EUROGRAPHICS 2022 / R. Chaine and M. H. Kim
(Guest Editors)

Volume 41 (2022), Number 2

ZoomShop: Depth-Aware Editing of Photographic Composition

Sean J. Liu[1], Maneesh Agrawala[1], Stephen DiVerdi[2] and Aaron Hertzmann[2]

[1]Stanford University
[2]Adobe Research

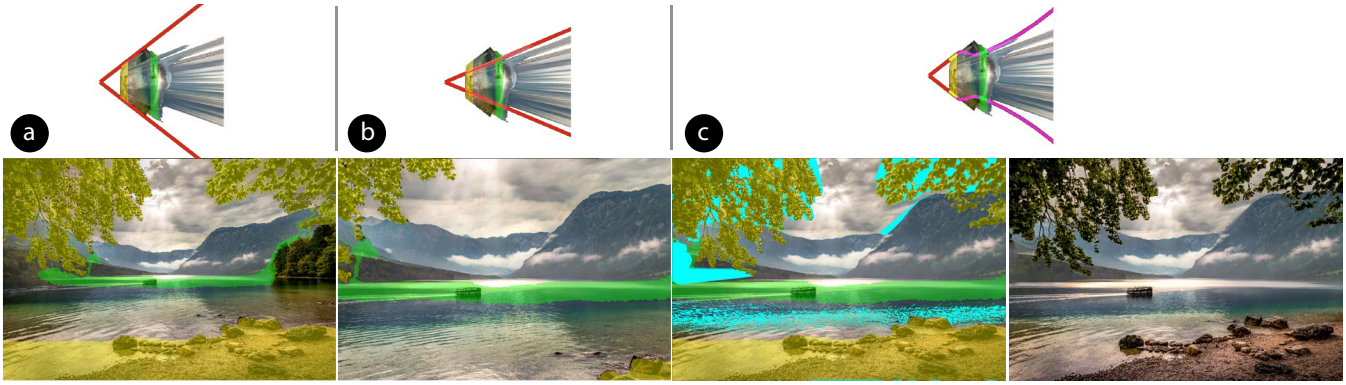


Figure 1: *Using ZoomShop to edit a photograph [Tha16]. (a) An image of mountains (background) framed by trees (foreground). The user's goal is to make the boat bigger while keeping the framing of foreground trees. (b) Zooming in and cropping scales up the boat, but cuts out most of the trees and shore. (c) Left: With ZoomShop users can select depth ranges (yellow and green) and independently adjust each region while maintaining scene structure. Disoccluded and stretched pixels are shown in cyan. Right: Cyan pixels from image are manually inpainted using Photoshop's Content-Aware Fill. We show the top-down view volume and boundary curve of each camera above the corresponding images in a, b, and c.*

Abstract

We present ZoomShop, a photographic composition editing tool for adjusting relative size, position, and foreshortening of scene elements. Given an image and corresponding depth map as input, ZoomShop combines a novel non-linear camera model and a depth-aware image warp to reproject and deform the image. Users can isolate objects by selecting depth ranges and adjust their scale and foreshortening, which controls the paths of the camera rays through the scene. Users can also select 2D image regions and translate them, which determines the objective function in the image warp optimization. We demonstrate that ZoomShop can be used to achieve useful compositional goals, such as making a distant object more prominent while preserving foreground scenery, or making objects both larger and closer together so they still fit in the frame.

CCS Concepts

• ***Computing methodologies*** → ***Graphics systems and interfaces; Image manipulation;***

1. Introduction

An important task in photographic composition is to adjust relative object sizes and positions. Consider Figure 1a, a photograph of a boat on a lake framed by trees. The photographer may wish to make the boat appear larger, which can be achieved by zooming, but the trees and the shore go out of frame (Fig. 1b). A longer lens from further back might satisfy both goals, but may be impossible due to physical constraints. With varying degrees of manual effort, today's digital tools allow adjusting the sizes of scene objects in a plausible, if not geometrically accurate, way [Mat21]. There are many methods that can arbitrarily adjust object sizes [AS07; CAA09; SLL19; BSFG09; SDM19], but these methods can fail to preserve important spatial relationships because they do not incorporate 3D scene understanding.

We present ZoomShop, a digital image editing tool that uses knowledge of the image's 3D geometry to provide the specific

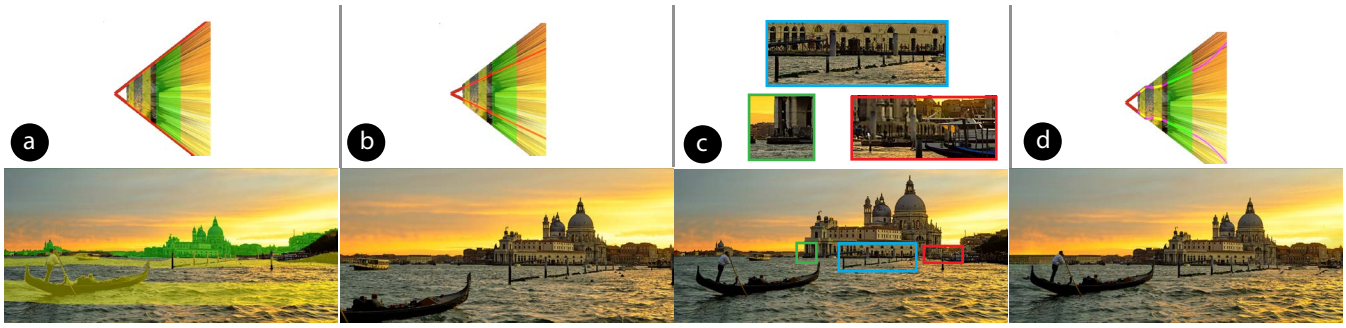


Figure 2: *Venice [Sze14b]. Goal: scale up distant building while keeping the boat in the same place. (a) Original photo. (b) Zooming in and cropping. (c) Cutting out the building, scaling it, and pasting it back in may not preserve depth structure due to lack of scene depth understanding. Here, the building occludes side buildings and boats, and the poles in front are cut in half (marked with boxes). (d) Our result with inpainting (manual guidance).*

controls needed to edit photographic composition while maintaining important image structures (Fig. 1c). With ZoomShop, users can select image regions based on depth, adjust relative scaling of each region, adjust foreshortening within each region, and translate objects horizontally within regions. These controls allow users to make high-level adjustments to the 3D structure of the image: they can enlarge distant objects, maintain foreground framing, adjust the foreshortening of an object to emphasize its 3D shape, and adjust the relative sizes and positions of objects at the same depth.

Our key technical contribution is to support user editing goals with a combination of a non-linear camera model and a depth-aware rubber sheet warp. Our novel camera model consists of a depth-varying scale function that is defined by a piecewise linear, smooth, and/or discontinuous curve. Given an input image and its depth map (either measured or estimated), ZoomShop maps the user's edits to our non-linear camera model and then reprojects the scene. Then the user can select regions and translate them horizontally. These translations define the objective function of our depth-aware rubber sheet warp, which is used to warp the image for the final result. The ZoomShop interface enables the user to select where and how deviations from linear perspective occur to support their size, position, and foreshortening goals, providing complete control over the final image appearance to the user.

We demonstrate ZoomShop's capabilities on examples, adjusting scale and foreshortening while maintaining framing, and show how it differs from other linear and non-linear imaging models. We use off-the-shelf computer vision algorithms to infer scene depth [RLH*20] and fill holes [Ado21; BSFG09]; we expect even better methods to be available for these subtasks in the near future, given the fast pace of progress in these fields. While we focus on outdoor scenes, ZoomShop is potentially useful for any scene exhibiting a large range of depths. We believe ZoomShop provides a useful new way to approach photography.

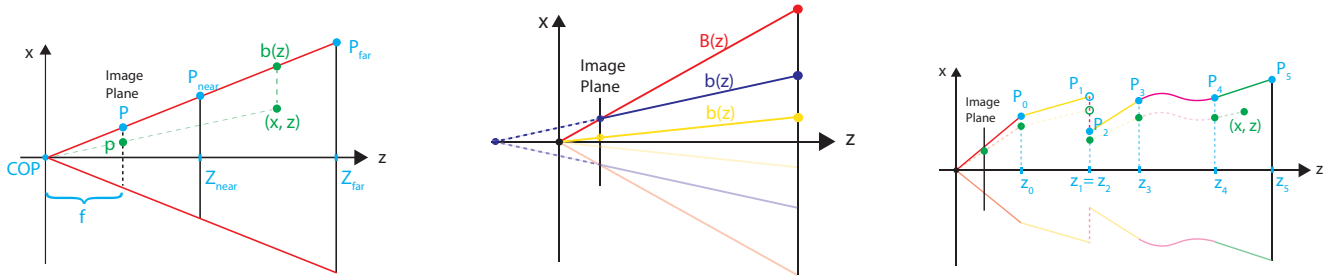
2. Related Work

Many image retargeting techniques do not consider depth. Cropping [SLBJ03] can make an object appear larger in the image but cuts out peripheral content. Seam carving [ADCS19; AS07] removes pixels as well but focuses on non-salient ones along an image dimension. Shift maps [PKP09], cut-and-paste [STR*05], and patch-based image editing methods [SCSI08; BSFG09; SDM19] support arbitrary reshuffling and allow new pixels to overlay existing pixels. While these image retargeting methods are powerful, they lack scene depth understanding and thus may not preserve occlusion relationships, depth structure, or provide direct control over adjusting perspective. One contribution of ZoomShop is to use depth information for warping, as depth information is becoming more readily available, either from consumer mobile phone cameras or estimated via state-of-the-art computer vision methods.

