Reproducibility Challenge : Visual Jenga

Damien Thai *damien.thai@telecom-paris.fr*
MS IA
Telecom Paris

Alexandre Malfoy *alexandre.malfoy@telecom-paris.fr*
MS IA
Telecom Paris

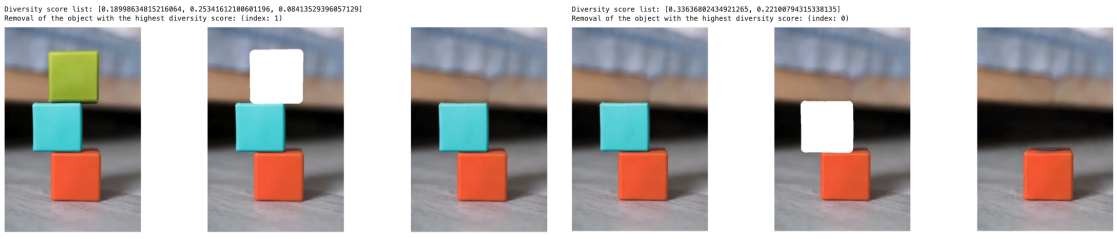
Alexandre Movsessian *alexandre.movsessian@telecom-paris.fr*
MS IA
Telecom Paris

Alexandre Rocchi Henry *alexandre.rocchi–henry@telecom-paris.fr*
MS IA
Telecom Paris

Baptiste Cervoni *baptiste.cervoni@telecom-paris.fr*
MS IA
Telecom Paris

Abstract

Understanding the structural dependencies between objects in a visual scene is a key component of scene understanding, yet remains challenging for current vision models. In this work, we explore the task of Visual Jenga [54], which aims to assess a model's ability to identify a physically coherent order in which objects can be removed from a scene. Building upon the original Visual Jenga framework, we present a fully automated pipeline that leverages large vision-language and generative models—namely MOLMO for object localization, SAM for segmentation, and Stable Diffusion for counterfactual inpainting. By analyzing the diversity of inpainted outputs, our method infers asymmetrical dependencies between objects and determines a plausible removal sequence. We evaluate our pipeline on the NYU-v2 and HardParse datasets, achieving competitive results compared to existing baselines. Our approach demonstrates the feasibility of counterfactual reasoning for structural scene understanding without human intervention, while also identifying key limitations and areas for future improvement.



(a) First object removal (b) Second object removal

Figure 1: **Example of object removal in a scene.** This illustrates how the visual jenga task can allow to remove successively and in a logical order some objects in a scene.

1 Introduction

In the field of computer vision, some models are able to achieve excellent performance in classic tasks. For example some neural networks can be even better than humans in object recognition, but does it mean that these models really understand scenes? In their paper "Visual Jenga: Discovering Object Dependencies via Counterfactual Inpainting" [54], Bhattad et al. introduce a new task to quantify models' understanding of scenes called Visual Jenga. This task consists of removing objects from a scene in a physically plausible order. They also developed a pipeline to deal with this task and two metrics to quantify the plausibility of the object's removal. The goal of this work is to reimplement their pipeline and try to reproduce their results.

2 Related Works

In the early days of computer vision during the 1960s, perception and action were seen as deeply interconnected, and the overarching goal of image understanding was to reason about the physical world from visual input. One of the earliest milestones in the field, Roberts' BlocksWorld [1]—the first PhD thesis in computer vision—explored how a robot could interpret spatial relationships between simple block objects to reconfigure them into new arrangements. Over the decades, however, the ambition of image understanding has largely narrowed to tasks such as object detection, semantic segmentation [2], and more recently, image captioning. In this section, we revisit prior efforts that align with the original, more holistic view of scene understanding.

Psychologist Irving Biederman's foundational work on visual perception [3] emphasized that human understanding of scenes goes well beyond recognizing individual objects or producing textual descriptions. He proposed a set of physical and geometric relational constraints—such as occlusion and physical support—that govern coherent scene interpretation. Building on this, Hoiem and colleagues incorporated such constraints into computational models [4, 5, 6, 7]. Subsequent work expanded this direction with techniques like layered scene decomposition [8, 9], object de-occlusion [10], and hierarchical reasoning across multiple scene layers [11]. Recent interest has revived physics-based scene understanding in the spirit of BlocksWorld, with both synthetic environments [12, 13] and real-world generalizations [14, 15, 16].

