Perspective Plane Program Induction from a Single Image

Yikai Li[1,2*] Jiayuan Mao[1*] Xiuming Zhang[1]
William T. Freeman[1,3] Joshua B. Tenenbaum[1] Jiajun Wu[4]
[1]MIT CSAIL [2]Shanghai Jiao Tong University [3]Google Research [4]Stanford University

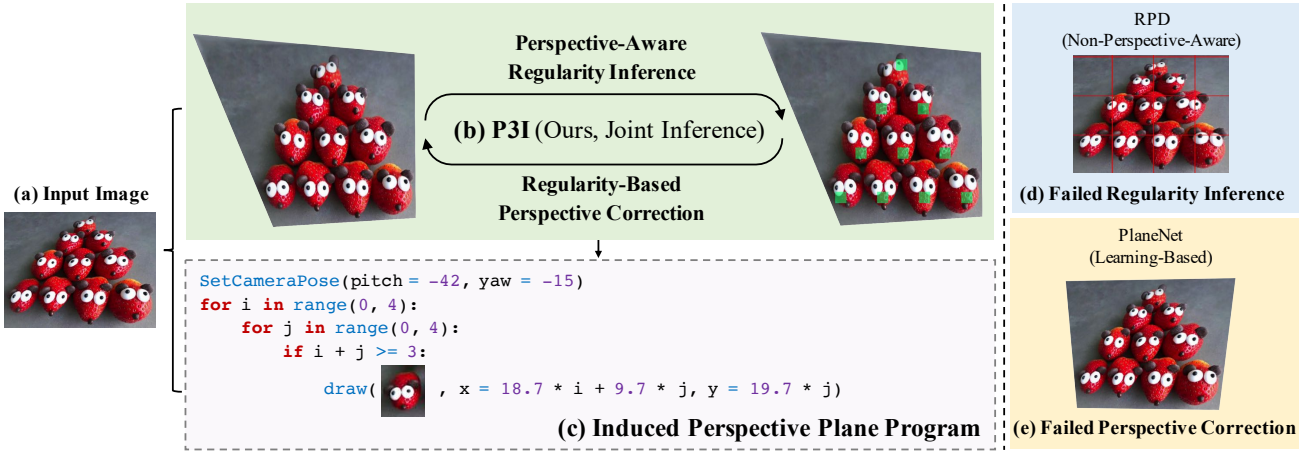

Figure 1: Perspective effects and scene structure regularity are ubiquitous in natural images (a). To detect such regularity, one may directly apply regularity structure detection (RPD) [21] to natural images, but this often fails due to the existence of perspective effects (d). Attempting to remedy this, one may perform perspective correction as an independent preprocessing step, but perspective correction often relies on line and/or vanishing point cues, and fails when such cues are missing (e). We observe that these two tasks are interconnected: image regularity serves as a new perspective correction cue, and regularity detection, in turn, also benefits from perspective correction. Thus, we propose to jointly solve perspective correction and regularity structure detection (b) by simultaneously seeking the program and perspective parameters that best describe the image (c). Project page: http://p3i.csail.mit.edu

Abstract

We study the inverse graphics problem of inferring a holistic representation for natural images. Given an input image, our goal is to induce a neuro-symbolic, program-like representation that jointly models camera poses, object locations, and global scene structures. Such high-level, holistic scene representations further facilitate low-level image manipulation tasks such as inpainting. We formulate this problem as jointly finding the camera pose and scene structure that best describe the input image. The benefits of such joint inference are two-fold: scene regularity serves as a new cue for perspective correction, and in turn, correct perspective correction leads to a simplified scene structure, similar to how the correct shape leads to the most regular texture in shape from texture. Our proposed framework, Perspective Plane Program Induction (P3I), combines search-based and gradient-based algorithms to efficiently solve the problem. P3I outperforms a set of baselines on a collection of Internet images, across tasks including camera pose estimation, global structure inference, and down-stream image manipulation tasks.

1. Introduction

From a single image in Fig. 1, humans can effortlessly induce a holistic scene representation that captures both local textures and global scene structures. We can localize the objects in the scene (the "strawberry mice"). We also see the global scene *regularities*: the mice collectively form a 2D lattice pattern with a triangular boundary. Meanwhile, we can estimate the camera pose: the image is shot at an elevation of roughly 45 degrees.

Building holistic scene representations requires scene understanding from various perspectives and levels of detail: estimating camera poses [26, 9, 3], detecting objects in the scene [17, 21], and inferring the global structure of scenes [15, 29]. Humans are able to resolve these inference tasks simultaneously. The estimation of global camera pose guides the localization of individual objects and the summarization of the scene structure, such as the lattice pattern in Fig. 1(a), in a top-down manner. Meanwhile, the localization of individual objects provides bottom-up cues for the inference of both scene structures and camera poses.

* indicates equal contribution.

While various algorithms have been developed to tackle each individual task, there is still a lack of studies on the integration of these methods and how they can benefit from each other. In this paper, we present the framework, Perspective Plane Program Induction (P3I), for the joint inference of the camera pose, the localization of individual objects, and a program-like representation that describes lattice or circular regularities of object placement. The inferred holistic scene representation, namely the perspective plane program, has a program-like structure with continuous graphics parameters. The key assumption is that the image, possibly captured with perspective effects, is composed of a collection of similar objects that are placed following a *regular* pattern.

The integrated inference has three advantages. First, conventional estimations of camera poses (specifically the 3D rotations) mainly rely on geometric cues, such as straight lines [37, 3] and manually designed texture descriptors [1], or learning from human annotations [24]. Thus, they fail when no straight lines or textual regions can be detected and exhibit poor generalization to unseen complex scenes. In this work, P3I exploits regular structures on 2D planes to accurately estimate the camera pose. For example, in Fig. 1(b), the estimated camera pose can perspective correct the image such that all adjacent mice share roughly the same displacement.

Second, classic object localization algorithms mostly rely on human heuristics [39, 51] or require large-scale datasets [17]. In this paper, we present a complementary solution based on the similarity among objects in a single image and the global scene *regularity*. Such regularities are modeled with the proposed perspective plane programs.

