that the *SMALST* code base was properly refactored, it is evaluated on the *Grevy's Zebras* dataset so that results can be compared to those reported in the original *SMALST* paper [98]. This comparison is shown in Table below. Note that the achieved metrics are lower than whose reported in [98]. Since most of the used Python packaged have been updated to versions, different from those used by the *SMALST* authors, it is not surprising that the original result from [98] is not easily reproducible. Still, most of the outputted meshes seem to be decently reconstructed, as shown in Figure 6.9.

SMALST	PCK@0.1	IoU
from [98]	80.3	0.416
achieved	62.2	0.292

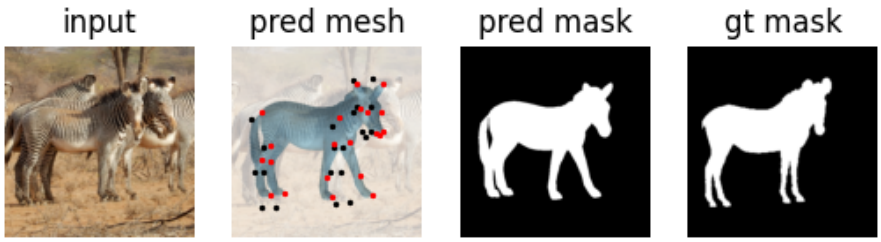

Figure 6.9: Example results of the *SMALST* pipeline on the *Grevy Zebra*'s test set. *From left to right:* cropped input image, overlayed predicted mesh (predicted keypoints in red, ground-truth keypoints in black), predicted mask, ground-truth mask.

6.2.2 Experiments with the WLDO pipeline

Code for *WLDO* was downloaded from *https://github.com/benjiebob/WLDO*. A mesh predictor, pre-trained on the *StanfordExtra* dataset, is taken from the same source. Since this thesis aimed at comparable evaluation between different models, the *WLDO* data loader and mesh predictor were adapted to the refactored *SMALST* format and integrated in the *SMAL4V* repository. Again, to validate the correctness of the created code, the pre-trained model is evaluated on the *StanfordExtra* test set, and the results are compared to those from [99], as shown in Table below. Note that the reported *PCK* metrics differ between the original *WLDO* article and their github repository. The *PCK*@0.15, as reported in the repository, is chosen because it is more likely to be updated and corresponding to the released model version. Once again, the reported and achieved metrics slightly differ, which can be caused by different *SMAL* model templates used or different *PCK* computation procedures.

WLDO	PCK@0.15	IoU
from [99]	67.5	74.2
achieved	59.3	75.3

6.2.3 Comparison on *AnimalKey* and *TigDog*

To test the generalization abilities of the *SMALST* and *WLDO* pipelines, they are now tested on the *AnimalKey* and *TigDog* datasets without fine-tuning. For this and all the following experiments, 102 annotated images from *AnimalKey* (sequences 295-297, 301-302, 306-307, 313, 343, 352-353, 460-461, 476, 484 and 589) are used as a test set for the single-frame evaluation. Predictions for several selected frames are shown in Figure 6.10 for *SMALST* and in Figure 6.11 for *WLDO*.

Note that two tested pipelines exhibit different nature of errors. *SMALST* clearly has problems to generalize to non-zebra shapes and to previously unseen camera and animal poses. Thus, it often outputs a mesh facing forward of backward, far from the actual position of the animal. Instead, *WLDO* is often able to correctly capture the pose of a cow even though it was not trained for them. This supposes better generalization ability, which is not surprising considering that *WLDO* was trained on a large-scale real-world dataset.

For *TigDog*, 10% of the dataset is taken for evaluation purpose. While bigger than *AnimalKey*, *TigDog* contains lower-resolution images and many erroneous bounding boxes. Extremely small images (i.e. those that are at least twice smaller than the mesh predictor's required input) and images with missing annotations are

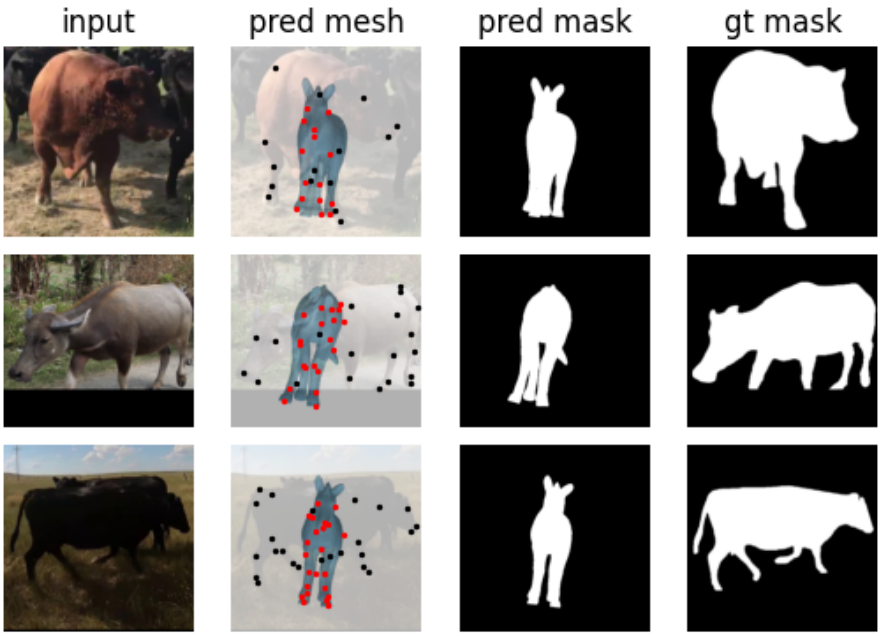

Figure 6.10: Example results of the *SMALST* pipeline on the *AnimalKey*'s single-frame test set. *From left to right:* cropped input image, overlaid predicted mesh (predicted keypoints in red, ground-truth keypoints in black), predicted mask, ground-truth mask.

