An Efficient 3D Synthetic Model Generation Pipeline for Human Pose Data Augmentation

Kathan Vyas, Le Jiang, Shuangjun Liu, Sarah Ostadabbas*
Augmented Cognition Lab (ACLab)
Northeastern University, Boston, MA, USA
*ostadabbas@ece.neu.edu

Abstract

3D modeling of articulated bodies of humans or animals and using these models for synthetic 2D and 3D pose data generation can mitigate the small data challenges faced by many critical applications such as healthcare. In this paper, we present our efficient 3D synthetic model generation (3D-SMG) pipeline used for body pose data augmentation. 3D-SMG pipeline starts with scanning point clouds from various angles around the subject using an off-the-shelf RGBD camera. We then implement a dual objective iterative closest point (ICP) algorithm that uses both color (if available) as well as geometric information from point cloud and apply a pose graph node optimization to form one single rigid body mesh. 3D-SMG also includes a series of post processing steps to obtain a smooth mesh at the end of the pipeline. The approach allows it to be applied to any articulated object such as a human body or an animal. Our experiments also show high level of accuracy in dimensions of obtained 3D meshes, when compared to the original subject. As the final step towards developing augmented pose dataset, we perform model rigging to articulate the 3D model of the subject and generate dynamic avatars within variety of context-feasible poses[1].

1. Introduction

Identifying human pose over time provides critical information towards understanding human behavior and their physical interaction with the environment surrounding them [2, 18, 40, 50, 52]. In the past few decades, human pose estimation has witnessed groundbreaking advances in the computer vision field thanks to the powerful deep learning models. These models are trained using several thousands of labeled sample images if not more [1]. Such extensive data requirement poses a fundamental problem for "Small Data" domains, such as healthcare and military applications, in which data collection or labelling is expensive or limited due to privacy or security concerns. In this paper, we present an efficient data augmentation pipeline to address the small data problem in applications that require 2D and 3D body poses by generating 3D synthetic models of the subjects. Our synthetic data generation pipeline enables pose data augmentation especially for non-cooperative subjects, such young children and animals.

Although, in the last few years, several large-scale 2D human pose datasets have been released to the public, these benchmarks are still limited in the age-range of their involved subjects and the diversity of the represented activities. For instance, poses in the context such as sport (e.g. Leeds sports pose (LSP) [30]) typically include highly articulated activities, but are limited with respect to variety of the subjects' ages and their appearances (e.g. mostly young adults being in tight sports outfits). In turn, datasets such as FashionPose [13], Max Planck Institute for Informatics (MPII)'s human pose [1] and Armlets [20] aim to collect images of people wearing a variety of different clothing types, and include occlusions and truncation, however are dominated by adult humans in simple upright standing poses performing activities of daily living.

The diversity of the subjects and contexts becomes much more restricted in the publicly available 3D human pose datasets [47, 29]. These benchmarks are collected from professional motion capture (MoCap) devices in controlled lab settings [29]. Although large in quantity of the frames [29], they only represent limited number of characters with a few selected daily activities [37]. This becomes evident when many well-performed state-of-the-art human 3D pose estimation models over an existing benchmark show poor performance in a cross-set evaluation [34, 53, 55]. Therefore, the generalizability of these models to novel application-specific 3D human pose estimation remains questionable. There is an unmet need for production of 3D human pose data with high variability and automatic labeling in the con-

[1]The code and models are available at: https://github.com/ostadabbas/An-Efficient-3D-Synthetic-Model-Generation-Pipeline

text of interest, where our 3D synthetic model generation (3D-SMG) pipeline and the augmented 2D/3D pose dataset are going to be a worthwhile solution.

3D-SMG is a time- and cost-efficient pipeline that allows scanning and collection of 3D point clouds from any articulated subject (human or animal) using inexpensive off-the-shelf RGBD cameras, eliminating the need to use expensive 3D scanning devices. 3D-SMG pipeline then uses a novel 3D registration approach based on RANSAC and implement iterative closest point (ICP) to obtain smooth 3D meshes from the scanned point clouds. After a series of pre-processing and post-processing steps, a 3D rigid body (i.e. the avatar of the subject) will be constructed to be used for synthetic 3D data generation and pose augmentation. We apply 3D-SMG pipeline on a set of animal toys as well as a kid mannequin to show the feasibility of the proposed pipeline for high quality yet efficient 3D model generation. These 3D models then are rigged and articulated in the open-source Blender software to generate our dynamic avatars in variety of context-feasible poses. An overview of the 3D-SMG pipeline is shown in Fig. 1.

2. Related Work

The performance of the majority of the state-of-the-art 2D and 3D human pose estimation models crucially depends on the availability of the annotated training pose images with high variety in the appearance of people, their articulations, as well as the diversity of the contexts. The data limitation challenges gets magnified when the pose estimation is in 3D and/or needs to be done in a novel application or in the "wild". Research works like [50, 40, 26, 36] provide efficient pipelines to estimate 3D poses by tweaking traditional networks like MaskRCNN [25] and ResNet [33] or implementing innovative solutions like part affinity fields [7], few-shot learning [16], and transfer learning [14] to mitigate scarcity of labeled data.

An alternative approach to address the small data problem is data augmentation through synthetic data generation. A possible solution towards generation of 3D synthetic dataset is reconstructing rigid 3D bodies from a series of available 3D scans. This can be accomplished using a non-trivial algorithm called 3D registration, which is one of the fundamental tasks in 3D computer vision [36]. In this problem, inputs are two sets of 3D point clouds and the task is to optimally align these by estimating the best transformation between them. Due to its fundamental importance, it arises as a sub-task in many applications, including object recognition and tracking [51], range data fusion [27], graphics [35], medical image alignment [31, 54], robotics and structural bioinformatics [22], among others.