Third, although graphics programs, as shown in Fig. 1(c), have been found useful for both low-level manipulation and high-level reasoning tasks [41, 15, 27], the inference is usually not done in an end-to-end manner. These methods work on estimated or known camera parameters and object detection results by off-the-shelf tools, and formulate the inference problem as a pure program synthesis problem in a symbolic space. This restricts the applicability of these algorithms to natural images. By contrast, in this work, P3I removes such dependencies by formulating the whole problem as a joint inference task of the camera pose, object locations, and the global scene structure. We show that our model can infer holistic perspective plane programs from a single input image without extra tools for any of the tasks.

We collect a dataset of Internet images, namely the Nearly-Regular Patterns with Perspective dataset (NRPP), for evaluation. The dataset contains non-fronto-parallel images that are composed by a set of objects organized in regular patterns. P3I is evaluated on NRPP in two metrics: accuracy of camera pose estimation and that of graphics programs. Our model outperforms all baselines that tackle these problems separately. Moreover, we show how such holistic representations can be used to perform lower-level image manipulation tasks such as image inpainting and extrapolation. Our approach outperforms both learning-based and non-learning-based baselines designed for such tasks.

2. Related Works

Camera pose estimation and shape from texture. The idea of inferring camera poses (the perspective angles) from regularity draws deep connection to the classic work on shape from texture, dated back to the 80's [8, 2, 28, 33]. The key assumption here is the uniform density assumption (texels are uniformly distributed). Thus, a perspective view of slanted textured surface will show systematic changes in texture density, area, the aspect ratios. Blostein et al. [8] and Aloimonos [2] recover the slant and tilt of the camera for images containing a single plane, while Malik and Rosenholtz [28] consider curved surfaces. Aiger et al. [1] finds homography transformations by running statistical analysis on the detected regions of textures. Furthermore, Ohta et al. [33] combines perspective from texture and the estimation of vanishing points. Recently, there have been attempts that leverage deep learning for 3D line, vanishing point, and plane estimation [24, 3, 49]. While these methods focus on camera pose estimation, In this work, we propose to jointly tackle the problem with object localization and scene structure prediction via programs.

Program induction and inverse graphics. Procedural modeling is well-established topic in computer graphics, mostly for indoor scenes [40, 23, 32] and 3D shapes [22, 38]. Recently, researchers propose to augment such algorithm with deep recognition networks. Representative works include graphics program induction for hand-drawn images [15], 3D scenes [27], primitive sets [35], and markup code [12, 7]. However, they only work on synthetic images in a constrained domain, while here we study natural images. SPIRAL [16], and its follow-up SPIRAL++ [30], both used reinforcement learning to discover 'doodles' that are later used to compose the image. Their models are no longer restricted to constrained domains, but are also not as transparent and interpretable as symbolic program-like representations, which limits their applications in tasks that involve explicit reasoning, such as image extrapolation.

Most relevant to our papers are the work from Young et al. [45] and from Mao et al. [29], where they both used formal representations within deep generative networks to represent natural images, and later applied the representation for image editing. Unlike Young et al. [45], which requires learning semantics on a pre-defined dataset of semantically similar images, our P3I learns from a single image, following the spirit of internal learning [36]. Unlike Mao et al. [29], which assumes a top-down view and fails on images with perspective distortions, P3I simultaneously infers the camera pose, object locations, and scene structures.

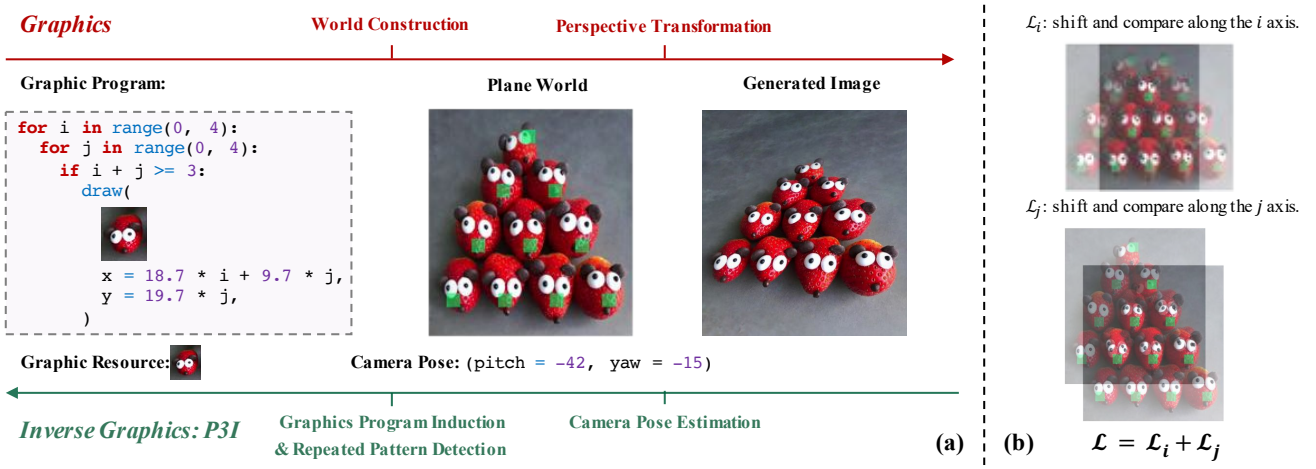

Figure 2: **(a)** Our model P3I solves an *inverse graphics* problem. Given an input image, P3I jointly infers the camera pose, object locations, and the global scene regularity, which is an inversion of a simplified graphics pipeline. **(b)** We compute the fitness of a program on the image based on a shift-and-compare routine, which we illustrates on a lattice pattern case.

Image manipulation. Image manipulation is most commonly studied in the context of image inpainting. Inpainting algorithms can be based on pixels, patches, or global image representations. Pixel-based methods [4, 5] and patch-based methods [14, 6] perform well when the missing regions are small and local, but cannot deal with cases that require high-level information beyond background textures. Darabi et al. [11] extended patch-based methods by allowing additional geometric and photometric transformations on patches but ignored global consistency among patches. Huang et al. [18] also used perspective correction to help patch-based inpainting, but their algorithm relies on the vanishing point detection by other methods. By contrast, P3I estimates the camera parameters based on the global regularity of images.

The advances of deep nets has led to many impressive inpainting algorithms that integrate information beyond local pixels or patches [19, 44, 46, 25, 47, 50, 43]. Most relevant to our work, Xiong et al. [42] and Nazeri et al. [31] proposed to explicitly model contours to help the inpainting system preserve global object structures. Compared with them, P3I manipulates the image based on its latent perspective plane program. Thus, we can preserve the global scene *regularity* during manipulation and requires no extra training images.

