DepthScape: Authoring 2.5D Designs via Depth Estimation, Semantic Understanding, and Geometry Extraction

Xia Su*
Paul G. Allen School of Computer Science and Engineering
University of Washington
Seattle, Washington, USA
xiasu@cs.washington.edu

Cuong Nguyen
Adobe Research
San Francisco, California, USA
cunguyen@adobe.com

Matheus A Gadelha
Adobe Research
Seattle, Washington, USA
gadelha@adobe.com

Jon E. Froehlich
Paul G. Allen School of Computer Science & Engineering
University of Washington
Seattle, Washington, USA
jonf@cs.uw.edu

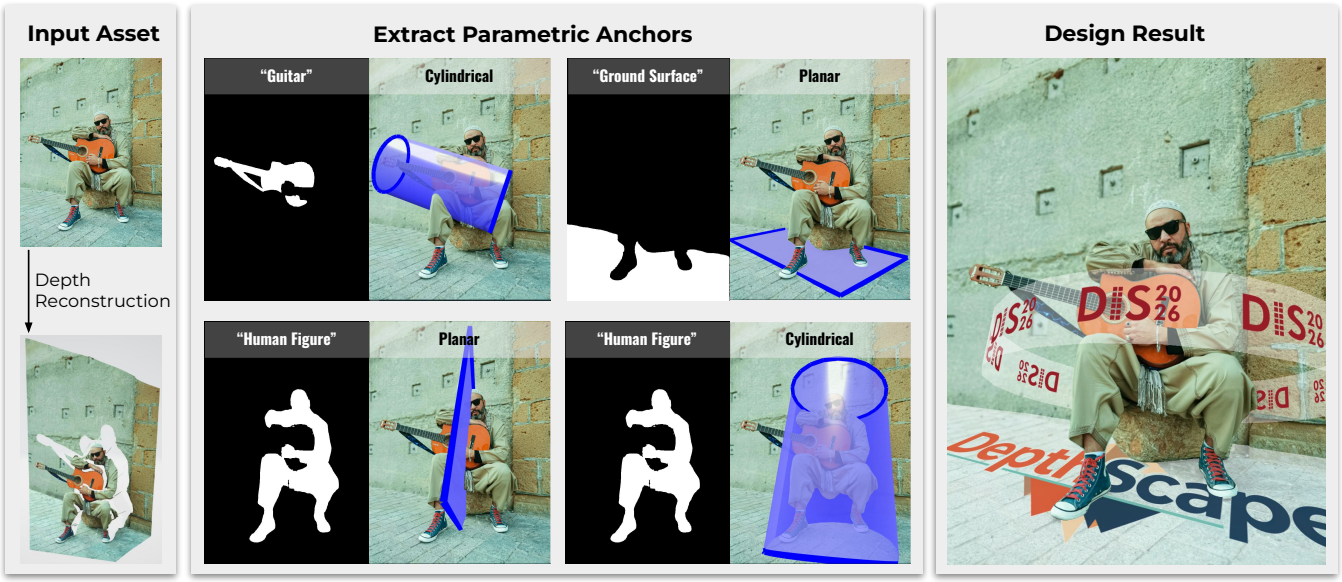


Figure 1: We introduce *DepthScape*, a Human-AI collaborative authoring system for 2.5D visual design. DepthScape takes input image assets and uses 3D reconstruction to estimate its inherent depth information. With AI-assisted design recommendation, users can quickly layout design elements in the implicit 3D space. The output is a visual design with realistic occlusion effects following depth cues in the input image.

Abstract

2.5D effects, such as occlusion and perspective foreshortening, enhance visual dynamics and realism by introducing 3D depth cues into 2D designs. However, creating these effects remains challenging, as designers must manually infer and author depth relationships—such as relative ordering, occlusion boundaries, and perspective scaling—within 2D representations. We introduce DepthScape, a human–AI collaborative system that facilitates 2.5D effect creation by placing design elements directly into 3D reconstructions. Using monocular depth reconstruction, DepthScape transforms images into 3D scenes, enabling depth-based blending that produces realistic occlusion and perspective foreshortening. To simplify 3D placement, DepthScape leverages a vision-language model to analyze source images and extract key visual components as parametric anchors, which support direct manipulation editing. The system design was iteratively refined through a formative user study with an early prototype. We evaluate DepthScape through a technical

*Work completed during an internship at Adobe Research.

This work is licensed under a Creative Commons Attribution 4.0 International License.

DIS '26, Singapore, Singapore

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2563-0/26/06
https://doi.org/10.1145/3800645.3813038

study on 100 professional stock images to assess robustness, alongside an expert evaluation confirming design quality, usefulness, and broad application potential, further illustrated through five example scenarios.

CCS Concepts

• **Human-centered computing → Interactive systems and tools**.

Keywords

2.5D Design, Depth Estimation, Vision Language Model, Semantic Understanding, Geometry Extraction, Design Tool, Creativity Support

ACM Reference Format:
Xia Su, Cuong Nguyen, Matheus A Gadelha, and Jon E. Froehlich. 2026. DepthScape: Authoring 2.5D Designs via Depth Estimation, Semantic Understanding, and Geometry Extraction. In *Designing Interactive Systems Conference (DIS '26), June 13–17, 2026, Singapore, Singapore.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3800645.3813038

1 Introduction

Humans gain 3D perception from 2D images through depth cues like occlusion, perspective, and shading [6, 15, 25]. Leveraging this visual ability, creators have long applied depth-enhancing techniques, such as layering and projecting, to boost visual impact and realism, leading to 2.5D visual designs. However, creating 2.5D visual designs can be challenging and arduous, as designers must manually imitate depth cues—such as relative ordering, occlusion boundaries, and perspective scaling—within flat 2D representations. This requires professional editing skills and iterative trial-and-error.

For example, to create a simple "surrounding figure" effect (Figure 2 BC), the creator needs to simulate the perspective placement of the content ring and carefully layer it partially in front of and partially behind the figure. This can be done using 2D design tools like Photoshop or Illustrator, but it relies on manually shaping and aligning visual elements to match 3D perspective; as a result, even small adjustments require repeated refinements, making iteration slow and frustrating. Alternatively, the creator may choose a native 3D tool like Blender, which sets up a 3D scene to obtain accurate depth cues and vanishing points. However, 3D tools have steep learning curves and introduce additional challenges, including aligning 3D camera viewpoints and projection geometry with the source image. Both workflows shift attention from creative exploration to technical problem-solving, rendering the process tedious, error-prone, and resistant to rapid experimentation.

We introduce DepthScape, a human–AI collaborative authoring interface that bridges 2D editing with 3D rendering, enabling intuitive creation of 2.5D visual effects without requiring 3D expertise. DepthScape builds on two interaction innovations: (i) **depth-based blending** that renders text and images in the 3D depth scenes of the source images to enable precise occlusion and perspective foreshortening; (ii) **parametric anchors** that bind content layers to objects in the original scene thereby enabling 3D placement through a 2D viewport. When the user uploads an input image, DepthScape uses monocular depth estimation [33] to produce a 3D scene where visual elements can be rendered with realistic occlusion and perspective that aligns with the original image.

To further simplify the 3D placement task, DepthScape explores a scalable and generalizable vision–language model (VLM) [16] pipeline that jointly interpret semantic content (what is in the image) and spatial structure (layout and shape) to synthesize *visual programs* [13], which guide 3D geometry extraction that mask key objects from image [30], and fit planes [22], cylinders, spheres, and skeletons [21] to them, resulting in parametric anchors that align with key objects in the input image and support direct-manipulation editing. While we focus on VLM-based pipeline in this paper, we note that user inputs remain valuable and highly complementary, and position our approach as a first step toward a hybrid human–AI pipeline for anchor generation.

We created DepthScape using a human-centered, iterative design process. We initially started with a proof-of-concept prototype to test the feasibility of depth-based blending and AI design assistance for 2.5D design. We conducted a formative user study with nine participants to observe usage patterns, evaluate usability, explore potential application scenarios, and gather feedback on interaction methods. Participants replicated example 2.5D designs and created open-ended designs, confirming that our approach facilitate 2.5D visual design by (i) offering recommendations that help novice designers explore what is possible, and (ii) enabling experienced designers to rapidly try ideas. The study also uncovered desired system improvements in rendering, AI recommendation, and editing interactions. Findings informed the development of our final DepthScape system, featuring auto-extraction of parametric anchors as a stronger AI design assistance, which also enables 3D placement through direct manipulation in 2D viewport.

To evaluate the final DepthScape prototype, we performed a technical evaluation with 100 professional stock photos that reflects real-world design types and content. Feeding these images into DepthScape and logging the outputs, we confirm the efficiency, robustness, and output diversity of our creation pipeline. We also conducted an expert evaluation session with three professional designers, who positively rated the quality of the suggested anchors and design outputs. Finally, we close with a demonstration of DepthScape's potential and flexibility by introducing five application scenarios including 2.5D video creation, real-world scene modification, and also storyboarding for hand-sketched scenes.

Our contribution is threefold. First, we propose a novel depth-based blending authoring pipeline for 2.5D visual designs, which places visual elements into monocular depth reconstructions of input images to enable precise occlusion and perspective foreshortening. Second, we employ a VLM to generate visual programs to automatically extract parametric anchors from input images based on image semantics, which further enables direct-manipulative editing. Third, we demonstrate the flexibility, robustness, and potential of our approach via our formative user study, technical performance evaluations with expert review, and five application examples.

2 Related Work

We situate our contributions in the literature on 2.5D design, depth-aware design tools, and semantic geometry extraction.