Several previous methods specifically aim to adjust scene perspective. Carroll et al. [CAA09; CAA10] introduced warping-based methods for manipulating the local perspective in an image, creating non-linear warps of a fixed camera projection. Fried et al. [FSGF16] show how to modify foreshortening in portraits by fitting a virtual perspective camera and then warping the image. Other transformation techniques [ZB95; SLL19; CAA09] minimize distortions in wide-angle images by combining different perspectives in different regions of the image. Other works aim to create realistic-looking panoramas through perspective deformation [AAC*06; ZPP05; SPG10]. These methods are all constrained to produce one-to-one mappings between the input and output images, which means they do not support changes in occlusion relationships between objects. This fundamentally limits their ability to deform scenes involving both foreground and background elements, a common aspect of photographs that ZoomShop addresses.

Our method is also related to work that reprojects a single image using depth estimation and conventional projection models [SSKH20; KMA*20]. Niklaus et al. [NMYL19] recreate "Ken Burns" effects by translating a linear perspective camera through a photograph's 3D scene. These methods all utilize conventional linear perspective camera models, which limits their results to only geometrically accurate reprojections. ZoomShop on the other hand uses a non-linear camera model and warp, enabling a wider class of image transforms that better supports photographers compositional goals, such as enlarging a distant object while keeping foreground elements fixed.

Most closely related to ZoomShop is the work of Badki et



(a) *Linear Perspective. P is the half-width of the image plane in world space. Z_{near} and Z_{far} are the z-values of the near and far plane, and f is the focal length of the camera.*

(b) *The original boundary $B(z)$ is in red. Scaling is equivalent to zooming in and cropping (yellow). Decreasing the foreshortening of some depth range is equivalent to using a longer focal length and moving the camera back (blue).*

(c) *$b(z)$ with curved, linear, continuous, and discontinuous segments. To support discontinuous $b(z)$ curves, our tool allows $z_i = z_{i+1}$, $P_i \neq P_{i+1}$. In such cases, our tool sets $b(z_i = z_{i+1}) = P_{i+1}$.*

Figure 3: *Our non-linear camera model is defined by its boundary curve $b(z)$.*

al. [BGKS17], which combines multiple photographs to reproject the scene with a piecewise linear camera. We extend their camera model by including linear, curved, and discontinuous segments in our camera model, which support different types of edits and scenes. ZoomShop also uses a simpler workflow, relying on a single image with a measured or estimated depth map, which enables editing a broader range of images, such as dynamic scenes or historical events.

Other techniques have been proposed for combining perspectives from multiple cameras to create a single non-linear artistic projection [AZM00; Fou08; Sin02; CH03; ZP07; CS04; YM04a], or by non-linear ray tracing [BSCS07; BCS*09; LG96; YM04b]. General purpose non-linear camera models are very powerful and can produce a wide range of image appearances. Our camera model is a subset of these general models, specifically designed for ease of editing and maintaining scene spatial relationships. We also use a depth-aware rubber sheet warp to support scene element translation. Our results could be created by a more general non-linear camera model alone; our contribution is the alignment between ZoomShop's user controls and its underlying implementation.

Finally, our work is motivated in part by an intriguing observation from the perception literature: in some cases linear perspective photographs may not capture subjective visual experience as well as nonlinear images, particularly those that inflate the size of objects in or near the fovea [BBP14; BPR18; Rau82; KDPP16]. Allowing users to resize objects in images could be useful for better conveying scenes and describing visual experiences [Mat21].

3. Editing Photographic Composition

To motivate the design of ZoomShop, we first understand a hypothetical photographer's editing goals with the example in Figure 2. Upon taking the photo, the photographer realizes the distant building, the Santa Maria della Salute ("Salute"), appears smaller in the image than they would like and wishes to make it larger.

Goal 1. Make a target object appear larger or closer.

Zooming in and cropping scales the Salute but cuts out the foreground boat (Fig. 2b). As composition is important to balance the image, the photographer wishes to preserve it as in the original photo.

Goal 2. Maintain scene element visibility.

Cutting out the Salute, resizing it, and pasting it back in (Fig. 2c) create artifacts where the docking poles in front are cut in half and nearby elements are occluded (depth ordering), and care must be taken to ensure the resized Salute contacts the unedited ground (depth continuity).

Goal 3. Maintain scene element spatial relationships.

To achieve these goals, we split the problem into two complementary sub-problems: editing objects at different depths, and editing objects at the same depth. To edit objects at different depths, we reproject the image with a non-linear camera model that enables depth-dependent scaling. The user can resize an object at a target depth (Goal 1), maintain the scale (and thus visibility) of objects at other depths (Goal 2), and ensure smooth transitions between target regions to maintain spatial relationships (Goal 3). Since the scale is per-depth, objects at the same depth are all scaled the same, which may push some objects out of view. To support maintaining visibility in this case, we apply a rubber sheet warp to the image to translate peripheral objects back into view (Goal 2). The warp is also depth aware, to ensure it maintains depth continuity (Goal 3).

3.1. Editing Objects at Different Depths

To edit objects at different depths, we present a novel non-linear camera model defined by a *boundary curve* $b(z)$ that supports depth-dependent scaling.

Figure 3a shows a top-down view of a camera frustum, with the camera at the origin. Let f be the camera focal length, (P_w, P_h) the half-width and half-height of the image plane, and Z_{near} and Z_{far} the camera near and far planes. Linear perspective maps 3D scene

points $p = (x, y, z)$ to image coordinates $u, v \in [-1, 1]$ by

$$u = \frac{f}{P_w z} x, \; v = \frac{f}{P_h z} y \tag{1}$$

We allow $b(z)$ to be any function over $z \in [Z_{near}, Z_{far}]$. Given $b(z)$ and the image aspect ratio $\lambda = P_h / P_w$, the mapping from a scene point to image coordinates is

$$u = \frac{x}{b(z)}, v = \frac{y}{\lambda b(z)} \tag{2}$$

$b(z)$ defines the x-boundary of the view volume (Figure 3a), while the y-boundary is $\lambda b(z)$. Only points within the x- and y-boundaries are included in the output image: $Z_{near} \leq z \leq Z_{far}$, $-b(z) \leq x \leq b(z)$, and $-\lambda b(z) \leq y \leq \lambda b(z)$. For all boundary curves, scenes are rendered in reverse depth order (back to front) to maintain occlusion relationships.

Note that the view volume boundary $b(z)$ at a given depth z is inversely related to the scale of scene points at z in the image (Equation 2). Therefore, we can manipulate $b(z)$ to change the size and foreshortening of objects in the output image.

To resize an object at $p = (x, y, z)$ by a scale factor s so $(u', v') = (su, sv)$, we set $b(z) = B(z)/s$, where $B(z)$ is the input photograph's original camera boundary curve, $B(z) = P_w z / f$. This is a linear function, so applying $b(z)$ uniformly scales all image points by s and is equivalent to zooming and cropping (Fig. 3b). To resize the foreground and background separately, we can define $b(z)$ as a non-linear function of depth.

We can also use $b(z)$ to adjust scene foreshortening (change in size over depth) by changing $db(z)/dz$ over some range (Fig. 3b). For example, decreasing the slope of $b(z)$ decreases foreshortening, compressing depth and making background objects appear closer. This is equivalent to increasing the camera's focal length while pulling the camera back (as in a dolly-zoom shot). With a non-linear $b(z)$, we can achieve this result without moving the camera, and we can also independently adjust the foreshortening of different depth ranges.

Our camera model supports image formation under Equation 2 for $b(z)$ composed of any number of segments that are straight, curved, or discontinuous (Fig. 3c). Straight segments yield local linear perspective, but the image regions where different straight segments abut may have visible bending artifacts. Curved segments can be used to create smooth transitions between other segments by maintaining tangent continuity of $b(z)$, but will not preserve straight lines. Discontinuities occur when two $b(z)$ segments abut at different depths, creating a jump in scale in the output image. Discontinuities are useful when image features (e.g. texture or occlusions) allow hiding the transition. Choosing the correct set of segments depends heavily on the scene and desired results, so ZoomShop presents these options to the user.

3.2. Editing Objects at the Same Depth

Our camera model scales all points at the same depth uniformly. While this helps maintain spatial relationships between nearby objects, it can also move peripheral objects out of the image. For example, Figure 4 has two hills at the same depth on each side. Scaling them up pushes the hills out of view (i.e., no longer contained by $b(z)$). In order to maintain their new size *and* keep them visible, ZoomShop includes an image warp optimization to translate objects back into view. Our optimization is depth-aware, so pixels that have similar depths are translated by similar amounts, while pixels at different depths can be translated differently.

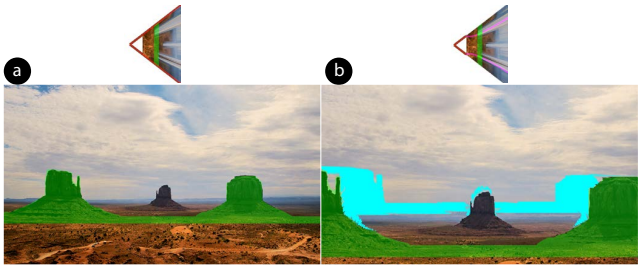


Figure 4: *Monument Valley [Sze14a]. ZoomShop scales scene points at the same depth by the same amount. (a) Original photo. (b) Side hills (green) are scaled up but are moved partially out of view; disoccluded pixels are shown in cyan. To fix this, ZoomShop includes a depth-aware image warp to translate the side hills back into view.*

Given user-defined regions $R_1, R_2, \ldots R_K$ and a corresponding desired translation $T^1, T^2, \ldots T^K$ for each one, we solve for a per-pixel translation map $t_{i,j}$ for each pixel i, j. To reduce the search space of $t_{i,j}$, we limit the search to horizontal translations only. This is a reasonable limitation because objects that are attached to the ground should remain on the ground after translation, but it is straightforward to extend to 2D translations if needed. For a pixel at (u_i, v_j), the new location is $(u_i + t_{i,j}, v_j)$.