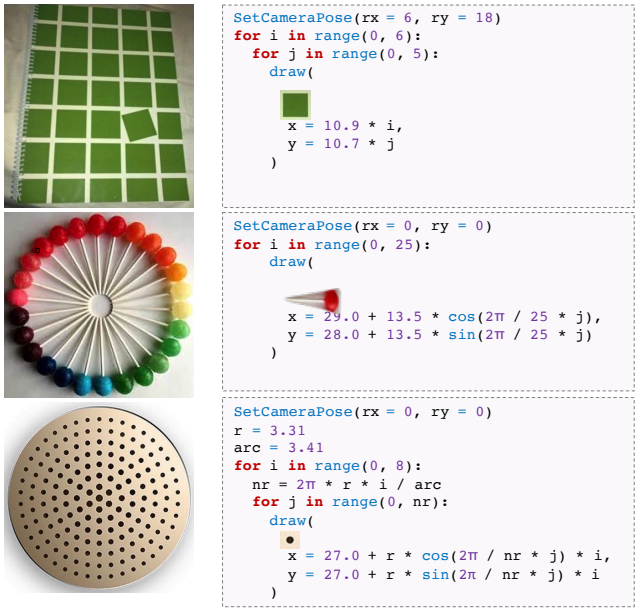

Figure 3: Example programs inferred by P3I. Our model can perform joint inference of camera pose, object localizations, and global scene structures of images having different regularity patterns: (a) lattice, (b) circular, and (c) a hybrid structure composed by a circular structure and a linearly repeated one.

3. Perspective Plane Program Induction

The proposed framework, Perspective Plane Program Induction (P3I), takes a raw image as input and infers a perspective plane program that best describes the image. In Section 3.1, we first present the domain-specific language of the program that we use to describe the scene and the camera, by walking through a graphics pipeline that generates a natural image. In Section 3.2, we present our algorithm for the inversion of such perspective plane programs and mathematically formulate it as a joint inference problem. Finally, in Section 3.3, we present a hybrid inference algorithm to perform the inference efficiently. Implementation details are supplied in Section 3.4.

3.1. Perspective Plane Programs

We introduce our perspective plane programs by walking through the graphics pipeline that generates a natural image. Suppose that the scene is composed of a collection of visually similar objects (or generally, patterns) placed regularly on a 2D plane. Thus, the generative process of the image can be divided into three parts: first, modeling of individual objects; second, global scene regularities such as the lattice

patterns in Fig. 3 (a) or circular patterns in Fig. 3 (b) and (c), represented using graphics programs; and third, camera extrinsic and intrinsic parameters, which defines the projection of the 3D scene onto a 2D image plane.

Illustrated in Fig. 3, a perspective plane program consists of the primitive `Draw` command which places objects at specified positions. Such `Draw` commands appear in (possibly nested) `For`-loop and `Rotate`-loop statements, which characterizes the global scene regularity. Finally, the program specifies camera parameters with the command `SetCameraPose`. We restrict the nested loops to be at most two-level because 1) it is powerful enough to capture most 2D layouts, and 2) in perspective geometry, a two-dimensional pattern is sufficient for inferring the vanishing line of a plane (and thus the plane orientation). However, we can expand the DSL to include new patterns; the inference algorithm we present is also not tied to any specific DSL and generalizes to new patterns.

Repeated patterns. The most basic command in a perspective plane program is `Draw`. Given 2D coordinates (x, y), a single call to the `Draw` command places an object or, generally, a pattern on the 2D plane, centering at (x, y). The `Draw` commands are enclosed in (nested) loops that define lattice or circular structures. Fig. 3 illustrates the latent perspective plane programs for a set of images.

Perspective transformations. The next step of the graphics pipeline is to project the 3D space onto a 2D image plane. Since we consider only a single 2D plane in the 3D world, the resulting transformation can be modeled as a perspective transformation (which gives the name, perspective plane programs). For simplicity, we only model the 3D rotational transformations given by the camera pose and make a set of assumptions on the other intrinsic and extrinsic parameters of the camera. Details could be found in Section 3.4.

3.2. Inversion of the Graphics Pipeline

The goal of P3I is to solve an inverse graphics problem: given the generated image of the scene, we want to estimate the camera pose, infer the regular pattern, and localize the individual objects or patterns. We view this problem as finding a program P that best fit the input image I. In this section, we demonstrate how our fitness function is computed. The backward direction of Fig. 2 gives an illustration.

Taking the RGB image as the input, we first extract its visual feature using an ImageNet-pretrained AlexNet [20]. Working on the feature space makes the inference procedure more robust to local noises such as luminance and reflectance variations, compared with working with RGB pixels directly. We denote $\mathcal{F}_{\text{AlexNet}}$ as the feature extractor and $F = \mathcal{F}_{\text{AlexNet}}(I)$ as the extracted visual features.

The second step is to invert the 2D projection. Assuming a pin-hole camera model, this is done by performing an inverse perspective transformation on the feature F. Specifically, we transform the extracted feature map as a fronto-parallel feature based on the XYZ rotations rx, ry, rz:

$$F^{fp} = \text{WarpPerspective}_{-rx,-ry,-rz}(F). \tag{1}$$

Note that, ideally, the transformation should be done on the input image. However, in practice, we swap the order of perspective transformation and AlexNet feature extraction. We find that transforming the feature map provides a good approximation of extracting features on the transformed image, i.e.

$$\begin{aligned} &\text{WarpPerspective}_{-rx,-ry,-rz}(\mathcal{F}_{\text{AlexNet}}(I)) \\ \approx\ &\mathcal{F}_{\text{AlexNet}}\big(\text{WarpPerspective}_{-rx,-ry,-rz}(I)\big). \end{aligned}$$

Moreover, performing feature map transformation is more computationally efficient: we do not need to run the AlexNet multiple times for different camera parameters.

The next step is to reconstruct the scene structure and localize individual objects. This is formulated as synthesizing a program that describes the transformed canvas plane. Each candidate program in the DSL space produces a set of 2D coordinates that can be interpreted as centers of objects. We compute the loss of each program based on the similarity of objects that are located by the program.

