SUPPLEMENTAL MATERIAL: Co-Constrained Handles for Deformation in Shape Collections

Mehmet Ersin Yumer   Levent Burak Kara

Carnegie Mellon University

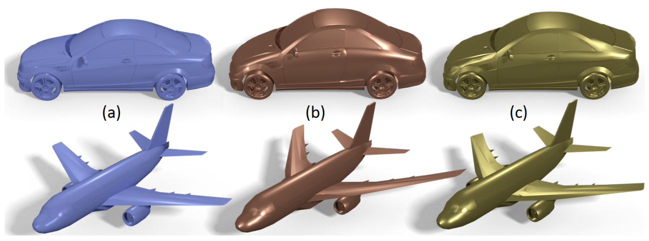

Figure 1: *(a) input models. (b) A user's deformation with Maya (car), Modo (airplane). (c) Same user's deformations in our system.*

1 Preliminary user study

We administered a small user study to benchmark our method against existing software suites. 13 users participated in our study. The users have mixed backgrounds including modelers in game industry, mechanical engineers, and product designers. All users are proficient in at least one of the following software: Maya, Modo, 3ds Max, ZBrush[1]. We asked them to perform two different deformation tasks in a guided fashion using both our software and the commercial software of their choice. This involved deforming the target models to make them visually similar to a deformed render model we showed them. After briefly introducing the user interface and the interaction modes in our software, we let the users take as much time as they needed for both tasks. An example creation of a user for both tasks is given in Figure 1. Similar to these results, all users were able to create the deformations in both our system and the commercial software. The average time to create the result with the commercial package was 8 minutes for the car and 11 minutes for the airplane wings (Figure 2). By contrast, all users were able to create the results in our software in less than 2 minutes. The main difficulties noted by the users with the commercial software were: (1) The absence of simple satisfactory mask or material to enable non-uniform local deformations, and (2) The need for extra processing to keep the disconnected parts intact without drifting during and after the deformation. These difficulties were not experienced in our system.

2 Constraint Propagation - Additional Results

Figure 3 shows how the randomly selected nodes for the subgraphs might change without adversely affecting the final deformations. Figure 4 shows a series of successive user edits and how the system responds automatically using constraint propagation described in the paper. Note that in Figure 4, hard constraints are disabled by the user (for instance the system permits the side surfaces to be rotated to create a taper from the top view). The system still utilizes the learned statistics to resolve the additional constraints for the remaining handles and deformation.

[1] *Maya* and *3ds Max* are trademarks of *Autodesk*. *Modo* is a trademark of *The Foundry*, and *ZBrush* is a trademark of *Pixelogic*.

3 Results of the Lamps Dataset

Figure 5 show additional results with the lamps dataset. Also, we demonstrate a one-to-one shape abstract style transfer in Figure 6, where two different targets deformed to match a single source. Note that, the co-analysis with a larger dataset (in this example with all the models in Figure 11) is still necessary to generate the co-constrained abstractions and cluster the handles. However, once this stage is performed, one can utilize the optimization in Equation 6 of the main paper with a single source.

4 Semi-Supervised Gaussian Mixture Model for Comparison with the Constrained Mean-Shift

Let $X = \{x_i\}, i = 1, \ldots, N$ be the observed surface feature vector from all the surfaces of all abstractions in category $\mathcal{C}$. The traditional M-component GMM is:

$$P(x|\Theta) = \sum_{k=1}^{M} \pi_k P(x|\theta_k) \tag{1}$$

where $\Theta = (\pi_1, \ldots, \pi_M, \theta_1, \ldots, \theta_M)$ are the parameters of the GMM ($P(x|\theta_k) = \mathcal{N}(\sigma_k, \Sigma_k)$ is a normal distribution and π_i are the mixing coefficients).

If we let $Y = \{y_i | i = 1, \ldots, N\}, y_i \in \{1, \ldots, M\}$ be the latent variables (cluster assignments of the observed data in the GMM), then the complete data-likelihood for the GMM becomes

$$P(X, Y|\Theta) = P(X|Y, \Theta) P(Y|\Theta) \tag{2}$$

where $P(X|Y, \Theta) = \prod_{i=1}^{N} P(x_i|\theta_{y_i})$, and $P(Y|\Theta) = \prod_{i=1}^{N} \pi_{y_i}$.