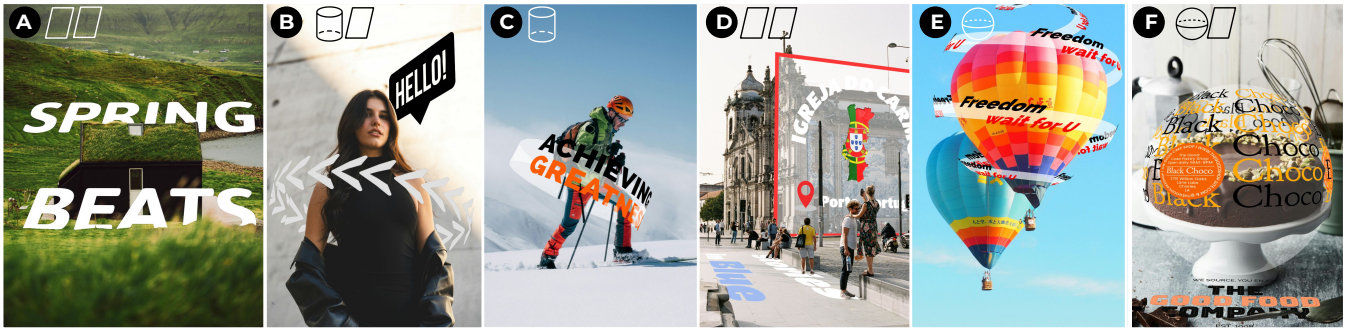


Figure 2: Design result gallery of DepthScape. Top row icons show the formation of designs, with one or multiple Planar, Cylindrical, and Spherical parametric anchors.

2.1 2.5D design

Humans gain 3D perception not only from binocular vision but also monocular cues to depth, such as linear perspective, interposition, Gestalt principles, and shadows [6, 15, 25]. Since such depth perception comes from how people mentally process visual information, some researchers also call them *psychological* depth cues [24]. Techniques based on this instinctive human ability are widely applied in graphics research and engineering. Since the 1970s, developers have been using methods like scaling sprites [1, 3–5] and parallax scrolling [2] to create pseudo-3D effects in arcade games. Though it is hard to identify the exact origin, the term *2.5D* gradually emerged from the animation and gaming community in the 1970s and 1980s to refer to the pseudo-3D effects seen in the 2D visuals.

Similar techniques are also widely applied in visual arts, from paintings — artists like Rembrandt manifest a full working knowledge of all monocular cues to depth in their pieces back to 17th century [15]— to modern designs of posters [17], illustrations [23], websites, *etc.* The development of modern creation tools like Adobe PhotoShop and Adobe Illustrator further simplifies and supports the creation of visual depth cues, thanks to their layering and distorting capabilities. This further blurs the line between 2D and 3D design spaces. To our knowledge, there isn't a clear definition for these depth-aware designs, either as a design genre or as a specific technique. In this case, we borrow from the gaming community, graphics research community [7, 11, 12, 26, 35, 42], and recent creativity support [46] and VR [9] research to address this visual design type as *2.5D design.*

2.2 Depth-aware Design Tools

In addition to using design techniques and graphic technologies to achieve 2.5D visual perception, researchers have also increasingly leveraged real estimated depth to assist creative tasks. With recent advancements in computer vision making monocular depth estimation more accurate and efficient, the depth information of a single image can now be easily obtained, whether as a depth map [27, 40, 41, 47] or a 3D model [33, 34, 37, 38, 45]. These output results can be applied in various creative applications. For example, *ZoomShop* [19] uses image depth information to edit image composition, enlarge distant objects, and adjust the relative size and positions of objects. Similarly, *VideoDoodle* [43] uses depth information of video scenes to blend hand-drawn animations into real-world video scenes. Lu *et al.*[20] use depth maps to vectorize images and better capture key contours in the input image.

In addition to using real or estimated depth information, existing research also enables creators to arrange 2D elements in 3D spaces to better support their creation. The *Mental Canvas* [8] allows creators to organize 2D architecture sketches into 3D space to ensure geometric consistency; *Stereoboard* [14] also allows users to arrange 2D storyboard sketches in 3D space so that they can better align with cinematic constructs. *PortalInk* [46] automatically arranges depth-based layers of 2D drawings into 3D spaces to create parallax effects and export 2.5D visual stories. By organizing 2D content in 3D spaces, these tools ensure the 2D content follow 3D depth cues and spatial relations.

Following this thread of work, we observe insights into how depth information helps creation tasks: depth information provides spatial details that ensure content blending true to reality. As 2.5D visual designs are also heavily dependent on depth perception and aim for visual realism, this observation inspires our **depth-based blending** method, which use reconstructed 3D depth scenes to blend and render content layers.

2.3 Semantic Geometry Extraction

To better understand reconstructed 3D scenes and support creativity, we aim to automatically extract geometry from the reconstructed depth scenes. Traditional techniques in geometry fitting like RANSAC [10] and tools such as PyRANSAC-3D [22] are commonly used to fit geometric primitives (*e.g.*, planes, spheres, cylinders) to noisy point cloud data. These methods are often paired with supervised segmentation models to extract meaningful structures from real-world scans [18, 36, 39]. As recent advancements in vision-language models (VLMs) [29, 44] have enabled AI systems to understand visual content in a zero-shot way, it is now possible to conduct such geometry extraction without any training data. For example, using zero-shot text-based segmentation like Grounded-SAM [31], which segments certain parts in images with input text prompts, we can automate the selection of certain objects in reconstructed point clouds, which can be subsequently fitted with primitive shapes. Combined with other CV models like MediaPipe [21] which detects human skeletons and landmarks, we can further expand this detection capability to match wider design intents.

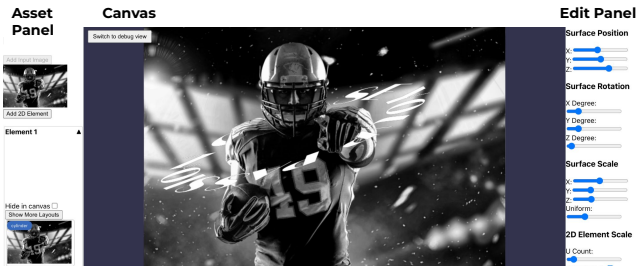


Figure 3: Proof-of-concept prototype interface.

Inspired by Visual Programming [13], which demonstrates how large language models (LLMs) can orchestrate modular computer vision tasks by generating visual programs that combine off-the-shelf computer vision modules, we explore a hybrid method to semantically extract geometric primitives from the reconstructed depth spaces. Our pipeline combines VLM-based semantic grounding and segmenting with classical geometric fitting, aiming for a zero-shot, modular approach to identify planar, cylindrical, and spherical structures that can be content anchors that accelerate the creation of 2.5D effects. This enables users to manipulate design elements in a depth-aware space while maintaining intuitive semantic control.

3 Designing DepthScape

Building upon the core idea of **depth-based blending**-placing content into reconstructed depth spaces-we conducted a three-stage iterative process to test its feasibility, explore user needs, and discover key interaction patterns. We first built a *proof-of-concept prototype* to test feasibility. Then, we used our prototype to conduct a *formative user study* with nine participants of varying design expertise. Finally, based on the study findings, we built the *final DepthScape system* with improved depth reconstruction, user interaction, and AI design suggestions.

3.1 Proof-of-Concept Prototype

We built a proof-of-concept prototype Figure 3 as a web interface. The prototype uses a single-image-to-3D-mesh model *CRM* [34] to reconstruct the main object of a user-uploaded image into a 3D mesh. We then add primitives like planes, cylinders, and spheres to render realistic occlusion and perspective effects. Users can edit the placement of these primitives with a series of sliders that adjust the position, rotation, scaling, and content placement of the primitive surfaces. We also explored AI recommendation of 3D placement by leveraging *CLIP* [29] and cosine similarity to retrieve 3D placement parameters based on the similarity of input images. The recommended parameters are rendered as thumbnail images for users to browse and select.

3.2 Formative User Study with the Prototype

To test the feasibility, usability, and helpfulness of our proposed 2.5D creation pipeline, we conducted a formative user study with the prototype among nine participants aged 20-60. To understand the design challenges of both expert and novice designers, we

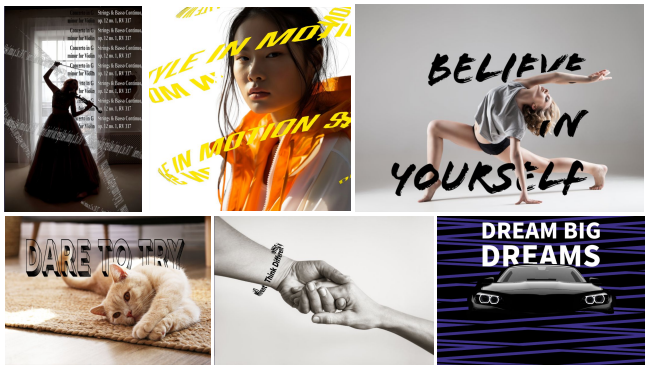


Figure 4: Design results of the open-ended exploration of the formative user study.

recruited four design enthusiasts and five professional designers, all with varying levels of 2D visual design and 3D design expertise. We presented the prototype interface and guided participants to recreate two example designs and then freely create designs as they wished. We finished the study with usability questions and Likert-scale ratings. More details of the study protocol and findings can be found in our prior work [32].

All participants successfully utilized the prototype interface to create multiple 2.5D designs. Participants highly agreed that *"the prototype supported my creativity."* (avg = 4.67/5). They also commonly agree that the prototype made occlusion (avg = 4.56/5) and surrounding effects (avg = 4.56/5) easy to achieve. We observed diverse and visually compelling designs in the open-ended designs Figure 4 .

The formative user study reveals consistent application scenarios centered around rapid design prototyping and ideation for 2.5D visual effects. Participants consistently identified the tool as ideal for *"trying out ideas at early stage"* where they could *"get a few AI brainstorm suggestions"* and create *"rough mock ups"* before transitioning to higher-fidelity tools (P8). Users emphasized it makes *"occlusion and surround effects easier and faster"*, that would otherwise require complex 3D software knowledge (P1). The overarching theme positions DepthScape as a specialized creative sandbox for exploring pseudo-3D design possibilities before committing to production-ready tools like Photoshop or 3D software.

