How2Sketch: Generating Easy-To-Follow Tutorials for Sketching 3D Objects

James W. Hennessey[1] Han Liu[2] Holger Winnemöller[3] Mira Dontcheva[3] Niloy J. Mitra[1]

[1]University College London [2]KAUST [3]Adobe Research

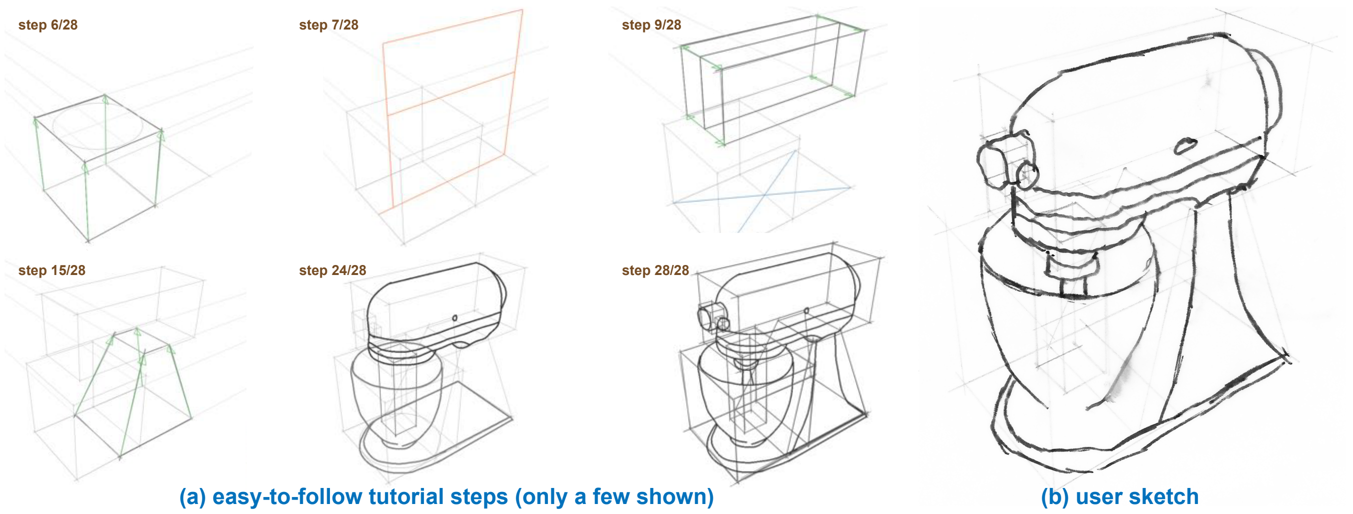

Figure 1: *(a) We present How2Sketch, a system that automatically generates easy-to-follow tutorials for drawing 3D models. Each generated tutorial presents a list of steps for drawing scaffolding primitives that help the user draw the object in correct perspective. To help the user draw the scaffolding, the tutorial shows how to construct guidelines that anchor object parts relative to one another. User study feedback on the tutorials indicates that users feel they are able to create more accurate drawings (b).*

Abstract

Accurately drawing 3D objects is difficult for untrained individuals, as it requires an understanding of perspective and its effects on geometry and proportions. Step-by-step tutorials break the complex task of sketching an entire object down into easy-to-follow steps that even a novice can follow. However, creating such tutorials requires expert knowledge and is time-consuming. As a result, the availability of tutorials for a given object or viewpoint is limited. How2Sketch (H2S) addresses this problem by automatically generating easy-to-follow tutorials for arbitrary 3D objects. Given a segmented 3D model and a camera viewpoint, H2S computes a sequence of steps for constructing a drawing scaffold comprised of geometric primitives, which helps the user draw the final contours in correct perspective and proportion. To make the drawing scaffold easy to construct, the algorithm solves for an ordering among the scaffolding primitives and explicitly makes small geometric modifications to the size and location of the object parts to simplify relative positioning. Technically, we formulate this scaffold construction as a single selection problem that *simultaneously* solves for the ordering and geometric changes of the primitives. We generate different tutorials on man-made objects using our method and evaluate how easily the tutorials can be followed with a user study.

1 Introduction

The ability to draw real-world objects is a useful and important skill across many disciplines. Product designers draw daily as they generate and refine product ideas, fine artists may spend hours in figure drawing classes learning how to replicate a shape from the real world, while hobbyists use sketches for visual expression. Still, sketching requires skill and practice. One of the major challenges in drawing real-world objects is learning to draw *what you see* rather than *what you know* [Edwards 1999]. A simple cylinder, for example, is *known* to have a circular cross-section with equal widths at the top and bottom. However, when we actually *see* a cylinder, it is subject to perspective distortion: circles become ellipses while projected radii diminish with distance from the viewer.

Tutorials are commonly employed to teach novices how to draw a specific object using correct drawing practices. Manual authoring such tutorials requires significant expertise and time commitment even for trained artists. Consequently, objects and viewpoints in existing tutorials tend to be limited and are chosen by the expert, rather than the users of the tutorials. To address these issues, we present an approach for *automatically generating easy-to-follow tutorials for drawing part-segmented 3D models from user specified viewpoints.* Figure 1 shows parts of a tutorial generated by our system and the drawing by one of our study participants based on that tutorial. Our algorithm targets man-made objects where part relations and proportions tend to be inherently meaningful and crucial for accurate depiction.

Inspired by instructional books and online tutorials, we take explicit steps to make a sketching tutorial *easy-to-follow*: (i) focus on accurate inter-part proportions and *relations* via a drawing scaffold, followed by detailing of the object contour; (ii) proceed in a *coarse-to-fine* fashion, where object parts are abstracted as primitives (e.g., cuboids, cylinders) over several levels of detail to build up said scaffold; (iii) propose a particular drawing *order* among the scaffolding primitives such that those sketched later can be easily anchored (i.e., drawn with guidance) off already drawn primitives; and (iv) provide explicit steps for the construction of *guidelines* to accurately anchor the scaffolding primitives.

Our key observation is that in easy-to-follow tutorials the dimensions and arrangements of object-parts tend to have ratios that are easy to construct. For example, it is easier to construct the center line of a rectangular face compared to its one-fifth line. Tutorial authors choose to construct with such 'easy ratios' to simplify the drawing process and to focus on the procedure, rather than incidental and arbitrary measurements (see Figure 2). To apply this technique to existing objects, How2Sketch proposes small geometric changes while keeping overall deviations from the source model minimal. Since in each step new primitives and guidelines are anchored with respect to those drawn in the previous steps, the ordering of steps significantly affects the simplicity of ratios that can be employed, and the incurred geometric approximations. This tight interdependence between ordering of primitives and their geometric changes makes the problem non-trivial. A further challenge is to preserve the original inter-part relationships of objects, even under geometric perturbations. For example, in Figure 1 the coaxial relationship between the mixer bowl and mixer blade is preserved.

Technically, we map the geometric adjustment and ordering of parts to a single selection problem. We first generate a set of potential candidate primitives by enumerating different anchoring possibilities. Since such anchoring requires drawing guidelines, and some guidelines are easier to construct than others, the algorithm prefers anchoring possibilities that rely on easy-to-construct guidelines, such as the top edge, bottom edge, center line, etc., of existing primitives. Our key insight is that the problem of geometric adjustment and ordering of parts can be simultaneously solved by *selecting* an appropriate subset from the candidate primitives, in order to balance between geometric changes and ease of constructing necessary guidelines.

We test our algorithm on a range of examples and evaluate our algorithmically generated easy-to-follow tutorials with a user study, which finds that H2S tutorials can help both with objective as well as perceived accuracy of sketches, and are easier to follow.

2 Related Work

Assisted drawing. Various applications have been proposed to assist a user in sketching: Correcting user input based on geometric analysis of the users input strokes [Igarashi and Hughes 2001; Bae et al. 2008; Schmidt et al. 2009b]; relying on an underlying image to guide the user [Iarussi et al. 2013; Xie et al. 2014; Benedetti et al. 2014; Fernando et al. 2014]; or using crowdsourced data (e.g., many sketches) to improve the users drawing [Dixon et al. 2010; Lee et al. 2011; Gingold et al. 2012; Limpaecher et al. 2013; Simo-Serra et al. 2016] at a local stroke level.

