Contextually-adapted abstractions explain contextually-adapted abstraction in language

Anonymous CogSci submission

Abstract

XXX

Keywords: TBD

Introduction

1. Main question: Language isn't a monolith – people can choose many different ways to describe the exact same thing. What explains that variation in language? Specifically, we focus on grounded language that is about things in the world – **what explains the set of concepts people choose to convey that thing, and in particular, what determines the level of abstraction that they choose?**

2. Prior work (eg. Tian) has demonstrated that people perceive and quickly learn abstract domain structure - in particular, that they are attentive to the set of abstract concepts used to carve up the space, and that the way they parse images drawn from a generative model is consistent with the optimal concept library fit to the stimuli they observe.

3. We hypothesize that this is reflected in language – that people produce language that reflects an abstract, domain-specific concept library.

4. We adopt the more formal modeling hypothesis in (LOT, Tian, McCarthy, Wong, others): that people represent underlying domain-structure in a compositional, language-like LOT, and specifically, a domain-specific concept library with program-like structure. Therefore, our hypothesis can be evaluated by looking for correlations in the language produced and underlying programs, drawn from differing formal DSLs at different levels of abstraction.

5. To evaluate this:

   (a) We collect a corpus of grounded language in which this can be studied systematically in two domains (Section **??**), comprised of two domains with hierarchical structure in how the stimuli vary, and permitting multiple levels of abstraction - in which we have access to differing ground truth DSLs that generate the stimuli.

   (b) We first compare a base DSL of low-level parts to two increasingly higher order DSLs and find that the language people produce is better explained by a higher-order, part-based concept library (Section B.)

Part I: Domain context influences vocabulary choice

Consider how you might describe the drawing or block tower in Figure **??**A. Both images could be described in language that carves up the image, intuitively, at multiple levels of abstraction – how would you choose between describing the drawing using a simple but low-level vocabulary of basic strokes (like *lines, circles, and squares*) vs. a set of names for some of its functional parts (like *antenna* or *dial*); or between describing the block tower as composed of *red blocks* and *blue blocks* vs. of higher-level parts like a *roof* and *floors*?

Our first goal was to investigate the basic claim that people can adapt the linguistic vocabularies they use to describe visual objects like these contextually, based on the distribution of objects they intend to describe. To accomplish this, we construct two domains of stimuli (*gadgets* and *structures*) (Fig. **??**B) that are generated from a shared base set of procedural, symbolic primitives, but that vary hierarchically subdomains in terms of their compositional parts. We then run a language production experiment to elicit language from subjects familiarized with stimuli from a specific subdomain. We hypothesized that the vocabularies people used to describe each stimulus would be *context-specific*: that participants would tend to use different vocabularies tailored to the sub-domain distribution of stimuli they were shown.

Methods: Language production experiment

Hierarchical stimulus generation To investigate how the compositional structure of an object's domain affects the language people use to describe that object, we designed a collection of artificial image stimuli that were hierarchically constructed from smaller parts (Fig. **??**B).

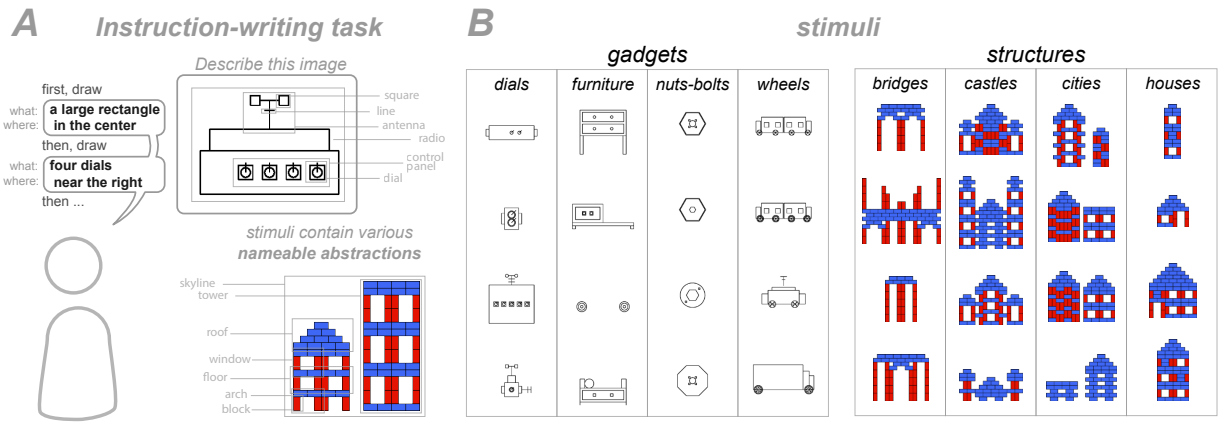


Figure 1:

Our stimulus set consisted of two distinct domains that were each procedurally generated from a set of base primitives. *Gadgets* were designed to resemble technical drawings of functional objects, and were composed of digital pen strokes. *Structures* were designed to resemble architectural structures, and were composed of horizontal and vertical blocks. Each item corresponded to a generative program involving the base primitives of its domain. Critically, each domain involved a set of mid-level abstractions (e.g. *wheels*, *roofs*) that could appear across different stimuli.

Furthermore, each domain was further subdivided into four subdomains, which were each defined by a distinct generative procedure that was hand-designed to produce objects of a recognizable subordinate category (e.g. *furniture*, *castle*) from a set of predefined abstractions (e.g. *legs*, *tower*). Several of these abstractions were shared between subdomains (*roofs*, for example appeared in both *houses* and *skyscrapers*), which allowed us to compare the language used to pick out abstraction types in different contexts. We enumerated all possible stimuli for each subdomain, and selected a random but biased sample of each to obtain 250 stimuli of varying size for each subdomain.

Procedural language production task To evoke a thorough decomposition of each object's component parts, we asked participants to produce a complete sequence of instructions for constructing each stimulus. Each participant produced instructions for 10 items from a single *subdomain* (e.g. only *castles*), either describing how to "draw" the item if describing a subdomain of *gadgets*, or how to "build" the item if it was a *structure*. To disentangle referring expressions from spatial information, we provided an interface in which participants could enter step-by-step instructions, where each step involved typing *what* would be placed/drawn *where* in separate "what" and "where" text boxes (Fig. **??**A). Participants could add as many instruction steps as they liked, and had no time limit.

To familiarize participants with the kinds of stimuli and abstraction that could appear in their subdomain, they were first required to click through images of 25 other stimuli from their subdomain, disjoint with the set to be described. While describing each image, they could also see up to 7 of the upcoming stimuli. We collected data from participants until all (2 domains * 4 subdomains * 250) stimuli had been described by at least 2 participants with complete datasets (i.e. had described all 10 of their items). Participants were paid around $15 per hour for their time.

Language preprocessing To better investigate the content of the instructions generated by participants, we use the spaCy NLP library to extract and lemmatize words.

Results

We collected 4961 sets of instructions from a total of 589 participants. Instructions produced for *castles* tended to be longer than those for *gadgets*, both in terms of the number of instruction steps provided ($b = XXX$, $t = XXX$, $p = XXX$) and the raw character counts ($b = XXX$, $t = XXX$, $p = XXX$). Nevertheless, it appeared that instructions produced for both domains spanned a range of levels of abstraction (Fig. **??** left vs. right), where some referred to lower level primitives (i.e. geometric shapes and individual blocks) and some to more abstract compositions of these lower elements (e.g. "desks", "pillars", and "castle-like structures"). Stimuli from each domain, however, were constructed from distinct sets of base primitives that can be combined according to distinct sets of constraints, allowing us to investigate whether participants systematically varied their vocabulary according