Importantly, the tool was perceived as valuable to both amateurs and professionals, though in distinct ways. Amateur designers valued its accessibility and freedom to experiment, noting that it *"enables you to do a lot of different stuff"* (P9). The intuitive depth cues and playful exploration encouraged creativity without requiring specialized 3D knowledge. Professionals, meanwhile, evaluated the tool through the lens of established workflows and quality standards. They appreciated the unique affordances but desired tighter integration with existing tools: as P6 remarked, *"(It's okay that) We don't have all those text editing capabilities because every other tool already has that. We don't want to rebuild the wheel."*. In summary, amateurs embraced it as an accessible entry point into 2.5D design, while professionals positioned it as a valuable complement to their advanced toolchains.

From these design explorations, we confirm the usability of the proposed editing method for both amateur and professional designers and observe three ways in which the prototype supports creativity: (1) the accurate depth reconstruction simplifies the creation of realistic occlusion effects; (2) the capability of rendering complex deformation and repetition effects in 3D space supports the creation of complex but orderly perspective effects; (3) the placement parameters create a design space that is large enough to accommodate serendipitous designs.

3.3 Improvements Based on Study

Besides the overall positive findings, we also observed aspects that need improvement. We elaborate on the key findings that support our iteration of system design.

Orthographic -> Perspective. Due to the reconstruction feature of CRM [34] and most existing depth estimation models [27, 37, 38, 40, 41, 45, 47], the reconstruction results match the input images only in orthographic rendering. Thus, our initial prototype renders the design canvas with an orthographic camera. This eliminates perspective foreshortening hence many participants found the depth effects hard to perceive and less visually impactful. In this case, we aim to switch to perspective rendering for both the depth reconstruction and the added visual elements.

Reconstructing Main Object -> Entire Scene The depth reconstruction model in our prototype system only captures the main object of input images, limiting the application to single-object images. More complex scenes, such as urban, natural, or even a dinner tabletop, cannot be supported. To expand the application to more generic images, we aim to switch to models that reconstruct the depth of entire 3D scenes in the input images.

Sliders -> Direct Manipulation. Many participants, especially those with less 3D design skills, find it hard to place elements into 3D spaces with parameter sliders. On one hand, they do not understand the meanings of the parameters and lack expectations for editing results; thus, they are trapped in a cycle of trial and error. On the other hand, the mapping between a conceptual goal of 3D placement, *i.e.*, surrounding the bullet with a ring of text (B), involves a complex interplay of multiple 3D placement parameters and is difficult to achieve. As participants strongly prefer direct manipulation controls that match design intentions and image semantics, we aim to better parse the input images and depth reconstructions to provide simpler and smarter controls. For example, when the image contains a bullet shape, our system should be able to understand the semantics and suggest surrounding placements that align with the bullet's direction and enable users to directly manipulate key parameters, like the cylinder radius.

Recommend Parameters -> Parametric Anchors Our initial exploration of AI recommendations for 3D placement parameters was not sufficiently helpful (rated avg= 3.67/5) since the retrieval-based parameter matching is not precise regarding image semantics and depth geometry. As visual designs are very specific and precise about layout, slight mismatches between image content and 3D placement can break the visual effects. In this case, we aim to provide a more semantic-aware parsing of the input image with VLM and directly extract geometry from the depth reconstruction to ensure the precision of placement. This leads to the extraction of **parametric anchors** which bind content layers to image objects.

4 The DepthScape System

Building on the above iterations, we introduce *DepthScape*, a novel Human-AI collaborative authoring pipeline that facilitates 2.5D graphic designs with depth estimation, semantic understanding, and geometry extraction. DepthScape employs a monocular geometry estimation model MoGE [33] to reconstruct 3D meshes that match visual depth cues in input images. This creates 3D spaces that can accommodate 2D design elements like text and images, enabling **depth-based blending** that yields realistic occlusion and perspective effects. The system further accelerates 2.5D design by parsing input images with VLM to extract key visual components and suggest **parametric anchors** in the 3D space that enable direct manipulative content placement.

4.1 Depth-based Blending

We use *MoGE* [33] to create a 3D mesh that reflects visual depth cues in a given 2D image. Unlike many existing depth estimation models, MoGE conducts affine-invariant monocular geometry estimation, with reconstruction results that match the input images in perspective rendering. This reconstructed 3D mesh encodes critical depth cues and serves as the foundation of our creation pipeline.

We then engineer a depth-based rendering by placing both the reconstructed depth mesh and any additional 2D visual elements into a shared 3D space(Figure 5 left). By default, rendering this 3D scene correctly produces occlusions between the depth mesh and the added 2D content. However, it also exposes the reconstructed depth mesh itself, whose visual quality is noticeably lower than that of original image. To address this, we engineer the rendering pipeline to make the depth mesh invisible while still creating occlusions. Specifically, we disable color pixel writing for the depth mesh, turning it into an invisible "ghost object" that blocks other elements and leaves holes in the rendered image. We fill these holes using pixels from the original input image so that the final output appears as if objects in the original image are occluding the added visual elements (Figure 5 right—note how the human figures occlude the text layers).

4.2 Parametric Anchors

During our formative user study, we observed challenges in precisely placing design elements in reconstructed 3D spaces. However, we also identified two design patterns that can help simplify the 3D placement task: (1) Design elements are often anchored to objects in the input images, *e.g.*, wrapping text around a human arm or aligning a plane with a building facade. (2) Designers explore variations by adjusting one or two key parameters while keeping others fixed to preserve design semantics, *e.g.*, shifting a plane along its normal vector while keeping it parallel to a building.

To formalize these patterns, we borrow the concept of *anchors* from augmented reality research to create *parametric anchors* that bind visual content to objects in images. Specifically, parametric anchors focus on certain objects in the input images, aligning the content with the objects' geometric properties. For example, planes parallel to a building facade, cylinders surrounding a human figure,

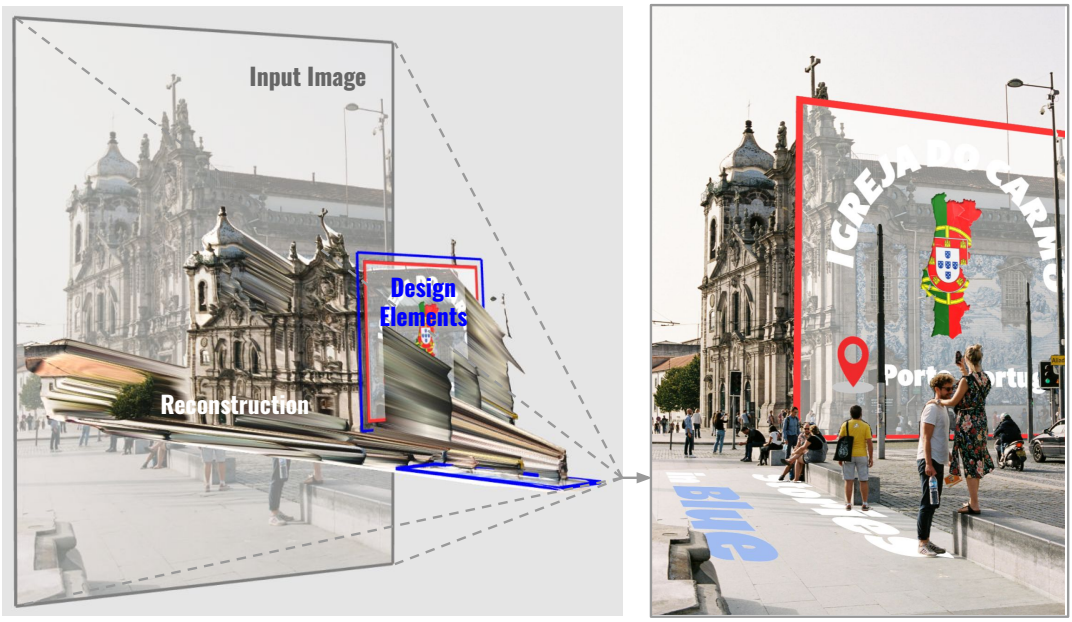


Figure 5: DepthScape's 3D space composition and rendering results. Left: 3D space viewed from a different angle. Text shows the original input image, the reconstructed pointcloud, and also the added in design elements. Gray dash indicates the camera position. Right: render results with design elements embedded into the scene of the input image.

and spheres that center around an owl (Figure 6). We use parametric anchors to constrain placement parameters and reduce degrees of freedom to the most relevant ones. For example, a Planar parametric anchor has a fixed reference plane and only moves along its normal direction. Similarly, a Cylindrical parametric anchor organizes placements along a shared central axis, allowing only radius or height adjustments; a Spherical parametric anchor centers placements around a fixed point, varying only in radius.

By leveraging these parametric anchors, we simplify unconstrained 3D placements into structured adjustments that align with scene geometry. This enables direct manipulation in a 2D viewport, which would otherwise be ambiguous. For instance, translating a plane freely in 3D from a 2D perspective is ill-defined, but within a Planar parametric anchor, it reduces to a simple 1D adjustment along the normal vector, which can be directly mapped to mouse input. See Figure 6 for examples.

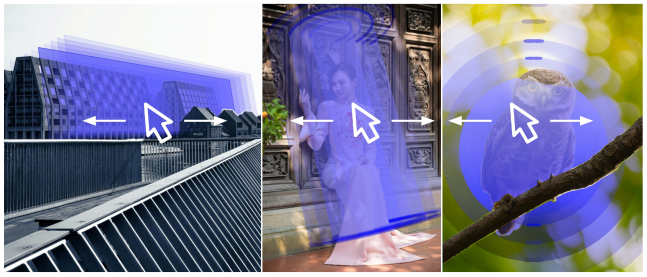


Figure 6: DepthScape currently supports three types of parametric anchors that can be direct manipulated by mouse moving. From left to right: Planar, Cylindrical, Spherical.