Understanding causal relationships from visual data has long been a challenge in both causal inference [17] and computer vision. For example, Lopez-Paz et al. [18] explored object-level causality based on the concept of causal disposition. Goyal et al. [19] employed visual interventions to interpret model behavior by observing how changes to specific image regions affected outcomes. Besserve et al. [20] introduced counterfactual manipulations within generative models to analyze modular internal representations. Zhou et al. [21] studied how humans perceive support relations using synthetic block towers. Building on these foundations, our work extends these ideas to real-world data using large pretrained generative models. While observational data offers only statistical correlations, recent advances in models trained on large-scale image [22, 23] and video datasets [24] show promising counterfactual reasoning capabilities using tools such as text prompts [25], visual prompts [26], and even classification tasks [27]. These models exhibit deep visual understanding of physical scene attributes, including geometry, material properties, lighting, and support [28, 29, 30, 31], and are increasingly used for tasks like object [32] and amodal segmentation [33].

An alternative approach to scene understanding represents objects and their relationships as graphs. Early efforts like Visual Memex [34] modeled objects as nodes linked by edges denoting visual similarity or spatial co-occurrence. This idea was scaled by Visual Genome [35], which constructed large, crowd-sourced scene graphs capturing object categories and relations, prompting further research [36, 37]. However, these approaches primarily capture 2D spatial relations and often overlook geometric or physical dependencies. Other methods aim to decompose scenes into object-centric latent spaces [38, 39, 40, 41, 42], but often model objects independently, missing crucial interaction and support dynamics.

Existing research on object removal and inpainting has mainly focused on visual realism [43, 44, 45]. Common benchmarks evaluate these tasks using standard datasets [46, 47, 48, 49] and assess performance in terms of visual fidelity and plausibility [50, 51]. However, current benchmarks rarely consider the structural and physical dependencies between objects or evaluate whether entire object removal sequences preserve

the scene's coherence. To our knowledge, there is no benchmark that systematically tests whether such operations respect the underlying scene structure.

In summary, our contributions build upon the original Visual Jenga formulation by adapting it for fully automated, human-free interaction. While the original task relied on human intervention to assess scene coherence, we redesign the task to operate end-to-end using generative models alone. Specifically: (1) we extend the Visual Jenga task to a setting where object removals and their evaluations are performed without human input; (2) we refine the counterfactual reasoning framework by leveraging large-scale inpainting models to estimate object dependencies through asymmetry in co-occurrence patterns; and (3) we demonstrate the feasibility and insightfulness of this fully automated setup through both pairwise object evaluations and complete scene decompositions.

3 Theoretical Aspects

The Visual Jenga task is designed to assess a model's ability to reason about the structural organization of objects in a static visual scene. It draws inspiration from the physical game of Jenga, in which players take turns removing wooden blocks from a tower in a sequence that avoids collapsing the structure. In the visual domain, the task similarly involves the removal of elements, but with a focus on semantic and physical plausibility rather than manual dexterity.

More precisely, given a single RGB image depicting a complex arrangement of objects, the goal is to iteratively remove objects one at a time in such a way that each resulting scene remains physically coherent and semantically interpretable. The task terminates once all objects have been removed and only the background remains. Crucially, the removal order is not arbitrary: it must reflect the latent support relationships, occlusion hierarchies, and contextual dependencies among the objects. For example, it would be unreasonable to remove a table while a cup still rests on it, as this would imply that the cup is either floating or has fallen, neither of which is visually represented in the scene.

The Visual Jenga task thus investigates the capacity of a model to infer unobserved causal and physical relationships from a single static image. It departs from conventional vision tasks, such as classification, segmentation, or detection, by emphasizing counterfactual reasoning: determining what the scene would plausibly look like if a given object were not there. This shift from recognition to intervention introduces a layer of complexity that requires the model to understand not only what objects are present but also how they interact structurally within the scene.

At the heart of the Visual Jenga task lies the notion of asymmetric dependency between objects in a scene. The fundamental intuition is that certain objects play a more structurally integral role than others. For example, in a kitchen scene, a countertop is likely to support several other objects, such as a bowl or a kettle. The removal of the countertop, while leaving its dependents in place, would result in a physically implausible configuration. In contrast, the removal of a small item like a spoon from the same scene would typically not disrupt the overall structural integrity. The goal, therefore, is to formalize and quantify such dependencies in a principled way.

Let $\mathcal{O} = \{o_1, o_2, \ldots, o_n\}$ be the set of segmented objects in a scene. We seek to define a removal order $\pi : \mathcal{O} \to \{1, \ldots, n\}$ such that at each time step t, the object $o_{\pi^{-1}(t)}$ can be removed without compromising the physical or semantic plausibility of the resulting image. To construct such an ordering, we consider pairwise dependency relations between objects, which we model using conditional probabilities.

Given two objects A and B, we interpret $P(A \mid B_{\text{rest}})$ as the probability that object A appears in the scene given that object B and the remaining context are present. Similarly, $P(B \mid A_{\text{rest}})$ denotes the probability of observing object B given the presence of object A and the rest of the scene. These conditional probabilities capture the extent to which one object is expected in the presence of another.