Our goal is to smoothly propagate translations across continuous depths. Thus, we minimize the following objective function:

$$\min_t \lambda_s E_{\text{smooth}} + E_{\text{reg}} \tag{3}$$

$$\text{subject to } t_{i,j} = T^k, \; \forall i, j \in R_k, \; k = 1 \ldots K \tag{4}$$

where the input region translations are hard constraints and λ_s adjusts how quickly translations fall off. We used $\lambda_s = 10^5$ in all of our results.

The smoothness term encourages $t_{i,j}$ to vary smoothly across pixels at the same depth:

$$E_{\text{smooth}} = \frac{1}{N} \sum_{i,j} [w^v_{i,j}(t_{i,j+1} - t_{i,j})]^2 + [w^h_{i,j}(t_{i+1,j} - t_{i,j})]^2 \tag{5}$$

where $N = 2WH - W - H$ is the total number of pairs of neighboring pixels for image width W and height H.

The smoothness weights are reduced at depth discontinuities:

$$w^v_{i,j} = \alpha_v \sigma(|z_{i,j+1} - z_{i,j}|) = \alpha_v \frac{1}{1 + e^{\beta(|z_{i,j+1} - z_{i,j}| - \gamma)}} \tag{6}$$

$$w^h_{i,j} = \alpha_h \sigma(|z_{i+1,j} - z_{i,j}|) = \alpha_h \frac{1}{1 + e^{\beta(|z_{i+1,j} - z_{i,j}| - \gamma)}} \tag{7}$$

α_v and α_h adjust the relative penalty between vertically and horizontally adjacent pixels. To avoid shearing of objects resting on the ground, we penalize translation differences across vertical pixels

more than horizontal pixels. We use $\alpha_v = 5, \alpha_h = 1$ for all results. The sigmoid function σ selects the range of depth differences where the energy term is nonzero. We empirically found that $\beta = 10^4$, $\gamma = 10^{-4}$ works well.

We additionally include a regularization term:

$$E_{\text{reg}} = \frac{1}{WH} \sum_{i,j} t_{i,j}^2 \tag{8}$$

This encourages the optimization to find the smallest deformation that satisfies the user's input.

4. The ZoomShop Application

ZoomShop implements our non-linear camera model and depth-aware rubber sheet warp as the basis for editing an image. The full workflow requires ingesting input images, representing boundary curves, the user interface, and rendering the final result.

4.1. Input Geometry

We begin with an input RGB image, and generate a non-metric disparity map from MiDaS [RLH*20; SSKH20]. We manually clean up any obvious errors using Adobe Photoshop 2021 [Ado21], convert it to a non-metric depth map, and then use it as the depth of a per-pixel triangle mesh as our 3D scene. For each pixel disparity $d \in [0,1]$, we compute depth $z(d) = \frac{1}{d+0.1}$, and we set the input camera's vertical field of view to $\theta_v = 55^\circ$. While this does not produce a geometrically accurate scene, it preserves relative depth ordering and is sufficient for our needs.

4.2. Boundary Curve Representation

The original boundary curve of the image is $B(z) = \frac{P_w z}{f}$ and the new boundary curve after user edits is $b(z)$. We represent $b(z)$ as a series of control points (z_i, P_i), $i = 1...N$ (Fig. 3c). $z_{1:N}$ is a list of depths in increasing order, and $P_{1:N}$ are the boundary positions at those depths. These control points are interpolated with linear and cubic segments to form the $b(z)$ curve. Discontinuities are supported by consecutive control points sharing the same z value. Initially, there are only two control points $z_{1:2} = \{Z_{near}, Z_{far}\}$ and $P_{1:2} = \{B(Z_{near}), B(Z_{far})\}$ that define the original boundary. As the user edits the image, ZoomShop updates the control points and reprojects the image in real-time.

4.3. User Controls

ZoomShop presents a set of controls to edit the image appearance, which are used to construct the boundary curve and rubber sheet warp.

First, the user **selects a depth range** by clicking on a target object to edit. ZoomShop creates a new linear boundary curve segment at the target object depth with two control points z_i, z_{i+1}. All image pixels within the depth range are highlighted, and the user may refine the selection further by adjusting the start and end depths. The boundary positions for the segment are initialized to $P_i = b(z_i)$ and $P_{i+1} = b(z_{i+1})$.

After selecting an image region, the user may **adjust its scale**, which changes P_i, P_{i+1} while keeping z_i, z_{i+1} fixed. For a scale value s, $P_i' = B(z_i)/s$ and $P_{i+1}' = P_i' + (P_{i+1} - P_i)$, which preserves the slope of the linear segment.

The user may also **adjust the foreshortening** of the selected region, which controls how image size changes over depth. This is akin to changing the camera focal length, a commonly used photographic technique to compress or emphasize the depth of an image. The user adjusts the foreshortening of a region by changing the scale at the back P_{i+1} while fixing the scale at the front P_i, which changes the slope of the boundary curve segment. Fixing P_i ensures the image size does not change, allowing foreshortening and scale to be adjusted independently.

Each image region the user adjusts corresponds to a linear segment in the boundary curve. In between those segments, the user can **select the interpolants** that complete the curve. By default, ZoomShop uses a cubic interpolant to ensure the boundary curve has smooth tangents so there are no abrupt changes in scale in the output image, but this may result in straight lines in the image becoming curved. Alternately, the user can select linear interpolation, which yields a piecewise linear boundary curve that preserves straight lines within regions, but may have visible bending artifacts at region transitions.

After editing an image, the user may find that some scaled objects have moved to undesirable locations (e.g. out of the image), so they can **translate the objects** to better positions. The user selects one or more 2D rectangles and moves them horizontally to the target locations, which become the hard constraints of our depth-aware rubber sheet warp.

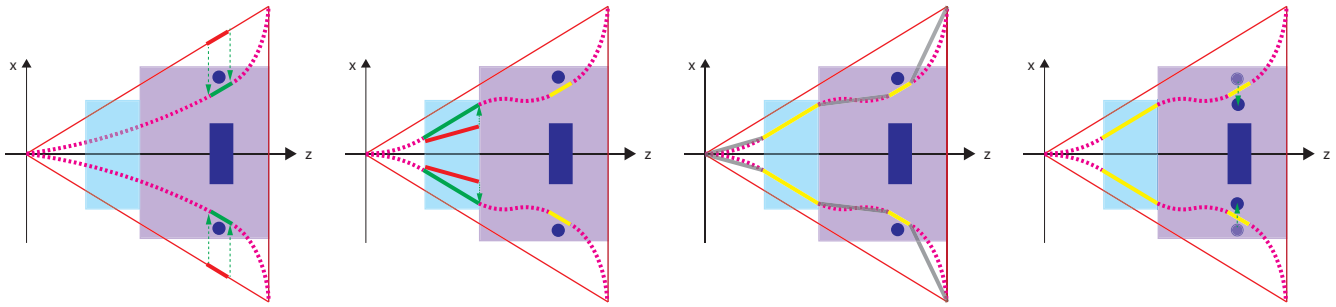
4.4. Translation Map

ZoomShop generates the per-pixel translation map from the user's edits by optimizing the constrained objective of the depth-aware rubber sheet warp. To save computation time, ZoomShop scales the image down to the same size as its depth map, computes a per-pixel translation map on the scaled-down image, and then upsamples the output translation map back to the original image size. ZoomShop uses PyTorch's L-BFGS optimizer, which can take 2–10 minutes to complete, depending on the size of the depth map.

The optimization relies on computing the change in depth for each pixel among its neighborhood, which requires depth values to be available at every pixel. However after reprojecting the image using our camera model, there are disoccluded regions that have no depth information. Therefore, while translation rectangles are visualized on both the input and scaled images, users mark and adjust the rectangles on the input image, and the translation map is also computed on the original image.

4.5. Image Synthesis

To synthesize the final image, ZoomShop first applies the translation map (if any) by displacing the input mesh vertices. Then ZoomShop applies a per-depth scale factor to the scene based on the new $b(z)$. Each vertex at depth z is scaled about the image center by $B(z)/b(z)$. When rendering the output image, we identify and



(a) *Step 1: Enlarge the statue (blue rectangle). Selecting the depth range adds a linear segment (bold red). Increasing the scale of the selection moves the bold red segment to the bold green segment. The slope remains the same, which preserves the foreshortening of the statue.*

(b) *Step 2: Expand depth of the steps (light blue rectangle). Selecting the depth range adds another another linear segment (bold red). Increasing the foreshortening changes the slope of the bold red segment to the bold green segment.*

(c) *Step 3: Toggle between smooth and linear interpolants. The smooth mode uses a cubic interpolant to connect the linear segments (dotted magenta). The linear mode uses linear segments (solid gray).*

(d) *Step 4: Translate pillars (blue circles) into view. Enlarging the statue (blue rectangle) caused the two pillars to move out of frame (i.e., no longer contained by the boundary curve). The image warp optimization moves them back into view.*

Figure 5: *Example user workflow. Top-down layout of a statue (blue rectangle) flanked by pillars (blue circles) on a platform (purple region), with steps (light blue) leading up to it.*

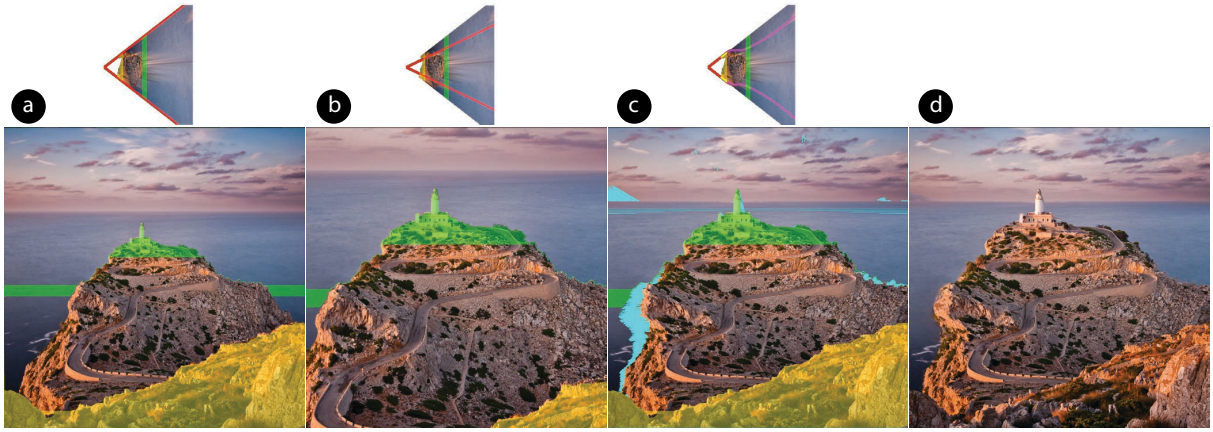


Figure 6: *Lighthouse [Miš11]. Goal: enlarge lighthouse while keeping compositional element of the foreground ridge. (a) Original photo. (b) Zooming in and cropping. (c) Our result. (d) Our result with inpainting (automatic).*

remove disoccluded, stretched, and sheared pixels; these are shown as cyan in our results.

