Pacific Graphics 2018
H. Fu, A. Ghosh, and J. Kopf
(Guest Editors)

Volume 37 (2018), Number 7

Automatic Mechanism Modeling from a Single Image with CNNs

Minmin Lin[1] Tianjia Shao[†2] Youyi Zheng[1] Zhong Ren[1] Yanlin Weng[1] Yin Yang[3]

[1]State Key Lab of CAD&CG, Zhejiang University
[2]University of Leeds
[3]University of New Mexico

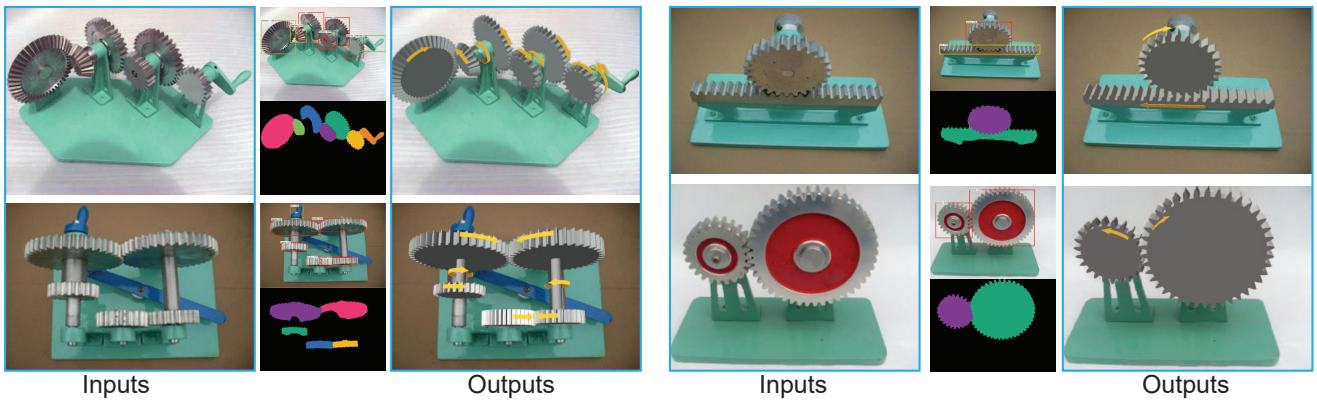

Figure 1: *We propose a fully automatic system that takes as input a single RGB image of a mechanism assembly and outputs its 3D instantiation. The resulting mechanism model is guaranteed to function in a physically valid way, and it well matches the input image. To the best of our knowledge, this is the first automatic mechanism modeling system that uses a single input image.*

Abstract

This paper presents a novel system that enables a fully automatic modeling of both 3D geometry and functionality of a mechanism assembly from a single RGB image. The resulting 3D mechanism model highly resembles the one in the input image with the geometry, mechanical attributes, connectivity, and functionality of all the mechanical parts prescribed in a physically valid way. This challenging task is realized by combining various deep convolutional neural networks to provide high-quality and automatic part detection, segmentation, camera pose estimation and mechanical attributes retrieval for each individual part component. On the top of this, we use a local/global optimization algorithm to establish geometric interdependencies among all the parts while retaining their desired spatial arrangement. We use an interaction graph to abstract the inter-part connection in the resulting mechanism system. If an isolated component is identified in the graph, our system enumerates all the possible solutions to restore the graph connectivity, and outputs the one with the smallest residual error. We have extensively tested our system with a wide range of classic mechanism photos, and experimental results show that the proposed system is able to build high-quality 3D mechanism models without user guidance.

CCS Concepts

•***Computing methodologies*** → ***Image processing; Shape modeling; Neural networks;***

1. Introduction

Modeling a 3D mechanism assembly consisting of various CAD parts is a labor intensive duty. It is normally dealt with professional packages designed for expert users like `Autodesk Inventor` or `SolidWorks`. This problem becomes even more challenging if one wishes to create a model that matches a specific real-world target. Many iterations will be needed to edit-and-test different geometry variations and inter-part connectivity while maintaining the similarity between the model and the target. Driven by this fact, considerable research efforts have been devoted to provide a more intelligent interface to assist regular users to create 3D mechanis-

† Corresponding author

https://doi.org/10.1111/cgf.13572.

m models by embedding advanced 3D modeling and geometric analysis algorithms into the modeling pipeline. For instance, Xu et al. [XLX*16] designed a system that takes multiple (typically a few dozen) images of a target mechanism assembly, which are roughly fused and annotated interactively by the user with a stroke-based interface to extract mechanical parts. Lin et al. [LSZ*18] proposed a mechanism modeling system based on raw depth scans of the target mechanism assembly. While this system does not require as frequent user interference as [LSZ*18], obtaining a 3D mechanism reconstruction needs extra hardware supports and extended pre-processing efforts.

The goal of our paper is to make the acquisition of 3D mechanism as simple as taking a photograph: the user only needs to provide a single color photograph of the target mechanism, and our system automatically returns a 3D mechanism model that well matches the input image. More importantly, the geometry, mechanical attributes, connectivity and functionality of each visible mechanical part are also prescribed by our system in a physically feasible way. Based on the recovered part geometry and interdependency, we can illustrate how mechanical assemblies work as in [MYY*10] or augment the input image with animated part motions to inspire and educate wider audience.

Obviously, an automatic construction of both geometry and functionality of a mechanism assembly using just a single image is a challenging problem. This is because the camera projection degenerates the original 3D geometry, and without 3D information there exist too many ambiguities to precisely interpret the flattened geometry and kinematic connections between mechanical parts. Our solution to this technical challenge is to leverage the deep learning technique to infer the necessary information out of the input image. Specifically, we use various powerful deep convolutional neural networks (CNNs) to enable automatic detection, segmentation, camera pose estimation and mechanical attributes retrieval for visible mechanical parts from the input. Based on this CNN-facilitated information mining, we recover the interdependency among all the mechanical parts via local and global optimizations so that the resulting mechanism assembly is physically valid, and it resembles the input image as much as possible. It is known that the effectiveness of CNNs highly depends on the quality of the training data. We leverage the fact that a mechanism photo only contains standard CAD objects, and generate dedicated CNN training sets to optimize various networks used. Our system is also robust, even in the situation of part occlusion. This is achieved by "guessing" the missing parts based on the existing knowledge of the mechanism assembly. Our system enumerates all the possible solutions to the occlusion and outputs the one with the smallest residual error in the global optimization stage. We tested our algorithm on a variety of mechanism photos, which are classic mechanical tools like the gear train, the worm gear drive arrangement, the cam mechanism and so on. The reconstructed functional model can be used for further motion illustration or 3D animation. In particular, the technical contribution of this paper can be summarized as follows:

- To the best of our knowledge, this is the first fully automatic system that takes only a single RGB image as input to recover both 3D geometry and functionality of a complicated mechanism assembly.
- Our system utilizes various CNN architectures to enable automatic and high-quality part detection, segmentation, camera pose estimation and mechanical attributes retrieval. We generate dedicated CNN training data sets to fully optimize each employed network.
- We formulate the part interdependency as an optimization problem, which consists a local pose adjustment for each individual part and a global adjustment after the part interdependency is inferred by its interaction graph.
- We enhance the system robustness by a novel occlusion amendment algorithm. The occlusion is detected when the kinematic chain is not able to propagate over all the parts. After that, our system enumerates all the possible solutions to restore the graph connectivity, and outputs the one with the smallest residual error.

2. Related Work

Mechanism modeling and interactive design The problem of mechanism modeling is not merely about recovering the 3D geometry of a given mechanism target. It is equally important to obtain its inter-part motion pattern and the system-wise functionality. As a seminal work on this problem, Mitra et al. [MYY*10] inferred parts' motion by analyzing interactions among individual parts based on the geometry of input mechanical assemblies. Creating a physically functional mechanism model that matches a real-world target is a tedious and labor intensive task. To expedite this procedure, Xu et al. [XLX*16] proposed an interactive approach to recover parts' shapes and their internal motion structures of the target mechanism from multi-view images. Ureta et al. [UTZ16] created physically realizable joints from initial mechanical parts and adapted the part geometry to the desired motion ranges when necessary. Lin et al. [LSZ*18] recovered the function of mechanical assemblies from raw depth scans by extracting parametric parts as well as their mechanical restrictions. These methods either require frequent user interactions or a time-consuming scanning process, while our approach is fully automatic and only needs a single image as input. The most relevant work is [HL15], which produces a functional 3D model from a 2D mechanism design. However, the 2D design is restricted to the planar mechanism, and all the mechanical parts must be given, whereas our method is able to process the image captured from an arbitrary viewpoint and extract the parts from the image automatically.

Different from mechanism modeling which aims to reconstruct geometry/motion of an existing mechanism, the goal of interactive design is to computationally design the geometry/motion according to the user's desires. Our work is also related to such interaction techniques that aid the non-experts with the mechanism design [ZXS*12, CLM*13, CTN*13, KLY*14, TCG*14, MZB*17, SWT*17, ZAC*17]. These contributions focus on facilitating designs of personalized mechanical artifacts from scratch under progressive user guidance. Our goal is different: the constructed 3D model should not only have correct motion, but also well match the input image.

