Mix3D: Assembly and Animation of Seamlessly Stitched Meshes for Creating Hybrid Creatures and Objects

Yimeng Liu
University of California, Santa Barbara
Santa Barbara, USA
yimengliu@cs.ucsb.edu

Misha Sra
University of California, Santa Barbara
Santa Barbara, USA
sra@cs.ucsb.edu

ABSTRACT

Hybrid creatures are a part of mythology and folklore around the world. Often composed of parts from different animals (e.g., a Centaur), they are also increasingly seen in popular culture, such as in films, video games, print media, clothing, and art. Similarly, hybrid fictional entities composed of parts from different objects are widely visible in popular culture. Thus, modeling and animating hybrid creatures and objects is highly desirable and plays an important role in 3D character design and creation. However, this is a challenging task, even for those with considerable prior experience. In this work, we propose Mix3D, an assembly-based system for helping users, especially amateur users, easily model and animate 3D hybrids. Although assembly provides a potentially simple way to create hybrids, it is challenging to extract semantically meaningful segments from existing models and produce interchangeable edges of topologically different parts for seamless assembly. Recently, deep neural network-based approaches have attempted to address parts of this challenge, such as 3D mesh segmentation and deformation. While these methods produce good results on those two tasks independently, they are not generalizable to human, animal and object models and are therefore not suitable for the task of heterogeneous component stitching as needed for creating the hybrids. Our system tackles this issue by separating the hybrid modeling problem into three automatic and holistic processes: 1) segmenting semantically meaningful components, 2) deforming them into interchangeable parts, and 3) stitching the segments seamlessly to create hybrid models. We design an user interface (UI) that enables amateur users to easily create and animate hybrid models. Technical evaluations confirm the effectiveness of our proposed assembly method, and a user study (N=12) demonstrates the usability, simplicity and efficiency of our interactive user interface.

CCS CONCEPTS

• **Human-centered computing** → **Human computer interaction (HCI)**; **Interactive systems and tools**.

KEYWORDS

3D modeling and animation, hybrid creatures and objects, neural networks

SUI '22, December 1–2, 2022, Online, CA, USA

ACM ISBN 978-1-4503-9948-7/22/12.
https://doi.org/10.1145/3565970.3567686

ACM Reference Format:
Yimeng Liu and Misha Sra. 2022. Mix3D: Assembly and Animation of Seamlessly Stitched Meshes for Creating Hybrid Creatures and Objects. In *Symposium on Spatial User Interaction (SUI '22), December 1–2, 2022, Online, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3565970.3567686

1 INTRODUCTION

There is an increasing variety of 3D content available to consumers in the form of novel experiences in virtual reality (VR) and augmented reality (AR), 3D video games, animations and films. Creating 3D content, however, is a laborious and time consuming task. Among the most prevalent types of 3D content, hybrid shapes are in high demand for games and films (e.g., the triceratops-tank hybrid Heavy Tank in Final Fantasy VII, or the the eagle-horse hybrid Hippogriff in the Harry Potter films). But hybrid models are challenging to create due to high diversity arising from a blend of components from multiple 3D models. Although various software tools, such as Blender [5], Maya [4] and 3DS Max [3], can be used for modeling complex 3D hybrid shapes, they require users to have formal artistic training and technical expertise in 3D modeling. Surface and solid handling tools, sculpting, manipulation of control points and viewports, all make the task even more challenging, thus preventing most novice and amateur users from exploring their creative ideas.

Unlike the complex interfaces, protocols and modeling pipelines of consumer software tools, assembly-based modeling enables novice users to create 3D shapes in a more intuitive manner. Introduced by Funkhouser et al. [19], assembly-based modeling allows the creation of new 3D meshes or scenes by automatically connecting components from existing 3D models in a database using methods, such as probabilistic reasoning [11, 29], computer graphics [15], and deep learning [68, 70]. Prior work shows that assembling parts from 3D models in a mix-and-match manner offers an efficient way to create new meshes from existing content. However, even with assembly-based tools, allowing end users to create inter-category hybrids has been less explored. Most prior methods focus on intra-category assembly of animals, humanoids, or furniture [10, 15, 62, 68, 70]. Inter-category hybrid shapes can be challenging to create even for those with extensive 3D modeling experience due to the need for smoothly sculpting heterogeneous structures from adjoining parts, such as the upper body of a human with the lower body of a horse to model a Centaur. We, therefore, find that there is need and opportunity for building interactive tools that support novice users to achieve their goals of creating and animating widely popular 3D shapes, such as hybrid creatures and objects, in a fast and easy manner. Since most prior assembly-based tools designed for end users have focused on computer graphics based methods,

our secondary motivation was to explore the use of deep learning for creating a realtime interactive 3D tool for novices.

In this paper, we present Mix3D, an interactive assembly-based modeling system with a machine learning backend. Our goal is to enable novice and amateur users to create and animate 3D hybrid creatures, all in a single system with an easy to use interface. To resolve the challenging task of combining heterogeneous parts to create a seamless new model that is rigged for animation, we decompose our task into a **pre-processing** and a **runtime** stage. In the pre-processing stage, part instances are automatically segmented from a 3D mesh database with semantically meaningful adjacency correlations (e.g., a human head is adjacent to the body but not to the legs). In the runtime stage, our user interface allows novice users to select part instances to be assembled by exploring a gallery of segmented parts. Our system analyzes the user selected parts and offers guidance to users for picking part instances that allow for legal adjacency correlations. Following this interactive process, the subsequent steps involving part mesh deformation, interchangeable sewing edge computation, part mesh assembly, and assembled mesh rigging, which are all automatically accomplished by the system's backend. The output rigged hybrid model is presented to the user in-situ for animation and download. The interactive front-end works holistically with the fast and automatic machine learning backend to empower amateur users to create and animate complex hybrids easily and quickly.

To train and evaluate the system backend with multiple components necessary for cross-breed hybrid creation, we used three publicly available datasets: PASCAL VOC 2010 [12] containing animals, SURREAL [59] containing humans, and ShapeNet-Part [65] containing objects. We performed a technical evaluation to study the performance of our proposed part segmentation module designed for semantically meaningful part instance partition. Compared with PointNet++ [45], SGPN [61], and Point Transformer [69], our part segmentation algorithm matches the state-of-the-art performance. To evaluate our user interface, we conducted a study with 12 participants. Our results show that our proposed system offers a good balance between simplicity, responsiveness and usability in creating and animating hybrid creatures and objects. The main contributions of our work are as follows:

- A machine learning enabled assembly-based 3D modeling pipeline prototype that has the potential to "reduce cost and time" [26] needed to create and rig similar models using graphics approaches.
- A web-based interface that allows users, especially novice and amateur users, to quickly assemble and animate a variety of 3D hybrid creatures and objects.
- An early prototype of a VR interface that enables in-situ creation and consumption of 3D models.
- An end-to-end pipeline through a part-selection user interface to generate new combinations from heterogeneous objects.
- A novel part segmentation method that enables extraction of semantically meaningful components, computation of interchangeable sewing borders, and creation of seamlessly connected parts to produce the final hybrid models that has not been enabled as a holistic data-driven pipeline by prior work.

2 RELATED WORK

2.1 Interactive 3D Modeling for Amateur Users

Assembly-based 3D modeling enables creating new 3D meshes by connecting components from given models in a database [19]. The user can cut desired parts of 3D models with "intelligent scissoring" and combine the parts to form new shapes. This technique has been studied widely to enable amateurs to easily create 3D content [8, 17, 24, 32, 39, 40, 43, 48, 52, 62, 68]. Among the prior work, Hecker et al. [24] inspired the Spore game [51] where players use a 2D interface to select, customize, assemble and animate 3D models, including animals, humanoids and vehicles. This graphic-based approach enables customization relying on manual editing, while we use a machine learning-based method aiming to build an automatic, simple and fast 3D modeling tool for amateur/novice users. Chaudhuri et al. [10] explored real-time interactions by building a 3D modeling website, and enabled the intra-category assembly of animals and airplanes.

