The Oomplet Dataset Toolkit: A flexible and extensible system for large-scale, multi-category image generation

John P. Kasarda
Carnegie Mellon University
Angela Zhang
Carnegie Mellon University Entertainment Technology Center
Hua Tong
Carnegie Mellon University Entertainment Technology Center
Yuan Tan
Carnegie Mellon University Entertainment Technology Center
Ruizi Wang
Carnegie Mellon University Entertainment Technology Center
Timothy Verstynen
timothyv@andrew.cmu.edu
Carnegie Mellon University
Michael Tarr
Carnegie Mellon University

Article

Keywords:

Posted Date: July 16th, 2024

DOI: https://doi.org/10.21203/rs.3.rs-4618904/v1

Additional Declarations: No competing interests reported.

The Oomplet Dataset Toolkit: A flexible and extensible system for large-scale, multi-category image generation

John P. Kasarda[1], Angela Zhang[2], Hua Tong[2], Yuan Tan[2], Ruizi Wang[2], Timothy Verstynen[1,3,*], and Michael Tarr[1,3,*]

[1]Department of Psychology, Carnegie Mellon University, Pittsburgh, 15213, USA
[2]Entertainment Technology Center, Carnegie Mellon University,Pittsburgh, 15213, USA
[3]Neuroscience Institute, Carnegie Mellon University, Pittsburgh, 15213, USA
[*]corresponding author(s): Timothy Verstynen (timothyv@andrew.cmu.edu), Michael J Tarr (michaeltarr@cmu.edu)

ABSTRACT

Understanding the dynamics of perceptual learning in humans, non-human animals, and artificial agents requires large stimulus sets with flexible features that can be used to discriminate across categorical groups. Here we introduce the Oomplet Dataset Toolkit (ODT), an open-source, publicly available toolbox for generating up to 9.1 million unique visual stimuli that are assembled across ten different feature dimensions. The resulting stimuli consist of cartoon-like humanoid characters – "Oomplets" – that are meant to be engaging, pleasant to look at, and can be appropriately used in research on a variety of populations, including children. Across several behavioral experiments, we show how eight of the ten possible dimensions that define an individual Ooomplet can be used by adults as effective classification boundaries for simple perceptual discrimination. The ODT thus provides a flexible and customizable way for generating very large and novel stimulus sets in order to study perceptual learning in both biological and artificial systems.

Background & Summary

The use of computer-generated stimuli in psychometric studies of behavior has a rich history. In Shepard and Metzler's[1] classic study of mental rotation, they used novel 3D objects "generated by digital computer and associated graphical output" to understand internal mental imagery (Figure 1A). Since then, the creation of visual stimuli has continued to be driven by advances in computer graphics. Salient examples from influential studies include 3D "wire-frame" objects[2] (Figure 1B) and 3D blob-like "amoebae" objects[3] (Figure 1C). As computer graphics tools became more widely accessible, the generation of novel visual objects as stimuli became increasingly more complex and realistic. Helping drive this trend, our lab has developed multiple complex visual stimulus datasets[4,5] and made them publicly accessible[6] – a notable example of this work being the "Greebles"[7] (Figure 1D) which have been used in well more than 100 different studies. A recent and non-exhaustive list of examples from the field includes "smoothies, spikies, and cubies"[8] (Figure 1E), "Ziggerins"[9] (Figure 1F), "digital embryos"[10] (Figure 1G), the NOUN Database[11] (Figure 1H), "Widgets"[12] (Figure 1I), and "Sheinbugs"[13] (Figure 1J), many of which emerged from collaborations within the Perceptual Expertise Network (PEN)[14].

Many of these visual datasets were created using a compositional approach in which individual parts from a dictionary were sampled and combined in different configurations to form complex objects[4,7,15]. Other datasets have been generated parametrically using varying values within mathematical functions to deform 3D shapes, define parts, and specify attachment points[2,3,8,10,16]. Still other datasets were created by hand, relying on 3D modeling skills rather than explicit algorithms[5,7]. One characteristic across almost all of these datasets is the relatively low number of available stimuli and stimulus categories. While some datasets with a broad dictionary of parts potentially allow for thousands of novel stimuli and hundreds of categories, the selection of parts, placement in different configurations, and category boundaries were all accomplished manually, making large-scale stimulus generation *ad hoc* and time-consuming[5]. Moreover, while the shape and configuration differences enable the possibility of many different well-defined visual categories, the actual number available is quite small – on the order of 10-20 at best. In contrast, visual datasets created through parametric variations allows for nearly an infinite number of different individual stimuli, but are less suited to being organized into a large number of natural visually-defined categories[2,3,8]. This is because the shape and configural variations across different parameter values are metric, meaning that visual categories, while definable, are not perceptually salient or aligned with how humans typically infer categorical boundaries[17].