Single-image based 3D modeling Recovering 3D geometry from a single image is extensively studied in computer vision and computer graphics communities. Since the problem is inherently ill-posed, many works tried to solve the problem by impos-

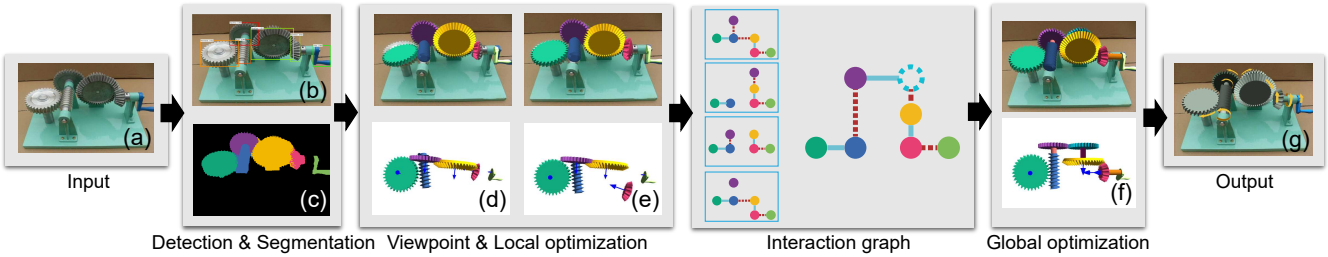

Figure 2: *An overview of the proposed single-image mechanism modeling system.*

ing constraints such as geometric priors [WSB05] and shape symmetry [JTC09]. It is also possible to utilize 3D models from a pre-built database [XZZ*11, HWK15] or 3D geometry proxies [ZCC*12, CZS*13, SD15] to approximate the object in the image. In our paper, the nature of mechanism systems allows us to narrow the focus on standard CAD models so that a representative part database can be built. We fine-tune the poses and shapes of mechanical parts to maximize the input-output similarity while retaining a physically correct inter-part motion pattern.

Recently, some research also leverages deep learning techniques [DMBR16] on large data sets. For example, the 3D-R2N2 technique attempted to recover 3D voxels from a single or multiple photographs [CXG*16]. Fan et al. [FSG17] recovered a dense set of 3D point clouds using the generation network. However, these methods cannot be directly applied to the 3D mechanism modeling as it is still not clear to us how to retrieve the motion information out of static voxels or point clouds.

CNN-based image understanding Deep neural networks have demonstrated significant success in image-based object detection and semantic segmentation tasks [LAE*16, HGDG17, LGG*17]. For example, Long et al. [LSD15] built fully convolutional networks (FCNs) without any fully connected layers that take input of arbitrary size and produce correspondingly-sized image output with efficient inference and learning. The Faster R-CNN [RHGS15] enabled real-time object detection with region proposal networks (RPN) integrated with the classifier net. Mask R-CNN [HGDG17] extended the Faster R-CNN by adding a branch of predicting object masks on top of bounding box extraction. Observing RPNs can only capture the rough shape of the object, its extensions like [BST16, LMSR17, PLCD16] aim to improve the segmentation boundary. Other semantic segmentation methods including [DHS16, DHL*16] used multi-task cascaded networks to identify instances with position-sensitive score maps. FCIS used inside and outside maps to preserve the spatial extent of the original image [LQD*17]. Our framework utilizes a powerful baseline system, which solves the objection detection and semantic segmentation problem with the Faster R-CNN [RHGS15] and FCNs [LSD15] respectively. These methods are not only conceptually intuitive, but also tends to be flexible, robust as well as fast in both training and inference [ISS17].

There also exists some other CNN-based work which estimates the camera viewpoint of the input image like Render-for-CNN [SQLG15]. We extend this work by not only estimating camera viewpoint, but also estimating some key mechanical and geometry attributes, like the tooth number of a gear part, which faciliates the subsequent part parametrization and instantiation.

3. System Overview

Fig. 2 sketches an overview of the proposed mechanism modeling system. The input of our system is a single RGB image of a given mechanism assembly (i.e. Fig. 2 (a)), and the output is a 3D model, of which the kinematics and geometry of each mechanical part are prescribed so that the reconstructed mechanism functions in a physically feasible way. It is assumed that the input image reveals the vision of most mechanical parts, nevertheless our system is also able to deal with occlusions if few mechanical connectors are missed/occluded in the original input. The input image is passed forward to a Faster R-CNN network [RHGS15], which returns types and bounding regions of all the visible mechanical parts (Fig. 2 (b)). We segment each type of mechanical part using a separate FCNs network [LSD15], and the result is further refined with the conditional random fields (CRF) method [KK11] (Fig. 2 (c)). In the next step, our system retrieves 3D part shape from a pre-built part database for each 2D part segment. To this end, we adopt a method similar to Render-for-CNN [SQLG15] that leverages a large volume of synthetic images by rendering the part database as the network training set. We replace the standard AlexNet used in the original Render-for-CNN with the ResNet [HZRS16], which is able to better utilize the network's depth and significantly improve prediction accuracy. At this stage, our system does not only retrieve the best matching 3D shape of each mechanical part, but also estimates camera poses and key mechanical attributes of the part such as its tooth number (Fig. 2 (d) & (e)). Parts interconnect with each other according to their types and spatial connectivity between adjacent neighbors, which is abstracted with an interaction graph. A global optimization procedure is followed to fine-tune the poses of all the parts so that the spatial constraints between adjacent parts are well satisfied, and the kinematic chain can be propagated in a physically-feasible manner. When there exist several possible assemblies with different interaction graphs, the assembly with the lowest optimization residual is chosen as the final output. Next, we elaborate each major step of our system in the following sections.

4. Mechanical Parts Detection & Segmentation

The first task, after an input picture of a certain mechanism is provided, is to determine what mechanical parts are visible in the pic-

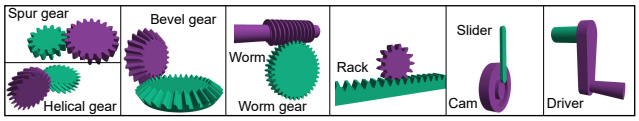

Figure 3: *In the detection stage, we focus on standard mechanical parts including spur gear, helical gear, bevel gear, worm gear, worm, rack, cam, slider and driver.*

ture, and what types of these parts are. Most existing works rely on users' guidance, e.g. using the stroke-based interface [XLX*16] to extract relevant information. Clearly, such processing is tedious and error-prone – a mistaken initial labeling fails all the following processing and leads to a completely faulty result. One of the major advantages of our system is to leverage the deep learning methods to make the processing pipeline fully automatic.

Mechanical part detection Unlike regular object detection tasks, where hundreds of different types of objects need to be recognized (e.g. in ILSVRC [RASC14]), for the problem of mechanism modeling we only focus on standard mechanical parts, namely spur gear, helical gear, bevel gear, worm gear, worm, rack, cam, slider and driver (shown in Fig. 3). The availability of mechanism pictures is quite limited, and we only managed to collect 1,058 images from the Internet (via `Google image search`). In addition, we generate 3,296 synthesized images. Each of the real-world or synthesized images is also mirrored along the vertical middle line, yielding 8,708 training images in total. The test set for detection consists of 336 images, and all of them are real-world pictures. Few examples of real pictures and synthesized ones are given in Fig. 4.

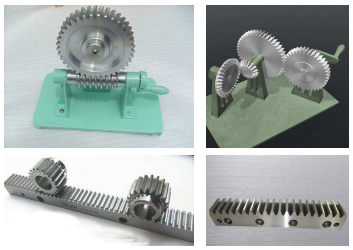

Figure 4: *In addition to real-world pictures collected from Internet, we also synthesize many mechanism images to make sure there are sufficient training data.*

Before detecting a mechanical part and identifying its category, a necessary step is to hypothesize its position in the input image, or the *region proposal*. We follow the method of Faster R-CNN, which uses the RPN to efficiently predict rectangular regions that potentially hold a mechanical part [RHGS15]. A sliding window is used over the features obtained from convolutional layers of the network. At each sliding window location, multiple region proposals are evaluated by combining different scales and aspect ratios of proposed rectangular region. We adopt the default setting of Faster R-CNN, which uses 3 scales and 3 aspect ratios, leading to 9 different *anchors*. With region proposals generated by the RPN, a Fast R-CNN [Gir15] which follows a VGG-16 structure [SZ14] is used to test if a mechanical part is indeed in the proposed region and if yes, what type it belongs to. The average precision (AP) of detecting each mechanical part is reported in Tab. 1.