While assembly-based modeling has been explored in prior work, cross-breeding of 3D shapes is still a challenging task due to the heterogeneous structures and the difficulty of obtaining the interchangeable sewing edges. To this end, we propose to use a machine learning approach to support 3D model assembly allowing "cutting" body parts from one model and attaching them seamlessly to another model. Given a library of 3D models and a deep learning backend, mixing-and-matching components offers the potential to easily and extensively create new models. In contrast to many prior methods, our system allows users to not only create new hybrid models fast and easily, but also supports automatic rigging and animation of the created models. We compared Mix3D with five most relevant prior work from four aspects: 1) the supported shape categories, 2) whether cross-category hybrid creation is supported, 3) whether created hybrid models are rigged for animation, and 4) running time. Table 1 reports the comparison results and shows that Mix3D allows fast cross-category hybrid model creation and animation which none of the reported prior papers do.

Our user interface is inspired by simple and fast interaction for 3D model creation as seen in sketch-based 3D modeling tools [16, 21, 27, 36, 67]. These systems convert user drawn strokes into 3D polygonal surfaces, in essence, inflating the 2D silhouettes. Like these systems, we provide an easy-to-use interface that allows users to interactively create hybrid shapes with part selection and assembly in real-time. In contrast to sketch-based modeling systems, we take advantage of mesh details present in 3D parts partitioned from existing 3D meshes to create fine-grained outputs. These details are challenging to obtain through a 2D-stroke-to-3D-mesh inflation mechanism.

2.2 Deep Learning for 3D Hybrid Shape Creation

2.2.1 3D Mesh Segmentation. 3D mesh understanding includes instance level segmentation, i.e., semantic segmentation, and component level segmentation, i.e., part segmentation. While there is a lot of research on instance segmentation to help large-scale 3D structure understanding, we focus this section on component-level

Table 1: Mix3D compared with most closely related recent assembly-based 3D modeling systems. (The reported timings are obtained from the source papers since code for direction comparison is unavailable.)

Method	Shape Category	Cross-category Hybrid ?	Rigged for Animation ?	Assembly (+Animation) Time per 3D Model on average
Kalogerakis et al. [29]	animal, airplane, ship, chair, construction vehicle	✗	✗	30 mins
Chaudhuri et al. [11]	animal, airplane, ship, humanoid	✓	✗	20 mins
Chaudhuri et al. [10]	animal, airplane, ship	✗	✗	10 mins
Duncan et al. [15]	animal, chair, humanoid, face, insect	✓	✗	7 mins
Zhu et al. [70]	airplane, bike, chair, candelabrum, lamp, table	✗	✗	1 min
Mix3D (ours)	animal, airplane, chair, car, humanoid	✓	✓	2.5 mins

segmentation of 3D objects for hybrid model assembly. Component-level 3D segmentation deals with partition of 3D meshes into semantic components (e.g., an animal can be decomposed into its head, body, legs and tail). Among a large amount of prior 3D part segmentation work [6, 13, 22, 23, 25, 30, 31, 34, 38], PointNet++ [45] enables segmentation of non-rigid models, such as animals, by a hierarchical neural network that recursively applies PointNet [44] to the input point clouds to enable the learning of local and global features. Point Transformer [69] is considered one of the state-of-the-art methods based on self-attention mechanisms for a variety of segmentation tasks, including scene and object segmentation.

Most prior work focuses more on boosting the algorithm performance and less so on post-segmentation applications. Our system builds on semantic part segmentation but follows it with subsequent focus on using the learned segments for shape assembly. Moreover, existing work typically treats part segmentation and cutting edge identification as independent tasks, and the latter is often not included in many segmentation pipelines. Seeking the interchangeable edge boundary, however, is an indispensable step for the hybrid model creation task and critical for assembly based modeling. We address the research gaps by integrating part segmentation and cutting boundary identification in a holistic pipeline.

2.2.2 3D Mesh Deformation. 3D mesh deformation is one promising technique to enable animating 3D objects and synthesizing shape variations. Cage-based deformation is one of the most popular 3D mesh deformation techniques that has been studied in the past 30 years. The original idea was introduced by Sederberg and Parry [47] on cage-based *Free Form Deformation*. Based on the original concept, subsequent work has greatly improved correlations between the cage and the enclosed 3D model using the Mean Value Coordinate (MVC) [18, 28], which was originally introduced by Möbius in 1827 as summarized in a survey [42]. In recent studies, MVC has been widely used in real-time 3D deformation applications, especially when supported by GPU computations [14, 46]. Among them, Wang et al. [66] proposed a novel differentiable cage-based deformation module that enables detail-preserving 3D deformations of humanoids and objects.

Motivated by the existing work, our approach utilizes cage-based 3D deformation to automatically re-scale components to be sewed driven by coarse cage deformation. It forms an important module of our pipeline since it greatly reduces the cost of computing 3D mesh connections as required by some existing surface stitching methods [15]. Unlike the prior cage-based 3D deformation work, we also focus on computing interchangeable sewing edges following the deformation module (Section 4.4). Our approach presents a solution to heterogeneous component stitching, namely assembly complexity [37], by proposing a novel deformed component stitching algorithm to enable seamless part instance connections.

3 USER INTERFACE

We built a web interface to allow users to interactively create and animate 3D hybrid shapes. Upon launching the web page, the user is presented with a part-selection panel on the left and a 3D modeling area on the right. Figure 1 shows all the components of the interface. The frontend of our web interface was implemented in HTML/CSS and JavaScript. The web interface functionalities include options for the user to select body parts, display and assemble them, manipulate the assembled model and animate it. The backend of our web interface is connected with the machine learning models via a TCP socket. Specifically, the user-selected parts are fed as input to the model for inference, i.e. the neural network's input is multiple 3D components chosen by the user to generate a hybrid model of their choice. The assembled and animated hybrid model is sent back to the interface for display, interaction, animation editing and export for use in any other system that uses 3D models, such as AR and VR. For the detailed descriptions of how to use the web interface, please refer to our supplementary materials.

Our web interface demonstrates an example of a realtime human-AI interaction system. Such interactive systems have been shown to be non-trivial to design due to multiple reasons, such as model complexity, realtime inference and computation needs, and the somewhat unpredictable nature of AI generated outputs [64]. By constraining the output with user input and enabling user's manipulation of selected part components, our system attempts to overcome some of the aforementioned limitations.

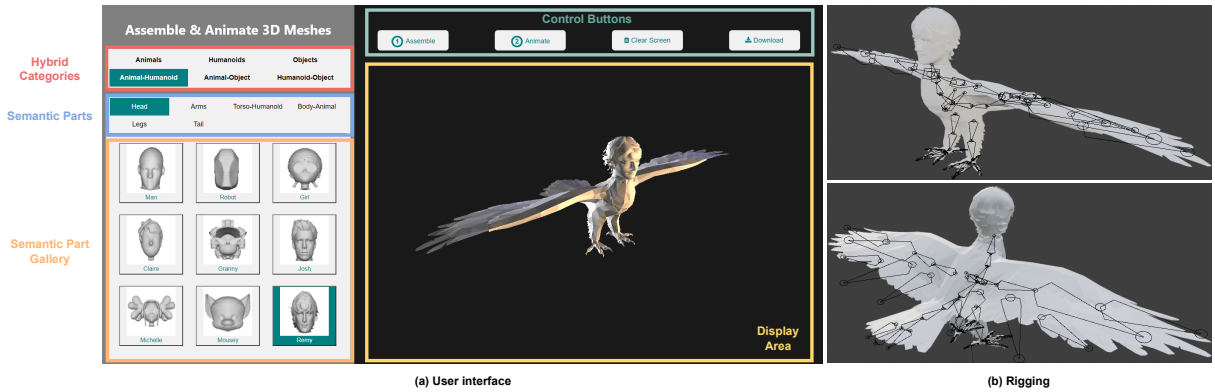

Figure 1: (a) User interface showing the different types of hybrid shapes that can be created and animated on the left top. The creation process involves selecting parts from different 3D models on the left and dropping them into the assembly area on the right. Following assembly, the user can use the buttons at the top to rig, animate, and download their hybrid model. (b) Rigged bird-human hybrid after being created by a user in (a). Some of the humanoid models are from Adobe Mixamo.