Following [Lu and Leen 2007] we can incorporate constraints into the model by introducing a weighting function $g(Y)$

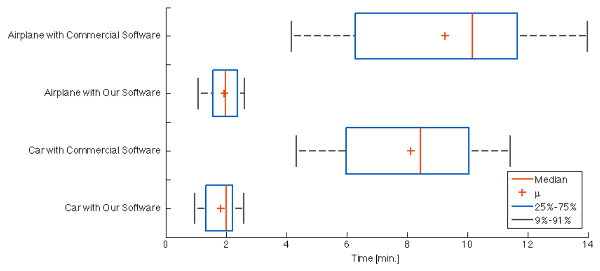

Figure 2: *Box and whiskers plot for the user study task completion times.*

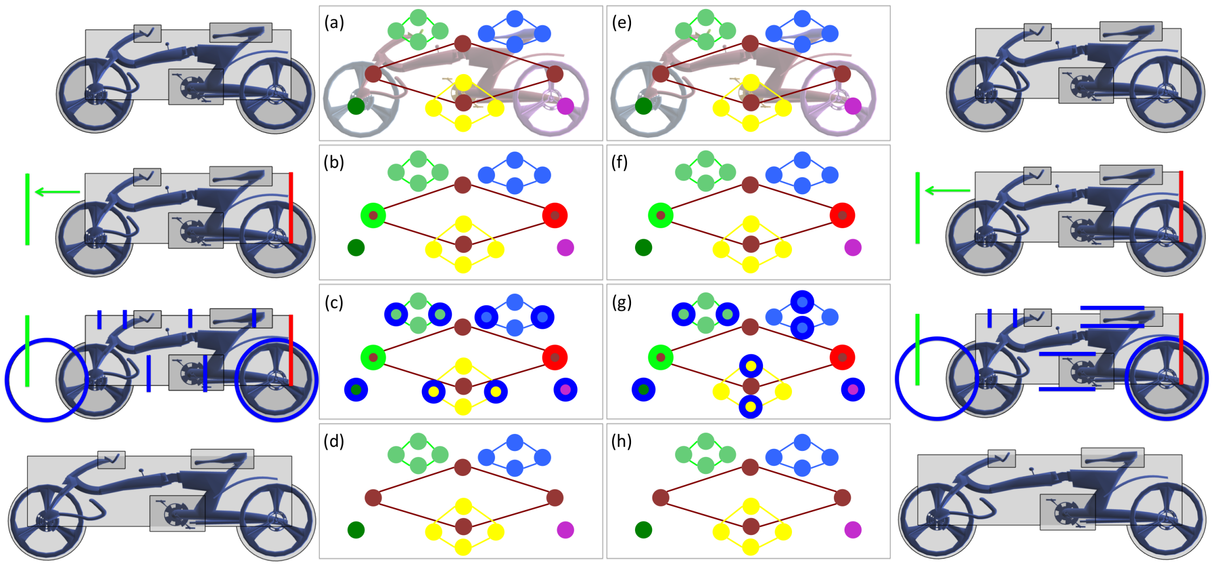

Figure 3: *Left: Constraint propagation presented in the main paper. (a) Contact graph of handles. Surfaces into the page are omitted for viewing simplicity. (b) User edited handle in green, and system anchored handle in red. (c) System selected additional handles to be constrained in blue, for which the positions are computed by Equation 4 in the paper. (d) Resulting deformations. Right: Constraint aggregation resulting from a another random pick of the initial handle node in subgraphs. (e-h) on the right corresponds to (a-d) on the left. Note the difference in the seat's and pedal's subgraphs and the resulting handle constraints (c & g).*

$$
\begin{aligned}
P(Y|\Theta, G) &= \frac{1}{C}\left(\prod_{i=1}^{M} \pi_{y_i}\right) g(Y) \qquad (3)\\
g(Y) &= \prod_{i \neq j} e^{W_{ij}\delta(y_i, y_j)}
\end{aligned}
$$

Here $C = \sum_Y \left(\prod_j \pi_{y_j}\right) g(Y)$ is a normalization term. Prior clustering constraints can be incorporated through the weights $W_{ij} = W_{ji}$. $W_{ij} > 0$ indicates a soft *link* preference and $W_{ij} < 0$ indicates a soft *do-not-link* preference between x_i and x_j.

GMM clustering constraints. For clustering we introduce soft *link* constraints between the surfaces in the *seed surface clusters* (as described in Section 4.1 of the main paper). Specifically, we set $W_{ij} = K$ where K is a constant that controls the contribution of W in the GMM ($K = 1$ in our experiments). With these constraints, we can fit the GMM model with EM [Dempster et al. 1977] following the update rules of [Lu and Leen 2007].