Importantly, these dependencies are not symmetric. The presence of a plate might strongly suggest the presence of a table, especially if the plate is located near the bottom of the image and appears to be resting on a flat surface. In that case, $P(\text{table} \mid \text{plate}_{\text{rest}})$ would be high. However, the reverse—$P(\text{plate} \mid \text{table}_{\text{rest}})$—might be significantly lower, since a table can plausibly appear without a plate on it. This

asymmetry reveals an underlying support relationship: the plate depends on the table, and thus the table should be removed only after the plate.

These conditional probabilities provide a probabilistic foundation for defining object dependencies. If $P(A \mid B_{\text{rest}}) \gg P(B \mid A_{\text{rest}})$, we say that A is more dependent on B than the other way around. Consequently, A should be removed before B in any valid removal sequence. This idea can be extended to all object pairs in the scene, forming a directed dependency graph where an edge from A to B indicates that A depends on B and should be removed earlier.

In this formulation, the Visual Jenga task reduces to the problem of inferring these asymmetric conditional relationships for all object pairs and constructing a removal order that respects the resulting dependency structure. While the exact conditional probabilities are unobservable, this probabilistic framework guides the development of approximations and algorithms that aim to infer relative dependencies in a data-driven and unsupervised manner.

In practice, the conditional probabilities discussed previously are not directly observable. To estimate the asymmetric object dependencies without explicit supervision, we adopt a counterfactual reasoning approach: we approximate the plausibility of an object o_i in a scene by masking it out and observing how easily a generative model can fill in the masked region, conditioned on the rest of the image. This process is repeated for each object in the image, enabling a ranking of objects according to their contextual necessity. The key idea is that if an object is *structurally dependent* on others, its removal will result in inpainting outputs that are inconsistent or divergent, indicating its contextual importance.

This approach is implemented through a multi-stage pipeline involving three major components: (i) object coordinates using Multimodal Open Language Model (MOLMO), (ii) object segmentation using the Segment Anything Model (SAM), and (iii) counterfactual image generation using Stable Diffusion. We now describe the mathematical foundations of each component.

Object coordinates with MOLMO. We define **MOLMO** as a multimodal model capable of localizing objects in an image based on a natural language prompt. The model takes an image I and a textual query T as inputs and outputs a set of spatial regions $\{b_1, \ldots, b_n\}$, where each b_i corresponds to a location in the image matching the description.

The image I is encoded by a visual encoder f_V into a spatial feature map $F = f_V(I) \in \mathbb{R}^{H \times W \times d}$. Simultaneously, the prompt T is encoded by a text encoder f_T into a vector $v_T = f_T(T) \in \mathbb{R}^d$.

To determine which regions of the image correspond to the prompt, a similarity score is computed at each spatial location (x, y):

$$S(x, y) = \frac{\langle F_{x,y}, v_T \rangle}{\|F_{x,y}\| \cdot \|v_T\|}$$

This yields a similarity heatmap $S \in \mathbb{R}^{H \times W}$ indicating the degree of semantic alignment between the image and the prompt across locations.

Based on the similarity map S, MOLMO can extract locations in two ways:

- The most likely region is selected via:

$$(x^*, y^*) = \arg\max_{x,y} S(x, y)$$

- Alternatively, all regions above a threshold τ are selected to define a mask:

$$M = \{(x, y) \mid S(x, y) > \tau\}$$

  and the minimal bounding box enclosing M is returned.

The model returns a set of regions $\{b_i\}$ or keypoints $\{(x_i, y_i)\}$ that identify the spatial grounding of the input prompt within the image.

Object Segmentation with SAM. Let I_0 denote the input RGB image. The first step is to extract the set of foreground object masks $\mathcal{M} = \{M_1, M_2, \ldots, M_n\}$ from I_0. We use the Segment Anything Model (SAM) 52 to compute these masks in a zero-shot manner. SAM is a promptable image segmentation model trained on over 1 billion masks. Internally, SAM consists of a Vision Transformer (ViT)-based encoder-decoder structure, which maps the input image I_0 to a set of latent features:

$$z = \text{ViT}(I_0) \in \mathbb{R}^{H \times W \times D}$$

Given a prompt (e.g., bounding box, point, or mask), SAM's decoder produces a binary mask $M_i \in \{0,1\}^{H \times W}$ that localizes object o_i. In the Visual Jenga setup, we use the automatic mask generation mode, which returns a set of non-overlapping object masks $\mathcal{M}$ that covers the foreground elements.

Each object o_i is thus associated with a mask M_i and a corresponding region $R_i = I_0 \odot M_i$, where $\odot$ denotes elementwise multiplication, masking out all pixels outside the object.