In addition to the web interface, we also built a VR interface for creating and animating models immersively. We show an example VR scene (Figure 2a) that includes models built using the VR interface (Figure 2b). We developed the VR interface in Unity 3D for the Oculus Quest VR device. Users interact with the 3D models and the interface using the Quest hand controllers. Similar to the web interface, users are presented with a 2D part-selection panel. In the VR environment, however, the user is free to move around the scene by walking or teleporting. This allows them the ability to view their generated 3D model at different scales from any perspective, and perform canonical manipulation operations on the created models, e.g., translate and rotate, using the hand controllers. The VR interface shortens the pipeline by allowing user's to create and use the generated animated models in the current scene vs having to export from the web interface and import into a VR scene.

We conducted extensive evaluations on the web interface (Section 6) due to the following considerations: 1) Since our target audience are novice and amateur users with limited or no experience in 3D modeling, a web interface has been demonstrated to be easier to quickly get started as shown in our user feedback. In contrast, a VR interface requires users to have experience in using joysticks to interact with the virtual scene and objects for 3D modeling, thus raising the interface use threshold as seen in existing VR 3D modeling systems [2, 20]. 2) A web and a VR interface are not mutually exclusive for our goal of 3D model creation, animation and consumption but instead work together. To clarify, creating and rigging 3D models with a simple web interface satisfies modeling simplicity; exporting created models to AR/VR/MR scenes enables easy use in immersive experiences for novice users. These design goals of our web interface are echoed in our participants' feedback (Section 6.4). 3) We were highly concerned about user safety due to sharing a somewhat difficult to sanitize VR device, especially as the evaluation was done during peak Omicron (BA.2) spread.

To receive initial feedback on the spatial interactions enabled by our VR interface, we conducted a pilot study with two participants and summarize early results and analyses in Section 6.4.2.

(a) (b)

Figure 2: (a) Hybrid creatures in a forest VR scene. (b) VR user interface.

4 SYSTEM PIPELINE

Figure 3 shows our system pipeline which consists of two main phases: the pre-processing phase and the runtime phase. In the pre-processing phase, three types of 3D meshes are sampled to point clouds and then fed into the part segmentation network to produce semantically partitioned components with labels and interchangeable edges (Section 4.1). These components are further clustered into groups for analyzing the adjacency relationships (Section 4.2). During the runtime, the user chooses part instances to construct hybrid meshes. The selected part labels are sent to the backend system. If a user tries to assemble parts that violate the adjacency rules, an error will pop up when the user clicks the **Assemble** button and guide them to select missing components. For example, a human head is not adjacent to their legs, the two meshes hereby cannot be connected unless the missing body component has been selected. To sew together the segments from multiple shapes into a single mesh and to reduce the cost of surface stitching, we propose to deform the source component to match the target with cage-based deformation (Section 4.3). Following it we compute the interchangeable semantic borders to build a seamlessly 3D model (Section 4.4). Lastly, the created hybrid model is rigged and presented to the user to view, animate and export.

The main strengths of our proposed pipeline are three-fold: 1) The part segmentation module automatically extracts the cutting

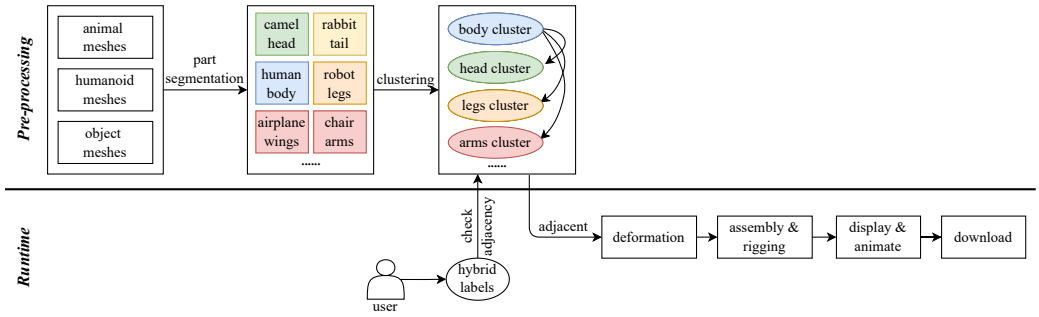

Figure 3: The system pipeline. The pre-processing stage shown above the line includes: part segmentation, part clustering and cluster adjacency analysis. The runtime elements below the line are: part selection from users, part adjacency verification, part deformation, assembly and rigging, hybrid mesh display and animation, and exporting created mesh.

boundaries. This is unsupported by many prior work and forms a crucial component for the subsequent part sewing steps. With boundary awareness, our part segmentation module achieves comparable computational complexity against recent work on part segmentation without boundary extraction (Section 5.4). 2) The part segmentation, interchangeable edge computation and component deformation modules work holistically to provide solutions to three open research problems: interchangeability of segmented parts, assembly complexity, and smoothly connected surfaces after stitching. 3) Our module-based system is flexible allowing removal of a module or using modules in a different order to achieve different shape creation goals. For example, re-scaling selected components after assembly enables making partial adjustments on the created hybrid models. Please refer to the supplementary materials for the technical details of our pipeline.

4.1 Part Segmentation of 3D Mesh

Table 2: Semantic Part Definition.

Category	Semantic Parts
Animal	head, body, leg, tail
Human	head, body, leg, arm
Object	head, body, arm

The first step in our pipeline is taking in 3D shapes and partitioning them into semantically meaningful components or parts as shown in Table 2. To extract the semantic regions of the input point cloud sampled from the corresponding mesh, we introduce a part segmentation model with feature encoding and decoding modules. Both the encoder and decoder have four stages.

In the encoding phase, the model performs downsampling on the input points. The downsampling process enables the ability to capture local context at different scales in the input points as been studied in [45]. In the decoding phase, we use a U-Net design to perform upsampling such that the input to each decoder layer is the interpolated features from the previous decoding step along with features from the symmetric encoding stage.

Following the feature learning step, we consider the pair-wise correlation of the features to help with grouping the input points for segmentation. As been studied by prior work [61], points are close together when they belong to the same group and are far away when they belong to different groups in the feature space. To model such feature distances, we take the $\mathcal{L}_2$ norm of every pair of feature vectors and store the computed Euclidean distances in an $N \times N$ matrix, namely similarity matrix. A threshold of feature distances (threshold_f) is used to determine if two points belong to the same group. The threshold value is obtained following the protocol proposed in [61]. Given the distances between pairs of input points in the feature space, we perform a binary classification such that points are grouped into either the same or different groups based on the threshold_f.