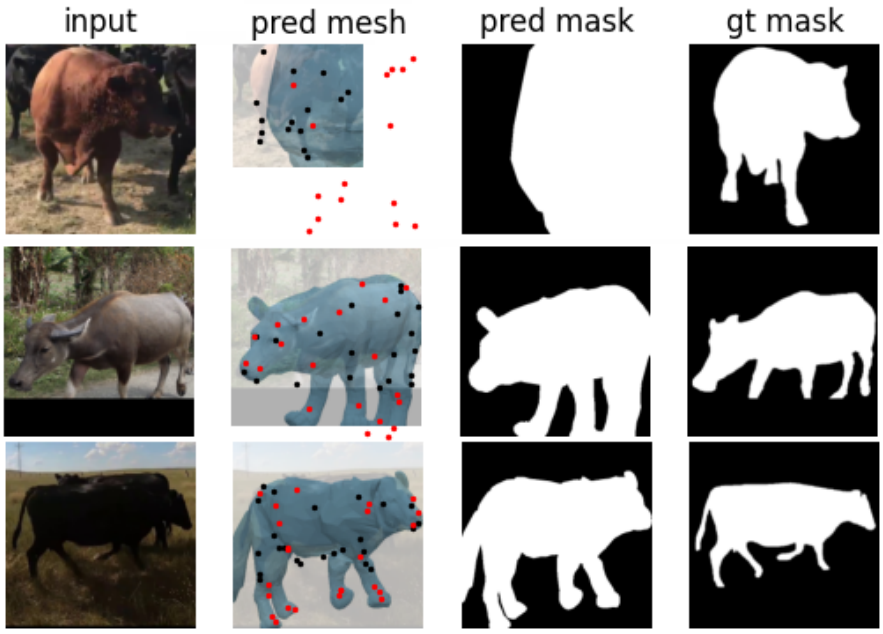

Figure 6.11: Example results of the *WLDO* pipeline on the *AnimalKey*'s single-frame test set. *From left to right:* cropped input image, overlaid predicted mesh (predicted keypoints in red, ground-truth keypoints in black), predicted mask, ground-truth mask.

filtered out. This results into approximately 500 annotated test images available (exact number of test images depends on the size of the network's input). Some selected results are shown in Figure 6.12 for *SMALST* and in Figure 6.13 for *WLDO*. Note how the quality of the *TigDog*'s silhouette annotations is significantly lower than that of segmentation masks from the *AnimalKey* dataset. Because of that, only *AnimalKey* is used to train the proposed *SMAL4V* method.

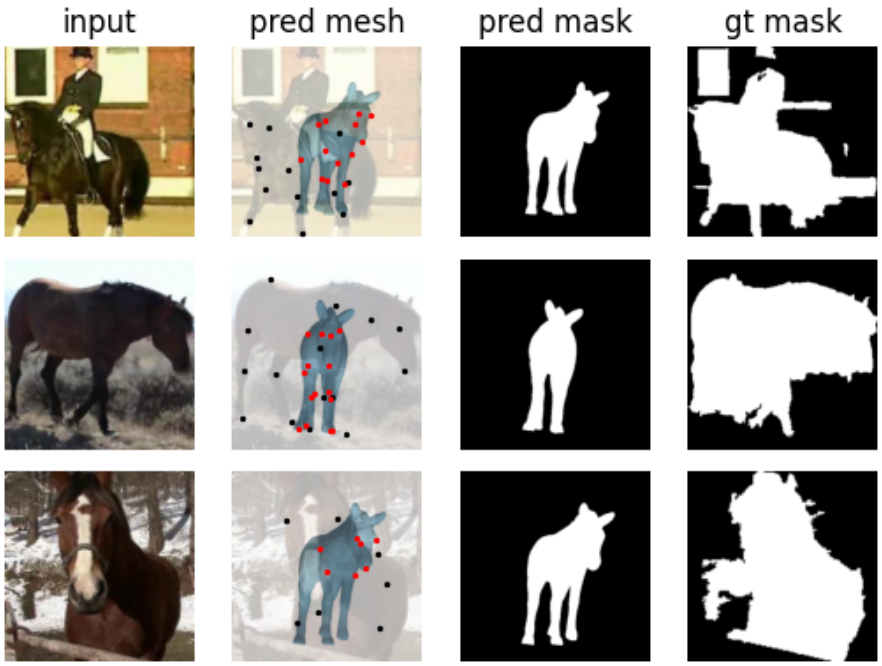

Figure 6.12: Example results of the *SMALST* pipeline on the *TigDog*'s single-frame test set. *From left to right:* cropped input image, overlayed predicted mesh (predicted keypoints in red, ground-truth keypoints in black), predicted mask, ground-truth mask.

Same as with *AnimalKey*, *SMALST* experiences generalization issues and have problems capturing the pose of the animals. At the same time, *WLDO*, though sometimes failing, better captures the poses of unknown animals thanks to its more diverse training dataset.