Comparison with constrained mean-shift. As can be seen in Figure 6 of the main paper, this more compute intensive method has not shown significant accuracy gain in our experiments. However, with large datasets ($\#of models > 10$), Semi-supervised GMM has significant time disadvantages compared to the constrained mean-shift algorithm described in the main paper.

Additional Remarks about Clustering. Note that clustering is a process where success is highly dependent on the assumption that the data is well representative. For example, in the airplanes example, if the shape database that is used for learning mainly has two types of wings; one with straight, one that has noticeably high curvature curved wings then our system will cluster these two types into separate clusters and define constraints for both sets separately. However, note that in the real world most shapes exhibit a variety of different styles. If the shape database samples enough variety from the real world models, it will have the two end points (straight and high curvature curved wing), but also a variety of curved wings that slightly differ from each other starting from the straight one. In such a case, our algorithm will cluster all the wings into the same cluster, successfully revealing the variety.

5 Collections Used for Statistical Analysis and Handle Synthesis

Figures 7- 11 show the datasets used to guide the results of this work. We will make these datasets public where there exists no licensing limitation.

6 Additional Remarks about the User Interface

Moving or scaling entire segments. Our prototype user interface offers a mode for moving or uniformly scaling an entire segment. The user simply turns the relevant mode on, and manipulators that appear on any handle of the segment will move or scale the segment uniformly depending on the mode activated.

Silhouette sketching. Silhouette sketching in our prototype UI enables the user to switch to an orthographic view and sketch the desired silhouette for the selected free-form handle. Once the user draws the desired silhouette, our system beautifies the user-sketched curve. Afterwards, the mesh vertices of the handle is moved such that their depth is kept constant but their positions in the 2D orthographic plane is fit to the sketched silhouette curve by first matching the end points.

Violation notification. We display the violations as a list based on their types (*i.e.,* position violation, orientation violation, *etc.*) in our prototype UI, and if the user clicks on a violation in the list, it is visually displayed to the user by highlighting the involved handles.

