



Designing Perceptual Puzzles by Differentiating Probabilistic Programs

Kartik Chandra
MIT CSAIL
Cambridge, MA, USA
kach@csail.mit.edu

Tzu-Mao Li
UC San Diego
San Diego, CA, USA
tzli@ucsd.edu

Joshua Tenenbaum
Department of Brain and
Cognitive Sciences
MIT CSAIL
Cambridge, MA, USA
jbt@mit.edu

Jonathan Ragan-Kelley
MIT CSAIL
Cambridge, MA, USA
jrk@mit.edu

ABSTRACT

We design new visual illusions by finding “adversarial examples” for principled models of human perception — specifically, for probabilistic models, which treat vision as Bayesian inference. To perform this search efficiently, we design a *differentiable* probabilistic programming language, whose API exposes MCMC inference as a first-class differentiable function. We demonstrate our method by automatically creating illusions for three features of human vision: color constancy, size constancy, and face perception.

CCS CONCEPTS

• **Computing methodologies** → **Perception**; **Probabilistic reasoning**; • **Mathematics of computing** → **Automatic differentiation**; **Markov-chain Monte Carlo methods**.

KEYWORDS

illusions, differentiable programming, probabilistic programming

ACM Reference Format:

Kartik Chandra, Tzu-Mao Li, Joshua Tenenbaum, and Jonathan Ragan-Kelley. 2022. Designing Perceptual Puzzles by Differentiating Probabilistic Programs. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings (SIGGRAPH ’22 Conference Proceedings)*, August 7–11, 2022, Vancouver, BC, Canada. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3528233.3530715>

1 INTRODUCTION

Human visual perception does not always match objective reality. Consider “The Dress,” a visual illusion discovered by chance in 2015. This photograph of a blue and black striped dress elicits a strange perceptual effect: while most viewers indeed describe the dress as blue and black, roughly a third instead confidently describe it as white and golden [Lafer-Sousa et al. 2015]. Because these two perceptual modes are so dramatically different, viewers are often baffled to learn that others see the same image differently.

Illusions like “The Dress” have long been studied for insight into perception. This paper is motivated by the question, *Can we systematically generate new such illusions in a principled manner?*



This work is licensed under a [Creative Commons Attribution International 4.0 License](https://creativecommons.org/licenses/by/4.0/).

SIGGRAPH ’22 Conference Proceedings, August 7–11, 2022, Vancouver, BC, Canada
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9337-9/22/08.
<https://doi.org/10.1145/3528233.3530715>

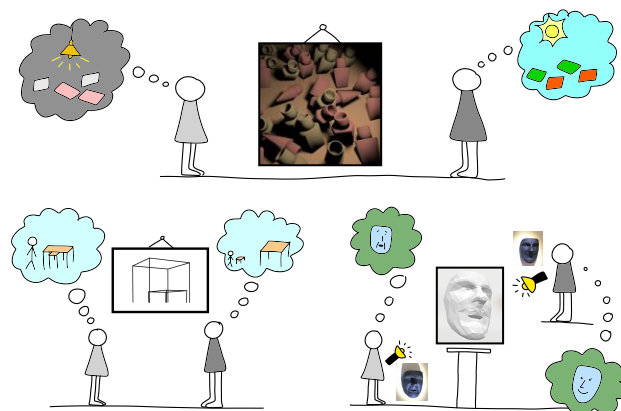


Figure 1: Illusion synthesis: We find scenes where different people perceive different colors (top), where most people infer incorrect geometry (left), and where changing the lighting causes a face to change expression (right). To find them, we perform gradient descent optimization over Bayesian models of human visual perception, which are implemented in our differentiable probabilistic programming language.

Following Durand et al. [2002], we would like to think of such perceptually-aware image synthesis as “inverse inverse rendering.” Given a model of human visual perception that infers a scene from an image (“inverse rendering”), we wish to search for input images that elicit interesting responses as output from the model (“inverse inverse rendering”).

What model should we choose? A long line of cognitive science research suggests that human visual perception is modeled well by *probabilistic models*, which treat perception as Bayesian inference. Under such a model, a viewer has some *prior belief* about the statistical distribution of scenes (objects, their colors, and lighting conditions) in the world. Faced with an *observation* (an image), the viewer performs *inference* to update their prior belief into a *posterior belief* using Bayes’ rule: $p(\text{Scene} \mid \text{Image}) \propto p(\text{Image} \mid \text{Scene})p(\text{Scene})$.

Probabilistic models have already been used for inverse rendering in a variety of settings. Here, however, we are interested in *inverse inverse rendering*: we seek an Image^* such that the model’s inferred posterior belief, given by the distribution $p(\text{Scene} \mid \text{Image}^*)$, is confidently incorrect. In this way, Image^* is like an “adversarial example” — not for a deep neural network, but instead for a small, principled model of one aspect of human perception.

In this paper, we offer a general-purpose tool for solving such inverse problems: a *differentiable probabilistic programming language*. Regular probabilistic programming languages (PPLs) allow users to express probabilistic models as structured generative processes, which are then inverted into efficient programs for performing Bayesian inference with respect to a given observation. By enabling differentiation through the inference process, a *differentiable* PPL allows users to optimize (by gradient descent) an observation to evoke some desired property in the inferred posterior distribution. Using this tool, we can model a variety of perceptual phenomena with simple generative models — just a few lines of code — and then search for stimuli that elicit interesting behaviors from those models.

This work is a synthesis of many ideas from a variety of fields. Section 2 reviews necessary background, situating our work in the broader context of current machine learning and cognitive science research. After that, we present our two contributions: a differentiable PPL (Section 3), and a formalization of “illusion synthesis” as an application of differentiable PPLs (Section 4). We apply our method to create illusions in three domains: color constancy (4.1), size constancy (4.2), and face perception (4.3). Finally, we reflect on our work’s broader implications for the graphics and cognitive science communities, and discuss scope for future work (Section 5).

2 BACKGROUND

2.1 Motivation: Why do we need Bayesian models?

Because our goal is to find visual illusions that work for humans, we need a model of vision whose edge-cases match those of humans. This choice is not straightforward: convolutional neural networks (CNNs), for example, do not meet this criterion because even though we can easily fool them with adversarial examples [Szegedy et al. 2014], nearly a decade of research teaches us that those examples do not transfer to fool humans [Sinz et al. 2019]. Indeed, this is precisely why adversarial examples are intriguing: they demonstrate how deep learning differs fundamentally from human perception.