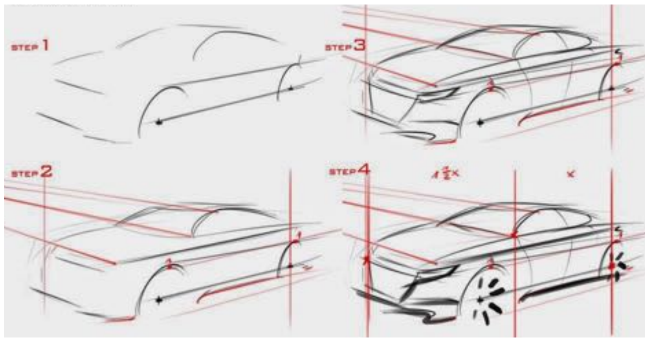

Figure 2: *A step-by-step sketching tutorial for drawing a car ©Czajkowski. The task is made simpler by breaking it into steps and by providing guidance about part proportions and alignments.*

Our focus is on suggesting a meaningful drawing order and easy-to-construct guides for accurate depiction of perspective and proportions. Stroke correction or beautification is orthogonal to our main contribution and may be used to complement the contour drawing phase of our tutorials. Other assisted sketching systems take as input 2D sketches and interpret them as 3D curve networks [Xu et al. 2014]. More advanced methods [Shao et al. 2012; Iarussi et al. 2015; Pan et al. 2015] use 2D input to infer 3D geometry or surface normals for complex shading. We focus on the automatic generation of sketching tutorials, rather than automatic inference based on the sketched curves.

Tutorials. A good tutorial greatly facilitates understanding. Many attempts have been made to automatically generate high-quality tutorials for different applications. A digital drawing tutorial system was proposed by Fernquist et al. [2011] that allows an expert to create tutorials for novices. Tutorial generation systems [Takagi et al. 2003; Cummmings et al. 2012] for specific sketching tasks have also been proposed, for example drawing a single scene with pre-defined objects, or 'eyes.' Grabler et al. [2009] developed a tutorial system for photo manipulation tasks. In contrast, we focus on generating for sketching 3D models of man-made objects.

Drawing expertise. Tchalenko [2007] found that novices and professional artists have comparable accuracy when performing basic line drawing tasks (straight lines and simple curves). However, in a follow-up study [Tchalenko 2009], he showed that when copying complex artworks, novices made significantly more errors than artists. The main difference in drawing strategy was that experts divided complex lines into easy-to-draw short segments. Schmidt et al. [2009a] found that experts made qualitatively similar errors to non-artists, indicating that perspective drawing is hard, even for trained users. Particularly for off-axis viewing angles, drawing error increased significantly. In an observational study, Grimm [2011] found that artists commonly used a coarse-to-fine strategy starting with blocking shapes and finishing by drawing detailed items at the end. How2Sketch assists the user by breaking the drawing process up into basic steps that are easy to execute and by explicitly indicating vanishing line directions.

Line drawings. Many methods for generating stylized artistic renderings of objects have been proposed (see [Kyprianidis et al. 2013] for a survey). We leverage stylization to visually distinguish the various line types of our tutorials (perspective lines, guides, contours, etc). Other researchers investigated which features artists typically draw to convey 3D shape [DeCarlo et al. 2003; DeCarlo et al. 2004; Burns et al. 2005; DeCarlo and Rusinkiewicz 2007]. Fu et al. [2011] and Liu et al. [2014] infer plausible contour ordering from 2D and 3D inputs, respectively. While the derived sequences are plausible, they are not tailored for tutorials and do not provide specific guidelines to make them easy to follow.

3 Learning How to Sketch

To inform the design of How2Sketch we studied several drawing books [Edwards 1999; Eissen and Steur 2007; Eissen and Steur 2011], consulted various sketching websites (e.g., Sketch-A-Day[2016], Draw-A-Box [2016]), carried out an expert interview with a professional artist, and participated in a drawing course.

Through this process we found that effective tutorials for drawing 3D objects typically include the following:

- *Parts are approximated by geometric primitives:* Plane, cubes and cylinders are heavily used to approximate shapes. They are easy to construct and verify visually.
- *Steps are coarse-to-fine:* First, the overall object is scaffolded

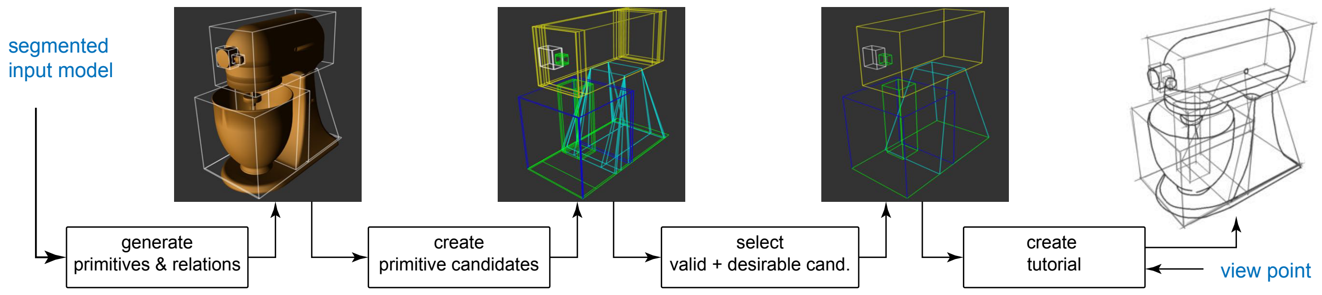
segmented input model
generate primitives & relations
create primitive candidates
select valid + desirable cand.
create tutorial
view point

Figure 3: *System Overview. Starting from a segmented input model and a user-specified viewpoint, How2Sketch generates easy-to-follow step-by-step tutorials. The system automatically makes subtle geometric modifications to simplify the tutorial steps.*

with approximate shapes, followed by finer contour details. Primitives are drawn sequentially, in optimized order.

- *Anchor shapes to each other:* Shapes are drawn with respect to previously drawn shapes, to aid with correct placement and proportions. Instructions for positing shapes relative to each other use simple measurements (e.g., draw box *half way* down the side, draw circle in the *center* of the rectangle), etc.
- *Vanishing lines for perspective:* Vanishing points are explicitly indicated to aid the user to draw correctly.

How2Sketch supports the above tutorial features as follows:

(a) Scaffolding primitives. How2Sketch utilizes scaffolding primitives to geometrically approximate each segmented object part. The system supports planes, cuboids, cylinders, and truncated pyramids, as they allow for planar guidelines to be used, which are simple to construct, and cover a wide range of shapes. (Note that in our visualization, cylinders are shown as axis-aligned bounding boxes, since the box faces are used for providing guidance for drawing ellipses for cylinder caps.) In addition to scaffolding, we guide users in drawing ellipses to better approximate some shapes.

(b) Ordering. Our algorithm provides the relative ordering of the scaffolding primitives. Further, How2Sketch offers detailed, sequenced instructions for constructing primitives.

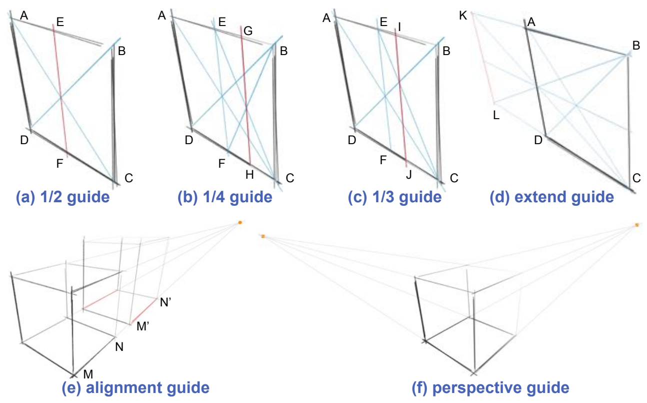

Figure 4: *Our system supports different forms of guidelines for drawing coplanar proportions (a-d), for anchoring alignments (e), and for previewing 2-point perspectives (f). Please refer to Sec. 3.*

(c) Placement, alignment, and proportions. We support a set of coplanar guidelines (see Figure 4). Given a face $ABCD$, its diagonals help construct the ½ line EF (Figure 4a). Two levels of ½ lines produce a ¼ line GH (Figure 4b); while intersecting a diagonal BD with line CE produces a ⅓ line IJ (Figure 4c). Similarly, we support extrusion towards a vanishing point as in Figure 4d where $ABCD$ is extended by reflection to form $BCLK$ such that $AB = AK$. Finally, we also support alignment, as in Figure 4e, $M'N'$ is aligned with MN.