The points at the boundary of two semantic parts cannot be simply classified using feature distances. Therefore, we propose to use the supervised information provided by the classification score vector indicating the probabilities of each point belonging to the corresponding groups. Specifically, the points on semantic boundaries have similar possibilities of belonging to two groups, i.e., the difference of possibilities is $\leq$ 10% in the classification score vector, and small Euclidean distances, i.e., $\leq \text{threshold}_f$ in the similarity matrix. Based on the classification score vector and the similarity matrix, we assign all the input points into their predicted groups with corresponding group labels. We use the points classified as boundary points for border stitching to connect meshes from different semantic parts (Section 4.4).

4.2 Component Clustering and Adjacency Analysis

With the segmented meshes, we cluster the segments based on the predicted segmentation labels. Each label is used to form a cluster category, e.g., heads of an animal, a human and an airplane are components of the head cluster.

During runtime, our system receives the choice of mesh components from the user. These labels are mapped to the corresponding clusters for adjacency verification. Components from adjacent clusters across different categories need to be connectable to support hybrid creation. For example, head and body are neighboring semantic clusters, so a camel head can replace a dinosaur head to be connected with a dinosaur's body. On the other hand, replacing a dinosaur's head with a rabbit's tail is not supported because the

head and tail are not adjacent clusters. Therefore, given the clusters of semantic components, we analyze their adjacency relationships to guide component connectivity. To learn the cluster adjacency, we adapt the Bayesian network introduced in [11]. Specifically, we train the Bayesian network on our segmented meshes with predicted labels obtained from the part segmentation module. Every labeled mesh is represented by two attributes: the adjacency relations and the symmetries. The attributes (adjacency and symmetry) are represented by X initialized with random variables. We denote every segmented mesh by $P(X)$ as a joint probability distribution of X. The Bayesian network is trained to learn a probabilistic model encoding part adjacencies and symmetries.

If the user's choices lead to a valid hybrid creation, i.e., the input mesh components are adjacent, we use the symmetry attributes to check if each component has its symmetric counterpart. Components with symmetric attributes are mirrored at symmetric positions during component assembly (Section 4.4) and novel hybrids are formed by connecting neighboring semantic parts.

4.3 Cage-based Component Deformation

Building cross-category assembled models involves the challenging task of determining the scale of each mesh component to support interchangeability, as introduced in [15]. To this end, we present a 3D cage-based deformation module to automatically re-scale the source component to match the target component while preserving component details. The source and target components are differentiated by which component is to be replaced. For instance, to replace aircraft wings with human arms, we define the source component as the human arms (the component that needs to be re-scaled), while the target component is the airplane wings (the component working as reference for the scaling). The source ($\mathcal{S}_s$) and target ($\mathcal{S}_t$) components are fed into the cage-based deformation model to obtain coarse bounding meshes: source cage (C_s) and target cage (C_t). The source cage encompasses the source mesh, and is used to adjust the size of the enclosed mesh. With the cages, the deformation model predicts an offset from the source to the target cages. The offset is stored in a deformed cage ($C_{s\to t}$) to produce a deformed mesh.

The cage-based deformation module matches the scales of adjoining meshes, so some visible junctions that would otherwise be visible due to the different component sizes can be eliminated. In addition, the deformation process reduces the computational cost of deforming entire meshes when searching for sewing edges required by prior work [15], and thereby facilitates the production of seamlessly connected hybrid meshes.

4.4 Component Assembly and Rigging

Following component deformation, the next step in our pipeline involves searching for sewing borders on the adjoining meshes, and meanwhile minimizes unsatisfactory stitch closures (e.g., holes and bumps). For every segmented component, we define a semantic border as the boundary of two adjacent clusters. Since the semantic borders can be heterogeneous in shape, e.g., a dinosaur's head-body boundary differs from a camel's head-body boundary, we need to find a general edge for each semantic border to link the different borders and form a smooth surface.

To solve the problem, we describe a component assembly module containing two steps: 1) computing a general edge for each semantic border, and 2) adjusting cutting boundaries to take on the shape of the corresponding general edge for connection. Specifically, the general edge is defined as a set of vertices representing its shape, and a set of parameters representing the correspondence between the general edge and each cutting boundary on the semantic border. To compute the general edge, we propose to solve a constrained optimization problem similar to [15]). Unlike their method, we perform segmented mesh deformation (Section 4.3) before this step to match the component sizes. Therefore, only the cutting boundaries need to be deformed towards the general edge to sew adjoining meshes by a simple union operator, as opposed to deforming both the segmented meshes and the cutting boundaries by solving a complex optimization problem.

The connected hybrid meshes are rigged before being presented to the user. To rig the animal- and human-like hybrids, we use an automatic tool, RigNet [63], and tuned the tool with tested datasets. With rigged 3D models generated by our system, users can apply any type of animations for various purposes. We provide some animations in our developed web interface; other animations are available from Adobe Mixamo [1] or Unity Asset Store [55].

5 TECHNICAL EVALUATION

5.1 Data

We evaluate the part semantic segmentation method on datasets containing three categories: animals, humans and solid objects (e.g., cars and airplanes). For animals, we use the SMAL model [71] to generate meshes consisting of six categories of animals from the PASCAL VOC 2010 dataset [12]: birds, cats, dogs, horses, cows and sheep. The training and testing sets are split following the standard proposed by Wang et al. [60]. The training set consists of ~3000 samples, and the testing set consists of ~300 samples. For humans, we use the SMPL model [35] to generate meshes with various poses from the SURREAL dataset [59]. The train-test split follows the standard proposed in BodyNet [58]. The training set contains ~$2 \cdot 10^5$ samples, and the testing set contains ~500 samples. For solid objects, we use the ShapeNet-Part dataset [65]. It contains ~14000 samples for training and ~2800 samples for testing following the official train-test split introduced in ShapeNet [9].

5.2 Metric

The metric we use to evaluate the part segmentation performance is point intersection over union, averaged across all part classes (mIoU). This number is between 0 and 1, and it specifies the amount of overlap between the predicted and the ground truth point clouds (higher is better).

5.3 Baseline Methods

PointNet++ [45] is a hierarchical network with MLP backbones and it proposes to capture the local geometric structures from the neighborhood of each input point. SGPN [61] uses PointNet++ to learn a feature vector for each input point, and then captures the distances between paired features. By adopting a double-hinge loss, their model adjusts the segmentation results and assigns groups to each point by a heuristic and non-maximal suppression technique.

Point Transformer [69] is built upon a self-attention framework to extract hierarchical geometric features from point clouds. It is one of the state-of-the-art methods for point cloud part segmentation.

5.4 Results

Part Segmentation Performance. The results of part segmentation performance comparison are reported in Table 3. As can be seen, on the task of partitioning semantically meaningful parts, our method outperforms the listed baseline methods on both the PASCAL VOC 2010 and SURREAL datasets, and matches the state-of-the-art method on the ShapeNet-Part dataset.

Table 3: Part segmentation performance. mIoU (%) is mean point intersection over union. Higher is better. (PN++: PointNet++, PT: Point Transformer)

Metric ↑	Dataset	PN++	SGPN	PT	Mix3D
mIoU	PASCAL VOC 2010	39.3	40.1	40.9	**41.2**
	SURREAL	52.5	53.1	53.6	**53.8**
	ShapeNet-Part	85.1	85.8	**86.6**	86.1

Computation Speed and Boundary Awareness. Table 4 shows the comparison of running time and boundary awareness between our part segmentation method and baseline methods. To benchmark the testing time and perform a fair comparison, we ran the compared methods on a single GeForce RTX 3090 graphics card. While our method is slightly slower than PointNet++ and SGPN, it is faster than the state-of-the-art Point Transformer method. Furthermore, our approach enabled the extraction of cutting boundaries, which is not supported by the compared methods. Therefore, a custom part segmentation method was necessary for us to compute connectable sewing edges for part assembly.

