NATURAL-LANGUAGE GUIDED OBJECT ADDITION THROUGH CONTEXTUAL REASONING

Amit Vikram Singh, Janmenjaya Panda, Niteesh Singh, Vineet Batra
{amitvikrams, janmenjayap, niteeshs, vbatra}@adobe.com

ABSTRACT

Recent advances in vision–language models enable image editing from text prompts, yet accurately localizing and integrating new objects into complex scenes remains challenging, often causing spatial misplacement or loss of visual fidelity.

We present a fully automated, context-aware pipeline for identity-preserving object addition from natural language prompts. A novel data generation framework combines GPT-4-Vision [9], Imagen-3.0 [2], FLUX.1-schnell [5], UniVG-R1 [3], and Object Peeling [1] to synthesize diverse training instances with metadata. Florence-2-large [8] is fine-tuned to predict a spatial insertion point, resolving ambiguities amongst analogous context subjects. The final composite is produced by conditioning FLUX.1-dev [4] on the source image, edit instruction, and predicted point of localization.

Our method improves spatial accuracy, prompt compliance, and visual fidelity over existing approaches, without requiring object reference masks, example images, or complex annotations, advancing scalable, high-quality natural-language-guided object addition.

Keywords: Vision–Language Models, Natural-Language Guided Image Editing, Object Addition, Contextual Disambiguation, Spatial Reasoning

1 PROBLEM STATEMENT

Image editing systems guided by natural language often struggle with spatial accuracy and visual consistency, particularly in complex, multi-entity scenes. When an edit instruction targets a specific subject, models frequently misplace the object or apply it to the wrong entity. For example, in an image with two people—one in an orange sweater and the other in a white shirt—the instruction *"add a hat to the person wearing an orange sweater"* may lead FLUX-based models such as FLUX.1 Kontext [6] to incorrectly modify the person in the white shirt, reflecting insufficient grounding between language and visual context.

Another challenge is over-editing, where systems such as GPT-Image-1 [11] alter unrelated regions, distorting important details like facial features or body structure. Furthermore, synthetic training data often introduces blending artifacts that bias models toward low-level pixel cues rather than true spatial reasoning. Existing solutions either require manual annotations (e.g., bounding boxes) or lack the spatial inference necessary for correct placement in complex scenes.

Moreover, training models for such tasks requires generating synthetic data. This data often contains visual artifacts or unnatural blending at the edit location. When used as-is, these artifacts introduce bias into the training process, causing the model to rely on low-level pixel irregularities rather than learning proper object placement and composition based on edit instruction.

Existing solutions either require manual annotations (e.g., bounding boxes) or lack the spatial reasoning needed to infer where the object should be added in a complex scene.

(a) Original Image (b) Edited Result

Instruction: "Add a red hat to the boy wearing green t-shirt"

Figure 1: Example illustrating the core challenge: given an instruction to insert an object, the system must correctly determine ***where*** the object should be added and ensure ***visual consistency*** with the scene. This requires disambiguating between similar subjects and preserving unrelated content.

The core problem can be formally stated as:

Problem Statement

Input:

- A source image, possibly with multiple ambiguous subjects.
- A natural-language instruction to add an object, potentially specifying contextual attributes.

Objective: Produce an edited image in which the specified object is inserted at the correct location for the intended subject, while preserving unrelated content and subject identity.

Constraints:

- No masks, bounding boxes, reference images, or annotations at test time.
- Automatic disambiguation among similar subjects.
- Minimal alteration outside the insertion region.
- Placement must follow spatial plausibility and instruction semantics.

This work addresses these challenges through:

- A synthetic data pipeline combining GPT-4-Vision, Imagen-3.0, FLUX.1-schnell, UniVG-R1, and Object Peeling to create context-specific training examples.
- Fine-tuning Florence-2-large [8] to predict a single spatial reference point for object insertion.
- Conditioning FLUX.1-dev [4] on the source image, edit instruction, and predicted reference point to generate image with object.
- And a transformation strategy applied during training to mitigate model bias toward visual artifacts introduced by synthetic data.

This enables accurate, context-aware, and visually faithful object insertion, improving spatial precision and content preservation over existing approaches.

2 PRIOR ART(S)

The task of object addition in images, particularly when guided by natural language prompts, has been explored extensively in the domains of general-purpose image editing and vision-language alignment.

Despite notable progress, current approaches face significant limitations in achieving precise spatial localization, identity preservation, and visual fidelity—especially when inserting objects onto spectific subject context in complex multi-entity scenes.

- **GPT-based vision models:**

    1. Recent advancements in multimodal large language models (LLMs)—such as GPT-Image-1 and related transformer-based architectures—have enabled end-to-end image editing guided purely by textual instructions.
    2. These models can generate or modify visual content without explicit region annotations, offering flexibility in diverse editing scenarios.
    3. However, they often suffer from spatial ambiguity in scenes containing multiple similar entities. For instance, when prompted to "*add a hat to the cat*" in an image containing both a cat and a dog, the system may incorrectly apply the object to the dog.
    4. Additionally, their generative nature can introduce unintended modifications to unrelated image regions, leading to structural distortions or feature degradation in semantically critical areas such as faces.
    5. These artifacts undermine both identity preservation and the overall realism of the output.

- **Flux Kontext:**

    1. The Flux Kontext framework adopts a more spatially aware editing paradigm, integrating contextual reasoning into the generation process.
    2. Compared to generic GPT-based vision models, Flux Kontext demonstrates improved subject identity preservation and enhanced visual consistency.
    3. Its context conditioning mechanism better aligns object additions with the surrounding visual scene, thereby reducing the distortions seen in earlier systems.
    4. Nevertheless, the model remains prone to entity disambiguation errors in multi-subject scenes and often fails to preserve fine-grained details—particularly where textural continuity with the original image is critical.
    5. These shortcomings reduce its suitability for high-precision editing applications requiring both semantic accuracy and detail preservation.

- **Clio Gen Fill:**

    1. In contrast to fully automated methods, Clio Gen Fill adopts a user-guided approach, requiring manual provision of a mask that defines the target edit region.
    2. This explicit localization input enables precise modifications, as the model restricts edits to a well-defined spatial area.
    3. While this approach effectively mitigates spatial ambiguity and misplacement errors, it places a significant burden on the user.
    4. The reliance on manual mask creation limits scalability and reduces the practicality of the approach for large-scale applications or workflows requiring minimal human intervention.

Table 1: Comparison of different model outputs for the same edit instruction.