(d) Perspective. To provide perspective information, we show the vanishing points (if within the drawing area) and also show the vanishing lines leading to them (Figure 4f). How2Sketch supports sketching in 2-point and 3-point perspective.

4 Generating Sketch Sequences

Given a 3D object (S) segmented into parts and a desired viewpoint, our goal is to establish an easy-to-follow sequence for drawing the object, starting with the scaffolding and progressing to the contour details. We make it easier to draw the scaffold by actively making small part-level geometric changes to facilitate relative anchoring using a set of guidelines.

As described in Section 3, we have adopted simple procedures to accurately draw guidelines at easy-to-construct ratios (½, ⅓, ¼, $1\times$, $2\times$, etc). Object part positions and sizes in the original models, however, rarely conform to such ratios. Hence, we propose to modify object parts, so that they end up with part relationships that are easy to draw. We motivate this choice twofold: (i) Scaffolding primitives in tutorials like those generated by H2S are already approximations of real geometry and thus contain a measure of error. Some of this error can actually be compensated by adjusting the fit of contours within the scaffold. (ii) Accurate estimation of lengths and ratios is difficult, even for experts, so errors are almost unavoidable. By enforcing that parts relate via simple ratios for which reasonable geometric constructs can guide the user H2S can minimize per-part error and make better global decisions about how to distribute the overall error.

Our algorithm proceeds in three main stages (see Figure 3): (i) generating part-level primitives and encoding inter-primitive relations; (ii) creating primitive candidates based on various inter-primitive anchorings strategies; and (iii) selecting a valid and desirable set of

Table 1: *Notation table.*

symbol	denotes
S	input part-segmented model
P_i	primitive corresponding to the i-th part of S
$R_{i,j}$	relation between primitive pairs (P_i, P_j)
C_j	Parent candidate that can be used for anchoring
$C^k_{j\to i}$	candidate for the i-th part primitive with (anchoring) parent from the j-th part primitive, where k denotes the k-th such instance
$\mathcal{C}_{*i}$	set of all the candidate primitives generated for part primitive P_i
$\chi(X)$	indicator variable corresponding to the selection of X
Λ	assignment of indicator variables denoting a set of selected candidates

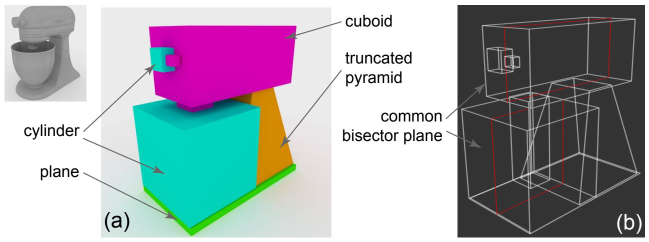

Figure 5: *Given a part-segmented input model S (top-left inset), the system abstracts the parts as different primitives (a) and identifies inter-part relations. For example, here the mixer bowl and mixer head primitives share a common bisector plane.*

primitives among the candidate selections. The result implicitly encodes how to geometrically modify each part (both their dimension and placement), and in which order to draw them. Intuitively, our algorithm produces an easy-to-follow primitive drawing sequence at the cost of deviating from the original geometry in a controlled fashion. We now elaborate each step. Please refer to Table 1 for symbols used in the following.

4.1 Generating Primitives and Inter-part Relations

H2S takes as input a pre-segmented 3D model and abstracts the model parts with primitive shapes. In our implementation we support planes, cuboids, cylinders, and truncated pyramids (see Figure 5a). For each part of the input model S, we use least-squares to fit different axis-aligned primitive types and take the one with the smallest residual. In case of ties, we prefer the simpler primitive. We denote the primitive for the i-th part as P_i (the type of primitive is not explicitly indicated in this notation).

Man-made objects often have dominant inter-part relations. We found it desirable to preserve such relations in the generated tutorials. Hence, we first detect such inter-part relations and later preserve them in the generated tutorials. We simply test (see [Mehra et al. 2009]) each pair of primitives P_i and P_j for any (supported) relations. In our implementation, we handle coplanar, coaxial, and common bisector plane relations. In case of multiple relations between a pair of primitives, we prefer common bisector plane over coaxial over coplanar. We represent a relation using a binary variable $R_{i,j}$ where i and j respectively denote the primitives P_i and P_j (type of relation is not explicitly indicated in this notation). If a relation is present, we mark $R_{i,j} = 1$ and $R_{i,j} = 0$ otherwise. Figure 5 shows some examples.

4.2 Creating Candidate Primitives

We now describe the candidate primitive generation step that creates additional primitives based on possible anchoring strategies. We use $\mathcal{C}_{*i}$ to denote the set of all the candidate primitives generated corresponding to the primitive part P_i. Since the original primitives (e.g. P_1, P_2 and P_3 in Figure 6) are always candidates, we start with $\mathcal{C}_{*i} := \{P_i\}$. We generate candidate primitives in three stages:

(i) For each pair of primitives P_i and P_j, we generate candidates of the form $C^k_{j\to i}$, where $j \to i$ indicates that a candidate is generated for primitive part P_i and is anchored off P_j with k denoting different anchoring possibilities. For example, parts can be anchored based on different guidelines described in Section 3 for different face- or plane-based anchors. We append these candidates to the respective candidate sets as: $\mathcal{C}_{*i} \leftarrow \mathcal{C}_{*i} \cup \{C^1_{j\to i}, C^2_{j\to i}, \dots\}$ (see Figure 6b).

(ii) The small part modifications introduced during anchoring in

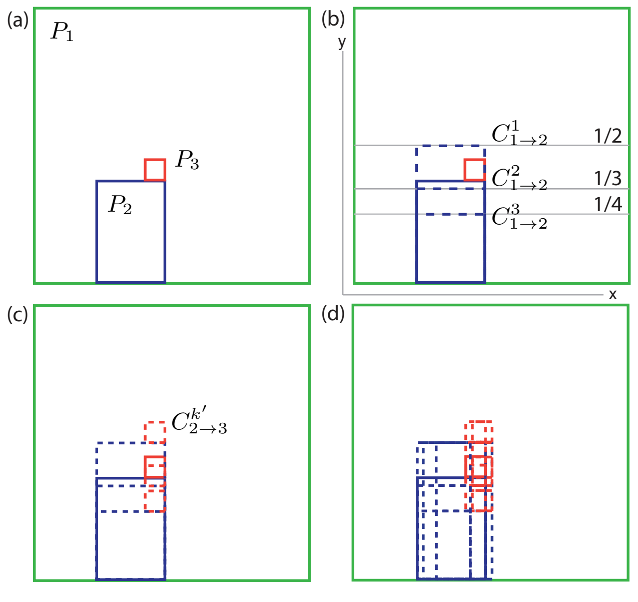

Figure 6: *(a) Starting from initial primitives P_1, P_2, P_3, for each pair of primitives we generate several adjusted primitive candidates. Candidates are generated for each axis independently, for example, in (b) we show how using P_1 as a parent several P_2 candidates are created by aligning its top edge to the 1/2 ($C^1_{1\to2}$), 1/3 ($C^2_{1\to2}$) and 1/4 ($C^3_{1\to2}$) guides on the y-axis. (c) As P_2 and P_3 have a co-planar relation for each $C^k_{1\to2}$ candidate a new P_3 candidate ($C^{k'}_{2\to3}$) is generated restoring this relation. This process is repeated both in the other dimensions and to generate second-level candidates (see Section 4.2) to generate the full set of candidates (d). This is an illustrative figure in 2D with only some of candidate primitives shown.*

step (i) may violate some of the relations $R_{i,j}$. For each pair of primitives (P_i, P_j) sharing a relation, we add additional primitives to their candidate sets to restore the relations. Specifically, corresponding to a candidate of the form $C^k_{j\to i}$ (created in stage (i)), we create a new candidate of the form $C^{k'}_{i\to j}$ such that $C^k_{j\to i} \leftrightarrow C^{k'}_{i\to j}$ are similarly related as in $P_i \leftrightarrow P_j$. We append all such relation-based additional candidate primitives to the respective candidate sets, i.e., $\mathcal{C}_{*i} \leftarrow \mathcal{C}_{*i} \cup C^{k'}_{i\to j}$ (see Figure 6c).