Counterfactual Inpainting with Stable Diffusion. To estimate how plausible an object is within the context of the scene, we remove it from the image and inpaint the missing region using a generative model. For this, we employ **Stable Diffusion Inpainting**, an extension of the Stable Diffusion model 53, which can generate coherent image completions conditioned on masked inputs.

Stable Diffusion is a latent diffusion model (LDM) trained to solve the denoising problem in a lower-dimensional latent space. Let x_0 denote the latent representation of the input image, obtained via an encoder $\mathcal{E} : \mathbb{R}^{H \times W \times 3} \to \mathbb{R}^{h \times w \times c}$:

$$x_0 = \mathcal{E}(I_0)$$

The diffusion process defines a Markov chain of latent variables $\{x_t\}_{t=0}^{T}$:

$$x_t = \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbb{I})$$

where α_t is a predefined noise schedule. The model learns a denoising network $\epsilon_\theta(x_t, t, c)$ conditioned on a context image c (in this case, the masked input), trained to predict the noise added at each timestep:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{x_0, \epsilon, t}\left[\|\epsilon - \epsilon_\theta(x_t, t, c)\|^2\right]$$

During inpainting, the object o_i is masked in I_0 to create an input image $\tilde{I}_i$ with a mask M_i. The inpainting pipeline uses $\tilde{I}_i$ and M_i to reconstruct a complete image I'_i that fills the missing region. This process is repeated K times for each object o_i, producing a set of inpainted samples $\{I'_{i,k}\}_{k=1}^{K}$ conditioned on the rest of the scene.

4 Method

4.1 Dataset

To evaluate the performance of the model, we use two different datasets :

The NYU Depth v2 dataset[7] is a widely used benchmark for indoor scene understanding. The images were captured using a Microsoft Kinect. It consists of 1,449 annotated RGB-D images covering a variety of indoor environments such as living rooms, kitchens, bedrooms, and offices. The authors of the original article reduced it to a mere 485 unique images yielding 668 pairwise comparisons with unambiguous removal ordering.

The HardParse dataset was created by the authors of the original article. Because NYU-v2 dataset has very few examples of complex object depencencies (e.g. stacks of objects, hanging/leaning objects), they also

produced a more difficult dataset named HardParse. Using keywords such as “messy desk,” “messy room,” and “stacked objects,” they curated a test set of 40 challenging object pairs from 40 unique internet images, where human experts provided instance-level segmentations and non-trivial removal ordering.

Both of theses datasets were sent to us by the authors of the original article and we are deeply grateful to the authors for providing them.

4.2 Pipeline

The complete Visual Jenga pipeline proceeds in a recursive loop that iteratively removes objects from the image in an order consistent with their inferred dependencies. Given an initial input image I_0, the first step is to identify and localize all objects present. For this, we leverage the MOLMO model to detect salient object proposals and return a list of points $\{p_1, \ldots, p_n\}$ corresponding to the n most likely foreground elements. Each points p_i serves as a prompt to the Segment Anything Model (SAM), which produces a high-resolution binary mask M_i for object o_i. With M_i in hand, we apply Stable Diffusion Inpainting to generate multiple reconstructions of the image with o_i masked out, yielding a set of counterfactual completions $\{I'_{i,k}\}_{k=1}^{K}$. Using these reconstructions, MOLMO evaluates a visual likelihood score S_i for each object, quantifying its contextual necessity. The object with the lowest score—interpreted as the most independent—is selected for removal. The mask M_i is then used to erase the object from the image, and the region is filled using a final inpainting pass. This updated image becomes the input for the next iteration. The process continues until all objects have been removed, producing a sequence of inpaintings that respect the asymmetric dependency structure of the original scene.

4.2.1 MOLMO

Figure 2: **Output of MOLMO.** The model generates a point in each object of the scene that can be used in the rest of the pipeline.

4.2.2 SAM

We use the same model as the article SAM 2 52 to segment the objects. As an input, we give it the image and the coordinates of the object (obtained with Molmo) and the output is a mask of the object that we can then use for the inpainting.

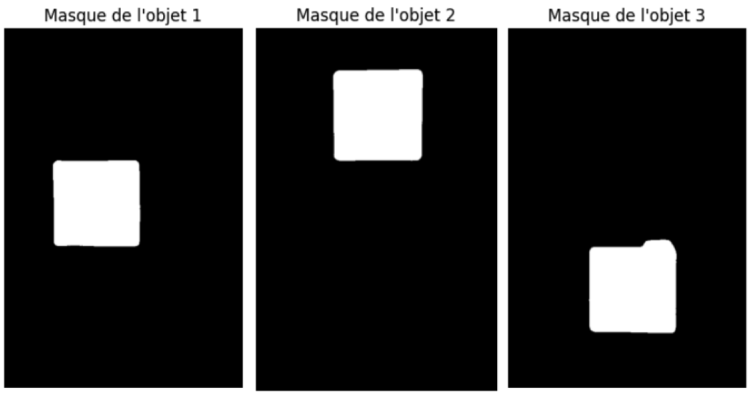


Figure 3: **Output of SAM2.** Given the coordinates obtained from MOLMO, SAM2 is able to segment the object and generate a mask of it.