In summary, existing approaches to (multi-subject) contexual object addition demonstrate one or more of the following limitations:

1. **Inaccurate spatial grounding** of object placement, particularly in scenes with multiple similar subjects.
2. **Degradation of identity-preserving features**, especially in semantically important regions such as faces or distinctive attributes.

3. **Dependence on manual localization inputs**, restricting scalability and automation.

To date, there is **no existing solution** that delivers a **fully automated**, **accurate**, and **identity-preserving** method for (multi-subject) contextual object addition using only **natural language prompts**, while maintaining both **spatial precision** and **high visual fidelity**. This gap underscores the need for methods that combine **robust vision-language understanding** with **spatially grounded generation**.

3 SOLUTION

To enable realistic object insertion into images guided by natural language instructions, we propose a data-driven framework that combines custom data generation and model training for spatially and semantically grounded editing. Our approach addresses key challenges in controllable image editing through the following core components:

- **Data Generation Pipeline:**
  - Automatically constructs high-quality training triplets of the form: `<image without the object, edit instruction, image with the object>`.
  - Supports both single-subject and multi-subject insertion scenarios.
  - Leverages generative models and structured scene prompts to synthesize diverse, realistic examples.
- **Spatial Grounding Module:**
  - Utilizes **UniVG-R1** to predict a **bounding box of the contextual region** where the object should be inserted.
  - This bounding box does not represent the object itself, but rather the surrounding context. For instance, given the instruction "add a hat on the person," the bounding box would cover the person, not the hat.
  - The predicted bounding box provides high-level spatial cues for object placement.
  - **Florence-2-Large** then refines this spatial grounding by predicting a **precise insertion point** within the context box, based on the input image and instruction.
  - This two-step process enables context-aware and spatially accurate object insertions.
- **Image Generation Module:**
  - Fine-tunes **FLUX.1-dev** to generate the final edited image.
  - The generation is conditioned on:
    - The object-free image,
    - The natural language instruction, and
    - The insertion point predicted by Florence.
- **Overfitting Mitigation:**
  - Applies a training-time transformation strategy to reduce overfitting to **pixel-level artifacts introduced during inpainting** in the data generation pipeline.
  - This helps the model focus on meaningful visual and semantic patterns, rather than low-level synthetic cues.

Together, these modules form a unified system capable of instruction-based, context-sensitive object insertion with high spatial precision and semantic alignment.

3.1 THE DATASET: DEFINITION

A single data point in our dataset comprises three primary aligned components (and the generated associated metadata):

1. **image without the object:** aka the source image, an image in which the target object is absent — for example, *"two boys standing in the image, one in a red and the other in a green t-shirt, with no one wearing a hat"*;

2. **the edit instruction**: a natural language suggestion specifying the intended edit — such as *"add a red hat to tallest person"*;
3. **image with the object:** aka the target image, the corresponding image in which the object has been inserted in accordance with the instruction, resulting in a visually coherent scene that reflects the described modification. This triplet structure ensures that each training example provides both spatial and semantic grounding for learning instruction-guided object insertion.
4. **the derived associated metadata:**
   (a) **object mask:** the binary mask of the object to be inserted
   (b) **object name:** the name of the object to be added — *"the red hat"*
   (c) **object description:** a short description of the object capturing the style and composition — *"a bold, stylish red hat that adds a vibrant touch to any outfit"*
   (d) **object location (description):** the natural language description of the location where the object shall be inserted in the image — *"on the head of the tallest person"*
   (e) **object location (co ordinate):** the spatial point (x_1, y_1) denoting where the object shall be inserted
   (f) **object context (description):** the natural language description of context with respect to the location of object insertion — *"the boy wearing the red t-shirt, standing on the left of the image, aka the tallest person"*
   (g) **object context (bounding box):** the bounding box in the format of ordered list $< x_1, x_2, y_1, y_2 >$ capturing the context, where (x_1, y_1) and (x_2, y_2) are the absolute coordinate of the top-left and bottom-right points of the box

3.2 THE DATASET: GENERATION

A comprehensive data generation pipeline has been developed to produce diverse and high-quality training samples, while effectively minimizing artifacts that may arise during the editing process. The methodology encompasses the following stages:

1. compiling a list of object names,
2. using GPT-4o [10] to generate detailed caption, object description, object location of manifold scenes where one or more subjects (another object/ animal/ character/ person) contain this object in the context,
3. generating images (aka the image with object of the dataset) from these captions using FLUX.1-schnell and Imagen-3.0,
4. filtering out (by observation) the images where the generated image does not follow the quadruple $<$ image with object, caption, object name, object description, object location $>$,
5. detecting the bounding box of the object of interest in the generated images using Open vocabulary detection task of Florence-2-large in the generated image, guided by the object description and object location,
6. obtaining the mask of the object of interest from the bounded region using Segment Anything Model [7],
7. mask Dilation and inpainting that region using object peeling-off to obtain the image without object,
8. filtering out (by observation) the samples where the obtained mask is inaccurate,
9. provided this image without object (aka synthetic source image), image with object, object name, object description and object location, producing a set of 10 edit instructions of increasing detailing, each of which shall lead to the addition of object in the synthetic source image,
10. applying an artifact transformation by randomly inpainting regions of the image without object at random locations using object peeling-off (This reduces the model's bias toward synthetic artifacts caused by data generation and improves generalization),

11. filtering out (by observation) the samples where the sextuplet <image without object, edit instruction, image with object, object name, object description, object location> is inconsistent.
12. given the image with object with the object marked with the bounding box, the object name, description and location, using GPT-4o to get object context and using UniVG-R1 to obtain the bounding box of the context.
13. filtering out the samples where the marked context is flawed.

    We use a minimum percentage overlap criterion for the bounding box of the object and that of the context for filtering. Afterwards, the remaining filtering was performed by observation.
14. computing the spatial point of object insertion by the arithmetic average (geometric centroid) of the binary mask of the object

3.3 THE DATASET: AUGMENTATION

To enrich the dataset and improve generalization, we adopt a cross-image augmentation strategy. Each image exists in two forms: *with object* and *without object*. Augmentation is performed by conditioning on edit instructions from a reference image (Image 1) while pairing it with different variants of another image (Image 2). Formally:

$$[\text{Image1\_wo}] + [\text{Image2\_variant}] \xrightarrow{\text{edit instruction (Image1)}} [\text{Image1\_w}] + [\text{Image2\_variant}],$$

where Image2_variant may be its *with object*, *without object*, or another variant (e.g., different pose or context). This yields multiple augmentation pairs for a single instruction.