Note that in the above a candidate is allowed to be anchored from one or multiple parents, as each axis can be independently anchored. Additionally a candidate can be partially unguided (e.g., the width and length of cuboid is guided but the height is not) or completely unguided (e.g., it is simply the input primitive) (see Figure 6). We defer further details to the implementation section.

(iii) We allow second-level candidates, i.e., candidate primitives as generated above are allowed to act as anchors for other primitives creating a hierarchy. To this end, we simply iterate one more time stage (i) and (ii) (e.g., in Figure 6d candidates $C^k_{1\to2}$ create second-level candidates for P_3). Note that before starting this step, we remove the undesirable candidate primitives with large changes in geometry or relative placements (more details in the implementation section).

At the end of this stage, we have a set of candidates for each part P_i of the input model resulting in the super set of candidate primitives of the form $\{\mathcal{C}_{*i}\}$ (see Figure 7).

4.3 Selecting Candidate Primitives

Having generated multiple candidates, our remaining task is to select a set of valid and optimal candidates, as explained next.

Valid candidate sets. We first characterize the notion of valid selections. We use indicator variables $\chi(X)$ to denote if a candidate primitive X is selected (i.e., $\chi(X) = 1$) or not (i.e., $\chi(X) = 0$). We have $\chi(C^k_{j\to i}) \in \{0, 1\}$ for each $C^k_{j\to i} \in \mathcal{C}_{*i}$. Let Λ denote a particular assignment for the indicator variables for *all* the candidate primitives.

Among the various possible selections, not all the subsets of candidates of the form Λ constitute *valid* selections. A valid selection of candidates should satisfy three conditions:
(1) for each part of S, exactly *one* candidate primitive should be selected;
(2) if a selected candidate primitive is anchored off one or more parent (candidate) primitives, then its parent primitive(s) *must* also be selected;
(3) if any two primitives P_i and P_j share a relation, then their corresponding selected candidate primitives should also respect the same relation.

We now express the above conditions in terms of the indicator variables in Λ.
(a) We encode (1) as

$$\sum_{j,k} \chi(C^k_{j\to i}) = 1 \qquad \forall i. \tag{1}$$

(b) We encode (2) as a quadratic constraint involving the binary selection variables as

$$\chi(C^k_{j\to i})\chi(C_j) - \chi(C^k_{j\to i}) = 0 \tag{2}$$

for each dependent pair $C^k_{j\to i} \in \mathcal{C}_{*i}$ and its parent C_j. Note that this condition *disallows* $\chi(C^k_{j\to i}) = 1$ AND $\chi(C_j) = 0$, but allows any of the other three assignments involving $\chi(C^k_{j\to i})$ and $\chi(C_j)$.
(c) We now encode (3). Let two primitives P_i and P_j share a relation, i.e., $R_{i,j} = 1$. Let $\mathcal{C}_{*i} = \{C^1_{*i}, C^2_{*i}, \dots\}$ be all the generated candidates for primitive P_i and similarly $\mathcal{C}_{*j} = \{C^1_{*j}, C^2_{*j}, \dots\}$ for primitive P_j. Then for *each* pair of the form $C^k_{*i} \in \mathcal{C}_{*i}$ and $C^{k'}_{*j} \in \mathcal{C}_{*j}$ that does *not* share the relation $R_{i,j}$, we require

$$\chi(C^k_{*i})\chi(C^{k'}_{*j}) = 0. \tag{3}$$

This condition disallows $\chi(C^k_{*i}) = 1$ AND $\chi(C^{k'}_{*j}) = 1$, i.e., candidate primitives that do not share the same relation as their primitive parts cannot be jointly selected.

Thus, a selection Λ is valid if and only if Equations 1-3 are all satisfied. Among all such valid selection sets, we next determine which one is the most desirable. Figure 7 shows a set of candidate primitives and a valid selection. Note that as each candidate primitive has a unique id and anchoring hierarchy, the constraints prevent dependency loops from being created.

Sequencing sketching as a selection problem. We balance the error due to making changes to the geometry with the difficulty of drawing arising from anchoring. Specifically, we consider unanchored parts to be most difficult to sketch. Further, among the anchored ones, we consider a primitive easier to draw if it requires fewer guides. We model this difficulty of drawing as the cost

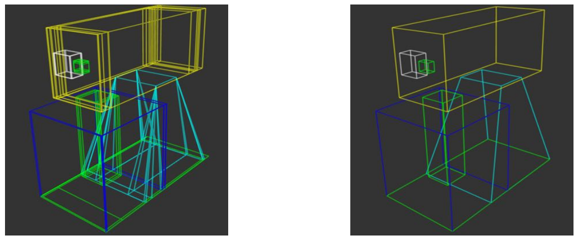

Figure 7: *From a set of candidate primitives (left), our algorithm selects a subset of primitives that is* valid *and* desirable *as shown on the right. The selection implicitly encodes in which order to draw the primitives and also how to change each primitive (size and/or placement) such that the resulting tutorial is easy to construct. Please refer to the text for details.*

$E_e(C^k_{j\to i})$ with a lower cost denoting *easier to draw* (see *"Error functions"*, below). The total cost is expressed as:

$$E_{\text{difficulty}}(\Lambda) := \sum_{i,j,k} \chi(C^k_{j\to i}) E_e(C^k_{j\to i}). \tag{4}$$

Selecting any primitive, however, incurs an associated error that we indicate as $E_d(C^k_{j\to i})$ due to deviation from original geometry. So, the total data cost of selecting a set of primitives is:

$$E_{\text{adjust}}(\Lambda) := \sum_{i,j,k} \chi(C^k_{j\to i}) E_d(C^k_{j\to i}) \tag{5}$$

with a higher cost indicating larger geometric deviations from the original parts.

Thus, we arrive at the final formulation for *desirable* selection as,

$$\Lambda^\star := \arg\min_{\Lambda} (E_{\text{adjust}}(\Lambda) + E_{\text{difficulty}}(\Lambda)) \tag{6}$$

subject to Equations (1)-(3) to ensure a valid selection. Thus, we have formulated our problem as a quadratically constrained linear program.

Error functions. The above formulation requires metrics for E_e and E_d. We use the following metrics in our implementation.

For the difficulty of drawing term $E_e(C^k_{j\to i})$, we associate a higher cost for anchors that are harder to replicate (e.g., requiring more construction lines). Specifically, we set the cost to the number of guidelines divided by the area of the parent plane where construction lines are to be drawn. This encourages fewer guides but also using planes/faces with larger areas for drawing sketch guides. (The effect of viewpoint is only considered at runtime as discussed in Section 5).

For the data error $E_d(C^k_{j\to i})$, we sum the changes in length along each axis, normalized by the original axis length, with the translation of the midpoint of each axis, again normalized by the input axis length. For an unguided axis we set the data error to the maximum of 2 to discourage unguided candidates.

Final drawing order. The solution to the above optimization directly gives us both the *ordering* and the *size and location modifications* of the parts. The ordering is represented as a directed graph, and we gain the final linear ordering via topological sorting. Note that the directed graph may have a fork in the ordering of candidates primitives. This implies that the relative drawing order of certain primitives are not specified. We break such ties only at runtime once the user selects a view (see Section 5).

4.4 Implementation details

We now clarify some implementation details. The 3D models were downloaded (e.g., from Turbosquid) and manually part segmented (if part level segmentation was missing). Segments that are not well approximated by one of the primitives described above can be represented as a custom primitive (e.g., line) but such primitives are excluded from our optimization step. Instead, their positions are updated after optimization by enforcing existing relational constraints with optimized primitives.

The candidate primitive generation works in two steps: first, we use the coplanar relations to generate candidate planes $c^k_{i\to j}$, and then depending on the primitive type we combine the planes to create a complete primitive $C^k_{i\to j}$ (here, lowercase c indicates a candidate plane rather than a complete primitive, C). This choice unifies candidate primitive generation across primitive types (recall cylinders are processed based on their axis-aligned bounding box).