To this end, we developed the Oomplet Dataset Toolkit (ODT), which is designed to create complex novel stimuli through a controllable generative process, using an extensive parts dictionary and contrastive part attributes. ODT was created as a stand-alone component of an interactive virtual environment designed to look at the dynamics of cooperative learning, where one task involves learning complex perceptual discriminations. Within this context, our objectives in creating ODT were as follows: 1) enable the generation of a millions of individual stimuli and a large number of categories; 2) enable the use of a large dictionary of reusable parts defined by a wide variety of visual dimensions (e.g., color, shape, orientation, spacing, etc.); 3) enable visually-salient conjunctions and disjunctions of parts so as to create well-defined categories and category hierarchies; 4) build a toolkit that is user controllable to enable automatic generation of stimuli, but with fine-grained user control over parts, part attributes, and categories; 5) build a toolkit that requires only standard end-user skills (e.g., no programming knowledge), but that is extensible for users with artistic and/or programming knowledge. ODT is unique in realizing these objectives, providing a powerful stimulus generation toolkit that allows users to create a large number of visually-defined natural categories with potentially hundreds of thousands of heirarchically-nested, individual exemplars per category. As such, ODT has potential applications in the psychological, neuroscientific, and artificial intelligence domains.

Methods

ODT is a user-friendly and customizable python-based pipeline for generating large sets of unique stimuli, “Oomplets”, and sorting them into hierarchically organized categories based on user-specified classification dimensions applied to the Oomplets’ visual features. The pipeline consists of two python scripts (`generate.py` and `categorize.py`) and 148 component images that are combined to create 9.1 million unique visual stimuli (Figure 2). These scripts, component images, and other relevant files are available in a publicly accessible repository (https://github.com/CoAxLab/OompletDatasetToolkit).

Components

The components consist of images of various types of body parts or features to be used as references in creating individual Oomplets – humanoid candy stimuli. These images are stored as *.png* files in the subdirectories of the repository’s “Components” directory. Each individual Oomplet stimulus image is made up of instances from seven classes of component images (Figure 2). Because some components provide more than one attribute, a total of ten different attributes are recorded in a JSON formatted identification (ID) file associated with each generated Oomplet.

Generate

The `generate.py` script consists of Python code that creates an Oomplet by selecting one file from each of the seven component directories and compiling these components into a complete Oomplet. To accomplish this, `generate.py` employs OpenCV’s[18] image processing functionality to visually parse the components and re-draw them jointly onto a common image depicting the newly created Oomplet. When invoked, `generate.py` is passed a number of required and optional arguments that allow user control of customization, computational processing, and output location. Full documentation of the script arguments and their functions is available in the repository.

Each Oomplet is defined by the user along 10 attributes nested within the 7 classes of components. To create a unique individual Oomplet, the generate process selects a value for each attribute, where there are 2-4 possible values for each attribute that have been randomly ordered. As mentioned, `generate.py` captures these values and writes them into the associated JSON ID file. These ID files are what allows the pipeline to then sort the Oomplets into distinct visual categories using `categorize.py`.

Categorize

The `categorize.py` script consists of Python code that categorizes each Oomplet through a set of user-defined attribute criteria. The user specifies which attributes (a minimum of 1 and up to all 10 attributes) will be used to determine category membership. For each attribute, the user specifies the value of that attribute that helps define the category, where the complete category definition is the intersection of all 10 attribute values. As illustrated in Figure 3, for each Oomplet that satisfies the criteria, `categorize.py` makes a copy of that Oomplet file and places it in automatically created output directories corresponding to Oomplets that match the criteria (“Match_[TIMESTAMP]”) and those that do not (“NoMatch_[TIMESTAMP]”). When invoked, `categorize.py` is passed a number of required and optional arguments that allow user control of input location, categorization criteria, and other customizations.

Because categorization is based on a concatenation of values for each attribute, categorical boundaries can be along a single attribute or the intersection of many attributes. Additionally, a hierarchy may be created by running `categorize.py` multiple times (i.e., once to categorize all Oomplets with a common set of attribute values and then a second time to further sort Oomplets in one of the first sort categories based on a new set of attribute values).

Example

As a snapshot of the whole process, let us suppose that a stimulus is compiled by the `generate.py` script using the components `<mouth,open,1.png>` as reference. This stimulus would be recorded to have the attribute “open” for “mouth openness” in its ID file. The `categorize.py` script, when specified to look for images with closed mouths, would put this stimulus into the “NoMatch” sorting directory.

Data Records

We used the process described in Methods to generate roughly all 9 million possible unique Oomplets. The Oomplets and the code used to generate them are organized according to the TIER Protocol 4.0 directory architecture[19] (Figure 4). Oomplet images were then stored in *PNG* format, with transparent backgrounds. Additional scripts used to help with building the validation study are included in the *Scripts* directory.

Technical Validation

In order to evaluate the perceptual discriminability of the different Oomplet attributes, we conducted a series of online studies using a forced choice discrimination task. We chose the eight most relevant attributes that can be used as binary classification boundaries and tested each attribute individually. In cases where attributes had more than two possible values (e.g., shape can be ’sharp’, ’mixed’, or ’round’), we only used the two most extreme values as the classification features (e.g., ’sharp’ and ’round’). Each experiment used its own set of roughly 40,000 unique Oomplets.

