# MetaCOG: Learning a metacognition to recover what objects are actually there

**Marlene Berke**
Department of Psychology, Yale University
New Haven, CT, USA.
marlene.berke@yale.edu

**Zhangir Azerbayev**
Department of Psychology, Yale University
New Haven, CT, USA.
zhangir.azerbayev@yale.edu

**Mario Belledonne**
Department of Psychology, Yale University
New Haven, CT, USA.
mario.belledonne@yale.edu

**Zenna Tavares**
Zuckerman Institute and Data Science Institute
Columbia University New York, NY, USA
& Basis, New York, NY, USA
zt2297@columbia.edu

**Julian Jara-Ettinger**
Department of Psychology, Yale University
New Haven, CT, USA.
julian.jara-ettinger@yale.edu

## Abstract

Humans do not unconditionally trust what they see, but instead use their meta-cognition to recognize when a percept might be unreliable or false, such as when we realize that we mistook one object for another. Inspired by this capacity, we propose a formalization of meta-cognition for object detection and we present MetaCOG, an instantiation of this approach. MetaCOG is a probabilistic model that learns, without supervision, a meta-cognition for object detection systems and uses this meta-cognition to refine beliefs about the locations and semantic labels of objects in a scene. We test MetaCOG's performance when paired with a variety of modern neural object detectors (Exp 1), and when paired with simulated object detectors that enable us to test its robustness under different conditions (Exp 2). We find that MetaCOG can quickly learn an accurate meta-cognitive representation of object detectors and use this meta-cognition to infer the objects in the world responsible for the detections.

## 1 Introduction

Building accurate representations of the world is critical for prediction, inference, and planning in complex environments [1]. In human vision, these representations are generated by a perceptual system that transforms light entering the retina into 3D representations of the physical space and the objects in it [2, 3]. While human perception is generally robust and reliable, it nonetheless suffers from errors, such as when we confuse one object for another, or when we experience a visual illusion. In these situations, people use their meta-cognitive representation—their *meta-cognition*—of the workings of another cognitive system [4] to help them recognize that their visual system has failed them and adjust their representations accordingly.

End-to-end object recognition models face a similar challenge: how can we identify when a non-existent object is incorrectly detected (hallucinations), or when a real object is not reliably recognized

Preprint. Under review.

(misses)? Traditional approaches to this problem have focused on increasing the training data [5, 6] or improving the model's architecture [7, 8]. While valuable, these methods continue to be prone to errors, particularly when confronted with data that was not adequately captured in training [9]. We propose that this problem can be reduced by augmenting object detection models with a meta-cognitive system that represents the object detector's behavior.

The central component in our characterization of a meta-cognition is a joint distribution expressing the relationship between the object detections produced by a detector, and the true objects in a scene. This distribution therefore represents the object detector's performance, which helps determine which object detections are trustworthy, and which should be questioned or dismissed. The challenge therefore lies in learning an accurate meta-cognitive representation of the object detector.

Here we propose a formulation that can learn a meta-cognition of an object detector without any access to its internal architecture or performance metrics and without any human annotation or feedback. Given a pre-trained object detector and a dataset of images (partitioned into scene-specific sets) and corresponding viewpoints, our approach performs joint inference over the objects truly in a scene and outputs of the object detector. To construct an accurate meta-cognition we draw insight from cognitive science. Infants come into the world equipped with a basic form of 'core knowledge' [1, 10], grounding their visual experience in three-dimensional representations of the world constrained by *Spelke principles* [11, 12]: objects persist through time and move continuously through space. By grounding percepts on a 3D representation with object permanence, our model, MetaCOG (Meta-Cognitive Object-detecting Generative-model), can learn an accurate meta-cognitive representation of an object detector, and use it to refine its inferences about the contents of scenes.

We evaluate MetaCOG in two experiments that evaluate its ability to 1) infer an accurate representation of the detector's performance and 2) use this meta-representation to identify and correct missed or hallucinated objects. In Experiment 1, we explore MetaCOG's performance when paired with three modern object detection networks, testing MetaCOG on a dataset of images of scenes rendered in the ThreeDWorld (TDW) virtual environment [13]. In Experiment 2, we explore the MetaCOG's tolerance to faulty inputs.

Our work makes two main contributions. First, we propose a novel formalization of meta-cognition in the context of object detection, and an inference approach where a meta-cognition can be learned without feedback by conditioning inferences on cognitively-inspired Spelke principles. Second, we present MetaCOG, an instantiation of this model that can learn a meta-cognition for object detection models. We show that MetaCOG can quickly learn an accurate meta-cognition of an object detection model without feedback and use it to make better inferences about what objects are where.

## 2 Related work

**Meta-cognition in AI.** Previous work has shown promise for the use of meta-cognition in improving classification accuracy [14, 15]. While previous work focused on engineering an inflexible meta-cognition to guide learning, we focus on a complimentary problem: learning a meta-cognition without feedback for monitoring a neural network's detections.

**Computational cognitive science.** Our work is also related to computational models of human core knowledge [12, 16]. The difference is that our work uses object principles to learn a meta-cognition, whereas past work has focused on modeling object principles themselves.

**Uncertainty-aware AI.** Our work also relates to uncertainty-aware AI. This work focuses on building end-to-end systems that express uncertainty in their inferences [17, 18, 19]. Our work focuses instead on how to learn a model of uncertainty over a pre-trained model. These two approaches complement each other. In humans, meta-cognitive uncertainty supplements the intrinsic uncertainty in visual perception.

## 3 MetaCOG

Figure 1 shows a conceptual schematic of MetaCOG and the domain that we apply it to. Here, a set of images (Fig. 1A) are taken from the a single scene and then processed through an object-detector to yield *detector observations* (Fig. 1B). The goal is to simultaneously infer 1) a representation of the

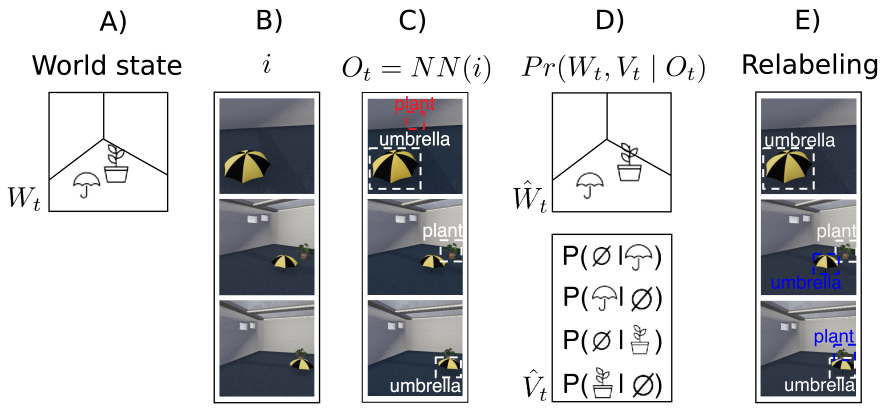


Figure 1: Conceptual schematic of MetaCOG in the context of Experiment 1. A) A world state $W_t$ (consisting of an umbrella and a potted plant in a room). B) Images $i$ taken from three viewpoints sampled from a camera trajectory. C) Detections $O_t$ output by an object detection network applied to those images. D) MetaCOG's inferences, estimating the world state $\hat{W}_t$ (objects with semantic labels and locations in 3D space), and a meta-cognition $\hat{V}_t$ (probabilities of missing or hallucinating objects of each category). E) Re-labeling the images based on MetaCOG's inferences, which can then be used to re-train the neural network. Throughout the panels, correct detection are illustrated in white, hallucinations in red, and MetaCOG's filling in of previously missed objects in blue.

object detector's performance (called the *meta-cognitive representation* or *meta-cognition* for short; Fig. 1C), and 2) what objects are actually in the scene (the *world state*).

This meta-cognition is characterized by two category-specific probability distributions: one which represents a belief over the number of hallucinated objects (detected, but not actually there) of each category, and another which represents a belief over the number of missed detections (not detected, but actually there) of each category. We refer to these collectively as the $V$-structure as it expresses a belief about the veracity of the object detector.

MetaCOG consists of a hierarchical generative model with two levels (see Appendix, Figure 4). The *lower* level captures a joint distribution between world states and object detections that is dependent on $V$. The *higher* level describes the dynamics of the prior over $V$, capturing the idea that a meta-cognitive representation can—and should—change with experience. We begin by presenting this generative model (3.1) and then turn to how we perform joint inference over the world state and the meta-cognitive representation (3.2).