Mathematically, we denote C as a set of 2D coordinates generated by a program graphics program P, defined on the (transformed) canvas plane. Since a perspective plane program contains at most a two-level nested loops, we denote the loop variables as i and j and view each coordinates (x, y) in C as a function of the loop variables. That is, $(x, y) = (x(i, j), y(i, j))$. The coordinate functions can be chosen to fit either lattice or circular patterns. We define the loss function as:

$$\begin{aligned} \mathcal{L} = &\sum_{i,j} \left\| F^{fp}[x(i,j), y(i,j)] - F^{fp}[x(i+1,j), y(i+1,j)] \right\|_2^2 \\ &+ \sum_{i,j} \left\| F^{fp}[x(i,j), y(i,j)] - F^{fp}[x(i,j+1), y(i,j+1)] \right\|_2^2, \end{aligned} \tag{2}$$

where $\|\cdot\|_2$ is the $\mathcal{L}_2$-norm, which computes the difference between two feature vectors at two spatial positions. In the lattice case, illustrated in Fig. 2(b), one can interpret this fitness function as the following operations: we shift the feature map by a displacement; we then compute the feature similarity between the shifted and the original feature map.

In contrast to previous work by Lettry et al. [21], which detects repeated patterns based on a lattice global structure assumption, our program-based formulation allows a more flexible and compositional way to define global scene structures and perform inference. As an example, in Fig. 3(c), our model can detect repeated patterns in a hybrid structure composed by a circular structure and a linearly repeated one.

3.3. Grid Search and Gradient-Based Optimization

We present a hybrid inference algorithm to solve the problem of finding a graphics program P that best fits the input image I. The output of the algorithm includes both

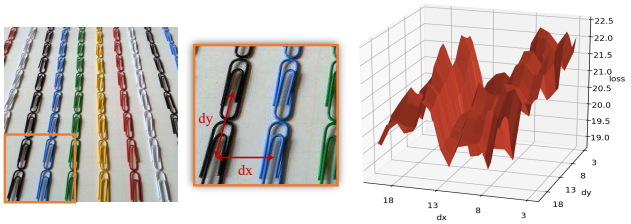

Figure 4: Loss Surface of the displacement parameter in a perspective plane program. On the left we show the input image. The regularity in the image directly leads to many cliffs on the loss surface ($dx = 10, 12.5, 15, \cdots$) and many local peaks.

the layout patterns (lattice, circular, etc.) and the parameters. This requires solving an optimization problem of choosing a discrete structure (e.g., lattice or circular) and a collection of continuous parameters (rotation angles, object locations, etc.). Previously, graphics program inference is tackled mostly by program synthesis via search in the symbolic program space [15, 29]. This search process is slow, because the symbolic space is huge, growing exponentially with respect to the number of parameters. It is often required to quantize parameters (e.g., to integers) and use heuristics to accelerate the search process. Unlike these approaches, P3I tackles such inference with a hybrid version of search-based and gradient-based optimization.

The key insight here is that both WarpPerspective transformations and feature map indexing are differentiable w.r.t. the parameters (the rotational angles and the continuous 2D coordinates), since they are both implemented using bilinear interpolation on the feature maps. Thus, the loss function (Equation 2) is differentiable with respect to all parameters in perspective plane programs, including camera poses and constants in coordinate expressions, making gradient-based optimization applicable to our inference task.

However, directly applying gradient descent on $\mathcal{L}$ remains problematic: the discrete nature of object placements makes the loss function (Equation 2) non-convex. As shown in Fig. 4, the regularity in the image leads to many cliffs and peaks (local optima). Therefore, direct application of gradient descent will get stuck at a local optimum easily.

Thus, we propose a hybrid inference algorithm to exploit the robustness of search-based inference and the efficiency of gradient-based inference. Specifically, we perform discrete search on the choice of regularity structure. Three structures are considered in this paper, as illustrated in Fig. 3: (a) lattice, (b) circular, and (c) a hybrid one. For continuous parameters, we perform grid search on a coarse scale and apply fine-grained gradient descent only locally. This is simply implemented by perform a grid search of initial parameters and performing gradient descent on each individual combination.

3.4. Implementation Details

During inference, in the grid search, the grid size for the coarse search of continuous values is 2. For lattice patterns, we do not perform search on the boundary conditions for the loop variables. Instead, the boundaries are generated based on the size of the image. In other words, we assume that the regular pattern covers the whole image plane.

Throughout the paper we consider a simplified camera model with only two rotational degree of freedom (the X-tilt and the Y-tilt). Thus, we assume that the optic axis is aligned with the image center and there is no Z-axis rotation. This is because the Z-axis rotation has been captured by the object coordinates. For example, with lattice patterns, objects can be placed along axes that are not in parallel to the X and Y axes. We do not assume a known focal length f and aspect ratio α. They cannot be recovered unequivocally from a single 2D plane, and different f and α yield to the same perspective correction and image editing results. The results shown in the paper, obtained with $f = 35$ and $\alpha = 1$, will remain the same with other f and α. Meanwhile, we ignore lens distortions, such as radial distortion. Our method can be integrated with camera calibration algorithms to correct them based on detected repeated patterns [13]

4. Experiments

We test our model on a newly collected dataset, Nearly-Regular Patterns with Perspective (NRPP), and evaluate its accuracy for camera pose estimation (Section 4.2) and repeated pattern detection (Section 4.3). We further demonstrate the model can be used to guide low-level image manipulation (Section 4.4).

4.1. Dataset

We collected a dataset of 64 Internet images that each contain a set of objects organized in regular patterns (lattice and circular). Unlike a similar dataset, Nearly Regular Patterns [21], all images in our NRPP are not fronto-parallel; Fig. 5 gives some examples. We augment the dataset with human annotations of the camera pose and object locations, in the form of 2D coordinates of object centers. This supports a quantitative evaluation for camera pose estimation and repeated object detection.

4.2. Camera Pose Estimation

We evaluate the performance of P3I on camera pose estimation from single images, compared against both learning-based and non-learning-based baselines.