4.2.3 Stable Diffusion

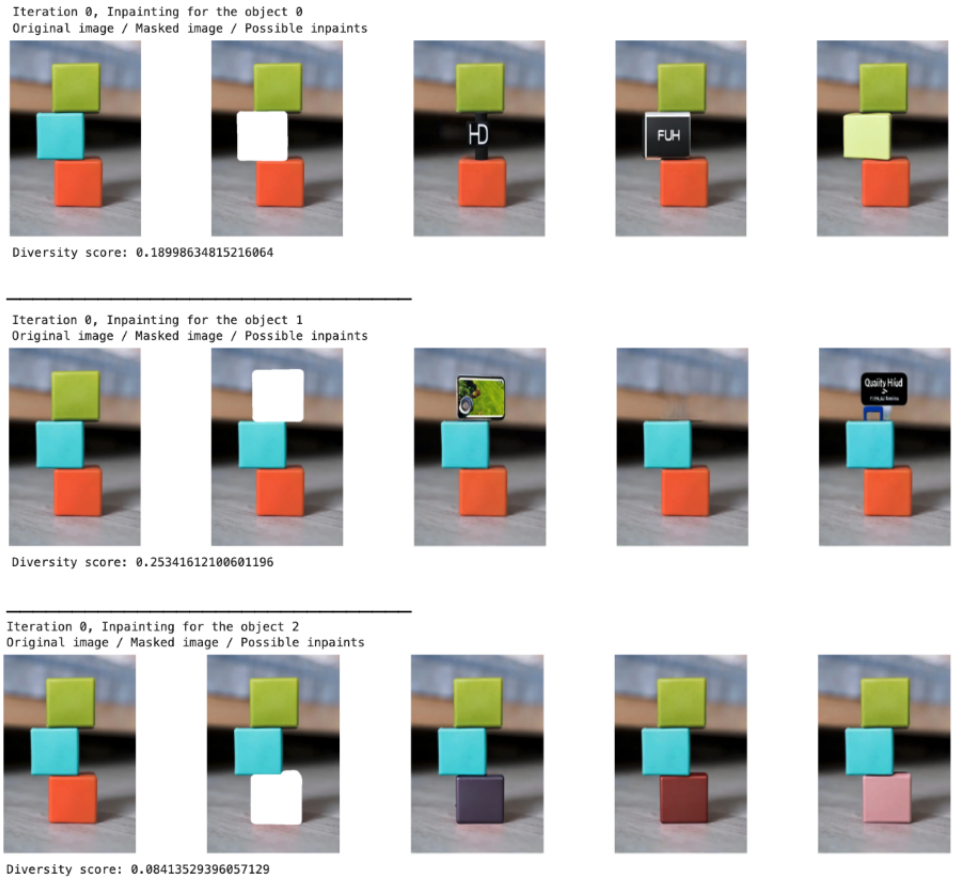


Figure 4: **Inpainting results.** This illustrates what kind of inpanintings Stable Diffusion can perform to replace the original objects. Diversity scores can be computed out of these inpaintings.

5 Results

The evaluation data set includes images with masks of two objects (A and B) of the scene. These two objects have been manually selected with a logical removal order: object A should be removed before object B. The images come from the nyu-v2 dataset and hardparse dataset for harder examples with, respectively, 669 and 40 images. For evaluating the pipeline,the MOLMO and SAM steps are skipped as the masks are already given in the datasets. Thus, only the part which chooses which object to remove is evaluated, which is composed of the generation of the inpainting images and the calculation of the diversity score. The test returns the object to be removed first between A or B. If A is returned, the test is successful as object A should be removed before B in the ground truth.

Figure 5: **Example of the pipeline results.** In this pipeline, the aim is to find the order in which to remove a pair of objects from a scene. We generate 3 inpaintings to replace each object and then calculate the diversity scores.

5.1 Presentation

Method \ Dataset	Nyu-v2	Hardparse
Top2Bottom	56.06%	52.50%
Small2Large	89.39%	50.00%
Ours	83.33%	57.50%

Table 1: Comparison of pipeline and baseline performance

Although we may not have been able to replicate the impressive 91.32% precision in the NYU-v2 dataset as reported in the original article, we still achieved a competitive approach, with our method reaching 83.33% accuracy in NYU-v2 and outperforming others in the Hardparse dataset with 57.50% precision, compared to 52.50% for Top2Bottom and 50.00% for Small2Large.