### 3.1 Generative model

#### 3.1.1 Object Detection Module

The goal this paper is to present an approach for buidling a meta-cognitive representation of an object detection model that transforms images into object detections, each detection consisting of a category and a position in the 2D image. We assume access to the camera's state (its position and orientation) for each image, which helps with inferring and tracking the location of different objects (as this is not the main focus of our work). We additionally assume that images are grouped into sets captured from the same world, but do not assume access to the underlying world state itself. Finally, the object detector is a black box—MetaCOG does not have any access to its internal state or knowledge about its performance.

Formally, let us assume as given a set $\vec{O} = (O_1, \ldots, O_T)$ of $T$ detector observations. Each observation $O_t = (NN(i_1), NN(i_2), \ldots)$ is a collection of detections output from a given object detector $NN$, which maps an image to detected objects and and their corresponding positions. Specifically, $NN(i)$ is a collection $((x_1, y_1, c_1), (x_2, y_2, c_2), \ldots)$ where $x_i$ and $y_i$ are pixel-coordinates and $c \in \mathbb{C}$ is an object category.

### 3.1.2 World-state representation and Spelke principles

The images corresponding to each observation $O_t$ are assumed to be snapshots of an underlying world state $W_t \in \mathbb{W}$. A world state is represented as a collection objects, each a tuple of the form $(x, y, z, c)$ where $c \in \mathbb{C}$ is an object category and $x$, $y$ and $z$ are coordinates in 3D space. Correspondingly, we denote by $\vec{W} = (W_1, \ldots, W_T)$ the collection of world states.

In our approach, Spelke principles are expressed as a prior over world states. The assumption of object permanence implies that each world state has a fixed set of objects that does not change. Next, the assumption that two objects cannot occupy the same physical position is implemented as a prior over the locations of the objects in a world state. Finally, the assumption of spatio-temporal continuity implies that any moving object must exhibit a continuous trajectory. Our Experimental task focuses on inferring permanent fixtures in a room, so we therefore do not implement this constraint, although doing so is a solved problem [12, 16]. Implementation details are available in A.1.3.

### 3.1.3 Meta-cognitive representation and dynamics

The object detection module is represented as two probability distributions per object category. The first distribution captures the model's propensity to hallucinate as a Poisson distribution with rate $\lambda_c$, representing a belief over the number of hallucinations of objects of category $c$ will be hallucinated in a given frame. The Poisson distribution was chosen to represent hallucinations as we assume that hallucinations are independent of each other and randomly distributed within and across images. The second distribution captures the model's propensity to correctly detect an object that's in view. We represent this through a Geometric distribution with rate $p_c$ encoding a belief over the number of times an object of category $c$ will be detected when it is present in the image. Conveniently, under this formulation, the model's miss rate for an object in category $c$ is $1 - p_c$. Because each distribution is captured by a single parameter, the parameters of the probability distributions in the meta-cognition $V$ can then be represented as a pair of vectors of length $|C|$ storing each category's hallucination rate $\lambda_c$ and miss rate $1 - p_c$. We call this pair of vectors $\theta$.

The generative model described so far captures how MetaCOG represents the object detector's performance. However, an object detector's propensity to hallucinate or miss objects can vary across scenes, so leaving adjustable flexibility in $V$ is a desirable property. At the same time, experience in a previous scene includes critical information about the object detector that should inform expectations about its performance in a new scene. Our generative model therefore includes an evolving kernel $G_t$ (Figure 4), capturing changing priors over the parameters $\theta$ for the probability distributions $V$. See A.1.1 for details. For notation simplicity, $V$ will refer to $V|\theta$, and $V_t$ to $V_t|\theta_t$. We denote by $\vec{V} = (V_1, ..., V_T)$ the collection of probability distributions corresponding to each world state in $\vec{W}$.

## 3.2 Inference

Given a collection of observations $\vec{O}$ (multiple sets of detections from multiple world states) and the corresponding camera trajectories $\vec{cj}$, our goal is to infer $Pr(\vec{V}, \vec{W}|\vec{O}, \vec{cj})$, given by

$$Pr\left(\vec{V}, \vec{W} \mid \vec{O}, \vec{cj}\right) \propto \prod_{t=1}^{T} Pr\left(O_t|W_t, V_t, cj_t\right) Pr(V_t) Pr(W_t) \tag{1}$$

The posterior, Eq. 1, is approximated via Sequential Monte-Carlo using a particle filter. The details of the algorithms used for inference are described in A.1.4.

An estimate of the joint posterior can be sequentially approximated via:

$$Pr\left(\vec{V}, \vec{W}|\vec{O}, \vec{cj}\right) \approx Pr\left(\hat{V_0}^0\right) \prod_{t=1}^{T} Pr\left(O_t|\hat{V_t}^t, \hat{W_t}^t, cj_t\right) Pr\left(\hat{W_t}^t\right) Pr\left(\hat{V_t}^t|\hat{W_{t-1}}^t, \hat{O_{t-1}}^t\right) \tag{2}$$

where $\hat{W_1}^T, \ldots, \hat{W_T}^T$ is the estimate $W_1, \ldots, W_T$ after $T$ observations, and $\hat{V_1}^T, \ldots, \hat{V_T}^T$ is the estimate $V_1, \ldots, V_T$ after $T$ observations. Here the transition kernel, $Pr(\hat{V_t}^t | \hat{W_{t-1}}^t, \hat{O_{t-1}}^t)$ is governed by the belief dynamics described in A.1.1.

### 3.2.1 Estimating $V$

After all $T$ time steps in the particle filter, we estimate $V_T$ by taking the expectation of the marginal distribution by averaging across particles weighted by their likelihood $l$: $\hat{V}^T_{T,\mu} = E[\hat{V}^T_T | \vec{O}] = \frac{1}{M} \sum_{m=1}^{M} (\hat{V}^T_{T,m} * l_m)$, where $m$ indexes the particles. This $\hat{V}^T_{T,\mu}$ is the final estimate of the belief about the true $V$ after all observations have been observed. Given $\hat{V}^T_{T,\mu}$, the posterior predictive distribution is defined as:

$$Pr\left(\hat{\vec{W}} | \vec{O}, \vec{cj}, \vec{V} = \hat{V}^T_{T,\mu}\right) \propto \prod_{t=1}^{T} Pr\left(O_t | \hat{W}_t, cj_t, V_t = \hat{V}^T_{T,\mu}\right) Pr\left(\hat{W}_t\right) \tag{3}$$

This posterior predictive distribution can then be used to make better inferences about world states for novel scenes, $W_{T+1}, ...$ or for reassessing previous world states $W_1, ..., W_T$, that were originally inferring using a less informed $V$.

## 4 Experiments

Our experiments have two goals: first, to test whether MetaCOG can learn an accurate meta-cognition of an object detector, and second, to explore whether this learning confers any benefits to the system's overall accuracy. Experiment 1 tests MetaCOG's performance when processing outputs of three popular object detection systems (Section 4.1). Experiment 2 presents an analysis of how MetaCOG performs as a function of an object detector's baseline performance (Section 4.2). Throughout, we evaluate MetaCOG by sampling scenes (arrangements of objects in a room), and then sampling a collection of viewpoints from each scene (with the collection of viewpoints from a scene called a video).

### 4.1 Experiment 1: Enhancing neural networks for object detection with a meta-cognition

**Object detection models** To test MetaCOG's capacity to learn and use a meta-cognition, we tested its performance when paired with three modern neural networks for object detection. These networks represent three popular architectures: RetinaNet, a one-stage detector [20]; Faster R-CNN, a two-stage detector [21]; and DETR, a vision transformer [7], all pre-trained (see A.3.1). See A.3.2 for details for on post-processing. A couple of the networks were validated to ensure that their baseline performance on our dataset was within their expected range (see A.3.3).

**Dataset** We evaluate MetaCOG on a dataset rendered in the ThreeDWorld physical simulation platform [13] using one canonical object model per category (See A.2.1 for details). First, 100 scenes were generated by sampling objects and placing them in a room with carpeting and windows (Fig 1a). For each scene, we then generated a video by sampling 20 frames from an agent's trajectory moving around the room. See A.2.1 for details. The final set of 100 videos was then randomly split into a training and test set (each with n=50 videos). To avoid order effects, all reported results show averages across four different video orders (See A.2.3).

**Comparison Models** We compare MetaCOG to two baselines in order to test if the learned meta-cognition improves performance. First, we compare MetaCOG's inferences against the processed neural network output without meta-cognition. To isolate the contribution of MetaCOG, we use the exact output that served as input to MetaCOG (see A.3.2 for details).