To fill in the removed pixels, we use Photoshop's Content-Aware Fill (CAF) [Ado21; BSFG09]. For some results, the fully automatic CAF works well, but for other results, it was necessary to manually specify regions for CAF to sample from when inpainting. In those cases, we show both automatically and manually inpainted results side-by-side. The final image quality depends on the inpainting, which is an active area of research [LRS*18; YLY*19] that we expect to continue to improve rapidly.

4.6. Example Workflow

We give an example of a hypothetical user workflow in Figure 5, which shows a top-down illustrated layout of a statue (blue rectangle) flanked by pillars (blue circles) on a platform (purple region), with steps (light blue) leading up to it. Upon taking the photo, the user realizes the statue appears too small and wants to enlarge it. So they first select a depth range that contains the statue and increase its scale (Fig. 5a). After scaling the statue, the user realizes the foreground steps appear too short and wants to expand their depth. So they select a depth range containing the steps and increase its foreshortening (Fig. 5b). After modifying these two depth ranges, the user toggles between the smooth and linear interpolants (Fig. 5c), and finds that linear interpolation creates a noticeable bend in between the steps and the platform, while the platform does not contain visually important straight lines. Therefore, they select smooth interpolation. Finally, the user finds that the pillars next to the statue have been moved out of the image, so they select the pillars and translate them back into view (Fig. 5d).

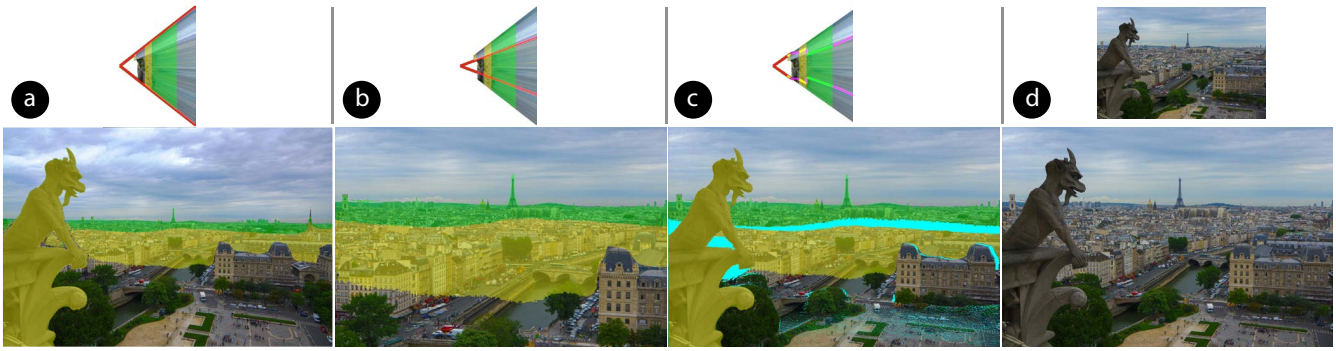


Figure 7: *Eiffel Tower [Mar12]. Goal: enlarge the Eiffel Tower while keeping the Gargoyle visible. (a) Original photo. (b) Zooming in and cropping. (c) Our result. (d) Our result with inpainting (top: automatic; bottom: manual guidance). We break the scene up into three depth ranges; the gargoyle, the intermediate buildings, and the Eiffel Tower. The intermediate depth range maintains linear perspective for the buildings, and we place the discontinuity between the far two depth ranges in a highly textured region for easier inpainting.*

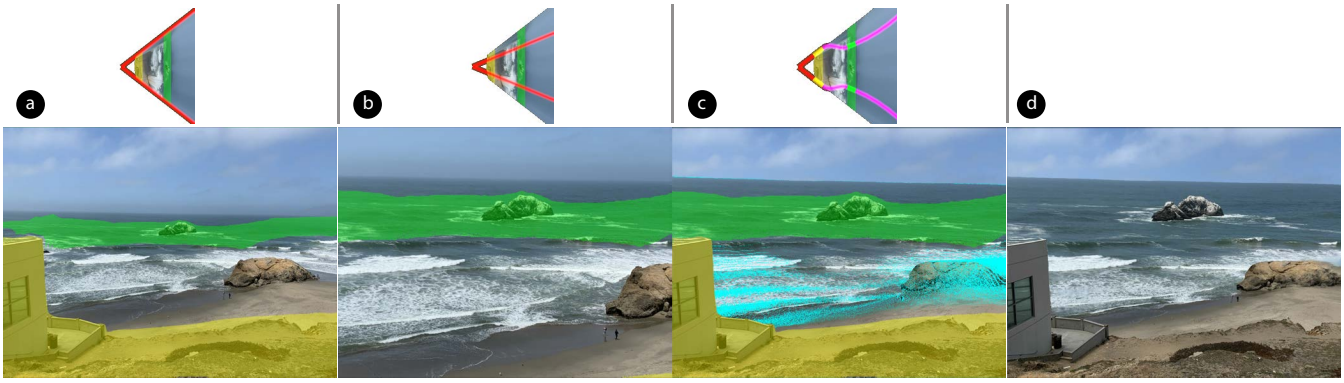


Figure 8: *Seal Rocks. Goal: enlarge Seal Rocks (background) while keeping Cliff House (foreground, left) in view. (a) Original photo. (b) Zooming in and cropping. (c) Our result. (d) Our result with inpainting (automatic).*

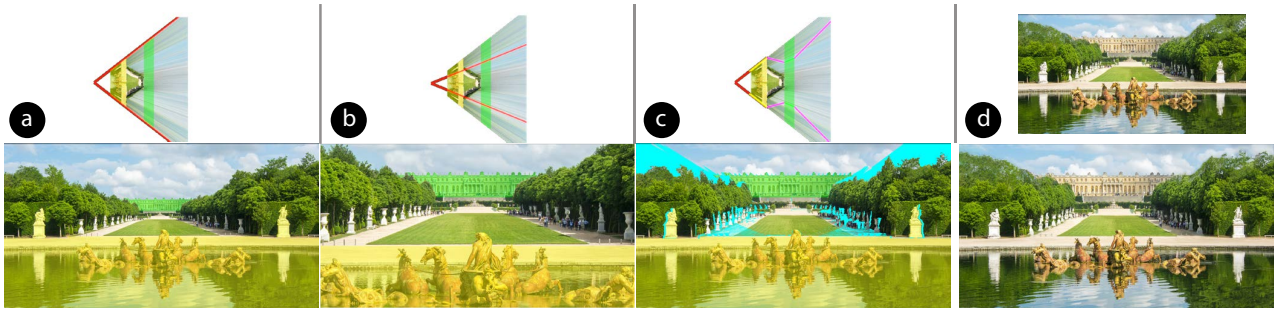


Figure 9: *Versailles [Mis]. Goal: enlarge the Versailles palace while keeping the fountain fixed. a) Original photo. (b) Zoom in and crop. (c) Our result. (d) Our result with inpainting (top: automatic; bottom: with manual guidance). We use a linear interpolant and select depth ranges that border the start and end of the lawn. This hides bending artifacts where the perspective changes.*

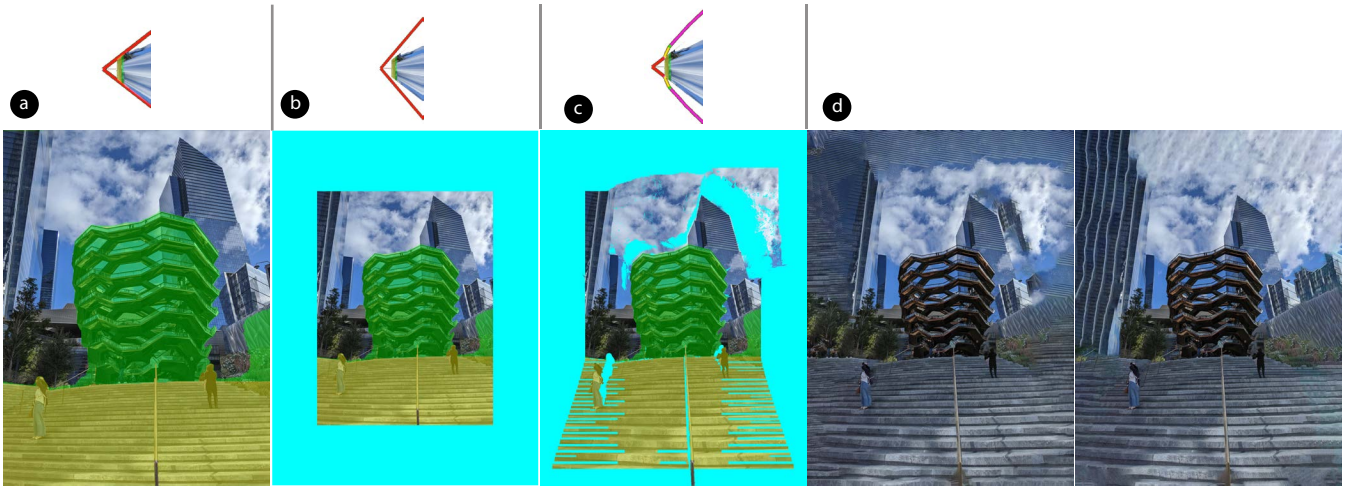


Figure 10: *The Vessel. Goal: expand depth of the steps leading up to the Vessel. (a) Original photo. (b) Zooming out and cropping. (c) Our result. (d) Our result with inpainting (left: automatic; right: with manual guidance).*