While this consensus is occasionally disputed, the evidence still does not suggest any method for generating robust, compelling illusions like “The Dress.” For example, Elsayed et al. [2018] optimized images to be misclassified by an ensemble of ten CNNs, and found that the resulting adversarial examples could also cause humans to select the same incorrect labels. However, the humans were only fooled in the “time-limited” setting where images were flashed for just 60-70ms — the effect disappeared for longer presentations. Similarly, Zhou and Firestone [2019] found that humans could pick out the incorrect label that a fooled CNN would assign. However, this was only in a forced-choice setting where participants were given a constrained set of incorrect labels to choose from. Neither of these effects is like “The Dress,” which persists even under natural viewing and reporting conditions.

Seeking to capture exactly these robust perceptual effects, we turn away from CNNs, and instead consider an alternate model of vision: the *Bayesian* model, which seeks to explain human vision as the computations of a rational perceptive agent. Below, we provide a brief introduction to Bayesian models, why we expect them to

match human perception, and how they are implemented in practice (we direct readers seeking further background to a textbook by Goodman et al. [2016]). Finally, we discuss algorithms for differentiating over those models, which in turn enable us to optimize “adversarial examples” that indeed transfer to humans.

2.2 Modeling vision as Bayesian inference

Probabilistic models of perception. A long line of work from the cognitive science community (see, e.g., a book by Knill and Richards [1996] or surveys by Mamassian et al. [2002] and Kersten et al. [2004]) has argued that because many possible scenes can map to the same image on the human retina, parsing an image into a scene by “inverse graphics” is an ill-posed problem. Therefore, vision must be probabilistic in nature: it must rationally infer probable scenes from an image based on prior beliefs about the statistics of scenes in the viewer’s environment. For example, this view motivates a simple Bayesian model of color constancy [Brainard and Freeman 1997]: We start with a distribution $p(\text{Light, Color})$ representing our prior belief over scenes (for example, that daylight is likelier than bright green light), and a distribution $p(\text{Image} \mid \text{Light, Color})$ that encodes how scenes are rendered into images along with any uncertainty in that process (for example, sensor noise in the eye or camera). Bayes’ rule then allows us to infer the posterior distribution of scenes, conditioned on a given observed image: $p(\text{Light, Color} \mid \text{Image}) \propto p(\text{Light, Color})p(\text{Image} \mid \text{Light, Color})$.

This account of vision posits a natural “specification” of what vision computes, independent of what the algorithmic implementation or biological realization of this specification may be [Marr 1982]. A variety of recent systems have sought to apply this insight to build artificial models of visual scene perception [Gothoskar et al. 2021; Kulkarni et al. 2015; Mansinghka et al. 2013]. Ahead of time, the authors of these systems define a probability distribution over scenes in their target domain, expressing their prior belief about the statistics of natural scenes. When given an input image, the systems condition their prior distribution on having observed that image. They then apply Bayes’ rule to infer the posterior distribution of scenes that best explain the image. Such models can account for a variety of perceptual phenomena in humans, including some illusions [Geisler and Kersten 2002; Weiss et al. 2002].

Probabilistic programming languages (PPLs). PPLs are domain-specific languages that enable users to express complicated probability distributions by implementing generative processes that sample random variables as they execute [Carpenter et al. 2017; van de Meent et al. 2018]. For example, we could describe the distribution $p(\text{Image} \mid \text{Light, Color})$ in a PPL by writing a small graphics engine that randomly samples camera parameters, and then uses those parameters to render an image of the scene using the given light and color. The PPL compiles this generative procedure into a function q that evaluates the joint probability density of a given concrete instantiation of the random variables. PPLs also allow users to express conditional distributions by fixing the values of the observed random variables.

Performing inference using MCMC sampling. Once we have a way to evaluate the conditional probability density, we would like to get a concrete sense of the distribution — for example, approximate

its mean and variance — by drawing samples from it. A standard technique for sampling from a probability distribution is the Markov Chain Monte Carlo (MCMC) algorithm: We perform a random walk over the domain of q , choosing transition probabilities such that the random walk’s stationary distribution is proportional to q . For distributions with continuous domains, a common choice of transition probabilities is given by the Hamiltonian Monte Carlo (HMC) algorithm [Duane et al. 1987; Neal et al. 2011]. HMC treats q as a physical potential energy well and simulates a randomly-initialized particle’s trajectory in that well for a fixed time window. Intuitively, such a particle would indeed spend longer in regions of low potential (or high probability), exactly as desired.

2.3 Differentiating inference

Now that we can approximate expectations taken over conditional probability distributions, we would like to differentiate these expectations with respect to parameters of the distribution. Fortunately, because each step of the HMC random walk is given by a physics simulation with continuous dynamics, we can simply apply automatic differentiation to compute derivatives of *samples* with respect to parameters of the probability distribution.¹

In the language of statistical machine learning, this technique is analogous to the “reparametrization trick” [Kingma and Welling 2014] or “pathwise gradients” technique [Mohamed et al. 2020] where the “reparametrization” or “path” maps HMC’s random inputs (i.e. the particle’s random initializations) to a sample from the target distribution. Indeed, to the best of our knowledge, this technique was first proposed by the ML community, in the context of variational inference [Salimans et al. 2015]. More recently, it has been used to train energy-based models [Dai et al. 2019; Du et al. 2021; Vahdat et al. 2020; Zoltowski et al. 2020], for Bayesian learning [Zhang et al. 2021], and to optimize the hyperparameters of HMC itself [Campbell et al. 2021].

In the above settings, the target probability distribution is typically parametrized by a black-box model (e.g. a deep neural network) whose weights are to be optimized. Our work, in contrast, applies these ideas to differentiate through richly-structured generative models expressed in PPLs (with embedded renderers/simulations), focusing specifically on perceptual models.

2.4 Space-efficient differentiation via reversible dynamics

The algorithm as described above does not scale to our setting because it uses too much memory. Backpropagation requires storing all the intermediate values computed by the target function, and HMC sampling produces a tremendous amount of intermediate data because it has two nested loops (ranging over samples and steps of the physics simulation). For our applications, which involve rendering within the generative model, HMC’s total memory usage quickly multiplies beyond what is practical. Our solution is to apply “reversible learning,” a technique originally developed for hyperparameter optimization in deep learning [Maclaurin et al. 2015], also

¹HMC has one non-differentiable component: the “accept/reject” step or “Metropolis-Hastings correction,” which accounts for numerical imprecision in the physics simulation. Our work, along with the others referenced in this section, elides this step to make the sampler fully differentiable, even though it comes at the cost of slightly biasing the samples.

later used in variational inference [Zhang et al. 2021]. Rather than *storing* intermediate values, we *recompute* them dynamically during backpropagation by running the physical simulation in reverse, backwards through time.