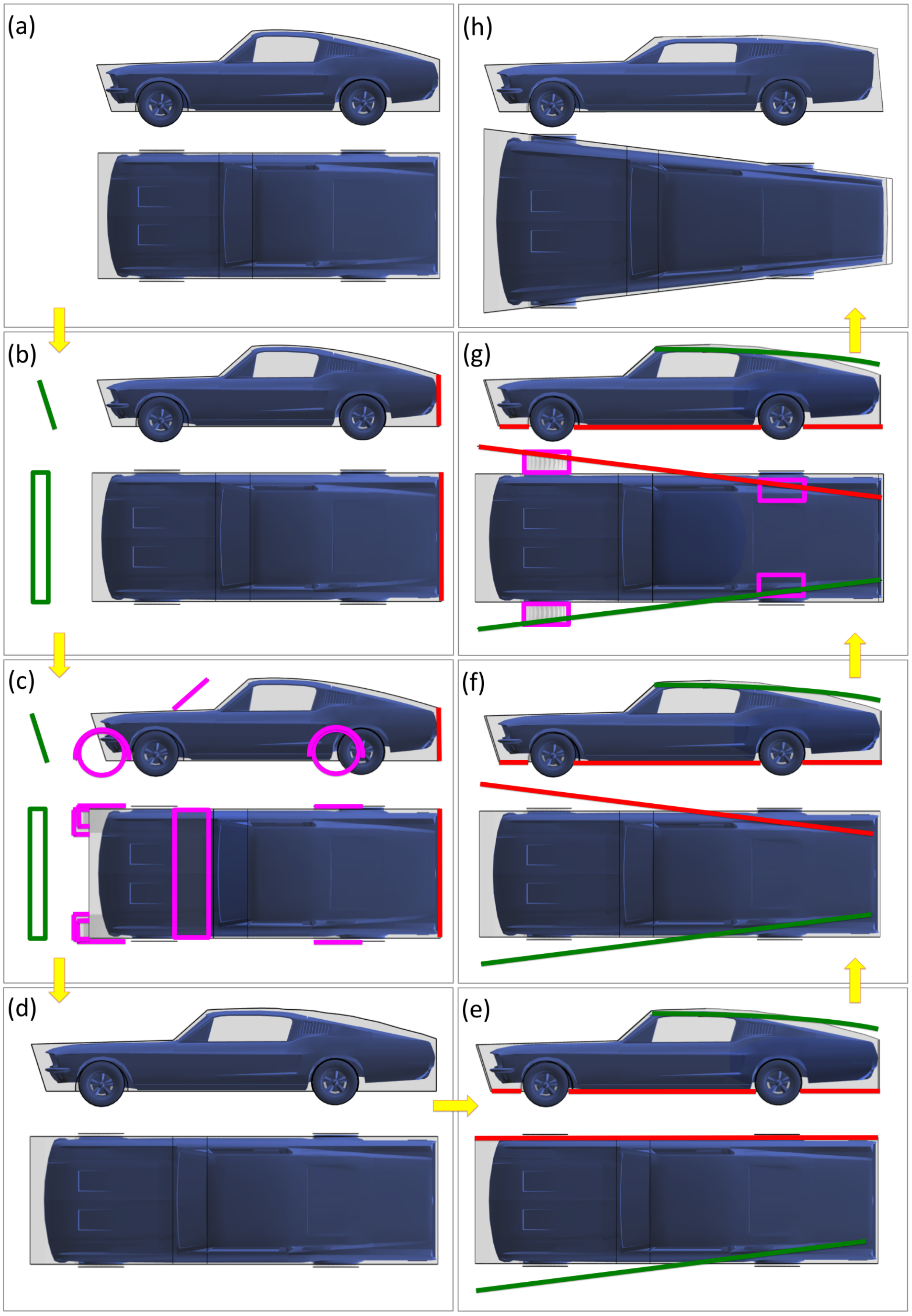

Figure 4: *(a) Initial model with handles. (b) User edit in green, automatically determined anchor in red. (c) User edit in green, anchor in red, system detected and propagated constraints in magenta. (d) Resulting model. (e) Further edits: free-form sketched top surface and rotated side surface in green, detected anchors in red. (f) Symmetry detected and rotation propagated to the anchor accordingly by the system. (g) User edit in green, anchor in red, system detected and propagated constraints in magenta. (h) Resulting model.*