Participants

All study procedures were approved by the Carnegie Mellon University Institutional Review Board and informed consent was obtained prior to each participant starting the study. Studies were hosted on Connect[20] , CloudResearch’s online crowd-sourcing platform. We recruited 50 participants for each study. Participants were excluded from the final analysis if their responses were improperly submitted to the cloud server or if they responded to fewer than 50 trials. The final sample sizes per condition were: Shape (N = 50), Pattern (N = 48), Mouth Openness (N = 49), Leg Length (N = 50), Eye Lash (N = 43), Eye Separation (N = 50), Hue (N = 47), and Arm Orientation (N = 48). Individuals who reported being colorblind were excluded from recruitment. We did not collect or restrict recruitment along any demographic category.

Task

We built the eight single-task studies using Gorilla’s Experiment Builder (Task Builder 2)[21], with each task reflecting a single attribute for the classification boundary. In the task, participants were presented with 300 Oomplets (presented to participants as "candies" in this study), one at a time, and were asked whether the Oomplet is Bitter ("f" key) or Sweet ("j" key). Trials were counterbalanced, assuring that 150 images of each type were always presented. The terms “bitter” and “sweet” were chosen to avoid bias towards any of the humanoid characteristics; the bitter and sweet sets were created using Match and NoMatch criteria in the set creation with `categorize.py`.

Each trial consisted of three distinct phases (see Figure 5). The trial started with a *fixation* phase, where the participant was presented with a centrally presented cross to bring their attention to the middle of the screen. This phase lasted 200ms, after which the cross was removed. After 100ms, the stimulus was presented (*stimulation* phase) with the words "Bitter" (left) and "Sweet" (right) presented on either side of the Oomplet, along with the keyboard response associated with each choice. Participants were given 2000ms to respond. Responses occurring after 2000ms were not recorded. Key presses were also not recorded for the first 250ms following stimulus onset in order to avoid false start responses. Finally, during the *feedback* phase, participants were informed via icons as to whether their response was correct or incorrect. Importantly, participants were not given explicit instructions as to what attributes defined the two categories and had to simply rely on this feedback to learn the relevant category boundaries.

In order to avoid any potential biases from stimulus characteristics (e.g., implicit assumptions on color to bitter/sweet mapping) in the resulting choices, we counterbalanced the bitter/sweet mapping across participants. Half of the participants would get one mapping and the remaining half the other. Task assignments were random without replacement, targeting 25 participants per group.

Analysis

To visualize how well each attribute could be detected as a classification dimension, we calculated two signal detection measures[22]. First, we estimated the d' for each participant as $d' = \Phi(hits) - \Phi(fa)$, where $hits$ represents the true classification rate, and fa reflects the false alarm rate for incorrectly classifying a stimulus as sweet or bitter. The d' measure reflects the signal-to-noise ratio of the discrimination as standard deviations away from the noise distribution. The distribution of d'

measures, across participants, was evaluated independently for each task. When participants had a perfect classification rate, we capped the d' value at 5.

In addition, we plotted the receiver operating characteristic (ROC) curve across tasks. This presents the joint distribution of fa and $hits$ rates, and allows for visualizing when inter-subject responses vary along d' (reflecting consistent varying thresholds applied to the same signal-to-noise ratio) or criterion (reflecting varying signal-to-noise ratios along the same selection threshold).

Sensitivity Analysis

Figure 6 shows the distribution of d' scores, across participants, for each attribute tested. Attributes are sorted from lowest to highest average d' and errorbars reflect the 95% confidence intervals. We see that the eye distance, leg length, arm orientation, and eye lash attributes are unreliable dimensions for classification, reflected by the fact that the confidence intervals overlap with zero. The mouth openness and texture pattern show a modest discriminability, with mean d' values of 0.873 and 1.046 respectively. However, we see that this comes with a high degree of variability across participants, with a somewhat bimodal distribution of individual scores. One mode of participants sits around zero, indicating lack of discriminability. The other mode has very high d' values ranging from 1 to almost 4. Finally, body shape and color had the strongest discriminability, with mean values of 2.040 and 2.301 respectively. For both of these attributes, the spread of individual d' values was fairly broad, with some participants hovering near zero and 2 participants maxing out at d' values of 5 (reflecting perfect performance).

As an additional evaluation of participant performance, we also plotted the hit vs. false alarm rate for each participant in each task as an ROC curve. This allows for assessing the sensitivity of discrimination and response bias of each participant more clearly. We see two general patterns in these ROC plots, reflecting the general split between attributes with strong discriminability and those with weak perceptual discriminability. For attributes that had overall low d' scores, we see distributions of hit vs. false alarm rates centered near the unity diagonal, reflecting performance near chance. This suggests that those attributes have low signal-to-noise. In contrast, there is a separate, and somewhat orthogonal, cluster in the upper left portion of the plot that corresponds to attributes with high d' values. The direction of the distribution in this cluster reflects variation along a common threshold, suggesting that the discriminability of these attributes largely reflects a very high signal-to-noise ratio and thus variability across participants generally reflecting reliance along a common selection criterion.