A key motivation is to prevent the model from exploiting peeling artifacts in the *image without object*. When both Image 1 and Image 2 contain distortions in the same region (e.g., head for a hat), artifact cues become unreliable, forcing reliance on textual instructions instead.

Two design criteria are followed: (1) Image 1 and Image 2 should belong to the same object category (e.g., wearables) for semantic validity, and (2) entities should be comparable so that instructions transfer naturally. Overall, this augmentation increases sample diversity, discourages shortcut learning, and strengthens grounding to natural-language instructions.

3.4 THE TRAINING PIPELINE: FINETUNING FLORENCE-2-LARGE FOR PREDICTING SPATIAL POINT OF OBJECT INSERTION

The objective of this finetuning is to obtain a spatial point of insertion of object, provided image without object and the edit instruction.

Finetuning Florence-2-large to predict a bounding box for the object addition is challenging because the object is not originally present in the image, making bounding box supervision ambiguous. Instead, Florence is fine-tuned to predict a single spatial point that indicates where the object should be inserted. The approach goes as follows:

1. We introduce a new task <OBJECT_LOCATION> for finetuning of Florence-2-large.
2. The input to the model is image without object and a semicolon separated string of <edit instruction, object name, description location and context, the bounding box of the object context>. The bounding box was provided with normalized location tags for each of the four co ordinate. We apply several transformations (jpeg compression, grayscale transformation, random equalization, rotation by smaller amount) to induce variability and limit overfitting, effectively regularizing the model and improving its ability to generalize.
3. The output that is to be predicted is a spatial point of object insertion. We consider the normalized location tags of the point. Analogous to the previous step, we introduce minor noise to the point as a transformation to simulate task specific adaptation and further improve robustness during training.
4. We carried out several finetuning methods using LoRA, introducing several freezing modules; but, a full finetuning generated superior results amongst all.

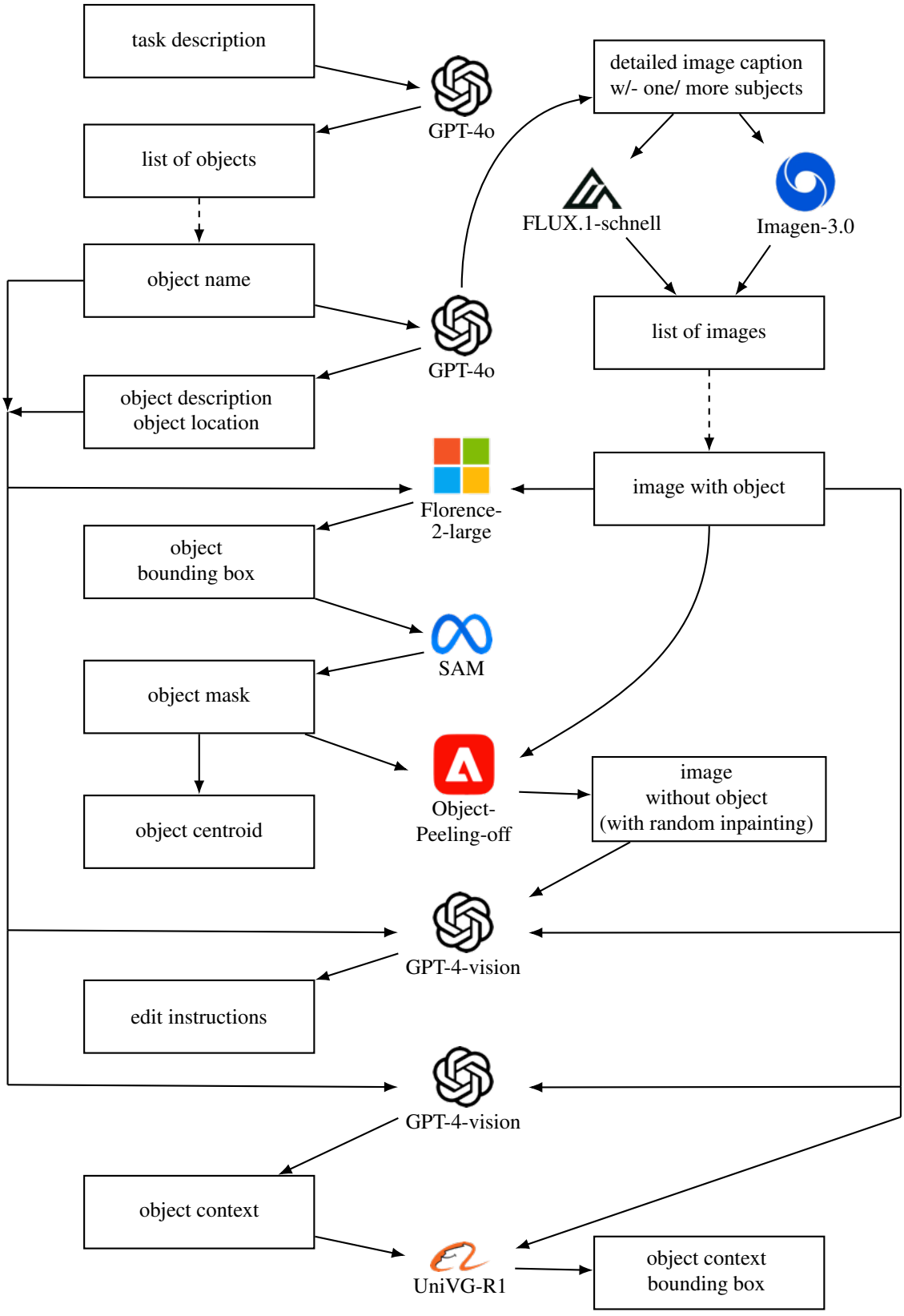


Figure 2: data generation pipeline

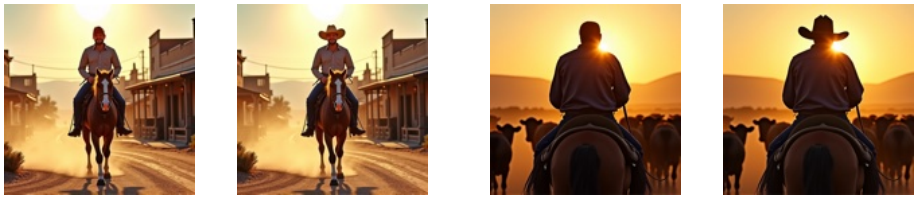
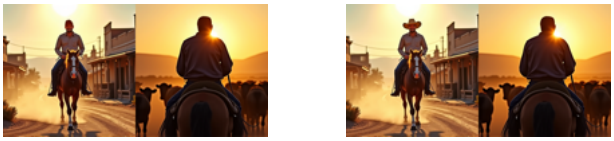