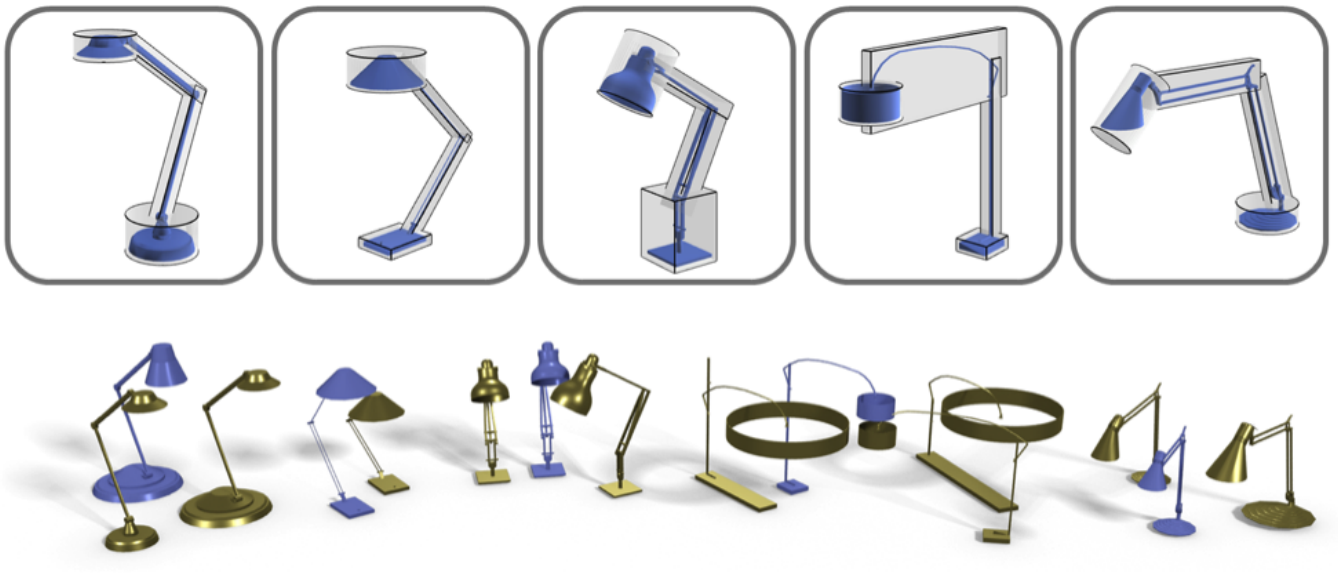

Figure 5: *Example handles and deformations from the lamps dataset.*

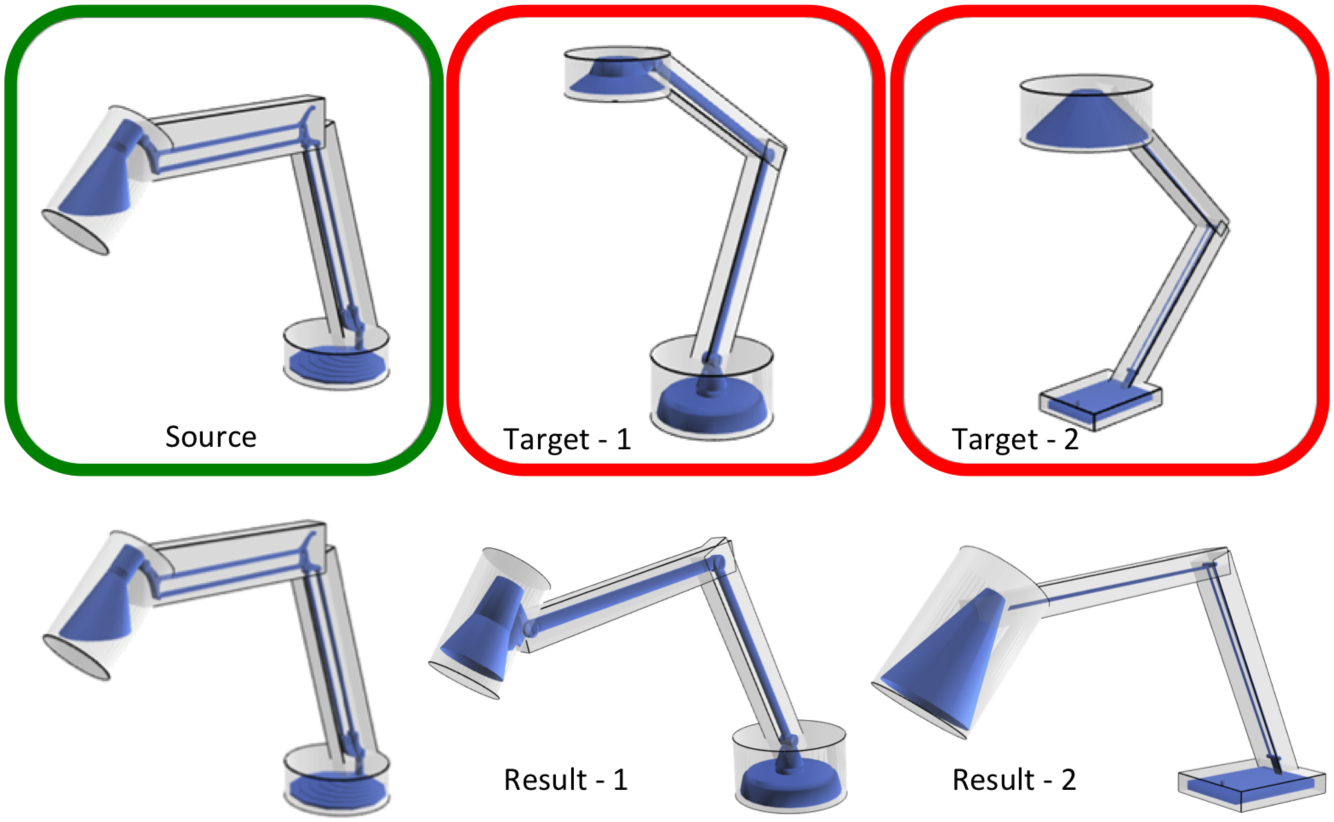

Figure 6: *Red: Source, Green: Targets where the style of the source will be transfered. Bottom: Results. In this example, both targets are independently optimized to match the target's style.*