The registration problem could also be termed as the simultaneous pose and correspondence (SPC) problem [36]. Among the most popular classes of methods for solving SPC problem are algorithms based on the expectation maximization (EM), which includes iterative closest point (ICP) [5], the soft-assign [21], and their variants [23] [56], all working in an alternative fashion. Independently, [8] published a similar iterative scheme using a specific pairing procedure based on surface normal vector. This formulation was only applicable to points on surfaces, however it provided some efficient results.

Recent research uses different variants of these algorithms to develop pipelines for 3D reconstruction. This is primarily due to advent of affordable RGBD cameras, which have enabled some significant headway for visual scene and rigid body reconstruction techniques. In [58], authors explained, compared, and critically analyzed the common underlying algorithmic concepts that enabled these recent advancements. Furthermore, they showed how algorithms are designed to best exploit the benefits of RGBD data, while suppressing their often non-trivial data distortions. Using similar techniques, researchers work on object reconstruction algorithms using 3D scanning cameras. In [6], authors estimated the 3D geometry and appearance of the human body from a monocular RGBD sequence of a freely moving user in front of the sensor. Similarly in [26], authors presented a method for learning a statistical 3D skinned multi-infant linear body model (SMIL) from incomplete, low-quality RGBD sequences of freely moving infants. In [59], authors developed an approach to capture animal pose in 3D, however they used an expensive camera scanning tool to improve the quality of their scans.

In this work, we present a novel 3D synthetic model generation (3D-SMG) pipeline that uses data recorded from a low-cost RGBD camera to reconstruction any complex rigid 3D body, which also enable us to rig and articulate them in any context-specific poses. The details of the 3D-SMG pipeline is introduced in Sec. 3, which describes the pipeline from data collection to the use of our dual objective-based general and color ICP algorithm. We also present the implementation of various post processing steps including screened Poisson surface reconstruction and Laplacian smoothing, which lead to creating a final smooth 3D mesh. We performed experiments using various 3D registration algorithms to compare with our approach and provide comparisons in addition to the ablation study in Sec. 4.

3. Introducing 3D Synthetic Model Generation (3D-SMG) Pipeline

Here, we demonstrate the 3D-SMG pipeline to obtain the 3D model using 3D scans acquired from an RGBD camera. We implement a registration algorithm that initially extracts the global geometric features from each point cloud to help with a quality registration. Fig. 1 demonstrates the entire process which is categorized in three steps. The "data collection and preparation" is described in Sec. 3.1. The "regis-

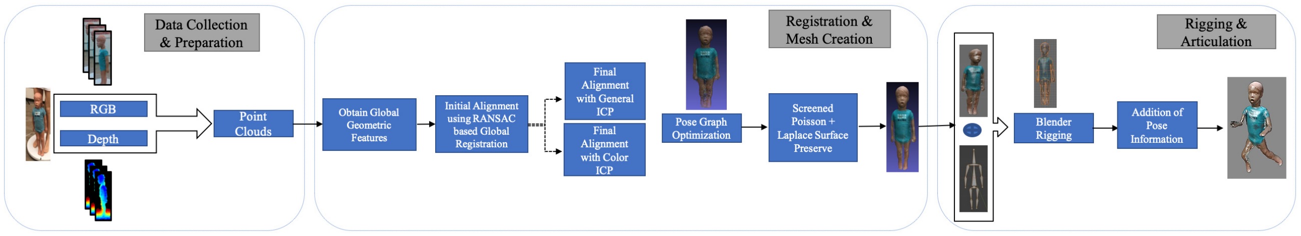
Data Collection & Preparation
RGB
Depth
Point Clouds
Obtain Global Geometric Features
Initial Alignment using RANSAC based Global Registration
Final Alignment with General ICP
Final Alignment with Color ICP
Pose Graph Optimization
Registration & Mesh Creation
Screened Poisson + Laplace Surface Preserve
Rigging & Articulation
Blender Rigging
Addition of Pose Information

Figure 1: An overview of our 3D synthetic model generation (3D-SMG) pipeline.

tration and mesh creation" is presented in Sec. 3.2. Finally, the "data rigging and articulation" is explained in Sec. 3.3. We show that for some larger size objects, where the fragments are hard to visualize, we could record RGB and depth sequences separately and synchronize them to then obtain a fragment point cloud for a given set of RGBD images.

Problem Formulation: Assuming the final 3D mesh as $M = f(G, C_K, N, H, \lambda)$, it is a function of the scanned point clouds dependent on the following parameters: G, the geometry that includes the fast point feature histogram and the surface normals obtained after voxel downsampling; C_K, the camera parameters which are intrinsic properties like focal length and field of view; N the number of point clouds captured; H a set of hyper parameters corresponding to each step, including RANSAC convergence criteria, pose graph edge pruning threshold, etc.; and λ, the screened Poisson parameter which governs the surface triangulation that converts the points to mesh. The final obtained mesh M is a smooth polished mesh that en-captures both color and geometry of the scanned subject. The final mesh M will possess a pose state descriptor θ that could be modified. The process includes infusing M with an armature (a process called rigging) in the Blender software to articulate the avatar in any context-specific pose.

3.1. RGBD Data Collection and Preparation

We use the Intel Real-Sense™ D435i depth camera, which can live-stream depth and color data at up to 90 frames per second, and all the processing to generate the depth data is done on-board by their embedded application-specific integrated circuits. The RGBD camera has a field of view of about 68×42 degrees. To communicate with the camera and extract camera parameters, we use the python API librealsense [28]. The librealsense post-processing functions have all been included into the Intel Real-Sense Viewer app which has been included as a part of Real-Sense software development toolkit (SDK). The app can be used as a quick testing ground to determine whether the Intel post-processing improvements are worth exploring [24].