Baselines. We compare P3I with three baselines. The first is AutoRectify [1], a texture-based baseline for camera pose estimation. AutoRectify statistically find homography transformation from intersects of detected ellipse regions. We decompose the output transformation matrix to get camera pose as the prediction of AutoRectify. The second is PlaneNet [24], a learning-based baseline for camera pose estimation. PlaneNet is a convolutional neural network-based

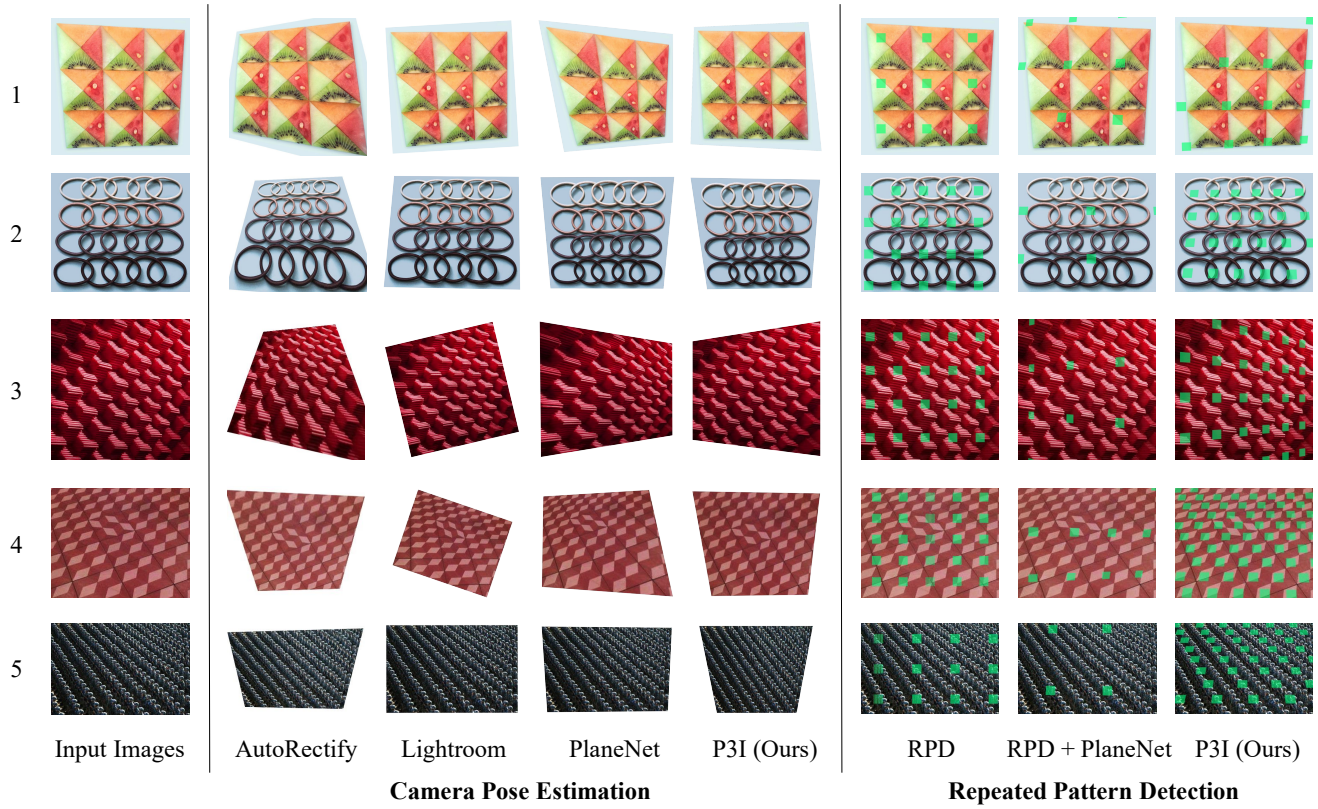

Figure 5: On the left, we show that P3I estimates the camera pose based on the global regularity of images. It outperforms AutoRectify, Lightroom and PlaneNet. Results are visualized by performing an perspective correction based on the estimated parameters. On the right, we show that P3I can perform perspective-aware repeated pattern detection, while both RPD [21] and RPD+PlaneNet fail.

Method	Camera Pose Error
AutoRectify [1]	30.54
PlaneNet [24]	23.75
P3I (Ours)	**4.54**

Table 1: Camera pose estimation. P3I outperforms texture-based baseline, AutoRectify, and the neural baseline, PlaneNet, by a remarkable margin on the NRPP dataset.

algorithm trained to detect 2D planes and their normals in 3D scenes from RGB images. Since all images in our dataset contain only one plane, we select the largest plane detected by PlaneNet and use its normal vector to compute the camera pose as the prediction of PlaneNet. Across all experiments, we use the PlaneNet model pretrained on ScanNet [10]. We also compare our results qualitatively with the auto-perspective tool provided by Adobe Lightroom.

Metrics. We evaluate the accuracy of the estimated camera poses, i.e., the camera orientation, by calculating their $\mathcal{L}_1$ distance to the human-annotated pose using Rodrigues' rotation formula. All error metrics are computed in degrees and averaged over all images in the dataset.

Results. We first present qualitative results in Fig. 5, visualizing the predictions of P3I, PlaneNet, and Adobe Lightroom. Our model achieves near-perfect estimations of the camera pose, whereas other baselines lead to incorrect perspective correction, possibly due to the absense of straight line cues. Quantitatively, as shown in Table 1, our model also outperforms AutoRectify and PlaneNet by a significant margin. Since Adobe Lightroom does not provide numerical values of the estimated camera pose, we are unable to make quantitative comparison with it.

These results indicate that our model can successfully use cues from global scene regularities to guide the inference of the camera pose. This differs from traditional visual cues such as vanishing points or straight lines.

4.3. Repeated Pattern Detection

The task of repeated pattern detection is to localize individual objects or patches in an image, assuming that these patches have similar visual appearances.

Baselines. We compare our algorithm with a non-learning-based algorithm, RPD [21], designed for localizing objects that form lattice patterns. We use a subset of NRPP of 56 images (lattice patterns only), each of which contains only lattice patterns. Because the original RPD algorithm is not designed to handle non-fronto-parallel images, we also augment RPD with perspective correction based on the camera pose estimated by PlaneNet.

Metrics. The output of both RPD and P3I is a list of object centroids, which we compare against the ground-truth annotations of object centroids in the image. Two complementary metrics are used: average distance from all detected centroids to their nearest ground-truth centroids ("Detected to

Method	Detected to GT	GT to Detected	Chamfer Dis.
RPD	0.0971	0.0909	0.1880
RPD + PlaneNet	0.1659	0.1013	0.2672
P3I (Ours)	**0.0639**	**0.0881**	**0.1520**

Table 2: Repeated object detection. P3I outperforms both RPD and RPD+PlaneNet. The degradation from "RPD" to "RPD + PlaneNet" is explained by incorrect perspective corrections, which lead to larger errors than not performing the correction at all.