Mechanical part segmentation As shown in Fig. 2 (b), the Fast R-CNN used in our detection stage outputs the type label and bounding box of each detected mechanical part, and we need to extract the part out of its bounding box in order to estimate the part's geometry and mechanical attributes. The segmentation task is handled by FCNs [LSD15], which also follow the VGG-16 network structure. In the segmentation phase, each type of mechanical part is trained using FCNs independently. We generate the training data by manually cropping parts from training pictures. Each cropped part is scaled to a 224×224 square image, and we create a mask extracting the part from the cropped image using the (supervised) paint selection method [LSS09]. The sizes of training and test data sets for each part type are listed in Tab. 1. We set the learning rate as 0.0001 for the FCNs training. The segmentation result from FCNs is further refined using a CRF-based optimization to improve the pixel labeling nearby the boundary of the mechanical part. As shown in Fig. 6, boundary geometry features of the part segmentation are better revealed after the CRF refinement, which improves the accuracy of inferred part geometry and mechanical attributes.

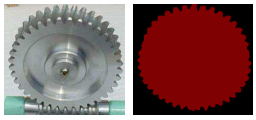

Figure 5: *We use the paint selection method to extract a mechanical part to build the training data for the FCNs.*

Figure 6: *The part mask obtained from the FCNs is refined based on CRF. After the refinement, subtle geometry features at the segmentation boundary are more clearly revealed.*

5. Part Parametrization and Instantiation

With all the visible mechanical parts in the input image extracted, we need to perform the parametrization and instantiation. The parametrization procedure infers a part's key mechanical attributes including the number of tooth (for gears, racks and worms) and the helix angle (for helical gears). Afterwards, the instantiation procedure retrieves its 3D geometry. This is a challenging problem because our system only takes a single input image. Conventional computer vision and graphics methods are not able to restore the depth information just from a single RGB image. We again, exploit the deep learning to tackle this technical obstacle.

During the parametrization, we also estimate camera pose, which is represented by the azimuth, elevation and tilt angles. They are discretized into 360, 180 and 360 intervals respectively. Each interval corresponds to a 1° span. Tooth numbers are discretized into 50 bins from 6 to 55 for spur gears, 26 bins from 15 to 40 for helical gears, and 36 bins from 15 to 50 for worm gears. Each bin contains only one tooth number. The helix angle is discretized into 9 bins from 15° to 60°, where a bin spans 5°. In other words, we

Part category	Spur gear	Helical gear	Bevel gear	Worm gear	Worm	Rack	Cam	Slider	Driver
Detect AP	93.79%	98.66%	90.45%	96.89%	96.53%	86.98%	98.02%	90.33%	89.46%
Seg. data size	3,561	1,298	1,440	1489	1,582	1,385	991	615	572
# 3D models	564	670	558	412	324	570	45	10	31
# Images	138*K*	282*K*	119*K*	100*K*	60*K*	150*K*	177*K*	30*K*	93*K*
Azimuth ($\pm 5^\circ$)	97.7%	98.9%	98.6%	97.8%	100%	99.4%	99.8%	98.7%	96.3%
Elevation ($\pm 5^\circ$)	93.5%	96.6%	95.9%	93.8%	99.8%	99.3%	99.4%	95.6%	92.2%
Tilt ($\pm 5^\circ$)	95.6%	98.7%	97.0%	95.9%	100%	99.3%	95.3%	92.3%	87.7%
# Tooth	84.8%	97.9%	85.3%	85.7%	99.7%	99.2%	–	–	–
Helix angle	–	98.7%	–	–	–	–	–	–	–

Table 1: *Training statistics for each mechanical part category.* ***Detect AP*** *is the detection average precision of each part at the detection stage.* ***Seg. data size*** *is the training and test data size used in the segmentation training. For each part type, 80% of the data are for network training and the remaining 20% are used for testing.* ***# 3D models*** *is the number of 3D part models in the database.* ***# Images*** *is the total number of synthesized images for camera viewpoint estimation.* ***Azimuth***, ***Elevation*** *and* ***Tilt*** *are the prediction accuracy of these three camera viewpoint angles. We set accuracy tolerance as* $\pm 5^\circ$ *meaning a prediction is considered accurate as long as the error is within* 5°. ***# Tooth*** *and* ***Helix angle*** *are the prediction accuracy of tooth number and helix angle with zero error tolerance.*

shape the part parametrization as a classification problem instead of a nonlinear regression, which allows us to better utilize existing CNN architectures.

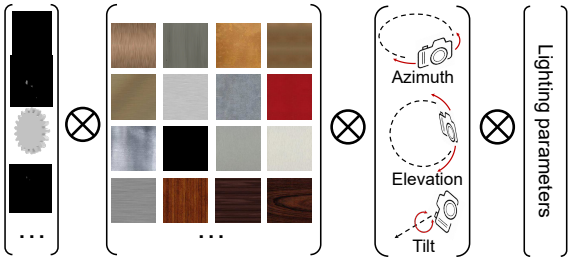

Figure 7: *In order to have sufficient training data for the ResNet, we generate over* 1.1 *millions synthetic 2D images by rendering 3D part models of different parameters with various textures, lighting and camera poses.*

Training data generation A 3D database consisting of various mechanical part models is built. On the top of it, we generate a large volume of synthetic training images (over 1.1 million) by rendering models in the database with known camera poses and parts' specifications to make sure that there are sufficient training data to optimize the network. All the part models in the database are within a unit bounding cube, and their geometry centers are placed at the origin of the world coordinate system. For models exhibiting certain geometric symmetry, we pose them in the way such that their primary symmetric axes are aligned with the z axis of the world coordinate system. For instance, if the model is a gear, its gear axis is in z direction, and if the model is a worm, the neural axis of its shaft is in z direction etc. As to be discussed later, doing so allows us to tweak local/global scaling of a part without destroying its mechanical functionality. The virtual camera for rendering is placed on a sphere surface with radius of 2.0. The camera always faces to the origin. The azimuth, elevation and tilt angles are evenly divided into 72, 36 and 72 sample intervals. Each interval spans 5°. The camera viewpoint is determined by picking a random angle out of each sample interval.

As each mechanical part has already been extracted from the input image, we do not render the background. The texture of the part however, is selected out of a texture palette (consisting of 23 common textures of CAD models) as shown in Fig. 7. The lightings are sampled in the same way as in [SQLG15]. The total numbers of 3D part models in the database and synthesized images are reported in Tab. 1 (i.e. rows **# 3D models** and **# Images**). We use 80% of the generated images for network training and the rest 20% for testing. One may notice that we have fewer 3D models for cam, slider and driver. This is because these mechanical parts have less geometric variation than others.

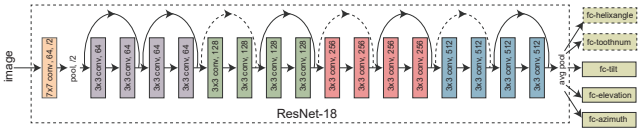

Figure 8: *The network structure of ResNet-18 used in our system.*

Network structure & performance In [SQLG15], a standard AlexNet [KSH12] is used for camera viewpoint estimation. We however, find that regular deep CNNs yield noticeable errors of both the viewpoint angle and gear tooth number. The prediction accuracy is barely above 80%. We conjecture this is because that the additional tasks of predicting the tooth number and helix angle escalate the nonlinearity between the network's input and output, and make the network training more difficult. To overcome this problem, we choose to use the deep residual net (ResNet) [HZRS16]. ResNet equips *shortcut connections* to push the network to optimize the residual error instead of the original loss (Fig. 8). Such differential-like operation reduces the degree of the nonlinearity, and the depth of the network can be fully utilized. As reported in Tab. 1, ResNet (with 16 hidden layers) well estimates the camera viewpoint, tooth number and helix angle. The test accuracy is often above 95% for viewpoint angles. Here, we consider the network

yields an accurate viewpoint prediction if the difference between the predicted angle and the ground truth is less than 5°. The prediction accuracy of the tooth number and helix angle are also impressive. Note that the accuracy reported for tooth number and helix angle prediction is the exact accuracy (with zero error tolerance). The tooth number/helix angle error is always less than $5/5^\circ$. That is to say, even if the network does not give the exact tooth number or the helix angle, the predicted value is still very close to the ground truth. We also tested the performance of the trained ResNet on a few real images by comparing the predicted the tooth number with the manually examined value (the helix angle in a real-world picture is difficult to be accurately assessed even by a human user). The prediction accuracy is also above 90% under the tolerance of ± 5.

Part instantiation We retrieve 3D shapes for each detected and segmented part. This is done by querying for the 3D model in the database whose 2D projection (using the viewpoint estimated by the ResNet) best matches the extracted part segment. For gears, racks and worms with their tooth numbers predicted by the ResNet, we require the instantiated 3D model has the same tooth number. If the part is a helical gear, we also require that the predicted helix angle is the same as on the instantiated one.