Table 4: Inference time per sample (ms, averaged across datasets), and boundary awareness. (PN++: PointNet++, PT: Point Transformer)

	PN++	SGPN	PT	Mix3D
Running Time	44	37	56	50
Boundary Aware ?	✗	✗	✗	✓

5.5 Ablation Study

We conducted a controlled experiment to examine the effectiveness of the introduced similarity matrix (Section 4.1). This study is conducted on the three evaluated datasets, and the results are shown in Table 5. As can be seen, the mIoUs are improved with similarity matrix added on all the three datasets. This demonstrates that considering feature distances between input points can improve the performance of classifying semantic groups.

6 USER EVALUATION

6.1 Participants

We invited 12 participants (4F, 8M; age range: 21-50, avg: 28y, SD: 7.5) to test our system via the interactive interface. Participants came from different backgrounds including Computer Science (2), Electrical and Computer Engineering (5), Media Arts and Technology (3), and Industry (2). Eight participants had limited to no 3D modeling, rigging or animation experience, while four participants were experienced at using one or more popular 3D software and sculpting tools, such as Blender, Maya, ZBrush, Unity [54] and Solidworks [50].

Table 5: Effectiveness of similarity matrix. mIoU (%) is mean point intersection over union. Higher is better. (SM: similarity matrix)

Metric ↑	PASCAL VOC 2010	SURREAL	ShapeNet-Part
mIoU w/o SM	39.7	52.9	85.5
mIoU w/ SM	41.2	53.8	86.1

6.2 Procedure

Before starting the study, all participants provided informed consent (Protocol #7-22-0512). Following that, we provided a brief introduction of our interface and a demonstration of the features. After the introduction (around 6 minutes), each participant spent about 2 minutes to familiarize themselves with the interface. On average, the study lasted for 40 minutes, including questionnaires.

6.3 Study Design

Our main goal was to evaluate whether our interactive machine learning based tool could provide a satisfactory 3D model creation experience, knowing that it takes approximately 2.5 minutes to assemble and rig a hybrid model in our system, not including animating. We asked each participant to create six hybrid models in the supported hybrid categories. Following the model creation, participants were asked to animate the rigged models with provided animators (e.g., walking, running, idle) within 20 minutes.

Each participant completed four questionnaires during the study. The NASA Task Load Index (NASA-TLX) questionnaire consisted of 21-point scale (1: Very low/Perfect, 21: Very high/Failure). All others were rated on a 5-point Likert Scale (1: Not at all, 5: A lot).

- Pre-study questionnaire containing nine items collected biographic data, along with information on prior 3D modeling, rigging and animation experience.
- System Usability Scale (SUS) [56] is a ten-item questionnaire and it evaluated the usability of our interface.
- NASA Task Load Index (NASA-TLX) questionnaire [41] evaluated the workload of our system. Of the six items on this questionnaire, we used four (mental demand, performance, effort, frustration) as two were unrelated to the study task (physical demand and temporal demand).
- Custom post-study questionnaire with open-ended questions (fourteen questions in total) asked about the participant's experience and reflection on using our web interface.

6.4 Results

We observed that all the participants felt confident and were able to quickly get started using the interface. All participants successfully

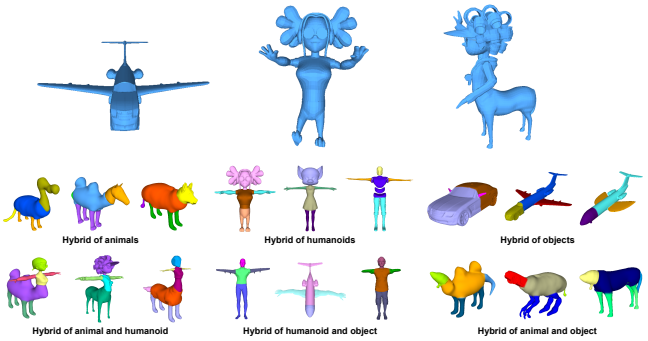

Figure 4: Top: 3D hybrid models created by the study participants using the web interface. (Left: An airplane with a car's head; Middle: A girl with mouse's arms and legs; Right: A Centaur with the upper body of a lady and lower body of a horse). Bottom: Representative examples created by the study participants in the six hybrid categories supported by our system with semantic parts shown in different colors.

built the requisite six models from different hybrid categories, and were encouraged to build more if they were interested and time permitted. Nobody had any incomplete models. The participants also animated the rigged hybrid models they created with the provided animators in the given 20 minutes. Figure 4 shows a few representative hybrids created by the participants. The models contain mesh details inherited from assembled part instances, which are often more challenging to reconstruct in sketch-based 3D modeling systems for amateur users.

Figure 5 shows the average score for each question in the SUS questionnaire. The questionnaire has ten items, and the items at odd indices are in positive tones (higher is better), while the items at even indices are in negative tones (lower is better). We compute

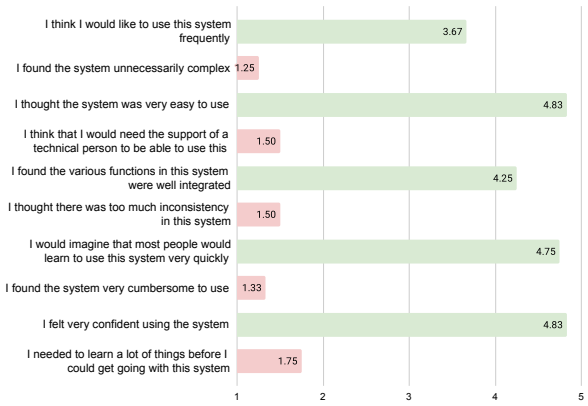

Figure 5: SUS results averaged over all participants. Green bars are odd indexed items with positive tones, and higher is better. Red bars are even indexed items with negative tones, and lower is better.

a SUS Score [53] using the following equation:

$$X = \sum_{i=1,3,5,7,9} S_i - 1, \quad Y = \sum_{j=2,4,6,8,10} 5 - S_j$$
$$\text{SUS Score} = (X + Y) \times 2.5 \tag{1}$$

where i and j are the indices of the odd and even items respectively; S_i and S_j signify the corresponding items in the SUS questionnaire. The SUS Score for our interface is 87.5 out of 100.0 on average, which indicates that our system is easy to use. Figure 6 shows the average score for the selected four questions in the NASA-TLX questionnaire. From the results, it appears that the participants felt successful at the given tasks and found it effortless to accomplish them. They rarely felt insecure or discouraged using the system.

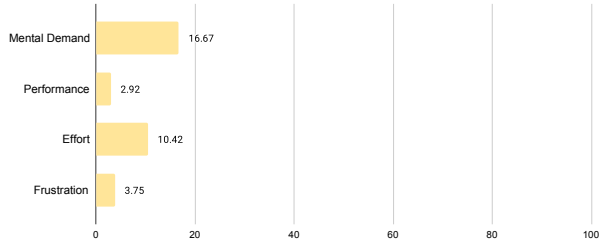

Figure 6: NASA-TLX results averaged over all participants. Lower is better.