Our technical validation reveals a wide range of perceptual discrimination abilities for human testers, both within and across attributes. Certain attributes are easier to use as classification boundaries than others. This allows for customizing the difficulty of perceptual classification depending on the experimenter's needs.

Usage Notes

Here, we provide detailed instructions to ensure bug-free usage of the Bit-or-Sweet pipeline. Start by completing the following steps to complete the initial setup of the stimulus generator.

Installation

1. Clone the repository locally
2. Set up a Python virtual environment in the root directory

```
% python -m venv venv
```

3. Activate the virtual environment based on OS.

   MacOS or Unix:

```
% source venv/bin/activate
```

   Windows:

```
% venv\Scripts\activate
```

4. Install all the requirements

```
% pip install -r requirements.txt
```

Implementation

Following installation, the `generate.py` script can be run from the command line. This is where users can specify any of the various options available to make their unique set of stimuli.

```
% python generate.py [-h] [-n N] [-p] [-c C] [-v] [-k] [-s S]
```

Options:

```
-h, --help  show this help message and exit
-n N        number of candies to generate N (def: all combinations)
-p          multiprocessing flag (def: off)
-c C        max number of processes to spawn if multiprocessing (def: 4)
-v          verbose (def: off)
-k          keep existing files in output folder (def: off)
-s S        seed value for randomly generated candies (def: 0)
```

The image and meta file output of this script will be located in the OompletToolkit/Output/Oomplets/ directory. Now, the `categorize.py` script may be run from the command line.

```
% python categorize.py [-h] [-d D] [-i I] [-k] [-a]
```

Options:

```
-h, --help     show this help message and exit
-d, --def      define your 'bitter' images (required)
-i, --input    name of the directory from which Oomplets will be sorted
-k             keep existing files in output folders (def: off)
-a, --any      flags Oomplets with ANY of defining attributes as Match (def: off)
```

The `categorize.py` script was made to be easily customized. The `-d` option allows users to choose any number of non-contradicting attribute values to define their Match and NoMatch Oomplet groups. Attribute value specifications must be typed in the terminal exactly as shown in the list below.

```
'color_cool', 'color_warm',
'shape_sharp', 'shape_mixed', 'shape_round',
'lash_yes', 'lash_no',
'wide_eyes', 'middle_eyes', 'narrow_eyes',
'short_legs', 'middle_legs', 'long_legs',
'feet_left', 'feet_right', 'feet_in', 'feet_out',
'open_mouth', 'closed_mouth',
'dots_pattern', 'stripes_pattern',
'right_arm_down', 'right_arm_up', 'left_arm_down', 'left_arm_up'
```

Example

This section will show each step a user would take in order to generate a set of 200 images, and sort them based on their pattern and eye lashes, using a MacOS computer. First, the user needs to set up their virtual python environment.

```
% python -m venv venv
% source venv/bin/activate
% pip install -r requirements.txt
```

Next, the user must navigate to the `Scripts/ProcessingScripts` directory. From here, they will run the generation script using this command:

```
% python generate.py -n 200
```

The user has now created 200 unique images in the `Output/Oomplets` directory. Now, they must navigate to the `Scripts/AnalysisScripts` directory, where the categorization script is located. To sort the images based on their desired attributes, the user must use this command:

```
% python categorize.py -d stripes_pattern lash_no
```

Once the script has finished running, the user will now have two new directories. Each image that has a striped pattern, *and* eye lashes will be located in the `Output/Match` directory. All images that do not meet this requirement will be located in the `Output/NoMatch` directory.

Extending ODT

One attractive characteristic of the ODT that is its extensibility. At present, the generated stimuli are static. However, because of their humanoid appearance and compositional structure, it would relatively straightforward to animate them (e.g. using Spine from esoteric software <https://esotericsoftware.com/>). This opens up the possibility for a wide range of dynamic attributes crossed with the predefined part attributes. It is also straightforward to introduce new parts through the creation of new component image files or to increase the number of levels per attribute so as to introduce more fine-grained category distinctions. These future advances would further expand the utility of Oomplets as research stimuli and improve the potential reach of the ODT.

Data availability

The code and assets used in the ODT stimulus generation pipeline are available on GitHub:
(https://github.com/CoAxLab/OompletDatasetToolkit).

Acknowledgements

The authors would like to thank Michael Christel for his feedback and support. This work was sponsored, in part, by AFOSR/AFRL award FA9550-18-1-0251

Author contributions statement

MT and TV conceived of the design of the project, oversaw completion of the project, and contributed to writing of this paper. AZ, HT, RW, and YT implemented the original Ooomplet design and code generation. JPK contributed to designing and implementing the experimental tasks, including all data analysis, as well as leading the writing of the manuscript.