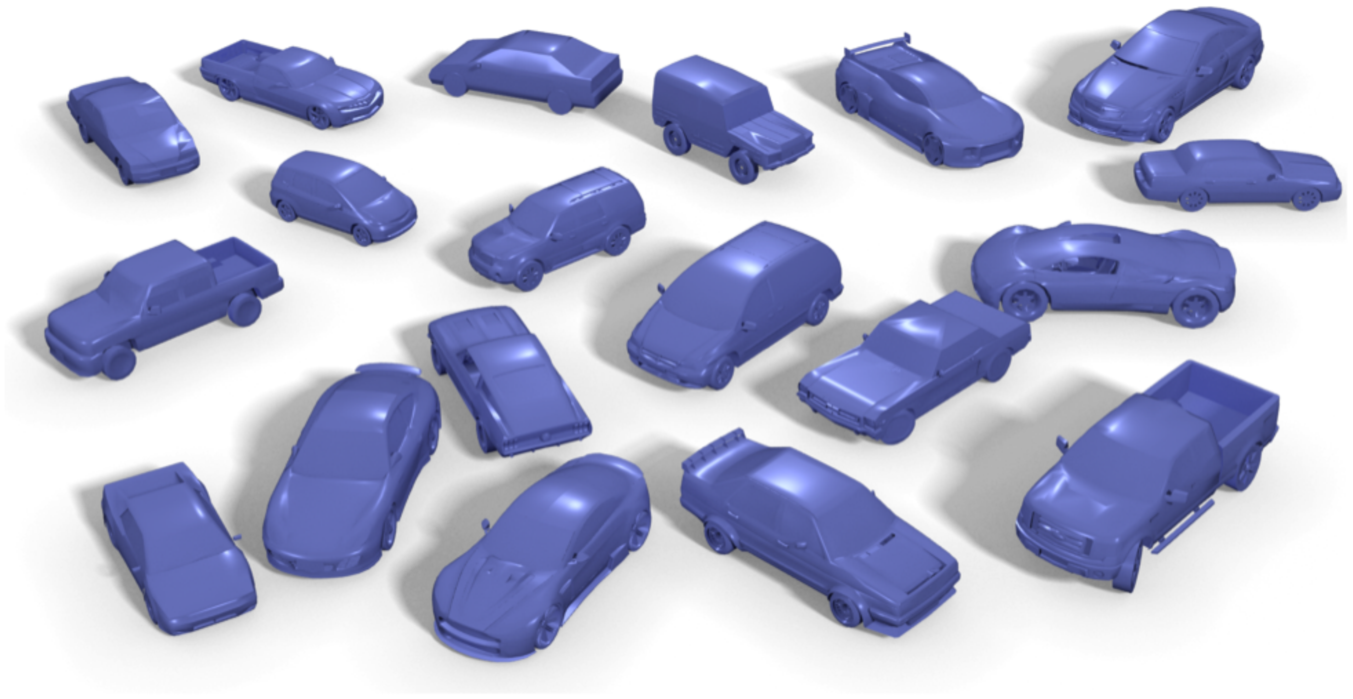

Figure 7: *Cars dataset used for in the statistical analysis and handle synthesis of results presented in Figure 1 of the main paper.*