Camera viewpoint estimation As we place all the 3D models at the origin of the world coordinate system, their projections are all centered at the resulting synthetic renderings. However, this is not the case for a part segment extracted from the input, which are typically translated or skewed. Let I_{seg} be a $w_{seg} \times h_{seg}$ sub-image containing a detected part segment, and w and h be the width and height of the input image. The camera intrinsic parameter matrix $\mathbf{K}$, assuming the skew angle is 90°, is:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

where we set $f_x = f_y = \max\{w,h\}$, and $c_x = w/2$, $c_y = h/2$. With ResNet estimated viewpoint angles, we obtain an extrinsic camera parameter matrix $\mathbf{T} \in \mathbb{R}^{3\times 4}$ as $\mathbf{T} = \left[\mathbf{R}(\alpha,\beta,\gamma), [0,0,2]^\top\right]$ for each I_{seg}. Here $\mathbf{R} \in \mathbb{SO}_3$ is the rotation matrix derived from azimuth (α), elevation (β) and tilt (γ) angles. The translation vector is set as $[0,0,2]^\top$ as this is the default (virtual) camera position for the rendering. The camera projection $\pi(\mathbf{K},\mathbf{T})$ that maps a world coordinate to an image coordinate can then be derived. Here, we ignore the coordinate homogenization for a succincter formulation.

Local optimization Mechanical parts in the input image have various positions and orientations, which are different from the default layout of the instantiated model. Therefore, we need to find out a local transformation to better align the synthesized 2D projection and the extracted part from the original input (I_{seg}). Specifically, we create another image I_{pro} by projecting the instantiated 3D model with π. Clearly, I_{pro} will sit at the center of the resulting image plane, and the contour of the part differs from the one in I_{seg}. We first calculate an initial translation to align the centers of I_{seg} and I_{pro} as: $\mathbf{t}_0 = \pi^{-1}\left([x_{seg}, y_{seg}, 2]^\top\right)$. Here, x_{seg} and y_{seg} are the image coordinate of I_{seg}'s center in the input image. In addition, we also apply an initial uniform scaling $\mathbf{S}_0$ to roughly match the dimension of I_{seg} and I_{pro} so that their widths (if $w_{seg} > h_{seg}$) or heights (if $w_{seg} \le h_{seg}$) are of the same size.

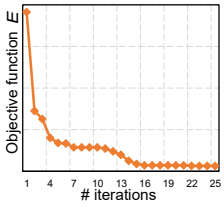

Figure 9: *A typical convergency curve for the local optimization.*

The local optimization stage seeks for a per-part transformation, which consists of a scaling $\mathbf{S}(s_x,s_y,s_z)$, a rotation $\mathbf{O}(o_x,o_y,o_z)$ and a translation $\mathbf{t} = [t_x,t_y,t_z]^\top$ so that when applied to the instantiated mechanical model, the part contour in I_{pro} resembles the one in I_{seg} as much as possible. s_x is set to be equal to s_y. Recall that all the CAD models in the database have their primary axes aligned with the z axis, the constraint of $s_x = s_y$ provides more shape variations than a uniform scaling does without losing the part's original mechanical functionality. For instance, this setting allows us to tweak radius and thickness of a gear while maintaining its circular shape. We first identify a set of correspondence vertices pairs in order to formulate our target function. The correspondence vertices pairs of a part form a set:

$$\mathcal{C} = \{ < \mathbf{v}_i, \widetilde{\mathbf{v}}_i > : \|\pi(\mathbf{v}_i) - \widetilde{\mathbf{v}}_i\| < \tau \}, \tag{2}$$

where $\widetilde{\mathbf{v}}_i \in \mathbb{Z}^2$ represents a pixel at the contour of the segmented part in I_{seg}. $\mathbf{v}_i \in \mathbb{R}^3$ is a vertex on the instantiated 3D model. In other words, $\mathcal{C}$ contains vertices on the 3D model which are close to the part contour in I_{seg} (the distance threshold τ is set as 30 pixels) under the camera projection π. The target function of the local optimization can then be formulated as:

$$E(\mathbf{S},\mathbf{O},\mathbf{t}) = \frac{1}{|\mathcal{C}|} \sum_{<\mathbf{v}_i,\widetilde{\mathbf{v}}_i>\in\mathcal{C}} \|\pi(\mathbf{v}_i) - \widetilde{\mathbf{v}}_i\|^2 + \lambda_{center} \|\mathbf{c}(I_{seg}) - \mathbf{c}(I_{pro})\|^2 + \lambda_{area} \left(\frac{|\mathcal{A}(I_{seg})| - |\mathcal{A}(I_{seg} \cap I_{pro})|}{|\mathcal{A}(I_{seg})|} \right)^2. \tag{3}$$

Here $\mathbf{c}(I_{seg})$ and $\mathbf{c}(I_{pro})$ are the image centers of I_{seg} and I_{pro}. $\mathcal{A}(I_{seg})$ and $\mathcal{A}(I_{pro})$ are sets of pixels that are inside of the part's contour in I_{seg} and I_{pro}. $|\mathcal{A}(I_{seg} \cap I_{pro})|$ denotes the pixel count in part's overlapping area. Two weight coefficients λ_{center} and λ_{area} are both set as 0.5 in our implementation. We use the L-BFGS method to optimize Eq. (3). After each iteration, the correspondence set $\mathcal{C}$ is updated. $\mathbf{t}_0$ and $\mathbf{S}_0$ are used as the starting values of $\mathbf{t}$ and $\mathbf{S}$, and $\mathbf{O}$ is initially set as an identity matrix. Thanks to these initial alignments, the optimization always reaches good minimum in our experiments (Fig. 9), and we set the maximum number of iterations as 20.

6. Functionality Reconstruction

A collection of 3D mechanical parts bears little insightful system-level information of the target mechanism. In order to retrieve a physically-valid mechanism model, we need to infer the functional and geometrical interdependency among all the parts. At the first sight, this task appears similar to existing mechanism modeling systems [XLX*16, LSZ*18]. However, this problem is much more challenging in our system. This is because our system takes as input a single RGB image. The depth information of an instantiated part is largely based on an assumed initial value (i.e. 2.0, which is default camera depth for training data generation). Therefore, even

though the ResNet is able to accurately predict the mechanical parameters and the camera viewpoint, the depth information of 3D parts is much less accurate. As a result, when inferring the possible connection between two adjacent parts, we make most use of the information of the part's orientation (i.e. its primary axis) and its projection on the input image. Its position in 3D obtained from the local optimization, on the other hand, turns out to be less helpful.

Interaction graph We encode interdependencies among all the part components with an *interaction graph*. An interaction graph is an undirected graph $\mathcal{G}$, which consists of a vertex set $\mathcal{V}$ representing all the mechanical parts and an edge set $\mathcal{E}$ that abstracts geometric constraints among parts including *meshing*, *parallel*, *coaxial* and *orthogonal*. Spatial relations of parallel and orthogonal between two parts P_a and P_b are identified by examining their primary axes $\mathbf{n}(P_a)$ and $\mathbf{n}(P_b)$. Concretely, P_a and P_b are considered parallel or $P_a \parallel P_b$ if $\langle \mathbf{n}(P_a), \mathbf{n}(P_b) \rangle < \varepsilon$. P_a and P_b are considered orthogonal or $P_a \perp P_b$ if $\langle \mathbf{n}(P_a), \mathbf{n}(P_b) \rangle < \pi/2 \pm \varepsilon$. The threshold angle ε is set to $\pi/6$. When P_a and P_b are in contact in the input image, and the combination of their types and spatial relation follows one of the cases listed in Tab. 2, we say that P_a and P_b are meshing each other or $P_a \sim P_b$, and an edge is created between them in $\mathcal{G}$. Fig. 10 elaborates this step with the same the example of Fig. 2, wherein we have six segmented parts in total from the input image. According to their types and spatial relations, a worm gear–worm edge and a bevel gear–bevel gear edge are created representing the meshing relation between them.

It is unlikely that $\mathcal{G}$ becomes a fully-connected graph only by the meshing relation. As shown in Fig. 10, there are two unconnected vertices in the interaction graph after all the meshing edges are inserted. Those isolated vertices may be connected by the coaxial relation. The coaxial relation does not require a hard engagement between two parts in the input image, and it is a special case of parallel. Therefore, before checking if P_a is coaxial to P_b or $P_a \simeq P_b$, we first ensure that $P_a \parallel P_b$. Ideally, $P_a \simeq P_b$ implies that P_a, P_b are of the same primary axis. However as discussed before, the center positions of parts (denoted as $\mathbf{c}(P_a)$ and $\mathbf{c}(P_b)$) resulted from the local optimization are less accurate and should not be directly used. Instead, we examine their projections on the image plane and determine the coaxial relation by checking two projected distances:

$$\begin{aligned} d_{a,b} &= \|\pi(\mathbf{n}(P_a) \otimes \mathbf{n}(P_a)(\mathbf{c}(P_b) - \mathbf{c}(P_a))) - \pi(\mathbf{c}(P_b))\|^2, \\ d_{b,a} &= \|\pi(\mathbf{n}(P_b) \otimes \mathbf{n}(P_b)(\mathbf{c}(P_a) - \mathbf{c}(P_b))) - \pi(\mathbf{c}(P_a))\|^2. \end{aligned} \quad (4)$$

Here, $\mathbf{c}(P_b) - \mathbf{c}(P_a) \in \mathbb{R}^3$ is the vector from P_a's center to P_b's. This vector is further projected on P_a's axis, and it should hit $\mathbf{c}(P_b)$ if P_a and P_b are strictly coaxial to each other in 3D. However, as the depth information of $\mathbf{c}(P_a)$ and $\mathbf{c}(P_b)$ is not trustable, we examine this value after projecting original vectors on the image plane. We consider $P_a \simeq P_b$ if both $d_{a,b}$ and $d_{b,a}$ are less than $0.1 \cdot \max\{w,h\}$ pixels (recall that w and h are the width and height of the input image). After that, an edge is inserted to connect P_a and P_b via the coaxial relation.