Competing interests

The authors have no competing interests to declare.

References

1. Shepard, R. N. & Metzler, J. Mental rotation of three-dimensional objects. *Science* **171**, 701–703 (1971).

2. Poggio, T. & Edelman, S. A network that learns to recognize three-dimensional objects. *Nature* **343**, 263–266 (1990).

3. Bülthoff, H. H. & Edelman, S. Psychophysical support for a two-dimensional view interpolation theory of object recognition. *Proc. Natl. Acad. Sci. United States Am.* **89**, 60–64 (1992).

4. Hayward, W. G. & Williams, P. Viewpoint dependence and object discriminability. *Psychol. Sci.* **11**, 7–12 (2000).

5. Tarr, M. J. Visual object recognition: Can a single mechanism suffice? In *Perception of Faces, Objects, and Scenes: Analytic and Holistic Processes* (Oxford University Press, Oxford, UK, 2006).

6. Tarr, M. J. tarrlab stimuli. https://sites.google.com/andrew.cmu.edu/tarrlab/stimuli (2024). Accessed: 01-18-2024.

7. Gauthier, I. & Tarr, M. Becoming a 'Greeble' expert: Exploring mechanisms for face recognition. *Vis. Res.* **37** (1997).

8. Op de Beeck, H. P., Baker, C. I., DiCarlo, J. J. & Kanwisher, N. G. Discrimination training alters object representations in human extrastriate cortex. *J Neurosci* **26**, 13025–13036 (2006).

9. Wong, A. C.-N., Palmeri, T. J., Rogers, B. P., Gore, J. C. & Gauthier, I. Beyond shape: how you learn about objects affects how they are represented in visual cortex. *PLoS One* **4**, e8405 (2009).

10. Hegdé, J., Thompson, S., Brady, M. & Kersten, D. Object recognition in clutter: Cortical responses depend on the type of learning. *Front. human neuroscience* **6**, 170 (2012).

11. Horst, J. S. & Hout, M. C. The Novel Object and Unusual Name (NOUN) Database: A collection of novel images for use in experimental research. *Behav. Res. Methods* **48**, 1393–1409 (2016).

12. Lebaz, S., Sorin, A.-L., Rovira, K. & Picard, D. Widgets: A new set of parametrically defined 3D objects for use in haptic and visual categorization tasks. *Eur. Rev. Appl. Psychol.* **70**, 100552 (2020).

13. Jones, T. *et al.* Neural and behavioral effects of subordinate-level training of novel objects across manipulations of color and spatial frequency. *Eur. J. Neurosci.* **52**, 4468–4479 (2020).

14. Gauthier, I., Tarr, M. & Bub, D. *Perceptual Expertise: Bridging Brain and Behavior* (Oxford University Press, Oxford, UK, 2009).

15. Tarr, M. J., Bülthoff, H. H., Zabinski, M. & Blanz, V. To What Extent Do Unique Parts Influence Recognition Across Changes in Viewpoint? *Psychol. Sci.* **8**, 282–289 (1997).

16. Vuong, Q. C. *et al.* Facelikeness matters: A parametric multipart object set to understand the role of spatial configuration in visual recognition. *Vis. Cogn.* **24**, 406–421 (2016).

17. Rosch, E., Mervis, C. B., Gray, W. D., Johnson, D. M. & Boyes-Braem, P. Basic objects in natural categories. *Cogn. Psychol.* **8**, 382–439 (1976).

18. Bradski, G. The OpenCV Library. *Dr. Dobb's J. Softw. Tools* (2000).

19. ProjectTIER. Tier protocol 4.0. https://www.projecttier.org/tier-protocol/protocol-4-0/ (2023). Accessed: 05-05-2023.