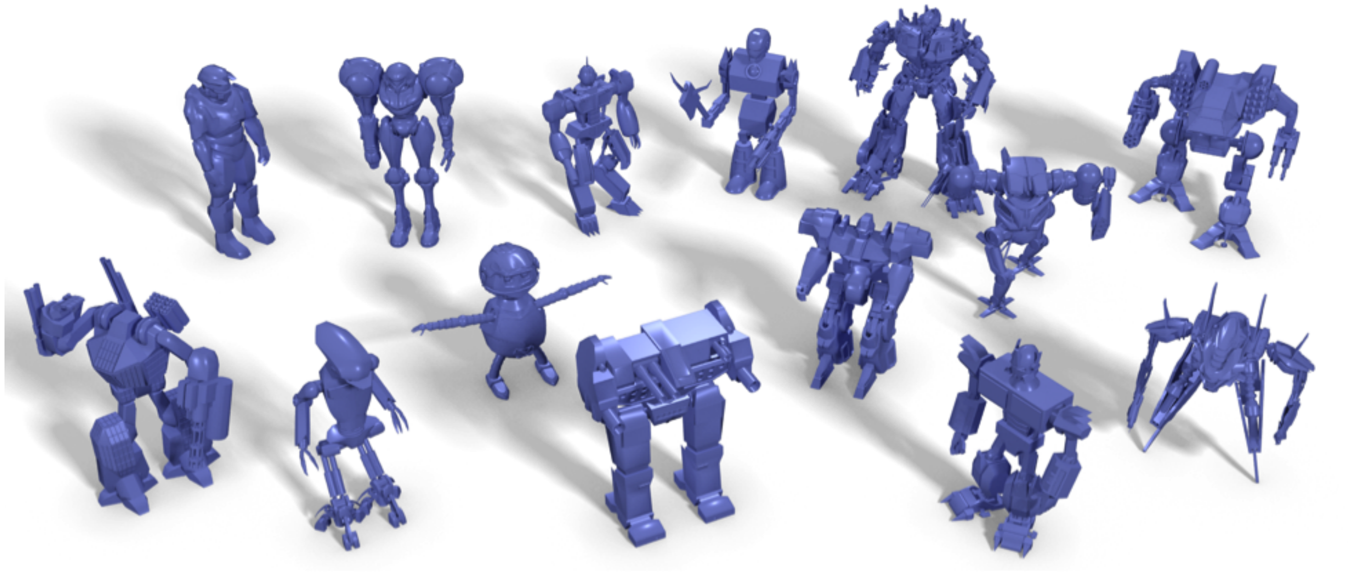

Figure 8: *Robots dataset used for in the statistical analysis and handle synthesis of results presented in Figure 19 of the main paper.*

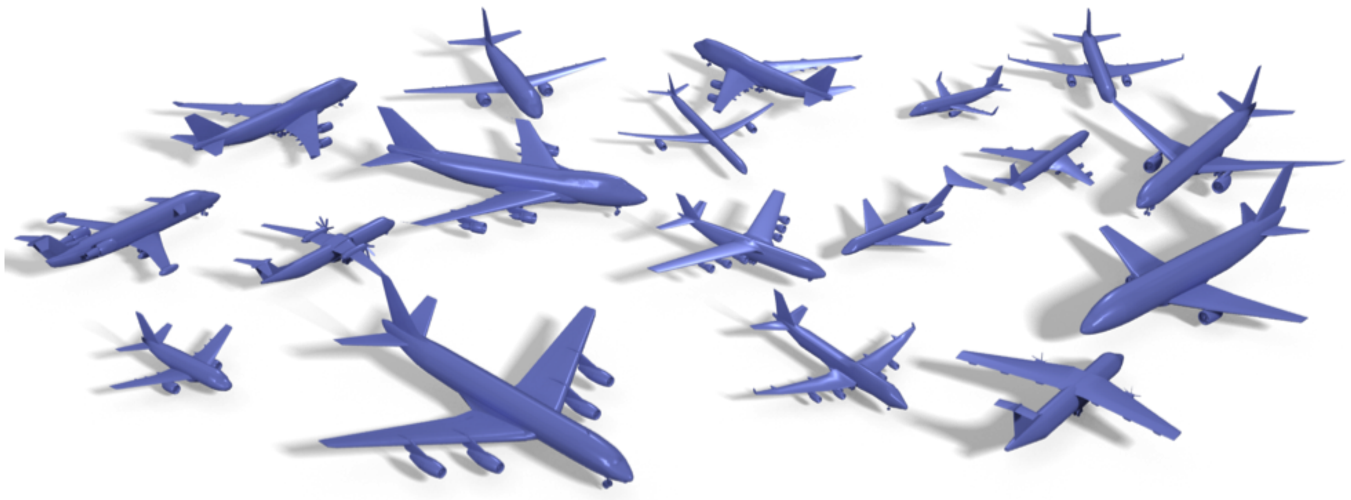

Figure 9: *Airplanes dataset used for in the statistical analysis and handle synthesis of results presented in Figure 19 of the main paper.*