4.3 Extract Parametric Anchors using Code Synthesis

The extraction of parametric anchors remains challenging. On one hand, heuristic methods fail in open-vocabulary settings and struggle to holistically capture object type, shape, and spatial layout. On the other hand, manual approaches introduce additional learning overhead, requiring users to understand both the concept of anchors and the intermediate steps involved. To address these challenges, we explore a fully automatic pipeline. We leverage a VLM (gpt4o [16]) to semantically parse the input image and guide the geometry extraction processes. Inspired by a prior work [13], we prompt VLM to follow a visual program template and enable modular and sequential geometry extraction that focuses on certain visual components of the input images. Each VLM-generated visual program will be based on a design intention, and expand into a combination of geometry functions from the list below:

Segment the input image. A function `Text2Mask(prompt)` can segment the original input image with a text prompt, which is generated in place along with the visual program. Upon this function, our backend system uses grounded-SAM-2 [30] to generate corresponding image masks for the text prompt (*e.g.*, *prompt = "the train in the image"*). Subsequently, we select the vertices of the reconstructed depth mesh that fall inside the mask, with a `Mask2Pointcloud` cell. The selected point clouds can be used for further geometry extraction.

Extract 3D geometry. We implement a variety of geometry extraction functions, which are selected by the VLM based on image semantics.

`Pointcloud2Plane`: For planar objects like building facades, grounds, water surfaces, and tabletops. Implemented with PyRANSAC-3D [22] which fits planes to the point clouds. We also implement derived geometries from an extracted plane, such as the primary direction and the perpendicular extrusion, to empower more complex geometry references between different objects.

`Pointcloud2Cylinder`: For elongated objects like human figures, tree trunks, or trains. This cell extracts a containing cylinder, with an optional parameter of axis direction. When direction is not provided, we use PCA to find the primary component of the point cloud as the axis direction. We then find the center location that contains all the points with the smallest radius.

`Pointcloud2Sphere`: For rounded objects. This cell extracts a center and a minimum radius to contain the point cloud.

Besides these functions that extract primitive surfaces, we also implement functions that extract human pose and facial structures.

`Pointcloud2Skeleton`: This cell extracts key pose landmarks from human figures with mediapipe [21] and cast the 2D coordinates to the depth point cloud to yield 3D skeleton points. Such extracted 3D skeleton can then yield frontal and median planes to support content placement. Other derived directions, like upward directions and front-facing directions, can also be derived to be used elsewhere.

`Pointcloud2Face`: For human figures zooming in more on faces, we extract the face geometries. Similar to skeleton extraction, this cell extracts key facial landmarks with mediapipe [21] and casts the 2D points to the depth point cloud to yield 3D face points. This extracted 3D face can then yield frontal and median planes, as well as derived directions like upward and front-facing direction.

Create parametric anchors. We then wrap extracted geometries, either as planes, cylinders, spheres, or directions, into parametric anchors and finish the visual coding sequence.

`Planar(plane)`: Create a planar parametric anchor with an extracted or derived plane, and some optional directions.

`Cylindrical(cylinder)`: Create a cylindrical parametric anchor with an extracted cylinder.

`Spherical(sphere)`: Create a spherical parametric anchor with an extracted sphere and an optional direction.

By combining different functions, DepthScape extracts various types of geometry from depth reconstructions based on the image semantics. See Figure 7 for an example extraction, including the actual visual program syntax, the middle steps involved, and the final result as a parametric anchor.

We prompt VLM with a list of 16 visual programs to enforce the visual program syntax and general design patterns, each includes description of the raw input image and a matching visual program based on the content. See Appendix A for examples. The VLM then generates semantically reasonable visual programs based on the input images. DepthScape parses the generated visual programs and conducts the extraction pipeline to create parametric anchors. The successfully extracted parametric anchors, along with the original design rationale, will be shown in our design interface for users to interact with.

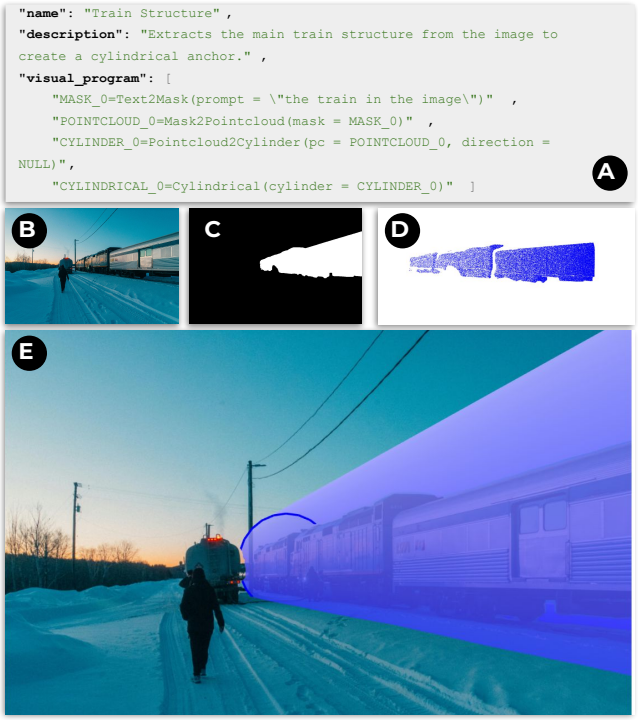


Figure 7: DepthScape employ a vision language model to (A) generate visual programs based on (B) input images. The program guides a cylinder extraction process, including (C) masking out the *prompt = "the train in the image"*, and (D) selecting the corresponding point cloud. The final result would be (E) an interactive cylinder that is precisely aligned with the train.

4.4 The DepthScape Interface

The DepthScape interface is implemented with *React* and *Babylon.js* as a webpage (Figure 8). The interface has three main parts: (1) the Asset Panel imports input images and shows suggested parametric anchors, as well as added layers in the current scene; (2) the canvas renders the 2.5D effects and enables direct manipulation of design elements; (3) the editing panel lists editing options for selected parametric anchors, including adding text or image content, and adjusting the design with parameters.

The user first imports an input image (Figure 8A), which is processed to reconstruct the 3D depth mesh and extract suggested parametric anchors in real time. Then the user can inspect each suggested parametric anchor by hovering on the enlisted options (Figure 8B), and click to select the ideal one.This triggers an initialization sequence for the selected anchor, which includes offsetting the surface, resizing the surface (for planar parametric anchors), centering the content's UV coordinates on the anchor, and scaling the content UVs. Each step is presented as a mouse-following direct-manipulation. After finishing the initialization sequence, the user can rotate and mirror the 2D content with buttons and a slider (Figure 8F, lower half). To further adjust the placement, we also include an advanced fine-tune panel which enables users to adjust



Figure 8: DepthScape's user interface. (A) The input button and the user uploaded image. (B) AI-suggested parametric anchors. (C) Added layers into the scene. User can select, hide/unhide or remove these layers. (D) The editing canvas that enables direct manipulation edits. (E) Parameter panel for the selected layer, during initialization. (F) After initialization, more editing options are enabled, including rotation of content, adding text or (G) image content, and (H) fine-tune the layer parameters.

anchor position, scale, and orientation, allowing for correction of inaccuracies and customization based on personal preferences (Figure 8H). When initially creating the placement, a white "LOREM IPSUM" text will be shown as placeholder content to help the placement. The user can then replace by typing in text or uploading images (Figure 8G). The final results can be one-click-exported as rendered images. See our video figure for a complete walk-through of our user interaction.

5 Evaluation

We evaluate DepthScape's performance in terms of efficiency, robustness, output diversity, and design quality. We construct a diverse image collection and run it through DepthScape to inspect the processing time, error rate, distribution of AI-suggested parametric anchors, as well as the design quality.

5.1 Collecting Test Images

To ensure a diverse selection of test images that reflect real-world designer needs, we collect images from Pexels [28], which is a free-stock image sharing platform. We download from Pexels' waterfall of trending images, which are constantly updated to reflect the popular choices of the designer community. We collected 50 images on two separate dates, creating a pool of 100 high-quality and diverse images. We observe four main content types from this collection: portrait, urban/building, nature/animal, and static objects. Figure 9 shows the distribution and example images.



Figure 9: Four types of test images.

5.2 Procedure

We sequentially upload all 100 collected images to DepthScape, and inspect the processing results including the processing time, visual programs generated by VLM, extraction results of each visual program, and also error logs of failure cases. We analyze the log data to understand the efficiency and robustness of the DepthScape system, and also the distribution of parametric anchor types and causes of errors.

Additionally, we evaluate the quality of the suggested parametric anchors and final outputs with an expert review session among three professional designers. See Table 1 for their demographics. We showcased DepthScape's UI recording along with the recommended parametric anchors for four images, one from each type of our image collection. We also presented design results to better showcase the system output quality (see Figure 10). We then collected Likert-scale ratings of the quality of parametric anchors and design outputs, while soliciting comments regarding the application scenarios and system improvements.



Figure 10: Example design results shown to experts.

Table 1: Expert review demographics

Expert	Gender	Age	YOE as Designer	Design Type
E1	Male	25-34	1.5	UI/UX
E2	Female	25-34	12	Game, UI/UX
E3	Male	45-54	25	Web, Animation, Motion Graphics

5.3 Technical Results

97/100 of our test images successfully went through the DepthScape system to generate at least one parametric anchor based on image semantics. On average, each image took 21.1 seconds (std=5.5s) to process and yields 4.2 (std=1.1) visual programs. 89% of all generated visual programs were successfully processed to yield parametric anchors available for user interaction.