Image 1 without **hat** | Image 1 with **hat** | Image 2 without **hat** | Image 2 with **hat**



Augmentation 1 Input ([image1_wo_hat] + [image2_wo_hat]) | Augmentation 1 Output ([image1_w_hat] + [image2_wo_hat])

Edit instruction from Image 1 is applied here

Augmentation 2 Input ([image1_wo_hat] + [image2_wo_hat variant 2]) | Augmentation 2 Output ([image1_w_hat] + [image2_wo_hat variant 2])

Edit instruction from Image 1 is applied here as well

Figure 3: Example of cross-image augmentation using the object **hat**. Top row: base images with and without the hat. Bottom rows: augmentation examples where the edit instruction for Image 1 (e.g., *"Place a hat on the person's head"*) is applied to paired images. The model learns to rely on text instructions rather than artifacts from object removal (inpainting) by mixing Image 1 with different Image 2 variants.

(a) Metadata (GPT-4o) | (b) Image **with Object** (FLUX/Imagen) | (c) Mask (Florence-2 + SAM) | (d) Image **without Object** (Object-Peeling)

Figure 4: An example run of the data generation pipeline. The process starts with GPT-4o metadata (a), generates the image (b), detects and masks the object (c), and produces the image without the object (d).

The key advantages include:

1. With the context guidance obtained using UniVG-R1, Florence excels at distinguishing between multiple analogous spatial point of object insertion (of the object that is not in the image) by effectively linking textual prompts to the correct target, such as differentiating the neck of a cat from that of a dog or identifying the head of a specific person.

2. The image generation model, FLUX, while strong in producing realistic edits, lacks the same level of precision in subject disambiguation.

3. By providing FLUX with Florence's spatial point, the system ensures more accurate localization of the object addition.

4. This approach leverages the strengths in vision-language understanding of UniVG-R1 and Florence to deliver reliable spatial guidance, improving the precision and contextual relevance of edits.

5. Fine-tuning Florence on synthetically generated data enables it to learn plausible and effective object addition locations.

A finetuned Florence-2-large is now able to accurately identify the spatial point of object insertion provided the image without object and the edit instruction with the associated relevant metadata.

3.5 THE TRAINING PIPELINE: FINETUNING FLUX.1-DEV FOR INSERTING THE OBJECT AT THE LOCATION OF INTEREST

OminiControl[12] is a framework built on top of Flux[4], introducing a unified sequence processing strategy that combines condition tokens with image tokens, enabling flexible and joint interactions across modalities during generation. We extend the original *OminiControl*[12] framework to support multiple image-based conditioning inputs, enabling more precise and context-aware object addition. While the original OminiControl architecture accepts a single image condition alongside a text prompt, our approach modifies the framework to take in **two image conditions** in addition to the textual description.

Dual Image Conditioning. Specifically, we condition the model on:

1. **Original Image** (I_{orig}): This image provides the full visual context and allows the model to preserve important background details, lighting conditions, and object continuity in the unmodified regions.

2. **Blacked-out Image** (I_{mask}): A copy of the original image in which a fixed-size (20×20 pixels) black region is placed at the desired object insertion location. This serves as a soft spatial prior, loosely indicating *where* the object should be added.

3. **Text Prompt** (T): A natural language instruction specifying the object to be added (e.g., "Add a cat on the sofa"), providing semantic guidance about *what* to add.

This dual-image input strategy allows the model to simultaneously reason about:

- **What to preserve**: from I_{orig}, ensuring high-fidelity background reconstruction.
- **Where to add**: from I_{mask}, which highlights the intended modification region.

Architecture Modification. To accommodate both image conditions within the OminiControl architecture, we modify the input encoding pipeline. Both I_{orig} and I_{mask} are passed independently through a shared Variational Autoencoder (VAE) encoder. Their resulting latent embeddings are then concatenated along the channel dimension to form a unified representation:

$$z = \text{concat}(\text{VAE}(I_{\text{orig}}), \text{VAE}(I_{\text{mask}})) \tag{1}$$

This concatenated image condition is then combined with the tokenized text prompt T following the OminiControl strategy, forming a unified token sequence that is passed to the Diffusion Transformer (DiT) backbone for generation.

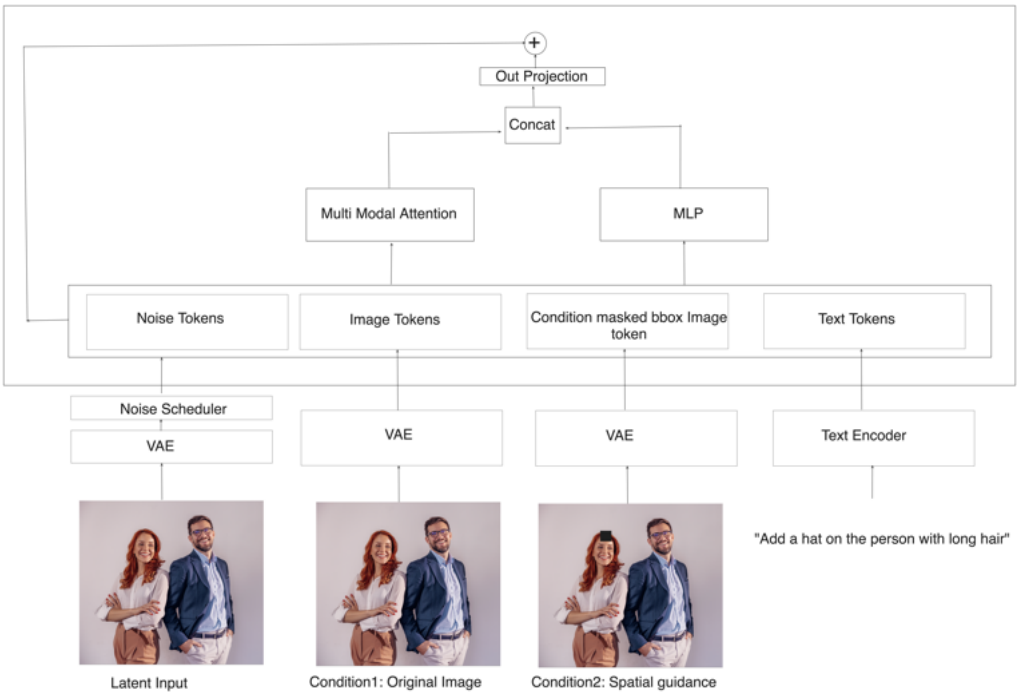


Figure 5: Overview of our modified *OminiControl* architecture for object addition. The model takes three inputs: the **Latent Input** X, which is the noised image used in the diffusion process; the **Context Image** I_{orig}, representing the original unmodified scene to guide content preservation; and the **Mask Image** I_{mask}, a version of the original image with a blacked-out region indicating the approximate location for object addition. Both I_{orig} and I_{mask} are encoded via a shared VAE encoder and concatenated to form a combined visual condition. This is fused with the tokenized text prompt to produce a unified token sequence, which is then processed by the Diffusion Transformer to generate the final output