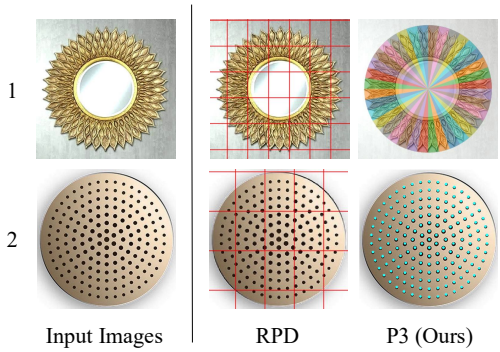

Figure 6: Besides lattice patterns, P3I is also able to detect (1) circular patterns (rainbow colors are used to visualize the inferred program), and (2) hybrid structures composed by a circular structure and a linearly repeated one.

GT" in Table 2) and average distance from all ground-truth centroids to their respective closest detected counterparts ("GT to Detected"). Using both metrics penalizes both cases with excessively many detections and those with very few. We also report the sum of two aforementioned asymmetric error metrics, a.k.a., the Chamfer distance.

Results. Qualitatively visualized in Fig. 5, the original RPD algorithm completely fails when the viewing angle deviates from the fronto-parallel view. The integration of PlaneNet helps correct the perspective effects on certain images, but overall degrades the performance due to large errors when the perspective correction is wrong. As Table 2 shows, our model quantitatively outperforms "RPD + PlaneNet" by a large margin, suggesting that our joint inference algorithm is superior to a pipeline directly integrating camera pose estimation and object detection.

Importantly, P3I suggests a general framework for detecting repeated objects organized in *any* patterns expressible by a program in the DSL. As an example, we show in Fig. 6 that our model successfully localizes objects with a global circular pattern. Specifically, our model discovers the mirror's radial peripheral and holes on the metal surface.

4.4. Image Manipulation

The induced perspective plane programs enable perspective-aware image manipulation. The neural painting network (NPN) [29] is a general framework for program-guided image manipulation; it performs tasks such as inpainting missing pixels, extrapolating images, and editing image regularities. Consider the representative task of image inpainting. The key idea of NPNs is to train an image-specific neural generative model to inpaint missing pixels based on the specification generated by the high-level program: *what* should to be put *where*. The original NPNs work only on fronto-parallel images with objects forming lattice patterns.

Thus, we augment the NPN framework to add support for non-fronto-parallel images and non-lattice patterns. Specifically, based on the inferred camera pose, we first transform the input image to a fronto-parallel view. We then train an NPN to manipulate the transformed image. Circular patterns are supported by introducing an extra rotation operation during image manipulation; the details can be found in our supplementary material.

Baselines. We compare our model against both learning-based (GatedConv [47]) and non-learning-based algorithms (Image Quilting [14] and PatchMatch [6]). Both non-learning algorithms (Image Quilting and PatchMatch) perform image manipulation based on internal statistics only, without referencing to external image datasets. Similarly, P3I-guided NPNs also learn from single images (not external image datasets), and then perform manipulation with learned internal statistics. In contrast, GatedConv is a neural generative model trained on a large collection of images (Places365 [48]) for image inpainting.

Metrics. We evaluate the performance of P3I-guided NPNs in image inpainting with two metrics: average $\mathcal{L}_1$ distance between the inpainted pixels and ground truth, and Inception Score (IS) [34] of the inpainted region.

Results. Qualitative results for inpainting are presented in Fig. 7. Our model can inpaint missing pixels in images at various viewing angles. Both Image Quilting and PatchMatch do not perform well, because they are designed for texture synthesis and assume a stationary texture pattern, and this assumption does not hold when the image is non-fronto-parallel or contains circular patterns. In addition, PatchMatch also modifies pixels near the inpainting region for improved global consistency, resulting in blurriness. More importantly, results by the baselines fail to respect the global perspective pattern (e.g., the lines in 1 and 4). Only P3I-guided NPNs are able to inpaint missing objects of proper sizes that respect the overall perspective structure.

Quantitatively, as shown in Table 3, our P3I-guided NPNs outperform all the baselines (both non-learning-based and learning-based) in terms of $\mathcal{L}_1$ distance between the inpainted pixels and the ground truth. Image Quilting receives a higher Inception Score (IS) than our P3I-NPNs, because high patch diversity, in addition to patch quality, leads to high IS [34], and Image Quilting tends to produce diverse inpainting across the test images.

Fig. 8 shows two intriguing failure cases of our model. In the first case (left), the complex scene consists of multiple planes at different orientations, and P3I struggles to