For each pairwise coplanar relation $R_{i,j}$ we have two participating planes in P_i and P_j: at this stage the relation is undirected and we produce candidate planes using both combinations $c^k_{j\to i}$ and $c^k_{i\to j}$. To generate a candidate plane, each axis is considered independently then all combinations of axis pairs are used to create planes $c^k_{i\to j}$. An axis can be anchored by the parent plane using the end points of the same axis. This means there are several anchoring possibilities. For example, anchoring the vertical axis of P_i on P_j might involve anchoring the top edge of P_i to the ⅓ line of P_j and the bottom edge P_i to the bottom edge of P_j. An alternative might be to anchor the top edge of P_i to the ⅓ line of P_j and the bottom edge P_i to ¼ line of P_j. We initially generate all such candidates but to reduce the number of candidates to select from we discard those where an axis length or translation change by more than 10% of the input length.

Having generated all the candidate planes using all the pairwise relations, we generate complete primitives by combining the different planes based on the primitive type. To generate a complete cuboid primitive, for example, we find the missing height axis from one of the other planes to complete the primitive. For truncated pyramids we combine top and bottom planes with a height axis to make a truncated pyramid. Finally, we repeat this process but use the first level candidates as the parent primitives to generate second level candidates.

We use the Gurobi Solver [Gurobi Optimization 2015] to solve the quadratically constrained LP as described above. Typically the solver takes 1-2 minutes in the presented examples.

5 Presenting Sketch Sequences

The sequence generated in Section 4 provides primitive ordering, sketching guidelines, and adjusted part geometry for drawing the scaffolding of the object. H2S tutorials can be adapted further

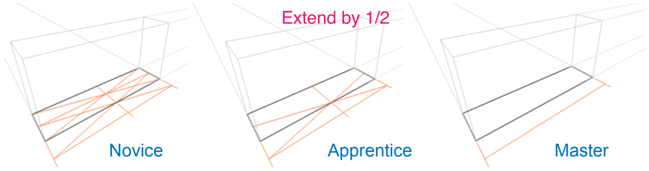

Figure 8: *User ability. The user specify a preferred drawing level (novice, apprentice, or master) which determines the number of intermediary guides presented for each step. For the 'extend by 1/2' step, novices (left) are shown 9 guidelines, apprentices (center) 6 guidelines, and masters (right) 3 guidelines.*

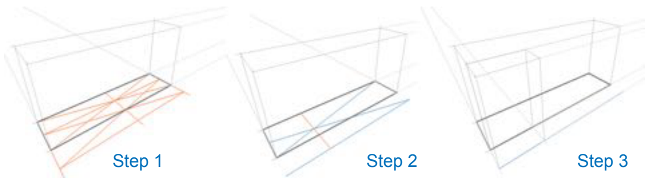

Figure 9: *Guide lifetime. Guides first appear in orange (left). In subsequent steps guides that are no longer required are removed, while those that are to be reused are marked in blue (middle, right).*

based on the user chosen viewpoint and user indicated drawing level (novice/apprentice/master), which can be controlled interactively. Our custom viewer indicates when guidelines can be erased and provides hints for drawing in perspective and object contours.

Viewpoint. We use the user specified viewpoint to customize the tutorial as follows: (i) Although primitive ordering is determined based on anchoring strategies, multiple primitives can anchor from the same parent, resulting in a tie. We break such ties by first choosing the primitive that is closest to the user from the indicated viewing position. (ii) The selected viewpoint can make some guidelines cumbersome to draw because of limited space on the projected area of a primitive face. We identify such instances by thresholding based on A_p/k, where A_p indicates the projected area and k the number of guides necessary to draw the primitive. If a primitive falls below a threshold of 0.01, we ask the user to simply 'eyeball' the primitive without drawing intermediate guides. (iii) Finally, a segment that is occluded and its primitive does not help anchor any other visible primitive is deemed unnecessary and hence is left out from the generated tutorial.

User ability. We adapt our tutorials to different sketching abilities by classifying the various guidelines as suitable for novice, apprentice, or master users. For example, dividing a face of a primitive into halves requires three guidelines. A novice is shown all the three, an apprentice only the ½ line itself, and a master is not provided with any intermediate guidance. Note that in all cases, the user is instructed to divide the highlighted face into half by a text label in the viewer (see Figure 8).

Guide lifetime. In order to reduce the amount of guidelines on a sketch at any point in time, we determine each guide's lifetime to inform the users when a guide can be safely erased. To this end, we first go over the list of generated guidelines to identify the equivalent ones, and store their *lifetime*, i.e., when they first appear and when they are last used. During the tutorial, a guideline is drawn in orange when it first appears. If the guideline is used in any later step, it is changed to blue. After the last step a guide is used, it is no longer shown. As a result, users do not have to unnecessarily erase/redraw guides, which helps to reduce clutter as they sketch (see Figure 9).

Vanishing points and ellipses. Vanishing lines and vanishing points are indicated with respect to the paper boundary (shown as green corners) to help users better position the lines. We additionally guide users in sketching ellipses on a primitive face by using guides to the vanishing points. These guides intersect with the edges of the face at the perspective mid-points, which are the points where the ellipse should touch the face of the primitive.

Contour ordering. Once the user has sketched the scaffolding and ellipses, we guide them to sketch the contours. We progressively display contours on the modified underlying model segments, following the order determined by the primitives. Already drawn parts are used to determine occlusion for the new primitives, thus reducing clutter (see Figure 1).

Interface. H2S tutorials can be presented in a few different forms.

They can be navigated using an interactive interface, they can be printed (see supplementary material), or sequenced into a tutorial video. Text instructions can be synthesized, as needed.

6 Results and Discussion

We used How2Sketch to generate sketching tutorials for four man-made objects - a Digital SLR Camera, Kitchen Mixer, Train, and Paint Roller. For these models, numerous tutorials depending on viewpoint and user ability can be generated. Parts of the tutorials are shown in Figure 10 (see supplementary material for full sequences). We encourage the readers to compare How2Sketch tutorials with existing online tutorials (e.g., Draw-A-Box [2016]). Each tutorial takes between 15 and 45 minutes (across the users who used the system) to complete due to their varying complexity. The small changes made to the input geometry by the method are illustrated in Figure 11. As desired, the alterations to geometry are subtle but now enable simple anchoring strategies based on the altered segment bounding boxes (also shown).

As demonstrated in Figure 10, our tutorials follow a coarse-to-fine strategy, starting with a single primitive that can be used to anchor subsequent primitives. Figure 10a shows excerpts from a tutorial sequence with master-user ability. Here, the grip of the camera is anchored on the edges of the camera body and a ¼ guide. Additionally, the grip and flash are both extended by one half the depth of the main body. The lens, an example of a second level anchoring, uses the flash for anchoring by extruding $1\times$. Guides for ellipses are provided before contours are drawn.

In the paint roller tutorial in Figure 10b the handle anchors the roller using the common bisector plane. The top edge of the roller is $1\times$ the length of the handle. The bottom edge is $\frac{1}{2}\times$ the length of the handle but due to the limited projected area and number of guides otherwise required, the step is unguided (as per Section 5).

Figures 1 and 10c both show novice-level tutorials for the food mixer but from different viewpoints. The plane primitive for the base of the mixer anchors the bowl using a planar relation and ½ guide. The common bisector plane between the base and the main body of the mixer is used for anchoring the length of the main body. The bisector plane is first drawn before being extended in both directions to create the cuboid primitive. The Mixer's stand is an example of a primitive with two parents, being anchored off both the main body and base. Difference in viewpoint between the two tutorials means that as one of the knobs is occluded from the view chosen in Figure 10c, it is omitted from the tutorial (see Section 5).

The train example, Figure 10d, anchors the second carriage as $1\times$ the length of the first carriage and the top edge of the wheels using the ¼ guide on the vertical axis of the first carriage. The driver's compartment is unguided.

Limitations. H2S only makes small changes to the input geometry. However, small gaps between object parts can have important semantic meaning. An example of this can be seen in Figure 11 where the main body of the mixer and the stand separate slightly in the adjusted version. We know these two segments would be joined by a hinge making such an adjustment unrealistic. Symmetry or regular structure can similarly be lost from the small geometry changes. An example of this is the roller in Figure 11, which ceases to be a perfect cylinder. Note that most of these violations are difficult to spot unaided and tend to get masked by drawing inaccuracies. Finally we find relations from the input segments but do not allow adjustments in geometry to create a relation that was not already present. In the future, we might enable such changes to allow for an even wider range of candidates.