The process of converting RGBD data to point cloud starts by recording a sequence of images (RGB + depth). The librealsense engine allows extraction of RGB and depth images from the recorded sequence. The obtained images are time and space synced, and therefore, they are easier to process. This is displayed in Fig. 1 within the data collection and preparation box. We next select a set of depth variables that help us manipulate depth visualization for our data. The variables we select include Z-accuracy (which evaluates the depth data accuracy), fill rate (which evaluates the percentage of the depth coverage of the image), root mean square error (RMSE) and spatial noise (which evaluate the spatial noise or spatial depth uniformity), and temporal noise (which evaluates the temporal uniformity over sequential frames). Together these depth variables help capture the depth around the edges of the object allowing to improve quality of captured point clouds. The more fine-tuned these parameters are, the better result we can obtain. The selected values for these variables are provided in Sec. **??**.

Once the depth variables selected, we proceed into data collection, by recording pairs of RGB and depth images. We then implement the RGBD registration to create individual point clouds (also referred as fragments) as described next. The algorithm used for this has been adopted from [57] which has been inspired from [12, 41]. The result of this RGBD fragment creation gives us point clouds with color information preserved.

3.2. Registration and Mesh Creation

With the execution of the previous step, we assure that the obtained point cloud contains improved depth quality due to the RGBD registration. We next move to obtaining geometric features related to each point cloud. Estimating two types of features helps us assess the geometry of the point cloud. These features are fast point feature histogram (FPFH) and surface normals. FPFH is a 33-dimensional vector that describes the local geometric property of a point, p. In [45], authors proposed the point feature histograms (PFH) as robust multi-dimensional features, which describe the local geometry around a point for 3D point cloud datasets. PFH are informative pose-invariant local features which represent the underlying surface model properties at the point p. Their computation is based on the combination of certain geometrical relations between p's nearest k neighbors. They incorporate 3D point coordinates (x, y, z) and estimate surface normals (n_x, n_y, n_z), but are extensible to the use of other properties such as curvature, 2nd order moment invariants. FPFH retains most of the discriminative power of the PFH.

Surface normals are vectors associated with each pixel, providing information related to the surface. They dictate an important information related to properties of geometric surface and play a significant role in determining the orientation of point clouds. Given a geometric surface, it is usually trivial to infer the direction of the normal at a certain point on the surface as the vector perpendicular to the surface at that point. We use estimate-normal function described in [57] , which computes normals for every point. It finds adjacent points and calculates the principal axis of the adjacent points using covariance analysis. We require to set two parameters that include a search radius and maximum number of nearest neighbours. This analysis will produce a set of two opposite directional normals, and both of these could be correct without the knowledge of the global structure. This problem is also known as normal orientation problem and in our case, the algorithm detects orientation by looking at the camera direction. It then uses k-dimensional tree (KDTree) with radius and number-of-neighbours as arguments to calculate the normals. Surface normals and FPFH are easy to calculate once we perform voxel downsampling, which allows us to reduce the number of points per point cloud. The selection of voxel downsampling value is one of the hyper-parameters ($\in H$).

Once the Geometric features are extracted, we next move to performing pairwise operations. We start by selecting two point clouds and naming them as source and target, and perform following iterative operations for the two point clouds: (1) implementing RANSAC [17] based global registration (see Sec. 3.2.1), and (2) implementing fine ICP (general ICP registration or color ICP registration) (see Sec. 3.2.2). Both ICP and Colored point cloud registrations are known as local registration methods because they rely on a rough alignment as initialization. Therefore before we implement the fine local registration, we apply coarse registration using RANSAC based global registration.

3.2.1 RANSAC and Global Registration

Global registration algorithms do not require an alignment for initialization, so they usually produce less tight alignment results which is why we use it as initialization of the local methods [19]. For each RANSAC iteration, random R_n points are picked from a source point cloud. The corresponding points in the target point cloud are detected by querying the nearest neighbour in the 33-dimensional FPFH feature space, which we discussed earlier. A pruning step is implemented, which efficiently rejects false matches using fast pruning algorithm. These correspondences between source and target point clouds are obtained with the following considerations: (1) "Check correspondence using distance": This checks if aligned point clouds are closer compared to a given threshold. These are the only parameters pre-defined by us. In our experiments, we use the distance threshold to be 1.6 times the voxel size chosen for downsampling. (2) "Check correspondence using edge length": This checks if the length of any two arbitrary edges (line formed by two vertices) individually drawn from source and target correspondences are similar.

In addition to this, another important hyper-parameter that helps us consider the correspondence is the "convergence criteria". It helps setting the max number of iterations and validation steps and it comes with a trade-off in accuracy of result and time taken for the algorithm process. The selected values are provided in Sec. 4. Based on obtained correspondences from RANSAC, the result of the algorithm is a rough transition matrix $\mathcal{R}_{Ti}$, which is a 4×4 matrix containing rotation and translation

3.2.2 Fine Local Registration

We now apply $\mathcal{R}_{Ti}$ as initial alignment to the fine ICP. We use two variants of ICP, one a general ICP and the other a color variant which takes into account the RGB values associated with each pixel. If the point clouds contain RGB information, we choose to implement color ICP variant. General ICP with point to plane operation is implemented when no color information is available for point clouds. Input to the algorithm includes: (1) Voxel-size, which is required for downsampling and other parameters; (2) Two point clouds as source and target, which are downsampled based on the voxel-size parameter; (3) The maximum-corresponding-distance-threshold, which decides the correspondence threshold for fine ICP. This is set similar to distance threshold set in global registration; (4) Input initial alignment (since fine ICP requires an input alignment) as the transition matrix $\mathcal{R}_{Ti}$; (5) Registration type, which is either point-to-plane or point-to-point.