6.4.1 *User Reflection.*

Questionnaire. We created a custom user reflection questionnaire with eight questions on a 5-point Likert scale (1: Not at all, 5: A lot) to get feedback on the creation and animation processes as well as overall system design. We report the questions and results of each question averaged over all participants in Figure 7. Overall, the results are positive with high average scores for all the questions. Q1 and Q2: Participants (10/12 rated above or equal to 4) found our system offered the potential to create various 3D models and fantasy characters through assembly, indicating the effectiveness and usefulness of our proposed machine learning-based system. Q3, Q4 and Q5: Participants (11/12 rated above or equal to 4) said our interface provided a fast and easy 3D modeling method, suggesting that despite the 2.5 minutes needed for inference, participants found the system to be interactive in realtime. Q6: Participants (6/7 rated above or equal to 4; participants with no 3D modeling experience skipped this question) found our system offered the potential to help them create 3D models much faster than their currently preferred tools, implying that our pipeline could be used as a stepping-stone for professionals to prototype complex 3D models and animations quickly before spending large amounts of time and effort to create final models using their graphics tools of choice. Q7 and Q8: Participants (10/12 rated above or equal to 4) liked that the 3D models were rigged and ready for animation and found exporting models easy to do. Since modeling and rigging are often independent and individually time-consuming steps in graphics tools, the participant feedback indicates a quick pathway directly from our system to the use of generated models in other tasks, such as gaming, animations, AR and VR.

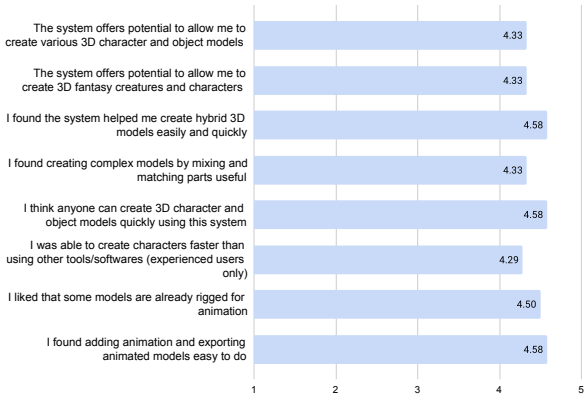

Figure 7: User reflection questionnaire results averaged over all participants. Higher is better.

Open-ended Questions. The user reflection also included six open-ended questions asking participants for feedback. Q1 and Q3 asked about the desirable aspects of our system. Our participants commented on the features they liked from different perspectives, including efficiency, simplicity and intuitiveness of our system. Overall, participants provided positive responses with P02 saying, *"If I am a modeler or animator this would be a great time-saver."* P03 found our interface to be *"straightforward to learn and use."* P10 remarked, *"It felt like I was being given a lot of customization in making a video game character."*

Q2 and Q4 asked for feedback on newer features to be included in the next version of the prototype. We received constructive suggestions from our participants that focused on enhancing customization and fine-grained adjustments to the created hybrid meshes. P06 highlighted that the current prototype supported quick modeling for amateurs and it lacked support for editing details like some industry-leading software enables saying, *"For any type of work that requires high-precision sculpting, I probably wouldn't use it as it does not provide enough control and detail editing."* P11 expressed desire for more fine-grained editing of individual body parts saying, *"If I want the legs of a man, I could not adjust the length or thickness of it. I can only use the default mesh given to me."*

Q5 and Q6 asked about who and what purposes the participants thought the prototype was best suited for. The responses mostly aligned with our motivation of building a system for amateur users to easily and quickly create 3D hybrid models, a task that is challenging and time consuming even for highly skilled modelers. P12 remarked upon how our system can serve as a first step in a larger complex process saying, *"It would be best suited for creating animated 3D models very fast and can generate motion and mesh data for other animation applications."* P06 compared it to strategies used by experienced modelers to build quick prototypes before committing time and resources to build the detailed models and animations. They said, *"Many 3D artists often use one system (e.g. Maya, 3DS Max) to assemble low poly models together to build a rough shape and then move to a sculpting software (e.g., ZBrush) to add extra details. I do see this system can work well for 3D artists at the assembling step as a good and fast preparation for fine-detail sculpting. Also it can be used by hobbyists to explore and prototype new creatures, using them for fantasy storytelling."*

Note that from user feedback, none of the participants, amateur or experienced, commented upon the time taken (reported in Table 1) for the assembled model to be presented. This is an interesting finding because it indicates the potential for using machine learning to create 3D modeling tools that can provide a near-realtime yet satisfactory experience via a simple interface.

6.4.2 VR Interface Pilot Study. We conducted a pilot study of our VR interface with two participants. They were asked to create a hybrid creature by selecting part components, and interact with the generated output using the Oculus Quest hand controllers. After creating their own hybrids, participants were asked to explore a VR scene with multiple pre-created creatures. After using the generation and interaction interfaces, we asked participants what they liked and/or disliked about the experience. P01-VR commented *"I have a better sense of the object shapes in 3D. I like the VR scenes provided and I hope to choose a scene from multiple options, and edit the scenes as I want. However, the controllers are hard to use. I can hardly click the buttons using them."* P02-VR said *"I think the interface is easy to use as it follows the creation steps. The VR scenes are immersive. It feels like I was in a magic world with all the fantasy creatures surrounding me."*

As a followup, we asked about their experience with the spatial interactions in VR, both for creating and interacting with the 3D models. P01-VR thought *"Rotating is harder using hand controllers than a mouse. I prefer to walk to the back of a 3D model."* P02-VR felt *"In VR it is great to move around freely in the scene, so that I can zoom in/out intuitively by walking in the 3D space."*

Last, we asked an open-ended question about what types of interaction mechanisms would make it easier for them to create and interact with the 3D content. P01-VR suggested *"I hope for an easier way to interact with the objects. For example, I would like to use my hands to move and edit the objects directly, or use my fingers to click the buttons."* P02-VR said *"I would like to build a studio-like interface when creating 3D models. For instance, making there a spotlight around the created model so that I can focus on it in the modeling process, otherwise many buttons on a UI will distract me."*

From this pilot study, we learned that with spatial interactions in VR, 3D modeling becomes an immersive experience as users are able to step inside the scene. This mimics some of the interactions and experience of recent consumer 3D modeling software, though our interface is vastly simpler and targeted at novice creators vs experts [2, 7, 20, 49]. Although our VR system enables "immersive authoring" [33], i.e., the creation process that occurs in an immersive environment, and in-situ authoring [57], i.e., creation happens in the same application used for consuming the content, the pilot indicates a need to explore specifics of interface design that can make the process intuitive for users who are novices, both to VR and the 3D modeling task.

Based on the initial feedback, the interaction modality can make a big difference in the user's experience. VR hand controllers seemed to present a relatively high threshold for amateur/novice users. Future work can analyze how different interactions, such as using hands, or joysticks, or even custom devices (e.g., gloves), could

enable more efficient and intuitive use. A related question is whether 3D modeling is a task suitable for relying only on spatial interaction paradigms. Prior work [2, 49] has combined the use of 2D and 3D interactions for 3D modeling, where users perform complex model editing in 2D using a mouse or pen, and export the output for use in VR or other spatial environments. This setup aligns with our design of importing created models from the web interface into AR/VR scenes. We believe further work is needed to evaluate this fundamental question of whether 3D modeling is intuitively and immersively achievable for novices in AR/VR.

7 LIMITATIONS, FUTURE WORK AND LESSONS LEARNED

Algorithm and Computation Complexity. Although we present an easy and automatic 3D modeling system, it has some limitations that result from the complex procedures of an integral pipeline and the inherent limitations of machine learning based methods. For example, the inability to create all types of models due to the unavailability of relevant datasets. However, the pipeline showcases a prototype for the design of an automatic 3D assembly and animation system with minimal manual effort. Furthermore, we report the average time taken to create plus rig a model using Mix3D is around 2.5 minutes in Table 1. While compared to photo editing apps or video games, this might seem slow, the time compares favorably with accomplishing similar goals using commonly used 3D modeling tools. For example, Adobe's cloud-based service takes 2-3 minutes just to rig humanoids (only type of model supported currently) with more time often needed to make manual corrections to the output rig. Another example is that the time taken to assemble and rig our hybrid models is considerably faster than creating a similar model in Blender or Maya by an expert 3D modeler, or similar to time taken to create an un-rigged model with photogrammetry or 3D scanning mobile apps (e.g., Capture3D or Polycam that use Lidar sensors to build the 3D models). These apps, however, only work if the physical object is available to scan into a model which is more challenging for our fictional hybrid use case. In the interest of enhancing the user experience, reducing assembly time is worth exploring by refining neural network structures in the future.