Relation to Inpainting. The use of a blacked-out region is inspired by prior work in inpainting, particularly in Flux-based generative models. However, unlike traditional inpainting approaches that rely solely on masked inputs, our method combines both the masked image and the unmasked original. This dual input strategy allows the model to learn both what content must remain unchanged and where modifications are permissible.

3.6 THE INFERENCE PIPELINE

The inference pipeline can be summarized as follows:

1. User provided the image without object (aka source image), and a natural language edit instruction.
2. From these input, we derive the metadata: object name, description, location, context using gpt-4-vision; followed by the bounding box of the context from its description using UniVG-R1.
3. We obtain the spatial point of object insertion using the finetuned Florence-2-large from the source image and the edit instruction alonwith object name, description, location, context and its bounding box.
4. Finally, we generate the image with object from the source image and the edit instruction guided by the spatial point of insertion.

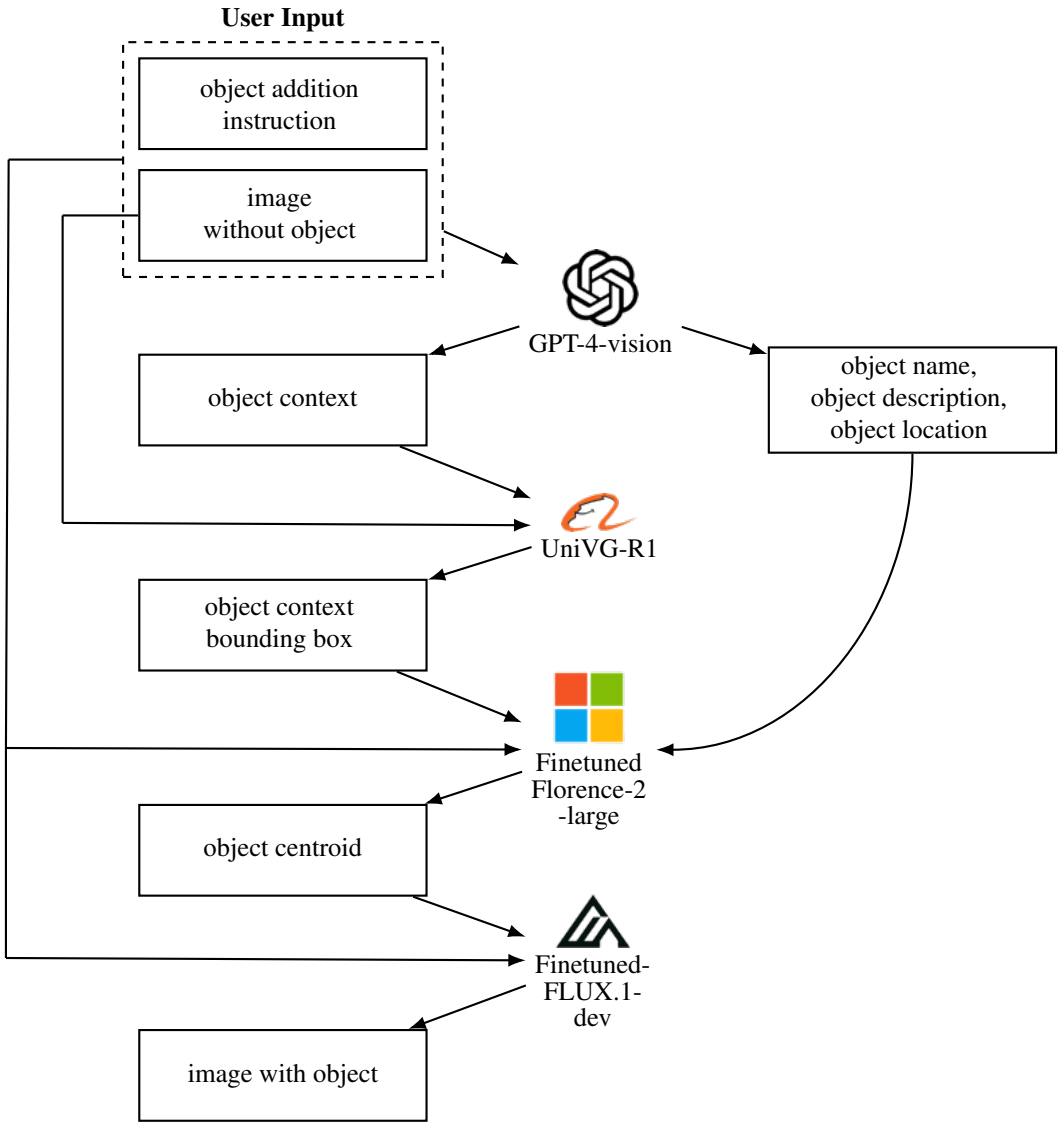


Figure 6: Inference pipeline

4 NOVELTY, INNOVATION, AND ADVANTAGES

The proposed work introduces several key innovations that collectively advance the state of the art in natural-language guided object addition, particularly in complex multi-entity scenes.

4.1 NOVELTY AND INNOVATION

1. **Novel Data Generation Framework:** We present a fully automated data generation pipeline that integrates the vision–language reasoning capabilities of GPT-4o with the generative strengths of FLUX.1-schnell and Imagen-3.0, complemented by contextual grounding via UniVG-R1 and precise object isolation using the Segment Anything Model (SAM) and object peeling techniques. This framework synthesizes high-quality, context-rich triplets—comprising source images, edit instructions, and target composites—along with comprehensive metadata describing the object, its location, and the surrounding context.