In summary, we somehow simplify the part coupling mechanism and establish a kinematic linkage between P_a and P_b only when $P_a \sim P_b$ or $P_a \simeq P_b$. Besides, the orientation relation between two adjacent parts in $\mathcal{G}$ is limited to be either parallel or orthogonal. While this simplification may not be valid for sophisticated mechanism devices, it works well in practice for many commonly seen mechanism models and significantly improves the robustness of our system.

Part types	Spatial relation
spur gear + spur gear	parallel/orthogonal
spur gear + rack	parallel
worm gear + worm	orthogonal
helical gear + helical gear	orthogonal
bevel gear + bevel gear	orthogonal
cam + slider	parallel

Table 2: *Valid combinations of types and spatial relations for the meshing relation between two parts.*

Occlusion amendment While we expect that most mechanical parts are visible from the input, occlusion is inevitable, which prevents $\mathcal{G}$ from being fully connected. We propose a simple and effective approach to repair the information lost induced by the occlusion and establish necessary linkages connecting all the detected mechanical parts.

First of all, we assume that the occluded part is not at the start or the end of the entire kinematic chain. In other words, the part occlusion does impede the connectivity of the interaction graph. Suppose that $\mathcal{G}_1(\mathcal{V}_1, \mathcal{E}_1)$ and $\mathcal{G}_2(\mathcal{V}_2, \mathcal{E}_2)$ are two connected non-overlapping sub-graph of $\mathcal{G}$ such that $\mathcal{V}_1, \mathcal{V}_2 \subset \mathcal{V}$, $\mathcal{E}_1, \mathcal{E}_2 \subset \mathcal{E}$, and $\mathcal{V}_1 \cap \mathcal{V}_2 = \mathcal{E}_1 \cap \mathcal{E}_2 = \emptyset$. $P_a \in \mathcal{G}_1$, $P_b \in \mathcal{G}_2$ are two parts in $\mathcal{G}_1$ and $\mathcal{G}_2$ respectively, and we try to "derive" another part P_c so that the linkage P_a–P_c–P_b connects $\mathcal{G}_1$ and $\mathcal{G}_2$[†]. In our system, we can have either $P_a \parallel P_b$ or $P_a \perp P_b$, and without losing generality, we assume that $P_a \parallel P_b$, which implicitly requires that $P_a \parallel P_c$ and $P_b \parallel P_c$. We then check the valid type-relation combinations in Tab. 2 and enumerate all the possible types of P_c. For instance, if both P_a and P_b are spur gears, P_c can be either another spur gear or a rack. We do the similar analysis if $P_a \perp P_b$. Our system favors picking $P_a \in \mathcal{G}_1$ and $P_b \in \mathcal{G}_2$ that are close to each other in the input image. The occlusion amendment provides a instrumentality to ensure the connectivity of $\mathcal{G}$. It is possible that the original input image does not miss any parts, but due to bad initial poses, meshing and coaxial edges are not able to fully connect all the parts. In this case, we can still add derived parts to the system as needed making our system robust under ill inputs. P_c is induced to $\mathcal{G}$ because it is not visible from the input image, we penalize the configuration of P_c if it leads to a large visible region when projected under π. This penalty term is added to the global optimization formulation as to be detailed next.

Remark. If a single part P_{c1} is not able to re-establish the connectivity between P_a and P_b, we will examine all the possible candidates P_{c2} to enable either $P_a - P_{c1} - P_{c2} - P_b$ or $P_a - P_{c2} - P_{c1} - P_b$ link that satisfies the spatial/type constraints listed in Tab. 2.

Global optimization With occlusion amendment, the interaction

[†] Clearly, one can always add more parts to connect $\mathcal{G}_1$ and $\mathcal{G}_2$. Our philosophy of occlusion amendment is to add as few parts as possible.

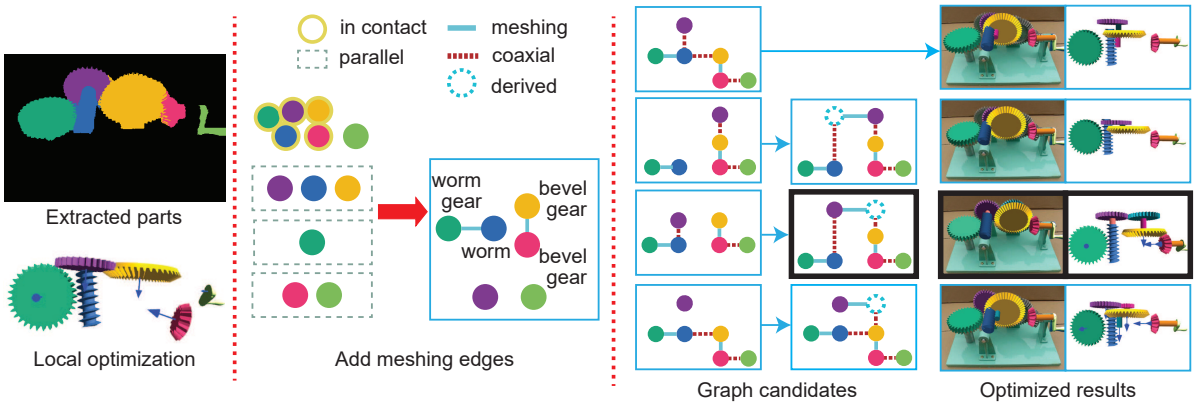

Figure 10: *Our system first clusters vertices via the meshing relation. The remaining isolated vertices are connected via the coaxial relation. If the resulting graph is still not a connected one. We use the occlusion amendment to connect the kinematic chain by derived parts that are invisible from the input image. Doing so could lead to several possible connectivity relations among the parts and this ambiguity is resolved by performing the global optimization. The candidate with the smallest energy will be chosen as our final output.*

graph $\mathcal{G}$ is now connected, and we further adjust the poses of part models to ensure that the geometry constraints indicated by $\mathcal{E}$ are faithfully satisfied while making their 2D projections and the corresponding segments aligned as much as possible. Because parts are now mutually coupled in $\mathcal{G}$, this optimization is carried over all the part models simultaneously including the derived parts for the occlusion amendment. Therefore, we refer to this second round optimization as the global optimization. In the global optimization, we use an alternating strategy by seeking for the optimal rotation/scaling and optimal translation/scaling of all the parts alternatively.

The optimization function for rotation/scaling step is $\sum E$, where E is defined as in Eq. (3) for each part, except that $\mathbf{t}$ is no longer an optimization parameter. In addition, we require that the spatial relation between a pair of adjacent parts is strictly enforced such that if $P_a \parallel P_b$, then $\mathbf{n}(P_a)\cdot\mathbf{n}(P_b) = 1$ etc. Such nonlinear constraints are handled using the trust region method based on interior point nonlinear programming [LWC*11].

The optimization for the translation/scaling step is however, handled in a quite different way. This is again, because the initial position of each part is less accurate. Exactly enforcing part positions as we have done in the rotation/scaling step yields faulty results due to the poor initial value. (Fig. 11). Therefore, we use a soft (penalty-based) constraint to encode the required position constraint for adjacent parts. Specifically, the translation/scaling constraint is formulated as:

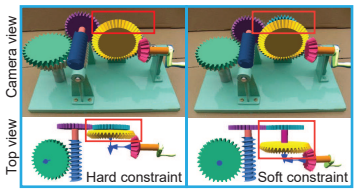

Figure 11: *Strictly enforcing position constraint yields noticeable artifact because of bad initial values.*

$$E_{translation} = \sum E + E_{center} + E_{occlusion}. \tag{5}$$

Similar to the rotation/scaling step, $\sum E$ is the summation of Eq. (3) over all the parts with $\mathbf{O}$ fixed. E_{center} is the penalty energy to push the part center to its ideal configuration $\mathbf{c}'(P_a, P_b)$ given its adjacent part P_b's type and the spatial relation between P_a and P_b:

$$E_{center} = \lambda_{center} \sum_{<P_a,P_b>\in\mathcal{E}} \left\|\mathbf{c}(P_a) - \mathbf{c}'(P_a,P_b)\right\|^2. \tag{6}$$

The detailed formulation of $\mathbf{c}'(P_a, P_b)$ can be found in Tab. 3. The third term $E_{occlusion}$ in Eq. (5) is the penalty term on the visible areas of derived parts. As discussed before, a derived part is supposed to be invisible from the input image. Therefore, we want projections of all the derived parts to be as small as possible, and this penalty term can be intuitively formulated as $E_{occlusion} = \lambda_{occlusion} \sum |\mathcal{A}(\pi(P_i))|$ for all the derived part P_i. Two weight coefficients λ_{center} and $\lambda_{occlusion}$ are set as 5.0 and 0.5 respectively in our implementation. The translation/scaling step is handled using the L-BFGS algorithm. Fig. 12 visualizes the evolution of parts' poses under the global optimization. We can see that, while the input poses from the local optimization are not physically valid, after one rotation/scaling step and one translation/scaling step, the global optimization quickly fixes ill-posed parts and the resulting projection is also very similar to the input image.