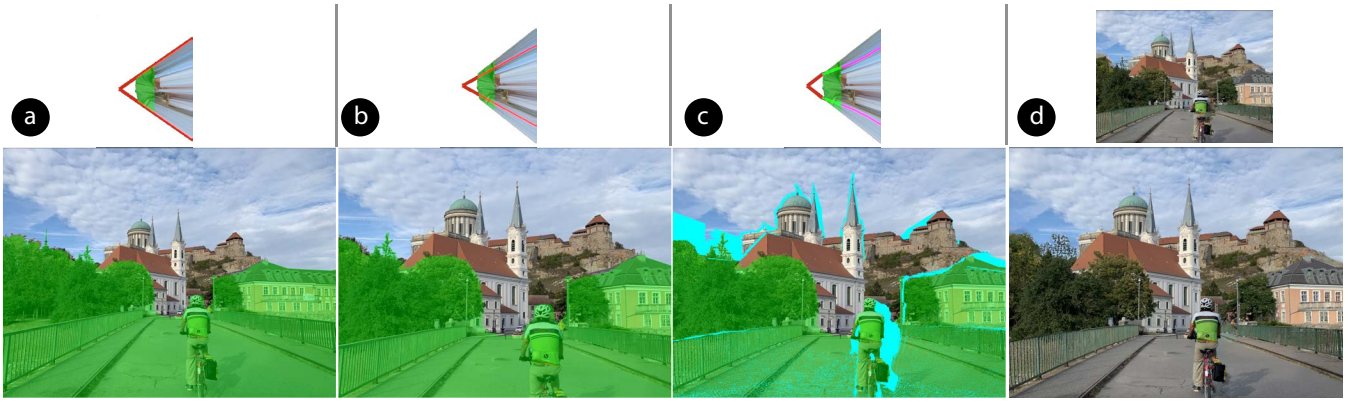


Figure 11: *Biker Bridge. Goal: compress the depth of the bridge to better match our memory. (a) Original photo. (b) Zooming in and cropping. (c) Our result. (d) Our result with inpainting (top: automatic; bottom: with manual guidance).*

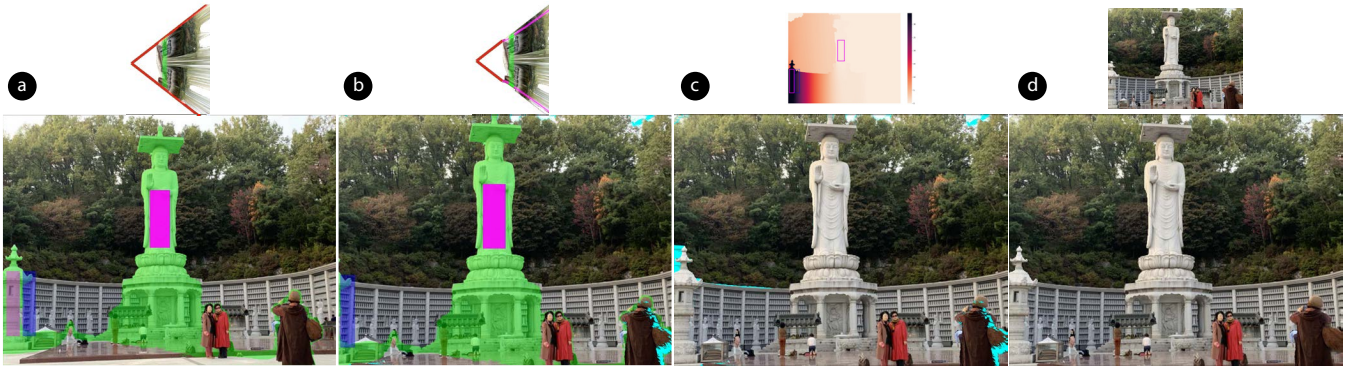


Figure 12: *Buddha. Goal: enlarge the Buddha statue while keeping the left tower in view. Magenta and blue rectangles mark the input to our translation optimization. (a) Original photo. (b) Scaled up Buddha statue. (c) Our result (top: translation map). (d) Our result with inpainting (top: automatic; bottom: with manual guidance).*

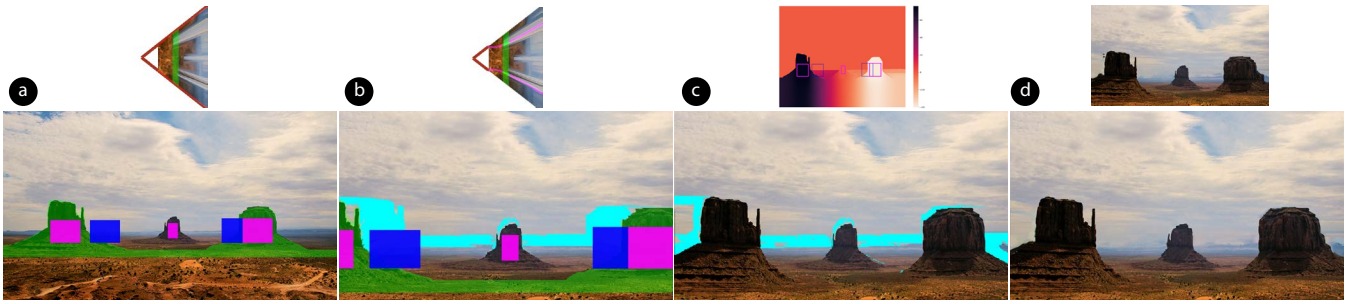


Figure 13: *Monument Valley [Sze14a]. Goal: enlarge the hills, keeping all three fully visible. Magenta and blue rectangles mark the input to our translation optimization (each pair of rectangles has the same size; magenta overlays blue rectangles). (a) Original photo. (b) Scaled up hills. (c) Our result (top: translation map). (d) Our result with inpainting (top: automatic; bottom: with manual guidance).*

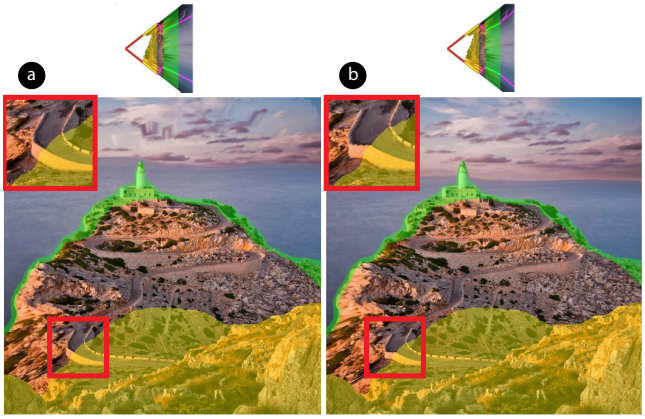


Figure 14: *Lighthouse [Miš11]. Two depth ranges are connected by (a) a curved (cubic Bézier) segment; (b) a linear segment. A linear segment yields an artifact at the far boundary of the yellow depth range, where the road bends suddenly (red boxes). A cubic segment smoothly varies the transition and avoids the artifact.*

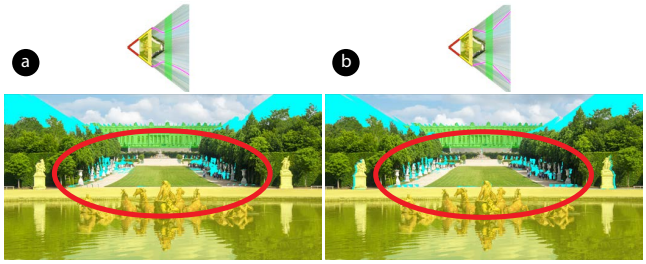


Figure 15: *Versailles [Mis]. Two depth ranges are connected by (a) a curved (cubic Bézier) segment; (b) a linear segment. A curved segment does not preserve the straight lines of the lawn, creating a visible artifact (red circles). A linear segment preserves straight lines but can introduce a bend artifact. The artifact is hidden here because the segment boundaries are at the start and end of the lawn.*

5. Results

We used ZoomShop to edit a variety of images with different composition goals. For each result, we show the original photo, the zoomed-and-cropped baseline, our modified image after applying a nonlinear $b(z)$, and inpainted final results. In cases where manually guided inpainting is necessary, we show both automatic and manual results side-by-side. We also show the view boundary used for each camera model (shown as top-down diagrams).

Our results are 2536 pixels wide and depth maps are 640 to 2048 pixels wide, with heights determined by aspect ratio. Manual clean-up of estimated disparity maps (by a novice user) takes 5 minutes to 3 hours. Scale and foreshortening edits are real-time with immediate feedback. The optimization for object translation takes 2 to 10 minutes. Inpainting with manual guidance takes 5 minutes to 1 hour. Although some manual effort is necessary, we focus on ZoomShop's core contribution of editing composition and rely on the rapidly advancing research for depth estimation and inpainting.