Figure 12 shows the averaged distribution of processing time. Note that more than half of the processing time (63.7%, avg=14.06s, std=4.04s) is spent waiting for GPT responses, which can be easily reduced in future iterations with newer and faster models. The processing time for depth reconstruction (8.4%, avg=1.86s, std=2.02s), masking (13.1%, avg=2.89s, std=1.23s; each masking time avg=0.89s, std=0.19s), and geometry extraction (14.8%, avg=3.28s, std=1.30s) are fairly fast.

In total, 419 visual programs were generated by the VLM, among them 370 successfully went through geometry extraction processes and yielded interactive parametric anchors. Figure 11 shows the distribution of parametric anchor types of all generated visual programs. The most common parametric anchor type is Planar, which we further break down as object planes (180 occurrences, 42.96% of all) and skeletal/facial planes (113 occurrences, 26.97% of all). The VLM also suggested 94 Cylindrical and 28 Spherical visual programs.

We observe two types of errors in the 46 failed visual programs, all of which are Planar visual programs. The most common error type (34 occurrences) is failure to detect human skeletal and facial landmarks. This is due to incomplete or failed masking of human figures. The other type (12 occurrences) involves errors in visual program formatting, *e.g.*, hallucinated visual program functions, not ending with a valid parametric anchor. Figure 13A shows three failed images, in which human figure extraction failed.

We also observed errors that led to unsatisfactory parametric anchors. In Figure 13B, the depth reconstruction of a glass surface was flat and concealed all geometry behind the transparent surface. However, the VLM still suggested plane fitting for the ground surface behind the glass, leading to misaligned results. In Figure 13C, VLM suggested masking *"the cake slice"* to focus on the smaller slice in the image, but the segmentation model ended up masking the entire cake. In Figure 13D, the cylinder fitting module tried to capture the cake geometry but adopted a horizontal axis direction instead of the shorter but more semantically-valid vertical axis. Besides these geometric errors, we noticed that occluded or warped text layers can be difficult to read. As shown in Figure 10, balancing 2.5D visual realism with text readability depends heavily on the text content and can vary substantially across designs. Achieving an appropriate balance therefore requires careful design of the text layout, textual content, and occlusion ratio.

5.4 Expert Review Results

Experts rated our parametric anchor quality positively (7, 4, 5 on a 1-7 scale) as well as the design output quality (6, 4, 5). They appreciated DepthScape's ability to parse complex scenes and generate diverse parametric anchors based on image content. E1 remarked, *"I can totally imagine that a Vogue cover designer would use it very powerfully."* Experts noted that the current system is already well-suited for design hobbyists. E2 highlighted its accessibility, stating, *"I think the value of this is that 99% of people who have no idea how to do this [2.5D effects] will benefit a lot from this."* However, they also pointed out that if intended for professional use, there are several areas for improvement, *e.g.*, occlusion precision, rendering quality, and perspective accuracy. For instance, E3 observed that while the cylinders and spheres were effective, plane edges were sometimes slightly misaligned with the base objects, requiring manual adjustments to meet professional standards.

Design experts proposed various application scenarios. E1 suggested DepthScape could be used for various types of visual art, like book covers, T-shirts, and even memes. They also envisioned the possibility of generating 2.5D animations by simply binding a position parameter to time. E2 emphasized utility in design ideations, stating, *"I think it's nice to use [DepthScape] to quickly put together some concepts. It's definitely helpful to accelerate the workflow."* They also noted its potential for users of platforms like Canva or Instagram who want to create striking text effects without needing expertise in complex software like Photoshop. E3 suggested additional use cases, including UI mockups, presentation landing pages, and promotional materials. E3 was particularly interested in DepthScape's ability to analyze images in terms of both semantics and

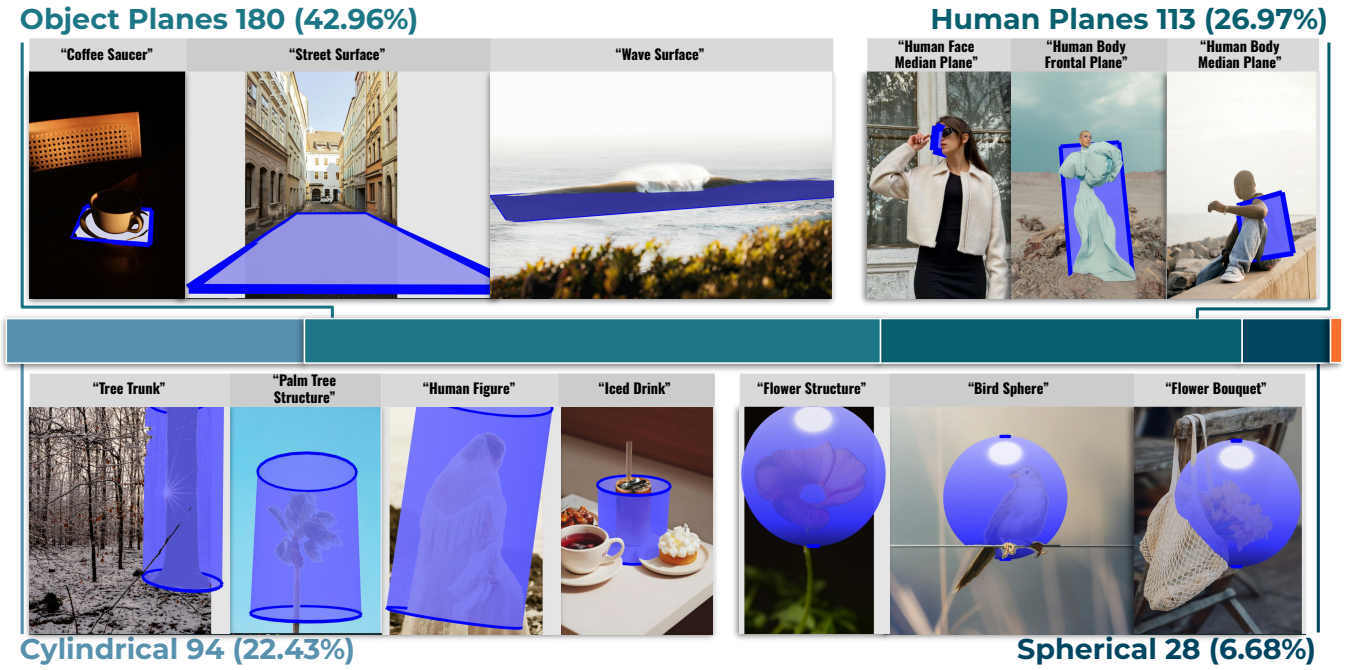


Figure 11: Four types of parametric anchors. Each example shows the name of the parametric anchor on top.

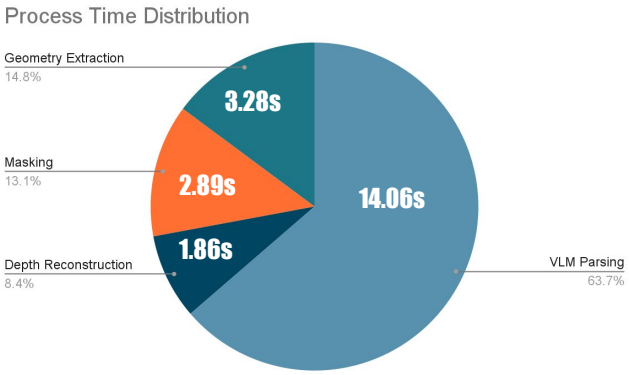


Figure 12: The distribution of processing time.

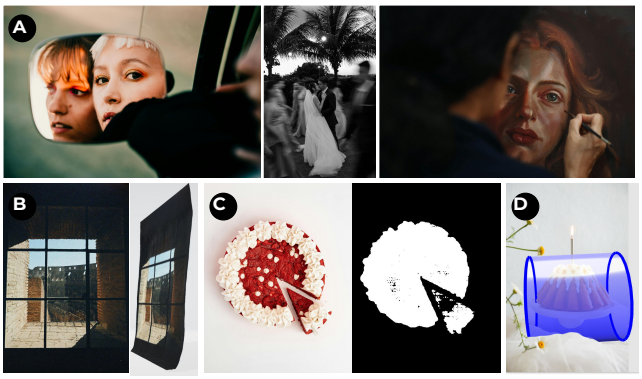


Figure 13: Failed cases. (A) Three images with no parametric anchor generated. (B) Depth reconstruction on a transparent glass surface loses the geometry behind the glass. (C) The masking of "the cake slice" ended up with the entire cake. (D) The cylinder fitting fails to capture the shorter but symmetrical axis.

depth, believing that its capacity could support the creation of video effects.

Experts also wished for additional features. E3 hoped for more sophisticated content anchors that incorporate multiple directional cues from the scene. They also emphasized the need for text readability enhancements when adding text elements. E2 envisioned DepthScape being integrated into existing design platforms like Figma, Canva, or Instagram, to streamline content creation. All three experts requested improved blending between the depth scene and added content, including support for light estimation, shadow casting, and blur effects to further enhance realism. Additionally, E2 and E3 advocated for the ability to export designs in editable, multi-layer formats (*e.g.*, PSD files) for further refinement in professional design software.

Overall, experts viewed DepthScape as a promising tool for enabling accessible and expressive 2.5D design, particularly for rapid ideation and creative exploration. At the same time, their feedback highlights a gap between current capabilities and the precision required for professional workflows, suggesting that future improvements should focus on enhancing geometric accuracy, visual

realism, and usability to better support high-quality design outcomes.

6 Application Scenarios

Inspired by our formative study and expert review, we discuss and implement five application scenarios to further demonstrate the creative power and future potential of DepthScape. Please also check the video figure for additional details and animations.