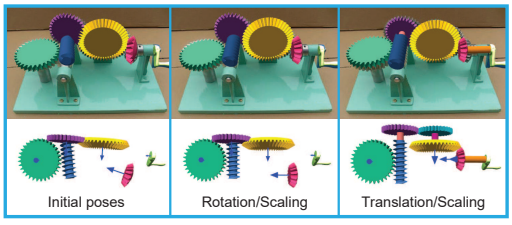

Figure 12: *Visualizing the alternating global optimization procedure. Even the initial poses from local optimization are not particularly satisfying, a single alternating interaction is able to greatly improve the part layout.*

As shown in Fig. 10, in practice there may exist multiple con-

figurations of the interaction graph due to different derived parts used. Our system chooses the one with the smallest residual error after the global optimization and constructs the mechanism model that best fits the input image. For applications such as VR/AR, we extract the background from the original input and overlay it with the rendered result, which enables an animated mechanical and kinematic annotation of the input image. Note that since CNNs provide a good prediction of part parameters (see Tab. 1), optimizing the position/orientation/scale of each part is enough to yield a physically plausible animation, where angular velocities of gears are constrained according to their radius ratios. For physical fabrication, the part parameters can be further refined with the strategy in [LSZ*18].

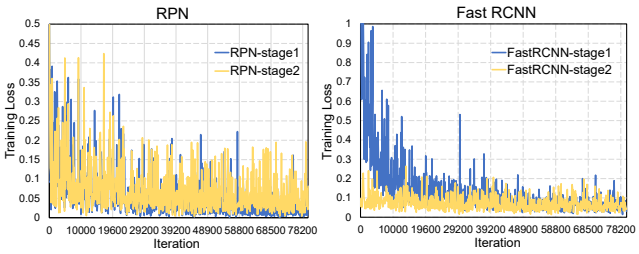

Figure 13: *Converge curves of RPN and Fast R-CNN training (for part detection).*

7. More Results

Our system was implemented on a desktop PC equipped with an `intel i7-4770` CPU@3.4 GHz and 16 GB onboard memory. The computer also houses an `nVidia 1080 Ti` GPU with 11 GB GDDR5 memory. All the deep neural networks used in our system including Faster R-CNN, FCNs and ResNet were constructed and trained with `caffe` framework [JSD*14].

Most training details have already explained and discussed in corresponding sections (i.e. Sec. 4 and Sec. 5). The converge curves for RPN and Fast R-CNN (for the part detection) are plotted in Fig. 13. Recall that we used the two-stage training. In the first stage, which consists of 80K iterations for RPN training and 100K iterations for Fast R-CNN training, we used a more aggressive learning rate of 0.001. In the second stage, which consists of 80K iterations for RPN training and 100K iterations for Fast R-CNN training, a more conservative learning rate of 0.0005 was used. The converge curves for both stages are reported in the figure. In the camera viewpoint and mechanical attributes estimation network, we use the same loss function as in [SQLG15]. In the training process, we set the loss weight as 5.0 for tooth number prediction and 1.0 for other outputs. The training curves of the ResNet used for the camera viewpoint and mechanical attributes prediction are reported in Fig. 14.

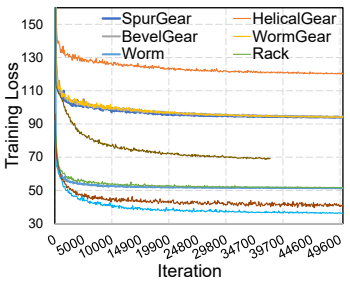

Figure 14: *The converge curve of ResNet training (for camera viewpoint and mechanical attributes prediction).*

To demonstrate the effectiveness of each energy term in the local/global optimization, we compare the modeling results including/excluding the term in the local/global optimization. As shown in Fig. 16, we can see that the optimization with all terms considered achieves the best modeling result.

In addition to the examples reported in Figs. 1 and 2, we have tested our system extensively with input images of various mechanisms, and typical results can be founded in Fig. 15. In the figure, the leftmost and the rightmost columns are the inputs and outputs of the system. The step-by-step snapshots of part detection, segmentation, local and global optimization are reported in the middle columns. In practice, as all the deep neural networks are pretrained, the most time consuming steps along our pipeline are the local and global optimization. Tab. 4 reports the detailed time statistics for all the examples shown in the paper.

In total, we have tested our system with 65 input images of various mechanisms. In these tests, 7 failed at the detection and segmentation stage (1 due to the large occlusion; 1 due to the similar appearance as the background; 1 due to the unseen viewpoint in training data; and 4 due to the unseen shape/appearance in the training data), and 8 failed in the camera viewpoint estimation. Typical failure cases are illustrated in Figs. 18.

To evaluate the robustness of our system, we captured 15 images of the same mechanism from different viewpoints (see Fig. 17). Among these images, 4 failed in detection stage due to large occlusion (e.g. Fig. 18(a)) and 1 failed in the segmentation stage. We obtain consistent modeling results for 9 images, but also get one different modeling result (see the rightmost column in the Fig. 17). It is because one spur gear is missing in the detection stage but the interaction graph is occasionally to be a valid one.

Example	# part	# $\mathcal{G}$	Local	Global
Fig. 1 top left	7	4	1.2 *min*	12.3 *min*
Fig. 1 top right	2	1	0.3 *min*	0.5 *min*
Fig. 1 bottom left	5	4	1.0 *min*	8.3 *min*
Fig. 1 bottom right	2	1	1.1 *min*	1.2 *min*
Fig. 2	7	4	1.1 *min*	12.9 *min*
Fig. 15 top	3	1	0.79 *min*	6.8 *min*
Fig. 15 mid	3	1	0.98 *min*	2.9 *min*
Fig. 15 bottom	3	1	1.1 *min*	2.9 *min*

Table 4: *Time statistics for all the examples shown in the paper.* ***# part*** *is the number of the parts in the example including derived parts.* ***#*** $\mathcal{G}$ *is the number of possible interaction graphs we could have using different derived parts.* ***Local*** *and* ***Global*** *are computation times used for local optimization and global optimization for one interaction graph candidate respectively.*

8. Conclusion and Limitation

In this paper, we present an automatic system that reconstructs both 3D geometry and functionality of a mechanism from a s-

Spatial relation	Type	Target center position
$P_a \simeq P_b$	any	$\mathbf{c}' = \mathbf{c}_b + \mathbf{n}_b \otimes \mathbf{n}_b \mathbf{c}_{a,b}$
$P_a \sim P_b$, $P_a \parallel P_b$	spur gear + spur gear / spur gear + rack	$\mathbf{c}' = \mathbf{c}_b + (r_a + r_b) \cdot \mathbf{n}/\|\mathbf{n}\|, \quad \mathbf{n} = (\mathbf{I} - \mathbf{n}_b \otimes \mathbf{n}_b)\mathbf{c}_{a,b}$
$P_a \sim P_b$, $P_a \parallel P_b$	cam + slider	$\mathbf{c}' = (t_a + t_b)/2d \cdot \mathbf{n}_b, \quad d = \mathbf{c}_{a,b} \cdot \mathbf{n}_b$
$P_a \sim P_b$, $P_a \perp P_b$	bevel gear + bevel gear	$\mathbf{c}' = \mathbf{c}_b - r_b \cdot \mathbf{n}_a + r_a \cdot \mathbf{n}_b$
$P_a \sim P_b$, $P_a \perp P_b$	worm gear + worm / helical gear + helical gear	$\mathbf{c}' = \mathbf{c}_b + (r_a + r_b) \cdot (\mathbf{n}_b \times \mathbf{n}_a)$

Table 3: *The target part center $\mathbf{c}'(P_a, P_b)$ according the type-relation combination of P_a and P_b. Here, we have $\mathbf{n}_a = \mathbf{n}(P_a)$, $\mathbf{n}_b = \mathbf{n}(P_b)$, $\mathbf{c}_a = \mathbf{c}(P_a)$, $\mathbf{c}_b = \mathbf{c}(P_b)$, and $\mathbf{c}_{a,b} = \mathbf{c}_a - \mathbf{c}_b$. r_a, r_b and t_a, t_b are the radius and thickness of P_a and P_b respectively.*

.