One reason why our model may be less performant compared to the original could be because we did not use adobe firefly for the inpaintings but rather generated them with stable diffusion.

5.2 Hyperparameters' effect

5.2.1 Number of inpaintings

Due to our limited resources, we were unfortunately not able to test a different number of inpaintings for all the dataset. However, in the few examples that we tested, we were unable to notice a significant difference in performance when using a small or a large number of inpaintings. In the following example (Figure 6), we used the pipeline for the same objects, the first time with only one inpainting per object and the second time with 16. In both cases, the model was wrong about the order in which to remove the objects, but the diversity scores are very close between the 1 inpainting version and the 16 inpainting version. This seems to suggest that the diversity score does not vary much from one inpainting to another.

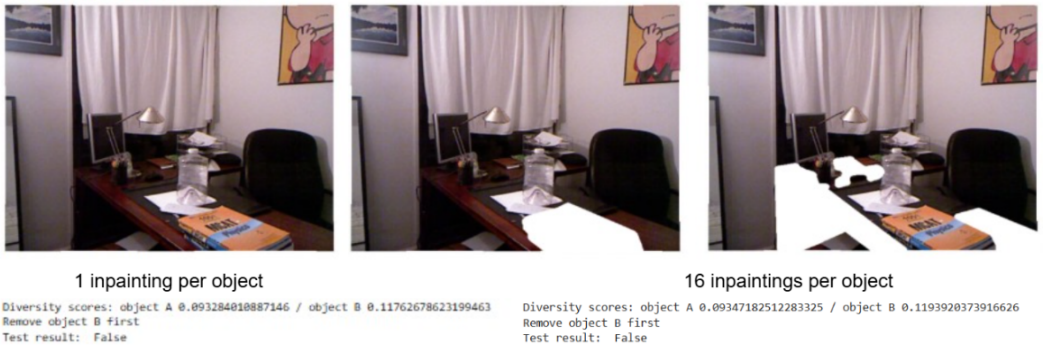


Figure 6: **Impact of the number of inpaintings per object.** For this image and these two objects (the book and the table), the diversity score and prediction of the model are shown for one inpainting per object (on the left) and 16 inpaintings per object (on the right). We were not able to demonstrate that this hyperparameter has a significant impact on performance because the diversity scores are very similar in both cases.

5.2.2 Vision models

Method \ Dataset	Nyu-v2	Hardparse
Clip	59.09%	55.00%
Dino	87.88%	45.00%
Clip + Dino	84.85%	52.50%

Table 2: Effect of diversity scores on performance

As shown in Table 2, DINO achieves the highest performance on the NYU-v2 dataset with 87.88%, while CLIP performs better on Hardparse with 55.00%. Interestingly, combining CLIP and DINO yields competitive but slightly lower results than DINO alone on NYU-v2 (84.85%) and does not surpass CLIP on Hardparse, suggesting that the fusion of both diversity scores may not consistently enhance performance across datasets. As mentionned in the original article, the combination of both these methods makes the score more robust to noise.

5.3 Comparison to original paper

Although we somehow achieve worst results with our model than the original article, we successfully created a fully automated pipeline for the Visual Jenga task. On a simple dataset like Nyu-v2, our model can achieve a 83.33% accuracy on selecting the right object to remove. On a more difficult dataset like Hardparse however, our model can only be slightly better than 50% accuracy.

It is evident that the automation of the pipeline comes at an efficiency cost. One possible lead in order to increase these results would be to use a more potent inpainting software rather than stable diffusion.

6 Discussion

6.1 Critics of the original paper

Human in the loop The pipeline lacks automatisation, as a human interaction is needed for it to be completed : the step of object removal is done with Adobe Firefly.

Evaluation pipeline The evaluation protocol is constrained : the model is only assessed on binary choices, which are to select between 2 pre-defined objects. This simplifies the task but one could argue that the model has learned a generalizable notion of stability. A more complex but more rigorous setup would be to involve prediction with more than 2 objects to select.

Reproducibility The protocol to follow in the original paper is not particularly clear, as not every step in the test pipeline is specified. Important details are omitted or vaguely mentioned. Moreover, the use of a paid software, Adobe Firefly, is a serious barrier for open scientific reimplementation.

6.2 Improvements

Fully automatised The entire pipeline is **fully automated**, requiring no human intervention during execution. Object removal is handled using *stable-diffusion-inpainting*, which provides photorealistic reconstructions of the background after object deletion. The system loops through the pipeline for each object *separately*, ensuring isolated and clean processing per instance. This loop structure maintains consistency in *mask generation*, *inpainting*, and *image updates* throughout the sequence.