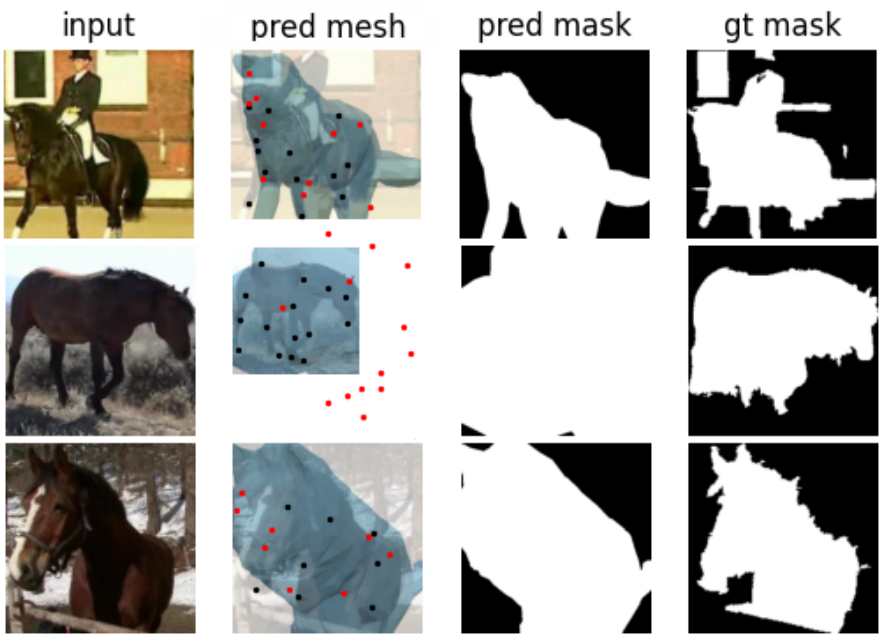

Figure 6.13: Example results of the *WLDO* pipeline on the *TigDog*'s single-frame test set. *From left to right:* cropped input image, overlaid predicted mesh (predicted keypoints in red, ground-truth keypoints in black), predicted mask, ground-truth mask.

The quantitative comparison of these two methods is shown in the Tables below.

AnimalKey	PCK@0.10	IoU
SMALST	6.1	23.6
WLDO	7.1	60.0

TigDog	PCK@0.10	IoU
SMALST	3.5	23.3
WLDO	7.4	52.7

As shown, the *WLDO* pipeline has better generalization abilities. This is why its version, pre-trained on *StanfordExtra*, was chosen as a base network for the proposed approach. The parent network is then created by fine-tuning it on a subset of the *AnimalKey* dataset. Then, the online training procedure, described in Section 5.3, is applied to individual video sequences with 2D keypoints annotations and dense silhouettes supervision from *OSVOS*. Since *SMAL4V* is created for processing video sequences, the qualitative results are provided in *gif*-format at *https://github.com/iegorval/smal4v*. Additional results on the *BADJA* dataset are provided without further fine-tuning of the parent network.

Chapter 7

Conclusions

7.1 Results

This thesis addressed the problem of 3D animal reconstruction which, while able to bring significant impact in industry, is still far from being solved. Results, achieved as part of this work, are listed below.

- **State of the Art overview**

A detailed overview of the state-of-the-art animal reconstruction methods was written. It lets the readers to better understand the complexity of the task and motivation behind the creation of the proposed *SMAL4V* method. These are further highlighted by the preliminary experiments performed with the off-the-shelf photogrammetry pipelines and RGB-D cameras.

- ***AnimalKey* dataset**

A new dataset of video sequences with sparsely labeled 2D keypoints and silhouettes was created and released to the research community. While still insufficient for the multi-sequence training, it allows the fine-tuning of other methods to a new animal species, while also allowing to perform online learning with the temporal supervision.

- **Refactored *SMALST* code base**

Most of the *SMALST*'s code base is rewritten to be compatible with Python 3.6+ as well as with modern versions of Python packages and CUDA drivers. This refactored code is provided as a subset of publicly released *SMAL4V* code base.

- **Results comparison on single-frame data**

A subset of reconstruction methods, based on the parametric *SMAL* model, was selected as the most promising research direction, motivated by the successes of similar models for human reconstruction. From them, two suitable pipelines, *SMALST* and *WLDO*, are evaluated on *TigDog* and *AnimalKey* datasets. Results from these experiments demonstrated the need for the creation of *AnimalKey* dataset: the *SMALST* pipeline, trained on the synthetic zebra data, had serious generalization issues, and *WLDO* pipeline, while better-performing, still required fine-tuning to provide plausible reconstructions.

- **Online learning on multiple-frame data**

A new method, called *SMAL for Videos* (*SMAL4V*), was proposed. It provided the trade-off between the *SMALR*, which requires dense 2D annotations at test time, from one side and *SMALST* and *WLDO* pipelines, which required only bounding boxes at test time, from the other side. Due to the scarcity of labeled training data, the proposed method was demonstrated only in the online learning setting, similar to *OSVOS*. Still, the architecture was designed to be also able to learn on multiple-sequence data, once such a dataset is available or once proposed *AnimalKey* dataset reaches sufficient size.

7.2 Future Work

Due to the importance and complexity of the problem, a lot of possible directions are available for this work. Those considered to be most promising are briefly described below.

- **Expansion of *AnimalKey***

Though several hundreds of hours have been invested to the creation of the current version of the *AnimalKey* dataset, its size is still not sufficient for multi-sequence training. If the future work is financed, the dataset can be significantly expanded by independent annotators. This can benefit not only the future versions of *SMAL4V* approaches, but also further research in other fields, such as object tracking or behavior analysis.

- **Multi-sequence training of *SMAL4V***

As already mentioned several times, the architecture of the *SMAL4V*'s test network was created to allow its future usage for the multi-sequence end-to-end training. Once such data is available, there is a plan to publish the results of such multi-sequence training and to try more complicated architectures.

- **Collection of 3D/mocap data**

While state-of-the-art human reconstruction networks heavily rely on the large-scale 3D and mocap datasets, almost no datasets like this exist for animals. *SMAL* and *SMBLD* models use toy scans and artist-defined models, which are by far not enough to create more detailed parametric models for different animal species. Several mocap sequences for horses, moving on a treadmill, is used in [22] but not released to public. Also, *RGBD-dog* dataset of motion capture dog sequences is available for the 3D pose estimation [121]. To the best of the author's knowledge, no other datasets, suitable for training the reconstruction pipeline with 3D supervision, are currently available.

Thus, the amount of data for all animal species combined is magnitudes away from the sizes of datasets used to train the human reconstruction pipelines. At the same time, the benefits of incorporating the 3D scans and motion capture information to the reconstruction pipelines have been long known. While it is not possible to collect such data for wild animal, it would be doable, though difficult, to do so for some domesticated animals. This direction for future work would, most likely, provide the biggest improvement to the modern animal reconstruction methods. Still, it is also the most difficult and the most expensive one to pursue.