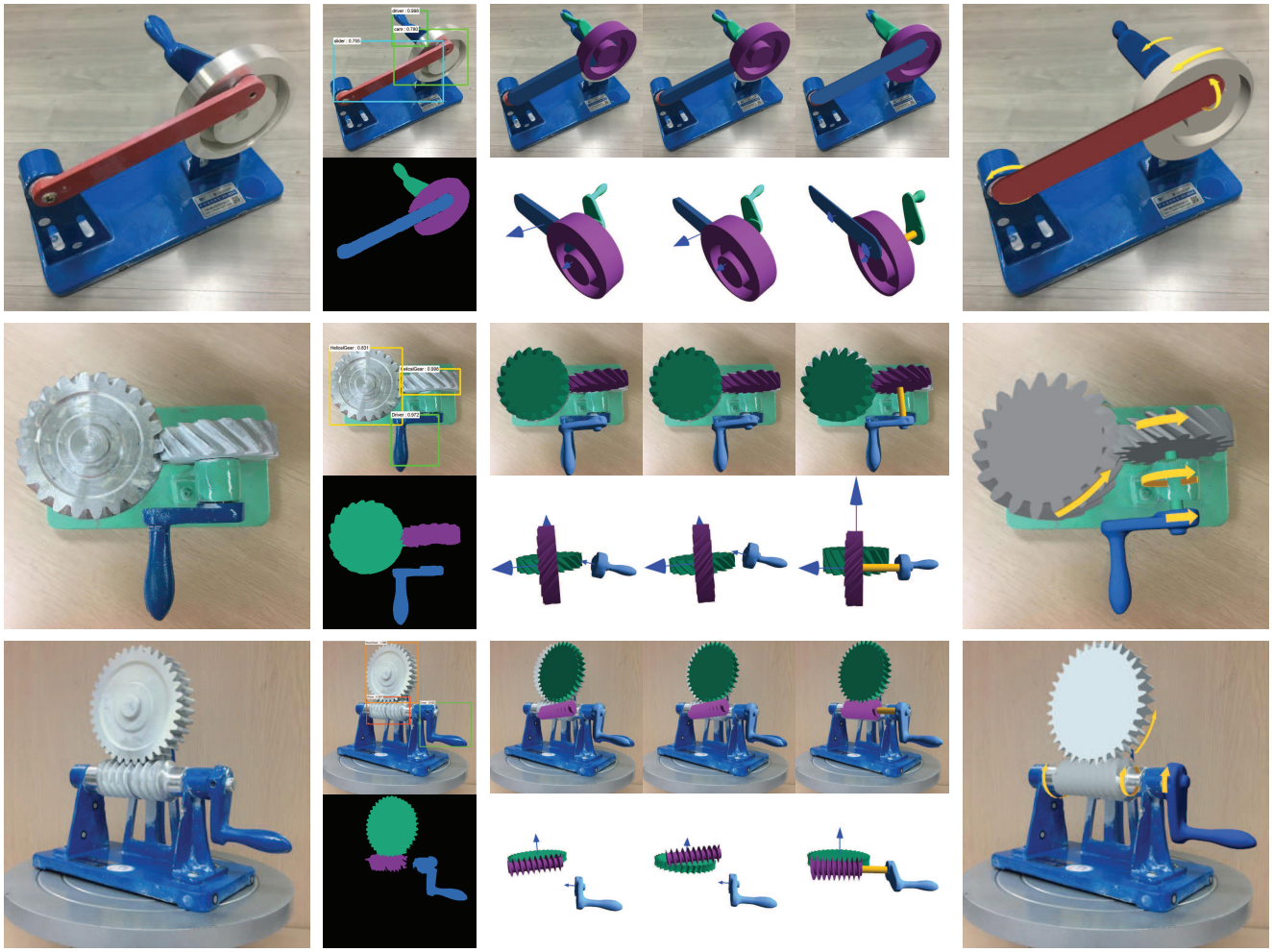

Figure 15: *More mechanism modeling results of our system. The leftmost column is the input images, and the rightmost column is the system output. Intermediate results of detection (with regional proposals), segmentation and optimization are reported in middle columns.*

ingle RGB image. Unlike existing mechanism modeling systems [MYY*10, XLX*16, LSZ*18], our system does not need any user interference along the processing. This is achieved by leveraging various deep CNN architectures to provide high-quality part detection, part segmentation, camera pose estimation, and mechanical attributes retrieval automatically. We utilize the fact that a mechanism image only contains standard CAD models. Based on it, we generate dedicated training data sets for CNNs used in our pipeline. Besides novel applications of CNN, our system also includes robust geometry processing algorithms that extract the interdependencies of detected mechanical parts with an interaction graph. We use a local/global optimization strategy to enforce the correct connectivity between adjacent parts and make the resulting 3D mechanism well match the input image.

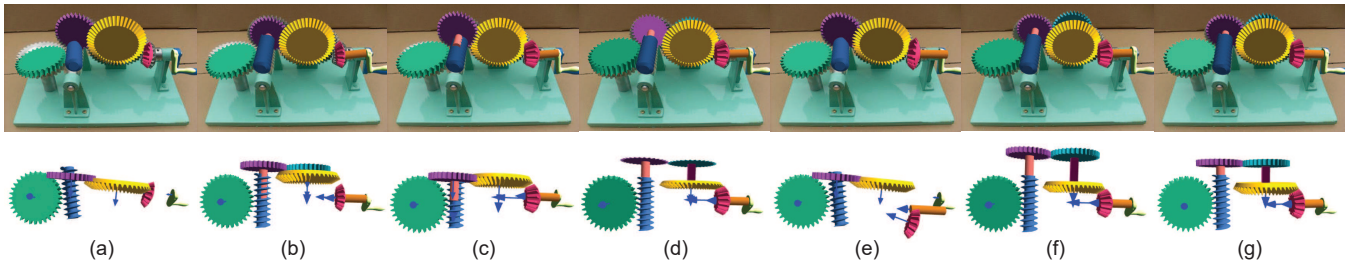

Figure 16: *(a) shows the initial poses of the mechanical parts; (b) to (f) show the modeling results without* $\sum_{<\mathbf{v}_i,\widetilde{\mathbf{v}}_i>\in\mathcal{C}} \|\pi(\mathbf{v}_i)-\widetilde{\mathbf{v}}_i\|^2$, $\|\mathbf{c}(I_{seg})-\mathbf{c}(I_{pro})\|^2$, $\left(\frac{|\mathcal{A}(I_{seg})|-|\mathcal{A}(I_{seg}\cap I_{pro})|}{|\mathcal{A}(I_{seg})|}\right)^2$, E_{center} *or* $E_{occlusion}$ *in the local/global optimization; (g) shows the modeling result with all energy terms considered.*

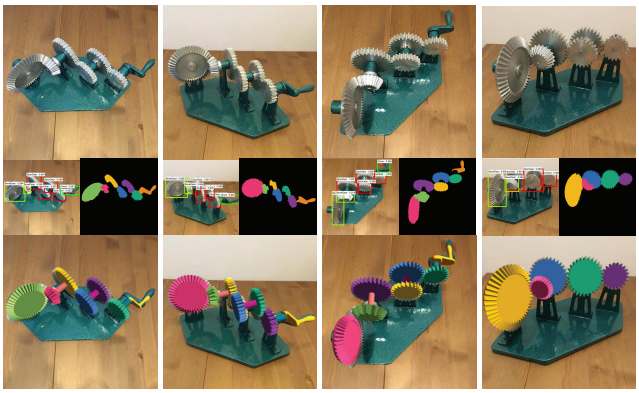

Figure 17: *Modeling results of the same mechanism from images captured from different viewpoints. From top to bottom are the input images, results of detection/segmentation, and modeling results.*

However, there are also some limitations in the current version of our system, which leave us many exciting future works to follow up. Firstly, the scalability is a major limitation of our current algorithm. When a mechanism system becomes more complex, the visible area of each mechanical part on the input image will be smaller, and the CNN based detection algorithm becomes more error-prone. Increased complexity could also lead to more missing parts. As our algorithm enumerates all the possible types of missing parts when an occlusion is detected (i.e. the interaction graph becomes disconnected) and performs the local-global optimization for each graph candidate, more missing parts will significantly slow our pipeline. Secondly, our system simplifies the spatial relation between two adjacent parts to be either parallel/coaxial or orthogonal, which works well for many standard mechanism tools. However, it is also not uncommon to see mechanical parts that are obliquely connected. Being able to incorporate such obliqueness will enhance the applicability of our system. Thirdly, while it is convenient to model the mechanism with a single image, we will investigate the possibility of incrementally refining the result when more images from different perspectives are fed to the system. The automatization of our system is originated from the application of CNNs. If a CNN fails at a certain stage, all the sequential operations are also likely to fail. Three typical failure cases of CNNs are shown in Fig. 18. In Fig. 18(a), two parts are not detected due to heavy occlusion; in Fig. 18(b), semantic segmentation fails as CNNs cannot distinguish the appearance of the rack and the base; in Fig. 18(c), the viewpoint estimation of the bevel gear is incorrect. Finally, it is of great interest to us to extend our system to model other assembled man-made artifacts like furniture [SLR*16], mechanical toys [ZXS*12], or even robots [TCG*14].

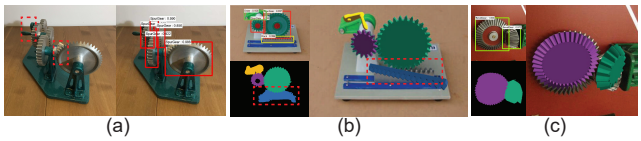

Figure 18: *Three typical failure cases of CNNs.*

9. Acknowledgements

We thank the anonymous reviewers for their feedback. This work is supported in part by the NSF of China (No. 61772462, No. U1736217, No. 61502306, No. 61772458, No. 61572429), Microsoft Research Asia, the China Young 1000 Talents Program, NSF 1717972 and AFRL FA9453-18-2-0022.