20. Cloudresearch. Connect by cloudresearch. https://www.cloudresearch.com (2023). Accessed: 06-05-2023 to 08-09-2023.

21. Gorilla. Gorilla task builder 2. https://www.gorilla.sc (2023). Accessed: 01-19-2023 to 08-23-2023.

22. Wickens, T. D. *Elementary signal detection theory* (Oxford university press, 2001).

Figures & Tables

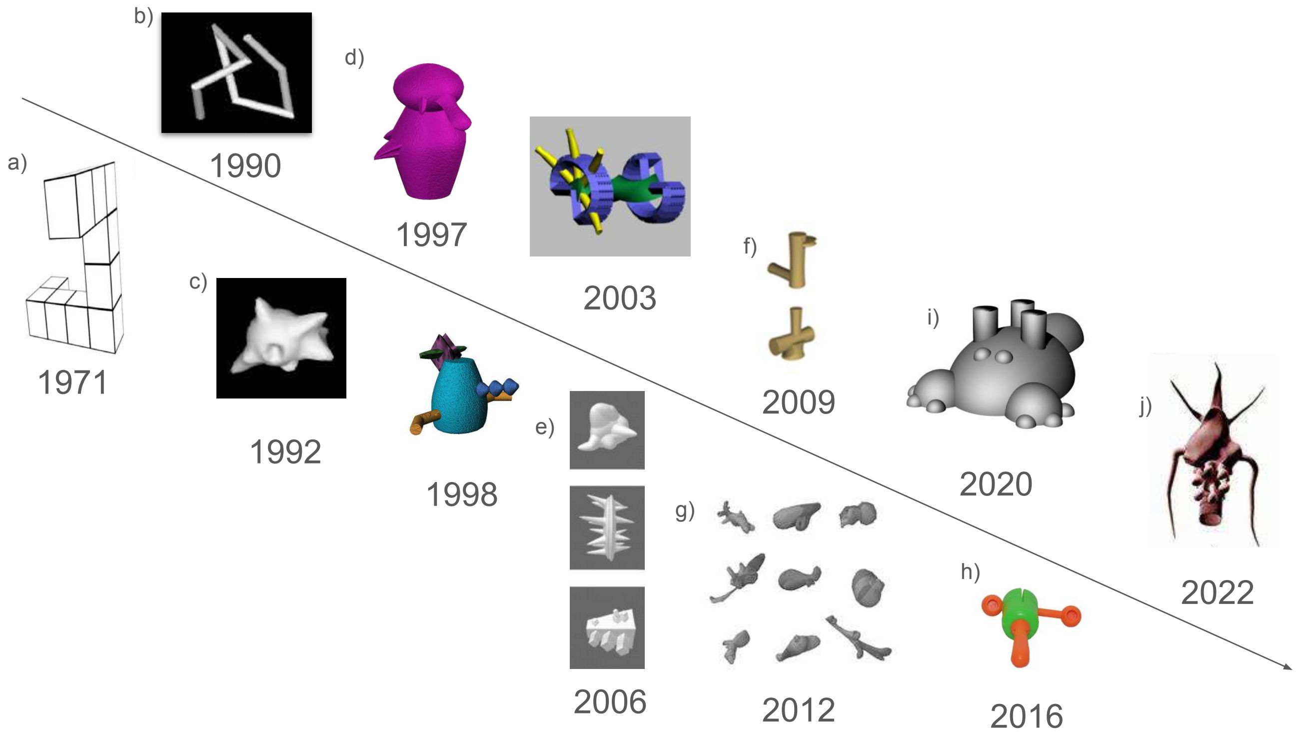

Figure 1. A historical tour of computer generated stimuli for psychometric studies. Timeline not to scale. a) Shepard, R., et al. (1971); b) Poggio, T., et al. (1990); c) Bülthoff, H., et al. (1992); d) Gauthier, I., et al. (1997); e) Op de Beeck, H. P., et al. (2006); f) Wong, A. C.-N., et al. (2009); g) Hegdé, J., et al. (2012); h) Horst, J. S., et al. (2016); i) Lebaz, S., et al. (2020); j) Jones, T., et al. (2020)