6.1 Integration into Image Editing Tools

Suggested by both our formative user study and expert review, DepthScape can serve as a good complementary tool for existing professional image editing tools.If integrated, the depth-based mixing in these image editing pipelines can be accelerated, and amateur users who do not know how to create 2.5D visual effects can be encouraged and enabled. In order to achieve this, one way is to include DepthScape as a plug-in for these platforms. Another simpler possibility is enabling DepthScape to export design results as layered and editable files and import them into existing platforms for further editing.

6.2 Video 2.5D Effects

DepthScape's 2.5D effects on static images can also be propagated to continuous image frames and create video 2.5D effects. Suggested by our study participants, there can be two types of video creation. First, in static images, the parameters of content placement can be bound to a time variable and rendered as animations. *E.g.* a surrounding effect of an arrow ring, or a moving effect of a text plane. Second, continuous frames of videos can be individually parsed and edited with consistent parameters to enable object-centric video editing. Examples of both cases are shown in the video figure.

6.3 Modify Real World Scenes

Since DepthScape parses and approximates real-world depth cues and blends extra visual elements into the original scene, it can be useful to quickly modify real-world scenes in realistic ways, especially by adding new objects into the scenes. One intuitive example is simulating interior designs by adding new decorative elements, like image assets of rugs or paintings, into the scene; another example is testing branding visuals like posters by adding them to public spaces. See Figure 14.

6.4 Simulate AR Scenes

By anchoring content to real-world objects and surfaces, DepthScape can also blend virtual assets, like UI and 3D models, into real-world images without actually tracking the environment. In this way, we are simulating and prototyping AR effects from single input images. For example, by detecting the planar orientation of the floor and wall, we can blend AR content into a real-world scene (Figure 15). We can also add animated 2D/3D content, like a video or a 3D Pokémon, into real-world scenes.



Figure 14: DepthScape can help adding contents, such as rug, wall painting, and ad posters, into real world scenes.

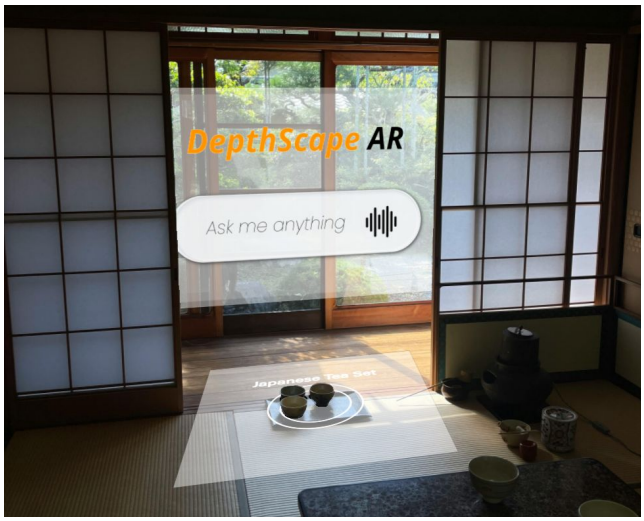


Figure 15: Simulated AR effects, including the plane and text for the tea set, and also the virtual UI located behind the sliding doors, in a real-world indoor scene.

6.5 Storyboarding

We observe that hand-sketched scenes also encode relative depth cues that can be reconstructed into 3D geometry. Thus DepthScape can also be applied to hand-sketched scenes, enabling rapid blending of 2D content for storyboarding. This capability supports content creators in visualizing and iterating on their designs more efficiently, allowing them to seamlessly integrate visual elements into sketched environments. In this case, DepthScape facilitates the blending of source materials with varying levels of fidelity, making it easier to unify assets ranging from rough sketches to more polished graphics. See Figure 16.

7 Discussion & Future Work

In this paper, we present *DepthScape*, a human-AI collaborative authoring tool that enables 2D designers to create rich 2.5D visual effects by simply arranging assets within a *monocular depth reconstruction*. A key strength of DepthScape is that it allows designers to work entirely on a 2D canvas to create diverse 2.5D effects with

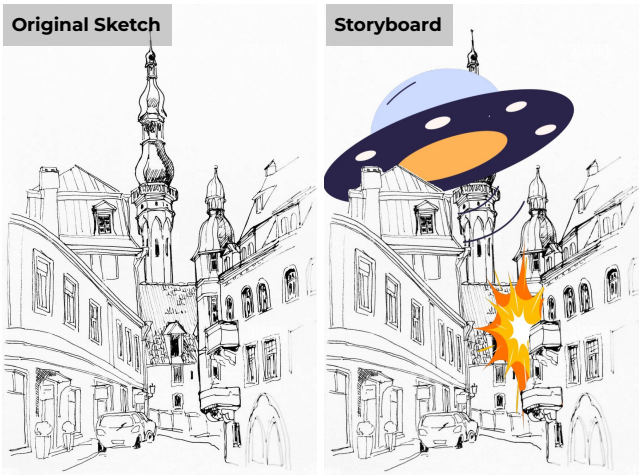


Figure 16: DepthScape also works on sketched scenes, helping users quickly story board their ideas by mixing visual assets with varying fidelity.

minimal 3D expertise. DepthScape leverages an AI agent that orchestrates RGB–and–3D pipeline to analyze the input image and synthesize 3D geometry extraction code which, when compiled and executed, can generate semantic-aware parametric anchors that significantly simplify 2.5D editing and exploration. We evaluated DepthScape through a multi-pronged study: a user study with nine participants, technical performance on 100 stock images, expert validation with three professional designers, and exploratory demonstrations with five application scenarios.

Compared to existing editing systems, DepthScape occupies a distinctive middle ground in the creativity support landscape. Our evaluation revealed its intuitive usability and flexible editability. Designers can quickly blend in 2D assets within a pseudo-3D space while retaining the ability to edit with precision—bridging the gap between playful exploration and controlled refinement. On the contrary, when creating 2.5D visual designs, traditional 2D editors (*e.g.,* Photoshop) demand tedious adjustments and strong spatial intuition. 3D software (*e.g.,* Blender) automates perspective rendering but imposes steep learning curves and technical overhead. Generative editing tools (e.g., diffusion-based methods such as "nano banana"[1]) provide image editing capabilities but lack control and interactivity. Generative AI methods also alter the entire original image, making them unsuitable when pixel-level fidelity must be preserved. DepthScape bridges these extremes by offering real-time direct-manipulation of 2D content in a pseudo-3D interface, retaining the quality of the original image while fostering playful exploration and serendipity. In this way, DepthScape complements both established editing pipelines and emerging generative tools, providing an interactive, controllable, and exploratory pathway for depth-rich design.

[1]https://aistudio.google.com/models/gemini-2-5-flash-image

7.1 Semantics and Geometry Extraction

Currently, DepthScape's geometry extraction for parametric anchors primarily handles single objects and simple geometric types. A promising direction for future work is to enable more complex forms of composition. For instance, combining the front-facing direction of a human figure with the normal of a ground plane could yield richer planar anchors. Our current visual program format supports such multi-element interactions, but realizing them will require additional example data. Future deployments can collect this data and expand the repertoire of supported geometries.

Our current system supports planes, cylinders, and spheres as anchor primitives, informed by common 2.5D design patterns observed in our design research and findings from our formative study. Nevertheless, certain creation needs fall outside this taxonomy, such as decal effects that place content on irregular surfaces (e.g., cloth), more complex geometries (e.g., cones or tori), or free-floating content not associated with any object. While our interface currently allows users to manually adjust placement of primitives via parameter controls, extending support to richer geometry types is an important direction for future work.

7.2 Advanced Blending Effects

DepthScape uses depth-based blending to enable realistic occlusion and perspective cues, but expert feedback points to opportunities in richer physical effects. Shadows, reflections on water or glass, color bleeding, fog, depth-of-field blur, and motion blur could further blend the inserted content into the scene. We plan to enable these effects in future iterations by estimating additional environmental cues and upgrading the rendering pipeline.

7.3 Limitations

Despite these contributions and future directions, several limitations remain in this paper.

Evaluation design. We did not conduct a direct comparison study against industry tools such as Photoshop or Blender. DepthScape is currently tailored to 2.5D effects rather than the broader image/3D editing capabilities of these established platforms. At the same time, existing 2D, 3D, and generative approaches support different workflows and affordances, making it difficult to design fair and meaningful cross-platform tasks for comparative evaluation. Future work could develop task-specific benchmarks that highlight complementary strengths and situate DepthScape more clearly within the ecosystem of design tools.

System performance. Processing of a single image currently takes approximately 20 seconds, with the majority of the time spent on waiting for VLM responses while other processing steps remain efficient. We expect processing times to decrease as newer and faster VLMs become available. Future work could also explore using smaller, task-specific VLMs and parallel execution of visual programs to further reduce latency.

User interaction. While our current design provides VLM-selected parametric anchors and intuitive direct manipulation editing, expert users desired finer controls—such as explicitly selecting target objects, specifying anchor types, or manipulating more sophisticated geometries (e.g., splines). We expect to extend the interaction options to incorporate such suggestions and better facilitate

professional use. We also plan to expose additional intermediate AI steps and reasoning underlying the automatically generated parametric anchors, including image masks, geometry extraction processes, and VLM reasoning. This can help users better understand how parametric anchors are derived, identify potential errors in intermediate steps, and adjust results based on their preferences.

8 Conclusion

Leveraging recent advancements in depth estimation and vision-language models, we present DepthScape, a novel graphic authoring tool for 2.5D design. DepthScape uses depth estimation to reconstruct a 3D scene from 2D input images, enabling depth-based blending that creates realistic occlusion and perspective effects by placing and rendering visual elements within this 3D space. By incorporating vision-language models for the semantic understanding of the scene, the system extracts parametric anchors from the input images, allowing users to place content through intuitive direct manipulation and fine-grained parameter editing. We test the feasibility and usability of the system with a user study among both amateur and professional designers. The study results confirm the creativity benefits of DepthScape and guide the implementation of the final DepthScape interface. We further conduct a technical evaluation on 100 professional stock images, demonstrating efficiency, robustness, and versatility. Additionally, we validate the quality of DepthScape's outputs through an expert evaluation and five real-world application scenarios.