3 DIFFERENTIABLE PROBABILISTIC PROGRAMMING

Section 2 reviewed the cognitive-science and algorithmic foundations of our work. Now, we will briefly put aside questions of perception, and provide a tour of our key tool: the differentiable probabilistic programming language, which augments regular PPLs with the ability to differentiate through inference. To keep our discussion concrete, we will adopt the following small example problem:

In summer, the daily high temperature T in San Francisco is distributed normally, with mean 70° and standard deviation 5° . Your old thermometer reports a measurement M of the true temperature, with some added Gaussian noise of $\sigma = 2^\circ$.

- (1) On a hot day, your thermometer reports 100° . Knowing your prior belief that $T \sim N(70, 5)$ and $M \sim N(T, 2)$, what do you think the true temperature is?
- (2) What measurement would the thermometer have to report for you to believe the temperature is truly 100° ?

Question (1) is a standard Bayesian inference problem that seeks to invert the observed measurement into an inferred temperature; it is easily solved by existing PPLs. Question (2) is the “inverse inverse problem” that corresponds to (1) and requires our differentiable PPL. (In the same way, we will later treat illusion synthesis as an “inverse inverse problem” corresponding to inverse graphics.)

In this simple Gaussian setting, we can analytically solve both problems in closed form. For Question (1), we have $E[T \mid M = 100^\circ] = (2780/29)^\circ \approx 95.86^\circ$, and for question (2), the solution to the equation $E[T \mid M = m] = 100^\circ$ is $m = 524/5^\circ = 104.8^\circ$. Below, we will use our differentiable PPL to approximate these values numerically.

3.1 Writing a generative model

The first step is to encode our problem setup into a probabilistic generative model. Following languages like Anglican [Wood et al. 2014], we provide two important primitives in our PPL: **sample**, which yields a fresh, independent sample from a given distribution, and **observe**, which conditions the generative process on a sample from a distribution being equal to a given observed value.

```
def model(M):
    sample T ~ N(70, 5)
    observe M from N(T, 2)
    return T
```

Here, we define a model that can be conditioned on M . The model samples a true temperature T from the given distribution, then observes that the noisy measurement is equal to M . Finally, it returns T to signal that it is the value to be inferred.

This generative process implicitly defines a joint probability distribution over the random variables in the model. In particular, applying Bayes’ rule, we have $p(t, m) = p(t)p(m | t) = \Phi\left(\frac{t-70}{5}\right)\Phi\left(\frac{m-t}{2}\right)$ where Φ is the Gaussian probability density function. More generally, each `sample` and `observe` statement appends a multiplicative factor to the joint density. The PPL automatically compiles the generative process into a function that evaluates the (unnormalized) conditional density, given values for the latent random variables and the observations on which to condition.

3.2 Performing MCMC inference using HMC

Our language provides a function `hmc_sample`, which takes as input the HMC hyperparameters (N , the number of samples to take; L , the number of iterations of the physical simulation; and ϵ , the epsilon value used by the simulation’s integrator) and the observation on which to condition. The appropriate HMC hyperparameters vary by model, and selecting them typically requires some trial and error.

The function `hmc_sample` performs HMC inference and returns an array of N samples from the conditional distribution (of the random variable T , in our model). Typically, it is helpful to skip the first few samples, which are biased by the sampler’s “burn-in period,” during which the initial conditions’ influence has not yet been washed out by the random walk. We skip the first 100 samples for this reason.

```
samples = hmc_sample(
    N=1000, L=300, eps=0.01, skip=100,
    M=100 # observation on which to condition
)
print(samples.mean())
```

This outputs the approximation $E[T | M = 100] \approx 95.959^\circ$ (the analytical solution is 95.86°).

For brevity, from here on we will elide the HMC hyperparameters and the `.mean()` by defining the convenience function `infer`, which simply calls `hmc_sample` with appropriate hyperparameters and takes the mean of the resulting samples. Thus, the above lines of code can be shortened to `print(infer(M=100))`.

3.3 Solving the “inverse inverse problem” by gradient descent

Finally, we are ready to solve the “inverse inverse problem”: finding an m such that the inferred value of T is exactly 100° . To compute this value by gradient descent optimization, we must write down an optimization target. Here, we seek to minimize the difference between the inferred $E[T | M = m]$ and the desired value of 100° .

```
def loss(m):
    return (infer(M=m) - 100) ** 2
```

Because `infer` is end-to-end differentiable (see Sections 2.3 and 2.4), we can simply apply automatic differentiation and backpropagate through the computation of `loss`. After 100 steps of gradient descent with a step size of 0.1, the optimization converges, and we obtain our solution $m = 104.80379^\circ$ (the analytical solution is 104.8°).

4 APPLICATIONS

We are now ready to apply differentiable PPLs to find adversarial examples for probabilistic models of perception by gradient descent. The computations described below were implemented in Python using the JAX automatic differentiation library and run on an NVIDIA TITAN X GPU. Each result is reproducible in minutes using the source code included in the supplementary materials and online at <https://people.csail.mit.edu/kach/dpp-dpp/>.

4.1 Color constancy

We begin with our motivating example of synthesizing new color constancy illusions like “The Dress.” To do so by “inverse inverse rendering,” we need a probabilistic model and an optimization target to minimize.

Our probabilistic model samples scenes that contain colored objects and a single light source. For the light source, we randomly sample a color temperature from $N(6500K, 1000K)$ and uniformly select its brightness from a large range. For the objects, we choose to render colored wedge-top pencil erasers, for three reasons: First, erasers are highly Lambertian, which minimizes lighting cues given away by specular highlights. Second, the distinctive shape of wedge-top erasers makes it easy to recognize the objects as erasers. Finally, erasers come in a wide variety of colors, which allows us to use a simple uniform prior over object color. We fix all other scene parameters, such as the positions of the camera and objects.

To condition our model on an input, we follow Kulkarni et al. [2015] and add a small amount of pointwise Gaussian noise at each pixel of the rendered image, then assert that the result is the observed image. In our PPL, all of this is expressed as follows:

```
def model(observed_img):
    sample temp ~ N(6500, 1000)
    sample brightness ~ U(0, 2)
    light = planck_to_RGB(temp) * brightness
    sample color1[r,g,b], color2[r,g,b] ~ U(0, 1)

    img = render(light, color1, color2)
    observe img from N(observed_img, 0.1)
    return color1, color2
```

Finally, for our optimization target, we follow the usual “adversarial examples” recipe: we optimize for an image that causes the model to infer colors as different from the true colors as possible.²

```
def loss(light, true_color1, true_color2):
    perceived_color1, perceived_color2 =
        infer(render(light, true_color1, true_color2))
    return
        -length(true_color1 - perceived_color1) +
        -length(true_color2 - perceived_color2)
```

Why should this optimization target yield *diverse* percepts, as opposed to simply consistently-wrong ones? Recall that human viewers vary in their perceptual priors, as evidenced by “The Dress” [Wallisch 2017]. However, our model only has a single, fixed-but-reasonable prior. We thus expect viewers whose priors match our model to be “fooled” by the adversarial examples, and viewers with

²Our implementation adds an additional term to the loss that encourages `color1` and `color2` to be different, which yields more visually interesting results.