Dilatation filter on masks A **dilatation filter** is applied to all masks before the inpainting stage to compensate for possible inaccuracies from the *Segment Anything Model (SAM)*. This operation expands the mask boundaries slightly, ensuring full coverage of the target object and accounting for small segmentation errors. By doing so, it prevents artifacts or residual traces after object removal, improving the visual quality of the output.

Calculate masks at each steps Masks are **recalculated at every step** of the pipeline rather than relying on a static, initial set. This allows the system to adapt to image changes caused by previous inpainting operations. It ensures that each object is detected and masked based on the *current image state*, leading to more accurate segmentations and consistent results throughout the iterative object removal process.

6.3 Limitation of the current implementation

Quality of the mask In this example, the mask only covers the rim of the car wheel causing the object to remain after the object removal step.

Quality of the inpainting images / object removal With our implementation, we encountered the same problems as in the original article with shadows causing the inpainting model to generate a object.

Non determinism Due to the non determinism nature of AI models, several occurrences can lead to different results for the same images. Indeed, object points can vary with Molmo, masks with sam or inpaintings with stable diffusion. Then, these variations can affect the diversity scores of the different objects of the scene changing the order of removal.

Example 2 occurrences of the same object passing through the pipeline lamp and desk.

Figure 7: **Exemple of failed mask generation.**

Long execution time The pipelines' execution time can be very long. Regarding the object removal pipeline, for one image composed of n objects, multiples inpainting images need to be generated for the n objects of the scene and one more to remove the selected object. Then, the process is redone for n-1 objects until the last object is removed. One generation takes 30s in our working environment which lead to the total runtime for one image being equal to (excluding the molmo and sam steps):

$$\text{object removal runtime} = ((n + n - 1 + n - 2 + \ldots + 1) * nb_{inpaint} + n) * 30s$$

For the evaluation pipeline, the execution time is shorter because only two masks are used. However on the dataset of images, the total calculation time is very long.

$$\text{evaluation runtime} = 2 * nb_{inpaint} * nb_{dataset\_images} * 30s$$

References

[1]Lawrence G Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963.

[2]Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollar. Panoptic feature pyramid networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019.

[3]Irving Biederman. On the semantics of a glance at a scene. In *Perceptual Organization*. Routledge, 1981.

[4]Ruiqi Guo and Derek Hoiem. Support surface prediction in indoor scenes. In *2013 IEEE International Conference on Computer Vision*. IEEE, 2013.

[5]Derek Hoiem, Alexei A Efros, and Martial Hebert. Recovering surface layout from an image. *Int. J. Comput. Vis.*, 2007.

[6]Derek Hoiem, Alexei A. Efros, and Martial Hebert. Recovering occlusion boundaries from an image. *International Journal of Computer Vision*, 2011.

[7]Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *Computer Vision – ECCV 2012*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[8]Phillip Isola and Ce Liu. Scene collaging: Analysis and synthesis of natural images with semantic layers. In *ICCV*, 2013.

[9]Chuanxia Zheng, Duy-Son Dao, Guoxian Song, Tat-Jen Cham, and Jianfei Cai. Visiting the invisible: Layer-by-layer completed scene decomposition. *International Journal of Computer Vision*, 2021.

[10]Zhengzhe Liu, Qing Liu, Chirui Chang, Jianming Zhang, Daniil Pakhomov, Haitian Zheng, Zhe Lin, Daniel Cohen-Or, and Chi-Wing Fu. Object-level scene deocclusion. In *ACM SIGGRAPH 2024 Conference Papers*, 2024.

[11]Helisa Dhamo, Nassir Navab, and Federico Tombari. Object-driven multi-layer scene decomposition from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[12]Adam Lerer, Sam Gross, and Rob Fergus. Learning physical intuition of block towers by example. In *Proceedings of The 33rd International Conference on Machine Learning*. PMLR, 2016.

[13]Wenbin Li, Ales Leonardis, and Mario Fritz. Visual stability prediction for robotic manipulation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2606–2613. IEEE, 2017.

[14]Abhinav Gupta, Alexei A. Efros, and Martial Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *European Conference on Computer Vision(ECCV)*, 2010.

[15]Tom Monnier, Jake Austin, Angjoo Kanazawa, Alexei A Efros, and Mathieu Aubry. Differentiable blocks world: Qualitative 3D decomposition by rendering primitives. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[16]Vaibhav Vavilala, Seemandhar Jain, Rahul Vasanth, Anand Bhattad, and David Forsyth. Blocks2world: Controlling realistic scenes with editable primitives. *arXiv preprint arXiv:2307.03847*, 2023.

[17]Joris M Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. Distinguishing cause from effect using observational data: methods and benchmarks. *The Journal of Machine Learning Research*, 2016.