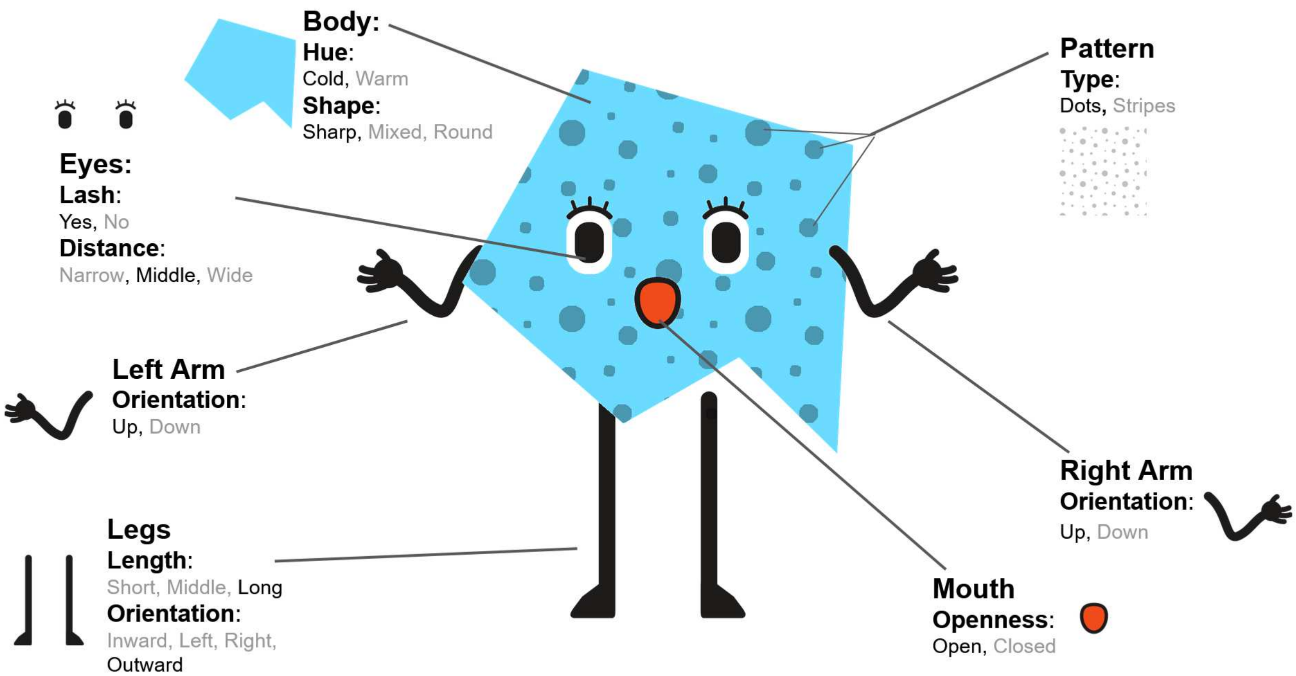

Figure 2. An example Oomplet with each component and attribute.

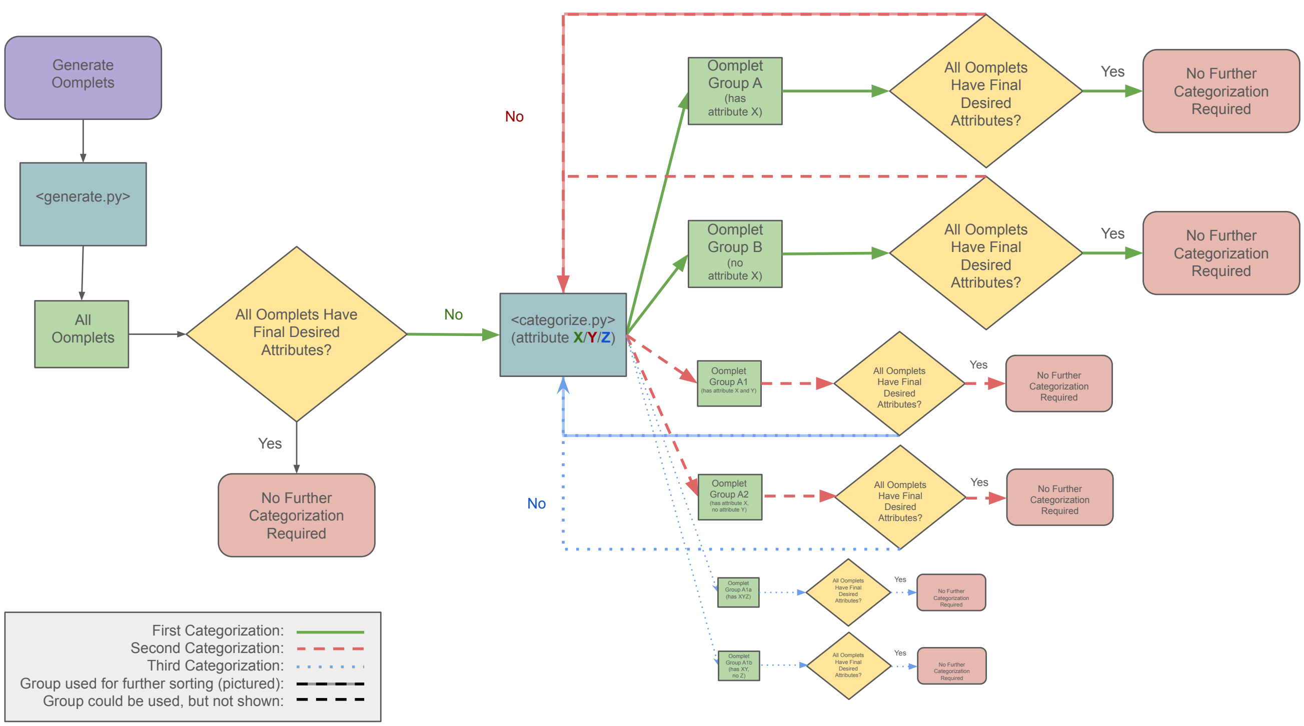

Figure 3. The categorization pipeline. Dashed lines represent potential paths while solid lines represent executed paths for one example run of the pipeline.