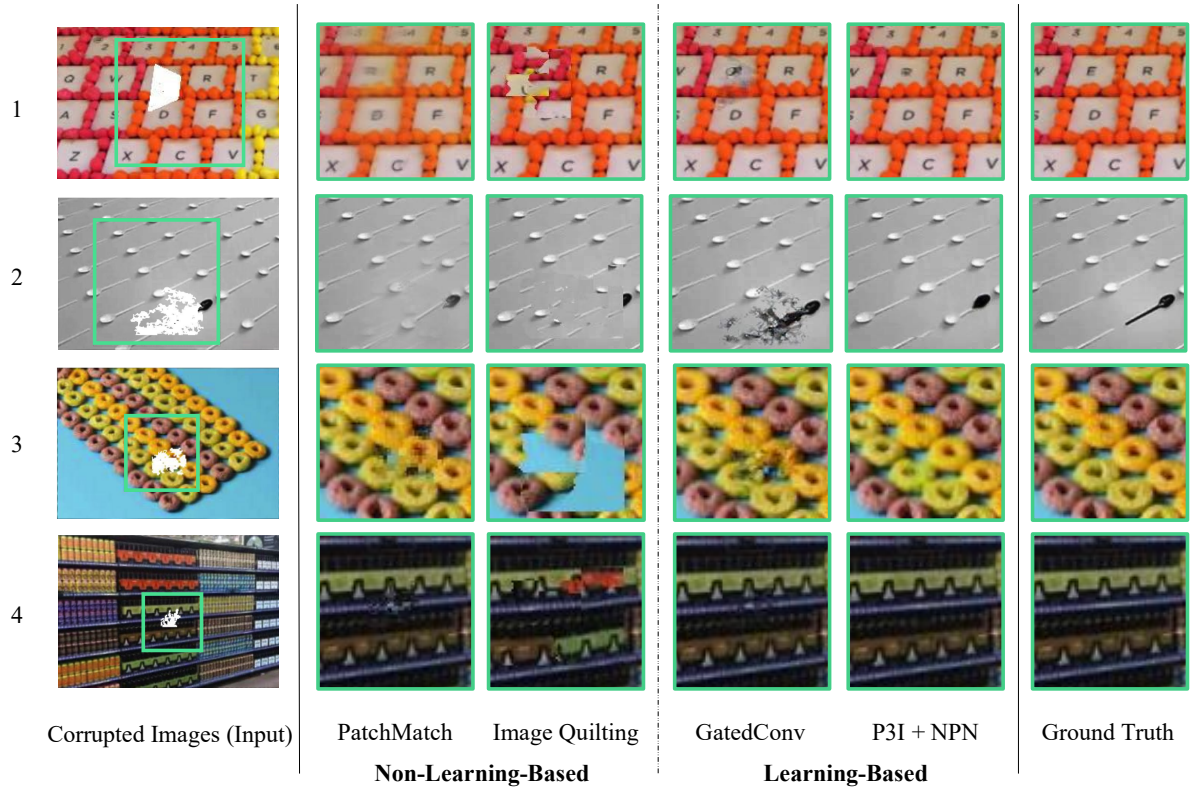

Figure 7: P3I-guided NPNs perform inpainting in a perspective-aware fashion. Results generated by P3I-guided NPNs are sharp (compared with PatchMatch), consistently connect to the global structure (e.g., the lines in 1 and 4), and respect the global perspective effects.

Method	$\mathcal{L}_1$	Inception Score
Image Quilting	35.76	**1.16**
PatchMatch	21.92	1.13
GatedConv	23.20	1.12
P3I +NPN (Ours)	**18.72**	1.14

Table 3: Image inpainting. P3I-Guided NPNs outperform both classic, non-learning baselines and the learning-based baseline in $\mathcal{L}_1$ loss. Image Quilting achieves better Inception Score (IS) than P3I-NPNs, because besides patch quality, patch diversity also improves IS [34], and Image Quilting tends to produce diverse inpainting across the test images.

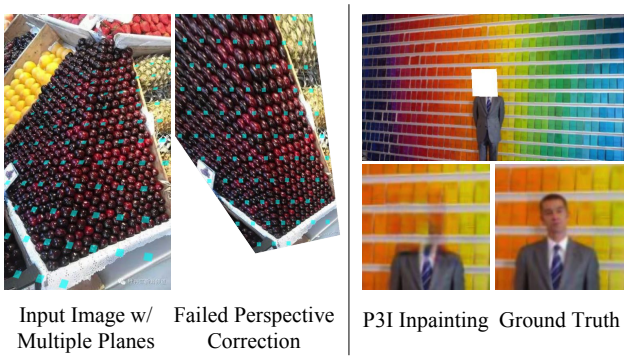

Figure 8: Left: Perspective correction in the presence of multiple planes is a future direction that P3I can take. Right: P3I inpainting learns low-level texture statistics from single images, so it is unable to inpaint the person's head using high-level semantics.

perspective-correct both planes. In the second case (right), P3I only learns low-level texture statistics, therefore failing to inpaint the person's head using high-level semantics.

5. Conclusion

We have presented the perspective plane program induction (P3I), a framework for synthesizing graphics programs as a holistic representation for images. A graphics program models camera poses, object locations, and the global scene structure, such as lattice or circular patterns. The algorithm induces graphics programs through a joint inference of the scene structure and camera poses on a single input image, requiring no training or human annotations. A hybrid approach that combines search-based and gradient-based algorithms is proposed to solve the challenging inference task. The induced neuro-symbolic, program-like representations can further facilitate image manipulation tasks, such as image inpainting. The resulting P3I-guided neural painting networks (NPNs) are able to inpaint missing pixels in a way that respects the global perspective structure.

Acknowledgements. This work is supported by the Center for Brains, Minds and Machines (NSF STC award CCF-1231216), NSF #1447476, ONR MURI N00014-16-1-2007, and IBM Research.

References

[1] Dror Aiger, Daniel Cohen-Or, and Niloy J. Mitra. Repetition maximization based texture rectification. *CGF*, 2(31):439–448, 2012. 2, 5, 6

[2] John Aloimonos. Shape from Texture. *Biological Cybernetics*, 58(5):345–360, 1988. 2

[3] Syed Ammar Abbas and Andrew Zisserman. A Geometric Approach to Obtain a Bird's Eye View From an Image. In *ICCV Workshop*, 2019. 1, 2

[4] Michael Ashikhmin. Synthesizing Natural Textures. In *I3D*, 2001. 3

[5] Coloma Ballester, Marcelo Bertalmio, Vicent Caselles, Guillermo Sapiro, and Joan Verdera. Filling-in by Joint Interpolation of Vector Fields and Gray Levels. *IEEE TIP*, 10(8):1200–1211, 2001. 3

[6] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan Goldman. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *ACM TOG*, 28(3):24, 2009. 3, 7

[7] Tony Beltramelli. Pix2Code: Generating Code from a Graphical User Interface Screenshot. In *EICS*, 2018. 2

[8] Dorothea Blostein and Narendra Ahuja. Shape from Texture: Integrating Texture-Element Extraction and Surface Estimation. *IEEE TPAMI*, 11(12):1233–1251, 1989. 2

[9] Tom Bruls, Horia Porav, Lars Kunze, and Paul Newman. The Right (Angled) Perspective: Improving the Understanding of Road Scenes Using Boosted Inverse Perspective Mapping. In *IEEE Intelligent Vehicles Symposium (IV)*, 2019. 1

[10] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017. 6

[11] Soheil Darabi, Eli Shechtman, Connelly Barnes, Dan B. Goldman, and Pradeep Sen. Image melding: combining inconsistent images using patch-based synthesis. *ACM TOG*, 31:82:1–82:10, 2012. 3

[12] Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M Rush. Image-to-Markup Generation with Coarse-to-Fine Attention. In *ICML*, 2017. 2

[13] Frédéric Devernay and Olivier D Faugeras. Automatic calibration and removal of distortion from scenes of structured environments. In *Investigative and Trial Image Processing*, volume 2567, pages 62–72, 1995. 5

[14] Alexei A. Efros and William T. Freeman. Image Quilting for Texture Synthesis and Transfer. In *SIGGRAPH*, 2001. 3, 7