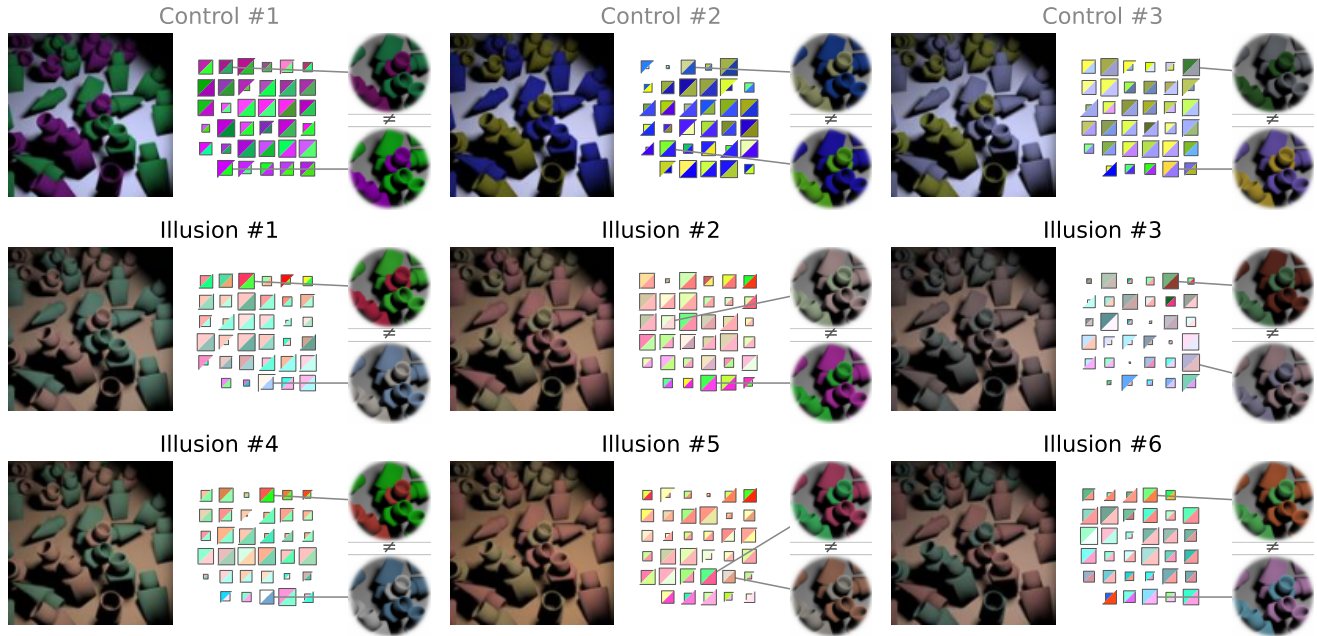


Figure 2: Color constancy illusions created by our method (Section 4.1). Each plot shows the image we presented to participants in our study, along with swatches representing the colors that they reported for the two kinds of erasers. The size of each half-square is proportional to the participant’s self-reported confidence in that color response. Selected high-confidence responses ($\geq 7/10$ for both colors) are shown in circled insets, re-lit in neutral light as the participants themselves saw in the guide display when picking colors. For controls, we selected the most distant pairs in CIELAB space; for illusions, we handpicked responses to represent significant perceptual modes. The reported colors for the illusions vary dramatically in hue, while the reported colors for the controls (top row) are tightly clustered among perceptually similar hues, only varying slightly in saturation. Note: A small fraction of participants swapped the two colors in their responses. This does not affect the experiment’s results, so we present the data without manually correcting their errors. (This figure is best viewed on a high-resolution color display.)

different priors to perceive the colors correctly. Indeed, when we evaluated our illusions with human viewers, we observed diverse responses rather than consistent ones, as discussed below.

Evaluation. Because phenomena like “The Dress” depend on diversity in perceptual priors, they are not apparent to individuals in isolation: they are only revealed when a group of people disagree about their percepts. Thus, to evaluate our illusions, we conducted an online human subject study.³

First, we performed the optimization described above for 30 random seeds. For most seeds, gradient descent converged to one of three “classes” of illusions. We hand-picked two representative samples from each class to evaluate, for a total of 6 illusions. Next, we recruited 60 naïve participants and presented each one with our 6 illusions in randomized order. Randomly-selected controls (images with poor loss in our model) were interspersed every 3 images. Participants reported the colors of the two kinds of erasers using color pickers. They also reported their confidence using sliders labeled “just guessing” to “very confident.” A guide display re-rendered their

selections under neutral lighting in real time. Participants were instructed to disable blue-light filters and the study was conducted during daytime hours in the continental US. Only one participant self-reported any color vision deficiency. We discarded responses from participants who failed attention checks (e.g. “what did the sliders measure?”), leaving 35 reliable responses, shown in Figure 2. Among these responses, we found that the colors reported for the controls clustered tightly among perceptually similar colors, while the colors reported for the illusions indeed varied widely, as desired.

Other approaches. We end this section with a brief discussion of related work on synthesizing Dress-like illusions. Witzel and Toscani [2020] transplant the original colors of “The Dress” onto new objects and show that this preserves the illusion. However, their method cannot find new “color schemes” as ours does. Wallisch and Karlovich [2019] photograph colorful slippers and white socks under colored light that makes the slippers look gray. Most viewers correctly perceive colored slippers with white socks; however, some instead perceive gray slippers with colored socks. This illusion was cleverly handcrafted with respect to specific qualitative priors (i.e. socks are typically white), and required careful manual tuning to perfect. In comparison, our method automatically and

³This study was conducted with IRB approval, and all subjects were compensated for their time.

efficiently optimizes illusions under principled natural priors. Finally, our prior work explores a similar effect in *speech* perception, the “Laurel/Yanny” illusion, by using heuristic search and large-scale crowdsourcing to find new audio clips that exhibit the same phenomenon [Chandra et al. 2021]. Our present method directly creates illusions, instead of finding them among natural examples.

4.2 Size constancy

Another “constancy” in human visual perception is size constancy: we infer objects to be the same size, even when they are viewed from different perspectives. Famous illusions of size constancy include the Ponzo illusion and the “Shepard Tables” [Shepard 1994]. Here, we try to produce illusions of *forced perspective*, where an unusual camera angle is used to make an object appear larger or smaller (or farther or nearer) than it truly is. This technique is often used by architects to enhance a building’s grandeur; oppositely, it is often used by tourists to take souvenir photographs of themselves appearing to hold monuments in their hands.