The computational structure of MetaCOG might improve accuracy without meta-cognition mattering per se. For instance, mapping detetctions to 3D representations with object permanence alone might improve accuracy. Alternatively, the meta-cognitive representation might provide tolerance for rejecting sparse detections or interpolating missing detections on frames, but the exact learned meta-cognition (i.e., the precise values of the parameters $\theta$ in $V$) might not matter. To test these

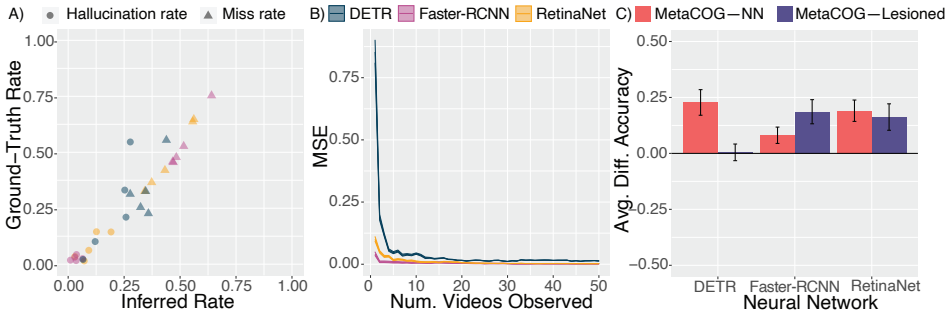


Figure 2: Results for MetaCOG and comparison models. A) Scatterplot showing MetaCOG's inferred values for the hallucination rates (circles) and miss rates (triangles) against the ground-truth values. Blue-grey codes for DETR, magenta for Faster-RCNN, and yellow for RetinaNet. B) The MSE of MetaCOG's inferences about $\theta$ as a function of number of videos observed, color-coded by NN. Shading depicts 95% CIs. C) Accuracy difference between MetaCOG and the two baseline models, the NN's output (red) and Lesioned MetaCOG (purple). Positive values indicated MetaCOG outperforms the comparison models.

Table 1: Accuracy results for Experiment 1. Values in parentheses represent 95% bootstrapped confidence intervals.

| Object Detector | Model | Acc. Training | Acc. Test |
|---|---|---|---|
| **RetinaNet** | MetaCOG | **0.721** (0.653, 0.785) | **0.726** (0.664, 0.783) |
| | NN Output | 0.561 (0.518, 0.604) | 0.535 (0.499, 0.572) |
| | Lesioned MetaCOG | 0.639 (0.568, 0.709) | 0.564 (0.502, 0.626) |
| **FasterR-CNN** | MetaCOG | **0.790** (0.727, 0.846) | **0.766** (0.709, 0.821) |
| | NN Output | 0.750 (0.698, 0.799) | 0.686 (0.635, 0.736) |
| | Lesioned MetaCOG | 0.658 (0.589, 0.728) | 0.581 (0.520, 0.642) |
| **DETR** | MetaCOG | 0.474 (0.425, 0.523) | **0.438** (0.374, 0.507) |
| | NN Output | 0.211 (0.186, 0.238) | 0.211 (0.185, 0.238) |
| | Lesioned MetaCOG | **0.483** (0.423, 0.546) | 0.433 (0.369, 0.497) |

possibilities we also compare our results against a *Lesioned MetaCOG*, where the values of $\theta$ were set to the mean of the initial prior over $\theta$ (see A.5.3). Although this model has a meta-cognitive representation $V$, that the parameters $\theta$ of that meta-cognition are neither learned nor updated in light of new observations. This lesioned model contains the same representations and parameters as the full model and therefore serves as a control for model complexity.

**Results** We first assessed MetaCOG's ability to learn an accurate meta-cognitive representation $V$ (see A.4.2 and A.4.1 for operationalization). Figure 2A shows the relationship between the parameters in MetaCOG's inferred $\hat{\theta}$ and each network's true $\theta$, with an overall correlation of $r = 0.951$. Fig 2B shows the learning trajectory, visualizing the MSE of $\hat{\theta}$ for each network as a function of the number of observed videos. After the 50-video training, MSEs decreased, on average, by 97.4% of their initial values, and had already decreased by 93.5% after only 20 videos. This demonstrates that MetaCOG can learn an accurate meta-cognition $V|\theta$ with access to a relatively small number of scenes, all without access to ground-truth world states.

Table 1 and Figure 2C show MetaCOG's accuracy relative to our comparison models (see A.8 for operationalization). MetaCOG outperformed the two comparison models on the test set for all three neural networks. On average, MetaCOG increased accuracy by 16.6% relative to the NN outputs in the test set. Additionally, MetaCOG increased accuracy by 11.7% relative to lesioned MetaCOG, confirming that MetaCOG's success can be partially attributed to its learned meta-cognition, rather than purely to the Spelke principles (see A.5.1 for additional results showing that MetaCOG's inferences about the presence and locations of objects at the level of 3D scenes rather than 2D images also outperform those of Lesioned MetaCOG).

Table 2: Faster-RCNN re-trained on MetaCOG's inferred labels

| Model | Acc. Training | Acc. Test |
|---|---|---|
| Pre-trained NN | 0.750 (0.698, 0.799) | 0.686 (0.635, 0.736) |
| NN re-trained with MetaCOG | **0.843** (0.813, 0.871) | **0.812** (0.789, 0.835) |

While MetaCOG and Lesioned MetaCOG were matched in architecture and computational complexity, MetaCOG and NN Output were not. This raises the possibility that MetaCOG's success relative to the neural network's outputs is due to the additional structure, computational complexity, and access to camera trajectories. To create a like-to-like comparison, we ran a secondary test where we train Faster-RCNN (initialized to its pre-trained weights) using MetaCOG's inferences. That is, because MetaCOG infers what is where on each frame, these inferences can be treated as new synthesized training set that can be used to train the neural network (all done without any human annotation; see A.5.2 for details on the training procedure). Faster-RCNN trained on MetaCOG's inferences performed similarly to MetaCOG and better than its baseline performance before training (Table 2). Beyond controlling for computational complexity, this results also demonstrates MetaCOG's potential to become a self-supervised learning system, and serves as a proof-of-concept that MetaCOG can improve an object detection system.

### 4.2 Experiment 2: Robustness analysis

The results of Experiment 1 show that MetaCOG can quickly learn a meta-cognition for an object detector and use it to improve accuracy. This provides proof-of-concept that MetaCOG can be applied to modern neural network object detection systems. However, the performance of these three networks does not capture the full space of possible object detection performances. How robust is MetaCOG to different degrees of faults in its inputs? To evaluate MetaCOG's robustness, Experiment 2 tests MetaCOG paired with a large range of possible object detectors. To achieve this, we created a synthetic dataset of processed videos passed through simulated faulty object detectors. To reduce the computational cost, these world states and observations were no longer scenes and images with detections, but an abstraction of this process. Similarly, to reduce computational costs and to focus on the contribution of meta-cognition rather than 3D representations, we used a lightweight, general-purpose version of MetaCOG (see A.6) to infer a meta-cognition of the synthetic object detectors and the world states causing the observations.

**Dataset** The dataset consisted of world states passed through simulated object detectors with different degrees of faulty performance. To focus on the contribution of meta-cognition for determining the presence or absence of different object categories, the dataset consisted of hypothetical collections of objects with no spatial information. Specifically, we sequentially sampled world states (vectors of 1s and 0s indicating the presence or absence of five possible object categories), object detectors (*miss* and *hallucination* distributions $V$ with a wide variety of parameters $\theta$), and then generated a set of observations passed through the probabilistic artificial object detector. The final dataset consisted of 40000 randomly sampled object detectors, each processing 75 world states, with 5-15 processed frames per world state. The details of synthesizing this dataset are left to A.7.

**Comparison Models** As in Experiment 1, we compare MetaCOG to Lesioned MetaCOG, which fixes the parameters $\theta$ of the meta-cognitive representation $V$ to the expectation of the prior over $\theta$ (See A.6 for details on the prior over $\theta$, and A.9 for details on Lesioned MetaCOG). This enables a fair comparison where the two models are matched for computational complexity. Models are described more formally in A.9.

**Results** Here, we demonstrate that, as in Experiment 1, MetaCOG can learn a system's hallucination and miss rates without feedback, and that its improvements in accuracy track with its learning of a meta-cognition. See A.8 for definitions of the metrics used.