Acknowledgments

This research was initially conducted during an internship at Adobe Research and further supported by a gift funding from Adobe Research. We thank Dr. Yu Shen and Dr. Stefano Petrangeli for their contribution to the early stages of this research.

References

[1] Giant Bomb. n.d.. Fonz. https://www.giantbomb.com/fonz/3030-36495/ Accessed: 2024-09-11.
[2] Giant Bomb. n.d.. Parallax Scrolling. https://www.giantbomb.com/parallax-scrolling/3015-2915/ Accessed: 2024-09-11.
[3] Giant Bomb. n.d.. Pole Position. https://www.giantbomb.com/pole-position/3030-10087/ Accessed: 2024-09-11.
[4] Giant Bomb. n.d.. Sprite Scaling. https://www.giantbomb.com/sprite-scaling/3015-7122/ Accessed: 2024-09-11.
[5] Giant Bomb. n.d.. Turbo. https://www.giantbomb.com/turbo/3030-18177/ Accessed: 2024-09-11.
[6] Eli Brenner and Jeroen B Smeets. 2018. Depth perception. In *Stevens' handbook of experimental psychology and cognitive neuroscience: Sensation, perception, and attention*. Wiley, 385–414.
[7] Joao Coutinho, Bruno AD Marques, and Joao Paulo Gois. 2016. Puppeteering 2.5 D models. In *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 1–8.
[8] Julie Dorsey, Songhua Xu, Gabe Smedresman, Holly Rushmeier, and Leonard McMillan. 2007. The mental canvas: A tool for conceptual architectural design and analysis. In *15th Pacific Conference on Computer Graphics and Applications (PG'07)*. IEEE, 201–210.
[9] Stefan P. Feyer, Bruno Pinaud, Stephen Kobourov, Nicolas Brich, Michael Krone, Andreas Kerren, Michael Behrisch, Falk Schreiber, and Karsten Klein. 2024. 2D, 2.5D, or 3D? An Exploratory Study on Multilayer Network Visualisations in Virtual Reality. *IEEE Transactions on Visualization and Computer Graphics* 30, 1 (2024), 469–479. https://doi.org/10.1109/TVCG.2023.3327402
[10] Martin A Fischler and Robert C Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (1981), 381–395.
[11] Tsukasa Fukusato and Akinobu Maejima. 2021. View-dependent formulation of 2.5 d cartoon models. *arXiv preprint arXiv:2103.15472* (2021).
[12] Joao Paulo Gois, Bruno AD Marques, and Harlen Costa Batagelo. 2015. Interactive shading of 2.5 D models.. In *Graphics Interface*. 89–96.
[13] Tanmay Gupta and Aniruddha Kembhavi. 2023. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14953–14962.
[14] Rorik Henrikson, Bruno De Araujo, Fanny Chevalier, Karan Singh, and Ravin Balakrishnan. 2016. Storeoboard: Sketching stereoscopic storyboards. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 4587–4598.
[15] Ian P Howard and BJ Rogers. 2002. Depth perception. *Stevens' handbook of experimental psychology Sensation and Perception* 1 (2002), 77–120.
[16] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).
[17] Ruth E Iskin. 2014. *The Poster: Art, Advertising, Design, and Collecting, 1860sÐ1900s*. Dartmouth College Press.
[18] S Kyriakaki-Grammatikaki, EK Stathopoulou, E Grilli, F Remondino, and A Georgopoulos. 2022. Geometric primitive extraction from semantically enriched point clouds. In *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLVI-2/W1-2022*. Vol. 46. 291–298.
[19] Sean J Liu, Maneesh Agrawala, Stephen DiVerdi, and Aaron Hertzmann. 2022. ZoomShop: Depth-Aware Editing of Photographic Composition. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 57–70.
[20] Shufang Lu, Wei Jiang, Xuefeng Ding, Craig S Kaplan, Xiaogang Jin, Fei Gao, and Jiazhou Chen. 2019. Depth-aware image vectorization and editing. *The Visual Computer* 35 (2019), 1027–1039.
[21] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Yong, Juhyun Lee, et al. 2019. Mediapipe: A framework for perceiving and processing reality. In *Third workshop on computer vision for AR/VR at IEEE computer vision and pattern recognition (CVPR)*, Vol. 2019.
[22] Leo Mariga. [n. d.]. pyRANSAC-3D. https://leomariga.github.io/pyRANSAC-3D/. Accessed: 2025-04-03.
[23] Philip B. Meggs. 2025. Graphic Design. https://www.britannica.com/art/graphic-design. Encyclopedia Britannica.
[24] Mostafa Mehrabi, Edward Peek, Burkhard Wuensche, and Christof Lutteroth. 2013. Making 3D work: a classification of visual depth cues, 3D display technologies and their applications. *AUIC2013* (2013).
[25] John Murray. 1994. Some perspectives on visual depth perception. *ACM SIGGRAPH Computer Graphics* 28, 2 (1994), 155–157.
[26] Jay Prakash Narayan. 2005. Study of basin-edge effects on the ground motion characteristics using 2.5-D modelling. *Pure and Applied Geophysics* 162, 2 (2005), 273–289.
[27] Suraj Patni, Aradhye Agarwal, and Chetan Arora. 2024. ECoDepth: Effective Conditioning of Diffusion Models for Monocular Depth Estimation. *ArXiv* abs/2403.18807 (2024). https://api.semanticscholar.org/CorpusID:268723777
[28] Pexels. 2025. Pexels – Free Stock Photos and Videos. https://www.pexels.com Accessed: 2025-04-06.
[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. arXiv:2103.00020 [cs.CV] https://arxiv.org/abs/2103.00020
[30] 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. 2024. SAM 2: Segment Anything in Images and Videos. arXiv:2408.00714 [cs.CV] https://arxiv.org/abs/2408.00714
[31] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. 2024. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159* (2024).
[32] Xia Su, Cuong Nguyen, Matheus A Gadelha, Yu Shen, Stefano Petrangeli, and Jon E. Froehlich. 2025. Authoring 2.5D Designs with Depth Estimation. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. Association for Computing Machinery, New York, NY, USA, Article 88, 7 pages. https://doi.org/10.1145/3706599.3719727
[33] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. 2024. MoGe: Unlocking Accurate Monocular Geometry Estimation for Open-Domain Images with Optimal Training Supervision. arXiv:2410.19115 [cs.CV] https://arxiv.org/abs/2410.19115
[34] Zhengyi Wang, Yikai Wang, Yifei Chen, Chendong Xiang, Shuo Chen, Dajiang Yu, Chongxuan Li, Hang Su, and Jun Zhu. 2024. Crm: Single image to 3d textured mesh with convolutional reconstruction model. *arXiv preprint arXiv:2403.05034* (2024).
[35] Lei Wu, Jingjing Wang, Mingyong Pang, and Shujuan Yu. 2015. Research on 2.5 D Graphics Applications in Teaching. In *2015 International Symposium on Educational Technology (ISET)*. IEEE, 75–78.

[36] Shaobo Xia, Dong Chen, Ruisheng Wang, Jonathan Li, and Xinchang Zhang. 2020. Geometric Primitives in LiDAR Point Clouds: A Review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 13 (2020), 685–707. https://doi.org/10.1109/JSTARS.2020.2969119

[37] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. 2024. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191* (2024).

[38] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. 2024. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arXiv preprint arXiv:2403.14621* (2024).

[39] Yusheng Xu, Sebastian Tuttas, Ludwig Hoegner, and Uwe Stilla. 2017. Geometric Primitive Extraction From Point Clouds of Construction Sites Using VGS. *IEEE Geoscience and Remote Sensing Letters* 14, 3 (2017), 424–428. https://doi.org/10.1109/LGRS.2017.2647816

[40] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. 2024. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10371–10381.

[41] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. 2024. Depth Anything V2. *arXiv preprint arXiv:2406.09414* (2024).

[42] Chih-Kuo Yeh, Peng Song, Peng-Yen Lin, Chi-Wing Fu, Chao-Hung Lin, and Tong-Yee Lee. 2013. Double-Sided 2.5D Graphics. *IEEE Transactions on Visualization and Computer Graphics* 19, 2 (2013), 225–235. https://doi.org/10.1109/TVCG.2012.116

[43] Emilie Yu, Kevin Blackburn-Matzen, Cuong Nguyen, Oliver Wang, Rubaiat Habib Kazi, and Adrien Bousseau. 2023. VideoDoodles: Hand-Drawn Animations on Videos with Scene-Aware Canvases. *ACM Trans. Graph.* 42, 4, Article 54 (jul 2023), 12 pages. https://doi.org/10.1145/3592413

[44] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024. Vision-language models for vision tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[45] Ruowen Zhao, Zhengyi Wang, Yikai Wang, Zihan Zhou, and Jun Zhu. 2024. FlexiDreamer: Single Image-to-3D Generation with FlexiCubes. *arXiv preprint arXiv:2404.00987* (2024).

[46] Tongyu Zhou, Joshua Kong Yang, Vivian Hsinyueh Chan, Ji Won Chung, and Jeff Huang. 2024. PortalInk: 2.5D Visual Storytelling with SVG Parallax and Waypoint Transitions. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)* (Pittsburgh, PA, USA). Association for Computing Machinery (ACM), New York, NY, USA. https://doi.org/10.1145/3654777.3676376

[47] Ruijie Zhu, Ziyang Song, Li Liu, Jianfeng He, Tianzhu Zhang, and Yongdong Zhang. 2024. HA-Bins: Hierarchical Adaptive Bins for Robust Monocular Depth Estimation Across Multiple Datasets. *IEEE Transactions on Circuits and Systems for Video Technology* 34 (2024), 4354–4366. https://api.semanticscholar.org/CorpusID:265359133

A Example Visual Programs

Example1:

The image depicts a draped figure covered entirely in a dark, flowing fabric, giving it an enigmatic and futuristic appearance. A circular digital text overlay with the words "4TH.CAPTERS" wraps around the figure, suggesting a technological or cyberpunk theme.