In our experiments, we have used point-to-plane [9], which according to [44] has a faster convergence speed. The point-to-plane algorithm implements an objective function to be minimized is in the form of:

$$E_G = \sum_{\forall (p,q) \in \mathcal{K}} \left((p - \mathcal{T} \cdot q) \cdot n_p \right)^2, \tag{1}$$

where p is a point in source (P) with corresponding point q in the target point cloud (Q), for every pair of (p, q) belonging to the correspondence set $\mathcal{K}$ $(\in (P, Q))$. n_p is the normal at point p and $\mathcal{T}$ is the transformation matrix. At every iteration, we update the matrix $\mathcal{T}$ by minimizing an objective function E_G defined over the correspondence set $\mathcal{K}$. The output of the algorithm is the transformation matrix $\mathcal{T}$, which will help with the registration of point clouds (P, Q) consist of point clouds which are registered versions of source and target along with two resultant metrics:

(1) Fitness: measures the overlapping area, and (2) Inlier RMSE: measures the RMSE of all inlier correspondences.

When there is availability of color information, we instead implement color ICP variant [43]. This version of ICP uses both geometric as well as color information for registration of two point clouds. The color information locks the alignment along the tangent plane. Thus this algorithm is more accurate and more robust than general ICP, while the running speed is comparable to it. The objective function to be minimized and used in color ICP is:

$$E(T) = ((1-\delta)E_C(\mathcal{T}) + \delta E_G(\mathcal{T})), \tag{2}$$

where E_C and E_G are the photo-metric and geometric terms, respectively. $\delta \in [0,1]$ is a weight parameter determined empirically by optimizing a nonlinear objective function provided in [43]. The term E_G defines the geometric objective and takes the form similar to 1. The color specific term E_C calculates the changes between the color of source point p and target point q, and the color of q's projection on the tangent plane of point p, as:

$$E_C(\mathcal{T}) = \sum_{\forall(p,q)\in\mathcal{K}} \big(C_p(f(\mathcal{T}\cdot q)) - C(q)\big)^2, \tag{3}$$

where $Cp(\cdot)$ is an approximated function that is continuously defined on the tangent plane of point p ($\in P$) and dependent on gradient of intensity at that point. Function $f(\cdot)$ projects a 3D point to this tangent plane. $C(q)$ depicts a discrete function that retrieves the intensity of point q ($\in Q$). Similar to 1, the objective function here also provides the optimized transformation matrix $\mathcal{T}$ that helps with registration of point clouds, (P,Q). In order to improve accuracy of the algorithm, as suggested in [43], we implement a multi-scale registration scheme. The RGB point clouds are converted into 3-layers of multi-resolution point clouds using voxel downsampling. Normals for each point clouds is calculated and then individual layer registration is performed. The output of the algorithm is a registered RGB point cloud along with fitness and inlier RMSE.

3.2.3 Final Pose Graph Optimization

The registration algorithm explained in Sec. 3.2.2 uses only two point clouds for the process of registration. To register more than two point clouds, not only a global space information is required, but also understanding of the relation or orientation of each point cloud to one reference point cloud in the known global space is needed. For implementation of multiple point cloud registration, we use the concept of "Pose Graph" from [10] to determine the order and transformation of all point clouds. A node in a pose graph is a piece of geometry (point cloud in our case) P_i associated with a pose matrix $\mathcal{T}_i$, which transforms P_i into the global space. The matrix $\mathcal{T}_i$ contains unknown variables that are to be optimized (done by registration algorithm). Initialization is done with the 'global space set,' which is already configured to be the space of first piece P_0. Thus, $\mathcal{T}_0$ is selected to be an identity matrix. The other pose matrices are initialized by accumulating transformation between neighboring nodes. The neighboring nodes usually have some overlaps and are registered using algorithm provided in Sec. 3.2.1 and Sec. 3.2.2.

A pose graph edge connects two nodes that overlap. Each edge contains the transformation matrix $\mathcal{T}_{i,j}$ that aligns the source geometry P_i to the target geometry P_j. Authors in [10] observed that pairwise registration is error-prone and false pairwise alignments can outnumber correctly aligned pairs. Therefore, they provided a way to partition pose graph edges into two classes: (1) odometry edges connect temporally close, neighboring nodes, and (2) loop closure edges connect any non-neighboring nodes. For both these edges we perform the global and general ICP/color ICP registration algorithms. So for given N point clouds, we start by selecting pairs of point clouds as source and target and proceed with the registration. For each registration we obtain a transformation matrix $\mathcal{T}_i$ as well as an information matrix ψ_i. We approximate root mean square error of the corresponding points between two nodes to estimate ψ_i, by line process weight given by:

$$E^2_{RMSE} \approx \frac{1}{|\mathcal{K}_{ij}|} \sum_{(p,q)\in\mathcal{K}_{ij}} \|\omega \times q + t\|^2_2, \tag{4}$$

where $\mathcal{K}_{ij}$ is the correspondence pair for point set (p,q), ω is the rotation between two points and t is the translation.