First, we focus on results where we adjust scaling. Figure 1 shows enlarging a boat while keeping foreground trees to maintain framing. Similarly, Figures 2 and 6 create balance by making important background objects (building, lighthouse) more prominent relative to less important foreground objects (boat, ridge). In Figures 7 and 8, the foreground objects establish the context of where each photo was taken from so these elements are preserved while making the primary subjects larger. In Figure 9 the background building is really much larger than the foreground fountain, so it is emphasized by enlarging it.

Next we adjust foreshortening. The structure in Figure 10 is diminished and the foreshortening of the stairs increased, expanding their depth, to create a more dramatic appearance. Conversely, in Figure 11, the foreshortening of the bridge is decreased, compressing its depth and making the scene feel closer.

Finally, we use translation to maintain visibility of important objects. In Figure 12 enlarging the statue pushes the tower out of the image, so it is smoothly translated with the connected wall back into view to restore framing. Multiple objects can also be translated, as in Figure 13, where two of the three hills get scaled out of the frame, and are both moved back towards the center with the ground in front smoothly varying.

We use smooth interpolation for all results except for Figure 9,

Figure 16: *Tourism [Ing; hre]. (a) Goal: enlarge Big Buddha. (b) Goal: enlarge the Eiffel Tower. Left column: original photos. Right column: output images inpainted with manual guidance.*

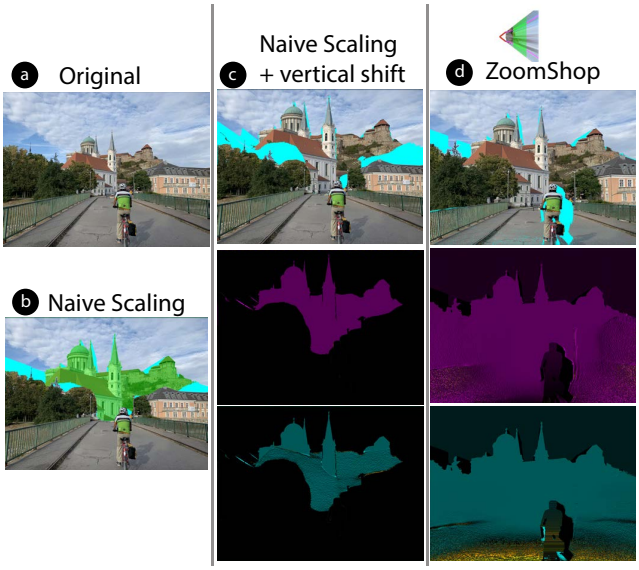


Figure 17: *Biker bridge. Comparison with naively scaling depths independently. (a) Original photo. (b) Naively scaling the church breaks depth continuity and causes the church to become detached from the ground. (c) Users must take care to vertically shift the scaled region to reconnect to the ground. As shown in the horizontal (purple) and vertical (blue) scale maps, this naive approach only uniformly scales pixels of the church. (d) ZoomShop's camera model automatically accounts for depth continuity and prevents the church from detaching from the ground. As shown in the scale maps, ZoomShop scales the pixels of the ground in front of the church as well, to provide a smooth transition of scale across depth.*

where smooth interpolation causes the lawn in front of the building to curve. In that case, we use linear interpolation and select depth ranges that border the start and end of the lawn to hide the perspective changes. Figures 14 and 15 give further examples of the impact of smooth vs. linear interpolation. Figure 16 shows other examples of tourism.

5.1. Comparisons to Alternatives

Scaling with ZoomShop's camera model is different from naively dividing the scene up into multiple depths and scaling each depth independently (Figure 17). Naively scaling scene elements can easily break depth continuity, whereas ZoomShop's camera model automatically accounts for it (unless the user intentionally chooses to break depth continuity by opting for a piecewise discontinuous model).

Niklaus et al. [NMYL19] create animations from an image by reconstructing the 3D scene and moving the camera through the scene. While their method also incorporates depth, their camera model uses linear perspective and thus cannot be used to scale foreground and background elements differently (Fig. 18). Since their goal is to add parallax in their animations, as opposed to photographic composition editing, they do not provide controls for adjusting object size, foreshortening, or position.

While our camera model is inspired by Computational Zoom [BGKS17], our model is more general and supports additional features that are significant to our results. First, ZoomShop supports non-monotonic boundary curves, i.e., segments of $b(z)$ with negative slope, which enables resizing background objects more dramatically (Fig. 19). Second, ZoomShop supports smooth interpolation of the boundary curve, which can be used to reduce visual artifacts in certain types of scenes (Fig. 14). Third, ZoomShop supports discontinuous boundary curves, which enables adjacent depths to be scaled very differently for greater artistic control (Fig. 7). Finally, ZoomShop can operate on a single image and depth map, whereas Computational Zoom requires acquiring multiple images to reconstruct the 3D scene, which would not be pos-

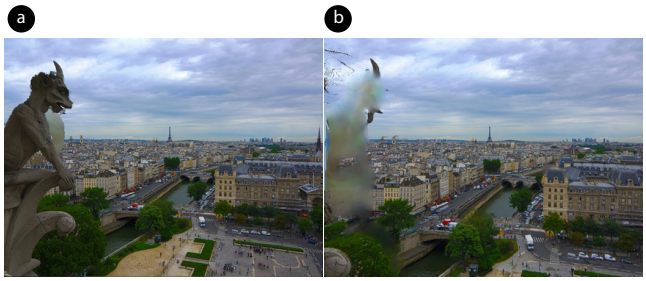


Figure 18: *Eiffel Tower [Mar12]. This result from Niklaus et al. [NMYL19] moves a virtual camera through the 3D scene creating a short animation. (a) The start of the animation. (b) The end of the animation. Because the camera uses linear perspective, it cannot enlarge the Eiffel Tower while keeping the gargoyle in view. Moving the camera forward makes the tower bigger but cuts out the gargoyle.*

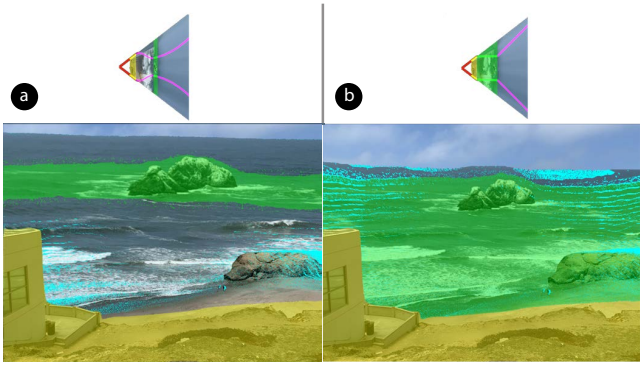


Figure 19: *Seal Rocks. Comparison with Computational Zoom [BGKS17]. (a) ZoomShop's camera model supports non-monotonic boundary curves with negative slope segments, such as on the water between the yellow and green regions, which allows greater changes in scale. (b) Computational Zoom does not support negative slopes, limiting how large we can scale up Seal Rocks while keeping the Cliff House in place.*

sible in may of our results due to physical constraints (Fig. 2) and scene motion (Fig. 11).

5.2. User Impressions

We asked two novice users to try out ZoomShop and give us feedback. Each user adjusted two images; one provided by us (Figure 6) and one they provided themselves. Before adjusting the images, they first stated their editing goals. Both users chose to enlarge some distant object while keeping some foreground elements fixed or reducing their size. In their feedback, both users claimed they achieved their photo editing goals and liked their results, except for artifacts due to depth estimation issues, and assuming that the removed pixels (in cyan) will be inpainted well. One user commented that they often wished to achieve similar edits in their photographs and claimed that ZoomShop was "fun to play with," "really cool," "useful and easy to make a depth composite image," "much easier than Photoshop." Due to unfamiliarity with the controls, both users had minor hiccups when adjusting images but were able to achieve their goals after 1-3 attempts. One user gave some suggestions on using different keyboard mappings for the controls, but both claimed that the controls were "intuitive."

6. Discussion

For most of the results, it was necessary to manually clean up the disparity maps from MiDaS [RLH*20; SSKH20] before using them as input to ZoomShop. Disparity values are inversely related to depth; for simplicity, we discuss our corrections in terms of depth instead of disparity. We include some examples before and after correction in Figure 20.

While metric-accurate depth is not necessary for ZoomShop to generate acceptable results, scene elements do need to have correct depth ordering. When two depths are scaled in different relative amounts under a new $b(z)$, any relative errors between the two depths become more pronounced after scaling and can lead to jarring inaccuracies in the scaled 3D scene. Example distortions include altered object borders (Figure 20a), stretched objects (Figure 20b), and wrong occlusions (Figure 20c).

To fix these errors, we manually masked objects with incorrect depth ordering (e.g., gargoyle) using a combination of Photoshop's Lasso Tool and color range selection [Ado21]. We then added a fill layer with the correct color for the masked region (e.g., fill entire gargoyle with the same color as the ledge it rests on). Because this process is time-consuming, the total time to make manual corrections depends on scene complexity (i.e., more regions to mask) and image resolution (i.e., mask boundaries need to be more accurate, and object outlines need to be more crisp).

For high-quality results, monocular depth estimation needs to produce correct depth ordering among scene elements, correct relative depth inside individual scene elements, and align depth boundaries with object edges. For photographs with humans, depth boundaries need to correctly align with the person's outline and may face more difficulty capturing fine hair. To avoid distortions of 3D face shapes, users can select the person inside a single depth range when making edits.

As mentioned in Section 4.5, Photoshop's CAF [Ado21; BSFG09] has limitations when automatically inpainting removed pixels. In general, CAF automatic inpainting works well for images whose missing pixels are only textured regions, but regions with more scene structure (e.g., the horizon, object boundaries) often require manually specifying regions for CAF to sample from (Figure 21) . For high-quality results, automatic inpainting should respect high-level scene structures in addition to extending textured regions.