High Quality Hybrid Models. An open research problem is improving the quality of the generated hybrid 3D models. Some of the created models by our system do not have a highly fine-grained mesh resulting in a more cartoony look, as if the models were sculpted from Play Doh©. While not undesirable, this is mainly due to the lack of high quality input models in the training dataset and can be addressed in the future as higher quality 3D data becomes available, or by post-processing with sculpting tools to add details using ZBrush or SimpModeling [36]. Additionally, a related issue is when assembling two meshes with highly different levels of detail, the stitching boundary might fail to be smooth, as shown in Figure 8. This limitation arises from balancing mesh detail preservation and sewing border smoothness. To solve this problem, in our future work we plan to investigate enforcing both C0 and C1 continuity [15] between assembled parts with distinct detail richness; meanwhile, minimizing deformation of original shapes.

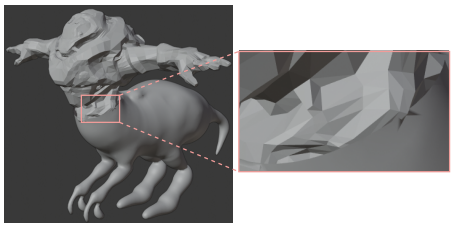

Figure 8: An example failure case of our stitching method resulting from adjoining parts having vastly different levels of mesh details.

Balance of Usability and Expressiveness. From the user reflection, experienced participants described their desire to customize the created models by adjusting individual parts of the models, or the ability to generate hybrid models not following the adjacency constraints (e.g., a creature with its head directly connected to its legs). Taking in a user's design intent is a challenging task for data-drive algorithms since they are trained on existing dataset with features or rules that are learned from that data. However, with growing research interest in human-AI collaboration, future work could study how to integrate a larger variety of user specified rules in the training phrase to tackle the limitations introduced by the dataset. On the other hand, although our system is not as highly expressive as Blender or Maya, our module-based pipeline has the potential to raise the ceiling by enabling partial scale adjustments on the created models by adding a cage-based deformation module after the creation of hybrid models, as described in Section 4. As more datasets become available and algorithms improve, we can expect the expressivity of such interactive machine learning-based tools to increase over time.

8 CONCLUSION

In this paper, we presented Mix3D, an assembly-based system for amateur users for creating 3D hybrid creatures and objects using a web interface. The proposed system enabled the creation of rigged and animated hybrid models across multiple categories of 3D shapes, including animals, human and objects. To our knowledge, ours is the first machine learning-enabled 3D model creator and animator with a web interface that provides an interactive user experience. Our technical evaluation of the underlying novel part segmentation module shows that it outperforms prior work and matches the state-of-the-art model's performance. The technical evaluation demonstrates the effectiveness of our proposed part segmentation network for generalizing to 3D shapes from multiple categories and identifying interchangeable sewing edges to enable the creation of seamlessly connected hybrid 3D models. Our user evaluation demonstrated that our system can help users easily and efficiently create and animate hybrid creatures and objects.

ACKNOWLEDGMENTS

We would like to thank the study participants for evaluating our system, and anonymous reviewers for their valuable feedback.

REFERENCES

[1] Adobe. 2022. Mixamo. Retrieved March 24, 2022 from http://www.mixamo.com/

[2] Adobe. 2022. Substance 3D Modeler. Retrieved July 8, 2022 from https://www.roadtovr.com/adobe-substance-3d-modeler-medium-vr-modeling-pro-workflows/

[3] Autodesk. 2022. 3ds Max: Create massive worlds and high-quality designs. Retrieved June 3, 2022 from https://www.autodesk.com/products/3ds-max/overview?term=1-YEAR&tab=subscription

[4] Autodesk. 2022. Maya: Create expansive worlds, complex characters, and dazzling effects. Retrieved June 3, 2022 from https://www.autodesk.com/products/maya/overview?term=1-YEAR&tab=subscription

[5] Blender. 2022. Blender. Retrieved September 5, 2022 from https://www.blender.org/

[6] Alexandre Boulch. 2020. ConvPoint: Continuous convolutions for point cloud processing. *Computers & Graphics* 88 (2020), 24–34.

[7] Medium by Adobe. 2022. Create a virtual masterpiece. Retrieved March 24, 2022 from https://www.adobe.com/products/medium.html

[8] Hao Cao, Rong Mo, and Neng Wan. 2019. 3D modelling of a frame assembly using deep learning and the Chu–Liu–Edmonds Algorithm. *Assembly Automation* (2019).

[9] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. *ShapeNet: An Information-Rich 3D Model Repository.* Technical Report arXiv:1512.03012 [cs.GR]. Stanford University — Princeton University — Toyota Technological Institute at Chicago.

[10] Siddhartha Chaudhuri, Evangelos Kalogerakis, Stephen Giguere, and Thomas Funkhouser. 2013. Attribit: content creation with semantic attributes. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 193–202.

[11] Siddhartha Chaudhuri, Evangelos Kalogerakis, Leonidas Guibas, and Vladlen Koltun. 2011. Probabilistic reasoning for assembly-based 3D modeling. In *ACM SIGGRAPH 2011 papers*. 1–10.

[12] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. 2014. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1971–1978.

[13] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 2016. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*. Springer, 424–432.

[14] Fabrizio Corda, Jean-Marc Thiery, Marco Livesu, Enrico Puppo, Tamy Boubekeur, and Riccardo Scateni. 2020. Real-Time Deformation with Coupled Cages and Skeletons. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 19–32.

[15] Noah Duncan, Lap-Fai Yu, and Sai-Kit Yeung. 2016. Interchangeable components for hands-on assembly based modelling. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 1–14.

[16] Marek Dvorožňák, Daniel Sỳkora, Cassidy Curtis, Brian Curless, Olga Sorkine-Hornung, and David Salesin. 2020. Monster mash: a single-view approach to casual 3D modeling and animation. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–12.

[17] Matthew Fisher, Daniel Ritchie, Manolis Savva, Thomas Funkhouser, and Pat Hanrahan. 2012. Example-based synthesis of 3D object arrangements. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–11.

[18] Michael S Floater, Géza Kós, and Martin Reimers. 2005. Mean value coordinates in 3D. *Computer Aided Geometric Design* 22, 7 (2005), 623–631.

[19] Thomas Funkhouser, Michael Kazhdan, Philip Shilane, Patrick Min, William Kiefer, Ayellet Tal, Szymon Rusinkiewicz, and David Dobkin. 2004. Modeling by example. *ACM transactions on graphics (TOG)* 23, 3 (2004), 652–663.

[20] Google. 2022. Google Blocks. Retrieved March 24, 2022 from https://arvr.google.com/blocks/

[21] Benoit Guillard, Edoardo Remelli, Pierre Yvernay, and Pascal Fua. 2021. Sketch2Mesh: Reconstructing and Editing 3D Shapes from Sketches. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 13023–13032.

[22] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. 2021. Pct: Point cloud transformer. *Computational Visual Media* 7, 2 (2021), 187–199.

[23] Tong He, Chunhua Shen, and Anton van den Hengel. 2021. DyCo3D: Robust instance segmentation of 3D point clouds through dynamic convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 354–363.