2. **Point-based Spatial Localization:** Rather than predicting bounding boxes—which are ill-defined when the target object is absent from the input image—we fine-tune the Florence-2-Large model to predict a single spatial point indicating the optimal insertion location. This point-based formulation reduces ambiguity, improves reliability, and facilitates accurate grounding of object additions.
3. **Context-aware Disambiguation:** By incorporating context descriptions and bounding boxes derived from UniVG-R1, the fine-tuned Florence-2-Large model demonstrates a superior ability to disambiguate between multiple visually similar entities, ensuring that the predicted insertion point aligns with the intended subject specified in the instruction.
4. **Spatially-guided Generation:** We fine-tune FLUX.1-dev to condition on three inputs: the original image without the object, the natural language edit instruction, and the Florence-predicted insertion point. This explicit spatial conditioning enhances object placement accuracy while preserving the integrity of unrelated image regions and maintaining subject identity.
5. **Artifact-aware Training Strategy:** To mitigate biases introduced by synthetic data, we introduce a targeted transformation strategy involving random inpainting of non-target regions in the source image during training. This approach discourages the model from exploiting low-level synthetic artifacts and promotes generalization to real-world data.

4.2 ADVANTAGES

The proposed approach offers several distinct advantages over existing methods:

1. **High-precision Spatial Guidance:** The Florence-predicted spatial point, informed by rich contextual cues, enables precise object localization even in scenes with multiple similar subjects.
2. **Enhanced Identity Preservation:** By restricting edits to the minimal necessary region and conditioning generation on accurate spatial cues, the method maintains the original subject's distinctive visual attributes and fine-grained features.
3. **Improved Placement Accuracy in Ambiguous Scenes:** The combination of Florence's disambiguation capability with FLUX's generative quality significantly reduces errors in object placement, particularly in multi-entity compositions.
4. **Full Automation Without Manual Annotations:** The pipeline eliminates the need for manual mask creation, bounding box specification, or reference images, thereby enabling scalable deployment in both consumer-facing and professional editing applications.
5. **Robustness to Synthetic Data Bias:** The inpainting-based artifact reduction strategy ensures that the model learns semantically relevant insertion cues rather than overfitting to generation-specific visual patterns.
6. **Scalable and Flexible Deployment:** The system supports a wide range of object types and scene complexities, enabling on-demand, natural-language-driven customization for applications spanning fashion visualization, AR/VR experiences, photo editing, and beyond.

5 PRODUCT USE CASES

To demonstrate the real-world impact and broad applicability of our object addition pipeline, we highlight a range of product scenarios across visualizing styles, social media, professional editing, AR/ VR, and beyond.

- **Visualizing styles:**
    1. Allow customers to see how a multitude of accessories look on themselves or on models, directly in the browser or app.
    2. A customer needs only to submit their original image along with a concise textual directive — "add a wide-brimmed hat," "display a baseball cap," "render a wool beanie" — and the system will seamlessly generate a composite with each specified

variation in situ, without depending on pre-existing product assets, thus enabling efficient, on-demand evaluation of styling options. It can be processed on an instance with several subjects where only selected ones are supposed to wear some specific objects.

- **Social Media and Messaging Filters:**
  1. Integrate into apps like Instagram, Snapchat or WhatsApp as "one-click" filters to add fun hats, glasses, or themed headgear onto pets or people.
  2. In a chat app, a user sends a photo of their pet and the (voice) message "put sunglasses on my dog." The filter instantly applies sunglasses in the correct location.
- **Professional Photo Editing Suites:**
  1. Embed as a plugin for Photoshop, GIMP or Affinity Photo so that photographers can quickly add objects — onto subjects, (for instance, addition of objects preserving hairlines and facial features).
  2. This cuts down manual selection time and improves consistency across large batches of wedding, portrait or product shoots.
- **Augmented Reality (AR) Apps and Games:**
  1. Power real-time AR experiences where characters (animals or avatars) automatically don themed headgear in response to voice or text prompts ("Give my dog a superhero mask!").
  2. Use in mobile games to let players customize pets or avatars on the fly. In an AR selfie app, you simply say "give me cat ears" or "put a tiara on the princess" while pointing your camera. The app renders the objects in real time using only your camera view and the text command.
- **Film, Animation and VFX Previsualization:**
  1. Assist directors and VFX artists in quickly prototyping costume or prop placement on characters before committing to expensive shoots or CGI.
  2. Maintains fidelity to original footage, making it easier to integrate with live-action plates.
- **Educational and Training Tools:**
  1. In language-learning or storytelling apps, visually demonstrate vocabulary ("place a backpack on the police dog") by dynamically editing illustrations or photos, reinforcing comprehension.
  2. Use in training or educational modules to demonstrate correct placement of objects or gear on specific entities within an image (e.g., "placing a night vision camera on the commando on the left").
- **Accessible Photo Customization:**
  1. For users, a simple voice command like — "add a nurse's cap on the portrait" suffices — no dragging or precise clicks, and no need to upload a separate cap graphic.

6 RESULTS AND DEMONSTRATION

To evaluate the effectiveness of our proposed pipeline, we compare its performance against representative state-of-the-art methods, including GPT-Image-1 and FLUX.1-Kontext, on the task of natural-language guided object addition (in images including complex, multi-subject scenes). The qualitative examples in Table **??** illustrate the improvements achieved by our method in terms of spatial accuracy, contextual disambiguation, and preservation of visual fidelity.