References

[BST16] BERTASIUS G., SHI J., TORRESANI L.: Semantic segmentation with boundary neural fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 3602–3610. 3

[CLM*13] CEYLAN D., LI W., MITRA N. J., AGRAWALA M., PAULY M.: Designing and fabricating mechanical automata from mocap sequences. *ACM Trans. Graph. 32*, 6 (2013), 186:1–186:11. 2

[CTN*13] COROS S., THOMASZEWSKI B., NORIS G., SUEDA S., FORBERG M., SUMNER R. W., MATUSIK W., BICKEL B.: Computational design of mechanical characters. *ACM Trans. Graph. 32*, 4 (2013), 83:1–83:12. 2

[CXG*16] CHOY C. B., XU D., GWAK J., CHEN K., SAVARESE S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV* (2016), Springer, pp. 628–644. 3

[CZS*13] CHEN T., ZHU Z., SHAMIR A., HU S.-M., COHEN-OR D.: 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 195. 3

[DHL*16] DAI J., HE K., LI Y., REN S., SUN J.: Instance-sensitive fully convolutional networks. In *European Conference on Computer Vision* (2016), Springer, pp. 534–549. 3

[DHS16] DAI J., HE K., SUN J.: Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 3150–3158. 3

[DMBR16] DWIBEDI D., MALISIEWICZ T., BADRINARAYANAN V., RABINOVICH A.: Deep cuboid detection: Beyond 2d bounding boxes. *CoRR abs/1611.10010* (2016). arXiv:1611.10010. 3

[FSG17] FAN H., SU H., GUIBAS L. J.: A point set generation network for 3d object reconstruction from a single image. In *CVPR* (2017), pp. 2463–2471. 3

[Gir15] GIRSHICK R.: Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)* (Dec 2015), pp. 1440–1448. 4

[HGDG17] HE K., GKIOXARI G., DOLLÁR P., GIRSHICK R. B.: Mask r-cnn. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 2980–2988. 3

[HL15] HERGEL J., LEFEBVRE S.: 3d fabrication of 2d mechanisms. *Comput. Graph. Forum 34*, 2 (2015), 229–238. 2

[HWK15] HUANG Q., WANG H., KOLTUN V.: Single-view reconstruction via joint analysis of image and shape collections. *ACM Trans. Graph. 34*, 4 (2015), 87:1–87:10. 3

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778. 3, 5

[ISS17] IZADINIA H., SHAN Q., SEITZ S. M.: Im2cad. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), IEEE, pp. 2422–2431. 3

[JSD*14] JIA Y., SHELHAMER E., DONAHUE J., KARAYEV S., LONG J., GIRSHICK R., GUADARRAMA S., DARRELL T.: Caffe: Convolutional architecture for fast feature embedding. In *MM '14* (2014), ACM, pp. 675–678. 9

[JTC09] JIANG N., TAN P., CHEONG L.-F.: Symmetric architecture modeling with a single image. *ACM Trans. Graph. 28*, 5 (2009), 113:1–113:8. 3

[KK11] KRÄHENBÜHL P., KOLTUN V.: Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems* (2011), pp. 109–117. 3

[KLY*14] KOO B., LI W., YAO J., AGRAWALA M., MITRA N. J.: Creating works-like prototypes of mechanical objects. *ACM Trans. Graph. 33*, 6 (2014), 217:1–217:9. 2

[KSH12] KRIZHEVSKY A., SUTSKEVER I., HINTON G. E.: Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105. 5

[LAE*16] LIU W., ANGUELOV D., ERHAN D., SZEGEDY C., REED S. E., FU C., BERG A. C.: SSD: single shot multibox detector. In *ECCV* (2016), pp. 21–37. 3

[LGG*17] LIN T., GOYAL P., GIRSHICK R. B., HE K., DOLLÁR P.: Focal loss for dense object detection. In *ICCV* (2017), pp. 2999–3007. 3

[LMSR17] LIN G., MILAN A., SHEN C., REID I.: Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR* (July 2017). 3

[LQD*17] LI Y., QI H., DAI J., JI X., WEI Y.: Fully convolutional instance-aware semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017). 3

[LSD15] LONG J., SHELHAMER E., DARRELL T.: Fully convolutional networks for semantic segmentation. In *CVPR* (2015), pp. 3431–3440. 3, 4

[LSS09] LIU J., SUN J., SHUM H.-Y.: Paint selection. *ACM Transactions on Graphics (ToG) 28*, 3 (2009), 69. 4

[LSZ*18] LIN M., SHAO T., ZHENG Y., MITRA N. J., ZHOU K.: Recovering functional mechanical assemblies from raw scans. *IEEE transactions on visualization and computer graphics 24*, 3 (2018), 1354–1367. 2, 6, 9, 10

[LWC*11] LI Y., WU X., CHRYSATHOU Y., SHARF A., COHEN-OR D., MITRA N. J.: Globfit: Consistently fitting primitives by discovering global relations. In *ACM Trans. Graph.* (2011), vol. 30, ACM, p. 52. 8

[MYY*10] MITRA N. J., YANG Y.-L., YAN D.-M., LI W., AGRAWALA M.: Illustrating how mechanical assemblies work. *ACM Transactions on Graphics-TOG 29*, 4 (2010), 58. 2, 10

[MZB*17] MEGARO V., ZEHNDER J., BÄCHER M., COROS S., GROSS M., THOMASZEWSKI B.: A computational design tool for compliant mechanisms. *ACM Trans. Graph. 36*, 4 (2017), 82:1–82:12. 2

[PLCD16] PINHEIRO P. O., LIN T.-Y., COLLOBERT R., DOLLÁR P.: Learning to refine object segments. In *European Conference on Computer Vision* (2016), Springer, pp. 75–91. 3

[RASC14] RAZAVIAN A. S., AZIZPOUR H., SULLIVAN J., CARLSSON S.: Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPRW, 2014 IEEE Conference on* (2014), IEEE, pp. 512–519. 4

[RHGS15] REN S., HE K., GIRSHICK R., SUN J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (2015), pp. 91–99. 3, 4

[SD15] SALA P., DICKINSON S.: 3-d volumetric shape abstraction from a single 2-d image. In *Proceedings of the IEEE International Conference on Computer Vision Workshops* (2015), pp. 1–9. 3

[SLR*16] SHAO T., LI D., RONG Y., ZHENG C., ZHOU K.: Dynamic furniture modeling through assembly instructions. *ACM Trans. Graph. 35*, 6 (2016), 172–1. 11

[SQLG15] SU H., QI C. R., LI Y., GUIBAS L. J.: Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *ICCV* (2015), pp. 2686–2694. 3, 5, 9

[SWT*17] SONG P., WANG X., TANG X., FU C.-W., XU H., LIU L., MITRA N. J.: Computational design of wind-up toys. *ACM Trans. Graph. 36*, 6 (2017), 238:1–238:13. 2

[SZ14] SIMONYAN K., ZISSERMAN A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014). 4

[TCG*14] THOMASZEWSKI B., COROS S., GAUGE D., MEGARO V., GRINSPUN E., GROSS M.: Computational design of linkage-based characters. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 64. 2, 11

[UTZ16] URETA F., TYMMS C., ZORIN D.: Interactive modeling of mechanical objects. *Eurographics Symposium on Geometry Processing 35*, 5 (2016), 145–155. 2

[WSB05] WILCZKOWIAK M., STURM P., BOYER E.: Using geometric constraints through parallelepipeds for calibration and 3d modeling. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 2 (2005), 194–207. 3

[XLX*16] XU M., LI M., XU W., DENG Z., YANG Y., ZHOU K.: Interactive mechanism modeling from multi-view images. *ACM Trans. Graph 35*, 6 (2016), 236. 2, 4, 6, 10

[XZZ*11] XU K., ZHENG H., ZHANG H., COHEN-OR D., LIU L., XIONG Y.: Photo-inspired model-driven 3d object modeling. *ACM Trans. Graph. 30*, 4 (2011), 80:1–80:10. 3

[ZAC*17] ZHANG R., AUZINGER T., CEYLAN D., LI W., BICKEL B.: Functionality-aware retargeting of mechanisms to 3d shapes. *ACM Trans. Graph. 36*, 4 (2017), 81:1–81:13. 2

[ZCC*12] ZHENG Y., CHEN X., CHENG M.-M., ZHOU K., HU S.-M., MITRA N. J.: Interactive images: Cuboid proxies for smart image manipulation. *ACM Trans. Graph. 31*, 4 (2012), 99:1–99:11. 3

[ZXS*12] ZHU L., XU W., SNYDER J., LIU Y., WANG G., GUO B.: Motion-guided mechanical toy modeling. *ACM Trans. Graph. 31*, 6 (2012), 127–1. 2, 11