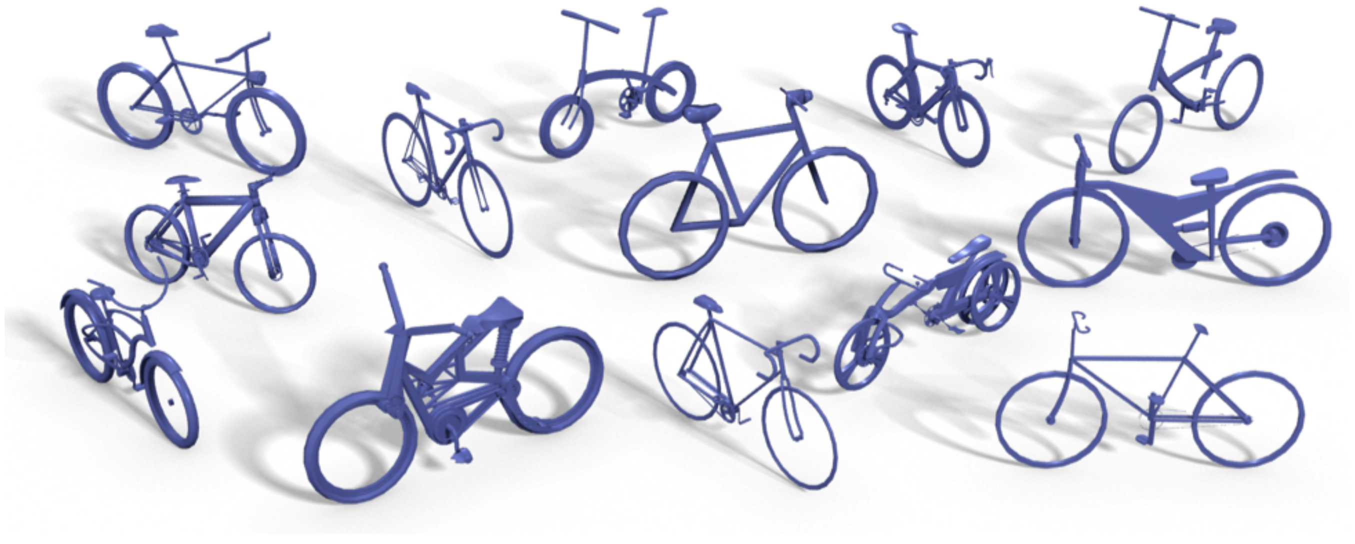

Figure 10: *Bicycles dataset used for in the statistical analysis and handle synthesis of results presented in Figure 19 of the main paper.*

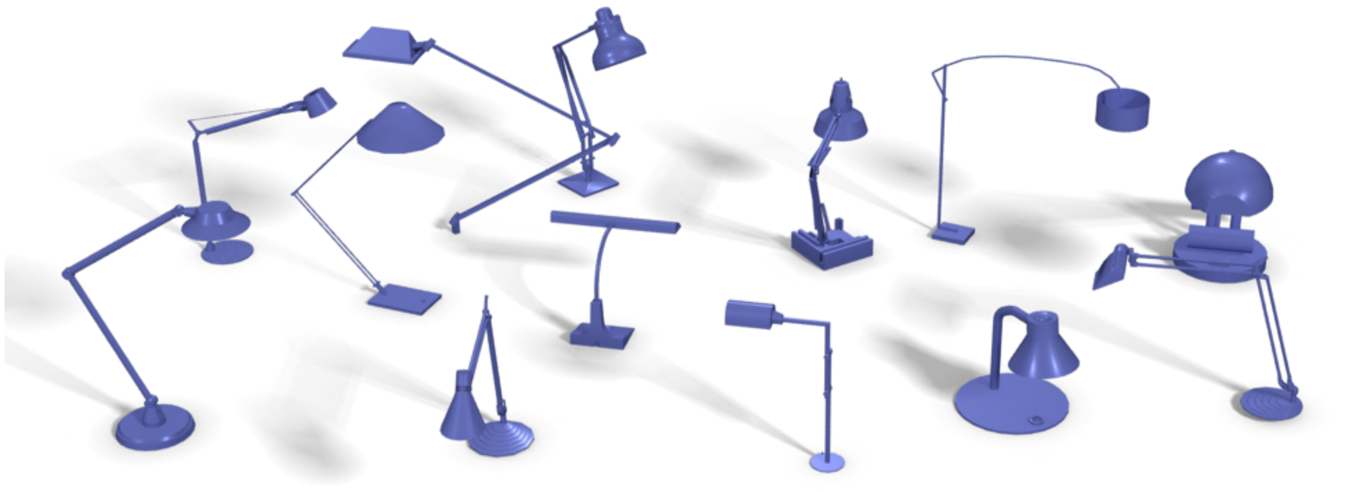

Figure 11: *Lamps dataset used for in the statistical analysis and handle synthesis of results presented in Figure 5 of this supplemental document.*

References

DEMPSTER, A. P., LAIRD, N. M., AND RUBIN, D. B. 1977. Maximum likelihood from incomplete data via the em algorithm. *J. of the Royal Statistical Society. Series B*, 1–38.

LU, Z., AND LEEN, T. K. 2007. Penalized probabilistic clustering. *Neural Computation 19*, 6, 1528–1567.