Visual Program:

MASK_0=Text2Mask(prompt = "the human figure")

POINTCLOUD_0=Mask2Pointcloud(mask = MASK_0)

CYLINDER_0=Pointcloud2Cylinder(Pointcloud = POINTCLOUD_0, direction = NULL)

CYLINDRICAL_0=Cylindrical(cylinder = CYLINDER_0)

Example2:

The image is a promotional poster for the "Kinosmena 2016" International Short Film Festival. It features a creative and modern design with a slightly side-facing woman's face framed by geometric panels that face the same direction, adorned with intricate blue foliage.

Visual Program:

MASK_0=Text2Mask(prompt = "the human figure")

FACE_0=FaceExtraction(mask = MASK_0)

PLANAR=Planar(plane = FACE_0.frontal)

Example3:

The image is a promotional poster for a "Senior Day" basketball event at Fifth Third Arena. It emphasizes the theme "Senior Day" with large white letters positioned prominently on the court, perpendicular to the ground and also in parallel to the square court's edge.

Visual Program:

MASK_0=Text2Mask(prompt = "basketball playground")

POINTCLOUD_0=Mask2Pointcloud(mask = MASK_0)

PLANE_0=Pointcloud2PLANE(Pointcloud = POINTCLOUD_0)

PLANAR=Planar(plane = PLANE_0.extruded)

Example4:

The image is a poster for a "James Bond Symposium," prominently featuring a shiny gold bullet in a 3D perspective as the central visual element. Bold white text reading "JAMES BOND" wraps around the bullet, creating a dynamic and eye-catching effect. Visual Program:

MASK_0=Text2Mask(prompt="the bullet")

POINTCLOUD_0=Mask2Pointcloud(mask=MASK_0)

CYLINDER_0=Pointcloud2Cylinder(Pointcloud=POINTCLOUD_0, direction=NULL)

CYLINDRICAL_0=Cylindrical(cylinder=CYLINDER_0)

Example5:

The image depicts a draped figure covered entirely in a dark, flowing fabric, giving it an enigmatic and futuristic appearance. A spherical digital text overlay with the words "Darkness" wraps around the figure, suggesting a technological or cyberpunk theme.

Visual Program:

MASK_0=Text2Mask(prompt = "the human figure")

POINTCLOUD_0=Mask2Pointcloud(mask = MASK_0)

SPHERE_0=Pointcloud2Sphere(Pointcloud = POINTCLOUD_0)

SPHERICAL_0=Spherical(sphere = SPHERE_0)

Example6:

The image features a vintage-inspired design with a sepia-toned aerial view of a long, straight street cutting through an urban landscape, surrounded by shadowy buildings. The title "Invisible Streets" is prominently displayed along the street in bold, staggered white and yellow text, enhancing the sense of depth and direction. Visual Program:

MASK_0=Text2Mask(prompt = "highrise bridge in the input image")
POINTCLOUD_0=Mask2Pointcloud(mask = MASK0)
PLANE_0=Pointcloud2Plane(Pointcloud = Pointcloud0)
PLANAR=Planar(plane = PLANE_0)

Example7:

The poster features a bold and energetic design, centered around a runner in motion. The background is white, with bright pink accents, including large, dynamic block text that forms an abstract, geometric pattern. This block text background is positioned to be in parallel to the running direction of the runner.

Visual Program:
MASK_0=Text2Mask(prompt = "runner")
SKELETON_0=SkeletonExtraction(mask=MASK_0)
PLANAR_0=Planar(plane = SKELETON_0.median)

Example8:

The image showcases a vibrant aerial view of a city at night, filled with illuminated skyscrapers, streets, and bustling urban life. Overlaid on the cityscape, bold white typography spells out the phrase "NO OTHER GAME," cutting across the buildings with a dramatic, immersive perspective.

Visual Program:
MASK_0=Text2Mask(prompt = "the front building in the input image")
POINTCLOUD_0=Mask2Pointcloud(mask = MASK_0)
PLANE_0=Pointcloud2PLANE(Pointcloud = POINTCLOUD_0)
PLANAR_0=Planar(PLANE = PLANE_0)

Example9:

The image features a relay race scene in a vibrant stadium setting, with one runner handing off the baton to another. The slogan "WE OFFER YOU THE BETTER, BETTER SOLUTION" is prominently displayed on the left side in bold, colorful text, placed in realistic perspective perpendicular to the ground but in parallel to the running direction of the runner.

Visual Program:
MASK0=Text2Mask(prompt = "ground")
Pointcloud0=Mask2Pointcloud(mask = MASK0)
MASK1=Text2Mask(prompt = "the runner in the middle")
SKELETON_0=SkeletonExtraction(mask = MASK1)
PLANE_0=Pointcloud2PLANE(Pointcloud = Pointcloud0)
PLANAR_0=Planar(plane = SKELETON_0.median)

Example10:

The image showcases two travelers walking towards an airport terminal. Both individuals are carrying luggage and backpacks, signaling they are either returning from or embarking on a journey. The text "BACK TO HOME" is prominently displayed in large white letters across the sky, with the perspective that looks like the text is parallel to the building facede.

Visual Program:
MASK_0=Text2Mask(prompt = "building in the image")
POINTCLOUD_0=Mask2Pointcloud(mask = MASK0)
PLANE_0=Pointcloud2Plane(Pointcloud = Pointcloud0)
PLANAR_0=Planar(plane = PLANE_0)

Example11:

The image is a conceptual and artistic design featuring a person with their face pixelated and obscured by bright yellow text reading "#31#." The repeated hashtag emphasizes anonymity and digital communication themes. Surrounding the figure are additional yellow text and graphic elements, including "Mélancolique Anonyme" (Melancholic Anonymous), phone numbers, and technical details like "Appel non taxé" (Non-taxed call), evoking the idea of an anonymous support line or mental health service.

Visual Program:
MASK_0=Text2Mask(prompt = "the human face")
FACE_0=FaceExtraction(mask = MASK_0)
POINTCLOUD_0=Mask2Pointcloud(mask = MASK_0)
SPHERE_0=Pointcloud2Sphere(Pointcloud = POINTCLOUD_0)
SPHERICAL_0=Spherical(sphere = SPHERE_0)

Example12:

The image is a minimalist and striking poster featuring a baseball player in mid-action, running against a solid bright blue background. Beneath the player, bold white text spells out "AUSTIN" in a stretched, perspective style, seemingly being placed on the ground and aligned with the facing direction of the player, adding depth and focus to the design.

Visual Program:
MASK0=Text2Mask(prompt ="floor")
Pointcloud0=Mask2Pointcloud(mask = MASK_0)
MASK1=Text2Mask(prompt = "the player")
SKELETON_0=SkeletonExtraction(mask = MASK_1)
PLANE_0=Pointcloud2PLANE(Pointcloud = POINTCLOUD_0)
PLANAR_0=Planar(plane = PLANE_0)

Example13:

The poster promotes a 3D indoor biking experience with a Parisian theme. The Eiffel Tower dominates the background, and the tagline "VENHA PEDALAR PELAS RUAS DE PARIS" ("Come Pedal Through the Streets of Paris") is displayed in bold red and green text, emphasizing the immersive aspect of the activity. The text is placed perpendicular to the ground, while also in parallel to the edge of the road.

Visual Program:
MASK0=Text2Mask(prompt = "driveway")
Pointcloud0=Mask2Pointcloud(mask = MASK_0)
PLANE_0=Pointcloud2PLANE(Pointcloud = POINTCLOUD_0)
PLANAR_0=Planar(plane = PLANE_0.extruded)

Example14:

The poster is a bold, surreal design featuring a white classical bust with a futuristic twist. In the middle of the poster is a human head, with the top of the head appears cut open, revealing a black void, while the eyes glow red, giving a cyberpunk aesthetic. Large white typography dominates the composition, reading "WE ARE BACK IN BLACK", surrounding the head as a cylinder-like shape.

Visual Program:
MASK_0=Text2Mask(prompt = "the human head")
FACE_0=FaceExtraction(mask = MASK_0)
POINTCLOUD_0=Mask2Pointcloud(mask = MASK_0)
CYLINDER_0=Pointcloud2Cylinder(Pointcloud = POINTCLOUD_0, direction = FACE_0.cranial)

CYLINDRICAL_0=Cylindrical(cylinder = CYLINDER_0)

Example15:

The poster promotes a vocal training school for extreme vocal techniques, emphasizing styles like "Pig Voice," "Guttural," "Growling," and "Screaming," displayed in dynamic, radiating white text. The main visual element is a stylized blue-toned portrait of a person screaming, with lines and text emphasizing the sound's intensity and energy coming from her mouth.

Visual Program:

MASK_0=Text2Mask(prompt = "the human figure")

FACE_0=FaceExtraction(mask = MASK_0)

PLANAR=Planar(plane = FACE_0.median)

Example16:

The poster features a bold and energetic design, centered around a runner in motion. The background is white, with bright pink accents, including a large, dynamic block text that surrounds the runner. Since there isn't any visible ground surface in the poster, the cylinder of text is formed based on the runner figure's shape.

Visual Program:

MASK_0=Text2Mask(prompt = "runner")

POINTCLOUD_0=Mask2Pointcloud(mask = MASK_0)

CYLINDER_0=Pointcloud2Cylinder(Pointcloud = POINTCLOUD_0, direction = NULL)

CYLINDRICAL_0=Cylindrical(cylinder = CYLINDER_0)