[18]David Lopez-Paz, Robert Nishihara, Soumith Chintala, Bernhard Schölkopf, and Léon Bottou. Discovering causal signals in images. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[19]Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International Conference on Machine Learning*. PMLR, 2019.

[20]Michel Besserve, Ar ash Mehrjou, Rémy Sun, and Bernhard Schölkopf. Counterfactuals uncover the modular structure of deep generative models. *International Conference on Learning Representations (ICLR)*, 2020.

[21]Liang Zhou, Kevin A Smith, Joshua B Tenenbaum, and Tobias Gerstenberg. Mental jenga: A counterfactual simulation model of causal judgments about physical support. *Journal of Experimental Psychology: General*, 2023.

[22]Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image transformers. In *International Conference on Learning Representations (ICLR)*, 2022.

[23]Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2022.

[24]Daniel M Bear, Kevin Feigelis, Honglin Chen, Wanhee Lee, Rahul Venkatesh, Klemen Kotar, Alex Durango, and Daniel L K Yamins. Unifying (machine) vision via counterfactual world modeling. *arXiv preprint arXiv*, 2023.

[25]T Brooks, A Holynski, and A A Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[26]Amir Bar, Yossi Gandelsman, Trevor Darrell, Amir Globerson, and Alexei A Efros. Visual prompting via image inpainting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[27]Alexander C Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[28]Anand Bhattad, Daniel McKee, Derek Hoiem, and David Forsyth. Stylegan knows normal, depth, albedo, and more. *Advances in Neural Information Processing Systems*, 2023.

[29]Anand Bhattad, James Soole, and David A Forsyth. Stylitgan: Image-based relighting via latent control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[30]Xiaodan Du, Nicholas Kolkin, Greg Shakhnarovich, and Anand Bhattad. Generative models: What do they know? do they know things? let's find out! *arXiv preprint arXiv:2311.17137*, 2023.

[31]Guanqi Zhan, Chuanxia Zheng, Weidi Xie, and Andrew Zisserman. A general protocol to probe large vision models for 3D physical understanding. In *Advances in Neural Information Processing Systems 37*, 2025.

[32]Deniz Oktay, Carl Vondrick, and Antonio Torralba. Counterfactual image networks. 2018.

[33]Ege Ozguroglu, Ruoshi Liu, Dídac Surís, Dian Chen, Achal Dave, Pavel Tokmakov, and Carl Vondrick. Pix2gestalt: Amodal segmentation by synthesizing wholes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[34]Tomasz Malisiewicz and Alyosha Efros. Beyond categories: The visual memex model for reasoning about object relationships. *Adv. Neural Inf. Process. Syst.*, 2009.

[35]Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael S Bernstein, and Fei-Fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations. In *International Journal of Computer Vision*. Springer, 2016.

[36]Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

[37]Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[38]Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

[39]Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A Efros. Blobgan: Spatially disentangled scene representations. In *European conference on computer vision*. Springer, 2022.

[40]Dave Epstein, Allan Jabri, Ben Poole, Alexei Efros, and Aleksander Holynski. Diffusion self-guidance for controllable image generation. *Advances in Neural Information Processing Systems*, 2023.

[41]Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International conference on machine learning*. PMLR, 2019.

[42]Dim P Papadopoulos, Youssef Tamaazousti, Ferda Ofli, Ingmar Weber, and Antonio Torralba. How to make a pizza: Learning a compositional layer-based gan model. In *proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[43]Alper Canberk, Maksym Bondarenko, Ege Ozguroglu, Ruoshi Liu, and Carl Vondrick. EraseDraw: Learning to insert objects by erasing them from images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[44]Daniel Winter, Matan Cohen, Shlomi Fruchter, Yael Pritch, Alex Rav-Acha, and Yedid Hoshen. ObjectDrop: Bootstrapping counterfactuals for photorealistic object removal and insertion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[45]Junhao Zhuang, Yanhong Zeng, Wenran Liu, Chun Yuan, and Kai Chen. A task is worth one word: Learning with task prompts for high-quality versatile image inpainting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

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

[47]Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014.

[48]Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, (2018), 2018.

[49]Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[50]Changsuk Oh, Dongseok Shim, Taekbeom Lee, and H Jin Kim. Object remover performance evaluation methods using class-wise object removal images. *IEEE Sensors Letters*, 2024.

[51]Min-Cheol Sagong, Yoon-Jae Yeo, Seung-Won Jung, and Sung-Jea Ko. Rord: A real-world object removal dataset. In *BMVC*, 2022.

[52]Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

[53]Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[54]Anand Bhattad, Konpat Preechakul, and Alexei A. Efros. Visual Jenga: Discovering Object Dependencies via Counterfactual Inpainting. In *arXiv preprint arXiv:2503.21770*, 2024.