Figure 3 shows MetaCOG's performance over the 40000 sampled object detectors. To assess MetaCOG's ability to learn an accurate meta-cognitive representation $V|\theta$, we examined the MSE as a function of the number of observed videos. Figure 3A shows that MetaCOG's estimates $\hat{\theta}_t$ of the hallucination and miss rates rapidly approach the true values $\theta$, with a final average MSE of 0.00166.

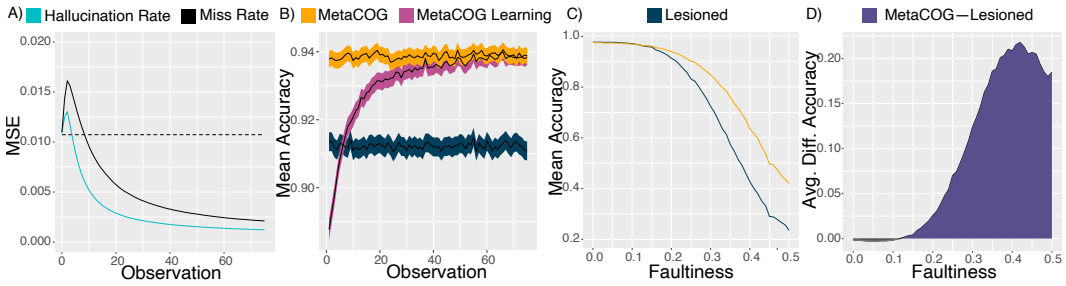


Figure 3: Model performance over 40000 sampled object detectors, each processing 75 world states. **A)** MSE between true and inferred hallucination and miss rates as a function of number of observations. Horizontal dotted line represents the average MSE for the mean of the prior (the $\theta$ used in Lesioned MetaCOG). **B)** Mean accuracy as a function of number of observations. **C)** Accuracy for MetaCOG and Lesioned MetaCOG as a function of faultiness ($\zeta$) in the observation. **D)** Difference in average accuracy between MetaCOG and Lesioned MetaCOG as a function of faultiness ($\zeta$).

To test if learning varied as a function of the object detector's errors, we next calculated MSE as a function of the detector's faultiness $\zeta$, defined as the detector's average proportion of errors per scene (see A.8 for formal definition). A linear regression predicting MSE as a function of faultiness revealed that MetaCOG's learned $\hat{\theta}$ is less accurate for faultier detectors ($\beta = 0.002$; $p < 0.001$), although the effect was minimal (predicting a MSE increase of $0.001$ from a perfect detector with faultiness 0 to a detector with faultiness 0.5, where detections are equally likely to be true or false). Together, these analyses confirm that the results of Experiment 1 hold for a wider set of object detector performances.

Figure 3B shows the average accuracy of the models' inferences about world states. The magenta line shows MetaCOG's rapid increase in accuracy over the first 40 observations. During these observations, MetaCOG's accuracy increases from 88.8% to 93.6% (with 93.8% final accuracy after the 75th observation). This increase in accuracy occurs simultaneously with the decline in MSE for the inferred meta-cognition's parameters. By comparison, the blue line shows lesioned MetaCOG's performance, which showed an average performance of 91.2%. The orange line shows MetaCOG's accuracy after conditioning on the parameters $\hat{\theta}_T$ of meta-cognition learned after observing all the videos (see A.9 for details).

To quantify MetaCOG's abilty to make accurate inferences under noise, we next computed accuracy as a function of the object detector's faultiness $\zeta$ on a range from 0 (perfect performance) to 0.5 (detections equally likely to be true or false). Figure 3C shows MetaCOG's and Lesioned MetaCOG's mean accuracies as a function of $\zeta$. When faultiness is low, MetaCOG and Lesioned MetaCOG performed near ceiling, as the object detector's output is already highly reliable. However, MetaCOG outperformed Lesioned MetaCOG in accuracy for detectors with faultinesses in $\zeta \in [0.12, 0.5]$.

Figure 3D shows the difference in average accuracy between MetaCOG and Lesioned MetaCOG as a function of faultiness. MetaCOG reaches its highest accuracy boost over Lesioned MetaCOG at faultiness level 0.42 (with a 21.8% improvement), but shows a consistent performance boost across faultiness values. Together, these results show that MetaCOG's success in Experiment 1 was not limited to the particular performance values of the neural networks used. Instead, MetaCOG can learn a correct meta-cognition for a wide range of object detectors and use it to improve accuracy.

## 5 Discussion

Here we proposed a formalization of meta-cognition for object detection which can be learned without human feedback and used to improve a system's inferences about the category and location of objects. We presented an implementation of this idea, MetaCOG, a model that learns a probabilistic relation between the detections produced by an object detection system and the true objects in the scene. Given a set of detections from an object detection model with unknown performance, MetaCOG performs joint inference over a meta-cognitive representation of the object detector and over the objects causing the detections. This joint inference is made possible by grounding priors in *Spelke object principles*. Applying MetaCOG to three modern object detection neural networks (Experiment

1) and to simulated object detectors (Experiment 2) showed that MetaCOG can quickly learn an accurate meta-cognition and use it to correctly infer the objects in a scene.

While our results show improvement in object detection accuracy, our formulation requires additional information not usually included in object detection tasks. Our formulation requires access to camera position and to scene boundary information (i.e., knowledge of when world state has changed). In principle, these assumptions could be relaxed by extending the inference procedure to also recover camera position and event boundaries. While this is a relatively straightforward extension, these additional degrees of freedom could leave the problem of inferring an accurate meta-cognition too under-determined for MetaCOG. On the other hand, our formulation might be particularly suitable for embodied agents that, by tracking their movements in space, can determine camera position and scene boundaries (e.g., resetting the scene when crossing through a doorway into a new room).

Beyond the problem of categorizing and locating objects in space, our approach may be useful in related domains. In the most general form, our approach requires two things: 1) some assumptions to enable disentangling which parts of a signal reflect the external state of the world and and which parts reflect the faults of the system (e.g., in our model, using Spelke principles to determine which detections revealed objects in the world, and which ones revealed errors in the neural network) and 2) a proposed meta-cognitive representation (e.g., in our model, the distributions of hallucinations and misses). For instance, our MetaCOG approach could be applied to the problem of face-recognition, perhaps by 1) assuming the object principle that a person cannot be in more than one place at the same time, and 2) a proposed meta-cognitive representation of a confusion matrix across face features. This could allow a MetaCOG model to learn the recognition system's biases, and perhaps to enforce an expectation that all faces should be equally discriminable.

This idea also highlights an important limitation of our current approach: the structure of the meta-cognitive representation is handcrafted to fit the problem. (In our case, the meta-cognitive representation was handcrafted to represent the distribution of hallucinations and misses. MetaCOG then learned the parameters of the distributions without supervision, but not the structure of the representation itself.) This means that extending our approach to new domains requires human intervention to determine what meta-cognitive representation to use. This limitation could be addressed by extending the inference procedure to search over a space of possible representations for expressing meta-cognition, using a method like the one in [22]. The ability to change the representation of the meta-cognition might also be further useful for the object detection problem that we considered. For instance, our representation captured hallucinations and misses at the category-level (which we tested by using a single exemplar per category). When an object detection system shows within-category variability (e.g., has a high chance of missing a lawn chairs but not desk chairs), breaking the category-level representation into sub-categories could be fruitful.

Finally, our work focused on learning a meta-cognition for the purpose of improving accuracy, but the abstract meta-cognitive representation also has potential benefits for interpretability. In our implementation, MetaCOG's $V$ captures the object detector's performance in a way that is easy for humans to understand. As such, this approach of learning a meta-cognition may also be able to support explainable AI by generating simplified meta-representations of a model's performance. We hope to explore this possibility in future work.

## References


[1] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

[2] Kohitij Kar and James J DiCarlo. Fast recurrent processing via ventrolateral prefrontal cortex is needed by the primate ventral stream for robust core visual object recognition. *Neuron*, 109(1):164–176, 2021.

[3] Umut Güçlü and Marcel AJ van Gerven. Increasingly complex representations of natural movies across the dorsal stream are shared between subjects. *NeuroImage*, 145:329–336, 2017.

[4] Thomas O Nelson. Metamemory: A theoretical framework and new findings. In *Psychology of learning and motivation*, volume 26, pages 125–173. Elsevier, 1990.

[5] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing.

[8] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[10] Susan Carey. *The origin of concepts*. Oxford university press, 2009.