First, we create a generative model for scenes containing a set of reasonably-sized tables, observed by a randomly-initialized camera. As in previous examples, the model is then conditioned on an observed image. (In this model, we express “images” as ordered sequences of vertices representing a 2D projection of the tables’ corners.) We seek to infer each table’s position and size.

```
def model(observed_tables[]):
    # Camera position (cylindrical coordinates)
    sample r ~ U(8,16);  $\theta \sim U(0,2\pi)$ ; h ~ N(0,3)
    camera = make_camera_at(r,  $\theta$ , h).look_at(origin)

    for i, obs in observed_tables:
        sample size[x,y,z] ~ U(.5,4); pos[x,y] ~ N(0,2)
        tables[i] = make_table(size, pos)
        image = perspective_project(camera, tables[i])
        observe obs from N(image, 0.01)
    return tables
```

Next, we express our optimization target. We are looking for an arrangement of tables and camera parameters such that the first table *appears* nearer than it truly is: that is, such that the position inferred by our model is closer to the origin than the true position with which the image was generated.

```
def loss(camera, true_tables[]):
    observed_tables =
        perspective_project(camera, true_tables)
    inferred_tables = infer(observed_tables)

    return
        length(inferred_tables[0].position) -
        length(true_tables[0].position)
```

Finally, we optimize this loss by gradient descent. Some examples of optimized illusions are shown in Figures 3a and 3b. Both our model, and human intuitions, are “fooled” by these images.

4.3 Face perception

For our last application, we were inspired by two effects related to human face perception. First, Troje and Siebeck [1998] observe that merely changing the direction of lighting on a human face can

change a viewer’s perception of the face’s orientation. Second, Oliva et al. [2006] show that small changes to a carefully-constructed “hybrid image” can change a viewer’s perception of the face’s expression, an effect previously observed by Livingstone [2000] in relation to the Mona Lisa. Hoping to combine these two effects to design a new illusion, we asked: *Can we find faces that appear to change their expressions in different lighting conditions?*

Following Kulkarni et al. [2015], we built a simple probabilistic model of face perception using the Basel Face Model [Geric et al. 2018], a 3D morphable model that outputs meshes of human faces given low-dimensional latent vectors encoding identity and expression. Our generative model samples latent vectors for the morphable model independently and shines a randomly-oriented directional light on the output mesh. This resulting scene is then rendered using the SoftRas differentiable rendering algorithm [Liu et al. 2019]. As in Section 4.1, we observe an input image by adding pointwise Gaussian noise to each pixel.

```
def model(observed_img):
    sample identity[0:n], expression[0:n] ~ N(0, 1)
    mesh = BaselFaceModel(identity, expression)
    sample lighting_dir[x,y,z] ~ N(0, 1)
    img = SoftRas(mesh, lighting_dir)
    observe img from N(observed_img, 0.1)
    return expression
```

Some examples of unconditional samples from this model are shown in the top row of Figure 3c. So far, the facial expressions appear mostly the same even if the lighting direction is changed. Indeed, our model is easily able to infer the expressions in all of these images.

Next, we wrote an optimization target to search for a face mesh and lighting direction such that the face “looks” different to our model if the lighting direction is flipped across the origin:

```
def loss(mesh, lighting_dir):
    expr1 = infer(SoftRas(mesh, +lighting_dir))
    expr2 = infer(SoftRas(mesh, -lighting_dir))
    return -(expr1 - expr2) ** 2
```

Finally, we performed the necessary optimization by gradient descent. We ran the optimization separately from 100 random seeds. Approximately 10% of the results were compelling to our eyes. Our top three illusions are shown in Figure 3c. For these, we were able to reproduce the effect in the physical world by 3D-printing the optimized meshes and shining a flashlight on the printed objects.

Of the remaining 90%, the most common failure mode was a mismatch between typical human face perception and our model. In unusual lighting conditions our model infers “correct” parameters more reliably than humans, thus missing opportunities to create illusions. Additionally, the Basel Face Model was designed using data mostly from European faces, so our generative model does not match the true distribution of faces in the world. Finally, some illusions did not work when 3D printed because of shadows, which SoftRas does not account for. We expect that by tuning the model to better match humans, one could obtain a higher illusion “yield.”