Despite current limitations in depth estimation and inpainting, the extraordinarily rapid pace of recent progress [RLH*20; SSKH20; MDM*21; WLS*21; JPY21; LSS*21; JCS*21; SPR21; LWH*21; ZBSA21; ZLLP21; MZH*21] suggest these areas will improve in the coming years. Our goal is to provide new, useful ways to edit photographs given these techniques.

7. Limitations and Future Work

Our non-linear camera model constrains all scene points at the same depth slice to scale by the same amount. While our optimization for object translation makes a step towards non-uniform scaling of scene points at the same depth, future work can investigate ways to make this feature more interactive and/or find other types of user control to non-uniformly scale depth. This could be useful if multiple objects are at the same depth, but the user would like to make some of them larger and more prominent than others.

ZoomShop does not automatically preserve straight lines in the image. Users can select depth ranges in regions where they want linear perspective (where straight lines are preserved) or use a linear interpolant. Future work could find ways to automatically preserve straight lines, regardless of the boundary curve.

ZoomShop lets users choose how to break up a scene by selecting depth ranges, how to piece the modified regions together using interpolants, and whether or not to break depth continuity in turn

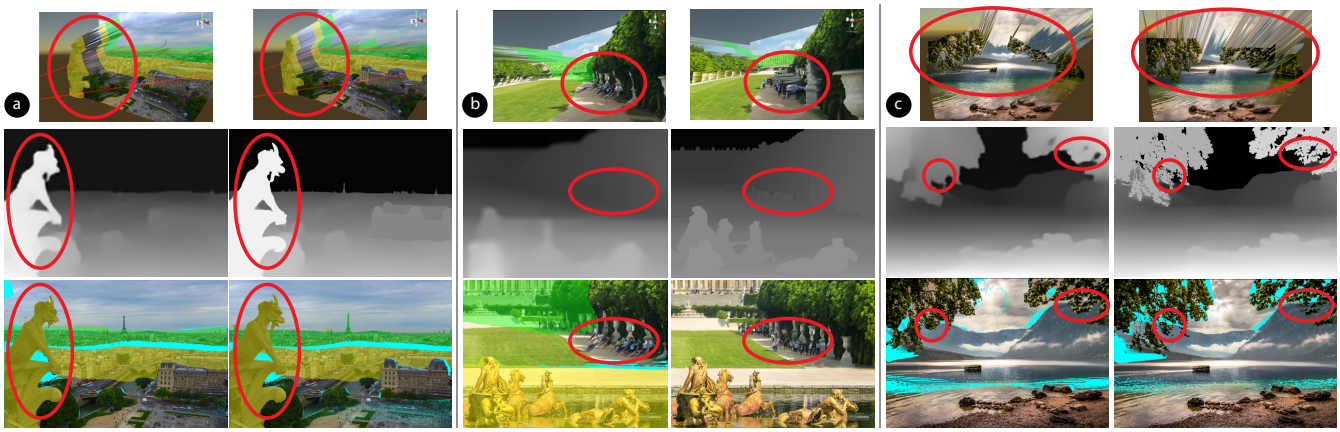


Figure 20: *Correction of Depth Estimation Errors (left: before correction. right: after correction). (a) Blurry edges in Eiffel Tower [Mar12]. Before correction, inaccurate depth estimation along the gargoyle's borders leads to depth bleeding in the 3D scene (top, left) and causes its borders to change after scaling (bottom, left). After correcting the depth inside the gargoyle, the gargoyle's outline is preserved (bottom, right). (b) Incorrect depth within scene elements in Versailles [Mis]. Before correction, the tourists received the same depth as the receding ground and became significantly distorted after scaling (bottom, left). After correction, the tourists remain vertical on the ground (bottom, right). (c) Missed fine features in Mountain Lake Trees [Tha16]. The sky region between the leaves received the same depth as the leaves (top, left); after scaling, the sky incorrectly occludes the mountains (bottom, left). Fixing the depth of the sky between the leaves preserves depth order after scaling (bottom, right).*

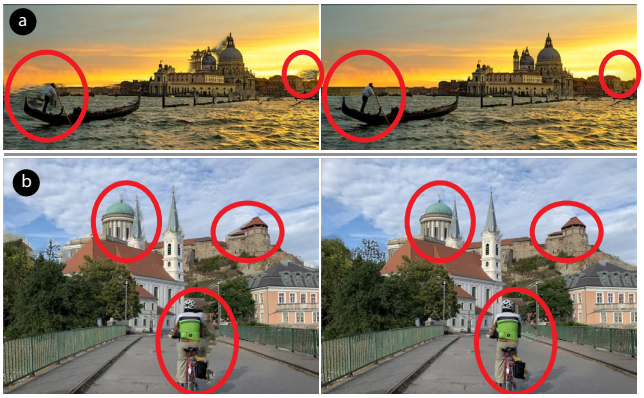


Figure 21: *Inpainting: automatic (left) vs. manually guided (right). (a) Venice [Sze14b]. Automatic inpainting pasted some water texture where the sky should be. (b) Biker bridge. Automatic inpainting caused objects to "bleed" pixels beyond their boundaries. In both of these examples, we fixed these issues by manually specifying regions for CAF to sample from (e.g., sky).*

for greater changes in scale. While these features give users lots of flexibility to achieve a wide range of compositions, future work can investigate ways to automatically determine salient depth ranges and recommend boundary curves to the user.

8. Conclusion

We have presented ZoomShop, a photographic composition editing tool for manipulating the relative sizes, foreshortening, and positions of scene elements. ZoomShop includes a non-linear camera model parameterized by the view boundary curve and a depth-aware image warp optimization to support object translation. These techniques enable a straightforward set of interactions for users to edit images by adjusting the appearance of objects without having to understand 3D scenes or camera models. We believe that ZoomShop increases the ways in which photographers can easily manipulate photographs to improve their expressiveness and better match their artistic intent.

Acknowledgements

We thank Robert Pepperell for helpful discussions; Vaidotas Mišeikis[†], Will Marlow and Wade Tregaskis for giving us permission to use their photos; Pedro Szekely and Bernd Thaller for making their photos available; and our users and reviewers for their valuable feedback. This work was partially supported by the David and Helen Gurley Brown Institute for Media Innovation.

References

[AAC*06] AGARWALA, ASEEM, AGRAWALA, MANEESH, COHEN, MICHAEL, et al. "Photographing long scenes with multi-viewpoint panoramas". *ACM SIGGRAPH 2006 Papers*. 2006, 853–861 2.

[ADCS19] ARAR, MOAB, DANON, DOV, COHEN-OR, DANIEL, and SHAMIR, ARIEL. "Image resizing by reconstruction from deep features". *arXiv preprint arXiv:1904.08475* (2019) 2.

[Ado21] ADOBE. *Adobe Photoshop*. Version 22.1.0. 2021. URL: https://www.adobe.com/products/photoshop.html 2, 5, 6, 11.

[†] Flickr profile: https://flickr.com/v4idas

[AS07] AVIDAN, SHAI and SHAMIR, ARIEL. "Seam Carving for Content-Aware Image Resizing". SIGGRAPH '07. San Diego, California: Association for Computing Machinery, 2007, 10–es. ISBN: 9781450378369. DOI: 10.1145/1275808.1276390 1, 2.

[AZM00] AGRAWALA, MANEESH, ZORIN, DENIS, and MUNZNER, TAMARA. "Artistic multiprojection rendering". *Eurographics Workshop on Rendering Techniques*. Springer. 2000, 125–136 3.

[BBP14] BALDWIN, JOSEPH, BURLEIGH, ALISTAIR, and PEPPERELL, ROBERT. "Comparing artistic and geometrical perspective depictions of space in the visual field". *i-Perception* 5.6 (2014), 536–547 3.

[BCS*09] BROSZ, JOHN, CARPENDALE, SHEELAGH, SAMAVATI, FARAMARZ, et al. "Art and Nonlinear Projection". Jan. 2009 3.

[BGKS17] BADKI, ABHISHEK, GALLO, ORAZIO, KAUTZ, JAN, and SEN, PRADEEP. "Computational Zoom: A Framework for Post-Capture Image Composition". *ACM Trans. Graph.* 36.4 (July 2017). ISSN: 0730-0301. DOI: 10.1145/3072959.3073687 3, 10, 11.

[BPR18] BURLEIGH, ALISTAIR, PEPPERELL, ROBERT, and RUTA, NICOLE. "Natural perspective: Mapping visual space with art and science". *Vision* 2.2 (2018), 21 3.

[BSCS07] BROSZ, JOHN, SAMAVATI, FARAMARZ F., CARPENDALE, M. SHEELAGH T., and SOUSA, MARIO COSTA. "Single Camera Flexible Projection". *Proceedings of the 5th International Symposium on Non-Photorealistic Animation and Rendering*. NPAR '07. San Diego, California: Association for Computing Machinery, 2007, 33–42. ISBN: 9781595936240. DOI: 10.1145/1274871.1274876 3.

[BSFG09] BARNES, CONNELLY, SHECHTMAN, ELI, FINKELSTEIN, ADAM, and GOLDMAN, DAN B. "PatchMatch: A randomized correspondence algorithm for structural image editing". *ACM Trans. Graph.* 28.3 (2009), 24 1, 2, 6, 11.

[CAA09] CARROLL, ROBERT, AGRAWALA, MANEESH, and AGARWALA, ASEEM. "Optimizing Content-Preserving Projections for Wide-Angle Images". *ACM Trans. Graph.* 28.3 (July 2009). ISSN: 0730-0301. DOI: 10.1145/1531326.1531349 1, 2.