[15] Kevin Ellis, Daniel Ritchie, Armando Solar-Lezama, and Josh Tenenbaum. Learning to Infer Graphics Programs from Hand-Drawn Images. In *NeurIPS*, 2018. 1, 2, 5

[16] Yaroslav Ganin, Tejas Kulkarni, Igor Babuschkin, S. M. Eslami, and Oriol Vinyals. Synthesizing Programs for Images Using Reinforced Adversarial Learning. In *ICML*, 2018. 2

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 1, 2

[18] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. Image completion using planar structure guidance. *ACM TOG*, 33:129:1–129:10, 2014. 3

[19] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and Locally Consistent Image Completion. *ACM TOG*, 36(4):107, 2017. 3

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NeurIPS*, 2012. 4

[21] Louis Lettry, Michal Perdoch, Kenneth Vanhoey, and Luc Van Gool. Repeated Pattern Detection Using CNN Activations. In *WACV*, 2017. 1, 4, 5, 6

[22] Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas Guibas. GRASS: Generative Recursive Autoencoders for Shape Structures. *ACM TOG*, 36(4):52, 2017. 2

[23] Manyi Li, Akshay Gadi Patil, Kai Xu, Siddhartha Chaudhuri, Owais Khan, Ariel Shamir, Changhe Tu, Baoquan Chen, Daniel Cohen-Or, and Hao Zhang. GRAINS: Generative Recursive Autoencoders for INdoor Scenes. *ACM TOG*, 38(2):12:1–12:16, 2019. 2

[24] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. PlaneNet: Piece-Wise Planar Reconstruction from a Single RGB Image. In *CVPR*, 2018. 2, 5, 6

[25] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image Inpainting for Irregular Holes Using Partial Convolutions. In *ECCV*, 2018. 3

[26] Miaomiao Liu, Xuming He, and Mathieu Salzmann. Geometry-Aware Deep Network for Single-Image Novel View Synthesis. In *CVPR*, 2018. 1

[27] Yunchao Liu, Zheng Wu, Daniel Ritchie, William T. Freeman, Joshua B. Tenenbaum, and Jiajun Wu. Learning to Describe Scenes with Programs. In *ICLR*, 2019. 2

[28] Jitendra Malik and Ruth Rosenholtz. Computing Local Surface Orientation and Shape from Texture for Curved Surfaces. *IJCV*, 23(2):149–168, 1997. 2

[29] Jiayuan Mao, Xiuming Zhang, Yikai Li, William T. Freeman, Joshua B. Tenenbaum, and Jiajun Wu. Program-Guided Image Manipulators. In *ICCV*, 2019. 1, 2, 5, 7

[30] John FJ Mellor, Eunbyung Park, Yaroslav Ganin, Igor Babuschkin, Tejas Kulkarni, Dan Rosenbaum, Andy Ballard, Theophane Weber, Oriol Vinyals, and SM Eslami. Unsupervised Doodling and Painting with Improved SPIRAL. *arXiv:1910.01007*, 2019. 2

[31] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. EdgeConnect: Generative Image Inpainting with Adversarial Edge Learning. *arXiv:1901.00212*, 2019. 3

[32] Chengjie Niu, Jun Li, and Kai Xu. Im2Struct: Recovering 3d Shape Structure from a Single Rgb Image. In *CVPR*, 2018. 2

[33] Tu-ichi Ohta, Kiyoshi Maenobu, and Toshiyuki Sakai. Obtaining Surface Orientation from Texels Under Perspective Projection. In *IJCAI*, 1981. 2

[34] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. In *NeurIPS*, 2016. 7, 8

[35] Gopal Sharma, Rishabh Goyal, Difan Liu, Evangelos Kalogerakis, and Subhransu Maji. CSGNet: Neural Shape Parser for Constructive Solid Geometry. In *CVPR*, 2018. 2

[36] Assaf Shocher, Nadav Cohen, and Michal Irani. "Zero-Shot" Super-Resolution Using Deep Internal Learning. In *CVPR*, 2018. 2

[37] Jean-Philippe Tardif. Non-iterative Approach for Fast and Accurate Vanishing Point Detection. In *ICCV*, 2009. 2

[38] Yonglong Tian, Andrew Luo, Xingyuan Sun, Kevin Ellis, William T. Freeman, Joshua B. Tenenbaum, and Jiajun Wu. Learning to Infer and Execute 3D Shape Programs. In *ICLR*, 2019. 2

[39] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective Search for Object Recognition. *IJCV*, 104(2):154–171, 2013. 2

[40] Yanzhen Wang, Kai Xu, Jun Li, Hao Zhang, Ariel Shamir, Ligang Liu, Zhiquan Cheng, and Yueshan Xiong. Symmetry Hierarchy of Man-Made Objects. *CGF*, 30(2), 2011. 2

[41] Jiajun Wu, Joshua B. Tenenbaum, and Pushmeet Kohli. Neural Scene De-rendering. In *CVPR*, 2017. 2

[42] Wei Xiong, Zhe Lin, Jimei Yang, Xin Lu, Connelly Barnes, and Jiebo Luo. Foreground-Aware Image Inpainting. In *CVPR*, 2019. 3

[43] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-Net: Image Inpainting Via Deep Feature Rearrangement. In *ECCV*, 2018. 3

[44] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-Resolution Image Inpainting Using Multi-Scale Neural Patch Synthesis. In *CVPR*, 2017. 3

[45] Halley Young, Osbert Bastani, and Mayur Naik. Learning Neurosymbolic Generative Models via Program Synthesis. In *ICML*, 2019. 2

[46] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative Image Inpainting with Contextual Attention. In *CVPR*, 2018. 3

[47] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-Form Image Inpainting with Gated Convolution. In *ICCV*, 2019. 3, 7

[48] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 Million Image Database for Scene Recognition. *IEEE TPAMI*, 40(6):1452–1464, 2017. 7

[49] Yichao Zhou, Haozhi Qi, Yuexiang Zhai, Qi Sun, Zhili Chen, Li-Yi Wei, and Yi Ma. Learning to Reconstruct 3D Manhattan Wireframes from a Single Image. In *ICCV*, 2019. 2

[50] Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Non-Stationary Texture Synthesis by Adversarial Expansion. *ACM TOG*, 37(4):49, 2018. 3

[51] C. Lawrence Zitnick and Piotr Dollár. Edge Boxes: Locating Object Proposals from Edges. In *ECCV*, 2014. 2