Bibliography

1. TANSKANEN, P.; KOLEV, K.; MEIER, L.; CAMPOSECO, F.; SAURER, O.; POLLEFEYS, M. Live Metric 3D Reconstruction on Mobile Phones. In: *2013 IEEE International Conference on Computer Vision*. 2013, pp. 65–72. Available from DOI: 10.1109/ICCV.2013.15.
2. GARCIA-GRASA, Oscar; BERNAL, Ernesto; CASADO, Santiago; GIL, Ismael; MONTIEL, J. Visual SLAM for Handheld Monocular Endoscope. *IEEE transactions on medical imaging*. 2013, roč. 33. Available from DOI: 10.1109/TMI.2013.2282997.
3. DONG, Jing; BURNHAM, John; BOOTS, Byron; RAINS, Glen; DELLAERT, Frank. 4D crop monitoring: Spatio-temporal reconstruction for agriculture. In: 2017, pp. 3878–3885. Available from DOI: 10.1109/ICRA.2017.7989447.
4. RUCHAY, A; DOROFEEV, K; KALSCHIKOV, V; KOLPAKOV, V; DZHULAMANOV, K. Accurate 3D shape recovery of live cattle with three depth cameras. *IOP Conference Series: Earth and Environmental Science*. 2019, roč. 341, p. 012147. Available from DOI: 10.1088/1755-1315/341/1/012147.
5. WILLIS, Mark; KOENIG, Charles; BLACK, Stephen; CASTANEDA, Amanda. Archeological 3D Mapping: The Structure from Motion Revolution. *Journal of Texas Archeology and History*. 2016, roč. 3, pp. 1–36. Available from DOI: 10.21112/ita.2016.1.110.
6. FRANCISCO, Fritz; NÜHRENBERG, Paul; JORDAN, Alex. High-resolution, non-invasive animal tracking and reconstruction of local environment in aquatic ecosystems. *Movement Ecology*. 2020, roč. 8. Available from DOI: 10.1186/s40462-020-00214-w.
7. MASCARICH, F.; KHATTAK, S.; PAPACHRISTOS, C.; ALEXIS, K. A multi-modal mapping unit for autonomous exploration and mapping of underground tunnels. In: *2018 IEEE Aerospace Conference*. 2018, pp. 1–7. Available from DOI: 10.1109/AERO.2018.8396595.
8. USENKO, V.; ENGEL, J.; STÜCKLER, J.; CREMERS, D. Reconstructing Street-Scenes in Real-Time from a Driving Car. In: *2015 International Conference on 3D Vision*. 2015, pp. 607–614. Available from DOI: 10.1109/3DV.2015.75.
9. RHODIN, Helge; ROBERTINI, Nadia; CASAS, Dan; RICHARDT, Christian; SEIDEL, Hans-Peter; THEOBALT, Christian. General Automatic Human Shape and Motion Capture Using Volumetric Contour Cues. In: 2016, sv. 9909, pp. 509–526. ISBN 978-3-319-46453-4. Available from DOI: 10.1007/978-3-319-46454-1_31.

10. MOULON, Pierre; MONASSE, Pascal; PERROT, Romuald; MARLET, Renaud. Openmvg: Open multiple view geometry. In: *International Workshop on Reproducible Research in Pattern Recognition*. 2016, pp. 60–74.

11. SCHÖNBERGER, Johannes Lutz; FRAHM, Jan-Michael. Structure-from-Motion Revisited. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

12. SCHÖNBERGER, Johannes Lutz; ZHENG, Enliang; POLLEFEYS, Marc; FRAHM, Jan-Michael. Pixelwise View Selection for Unstructured Multi-View Stereo. In: *European Conference on Computer Vision (ECCV)*. 2016.

13. ENGEL, Jakob; SCHÖPS, Thomas; CREMERS, Daniel. LSD-SLAM: Large-Scale Direct Monocular SLAM. In: FLEET, David; PAJDLA, Tomas; SCHIELE, Bernt; TUYTELAARS, Tinne (ed.). *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014, pp. 834–849. ISBN 978-3-319-10605-2.

14. MUR-ARTAL, R.; MONTIEL, J. M. M.; TARDÓS, J. D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*. 2015, roč. 31, č. 5, pp. 1147–1163. Available from DOI: 10.1109/TRO.2015.2463671.

15. SWEENEY, Chris. *Theia Multiview Geometry Library: Tutorial & Reference* [http://theia-sfm.org]. [N.d.].

16. MOULON, Pierre; MONASSE, Pascal; MARLET, Renaud. Adaptive Structure from Motion with a Contrario Model Estimation. In: *Proceedings of the Asian Computer Vision Conference (ACCV 2012)*. Springer Berlin Heidelberg, 2012, pp. 257–270. Available from DOI: 10.1007/978-3-642-37447-0_20.

17. JANCOSEK, Michal; PAJDLA, Tomas. Multi-view reconstruction preserving weakly-supported surfaces. In: *CVPR 2011*. IEEE, 2011. Available from DOI: 10.1109/cvpr.2011.5995693.

18. GABARA, G.; SAWICKI, P. Accuracy Study of Close Range 3D Object Reconstruction Based on Point Clouds. In: *2017 Baltic Geodetic Congress (BGC Geomatics)*. 2017, pp. 25–29. Available from DOI: 10.1109/BGC.Geomatics.2017.62.

19. *Agisoft Metashape* [https://www.agisoft.com/]. [N.d.]. Agisoft Metashape.

20. FAVREAU, Laurent; REVÉRET, Lionel; DEPRAZ, Christine; CANI, Marie-Paule. Animal gaits from video. In: *SCA '04*. 2004.