7 Evaluation

To evaluate the effectiveness of the H2S tutorials, we compare to a simple step-by-step tutorial that shows scaffolding primitives for each part of the object but does *not* simplify the sizes or locations of the primitives to make them easier to draw. In this Basic tutorial type, the scaffolding primitives are shown in order from largest to smallest with a base primitive anchored to the ground plane. No guidelines are shown. Please see the supplemental materials for the complete tutorials used in the study.

Participants. We recruited 10 participants (ages 18-55, 6 men, 4 women) with varied expertise in drawing. Two participants reported negligible drawing experience, four reported drawing once in a while, and four reported drawing at least once a month. Three had taken college-level art classes or private/non-accredited art classes. When asked (free-form) what they found most challenging about drawing, 4 mentioned perspective, proportions, scale, and relative positions. When asked to rate their drawing skills on a scale of 1 (poor) to 5 (great), only 4 people rated their drawing skills above 2.

Methodology. In advance, each participant filled out an introductory questionnaire about their experience with drawing. Upon arrival, each participant was told that they will be asked to draw two objects, a camera and a mixer, using two different tutorials. Participants always followed a H2S tutorial first to disadvantage H2S to any learning effect. The two objects (camera and mixer) counterbalanced with half of the participants using the H2S tutorial type for the camera and half using the H2S tutorial for the mixer. The H2S tutorial was set to the *novice* ability for all the participants.

Before the H2S tutorial, participants were given a written handout (see supplemental material) that described how to draw construction lines for ½, ¼, and ⅓ guidelines and extending planes (see Figure 4). This written tutorial was designed to give them context for what they would encounter in the H2S condition.

Both the Basic and H2S tutorials were followed using a 13" laptop; participants used the trackpad to advance forward and backward through the tutorial. All drawings were done on paper. Each participant was given two pencils (HB, 0.3mm and 0.7mm). They were allowed to use a provided straight-edge and eraser. For creating each drawing, the participants were given a sheet of paper that included the vanishing points and the ground plane of the first primitive. This initial anchoring allowed us to easily compare drawings across users. All users drew the scaffolding primitives first on the calibrated paper. For drawing the final contours of the object, the moderator attached a transparent sheet to the paper with the scaffolding. This allowed for easier digitization and separately compare the contour drawings and the scaffolding primitives across users.

Participant filled out a questionnaire after drawing each object, indicating their level of satisfaction with their drawing (1 - not at all, 5 - very much), perceived accuracy of their drawing (1 - not at all accurate, 5 - very accurate), enjoyment with the tutorial experience (1 - not at all, 5 - very much), and ease of following the tutorial steps (1 - not at all easy, 5 - very easy). They also gave free-form responses about what they liked about each tutorial type and how it could be improved. At the end of the study, subjects were asked which tutorial type they preferred (Basic or How2Sketch). We referred to the Basic tutorial type as the tutorial *without guides* and the H2S tutorial type as the *tutorial with guides*. Each participant was given a $25 gift card for his/her time.

User feedback. Nine out of ten participants preferred the H2S tutorial over the Basic tutorial. For each of the four questions, the users preferred H2S over the Basic tutorial (see Figure 12).

An ANOVA across tutorial type and object drawn reveals a sig-

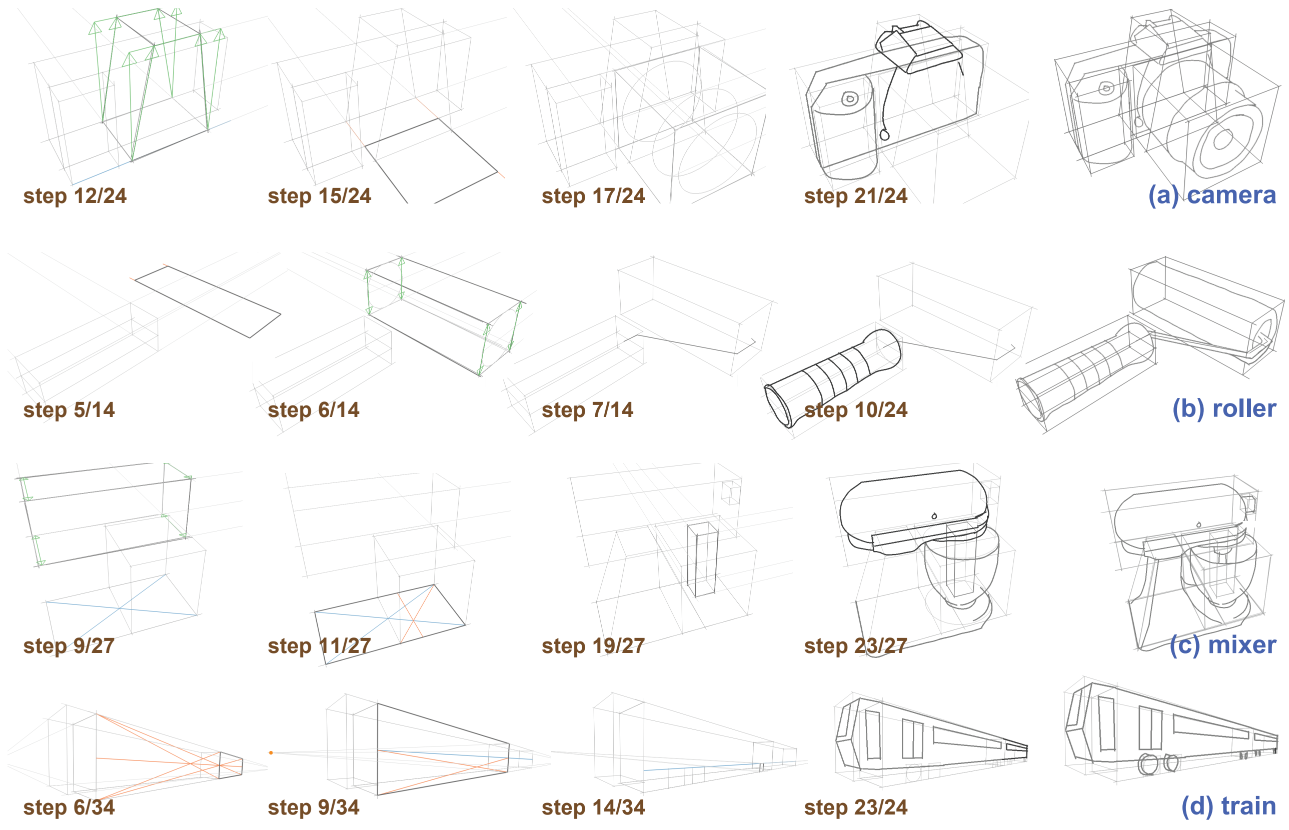

Figure 10: *Example step-by-step tutorials generated by our system: (a) and (b) were generated in the master-user setting, while (c) and (d) were generated in the novice-user setting. Please refer to the supplementary materials for complete examples.*

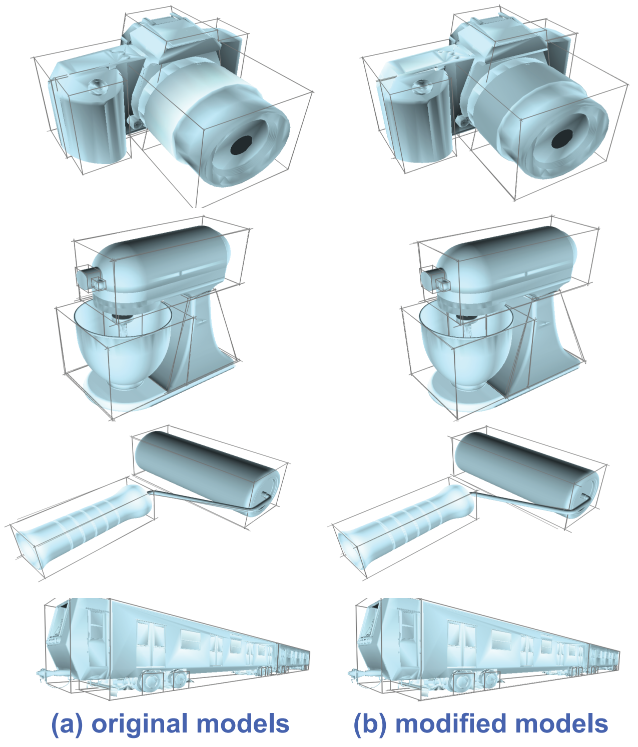

Figure 11: *(Left) Original models. (Right) Subtle changes proposed by our algorithm in order to make the objects easier to draw.*