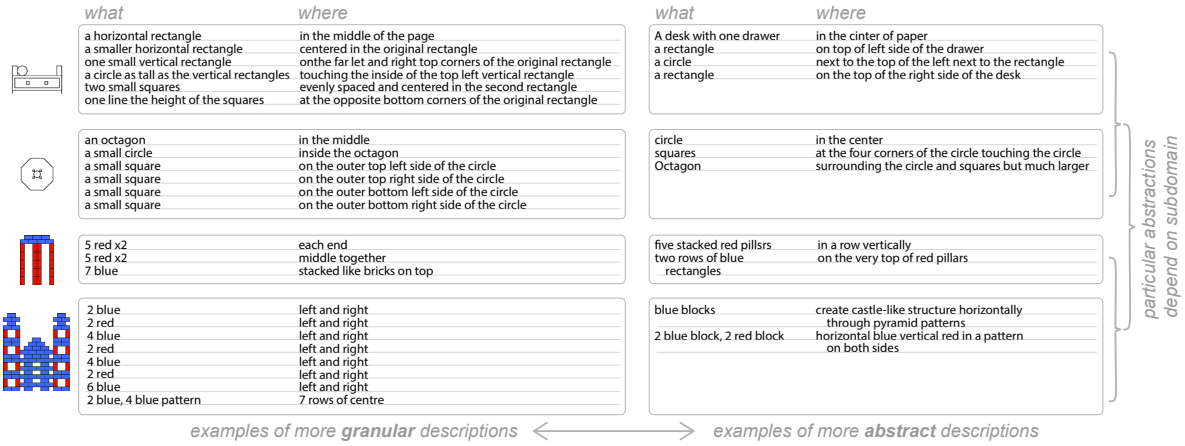


Figure 2: Examples instructions written by participants. Some, more granular descriptions (left) referred to the base primitives (strokes and blocks); others referred to higher-level constructs that combined multiple of these primitives. Differences between language can be seen between domains, but also within a single domain at the subdomain level.

to their context.

Domain-specific abstraction in language To explore any systematic differences in the vocabularies used for each domain, we calculated the TF-IDF score of each word for each domain, visualizing the most diagnostic words from domain (Fig. **??** A). This revealed a range of both lower-level ("red","blue","circle","diameter") and higher-level ("wheel","drawer","stack","pyramid") concepts that were diagnostic of each domain.

Subdomain-specific abstraction in language We also designed our stimuli to include several subdomains containing different abstractions over the base primitives used to construct items in the domain. Did the instructions produced by participants reflect these differences in subdomain structure? [todo:per-subdomain td-idf histogram. Do some subdomains have fewer distinctive (high tf-idf) words i.e. do people rely on lower-level primitives in some subdomains? (motivate upcoming sections).]

Part II: Language complexity correlates non-linearly with base primitive complexity

Each stimulus in Sec. **??** can be represented with its generating program in a base DSL of low-level primitives (*strokes* in gadgets; *blocks* in structures) shared across the domain. Before considering more abstract representations, we can first ask: do these underlying program representations at all predict the language people use to describe each stimulus? We conduct a control experiment to establish an initial correspondence between the procedural program representations in the base DSL and hu-

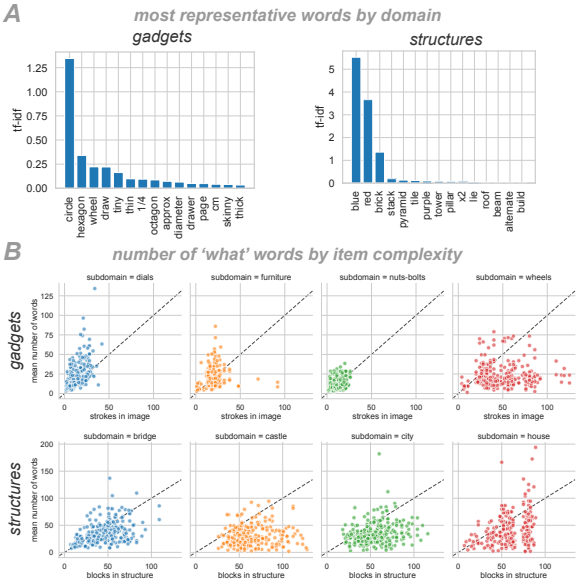


Figure 3: A) Domain-specific words used to write instructions. B) Number of words in *what* text-boxes was correlated with complexity of structures, but sub-linearly, suggesting that higher-level abstractions were used to refer to collections of primitives.

man language, by correlating *program description length* (program token counts) in the base DSL with *language description length* (language token counts.)

We hypothesized a positive, but non-linear, correlation between description lengths in the base DSL and human language: we expect that even when measured in low-level primitives, more complex stimuli should require more words to describe; but we also expect that the existence of domain-specific abstractions named in language (as established in Section I) should mean that people's language length should not scale linearly with the number of blocks or strokes in each stimulus. We found that instruction length did correlate with complexity (Fig. **??** B), and that this correlation was, in general, sub-linear. Furthermore, we found that the extent that this was true varied by subdomain, with some subdomains yielding stronger correlations between the instruction length and number of base primitives. This suggests that participants are utilizing the abstraction structure of the domain they are describing.

Part III: Contextually-adapted abstractions can better predict language production

Our results so far suggest that people adapt the vocabularies they use to the context of the subdomain they described (Part I); and that there is a positive, but non-linear, correspondence between the linguistic descriptions of each stimulus and their *generating program* in the base DSL shared across each domain. In this section, we now ask whether we can better explain the language people used in each subdomain using *contextually-adapted* DSLs containing higher-level abstractions than these base primitives.