21. BADGER, M.; WANG, Yufu; MODH, Adarsh; PERKES, Ammon; KOLOTOUROS, Nikos; PFROMMER, Bernd; SCHMIDT, Marc F; DANIILIDIS, Kostas. 3D Bird Reconstruction: a Dataset, Model, and Shape Recovery from a Single View. *ArXiv*. 2020, roč. abs/2008.06133.

22. LI, Ci; GHORBANI, Nima; BROOMÉ, Sofia; RASHID, Maheen; BLACK, Michael J.; HERNLUND, Elin; KJELLSTRÖM, Hedvig; ZUFFI, Silvia. hSMAL: Detailed Horse Shape and Pose Reconstruction for Motion Pattern Recognition. *ArXiv*. 2021, roč. abs/2106.10102.

23. HARTLEY, Richard; ZISSERMAN, Andrew. *Multiple View Geometry in Computer Vision*. 2. vyd. USA: Cambridge University Press, 2003. ISBN 0521540518.

24. PAJDLA, T. Elements of Geometry for Computer Vision. In: 2020.

25. FISCHLER, Martin A.; BOLLES, Robert C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*. 1981, roč. 24, pp. 381–395.

26. YUAN, Chang; MEDIONI, Gérard G. 3D Reconstruction of Background and Objects Moving on Ground Plane Viewed from a Moving Camera. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. 2006, roč. 2, pp. 2261–2268.

27. SOATTO, Stefano; YEZZI, Anthony. DEFORMOTION Deforming Motion, Shape Average and the Joint Registration and Segmentation of Images. In: 2002, sv. 53. ISBN 978-3-540-43746-8. Available from DOI: 10.1007/3-540-47977-5_3.

28. MCCULLOCH, Warren S.; PITTS, Walter. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*. 1990, roč. 52, pp. 99–115.

29. FUKUSHIMA, Kunihiko. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. *Biological Cybernetics*. 1980, roč. 36, pp. 193–202.

30. LECUN, Yann; BOTTOU, Leon; BENGIO, Y.; HAFFNER, Patrick. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*. 1998, roč. 86, pp. 2278–2324. Available from DOI: 10.1109/5.726791.

31. KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (ed.). *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105. Available also from: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

32. DUMOULIN, Vincent; VISIN, Francesco. A guide to convolution arithmetic for deep learning. *ArXiv e-prints*. 2016. Available from eprint: 1603.07285.

33. HOCHREITER, Sepp. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*. 1998, roč. 6, pp. 107–116. Available from DOI: 10.1142/S0218488598000094.

34. HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long short-term memory. *Neural computation*. 1997, roč. 9, č. 8, pp. 1735–1780.

35. CHO, Kyunghyun; MERRIENBOER, Bart; GULCEHRE, Caglar; BOUGARES, Fethi; SCHWENK, Holger; BENGIO, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. 2014. Available from DOI: 10.3115/v1/D14-1179.

36. VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan; KAISER, Lukasz; POLOSUKHIN, Illia. Attention Is All You Need. 2017.

37. KATO, Hiroharu; USHIKU, Yoshitaka; HARADA, Tatsuya. Neural 3D Mesh Renderer. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

38. CAELLES, S.; MANINIS, K.K.; PONT-TUSET, J.; LEAL-TAIXÉ, L.; CREMERS, D.; VAN GOOL, L. One-Shot Video Object Segmentation. In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

39. OZYESIL, Onur; VORONINSKI, Vladislav; BASRI, Ronen; SINGER, Amit. A Survey on Structure from Motion. *Acta Numerica*. 2017, roč. 26. Available from DOI: 10.1017/S096249291700006X.

40. TOMASI, Carlo; KANADE, T. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*. 2004, roč. 9, pp. 137–154.

41. LEE, Minsik; CHO, Jungchan; OH, Songhwai. Consensus of Non-rigid Reconstructions. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4670–4678.

42. RUSSELL, Chris; YU, Rui; AGAPITO, L. Video Pop-up: Monocular 3D Reconstruction of Dynamic Scenes. In: *ECCV*. 2014.

43. SEITZ, S.; CURLESS, Brian; DIEBEL, J.; SCHARSTEIN, D.; SZELISKI, R. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. 2006, roč. 1, pp. 519–528.

44. FAVARO, P.; SOATTO, Stefano. A geometric approach to shape from defocus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2005, roč. 27, pp. 406–417.

45. GOTARDO, P. F. U.; MARTINEZ, A. M. Kernel non-rigid structure from motion. In: *2011 International Conference on Computer Vision*. 2011, pp. 802–809. Available from DOI: 10.1109/ICCV.2011.6126319.

46. TORRESANI, L.; HERTZMANN, A.; BREGLER, C. Nonrigid Structure-from-Motion: Estimating Shape and Motion with Hierarchical Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2008, roč. 30, č. 5, pp. 878–892. Available from DOI: 10.1109/TPAMI.2007.70752.

47. SAPUTRA, Muhamad Risqi Utama; MARKHAM, Andrew; TRIGONI, Niki. Visual SLAM and Structure from Motion in Dynamic Environments: A Survey. *ACM Computing Surveys*. 2018, roč. 51, pp. 1–36. Available from DOI: 10.1145/3177853.

48. TORRESANI, Lorenzo; HERTZMANN, Aaron; BREGLER, Christoph. Learning Non-Rigid 3D Shape from 2D Motion. In: *Proceedings of the 16th International Conference on Neural Information Processing Systems*. Whistler, British Columbia, Canada: MIT Press, 2003, pp. 1555–1562. NIPS'03.

49. BREGLER, Christoph; HERTZMANN, Aaron; BIERMANN, Henning. Recovering non-rigid 3D shape from image streams. In: 2000, sv. 2, 690–696 vol.2. ISBN 0-7695-0662-3. Available from DOI: 10.1109/CVPR.2000.854941.

50. ZUFFI, S.; BLACK, Michael J. The stitched puppet: A graphical model of 3D human shape and pose. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3537–3546.