After we obtain the transformation and information for each node and edge, we move to the global optimization of pose graph. Two types of global optimization methods could be chosen, "Gauss Newton" or "Levenberg–Marquardt". We choose the latter as it has better convergence property. Along with this, selection is done for reference node (the origin node) and edge prune threshold (threshold for pruning each edge). Once all the variables are selected, the pose-graph is optimized to provide a series of registration of given point clouds. The global optimization is performed twice on the pose graph. The first pass optimizes poses for the original pose graph taking all edges into account and does its best to distinguish false alignments among uncertain edges. These false alignments have small line process weights, and they are pruned after the first pass. The second pass runs without them and produces a tight global alignment. In our case, using examples shown by [57], all the edges are considered as true alignments, hence the second pass terminates immediately. We then finally merge all point clouds into a single point cloud by applying the transformation in pose graph for each incoming point cloud to obtain a single registered point cloud out of N input point clouds. This explanation is influenced from [49]. At the end of this process, we have one single registered point cloud that incorporates the same global geometric space as of the input point clouds.

3.2.4 Screened Poisson Process

After obtaining the registered point cloud, we perform certain operations to obtain a well-polished mesh. For the majority of point cloud post-processing, we use the tool provided by [11] called Meshlab. Meshlab helps visualise and operate on point clouds and also allows perform various post-processing algorithms. In order to insert meshing operations to our point cloud, we start with first applying a screened Poisson Process introduced by [32]. The Poisson surface reconstruction method solves a regularized optimization problem to obtain a smooth surface compared to a point cloud. For this reason, Poisson surface reconstruction can be preferable to the methods like Alpha shapes with Convex Hull [15] and ball pivoting [4], as they produce non-smooth results since the points of the point cloud are also the vertices of the resulting triangle mesh without any modifications. Such a reconstruction creates watertight surfaces from oriented point sets.

Meshlab provides an efficient way to implement the process however it comes with a disadvantage. Poisson surface reconstruction will create triangles in areas of low point density, and even extrapolates into some areas. This also acts like a limitation or disadvantage of using screened Poisson where it creates additional triangulated meshes. With the use of density values, we remove vertices and triangles that have a low support using the code from [57].

3.2.5 Laplace Smoothing

The results of a screened Poisson surface reconstruction is a well connected watertight surface from the registered point cloud. It might still contain some bumps and edges at points with changing depth. We use smoothing to improve the quality of the obtained registered mesh. Mesh smoothing, like mesh extraction, is an operation that should be performed fast enough to enable real-time adjustment of parameters. For the purpose of smoothing, we implement a Laplace smoothing with surface preservation, which allows us to smooth the mesh with having limited modification on the surface of the mesh. This method sweeps over the entire mesh several times, repeatedly moving each adjustable vertex to the arithmetic average of the vertices adjacent to it. Variations weight each adjacent vertex by the total area of the elements around it, or use the centroid of the incident elements rather than the centroid of the neighboring vertices. Laplacian smoothing is computationally inexpensive and fairly effective, but it does not guarantee improvement in element quality. In fact, Laplacian smoothing can even invert an element, unless the algorithm performs an explicit check before moving a vertex [3].

Within this process, the points on the mesh are passed through a low-pass filter, that moves each vertex in the average position of the neighbour vertices, with a condition that the new position still lies on the original surface. Laplacian smoothing is controlled by two parameters, the weighting factor ψ and the number of iterations ρ [48]. For our experiments, we implement this process in MeshLab. At the end of this process, we finally obtain a fine smooth mesh M.

3.3. Data Rigging and Articulation

Once we obtain the smooth mesh M, we next use a special approach provided by authors in [38] to infuse new pose information. Similar to that work, we use Blender, a free and open-source 3D computer graphics software tool to manipulate 3D meshes. Within this work, we use a tool named "Riggify", which provides various armatures (set of bones forming skeletons) of human and animal. These armatures could be fused in the obtained mesh M and this process is called rigging. At this point, the pose state descriptor which incorporates the pose information of the rigged mesh is denoted by θ. This concept is explained in details in [39]. We have adopted a very similar approach by adding some modification such as reducing the number of bones and adding more rigging steps for animals. Once the rigging is done, the pose state descriptor θ could be modified to generate dynamic avatars in variety of context-feasible poses.

4. Experimental Results

Here, we demonstrate the outcome of our proposed 3D-SMG pipeline by showing developed mesh of a kid mannequin and some animal toys. We show the results of rigging as well as the quality of final meshes. In our experiments, we placed our subjects (including a kid mannequin and a set of animal toys) on a rotating table with the angular velocity of $\approx$0.066 rad/s. The camera is placed on a tripod with height of $\approx$90cm (for the kid mannequin) and height of $\approx$10cm (for the animal toys). The distance, height, and angle of camera for data collection were not recorded as the algorithm is independent of these parameters.

4.1. Parameter Tuning and Ablation Study

Selection of Depth Variables. As mentioned in Sec. 3.1, we work with depth variables like Z-accuracy, fill rate, spatial noise and temporal noise which allows us to control the depth quality and point fill within the captured images. Proper tuning of these variables allows us to improve the quality of the captured depth for each frame. The calculation of the depth values depend on the intrinsic camera properties which are constant to the type of camera we use.These values change for different subjects and in our cases (kid mannequin and animal toy), the selected set of values are provided in Tab. 1. Description about these variables is provided in supplementary section.

Table 1: The selected depth variables for experiments related to the two types of subjects in our experiments.