We introduce a method for contextually varying the level of DSL abstraction used to represent the generating program for each stimulus, by enriching the initial base DSL of primitives with program abstractions that capture context-specific parametric variation within each subdomain. We also introduce a richer, model-based correlation metric between programs and language for each stimulus, based on a statistical program-to-language model fit to the language data in each domain. Using this, we first hypothesize that generative programs rewritten using *contextually-specific abstractions* should better predict language than the initial base primitives.

However, we also hypothesize that speakers take into account the underlying *vocabulary size* as a representational cost for describing the domain as a whole – a speaker incurs the cost of using an increasingly large, and increasingly arcane, vocabulary of unique concepts as they consider additional context-specific abstractions to describe any one stimulus. Our model formalizes this cost explicitly: increasing the level of DSL abstraction increases the DSL size, as more and increasingly complex program abstractions are added to the small set of initial base primitives. Concretely, we expect the correlation between programs in the base DSL and language to increase only to an intermediate level of abstraction, and then decrease, yielding a characteristic U-shaped curve indicating this tradeoff.

Methods: Identifying language abstraction with varying levels of program abstraction

Generating DSLs at varying levels of abstraction As described in Part I, stimuli in each subdomain are generated procedurally using nested levels of recursive, parametric variation: for instance, the *dials* stimuli are generated first by nesting sets of primitive shapes in the base DSL to create *dial* parts, which are themselves repeated and translated to create the *rows of dials* in many of the images. Similarly, the *house* stimuli are generated by first composing the blue and red block primitive shapes to create higher-order *window* and *brick* parts, which are tiled to create individual *floors*, which are themselves stacked to create entire *walls*.

Using the underlying procedural generative model for each subdomain, we therefore introduce contextual program abstractions into the base DSL – and re-represent the generative program for each stimulus using these abstractions – at three increasing levels of abstraction, corresponding to nested parametric variation over parts. These levels of abstraction, while driven by the underlying generative procedure, are chosen manually based on parametric depth – clearly, there are more than three possible levels of variation in each domain. Future work can investigate additional possible levels of abstraction, as well as bottom-up methods for automatically discovering these abstractions and re-writing the generative programs from their base primitives (as in [CITE Ellis]).

We choose to separately consider DSLs at each level abstraction both with and without the numeric parameters that parameterize each stimulus. In total, we consider the following DSL representations, ordered by their DSL size (the number of unique program tokens across the full subdomain):

- **Base DSL**: the low-level shared primitives (strokes and blocks) across the full domain.
- L_1: a DSL corresponding to first-order parametric variation (eg. lines rotated into polygons)
- L_1 **with parameters**: the DSL with added tokens for numeric parameters (eg. the scaling constants for individual polygons.)
- L_2: a DSL corresponding to intermediate parametric variation (eg. polygons nested into 'dials')
- L_2 **with parameters**: the DSL with tokens for numeric parameters (eg. the number of nested polygons.)

- L_3: a DSL corresponding to high-level parametric variation (eg. specific 'bases' with varying dials)
- L_3 **with parameters**: the DSL with tokens for numeric parameters (eg. the number of dials.)

Fig. **??**A shows sample visualized components from these three increasingly abstract DSLs (L_1, L_2, L_3) along with the initial shared base DSL. We also release the full generative procedure (including the abstraction regeneration) in the code repository.

Correlating program abstractions with language In Part II, we measured correspondences between program representations of stimuli and language descriptions using a simple description length metric. Here, however, we are interested in correlating specific programmatic concepts – components in a given DSL that are composed to generate each image – with particular linguistic concepts – words and phrases composed to describe each image. Therefore, we introduce a richer model-based metric based on *statistical translation models* fit to predict linguistic descriptions from generating programs.

Specifically, given a particular DSL L, we consider the *heldout predictive likelihood* $P(description|program, L)$ over heldout batches (n=5) of stimuli by fitting the program-to-language translation model to all but the heldout set of stimuli, and then evaluating the model on the heldout stimuli. We use the statistical translation model known as IBM Model 1 (?, ?), which directly estimates token-token alignments between programs and language.

Figure **??**B. shows log perplexity in the translation model (mean log likelihood of predicting language tokens from program tokens, which normalizes for length of the linguistic description) for models fit to programs in varying DSLs, ordered by their DSL size (which corresponds to increasing levels of abstraction – as discussed earlier, higher levels of contextual abstraction introduce more context-specific program components.)