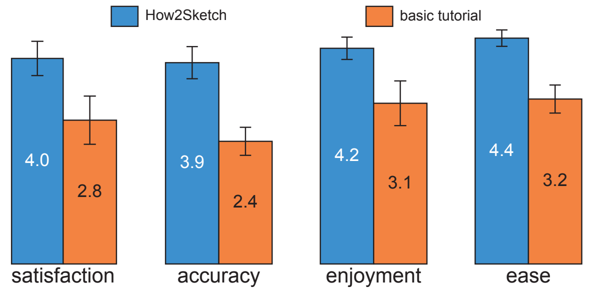

Figure 12: *Average user ratings for satisfaction, perceived accuracy, enjoyment, and ease of following were all higher for the How2Sketch tutorials than for the basic tutorials. Showing standard error of mean (SEM) bars for* $N = 10$.

nificant effect of tutorial type on accuracy and ease of following tutorial ($p<0.003$), significant effect on enjoyment ($p<0.034$), and marginal effect on satisfaction ($p<0.058$). The object drawn did not have an effect on any measure, despite their varying difficulty, and there was no interaction between tutorial type and object drawn.

Evaluating sketch quality. Figure 13 overlays the registered user sketches from the different conditions on the original model for the condition (e.g., H2S model after part level adjustments). While variation in contour placement is evident in both tutorial types, the variation in the basic tutorial sketches is greater.

In the camera tutorials the basic version starts with the ground plane for the lens and H2S with the ground plane of the main body. With this anchoring in the basic tutorial sketches, the width and length of the lens are accurate. However, the lens height and the other three primitives have a variety of errors in proportion and part placement.

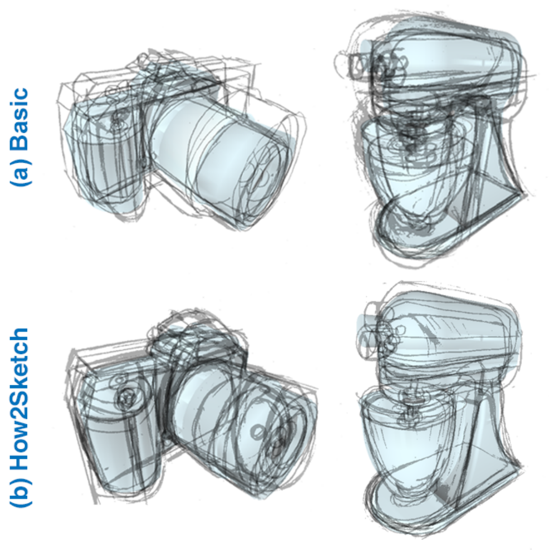

Figure 13: *User Study Sketches: All the user sketches overlaid on the target objects. Sketches from following basic tutorials (top) show much greater variation in proportions and alignment than sketches from following H2S tutorials (bottom).*

Comparing with H2S sketches, there are similar variations in the height of the main body. However, the guided steps for the grip and lens show reasonable consistency in positioning across the users. For the Mixer sketches - where both tutorials start with the base plane of the mixer - there is much more consistency of object part placements across users with the H2S.

As further validation, we conducted an additional user study using Amazon Mechanical Turk (AMT). In the study, we presented users with two sketches of the same object type overlaid and registered to their condition specific model (see supplementary material). The two pairs of object could be from the same or different tutorial types. We asked participants to "Please select the sketch that is more accurate to the underlying model. Do consider the proportions, alignment and perspective. Please ignore style or shading". The studies for the two objects were run independently, so did not have the same responders. Each participant evaluated the 45 unique image pairs for one of the objects. Each study had 220 sets of responses. AMT users were compensated \$0.01 per comparison.

The H2S Camera tutorial received 128/220 votes in pairwise comparisons against the basic tutorial. Similarly the H2S Mixer tutorial received 131/220 votes. Using the binomial test to evaluate these results, we can reject the null hypothesis with $p<0.018$ for the Camera and $p<0.006$ for the Mixer.

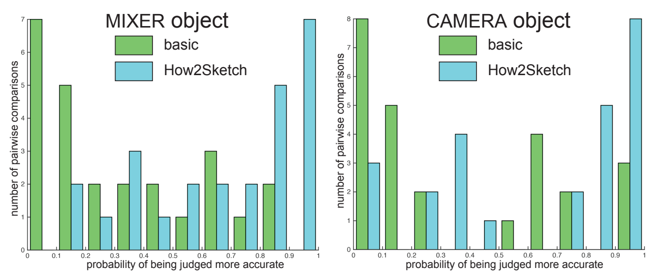

Figure 14: *Bradley-Terry Model for the Mixer and the Camera Sketches produced by users of our tutorials and evaluated by another user study with Amazon Mechanical Turk rankers.*

To evaluate the results further, we performed pairwise comparisons using the Bradley-Terry model. We plot probability histograms for both objects in Figure 14. The bottom axis is the probability of being judged more accurate in a pairwise comparison. The left axis is the number of pairwise comparisons that has this probability. Visually we observe H2S tutorials - in both object types - are more likely to be perceived as having higher accuracy.

Scaffold accuracy. To numerically evaluate accuracy we use the corners of the scaffolding primitives. We first rectify the scanned sketches into a normalised coordinate space using the corner registration marks on the paper. For both conditions, we manually marked the 32 corner landmarks and 40 corner landmarks of all scaffolding primitives in the Camera example and the Mixer example, respectively.

For each point, in each condition across users, we computed a mean position, standard deviation, and distance of the mean from ground-truth in 2D (Error) (see Figure 15). Standard deviations across conditions were similar, which we take to suggest comparable user drawing skills across conditions. For the Camera example, 2D drawing error was 71% higher in the Basic condition compared with H2S (highly significant with p<2E-10 using two-way ANOVA). For the Mixer example the error was only 3.5% higher in the basic tutorial and was not statistically significant.

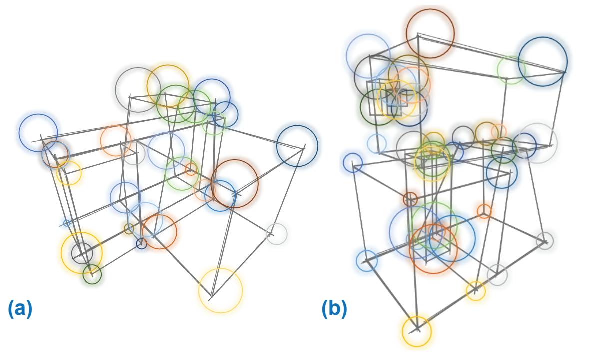

Figure 15: *How2Sketch Scaffold Accuracy: The mean location of the scaffold primitive corners plotted on the ground truth with the size of each bubble being the mean 2D error.*

We explain the variation across objects as follows: The predominant guidance in the Mixer H2S sequence is that many of the primitives share a common bisector plane. There is, however, no guidance for the height of the primitives. Thus it is unsurprising that the scaffold corners are not accurate compared to ground truth, as many primitives accuracy rely solely on getting the height of another primitive correct (see Figure 15b). We believe the guidance for the common bisector plane helps with the alignment and perspective of the sketch, hence the perceived accuracy improvement in the H2S condition from the AMT user study. In the Camera H2S sequence there are no primitives that are solely dependent on a parent primitive with an unguided axis, therefore the users are more accurate at drawing the scaffolding (see Figure 15a).

Limitations. Our user study has two potential sources of bias: (i) By having users follow H2S tutorials first we intended to disadvantage H2S against a learning effect, however, this could have potentially introduced a bias in our user feedback regarding enjoyment and satisfaction. (ii) By providing users with a tutorial on how to use construction lines it could imply that construction lines are good practice, therefore the absence of construction lines in the Basic tutorial could have biased the users. In future studies randomizing the ordering of tutorials may help avoid these issues.