Depth variables Subjects	Z-Accuracy (in mm)	Fill Rate (in mm)	Spatial Noise (in mm)	Temporal Noise (in mm)
Kid Mannequin	0.8	75	<=2	<=5
Animal Toys	0.8	80	<=2	<=2

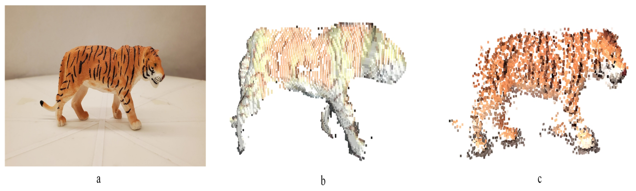

Figure 2: An improvement in quality of point clouds obtained after processing pairs of RGBD images: (a) the tiger toy's position for reference, (b) the result of point cloud captured directly from Real-Sense D435i, and (c) point cloud obtained after processing RGB and depth images and implementing RGBD registration.

Scanning RGBD vs. point clouds. As mentioned in Sec. 3.1, we collect RGB and depth images separately instead of scanning for point clouds. For experimentation, we first scanned point clouds directly and found out that the obtained results would lose a lot of color and depth information. This problem is particularly serious in small subjects (like animal toys), from which we can hardly expect a high-quality point clouds. A large number of broken point clouds would make the following registration work difficult to execute. However, using RGBD images to get the point cloud of each frame can decrease the dependency on depth data and improve the quality of pair wise registration. We performed ablation study by comparing these two data collection methods, and shows the results in Fig. 2. The results show that we can keep more details by using RGBD images.

RANSAC Variables. Once the point cloud is obtained, then we implement RANSAC based global registration. There are certain parameters that govern how the RANSAC will be implemented. The best possible choices (for our kid mannequin experiment) were one with voxel-size of 0.035 pixels, edge-length of 0.9cm, distance between point clouds as 0.0525 meters, and convergence criteria that included 4000000 iteration with validation set of 500. Using these values, the resulting parameter values included convergence set size of 255 points, execution time of 0.012 seconds and Inlier RMSE of 0.0134 points. These values can change with different subjects.

Coarse vs. Fine Registration. We next performed experiments over the two different registration algorithms: coarse and fine. Fine registration requires a rough initial estimation which is provided by the coarse registration. We provide some examples in Tab. 2, that shows the effect of registration before and after applying coarse registration indicating the improved registration quality with the two-stage process. The features we use to compare include point cloud (PC) overlap, Inlier RMSE, and time taken for a single pair of point cloud to register. The worst performance could be seen when applying only fine registration, since we used an identity matrix as its initial alignment.

Table 2: Comparison of results obtained from using coarse (global registration) and fine (ICP) registration. The given values correspond to one pair of point cloud.

Type of Registration	PC Overlap (%)	Inlier RMSE (mm)	Time Taken (ms)
Coarse Registration Only	8	0.1870	1.22
Fine Registration Only	2	0.8430	17.71
Coarse+Fine Registration	**19**	**0.0132**	**6.77**

3D-SMG vs. Other Approaches. Finally, we compare the performance of our 3D-SMG registration pipeline with other registration algorithms, including the Particle filter approach in [46], and the singular value decomposition (SVD) method in [42]. We use overlap obtained and Inlier RMSE between pairs of point clouds. Table 3 shows resulting parameters for pairwise point cloud registration. The overlap between the two point clouds is obtained by taking a ratio of overlap points (Inlier points) from two point clouds over total points in the point cloud. The overlap points depends on the correspondences obtained from respective algorithms. The more overlap obtained, the better the algorithm. Inlier RMSE is the error obtained after the transformation of the point cloud is processed.

Table 3: Comparison between 3D-SMG and two well-known point cloud alignment approaches.

Alignment Approach	PC Overlap (%)	Inlier RMSE (mm)	Time Taken (ms)
Particle Filter [46]	20	0.0098	122.71
SVD [42]	22	0.0132	1.10
Our 3D-SMG	**21**	**0.0117**	**1.32**

4.2. Scanned Results based on 3D-SMG

Our 3D-SMG pipeline demonstrates the improvements gained by the screened Poisson along with Laplace surface preserve process over registered point cloud as shown in Fig. 3. The RGB image (Fig. 3(a)) of the kid mannequin is scanned from Real-Sense camera, where individual depth and RGB frames are extracted using Real-Sense API.

The result of the registration algorithm to obtained the registered point cloud is shown in Fig. 3(b). We then use screened Poisson process (and Laplace surface preserve) to obtain a smooth mesh as shown in Fig. 3(c). The dimensions of the final mesh were measured and found out to be very close to the actual mannequin (their shoulder to shoulder length has an error of 0.2cm). Finally, we rigged the obtained mesh and infused several different 2D poses captured from a real child. The results are shown in Fig. 4.

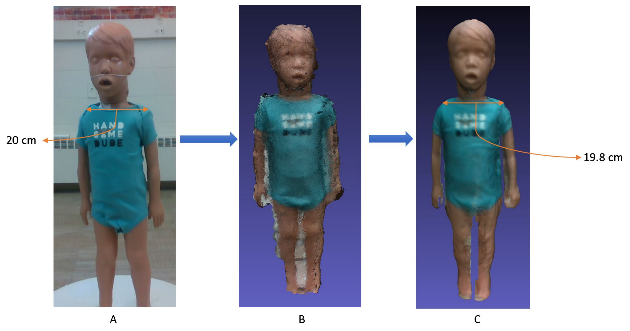

Figure 3: The final results for the 3D-SMG registration process. (a) an RGB image of the kid mannequin. (b) the registered point cloud. (c) registered mesh after implementing screened Poisson and Laplace surface preserve process.