51. KANAZAWA, Angjoo; TULSIANI, Shubham; EFROS, Alexei A.; MALIK, Jitendra. Learning Category-Specific Mesh Reconstruction from Image Collections. In: *ECCV*. 2018.

52. GARG, Ravi; ROUSSOS, Anastasios (Tassos); AGAPITO, Lourdes. Dense Variational Reconstruction of Non-Rigid Surfaces from Monocular Video. In: 2013. Available from DOI: 10.1109/CVPR.2013.168.

53. AGUDO, A.; MORENO-NOGUER, F.; CALVO, B.; MONTIEL, J. M. M. Sequential Non-Rigid Structure from Motion Using Physical Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2016, roč. 38, č. 5, pp. 979–994. Available from DOI: 10.1109/TPAMI.2015.2469293.

54. KANAZAWA, A.; KOVALSKY, Shahar Z.; BASRI, R.; JACOBS, D. Learning 3D Deformation of Animals from 2D Images. *Computer Graphics Forum*. 2016, roč. 35.

55. VICENTE, S.; AGAPITO, L. Balloon Shapes: Reconstructing and Deforming Objects with Volume from Images. In: *2013 International Conference on 3D Vision - 3DV 2013*. 2013, pp. 223–230. Available from DOI: 10.1109/3DV.2013.37.

56. GORDON, A.; LI, H.; JONSCHKOWSKI, Rico; ANGELOVA, A. Depth From Videos in the Wild: Unsupervised Monocular Depth Learning From Unknown Cameras. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 8976–8985.

57. BOVZIVC, Aljavz; PALAFOX, Pablo Rodríguez; ZOLLHOFER, Michael; THIES, Justus; DAI, Angela; NIESSNER, M. Neural Deformation Graphs for Globally-consistent Non-rigid Reconstruction. *ArXiv*. 2020, roč. abs/2012.01451.

58. SALVI, Joaquim; PAGES, Jordi; BATLLE, Joan. Pattern codification strategies in structure light systems. *Pattern Recognition*. 2004, roč. 37, pp. 827–849. Available from DOI: 10.1016/j.patcog.2003.10.002.

59. GIANCOLA, Silvio; VALENTI, M.; SALA, R. A Survey on 3D Cameras: Metrological Comparison of Time-of-Flight, Structured-Light and Active Stereoscopy Technologies. In: *SpringerBriefs in Computer Science*. 2018.

60. JANG, Wonkwi; JE, C.; SEO, Y.; LEE, S. Structured-light stereo: Comparative analysis and integration of structured-light and active stereo. *Optics and Lasers in Engineering*. 2013.

61. KOLB, Andreas; BARTH, Erhardt; KOCH, Reinhard; LARSEN, Rasmus. Time-of-Flight Sensors in Computer Graphics. *Proc. Eurographics (State Art Re.)* 2008, roč. 2009.

62. CUI, Y.; SCHUON, S.; CHAN, D.; THRUN, S.; THEOBALT, C. 3D shape scanning with a time-of-flight camera. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, pp. 1173–1180. Available from DOI: 10.1109/CVPR.2010.5540082.

63. NEWCOMBE, Richard; DAVISON, Andrew; IZADI, Shahram; KOHLI, Pushmeet; HILLIGES, Otmar; SHOTTON, Jamie; MOLYNEAUX, David; HODGES, Steve; KIM, David; FITZGIBBON, Andrew. KinectFusion: Real-time dense surface mapping and tracking. In: 2011, pp. 127–136. Available from DOI: 10.1109/ISMAR.2011.6162880.

64. NEWCOMBE, Richard; FOX, Dieter; SEITZ, Steven. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In: 2015, pp. 343–352. Available from DOI: 10.1109/CVPR.2015.7298631.

65. SLAVCHEVA, Miroslava; BAUST, Maximilian; CREMERS, Daniel; ILIC, Slobodan. KillingFusion: Non-rigid 3D Reconstruction without Correspondences. In: 2017. Available from DOI: 10.1109/CVPR.2017.581.

66. SLAVCHEVA, Miroslava; BAUST, Maximilian; ILIC, Slobodan. SobolevFusion: 3D Reconstruction of Scenes Undergoing Free Non-rigid Motion. In: 2018. Available from DOI: 10.1109/CVPR.2018.00280.

67. BOŽIČ, Aljaž; PALAFOX, Pablo; ZOLLHÖFER, Michael; DAI, Angela; THIES, Justus; NIESSNER, Matthias. *Neural Non-Rigid Tracking*. 2020.

68. BOŽIČ, Aljaž; ZOLLHÖFER, Michael; THEOBALT, Christian; NIESSNER, Matthias. *DeepDeform: Learning Non-rigid RGB-D Reconstruction with Semi-supervised Data*. 2019.

69. TOMASI, Carlo; KANADE, Takeo. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*. 1992, roč. 9, pp. 137–54. Available from DOI: 10.1007/BF00129684.

70. BIANCO, Simone; CIOCCA, Gianluigi; MARELLI, Davide. Evaluating the Performance of Structure from Motion Pipelines. *Journal of Imaging*. 2018, roč. 4, p. 98. Available from DOI: 10.3390/jimaging4080098.

71. AKHTER, I.; SHEIKH, Y.; KHAN, S.; KANADE, T. Trajectory Space: A Dual Representation for Nonrigid Structure from Motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2011, roč. 33, č. 7, pp. 1442–1456. Available from DOI: 10.1109/TPAMI.2010.201.

72. FRAGKIADAKI, Katerina; SALAS, Marta; ARBELAEZ, Pablo; MALIK, Jitendra. Grouping-based low-rank trajectory completion and 3D reconstruction. *Advances in Neural Information Processing Systems*. 2014, roč. 1, pp. 55–63.

73. DAI, Y.; LI, H.; HE, M. A simple prior-free method for non-rigid structure-from-motion factorization. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 2018–2025. Available from DOI: 10.1109/CVPR.2012.6247905.