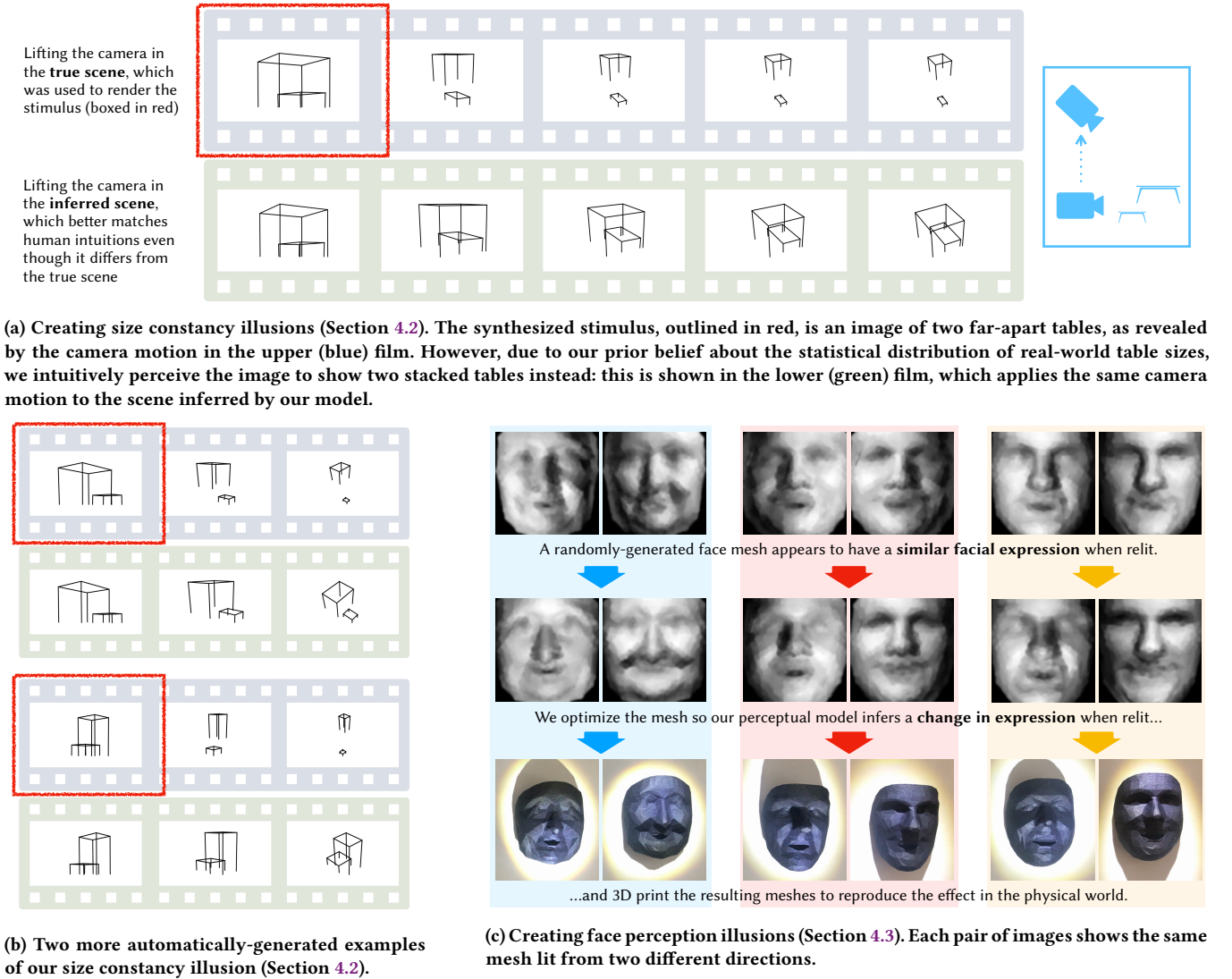


Figure 3: More illusions created by our method.

5 FUTURE WORK

Discrete random variables. Our method does not yet support discrete random variables. For example, we cannot model figure-ground reversals as in the Rubin vase illusion, because we cannot infer the binary choice between “faces” and “vase.” One promising solution is to make a differentiable version of an HMC variant like “reflective/refractive HMC” [Mohasel Afshar and Domke 2015], which is already used in some PPLs to allow mixing discrete and continuous random variables [Zhou et al. 2019]. Alternatively, we could apply recent advances in differentiating integrals (or expectations) containing discontinuities [Bangaru et al. 2021].

From visual illusions to visual applications. The graphics community has long applied insights from perception to create not only visual illusions [Chi et al. 2008; Chu et al. 2010; Ma et al. 2013; Okano et al. 2010; Oliva et al. 2006], but also perceptually-inspired

image processing algorithms [Bousseau et al. 2011; Hertzmann 2020; Huberman and Fattal 2015; Khan et al. 2006; Ritschel et al. 2008; Toler-Franklin et al. 2007; Walton et al. 2021]. These works span a wide variety of specialized methods, each targeting a particular aspect of perception. In this paper, we offer a single, general approach to illusion synthesis, which can be used to play with a whole host of perceptual mechanisms. While we were initially motivated by illusions, our approach should conceptually extend to any graphics task that is naturally posed as optimization over perceptual inference.

Beyond vision. The “inverse inverse” view of depiction straightforwardly extends to Bayesian models of cognition *beyond* perception, such as models of intuitive physics [Battaglia et al. 2013] and social cognition [Baker et al. 2009]. In future work, we thus envision posing animation as “inverse inverse simulation,” storytelling

as “inverse inverse planning,” and so on. More generally, we view differentiable PPLs as a new tool in the computational cognitive science toolbox, providing gradients that help probe human cognition in the same way that we currently probe neural networks.

6 CONCLUSION

We introduced a differentiable probabilistic programming language, which enables efficiently backpropagating through MCMC inference. Then, we used our new tool to generate a variety of visual illusions by “inverse inverse graphics”: that is, by modeling vision as Bayesian inference and using gradient descent to find “adversarial examples” that fool those models.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their feedback, as well as Gregory Valiant for early discussions about “The Dress,” Max Siegel and Thomas O’Connell for advice on conducting psychophysical experiments, and Liane Makatura for assistance with 3D printing. This research was supported by NSF Grants #2105806, #CCF-1231216, #CCF-1723445 and #CCF-1846502, ONR Grant #00010803, the Hertz Foundation, the Paul & Daisy Soros Fellowship for New Americans, and an NSF Graduate Research Fellowship under Grant #1745302.