8 Conclusion

We presented How2Sketch, a system that automatically generates *easy-to-follow* tutorials for drawing part-segmented man-made 3D objects from selected views. We evaluated our system using a user study, and found that sketches made by following H2S tutorials had more accurate proportions and relative part placements compared to a basic step-by-step tutorial with scaffolding primitives. Users preferred the H2S tutorials over the basic tutorial, giving significantly higher ratings for satisfaction, accuracy, and enjoyment.

One possible future direction is to provide stroke level support to help users draw the final object contours, possibly by explicitly providing guidelines with respect to the scaffold primitives. Another direction would to explore new types of guidelines that can help reduce the number of unguided steps in a tutorial. A very interesting future question is to investigate if H2S really *teaches* users to sketch better by drawing “what you know.” While this is the ultimate goal of any sketching tutorial, answering this question will require a much more involved user study where we have to track and quantify user-specific improvements.

Acknowledgements

We thank our reviewers for their invaluable comments and the user study participants for their time and feedback. We also thank Daichi Ito, Paul Guerrero, Moos Hueting, Carlo Innamorati, Aron Monszpart, Tuanfeng Yang Wang, Tom Kelly, Bongjin Koo and Martin Kilian for their help, comments and ideas. This work was partially funded by the ERC Starting Grant SmartGeometry (StG-2013-335373), EPSRC EngD Centre EP/G037159/1 and gifts from Adobe.

References

BAE, S.-H., BALAKRISHNAN, R., AND SINGH, K. 2008. ILoveSketch: As-natural-as-possible Sketching System for Creating 3D Curve Models. In *Proc. UIST*, 151–160.

BENEDETTI, L., WINNEMÖLLER, H., CORSINI, M., AND SCOPIGNO, R. 2014. Painting with Bob: Assisted Creativity for Novices. ACM, 419–428.

BURNS, M., KLAWE, J., RUSINKIEWICZ, S., FINKELSTEIN, A., AND DECARLO, D. 2005. Line drawings from volume data. *ACM SIGGRAPH 24*, 3 (Aug.), 512–518.

CUMMMINGS, D., VIDES, F., AND HAMMOND, T. 2012. I don’t believe my eyes!: Geometric sketch recognition for a computer art tutorial. In *Proc. SBIM*, 97–106.

DECARLO, D., AND RUSINKIEWICZ, S. 2007. Highlight lines for conveying shape. In *Proc. NPAR*.

DECARLO, D., FINKELSTEIN, A., RUSINKIEWICZ, S., AND SANTELLA, A. 2003. Suggestive Contours for Conveying Shape. *ACM SIGGRAPH 22*, 3, 848–855.

DECARLO, D., FINKELSTEIN, A., AND RUSINKIEWICZ, S. 2004. Interactive rendering of suggestive contours with temporal coherence. In *Proc. NPAR*, 15–24.

DIXON, D., PRASAD, M., AND HAMMOND, T. 2010. iCanDraw: Using Sketch Recognition and Corrective Feedback to Assist a User in Drawing Human Faces. 897–906.

2016. Draw-a-box. `http://drawabox.com/lesson/6`. Accessed: 2016-10-23.

EDWARDS, B. 1999. *The New Drawing on the Right Side of the Brain*. Jeremy P. Tarcher/Putnam.

EISSEN, K., AND STEUR, R. 2007. *Sketching: Drawing Techniques for Product Designers*. BIS Publishers.

EISSEN, K., AND STEUR, R. 2011. *Sketching: The Basics*. BIS.

FERNANDO, P., PEIRIS, R. L., AND NANAYAKKARA, S. 2014. I-draw: Towards a freehand drawing assistant. OzCHI, 208–211.

FERNQUIST, J., GROSSMAN, T., AND FITZMAURICE, G. 2011. Sketch-sketch revolution: An engaging tutorial system for guided sketching and application learning. In *Proc. UIST*, 373–382.

FU, H., ZHJOU, S., LIU, L., AND MITRA, N. 2011. Animated construction of line drawings. *ACM SIGGRAPH 30*, 6.

GINGOLD, Y., VOUGA, E., GRINSPUN, E., AND HIRSH, H. 2012. Diamonds from the rough: Improving drawing, painting, and singing via crowdsourcing. In *Proc. HCOMP*.

GRABLER, F., AGRAWALA, M., LI, W., DONTCHEVA, M., AND IGARASHI, T. 2009. Generating photo manipulation tutorials by demonstration. In *ACM SIGGRAPH*, 66:1–66:9.

GRIMM, C. 2011. Results of an observational study on sketching results of an observational study on sketching. *SBIM*.

GUROBI OPTIMIZATION, I., 2015. Gurobi optimizer reference.

IARUSSI, E., BOUSSEAU, A., AND TSANDILAS, T. 2013. The drawing assistant: Automated drawing guidance and feedback from photographs. In *Proc. UIST*.

IARUSSI, E., BOMMES, D., AND BOUSSEAU, A. 2015. Bendfields: Regularized curvature fields from rough concept sketches. *ACM SIGGRAPH*.

IGARASHI, T., AND HUGHES, J. F. 2001. A Suggestive Interface for 3D Drawing. In *Proc. UIST*, 173–181.

KYPRIANIDIS, J. E., COLLOMOSSE, J., WANG, T., AND ISENBERG, T. 2013. State of the ‘art’: A taxonomy of artistic stylization techniques for images and video. 866–885.

LEE, Y. J., ZITNICK, C. L., AND COHEN, M. F. 2011. Shadowdraw: Real-time user guidance for freehand drawing. ACM SIGGRAPH, 27:1–27:10.

LIMPAECHER, A., FELTMAN, N., TREUILLE, A., AND COHEN, M. 2013. Real-time drawing assistance through crowdsourcing. *ACM SIGGRAPH 32*, 4 (July), 54:1–54:8.

LIU, J., FU, H., AND TAI, C.-L. 2014. Dynamic sketching: Simulating the process of observational drawing. In *Proceedings of the Workshop on Computational Aesthetics*, CAe ’14, 15–22.

MEHRA, R., ZHOU, Q., LONG, J., SHEFFER, A., GOOCH, A., AND MITRA, N. J. 2009. Abstraction of man-made shapes. *ACM SIGGRAPH Asia 28*, 5, #137, 1–10.

PAN, H., LIU, Y., SHEFFER, A., VINING, N., LI, C., AND WANG, W. 2015. Flow aligned surfacing of curve networks. *ACM SIGGRAPH 34*, 4.

SCHMIDT, R., KHAN, A., KURTENBACH, G., AND SINGH, K. 2009. On Expert Performance in 3D Curve-drawing Tasks. In *Proc. SBIM*, 133–140.

SCHMIDT, R., KHAN, A., SINGH, K., AND KURTENBACH, G. 2009. Analytic Drawing of 3D Scaffolds. In *ACM SIGGRAPH Asia*, 149:1–149:10.

SHAO, C., BOUSSEAU, A., SHEFFER, A., AND SINGH, K. 2012. Crossshade: Shading concept sketches using cross-section curves. *ACM SIGGRAPH 31*, 4.

SIMO-SERRA, E., IIZUKA, S., SASAKI, K., AND ISHIKAWA, H. 2016. Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup. *ACM SIGGRAPH 35*, 4.

2016. Sketch-a-day. `http://www.sketch-a-day.com/`. Accessed: 2016-10-23.

TAKAGI, S., MATSUDA, N., SOGA, M., TAKI, H., SHIMA, T., AND YOSHIMOTO, F. 2003. A learning support system for beginners in pencil drawing. In *Proc. CGI*, 281–282.

TCHALENKO, J. 2007. Eye movements in drawing simple lines. *PERCEPTION 36*, 8, 1152.

TCHALENKO, J. 2009. Segmentation and accuracy in copying and drawing: Experts and beginners. *Vision Research 49*, 8, 791 – 800.

XIE, J., HERTZMANN, A., LI, W., AND WINNEMÖLLER, H. 2014. PortraitSketch: Face Sketching Assistance for Novices. In *Proc. UIST*.

XU, B., CHANG, W., SHEFFER, A., BOUSSEAU, A., MCCRAE, J., AND SINGH, K. 2014. True2form: 3d curve networks from 2d sketches via selective regularization. *ACM SIGGRAPH 33*, 4.