74. XIAO, Jing; CHAI, Jinxiang; KANADE, Takeo. A Closed-Form Solution to Non-Rigid Shape and Motion Recovery. *International Journal of Computer Vision*. 2006, roč. 67, pp. 233–246. Available from DOI: 10.1007/s11263-005-3962-9.

75. BARTOLI, Adrien; GAY-BELLILE, Vincent; CASTELLANI, Umberto; PEYRAS, Julien; OLSEN, Søren; SAYD, Patrick. Coarse-to-Fine Low-Rank Structure-from-Motion. In: 2008. Available from DOI: 10.1109/CVPR.2008.4587694.

76. NTOUSKOS, Valsamis; SANZARI, Marta; CAFARO, B.; NARDI, Federico; NATOLA, Fabrizio; PIRRI, F.; GARCIA, Manuel A. Ruiz. Component-Wise Modeling of Articulated Objects. *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 2327–2335.

77. AKHTER, Ijaz; SHEIKH, Yaser; KHAN, Sohaib. In defense of orthonormality constraints for nonrigid structure from motion. In: 2009, pp. 1534–1541.

78. PARASHAR, Shaifali; PIZARRO-PEREZ, Daniel; BARTOLI, A. Isometric Non-Rigid Shape-from-Motion with Riemannian Geometry Solved in Linear Time. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2018, roč. 40, pp. 2442–2454.

79. AGUDO, A.; MORENO-NOGUER, F. Simultaneous pose and non-rigid shape with particle dynamics. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 2179–2187. Available from DOI: 10.1109/CVPR.2015.7298830.

80. VAROL, Aydin; SALZMANN, Mathieu; TOLA, Engin; FUA, Pascal. Template-Free Monocular Reconstruction of Deformable Surfaces. In: 2009, pp. 1811–1818. Available from DOI: 10.1109/ICCV.2009.5459403.

81. CHHATKULI, Ajad; PIZARRO, Daniel; BARTOLI, Adrien. Non-Rigid Shape-from-Motion for Isometric Surfaces using Infinitesimal Planarity. *BMVC 2014 - Proceedings of the British Machine Vision Conference 2014*. 2014. Available from DOI: 10.5244/C.28.11.

82. BRUNET, Florent; HARTLEY, Richard; BARTOLI, Adrien; NAVAB, Nassir; MALGOUYRES, Rémy. Monocular Template-Based Reconstruction of Smooth and Inextensible Surfaces. In: 2010, pp. 52–66. ISBN 978-3-642-19317-0. Available from DOI: 10.1007/978-3-642-19318-7_5.

83. BARTOLI, Adrien; GERARD, Yan; CHADEBECQ, Francois; COLLINS, Toby; PIZARRO, Daniel. Shape-from-Template. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2015, roč. 37, pp. 1–1. Available from DOI: 10.1109/TPAMI.2015.2392759.

84. BARTOLI, A.; GÉRARD, Y.; CHADEBECQ, F.; COLLINS, T. On template-based reconstruction from a single view: Analytical solutions and proofs of well-posedness for developable, isometric and conformal surfaces. *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 2026–2033.

85. ANGUELOV, Dragomir; SRINIVASAN, P.; KOLLER, D.; THRUN, S.; RODGERS, J.; DAVIS, James E. SCAPE: shape completion and animation of people. *ACM Trans. Graph.* 2005, roč. 24, pp. 408–416.

86. LOPER, Matthew; MAHMOOD, Naureen; ROMERO, Javier; PONS-MOLL, Gerard; BLACK, Michael. SMPL: a skinned multi-person linear model. In: 2015, sv. 34. Available from DOI: 10.1145/2816795.2818013.

87. OSMAN, Ahmed A. A.; BOLKART, Timo; BLACK, Michael J. STAR: Sparse Trained Articulated Human Body Regressor. *ArXiv*. 2020, roč. abs/2008.08535.

88. ZUFFI, S.; KANAZAWA, A.; JACOBS, D. W.; BLACK, M. J. 3D Menagerie: Modeling the 3D Shape and Pose of Animals. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5524–5532. Available from DOI: 10.1109/CVPR.2017.586.

89. BOGO, Federica; KANAZAWA, Angjoo; LASSNER, Christoph; GEHLER, Peter; ROMERO, Javier; BLACK, Michael. Keep It SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image. In: 2016, sv. 9909, pp. 561–578. ISBN 978-3-319-46453-4. Available from DOI: 10.1007/978-3-319-46454-1_34.

90. KANAZAWA, Angjoo; BLACK, Michael J.; JACOBS, D.; MALIK, Jitendra. End-to-End Recovery of Human Shape and Pose. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7122–7131.

91. KANAZAWA, Angjoo; ZHANG, Jason Y.; FELSEN, Panna; MALIK, Jitendra. Learning 3D Human Dynamics From Video. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 5607–5616.

92. KOCABAS, Muhammed; ATHANASIOU, Nikos; BLACK, Michael J. VIBE: Video Inference for Human Body Pose and Shape Estimation. *2020 IEEE/-CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 5252–5262.

93. CHIBANE, Julian; ALLDIECK, Thiemo; PONS-MOLL, Gerard. Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion. In: 2020, pp. 6968–6979. Available from DOI: 10.1109/CVPR42600.2020.00700.

94. ZHENG, Zerong; YU, Tao; DAI, Qionghai; LIU, Yebin. Deep Implicit Templates for 3D Shape Representation. *ArXiv*. 2020, roč. abs/2011.14565.

95. ZHU, Hao; LIU, Yebin; FAN, Jing-tao; DAI, Q.; CAO, Xun. Video-Based Outdoor Human Reconstruction. *IEEE Transactions on Circuits and Systems for Video Technology*. 2017, roč. 27, pp. 760–770.

96. CASHMAN, T.; FITZGIBBON, A. What Shape Are Dolphins? Building 3D Morphable Models from 2D Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2013, roč. 35, pp. 232–244.