[11] Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

[12] Kevin Smith, Lingjie Mei, Shunyu Yao, Jiajun Wu, Elizabeth Spelke, Josh Tenenbaum, and Tomer Ullman. Modeling expectation violation in intuitive physics with coarse probabilistic object representations. In *Advances in Neural Information Processing Systems*, pages 8983–8993, 2019.

[13] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020.

[14] Giduthuri Sateesh Babu and Sundaram Suresh. Sequential projection-based metacognitive learning in a radial basis function network for classification problems. *IEEE transactions on neural networks and learning systems*, 24(2):194–206, 2012.

[15] Kartick Subramanian, Sundaram Suresh, and Narasimhan Sundararajan. A metacognitive neuro-fuzzy inference system (mcfis) for sequential classification problems. *IEEE Transactions on Fuzzy Systems*, 21(6):1080–1095, 2013.

[16] Charles Kemp and Fei Xu. An ideal observer model of infant object perception. In *Advances in Neural Information Processing Systems*, pages 825–832, 2009.

[17] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems*, pages 3179–3189, 2018.

[18] Lance Kaplan, Federico Cerutti, Murat Sensoy, Alun Preece, and Paul Sullivan. Uncertainty aware ai ml: why and how. *arXiv preprint arXiv:1809.07882*, 2018.

[19] Magdalena Ivanovska, Audun Jøsang, Lance Kaplan, and Francesco Sambo. Subjective networks: Perspectives and challenges. In *International Workshop on Graph Structures for Knowledge Representation and Reasoning*, pages 107–124. Springer, 2015.

[20] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017.

[21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[22] Charles Kemp and Joshua B Tenenbaum. The discovery of structural form. *Proceedings of the National Academy of Sciences*, 105(31):10687–10692, 2008.

[23] Marco F. Cusumano-Towner, Feras A. Saad, Alexander K. Lew, and Vikash K. Mansinghka. Gen: A general-purpose probabilistic programming system with programmable inference. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2019, pages 221–236, New York, NY, USA, 2019. ACM.

[24] Kevin Canini, Lei Shi, and Thomas Griffiths. Online inference of topics with latent dirichlet allocation. In *Artificial Intelligence and Statistics*, pages 65–72, 2009.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We believe we have shown a proof of concept that MetaCOG 1) can learn an accurate representation of an object detection system's performance, and 2) use this learned meta-cognitive representation to make better inferences about the objects causing the detections.
   (b) Did you describe the limitations of your work? [Yes] See Discussion in Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code and data for reproducing the main results will be located at: link
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The dataset (and split) is described in 4.1. Hyperparameters for MetaCOG in Experiment 1 are described in A.1 and following subsection, and for Experiment 2, they are described in A.6 and the following subsection.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] All errorbars are 95% bootstrapped CIs.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See A.10
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cited the NNs we used in A.3.1, the environment for rendering the dataset in A.2.1, and the programming language used for MetaCOG in A.1
   (b) Did you mention the license of the assets? [Yes] The license of the NNs are given in A.3.1, and the license of TDW (used to render the dataset) is given in A.2.1
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We included the dataset used in Exp 1 in the supplemental material located at: link
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

# A Appendix

## A.1 Additional MetaCOG details

The MetaCOG model and inference were implemented in the Julia-based probablistic programming language Gen [23], licensed under the Apache License Version 2.0.

### A.1.1 Meta-cognitive Dynamics Kernel

At $t = 0$, each category's initial hallucination rate $\lambda_c$ is initialized from a Gamma distribution with parameters $\alpha = 1$ and $\beta = 1$. After inference over a scene, the prior over $\lambda_c$ evolves by computing the inferred number of hallucinations at time $t$ (based on the difference between world state $W_t$ and observations $O_t$, see A.1.2), and updating the $\alpha$ and $\beta$ parameters of the Gamma distribution as the conjugate prior over the Poisson distribution in the meta-cognition $V$.

Beliefs about the miss rates evolve in an analogous manner. When $t = 0$, the detection rate $p_c$ for each category is initialized from a Beta distribution with parameters $\alpha = 1$ and $\beta = 1$ (equivalent to a uniform distribution). The parameters from the Beta distribution are then updated as the conjugate prior over the Geometric distribution, based on the inferred missed detections (by comparing $W_t$ against $O_t$, see A.1.2).

### A.1.2 Details of procedure for taking the difference between $W_t$ and $O_t$

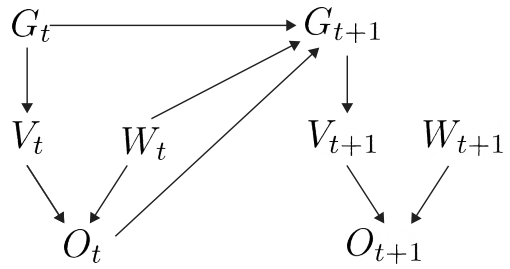


Figure 4: Schematic depicting the forward generative model. $G_t$ is the prior over meta-cognition ($V_t$) at time t; $W_t$ is the world state; and $O_t$ are the observations that are generated. $W_t$, $O_t$, and $G_t$ collectively influence $G_{t+1}$, the prior over $V_{t_1}$.

From a world state $W_t$ and a set of observations $O_t$, MetaCOG tries to infer which detection were hallucinations or misses so as to update the prior over the hallucination and miss rate. The procedure for this inference is as follows: first, the objects in $W_t$ are projected from 3D space to the 2D image. Then, for each object category, the projected objects are matched up with detections within radius $\sigma = 200$ pixels (equal to the spatial noise parameter). After matching, the number of unmatched objects per category are counted, and considered to have been missed. The number of unmatched detections are also counted, and considered to be hallucinations. This process informs the updating of the $\alpha$s and $\beta$s in the priors described in A.1.1.

### A.1.3 Spelke principles and generative model parameters

Implementation of Spelke principles was as follows. World states without object persistence were not included in $\mathbb{W}$, which is equivalent to implicitly setting their prior probability to 0. The assumption that two objects cannot occupy the same region in space was implemented through a prior over $\mathbb{W}$ where the probability of having one object near another decreased to 0 according to a Gaussian distribution with $\sigma^2 = 1$.

To account for noise in a detector's location detection $O_t$, each detection was modeled as having 2D spatial noise, following a Gaussian with $\sigma_{x,y} = 200$ pixels.

Finally, the full generative model requires specifying a prior distribution over camera positions and focal points (although these are observable), set as uniform over 3D space, and a prior distribution over expected number of objects in a scene, sampled from a geometric distribution with parameter $p = 0.9$ (with a uniform prior over category type).

### A.1.4 Inference Procedure Implementation

We sequentially approximate the joint posterior given in eq. 2 using a particle filter with 100 particles. The solution space to the inference problem that we consider is sparse, and Sequential Monte-Carlo methods can suffer from degeneracy and loss of diversity in these situations. Our inference approach solves these problems by implementing particle rejuvenation over objects, locations, and beliefs about $V$. Rejuvenation is conducted using a series of Metropolis-Hastings [24] moves with data-driven proposals designed to obtain samples from $Pr(\vec{V}, \vec{W}|\vec{O}, \vec{cj})$.

During object rejuvenation, a new state $W_t'$ is proposed by either adding or removing an object from the current state $W_t$. The probability $p$ of proposing adding or removing an object depends on the belief about $V_t$ and the observed detections. The number of detections across the five categories are summed. Call this sum $k$. The probability $p$ of $k$ or more detection being produced by hallucinations is calculated as $p = 1 - e^{-\lambda_{total}} \sum_{i=0}^{k} \frac{\lambda_{total}^i}{i!})$, where $\lambda_{total} = \sum_c(\lambda_c)$. (That $e^{-\lambda_{total}} \sum_{i=0}^{k} \frac{\lambda_{total}^i}{i!}$ term is the cdf of $Pois(\lambda_{total})$ from 0 to $k$.) This way, when the belief about the overall hallucination rate is high relative to the number of detection, the probability $p$ of proposing adding a new object to the world state is low, but when the hallucination rate is low relative to the number of detections, the probability $p$ of proposing adding a new objects is high.

Objects in $W_t$ have a weighted probability of being removed to produce $W_t'$, such that their probability of removal is inversely proportional to the number of times the object was observed in scene $t$. New objects are added based on a data-driven distribution. This distribution samples object categories from a categorical distribution biased towards categories observed in the current scene $t$ (see A.1.5). With probability 0.5, location is sampled from a 3-D uniform distribution, and otherwise sampled from a data-driven function with location biased toward 3D points likely to have caused the 2D detections (see A.1.5). The proposal for adding a new object or removing and existing object from the world state is accepted or rejected according to the MH algorithm.