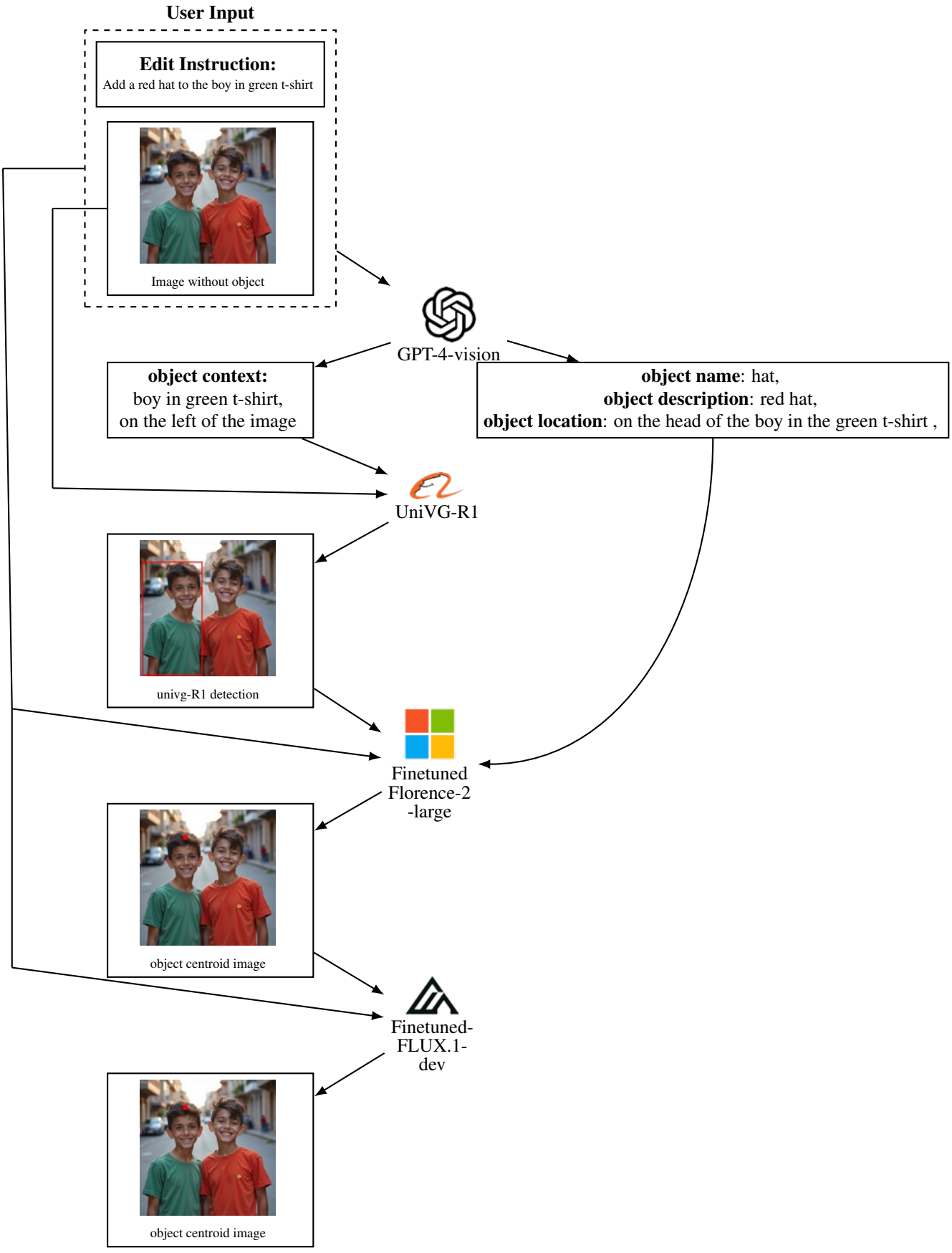


Figure 7: Inference pipeline Example

6.1 EDITING ASSETS WITH (AT MOST) ONE CONTEXT FOR THE OBJECT OF INSERTION

In this section, we present the results of addition of objects to images through edit instructions, wherein the source image itself contains at most one subject (that is, the image without object contains at most one context wrt the object described in the edit instruction).

Image without object **Image with object**

Edit instruction

Make the person wear a black leather jacket.

Add high-ankle shoes.

The person is wearing a pair of sunglasses.

Add a red jacket.

Add a tie.

Ensure the girl is wearing headphones.

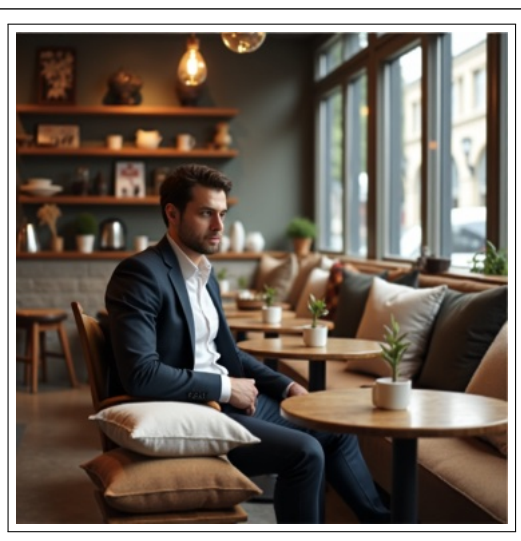
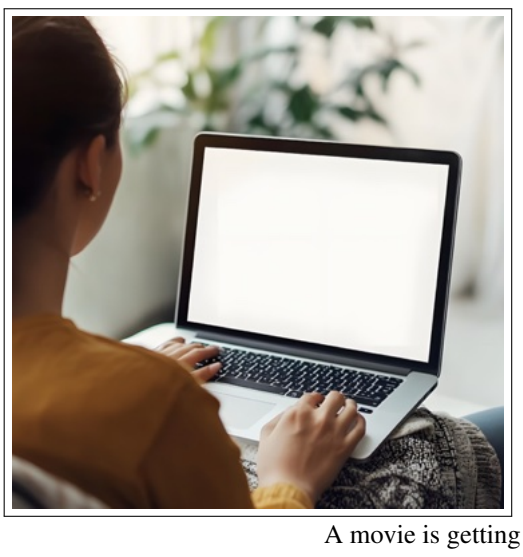
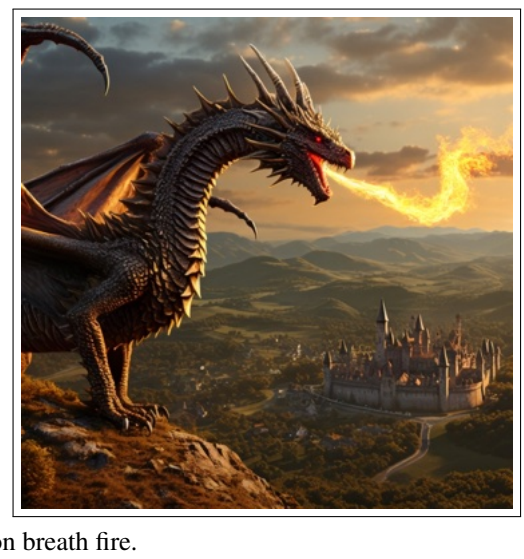
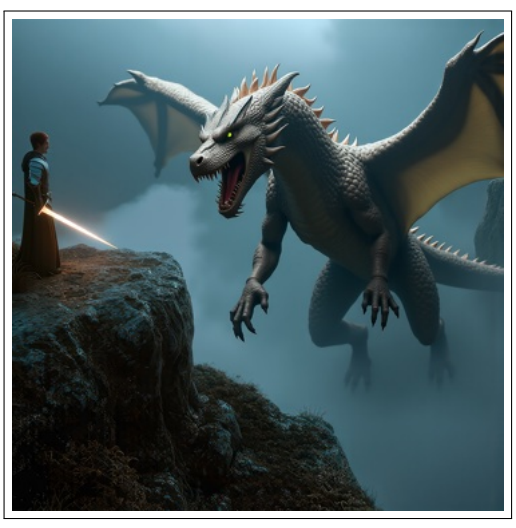
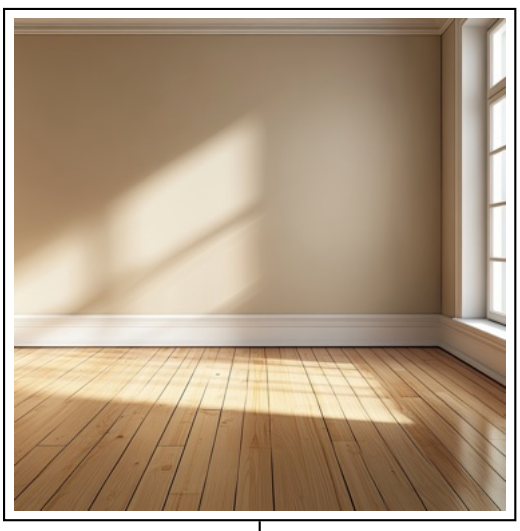
Image without object	Image with object
Edit instruction	
	