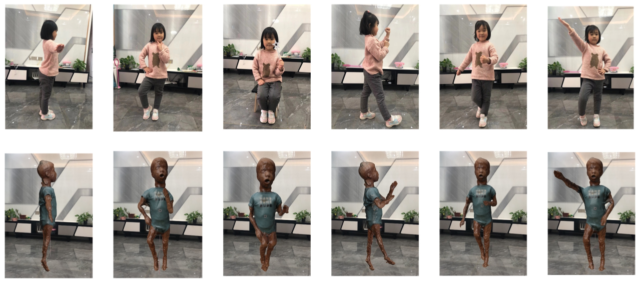

Figure 4: The rigging results of the final mesh according to the 2D poses captured from a series of real images.

In the first row, we present 6 typical real poses, and in the second row, we show the corresponding rigging results.

In addition to the kid mannequin, we also conducted experiments with various animal toys, which have much smaller size (the heights of the animal toys are all around 10cm). 7 groups of results are shown in Fig. 5. In each row, we presented the real 2D image of the animal toy and the three views of the final model. Various animals with rich patterns and extreme physical characteristics (tiger and giraffe's patterns, giraffe's leg, elk's horns) are well captured and modeled. All the results can exactly attest to the high precision of our 3D-SMG pipeline.

4.3. From 3D-SMG to Pose Augmentation

The 3D-SMG pipeline enable us to augment the pose data using the scanned mannequin or toys of children/animals, and then follow the steps presented in our previous work [38]. Within [38], our team introduced a novel synthetic human dataset called ScanAVA, and using this dataset, we were able to train a state-of-the-art 2D human pose model from scratch and achieve the pose estimation accuracy of 91.2% at PCK0.5 criteria after applying an efficient domain adaptation on the synthetic images. However in [38], we employed the Skanect, a commercially-available 3D forming software that since then has been discontinued. 3D-SMG works as an open-source tool that can be applied on the RGBD data collected from any depth-based camera. Once we obtain the 3d mesh, by randomly changing the pose and background of the obtained models, we can generate a large set of pose data.

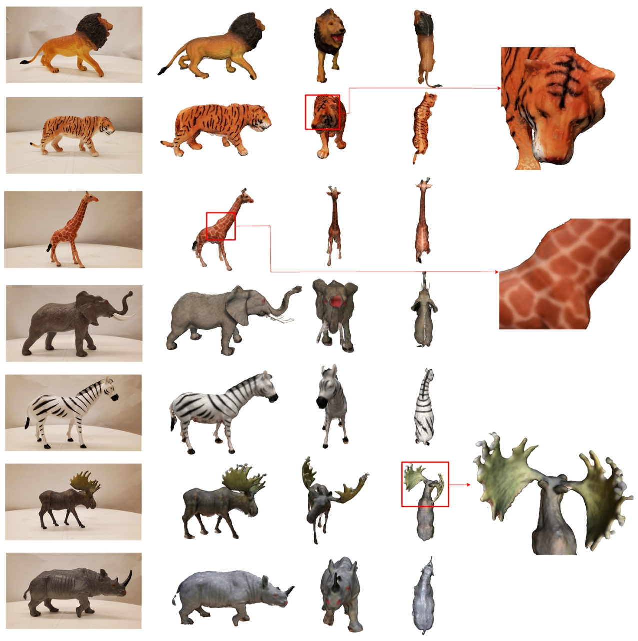

Figure 5: The 2D images of a series of animal toys and three views of the final 3D scanned model using our pipeline.

5. Conclusion

The presented 3D-SMG pipeline provides an efficient and effective solution to 3D reconstruction from 3D scans. We presented our parametrized technique to obtain depth scans from Intel Real-Sense D435i camera. Our main algorithm uses special geometric parameters, which are extracted from each point cloud and are then used to deduce the initial alignment using a deterministic RANSAC as well as a coarse registration step. The finer alignment could be obtained using our dual objective-based color ICP (if the color information is available), or a more general single objective based ICP. We finally show how we can use more than two point cloud alignments to obtain a 3D rigid reconstructed body using the implementation of pose graph optimization. Our approach can deal with significant noise around the point cloud. We also provide an approach which could be used to obtain the final smooth mesh using a screened Poisson surface reconstruction and Laplace surface smoothing. The mesh then can be used in blender to rig and obtain sets of various 2D/3D pose datasets in the contexts of interest.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 1

[2] Sudharshan Chandra Babu. A 2019 guide to human pose estimation with deep learning. https://nanonets.com/blog/human-pose-estimation-2d-guide/, September 2019. (Accessed on 03/24/2021). 1

[3] Marshall Bern and Paul Plassmann. *The Mesh Generation*. Elsevier, 2000. 6

[4] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999. 6

[5] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 2

[6] Federica Bogo, Michael J Black, Matthew Loper, and Javier Romero. Detailed full-body reconstructions of moving people from monocular rgb-d sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2300–2308, 2015. 2

[7] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017. 2

[8] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 2

[9] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 4

[10] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5556–5565, 2015. 5

[11] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. Meshlab: an open-source mesh processing tool. In Vittorio Scarano, Rosario De Chiara, and Ugo Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008. 6

[12] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. 3

[13] Matthias Dantone, Juergen Gall, Christian Leistner, and Luc Van Gool. Human pose estimation using body parts dependent joint regressors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3041–3048, 2013. 1

[14] Carl Doersch and Andrew Zisserman. Sim2real transfer learning for 3d human pose estimation: motion to the rescue. *arXiv preprint arXiv:1907.02499*, 2019. 2

[15] Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *IEEE Transactions on information theory*, 29(4):551–559, 1983. 6

[16] Siwei Feng and Marco F Duarte. Few-shot learning-based human activity recognition. *Expert Systems with Applications*, 138:112782, 2019. 2