After a proposed change to $W$, a second rejuvenation step is performed on locations, wherein an object in $W$ is randomly selected (with equal probability) to have a new location proposed. With probability 0.5, the new location is drawn from a multivariate normal distribution centered on the previous location, and it is otherwise sampled from a data-driven proposal (see A.1.5). The proposed world-state $W'$ with a perturbed location is then accepted or rejected according to the MH algorithm.

Finally, new parameters $\theta'$ for the meta-cognition are proposed by perturbing $\theta$. Each parameter in the pair of length $|C|$ vectors is perturbed, with new values generated for $\theta(i,j)_t$ and for $\theta(i,j)_{t-1}$ $\forall i, j$, where $i$ indexes the two vectors and $j$ indexes their elements. The new value for each $\theta(i,j)_t$ is sampled from the appropriate distribution (Beta for the miss rate, Gamma for hallucination rate) with $\alpha_t$ and $\beta_t$, while the new value for $V(i,j)_{t-1}$ is sampled with $\alpha_{t-1}$ and $\beta_{t-1}$. These new values are accepted or rejected according to the MH algorithm.

This three-step rejuvenation process for world states, locations, and parameters of the meta-cognition is done for each particle for 200 iterations and the last state reached in the chain is used as the new rejuvenated particle.

### A.1.5 Data-Driven Proposal Function

During rejuvenation, new objects are proposed to be added to $W_t'$ using a data-driven distribution. The category of the object is comes from a categorical distribution, the weight is divided proportionally to the number of times that object was observed in the scene $t$. Object that were not observed in the scene get a weight as if they were observed once. With probability 0.5, the location is sampled from a 3-D uniform distribution, and otherwise sampled from a data-driven function. Using the data-driven function, the point is sampled based on proximity to the line-segment that, when projected onto the 2D image, would result in the point where the detection was observed. The probability of proposing a particular point decreases with the distance from this line segment, following a Gaussian with $\sigma^2 = 0.01$.

In the second rejuvenation step a new location is proposed for an object. With probability 0.5, the new location is drawn from a multivariate normal distribution centered on the previous location, with $\sigma_{x,y,z} = 0.01$. Otherwise, it is sampled as described in the previous paragraph (based on proximity to the line segment that would results in the detection's 2D location).

### A.2 Experiment 1 Dataset

Evaluating Experiment 1 we needed a dataset of videos of the same COCO object categories (so pre-trained NNs could detect them) across multiple videos. These videos had to include the camera trajectory and position. Furthermore, so as to enable use to assess MetaCOG's accuracy locating objects in 3D space, we also needed ground-truth labels for the objects, including their locations in 3D space. Because we did not identify any existing datasets that satisfy all these constraints, we rendered our own, custom dataset.

#### A.2.1 TDW scene rendering

The TDW environment is licensed under the BSD 2-Clause "Simplified" License, Copyright 2020 Massachusetts Institute of Technology. We used the "box room 2018" model with a footprint with dimensions $\approx$ 12 x 8. For scale, the largest object model is about 1 unit along any face.

The number of objects in the scene is uniformly drawn from the counting numbers up to 3 and each object is uniformly assigned one of 5 object categories: potted plant, chair, bowl, tv, or umbrella. We sequentially place objects at locations drawn from a uniform distribution, resampling if two objects are within 1 units of each other or the walls so as to avoid collisions.

#### A.2.2 Camera Trajectory Sampling

A camera trajectory consists of a sequence of camera positions and focal points. We generate frames by querying the camera for an image at 20 linearly-spaced times.

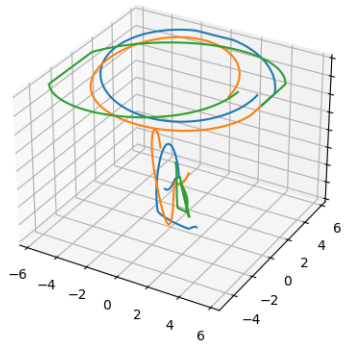


Figure 5: Three sampled agent trajectories. The wide circular patterns at the top are camera positions, and the smaller patterns near the bottom are camera focal points.

Camera trajectories consisted of a circular path around the periphery of the room with noise, generated by a Gaussian process with a radial basis function (RBF) kernel with parameters $\sigma = 0.7$ and $\ell = 2.5$. The height of the camera is held constant at $y = 2$. The focal point trajectory is also sampled from a Gaussian process with an RBF kernel, mean above the center of the floor and component-wise parameters $\sigma = 0.7$ and $\ell = 2$. Figure 5 shows some example trajectories.

#### A.2.3 Video Counter-balancing

To avoid order effects, all tests were run with four different videos orders. Videos were first randomly labeled from 1 to 50, and the four counterbalanced orders were: {1 ... 50}, {50 ... 1}, {26 ... 50, 1 ... 25}, and {25 ... 1, 26 ... 50}.

### A.3 Details About Neural Networks Used in Experiment 1

#### A.3.1 Pre-trained Weights

For Faster-RCNN, we used the resnet50 pre-trained weights. For RetinaNet, we used the resnet50 pre-trained weights. And for DETR, we used the resnet101 pre-trained weights. These NNs were pre-trained on the COCO dataset.

Faster-RCNN is licensed under The MIT License (MIT), RetinaNet under Apache License 2.0, and DETR under Apache License 2.0.

#### A.3.2 Post-processing

We process the outputs of the object detectors by filtering for the 5 object categories present in the scenes and then performed Non-Maximum Suppression (NMS) with an IoU (Intersection over Union) threshold of 0.4 (applied only for RetinaNet and Faster R-CNN, as it is not typically used for DETR).

As input to MetaCOG, we took the top five highest-confidence detections per frame. The reason for taking the top five highest-confidence detection per frame, rather than all of the detections per frame is that sometimes some networks, especially DETR, can output many detections for a single frame. Typically, a confidence threshold is selected by fitting to ground-truth labels. But our setting is unsupervised and meant to apply to situations where ground-truth labels are perhaps unavailable. So, instead of fitting the confidence threshold using ground-truth, we took the top five highest-confidence detections. On most frames and with most networks, the NNs rarely output more than 5 detections, so in practice, taking the top 5 detections rarely affected the NN's output.

#### A.3.3 mAP of NNs on our dataset

Using the COCO evaluator [5], we evaluated the pre-trained NNs on our dataset. Faster-RCNN has a mAP of 39.4 on the training set, and 41.1 on the test set. This is similar to the mAPs reported for the most challenging object categories, like plants. Faster-RCNN's reported mAP for plant is 39.1 and 40.1 (on the PASCAL VOC 2007 and 2012 test sets, respectively) [21]. RetinaNet had a mAP of 35.9 on the training set, and 36.3 on the test set. RetinaNet's reported mAP (across all object categories) on the COCO test dataset is 39.1 [20]. This confirms that both Faster-RCNN and RetinaNet are operating within their expected performance ranges.

### A.4 Metrics for Experiment 1

We evaluate MetaCOG's performance in three ways: 1) by testing whether it can learn an accurate meta-cognition of the object detection module, 2) by testing whether the learned meta-cognition improves inferences about which objects are where in 3D space, and 3) by testing whether the learned meta-cognition leads to improved accuracy at detecting objects in the 2D images.

#### A.4.1 MSE for $\theta$

To measure how well MetaCOG learned a meta-cognition, we calculated the MSE between the inferred parameters $\hat{\theta}$ of the meta-cognition and the true parameters $\theta$. The true $\theta$ was calculated for each object detection system as described in the following section A.4.2. The squared error was calculated for corresponding values in $\hat{\theta}$ and the true $\theta$, then summed and divided by the total number of values ($2 * |C|$).

#### A.4.2 Calculating ground-truth $\theta$

We needed a way, for a given image, to calculate how many times an object of each category was missed or hallucinated by the object detector. There are many possible ways to define and calculate ground-truth misses and hallucinations, but we chose a definition and procedure to match the definition and procedure MetaCOG uses (see A.1.2).

We used the 3D position of the object, the camera, and the camera's focus to determine whether and where the object would appear in the image. We considered an object to be missed if the object was in the image, but not detected within 200 pixels of its projected 2D location on the image. We calculated the miss rate for a category $c$ as the number of misses divided by the number of times an object of category $c$ was in view.

We considered a detection to be a hallucination if the detection occurred more than 200 pixels away from the ground-truth projection. We calculated the hallucination rate for a category $c$ as the number of hallucinations of an object of category $c$ divided by the number of frames.