Insert a headband.	
	
A person is sitting (on the left side of the image).	
	
Display the world-map on the screen.	
	
A movie is getting played on the screen.	
	
Add a truck carrying wooden logs.	
	
Make the dragon breath fire.	

Image without object	Image with object
Edit instruction	
	
Add a person working on the laptop.	
	
Insert a dog next to the bench.	
	
Place an alarm clock on the sofa.	
	
There is a dragon in front of the samurai	
	
Insert a flower pot.	
	
Insert a table and a chair in the garden. The table is on the left side of the image and the chair is on the right side. There is a cup and a plate on the table.	

6.2 SEQUENTIAL EDITING

In this section, we will perform sequential editing on the image of a home interior and subsequently generate manifold visualizations following successive addition of objects.

base image of the house interior



Insert an extended sofa
with five pillows at the corner.

There is a small black table
in front of the sofa with a toy on it.

Put a floor lamp
left to the sofa.

Add a huge batman logo
on the wall.

Add an artwork of butterflies
on the wall.

Place a colorful abstract art
on the wall.

There is a mat
in front of the table.

Make a brown teddy bear sit
at the sofa corner.

Some books are there
on the table.

resultant image of the house interior

6.3 EDITING ASSETS WITH (AT LEAST) TWO ANALOGOUS OBJECT-CONTEXTS

In this section, we present the results of addition of objects to images through edit instructions, wherein the source image itself contains at least two analogous subjects (that is, the image without object contains at least two homogeneous contexts wrt the object described in the edit instruction).

Image without object	**Image with object**	**Edit instruction**
		Make younger person wear a cowboy hat.
		There is ”R&K” in circle written on the back of the tshirt of the person on right.
		Make the child wear a pair of spotted pants.
		The person sitting is wearing a pair of sunglasses.
		Make the older person wear a pair of sunglasses.
		Add a headband to the younger person.

Table 2: caption

Image without object	Image with object
Edit instruction	
	
The cat has a hat on its head.	
	
Place a stethoscope on the doctor's shoulder.	
	
Add a blue cap to the person on left.	
	
Add a red beanie to the person with long hair.	
	
Add a small heart tattoo to the neck of the lady on the left.	
	
Put a red conical hat on the little girl.	

Table 3: caption

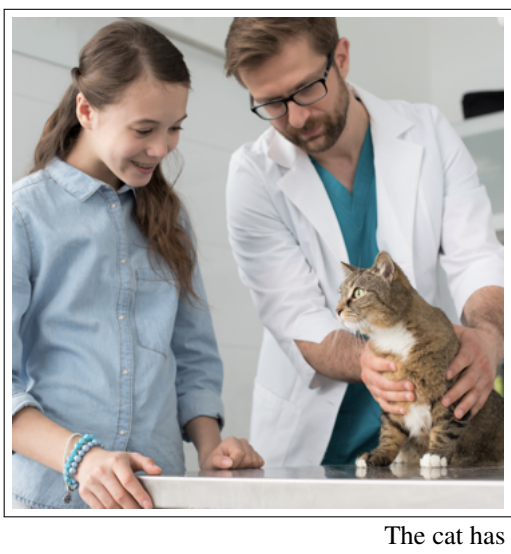
Image without object	Image with object	Edit instruction
		The cat has a hat on its head.
		Place a stethoscope on the doctor's shoulder.
		Add a blue cap to the person on left.
		Add a red beanie to the person with long hair.
		Add a small heart tattoo to the neck of the lady on the left.
		Put a red conical hat on the little girl.

Table 4: caption

7 RESOURCES

All resources developed in the course of this work, including the complete implementation of our pipeline, the curated dataset, and the fine-tuned model checkpoints, are available at:
https://github.com/amitvikrams_adobe/object-add

8 REFERENCES

[1] Adobe. object peeling-off, 2024.
https://adobe.brightidea.com/D54241.

[2] Google Deepmind. imagen-3.0, 2025.
https://ai.google.dev/gemini-api/docs/imagen.

[3] Sule Bai et. al. univg-r1: reasoning guided universal visual grounding with rl, 2025.
https://arxiv.org/abs/2505.14231.

[4] Black Forest Labs. flux.1-dev, 2024.
https://huggingface.co/black-forest-labs/flux.1-dev.

[5] Black Forest Labs. flux.1-schnell, 2024.
https://huggingface.co/black-forest-labs/flux.1-schnell.

[6] Black Forest Labs. flux.1-kontext, 2025.
https://huggingface.co/black-forest-labs/flux.1-Kontext-dev.

[7] Meta. Sam, 2023.
https://segment-anything.com/.

[8] Microsoft. florence-2-large, 2025.
https://huggingface.co/microsoft/Florence-2-large.

[9] OpenAI. gpt-4-vision, 2025.
https://platform.openai.com/docs/guides/images-vision.

[10] OpenAI. gpt-4o, 2025.
https://platform.openai.com/docs/models/gpt-4o.

[11] OpenAI. gpt-image-1, 2025.
https://platform.openai.com/docs/models/gpt-image-1.

[12] Zhenxiong Tan, Songhua Liu, Xingyi Yang, Qiaochu Xue, and Xinchao Wang. Ominicontrol: Minimal and universal control for diffusion transformer, 2025. URL: https://arxiv.org/abs/2411.15098, arXiv:2411.15098.