Results

People adapt the abstractions they use contextually to the subdomain. We hypothesized that higher-level DSLs, containing context-specific abstractions, would always better predict language in each subdomain. Our results in Fig. **??**A suggest that this is generally true, along with a more nuanced interpretation: in *most* of the subdomains, a contextual DSL improves perplexity under the translation model as compared to the base DSL. However, in two of the structures subdomains – the *cities* and *castles* domain, in fact the base DSL yields the best perplexity.

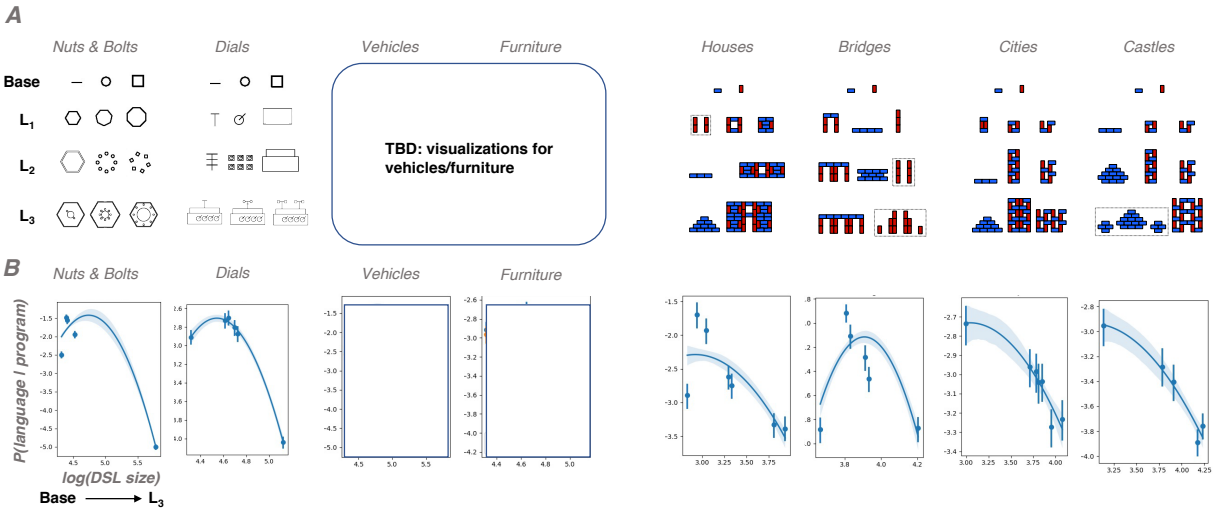
This result suggests that people do indeed adapt the level of abstraction they use in language, dependent contextually on the subdomain of stimuli – after all, while the base DSL can be used to describe *every* subdomain (eg. people could *always* have described each structure in terms of its low-level blocks), people seem to have chosen it selectively for some domains and other, more context-specific abstractions for others.

What explains when people fall back on the base DSL of primitives? A visual inspection of the *cities* and *castles* abstractions suggests one intuitive explanation: the availability of commonly-understood linguistic terms to describe these abstractions. While our results in many domains suggest that people flexibly pick out higher-level, contextual abstractions – and adapt their vocabulary to reflect them – humans performing a naive procedural description task, intended for other naive speakers, are also constrained by the basic English terms available to them. We see these results as especially promising for ad-hoc *convention formation* paradigms [CITE], to determine whether subjects can *further* adapt their language to a context with additional joint experience.

People generally choose an intermediate level of abstraction. Our secondary hypothesis suggests a tradeoff between contextual-abstraction – which reduces the cost of describing any given stimulus in a subdomain – and vocabulary size, modeled by the size of each enriched DSL. Our results in Fig. **??**A support this conclusion, finding a characteristic U-shaped curve for the domains where more abstract DSLs better predict language: perplexity in the translation model does not increase monotonically with abstraction level (and DSL size.)

However, as with the previous finding, this result suggests a promising avenue for future work to disentangle the cause of this trend: does this curve reflect an individual choice on the part of the speaker (to take into account the cost of a larger vocabulary), or a limit in the contextual abstraction afforded by language intended for naive listeners (which permits some variation in abstraction level, but may not contain sufficiently interpretable terms as abstrations grow more context specific)? Again, we see this as an especially promising route for considering language between paired speakers in an extended joint conversational context.

Discussion


A
Nuts & Bolts
Dials
Vehicles
Furniture
Houses
Bridges
Cities
Castles
Base
L1
L2
L3
TBD: visualizations for vehicles/furniture
B
Nuts & Bolts
Dials
Vehicles
Furniture
P(language | program)
log(DSL size)
Base
L3