The 200 pixel radius was set to match the radius used by MetaCOG in matching $W_t$s to $O_t$s (see A.1.2).

#### A.4.3 Jaccard Similarity

The Jaccard similarity coefficient is a useful metric for evaluating inferences about *which* objects are in a frame or scene. The Jaccard similarity coefficients measures the similarity between two sets by taking the size of the intersection of two sets and dividing it by the size of the union of the two sets. We can apply this measure to our setting by treating the object labels that were inferred as one set and

the objects that were actually present as the second set. The Jaccard similarity coefficient is given by:

$$J(\mathbb{I}, \mathbb{G}) = \frac{|\mathbb{I} \cap \mathbb{G}|}{|\mathbb{I} \cup \mathbb{G}|} \tag{4}$$

where, in our setting, $\mathbb{I}$ is the set of object labels that were inferred (i.e. $\mathbb{I}$ = {chair, chair, bowl}) and $\mathbb{G}$ is the set of objects in ground-truth (i.e. $\mathbb{G}$ = {chair, bowl}). (In this example, $J(\mathbb{I}, \mathbb{G}) = \frac{2}{3}$.)

This Jaccard Similarity coefficient can be applied either at the frame or scene level, and is used in both contexts in this paper.

#### A.4.4 2D Inference Accuracy

To assess the accuracy of of the both MetaCOG and the NNs in a comparable way, we wanted an accuracy metric that could incooporate both *which* object categories are present, and *where* they are. The NNs operate over 2D images, not 3D spaces, so we must make this comparison at the level of 2D frames. The standard metric for object detection accuracy on images is mean Average Precision (mAP), which requires both a ground-truth 2D bounding box and an inferred/detected bounding box. Unfortunately, MetaCOG outputs a point in 3D space, which can be projected to a point in 2D space, but cannot be used to make a bounding box. But we still needed a metric by which to compare MetaCOG's object detection accuracy to those of the neural networks without a meta-cognition.

To solve this problem, we projected MetaCOG's inferences about the 3D location of objects to a point on each 2D image, and for the neural networks, we took the centroid of the bounding boxes. This way, MetaCOG and the neural networks' detections have the same format: an object label and a point on the image, or a tuple of $(c, x, y)$. Then, for each object category, we used the Hungarian algorithm with Euclidean distances as cost to pair up the detections with the centroid of the ground-truth bounding box. For each pairing, we simply coded whether or not the detected point was within the ground-truth bounding box as 0 or 1. Last, we calculated the Jaccard similarity coefficient (see A.4.3) defining pairs as within the intersection of the sets if and only if they pair received the 1 coding. In other words, we counting up the number of pairs where the detection lay within the ground-truth bounding box, and divided by and the union of the detection and ground-truth sets: the sum of the number of unpaired detections, unpaired ground-truth bounding boxes, and number of total pairings. To extrapolate this accuracy metric from frames to videos, we simply average across frame accuracy for each frame in the video, and we average across videos to calculate overall accuracy. This accuracy metric encodes, for 2D frames, accuracy at detecting both *which* objects are present and *where* they are.

It is important to note that assessing accuracy at the 2D frame level is different from assessing accuracy at the 3D scene level. Accuracy at the 2D level should depend on the objects that are in the frame. For example, it could be the case that the scene contains a chair and a plant, but the chair is only in frame once, while the plant is in frame 20 times. In that case, 2D accuracy should depend more on identifying and locating the plant than the chair. This distinction will be important when we assess accuracy at the level of the 3D scene in the next section.

### A.5 Experiment 1 Supplemental results and information

#### A.5.1 3D Inference Accuracy

Our main results focused on MetaCOG's accuracy in 2D images. This was necessary so that we could evaluate its inferences against *NN Output*, because the object detection neural networks that we use only provide location on 2D images. Although not our main focus, here we report supplemental results evaluating MetaCOG's capacity to infer 3D world states in Experiment 1. This allows us to test MetaCOG's inferences at the scene level (i.e., what objects are in the room and where are they?), rather than at the frame level (i.e., what objects are visible at this time point and where are they in this 2D projection?). Throughout, we compare MetaCOG only to Lesioned MetaCOG, as it is the only comparison model that also produces results in 3D space.

We first evaluate MetaCOG's accuracy in determining *which* objects are present in a scene (independent of location). To do this, we calculated the Jaccard similarity coefficent (see A.4.3) between the object categories inferred to be present in the scene, and those actually present. Table 3 shows Jaccard similarity coefficient averaged across the scenes. We find that on the whole, MetaCOG outperforms

Table 3: Scene-level accuracy results for Experiment 1. Values in parentheses represent 95% bootstrapped confidence intervals.

| Object Detector | Model | Acc. Training | Acc. Test |
|---|---|---|---|
| **RetinaNet** | MetaCOG | **0.666** (0.577, 0.753) | **0.607** (0.523, 0.688) |
| | Lesioned MetaCOG | 0.578 (0.492, 0.664) | 0.481 (0.413, 0.554) |
| **FasterR-CNN** | MetaCOG | **0.757** (0.669, 0.838) | **0.664** (0.577, 0.749) |
| | Lesioned MetaCOG | 0.575 (0.493, 0.660) | 0.491 (0.425, 0.560) |
| **DETR** | MetaCOG | 0.455 (0.385, 0.527) | 0.436 (0.368 , 0.506) |
| | Lesioned MetaCOG | **0.478** (0.403, 0.557) | **0.456** (0.385, 0.528) |

Table 4: Average 3D distance between inferred and ground-truth object locations

| Object Detector | Model | Avg. Distance All Pairs | Avg. Distance Shared Pairs |
|---|---|---|---|
| **RetinaNet** | MetaCOG | **0.367** (0.306, 0.418) | **0.334** (0.271, 0.383) |
| | Lesioned MetaCOG | 0.380 (0.258, 0.472) | 0.356 (0.262, 0.434) |
| **FasterR-CNN** | MetaCOG | 0.372 (0.313, 0.422) | **0.298** (0.250, 0.339) |
| | Lesioned MetaCOG | **0.306** (0.253, 0.351) | 0.305 (0.250, 0.352) |
| **DETR** | MetaCOG | 0.621 (0.429, 0.787) | **0.413** (0.304, 0.504) |
| | Lesioned MetaCOG | **0.506** (0.330, 0.648) | 0.471 (0.333, 0.588) |

Lesioned MetaCOG, except for when paired with DETR, where the two perform comparably (as can be observed by noting that the confidence interval of each model's accuracy contains the mean of the other, indicating that there is no significant different between the means).

Next, we evaluate MetaCOG's accuracy in determining *where* the objects are in the scene. To do this, we first paired the inferred and ground-truth object locations of the same category using the Hungarian algorithm with Euclidean distance as the cost. We then averaged the distance between all of the pairs, in both the training and test sets, resulting in an average distance between inferred and ground-truth objects. The results in are shown in Table 4, in the Avg. Distance All Pairs column.

On the whole, Lesioned MetaCOG's meta-cognition represents a system with a relatively high hallucination rate (higher than the rates inferred by MetaCOG). That belief leads Lesioned MetaCOG to only infer that an object was present when there were more detections of that object, and more spatial information, leading to greater location precision for the objects it inferred to be present. Consequently, Lesioned MetaCOG only infers the presence of the objects that are easiest to locate.

For a more fair comparison, we can examine the average distance between pairs of inferred and ground-truth objects only for ground-truth objects that both MetaCOG and Lesioned MetaCOG's inferred to be present. These average distances are reported in the Avg. Distance Shared Pairs column of Table 4. For objects that both MetaCOG and Lesioned MetaCOG inferred to be present, MetaCOG consistently infers slightly better locations than does the Lesioned model.

For reference, the footprint of the room is $\approx$ 12 x 8, and the largest object is 1 unit across.

### A.5.2 Faster-RCNN Re-training Details

We trained Faster-RCNN on the training set using MetaCOG's inferences as ground-truth labels. Since MetaCOG infers an object category and a 3D point, rather than a bounding box, we constructed a 50x50 pixel bounding box centered on the MetaCOG inferred's 3D point projected onto the 2D image. Then, Faster-RCNN with pre-trained weights was trained on the 1000 images (50 videos * 20 frames) in the training set. The training set was augmented by flipping the images horizontally. Faster-RCNN was trained for 10 epochs with a SGR optimizer (learning rate = 0.005, momentum = 0.9, weight decay = 0.0005) and a learning rate scheduler (step size = 3, gamma = 0.1). The classification head layer was not replaced because we wished to retain the information about the COCO object categories Faster-RCNN had been pre-trained on, as the object categories in our dataset were a subset of COCO categories.