REFERENCES

- Chris L Baker, Rebecca Saxe, and Joshua B Tenenbaum. 2009. Action understanding as inverse planning. *Cognition* 113, 3 (2009), 329–349. <https://dspace.mit.edu/handle/1721.1/60852>
- Sai Bangaru, Jesse Michel, Kevin Mu, Gilbert Bernstein, Tzu-Mao Li, and Jonathan Ragan-Kelley. 2021. Systematically Differentiating Parametric Discontinuities. *ACM Trans. Graph.* 40, 107 (2021), 107:1–107:17. <https://people.csail.mit.edu/sbangaru/projects/teg-2021/teg-2021.pdf>
- Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. 2013. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences* 110, 45 (2013), 18327–18332. <https://www.pnas.org/content/pnas/110/45/18327.full.pdf>
- Adrien Bousseau, Emmanuelle Chapoulie, Ravi Ramamoorthi, and Maneesh Agrawala. 2011. Optimizing environment maps for material depiction. In *Computer graphics forum*, Vol. 30. 1171–1180. <https://dl.acm.org/doi/abs/10.1111/j.1467-8659.2011.01975.x>
- David H Brainard and William T Freeman. 1997. Bayesian color constancy. *JOSA A* 14, 7 (1997), 1393–1411. http://people.csail.mit.edu/billf/publications/Bayesian_Color_Constancy.pdf
- Andrew Campbell, Wenlong Chen, Vincent Stimper, Jose Miguel Hernandez-Lobato, and Yichuan Zhang. 2021. A Gradient Based Strategy for Hamiltonian Monte Carlo Hyperparameter Optimization. In *International Conference on Machine Learning*. PMLR, 1238–1248. <http://proceedings.mlr.press/v139/campbell21a/campbell21a.pdf>
- Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2017. Stan: A probabilistic programming language. *Journal of statistical software* 76, 1 (2017), 1–32. <https://www.jstatsoft.org/article/view/v076i01>
- Kartik Chandra, Chuma Kabaghe, and Gregory Valiant. 2021. Beyond Laurel/Yanny: An Autoencoder-Enabled Search for Polyperceivable Audio. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics, Online, 593–598. <https://doi.org/10.18653/v1/2021.acl-short.75>
- Ming-Te Chi, Tong-Yee Lee, Yingge Qu, and Tien-Tsin Wong. 2008. Self-Animating Images: Illusory Motion Using Repeated Asymmetric Patterns. *ACM Trans. Graph.* 27, 3 (aug 2008), 1–8. <https://doi.org/10.1145/1360612.1360661>
- Hung-Kuo Chu, Wei-Hsin Hsu, Niloy J Mitra, Daniel Cohen-Or, Tien-Tsin Wong, and Tong-Yee Lee. 2010. Camouflage images. *ACM Trans. Graph.* 29, 4 (2010), 51–1. <https://dl.acm.org/doi/abs/10.1145/1833349.1778788>
- Bo Dai, Zhen Liu, Hanjun Dai, Niao He, Arthur Gretton, Le Song, and Dale Schuurmans. 2019. Exponential Family Estimation via Adversarial Dynamics Embedding. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2019/file/767d01b4bac1a1e8824c9b9f7cc79a04-Paper.pdf>
- Yilun Du, Shuang Li, B. Joshua Tenenbaum, and Igor Mordatch. 2021. Improved Contrastive Divergence Training of Energy Based Models. In *Proceedings of the 38th International Conference on Machine Learning (ICML-21)*. <https://arxiv.org/abs/2012.01316>
- Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. 1987. Hybrid Monte Carlo. *Physics letters B* 195, 2 (1987), 216–222. <https://archive.org/download/wikipedia-scholarly-sources-corpus/10.1016%252F0361-9230%252887%252990129-8.zip/10.1016%252F0370-2693%252887%252991197-X.pdf>
- Frédo Durand, Maneesh Agrawala, Bruce Gooch, Victoria Interrante, Victor Ostromoukhov, and Denis Zorin. 2002. Perceptual and artistic principles for effective computer depiction. *SIGGRAPH 2002 Course# 13 Notes* (2002). http://people.csail.mit.edu/fredo/SIG02_ArtScience/DepictionNotes2.pdf
- Gamaleldin Elsayed, Shreya Shankar, Brian Cheung, Nicolas Papernot, Alexey Kurakin, Ian Goodfellow, and Jascha Sohl-Dickstein. 2018. Adversarial examples that fool both computer vision and time-limited humans. *Advances in neural information processing systems* 31 (2018). <https://proceedings.neurips.cc/paper/2018/file/8562ae5e286544710b2e7ebe9858833b-Paper.pdf>
- Wilson S Geisler and Daniel Kersten. 2002. Illusions, perception and Bayes. *Nature neuroscience* 5, 6 (2002), 508–510. <https://www.cs.utexas.edu/~dana/NVGeisler2.pdf>
- Thomas Gerig, Andreas Morel-Forster, Clemens Blumer, Bernhard Egger, Marcel Luthi, Sandro Schönborn, and Thomas Vetter. 2018. Morphable face models: an open framework. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, 75–82. <https://ieeexplore.ieee.org/abstract/document/8373814>
- Noah D Goodman, Joshua B. Tenenbaum, and The ProbMods Contributors. 2016. Probabilistic Models of Cognition. <http://probmods.org/v2>. Accessed: 2021-10-15.
- Nishad Gothoskar, Marco Cusumano-Townner, Ben Zinberg, Matin Ghavamizadeh, Falk Pollok, Austin Garrett, Josh Tenenbaum, Dan Gutfreund, and Vikash Mansinghka. 2021. 3DP3: 3D Scene Perception via Probabilistic Programming. *Advances in Neural Information Processing Systems* 34 (2021). <https://proceedings.neurips.cc/paper/2021/file/4fc66104f8ada6257fa55f29a2a567c7-Paper.pdf>
- Aaron Hertzmann. 2020. Why do line drawings work? a realism hypothesis. *Perception* 49, 4 (2020), 439–451. <https://arxiv.org/abs/2002.06260>
- Inbar Huberman and Raanan Fattal. 2015. Reducing Lateral Visual Biases in Displays. *Computer Graphics Forum (CGF)*. https://www.cs.huji.ac.il/w-raananf/projects/lateral_biases/
- Daniel Kersten, Pascal Mamassian, and Alan Yuille. 2004. Object perception as Bayesian inference. *Annu. Rev. Psychol.* 55 (2004), 271–304. <http://mamassian.free.fr/papers/KerstenMamassianYuille04.pdf>
- Erum Arif Khan, Erik Reinhard, Roland W. Fleming, and Heinrich H. Bühlhoff. 2006. Image-Based Material Editing. *ACM Trans. Graph.* 25, 3 (jul 2006), 654–663. <https://doi.org/10.1145/1141911.1141937>
- Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. arXiv:1312.6114 [stat.ML] <https://arxiv.org/pdf/1312.6114.pdf>
- David C Knill and Whitman Richards. 1996. *Perception as Bayesian inference*. Cambridge University Press.
- Tejas D Kulkarni, Pushmeet Kohli, Joshua B Tenenbaum, and Vikash Mansinghka. 2015. Picture: A probabilistic programming language for scene perception. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4390–4399. https://mrkulk.github.io/www_cvpr15/1999.pdf
- Rosa Lafer-Sousa, Katherine L Hermann, and Bevil R Conway. 2015. Striking individual differences in color perception uncovered by ‘The Dress’ photograph. *Current Biology* 25, 13 (2015), R545–R546. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4921196/pdf/nihms686513.pdf>
- Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. 2019. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7708–7717. <https://arxiv.org/pdf/1904.01786.pdf>
- Margaret S Livingstone. 2000. Is it warm? Is it real? Or just low spatial frequency? *Science* 290, 5495 (2000), 1299–1299. https://livingstone.hms.harvard.edu/sites/livingstone.hms.harvard.edu/files/publications/2000_November17_%20Livingstone_%20Is%20It%20Warm_%20Is%20It%20Real_%20Or%20Just%20Low%20Spatial%20Frequency_.pdf
- Li-Qian Ma, Kun Xu, Tien-Tsin Wong, Bi-Ye Jiang, and Shi-Min Hu. 2013. Change blindness images. *IEEE transactions on visualization and computer graphics* 19, 11 (2013), 1808–1819. <https://cg.cs.tsinghua.edu.cn/papers/TVCC-2013-changeblindness.pdf>
- Dougal Maclaurin, David Duvenaud, and Ryan Adams. 2015. Gradient-based hyperparameter optimization through reversible learning. In *International conference on machine learning*. PMLR, 2113–2122. <http://proceedings.mlr.press/v37/maclaurin15.pdf>
- Pascal Mamassian, Michael Landy, and Laurence T Maloney. 2002. Bayesian modelling of visual perception. *Probabilistic models of the brain* (2002), 13–36. http://mamassian.free.fr/papers/mamassian_mit02.pdf
- Vikash K Mansinghka, Tejas D Kulkarni, Yura N Perov, and Josh Tenenbaum. 2013. Approximate Bayesian Image Interpretation using Generative Probabilistic Graphics Programs. In *Advances in Neural Information Processing Systems*, C. J. C.

- Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2013/file/fa14d4fe2f19414de3ebd9f63d5c0169-Paper.pdf>
- David Marr. 1982. *Vision*. W. H. Freeman and Company. <https://mitpress.mit.edu/books/vision>
- Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. 2020. Monte Carlo Gradient Estimation in Machine Learning. *J. Mach. Learn. Res.* 21, 132 (2020), 1–62. <https://arxiv.org/pdf/1906.10652.pdf>
- Hadi Mohasel Afshar and Justin Domke. 2015. Reflection, Refraction, and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2015/file/8303a79b1e19a194f1875981be5bdb6f-Paper.pdf>
- Radford M Neal et al. 2011. MCMC using Hamiltonian dynamics. *Handbook of Markov chain Monte Carlo* 2, 11 (2011), 2. <https://arxiv.org/pdf/1206.1901.pdf>
- Yu Okano, Shogo Fukushima, Masahiro Furukawa, and Hiroyuki Kajimoto. 2010. Embedded Motion: Generating the Perception of Motion in Peripheral Vision. In *ACM SIGGRAPH ASIA 2010 Posters* (Seoul, Republic of Korea) (SA '10). Association for Computing Machinery, New York, NY, USA, Article 41, 1 pages. <https://doi.org/10.1145/1900354.1900400>
- Aude Oliva, Antonio Torralba, and Philippe G Schyns. 2006. Hybrid images. *ACM Transactions on Graphics (TOG)* 25, 3 (2006), 527–532. https://stanford.edu/class/ee367/reading/OlivaTorralb_Hybrid_Siggraph06.pdf
- Tobias Ritschel, Kaleigh Smith, Matthias Ihrke, Thorsten Grosch, Karol Myszkowski, and Hans-Peter Seidel. 2008. 3D Unsharp Masking for Scene Coherent Enhancement. Vol. 27. Association for Computing Machinery, New York, NY, USA, 1–8. <https://doi.org/10.1145/1360612.1360689>
- Tim Salimans, Diederik Kingma, and Max Welling. 2015. Markov chain Monte Carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*. PMLR, 1218–1226. <http://proceedings.mlr.press/v37/salimans15.pdf>
- Roger N Shepard. 1994. Perceptual-cognitive universals as reflections of the world. *Psychonomic Bulletin & Review* 1, 1 (1994), 2–28. http://rucss.rutgers.edu/images/personal-zenon-pylyshyn/docs/transfers/shepard_space_bbs2001.pdf
- Fabian H. Sinz, Xaq Pitkow, Jacob Reimer, Matthias Bethge, and Andreas S. Tolias. 2019. Engineering a Less Artificial Intelligence. *Neuron* 103, 6 (2019), 967–979. <https://doi.org/10.1016/j.neuron.2019.08.034>
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*. <http://arxiv.org/abs/1312.6199>
- Corey Toler-Franklin, Adam Finkelstein, and Szymon Rusinkiewicz. 2007. Illustration of complex real-world objects using images with normals. In *Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*. 111–119. https://pixl.cs.princeton.edu/pubs/Toler-Franklin_2007_IOC/rgbn.pdf
- Nikolaus F Troje and Ulrike Siebeck. 1998. Illumination-induced apparent shift in orientation of human heads. *Perception* 27, 6 (1998), 671–680. <https://journals.sagepub.com/doi/pdf/10.1068/p270671#also-see-slack>
- Arash Vahdat, Evgeny Andriyash, and William Macready. 2020. Undirected graphical models as approximate posteriors. In *International Conference on Machine Learning*. PMLR, 9680–9689. <http://proceedings.mlr.press/v119/vahdat20a/vahdat20a.pdf>
- Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. 2018. An introduction to probabilistic programming. *arXiv preprint arXiv:1809.10756* (2018). <https://arxiv.org/pdf/1809.10756>
- Pascal Wallisch. 2017. Illumination assumptions account for individual differences in the perceptual interpretation of a profoundly ambiguous stimulus in the color domain: “The Dress”. *Journal of Vision* 17, 4 (2017), 5–5. https://jov.arvojournals.org/article.aspx?articleid=2617976&mbid=synd_yahoostyle
- Pascal Wallisch and Michael Karlovich. 2019. Disagreeing about Crocs and socks: Creating profoundly ambiguous color displays. *arXiv preprint arXiv:1908.05736* (2019). <https://arxiv.org/pdf/1908.05736.pdf>
- David R. Walton, Rafael Kuffner Dos Anjos, Sebastian Friston, David Swapp, Kaan Aksit, Anthony Steed, and Tobias Ritschel. 2021. Beyond Blur: Real-Time Ventral Metamers for Foveated Rendering. *ACM Trans. Graph.* 40, 4, Article 48 (jul 2021), 14 pages. <https://doi.org/10.1145/3450626.3459943>
- Yair Weiss, Eero P Simoncelli, and Edward H Adelson. 2002. Motion illusions as optimal percepts. *Nature neuroscience* 5, 6 (2002), 598–604. <https://www.nature.com/articles/nn858>
- Christoph Witzel and Matteo Toscani. 2020. How to make a #theDress. *JOSA A* 37, 4 (2020), A202–A211. <https://www.osapublishing.org/josaa/abstract.cfm?uri=josaa-37-4-A202>
- Frank Wood, Jan Willem van de Meent, and Vikash Mansinghka. 2014. A New Approach to Probabilistic Programming Inference. In *Proceedings of the 17th International conference on Artificial Intelligence and Statistics*. 1024–1032. <https://probprog.github.io/anglican/assets/pdf/wood-aistats-2014.pdf>
- Guodong Zhang, Kyle Hsu, Jianing Li, Chelsea Finn, and Roger B Grosse. 2021. Differentiable annealed importance sampling and the perils of gradient noise. *Advances in Neural Information Processing Systems* 34 (2021). <https://proceedings.neurips.cc/paper/2021/file/a1a609f1ac109d0be28d8ae112db1bbb-Paper.pdf>
- Yuan Zhou, Bradley J. Gram-Hansen, Tobias Kohn, Tom Rainforth, Hongseok Yang, and Frank Wood. 2019. LF-PPL: A Low-Level First Order Probabilistic Programming Language for Non-Differentiable Models. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 89)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.). PMLR, 148–157. <https://proceedings.mlr.press/v89/zhou19b.html>
- Zhenglong Zhou and Chaz Firestone. 2019. Humans can decipher adversarial images. *Nature communications* 10, 1 (2019), 1–9. <https://www.nature.com/articles/s41467-019-08931-6>
- David M Zoltowski, Diana Cai, and Ryan P Adams. 2020. Slice Sampling Reparameterization Gradients. In *Third Symposium on Advances in Approximate Bayesian Inference*. https://openreview.net/pdf?id=cT_RMSqVF4