97. ZUFFI, Silvia; KANAZAWA, Angjoo; BLACK, Michael. Lions and Tigers and Bears: Capturing Non-rigid, 3D, Articulated Shape from Images. In: 2018, pp. 3955–3963. Available from DOI: 10.1109/CVPR.2018.00416.

98. ZUFFI, Silvia; KANAZAWA, Angjoo; BERGER-WOLF, Tanya; BLACK, Michael. Three-D Safari: Learning to Estimate Zebra Pose, Shape, and Texture From Images "In the Wild". In: 2019, pp. 5358–5367. Available from DOI: 10.1109/ICCV.2019.00546.

99. BIGGS, Benjamin; BOYNE, Oliver; CHARLES, James; FITZGIBBON, Andrew; CIPOLLA, Roberto. Who left the dogs out?: 3D animal reconstruction with expectation maximization in the loop. In: *ECCV*. 2020.

100. WANG, Yufu; KOLOTOUROS, Nikos; DANIILIDIS, Kostas; BADGER, M. Birds of a Feather: Capturing Avian Shape Models from Images. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 14734–14744.

101. BIGGS, Benjamin; RODDICK, Thomas; FITZGIBBON, Andrew; CIPOLLA, Roberto. Creatures Great and SMAL: Recovering the Shape and Motion of Animals from Video. In: 2019, pp. 3–19. ISBN 978-3-030-20872-1. Available from DOI: 10.1007/978-3-030-20873-8_1.

102. CHEN, Liang-Chieh; ZHU, Yukun; PAPANDREOU, George; SCHROFF, Florian; ADAM, Hartwig. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In: *ECCV*. 2018.

103. NEWELL, Alejandro; YANG, Kaiyu; DENG, Jia. Stacked Hourglass Networks for Human Pose Estimation. In: *ECCV*. 2016.

104. GOEL, Shubham; KANAZAWA, Angjoo; MALIK, Jitendra. Shape and Viewpoint without Keypoints. In: *ECCV*. 2020.

105. TULSIANI, Shubham; KULKARNI, Nilesh; GUPTA, Abhinav. *Implicit Mesh Reconstruction from Unannotated Image Collections*. 2020.

106. KOKKINOS, Filippos; KOKKINOS, Iasonas. *Learning monocular 3D reconstruction of articulated categories from motion*. 2021.

107. ZHAO, Shengyu; SHENG, Yilun; DONG, Yue; CHANG, Eric I-Chao; XU, Yan. MaskFlownet: Asymmetric Feature Matching With Learnable Occlusion Mask. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 6277–6286.

108. HE, Kaiming; GKIOXARI, Georgia; DOLLÁR, Piotr; GIRSHICK, Ross B. Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2980–2988.

109. ZHANG, Richard; ISOLA, Phillip; EFROS, Alexei; SHECHTMAN, Eli; WANG, Oliver. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. 2018.

110. KULKARNI, Nilesh; GUPTA, Abhinav; FOUHEY, David; TULSIANI, Shubham. Articulation-Aware Canonical Surface Mapping. In: 2020, pp. 449–458. Available from DOI: 10.1109/CVPR42600.2020.00053.

111. SORKINE, Olga; ALEXA, Marc. As-Rigid-As-Possible Surface Modeling. In: *Proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*. 2007, pp. 109–116.

112. HIRSHBERG, David; LOPER, Matthew; RACHLIN, Eric; BLACK, Michael. Coregistration: Simultaneous Alignment and Modeling of Articulated 3D Shape. In: 2012, sv. 7577, pp. 242–255. Available from DOI: 10.1007/978-3-642-33783-3_18.

113. LIN, Tsung-Yi; MAIRE, Michael; BELONGIE, Serge J.; HAYS, James; PERONA, Pietro; RAMANAN, Deva; DOLLÁR, Piotr; ZITNICK, C. Lawrence. Microsoft COCO: Common Objects in Context. In: *ECCV*. 2014.

114. KHOSLA, Aditya; JAYADEVAPRAKASH, Nityananda; YAO, Bangpeng; FEI-FEI, Li. Novel Dataset for Fine-Grained Image Categorization. In: *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*. Colorado Springs, CO, 2011.

115. DEL PERO, Luca; RICCO, Susanna; SUKTHANKAR, Rahul; FERRARI, Vittorio. Articulated motion discovery using pairs of trajectories. In: 2015. Available from DOI: 10.1109/CVPR.2015.7298827.

116. HUBEL, David H.; WIESEL, Torsten N. Effects of monocular deprivation in kittens. *Naunyn-Schmiedebergs Archiv für experimentelle Pathologie und Pharmakologie*. 1964.

117. BLAKEMORE, Colin; COOPER, Grahame F. Development of the Brain depends on the Visual Environment. *Nature*. 1970.

118. DEL PERO, Luca; RICCO, Susanna; SUKTHANKAR, Rahul; FERRARI, Vittorio. Behavior Discovery and Alignment of Articulated Object Classes from Unstructured Video. *International Journal of Computer Vision*. 2017, roč. 121. Available from DOI: 10.1007/s11263-016-0939-9.

119. PAPAZOGLOU, Anestis; FERRARI, Vittorio. Fast Object Segmentation in Unconstrained Video. In: 2013, pp. 1777–1784. Available from DOI: 10.1109/ICCV.2013.223.

120. MATHIS, Alexander; MAMIDANNA, Pranav; CURY, Kevin M.; ABE, Taiga; MURTHY, Venkatesh N.; MATHIS, Mackenzie W.; BETHGE, Matthias. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*. 2018. Available also from: `https://www.nature.com/articles/s41593-018-0209-y`.

121. KEARNEY, Sinead; LI, Wenbin; PARSONS, Martin; KIM, Kwang In; COSKER, Darren. RGBD-Dog: Predicting Canine Pose from RGBD Sensors. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.