### A.5.3 Lesioned MetaCOG

To whether learning the correct parameters $\theta$ for the meta-cognition actually matter, or if the priors (over $\theta$ and the Spelke object principle priors over world states) do all the work of improving accuracy, we tested a *lesioned* version of MetaCOG fixing the values in $\theta$ to the mean of the prior. The initial prior over the hallucination rate for each object category is a Gamma distribution with with $\alpha = 1$ and $\beta = 1$. The mean of that prior is then $1.0$, so in Lesioned MetaCOG, the hallucination rates are set to $1.0$ The initial prior over the miss rate for each object category is a Beta distribution with $\alpha = 1$ and $\beta = 1$. The mean of that prior is then $0.5$, so in Lesioned MetaCOG, the miss rates are set to $0.5$. So this lesion produces the expectation that an object of each category will be hallucinated on average once per frame, and that, when an object is in view, it has a 50% chance of being missed.

## A.6 Lightweight MetaCOG

In addition to the full MetaCOG model described in the main text, we also implemented a simplified version for the simplified setting used in Experiment 2. This simplified setting is object detection without location, wherein a black-box object detector generates labels (without associated locations) for the objects present in a scene. So observations are collections of object labels. Furthermore, the goal of inferences in this simplified setting is not to infer what objects are where in 3D space, but merely which objects are present in the scene. Camera trajectories are no longer included, and all object are assumed to remain in view at all times.

Additionally, the meta-cognition in this Lightweight MetaCOG is simplified as well: the prior over the meta-cognition, $G$, does not vary with time, but is fixed as a Beta distribution with parameters $(\alpha = 2, \beta = 10)$. Hallucination and misses are treated as Bernoulli random variables, with their parameters $\theta$ sampled from the Beta prior, separately for each object category.

The inference targets and procedure are largely unchanged, with the estimate of joint posterior now sequentially approximated via:

$$Pr\left(\vec{V}, \vec{W} | \vec{O}\right) \approx Pr\left(\hat{V_0}^0\right) \prod_{t=1}^{T} Pr\left(O_t | \hat{V_t}^t, \hat{W_t}^t\right) Pr\left(\hat{W_t}^t\right) Pr\left(\hat{V_t}^t | \hat{V_{t-1}}^t\right) \tag{5}$$

where the transition kernel, $Pr(\hat{V}^t | \hat{V}^{t-1})$, defines the identity function. The details of the inference procedure are left to A.6.1.

### A.6.1 Details of Inference Procedure for Lightweight MetaCOG

Equation 5 is estimated via particles filtering, with 100 particles.

We implemented rejuvenation using a series of Metropolis-Hastings MCMC perturbation moves over $\hat{\theta}$. The proposal function is defined as a truncated normal distribution with bounds $(0, 1)$:

$$\hat{\theta_{i,j}}^{t'} \sim \mathcal{N}(\mu = \hat{\theta_{i,j}}^t, \sigma^2 = 0.01) \tag{6}$$

where $\theta_{i,j}$ is the $j$th element of vector $i$ in the pair of vectors of parameters $\theta$. A proposal is accepted or rejected according to the Metropolis-Hastings algorithm. Each element in $\theta$ is rejuvenated separately and in randomized order.

## A.7 Experiment 2 Dataset

Here we discuss how we synthesized the dataset for evaluating Lightweight MetaCOG in Experiment 2.

In this context, the object detector can be represented by its parameters, $\theta$, which is a pair of vectors of length $|C|$ containing the hallucination and miss rates for each object category. The parameters for each object detector was generated by drawing ten independent samples from a beta distribution, $\sim \mathrm{B}(\alpha = 2, \beta = 10)$. This distribution allows us to sample object detectors with variable error rates (mean value = $\frac{1}{6}$) while maintaining a low probability of sampling object detectors that produced

hallucinations or misses more often than chance (0.005 chance of sampling values above 0.5; 0.06 chance that complete sampled object detector has at least one hallucination or miss rate above 0.5).

For each object detector, we sampled 75 world states. A Poisson distribution $N \sim Poisson(\lambda = 1)$ truncated with bounds $[1, 5]$ determined the number of objects in a world state. The object categories were samples from a uniform distribution. Each world state was a hypothetical collections of objects, summarized as a vector of 1s and 0s indicating the presence or absence of each category of objects. For each world state we used the object detector to synthesize the detections from $5 - 15$ simulated frames (number sampled from a uniform distribution), producing a total of $375 - 1125$ simulated frames per object detector. Inferences about the hallucination and miss rates of each object category are independent, and we thus considered situations with only five categories.

### A.8 Metrics for Experiment 2

To measure how well MetaCOG learned a meta-cognition, we calculated the mean squared error (MSE) between the inferred parameters of the object detector $\hat{\theta}$ and the true parameters $\theta$ generating the percepts given by

$$\text{MSE} = \frac{1}{2|C|} \sum_{c \in C} \left( (H_c - \hat{H}_c)^2 + (M_c - \hat{M}_c)^2) \right) \tag{7}$$

where $C$ is the the set of object classes, $H_c$ is the hallucination rate for category $c$, and $M_c$ is the miss rate for category $c$.

Accuracy was measured using the Jaccard similarity coefficient of the set of object classes in the ground-truth world state $W_t$ and the set of object classes in the inferred world state $\hat{W}_t$, see A.4.3.

To analyze MetaCOG's accuracy as a function of faultiness in the detected labels, we computed the average faultiness in an observation $O_t$ as

$$\zeta_t = \frac{1}{|O_t||C|} \sum_{c \in C} \sum_{x \in O_t} |1_{W_t}(c) - 1_x(c)| \tag{8}$$

where $C$ is the set of object classes, $O_t$ is the collection of detected labels generated from world state $W_t$, $1_{W_t}(c)$ is an indicator for whether an object of class $c$ is in $W_t$, and $1_x(c)$ is an indicator for whether an object of class $c$ is in the set of detected labels $x$.

Because our sampling-based dataset generation process (A.7) does not guarantee enough data points for every possible faultiness value, analysis in Figure 3C and D were computed using a rolling window such that each point shows average accuracy on the $[\zeta - .05, \zeta + .05]$ range.

### A.9 Comparison Models for Experiment 2

To test whether learning a meta-cognition improves inferences about the world state, we compare MetaCOG with and without the meta-cognitive learning. We call MetaCOG without learned meta-cognition Lesioned MetaCOG. Like the other MetaCOG models, Lesioned MetaCOG has a meta-cognitive representation of $V$ and uses an assumption of object permanence to infer the world states causing the detected labels. Lesioned MetaCOG, however, does not learn or adjust the parameters $\theta$ in its meta-represetation based on the observed labels. Formally, Lesioned MetaCOG assumes that the hallucination and miss rates for every category are the mean of the beta prior over hallucination and miss rates, call it $\hat{\theta}_{0,\mu}$. Lesioned MetaCOG then uses the same particle filtering process described in 5, except that it conditions on $\hat{\theta}_{0,\mu}$ instead of $\hat{\theta}_{T,\mu}$.

As described in A.6, the prior over both the hallucination rate and the miss rate is a Beta distribution parameters $(\alpha = 2, \beta = 10)$. The mean of this prior is then $\frac{1}{6}$. So in the Lesioned MetaCOG for Experiment 2, the hallucination rate and the miss rate are both set to $\frac{1}{6}$.

We also compare two variations of MetaCOG with learning. MetaCOG Learning performs a joint inference over $\theta$ and $\vec{W}$ based on observations $\vec{O}$ sequentially, as observations are presented sequentially. This model allows us to evaluate how MetaCOG's inferences improve as a function of the observations it has received. We name this model observation-by-observation inference MetaCOG Learning.

After having received all $T$ observations, MetaCOG could retrospectively re-infer the world states causing the $T$ observations. This model, simply called MetaCOG, re-infers the world states that caused its observations conditioned on its estimate $\hat{\theta}^T_{T,\mu}$, as described in 3.2.1.

MetaCOG Learning lets us interpret how MetaCOG learns a meta-cognition, and MetaCOG lets us test how MetaCOG performs after having learned that meta-cognition. Lesioned MetaCOG serve as a baseline model for comparison.

### A.10 Compute Resources

We estimate the compute resources used in producing the final results reported using our MetaCOG model in Experiment 1. Running the MetaCOG (and Lesioned) model four times per each of the three neural networks used 36 CPUs on an internal cluster for approximately 100 hours.