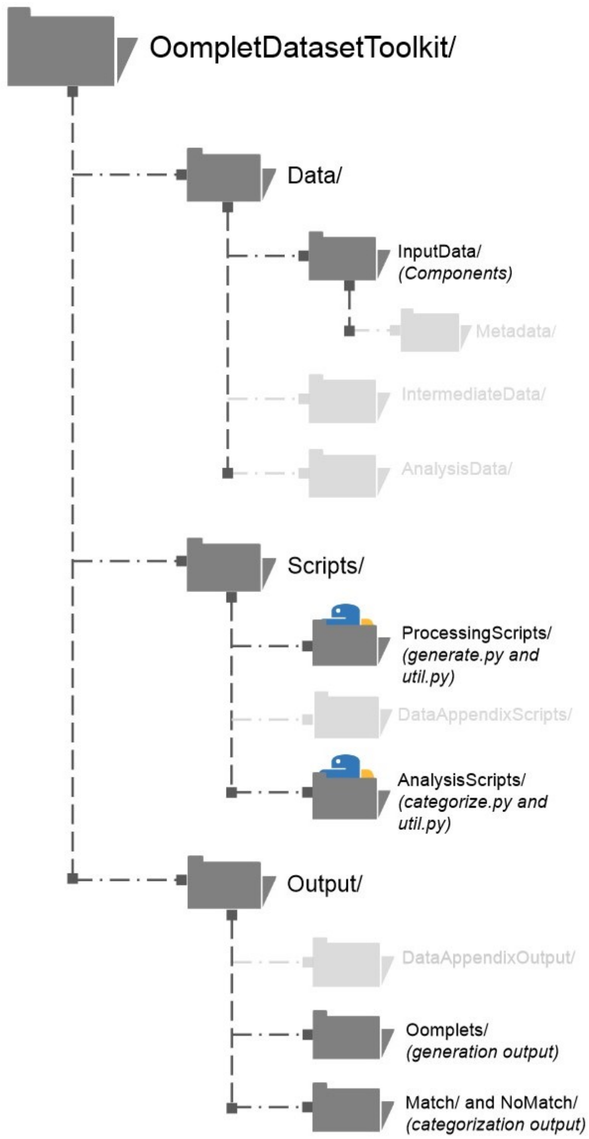

Figure 4. Directory architecture for the ODT. Light grey folders indicate standard TIER Protocol 4.0 directories that are unused.

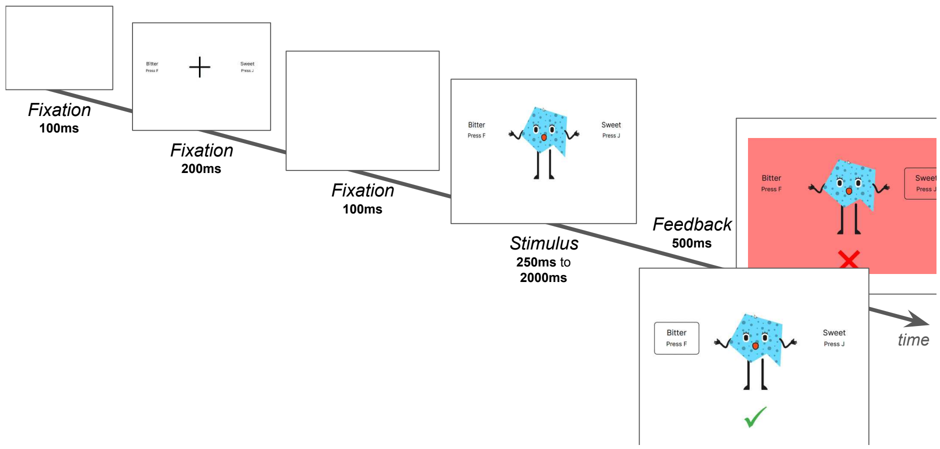

Figure 5. The Stimulus display was shown to participants for at least 250 ms, and up to 2000ms. This display would transition early to the Feedback display when participants selected a response.

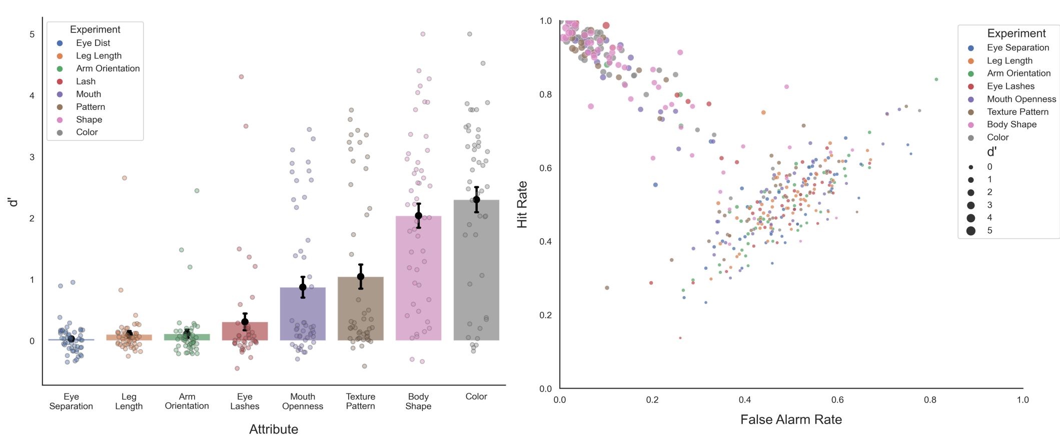

Figure 6. Left: D-Prime scores of all participants, across all experiments. Right: ROC curve including data from all eight experiments.