[17] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 4

[18] Prakhar Ganesh. Human pose estimation : Simplified — by prakhar ganesh — towards data science. https://towardsdatascience.com/human-pose-estimation-simplified-6cfd88542ab3, March 2019. (Accessed on 03/24/2021). 1

[19] Natasha Gelfand, Niloy J Mitra, Leonidas J Guibas, and Helmut Pottmann. Robust global registration. In *Symposium on geometry processing*, volume 2, page 5. Vienna, Austria, 2005. 4

[20] Georgia Gkioxari, Pablo Arbeláez, Lubomir Bourdev, and Jitendra Malik. Articulated pose estimation using discriminative armlet classifiers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3342–3349, 2013. 1

[21] Steven Gold, Anand Rangarajan, Chien-Ping Lu, Suguna Pappu, and Eric Mjolsness. New algorithms for 2d and 3d point matching: pose estimation and correspondence. *Pattern recognition*, 31(8):1019–1031, 1998. 2

[22] Jacob Goldberger. Registration of multiple point sets using the em algorithm. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 730–736. IEEE, 1999. 2

[23] Sébastien Granger and Xavier Pennec. Multi-scale em-icp: A fast and robust approach for surface registration. In *European Conference on Computer Vision*, pages 418–432. Springer, 2002. 2

[24] Anders Grunnet-Jepsen and Dave Tong. Depth post-processing for intel® realsense™ d400 depth cameras. *New Technologies Group, Intel Corporation*, 2018. 3

[25] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[26] Nikolas Hesse, Sergi Pujades, Michael Black, Michael Arens, Ulrich Hofmann, and Sebastian Schroeder. Learning and tracking the 3d body shape of freely moving infants from rgb-d sequences. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 2

[27] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *Josa a*, 4(4):629–642, 1987. 2

[28] IntelRealSense. Intelrealsense/librealsense. 3

[29] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013. 1

[30] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *Proceedings of the British Machine Vision Conference*, 2010. doi:10.5244/C.24.12. 1

[31] Frederic Jurie. Solution of the simultaneous pose and correspondence problem using gaussian error model. *Computer Vision and Image Understanding*, 73(3):357–373, 1999. 2

[32] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 6

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 2

[34] Chen Li and Gim Hee Lee. Generating multiple hypotheses for 3d human pose estimation with mixture density network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9887–9895, 2019. 1

[35] Chenglong Li, Hongwei Ma, and Xinyan Gao. Model-based 3d registration optimization based on graphics simulation method. *Procedia computer science*, 131:344–353, 2018. 2

[36] Hongdong Li and Richard Hartley. The 3d-3d registration problem revisited. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. IEEE, 2007. 2

[37] Shichao Li, Lei Ke, Kevin Pratama, Yu-Wing Tai, Chi-Keung Tang, and Kwang-Ting Cheng. Cascaded deep monocular 3d human pose estimation with evolutionary training data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6173–6183, 2020. 1

[38] Shuangjun Liu and Sarah Ostadabbas. A semi-supervised data augmentation approach using 3d graphical engines. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 6, 8

[39] Shuangjun Liu and Sarah Ostadabbas. A semi-supervised data augmentation approach using 3d graphical engines. *European Conference on Computer Vision*, pages 395–408, 2018. 6

[40] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2640–2649, 2017. 1, 2

[41] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. IEEE, 2011. 3

[42] Shinji Oomori, Takeshi Nishida, and Shuichi Kurogi. Point cloud matching using singular value decomposition. *Artificial Life and Robotics*, 21(2):149–154, 2016. 7

[43] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 143–152, 2017. 5

[44] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE, 2001. 4

[45] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009. 3

[46] Romeil Sandhu, Samuel Dambreville, and Allen Tannenbaum. Point set registration via particle filtering and stochastic dynamics. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1459–1473, 2009. 7

[47] Leonid Sigal, Alexandru O Balan, and Michael J Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International journal of computer vision*, 87(1-2):4, 2010. 1

[48] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and H-P Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 175–184, 2004. 6

[49] Roberto Toldo, Alberto Beinat, and Fabio Crosilla. Global registration of multiple point clouds embedding the generalized procrustes analysis into an icp framework. In *3DPVT 2010 Conference*, volume 2, 2010. 5

[50] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014. 1, 2

[51] Paul Viola and William M Wells III. Alignment by maximization of mutual information. *International journal of computer vision*, 24(2):137–154, 1997. 2

[52] Kathan Vyas, Rui Ma, Behnaz Rezaei, Shuangjun Liu, Michael Neubauer, Thomas Ploetz, Ronald Oberleitner, and Sarah Ostadabbas. Recognition of atypical behavior in autism diagnosis from video using pose estimation over time. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2019. 1

[53] Bastian Wandt and Bodo Rosenhahn. Repnet: Weakly supervised training of an adversarial reprojection network for 3d human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7782–7791, 2019. 1

[54] Lior Wolf, Amnon Shashua, and Yoni Wexler. Join tensors: On 3d-to-3d alignment of dynamic sets. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 1, pages 388–391. IEEE, 2000. 2

[55] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li, and Xiaogang Wang. 3d human pose estimation in the wild by adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5255–5264, 2018. 1

[56] Wenyi Zhao, David Nister, and Steve Hsu. Alignment of continuous video onto 3d point clouds. *IEEE transactions on pattern analysis and machine intelligence*, 27(8):1305–1318, 2005. 2

[57] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 3, 4, 5, 6

[58] Michael Zollhöfer, Patrick Stotko, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb. State of the art on 3d reconstruction with rgb-d cameras. In *Computer graphics forum*, volume 37, pages 625–652. Wiley Online Library, 2018. 2

[59] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6365–6373, 2017. 2