[24] Chris Hecker, Bernd Raabe, Ryan W Enslow, John DeWeese, Jordan Maynard, and Kees van Prooijen. 2008. Real-time motion retargeting to highly varied user-created morphologies. *ACM Transactions on Graphics (TOG)* 27, 3 (2008), 1–11.

[25] Sindhu Hegde and Shankar Gangisetty. 2021. PIG-Net: Inception based deep learning architecture for 3D point cloud segmentation. *Computers & Graphics* 95 (2021), 13–22.

[26] Aaron Hertzmann. 2003. Machine learning for computer graphics: A manifesto and tutorial. In *11th Pacific Conference onComputer Graphics and Applications, 2003. Proceedings.* IEEE, 22–36.

[27] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. 2006. Teddy: a sketching interface for 3D freeform design. In *ACM SIGGRAPH 2006 Courses*. 11–es.

[28] Tao Ju, Scott Schaefer, and Joe Warren. 2005. Mean value coordinates for closed triangular meshes. In *ACM Siggraph 2005 Papers*. 561–566.

[29] Evangelos Kalogerakis, Siddhartha Chaudhuri, Daphne Koller, and Vladlen Koltun. 2012. A probabilistic model for component-based shape synthesis. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–11.

[30] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. 2010. Learning 3D mesh segmentation and labeling. In *ACM SIGGRAPH 2010 papers*. 1–12.

[31] Seunghoi Kim and Daniel C Alexander. 2021. AGCN: Adversarial Graph Convolutional Network for 3D Point Cloud Segmentation. (2021).

[32] Hamid Laga, Michela Mortara, and Michela Spagnuolo. 2013. Geometry and context for semantic correspondences and functionality recognition in man-made 3D shapes. *ACM Transactions on Graphics (TOG)* 32, 5 (2013), 1–16.

[33] Gun A Lee, Claudia Nelles, Mark Billinghurst, and Gerard Jounghyun Kim. 2004. Immersive authoring of tangible augmented reality applications. In *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE, 172–181.

[34] Zhi-Hao Lin, Sheng-Yu Huang, and Yu-Chiang Frank Wang. 2020. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1800–1809.

[35] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34, 6 (Oct. 2015), 248:1–248:16.

[36] Zhongjin Luo, Jie Zhou, Heming Zhu, Dong Du, Xiaoguang Han, and Hongbo Fu. 2021. SimpModeling: Sketching Implicit Field to Guide Mesh Modeling for 3D Animalmorphic Head Design. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 854–863.

[37] Katia Lupinetti, Jean-Philippe Pernot, Marina Monti, and Franca Giannini. 2019. Content-based CAD assembly model retrieval: Survey and future challenges. *Computer-Aided Design* 113 (2019), 62–81.

[38] Yecheng Lyu, Xinming Huang, and Ziming Zhang. 2020. Learning to segment 3d point clouds in 2d image space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12255–12264.

[39] Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. 2011. Interactive furniture layout using interior design guidelines. *ACM transactions on graphics (TOG)* 30, 4 (2011), 1–10.

[40] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J Mitra, and Leonidas J Guibas. 2019. StructureNet: hierarchical graph networks for 3D shape generation. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–19.

[41] NASA. 2021. NASA TLX. Retrieved November 6, 2021 from https://humansystems.arc.nasa.gov/groups/tlx/

[42] Jesús R Nieto and Antonio Susín. 2013. Cage based deformations: a survey. In *Deformation models*. Springer, 75–99.

[43] Florent Poux and Roland Billen. 2019. Voxel-based 3D point cloud semantic segmentation: unsupervised geometric and relationship featuring vs deep learning methods. *ISPRS International Journal of Geo-Information* 8, 5 (2019), 213.

[44] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

[45] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems* 30 (2017).

[46] Andreas Scalas, Yuanju Zhu, Franca Giannini, Ruding Lou, Katia Lupinetti, Marina Monti, Michela Mortara, and Michela Spagnuolo. 2020. A first step towards cage-based deformation in Virtual Reality. In *Smart Tools and Applications in computer Graphics-Eurographics Italian Chapter Conference*. Eurographics, 119–130.

[47] Thomas W Sederberg and Scott R Parry. 1986. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*. 151–160.

[48] Chao-Hui Shen, Hongbo Fu, Kang Chen, and Shi-Min Hu. 2012. Structure recovery by part assembly. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–11.

[49] Gravity Sketch. 2021. Think in 3D. Create in 3D. Retrieved March 24, 2021 from https://www.gravitysketch.com/

[50] Solidworks. 2022. Solidworks. Retrieved September 25, 2022 from https://www.solidworks.com/

[51] Spore. 2022. Spore Game. Retrieved September 25, 2022 from http://www.spore.com/

[52] Minhyuk Sung, Hao Su, Vladimir G Kim, Siddhartha Chaudhuri, and Leonidas Guibas. 2017. ComplementMe: Weakly-supervised component suggestions for 3D modeling. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–12.

[53] UIUXTrend. 2022. Measuring and Interpreting System Usability Scale (SUS). Retrieved June 2, 2022 from https://uiuxtrend.com/measuring-system-usability-scale-sus/

[54] Unity. 2022. Unity. Retrieved September 25, 2022 from https://unity.com/
[55] Unity. 2022. Unity Asset Store. Retrieved March 24, 2022 from https://assetstore.unity.com/
[56] Usability.gov. 2021. System Usability Scale. Retrieved November 6, 2021 from https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html
[57] Anton van den Hengel, Rhys Hill, Ben Ward, and Anthony Dick. 2009. In situ image-based modeling. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*. IEEE, 107–110.
[58] Gul Varol, Duygu Ceylan, Bryan Russell, Jimei Yang, Ersin Yumer, Ivan Laptev, and Cordelia Schmid. 2018. Bodynet: Volumetric inference of 3d human body shapes. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 20–36.
[59] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. 2017. Learning from synthetic humans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 109–117.
[60] Peng Wang, Xiaohui Shen, Zhe Lin, Scott Cohen, Brian Price, and Alan L Yuille. 2015. Joint object and part segmentation using deep learned potentials. In *Proceedings of the IEEE International Conference on Computer Vision*. 1573–1581.
[61] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. 2018. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2569–2578.
[62] Kai Xu, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2012. Fit and diverse: Set evolution for inspiring 3d shape galleries. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–10.
[63] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. 2020. RigNet: neural rigging for articulated characters. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 58–1.
[64] Qian Yang, Aaron Steinfeld, Carolyn Rosé, and John Zimmerman. 2020. Re-examining whether, why, and how human-AI interaction is uniquely difficult to design. In *Proceedings of the 2020 chi conference on human factors in computing systems*. 1–13.
[65] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. 2016. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)* 35, 6 (2016), 1–12.
[66] Wang Yifan, Noam Aigerman, Vladimir G Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. 2020. Neural cages for detail-preserving 3d deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 75–83.
[67] Robert C Zeleznik, Kenneth P Herndon, and John F Hughes. 2006. SKETCH: An interface for sketching 3D scenes. In *ACM SIGGRAPH 2006 Courses*. 9–es.
[68] Zaiwei Zhang, Zhenpei Yang, Chongyang Ma, Linjie Luo, Alexander Huth, Etienne Vouga, and Qixing Huang. 2020. Deep generative modeling for scene synthesis via hybrid representations. *ACM Transactions on Graphics (TOG)* 39, 2 (2020), 1–21.
[69] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. 2021. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 16259–16268.
[70] Chenyang Zhu, Kai Xu, Siddhartha Chaudhuri, Renjiao Yi, and Hao Zhang. 2018. SCORES: Shape composition with recursive substructure priors. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–14.
[71] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 2017. 3D Menagerie: Modeling the 3D Shape and Pose of Animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.