[CAA10] CARROLL, ROBERT, AGARWALA, ASEEM, and AGRAWALA, MANEESH. "Image Warps for Artistic Perspective Manipulation". *ACM Trans. Graph.* 29.4 (July 2010). ISSN: 0730-0301. DOI: 10.1145/1778765.1778864 2.

[CH03] COLLOMOSSE, J. P. and HALL, P. M. "Cubist style rendering from photographs". *IEEE Transactions on Visualization and Computer Graphics* 9.4 (2003), 443–453. DOI: 10.1109/TVCG.2003.1260739 3.

[CS04] COLEMAN, PATRICK and SINGH, KARAN. "Ryan: rendering your animation nonlinearly projected". *Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*. 2004, 129–156 3.

[Fou08] FOURQUET, ELODIE. "Composition in perspectives". *Proceedings of the Fourth Eurographics conference on Computational Aesthetics in Graphics, Visualization and Imaging*. 2008, 9–16 3.

[FSGF16] FRIED, OHAD, SHECHTMAN, ELI, GOLDMAN, DAN B, and FINKELSTEIN, ADAM. "Perspective-aware manipulation of portrait photos". *ACM Transactions on Graphics (TOG)* 35.4 (2016), 1–10 2.

[hre] HRENIUCA. *Bride and groom embracing in Paris*. https://stock.adobe.com/images/bride-and-groom-embracing-in-paris/81288729 10.

[Ing] INGUSK. *Young girl standing by the Tian Tan Buddha, Big buddha in Hong Kong*. https://stock.adobe.com/images/young-girl-standing-by-the-tian-tan-buddha-big-buddha-in-hong-kong/189942971 10.

[JCS*21] JAMPANI, VARUN, CHANG, HUIWEN, SARGENT, KYLE, et al. "SLIDE: Single Image 3D Photography With Soft Layering and Depth-Aware Inpainting". *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, 12518–12527 11.

[JPY21] JUNG, HYUNYOUNG, PARK, EUNHYEOK, and YOO, SUNGJOO. "Fine-Grained Semantics-Aware Representation Enhancement for Self-Supervised Monocular Depth Estimation". *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, 12642–12652 11.

[KDPP16] KOENDERINK, JAN, DOORN, ANDREA, PEPPERELL, ROBERT, and PINNA, BAINGIO. "On Right and Wrong Drawings". *Art & Perception* 4 (Feb. 2016), 1–38. DOI: 10.1163/22134913-00002043 3.

[KMA*20] KOPF, JOHANNES, MATZEN, KEVIN, ALSISAN, SUHIB, et al. "One Shot 3D Photography". 39.4 (2020) 2.

[LG96] LÖFFELMANN, HELWIG and GRÖLLER, EDUARD. "Ray tracing with extended cameras". *The Journal of Visualization and Computer Animation* 7.4 (1996), 211–227 3.

[LRS*18] LIU, GUILIN, REDA, FITSUM A, SHIH, KEVIN J, et al. "Image inpainting for irregular holes using partial convolutions". *Proc. ECCV*. 2018 6.

[LSS*21] LI, SHUAI, SHI, JIAYING, SONG, WENFENG, et al. "Hierarchical Object Relationship Constrained Monocular Depth Estimation." *Pattern Recognition* 120 (2021), 108116 11.

[LWH*21] LIU, HONGYU, WAN, ZIYU, HUANG, WEI, et al. "PD-GAN: Probabilistic Diverse GAN for Image Inpainting". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, 9371–9381 11.

[Mar12] MARLOW, WILL. *View from the top of the Notre Dame in Paris*. https://www.flickr.com/photos/williammarlow/7643827866. Modified and included with permission. Original photo licensed under CC BY-NC-SA 2.0. July 2012 7, 10, 12.

[Mat21] MATIASH, BRIAN. *Add Impact to Your Photos with Free Transform in Adobe Photoshop*. https://petapixel.com/2021/04/12/add-impact-to-your-photos-with-free-transform-in-adobe-photoshop/. 2021 1, 3.

[MDM*21] MIANGOLEH, S. MAHDI H., DILLE, SEBASTIAN, MAI, LONG, et al. "Boosting Monocular Depth Estimation Models to High-Resolution via Content-Adaptive Multi-Resolution Merging". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, 9685–9694 11.

[Mis] MISTERVLAD. *Apollo fountain in Versailles gardens, Paris, France*. https://stock.adobe.com/images/apollo-fountain-in-versailles-gardens-paris-france/214090987 7, 9, 12.

[Miš11] MIŠEIKIS, VAIDOTAS. *Faro de Formentor*. https://www.flickr.com/photos/v4idas/6385364319. Modified and included with permission. Original photo licensed under CC BY-NC-ND 2.0. July 2011 6, 9.

[MZH*21] MA, XIN, ZHOU, XIAOQIANG, HUANG, HUAIBO, et al. "Free-form image inpainting via contrastive attention network". *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, 9242–9249 11.

[NMYL19] NIKLAUS, SIMON, MAI, LONG, YANG, JIMEI, and LIU, FENG. "3D Ken Burns effect from a single image". *ACM Transactions on Graphics (TOG)* 38.6 (2019), 1–15 2, 10.

[PKP09] PRITCH, YAEL, KAV-VENAKI, EITAM, and PELEG, SHMUEL. "Shift-map image editing". *2009 IEEE 12th international conference on computer vision*. IEEE. 2009, 151–158 2.

[Rau82] RAUSCHENBACH, BORIS V. "Perceptual Perspective and Cezanne's Landscapes". *Leonardo* 15.1 (1982), 28–33 3.

[RLH*20] RANFTL, RENÉ, LASINGER, KATRIN, HAFNER, DAVID, et al. "Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2020) 2, 5, 11.

[SCSI08] SIMAKOV, DENIS, CASPI, YARON, SHECHTMAN, ELI, and IRANI, MICHAL. "Summarizing visual data using bidirectional similarity". *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2008, 1–8 2.

[SDM19] SHAHAM, TAMAR ROTT, DEKEL, TALI, and MICHAELI, TOMER. *SinGAN: Learning a Generative Model from a Single Natural Image*. 2019. arXiv: 1905.01164 [cs.CV] 1, 2.

[Sin02] SINGH, KARAN. "A fresh perspective". *Graphics interface*. Vol. 2002. Citeseer. 2002, 17–24 3.

[SLBJ03] SUH, BONGWON, LING, HAIBIN, BEDERSON, BENJAMIN B, and JACOBS, DAVID W. "Automatic thumbnail cropping and its effectiveness". *Proceedings of the 16th annual ACM symposium on User interface software and technology*. 2003, 95–104 2.

[SLL19] SHIH, YICHANG, LAI, WEI-SHENG, and LIANG, CHIA-KAI. "Distortion-Free Wide-Angle Portraits on Camera Phones". *ACM Trans. Graph.* 38.4 (July 2019). ISSN: 0730-0301. DOI: 10.1145/3306346.3322948 1, 2.

[SPG10] SHARPLESS, THOMAS K, POSTLE, BRUNO, and GERMAN, DANIEL M. "Pannini: A New Projection for RenderingWide Angle Perspective Images." *Computational Aesthetics*. 2010, 9–16 2.

[SPR21] SUIN, MAITREYA, PUROHIT, KULDEEP, and RAJAGOPALAN, A. N. "Distillation-Guided Image Inpainting". *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, 2481–2490 11.

[SSKH20] SHIH, MENG-LI, SU, SHIH-YANG, KOPF, JOHANNES, and HUANG, JIA-BIN. "3D Photography using Context-aware Layered Depth Inpainting". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 2, 5, 11.

[STR*05] SETLUR, VIDYA, TAKAGI, SAEKO, RASKAR, RAMESH, et al. "Automatic image retargeting". *Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*. 2005, 59–68 2.

[Sze14a] SZEKELY, PEDRO. *Monument Valley*. https://www.flickr.com/photos/pedrosz/35250460535. Modified from original and used under CC BY-SA 2.0. Derivatives are licensed under CC BY-SA 4.0. Aug. 2014 4, 9.

[Sze14b] SZEKELY, PEDRO. *Venice, Italy*. https://www.flickr.com/photos/pedrosz/38269434695. Modified from original and used under CC BY-SA 2.0. Derivatives are licensed under CC BY-SA 4.0. Oct. 2014 2, 12.

[Tha16] THALLER, BERND. *Lake Bohinj*. https://www.flickr.com/photos/bernd_thaller/30816367150. Modified from original and used under CC BY 2.0. Aug. 2016 1, 12.

[Tre20] TREGASKIS, WADE. *Yosemite Valley*. https://www.flickr.com/photos/wadetregaskis/50156486633. Modified and included with permission. Original photo licensed under CC BY-NC 2.0. Feb. 2020.

[WLS*21] WANG, YIRAN, LI, XINGYI, SHI, MIN, et al. "Knowledge Distillation for Fast and Accurate Monocular Depth Estimation on Mobile Devices". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. June 2021, 2457–2465 11.

[YLY*19] YU, JIAHUI, LIN, ZHE, YANG, JIMEI, et al. "Free-form image inpainting with gated convolution". *Proc. ICCV*. 2019 6.

[YM04a] YU, JINGYI and MCMILLAN, LEONARD. "A framework for multiperspective rendering." *Rendering Techniques* 4 (2004), 61–68 3.

[YM04b] YU, JINGYI and MCMILLAN, LEONARD. "General linear cameras". *European Conference on Computer Vision*. Springer. 2004, 14–27 3.

[ZB95] ZORIN, DENIS and BARR, ALAN H. "Correction of geometric perceptual distortions in pictures". *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. 1995, 257–264 2.

[ZBSA21] ZHOU, YUQIAN, BARNES, CONNELLY, SHECHTMAN, ELI, and AMIRGHODSI, SOHRAB. "TransFill: Reference-Guided Image Inpainting by Merging Multiple Color and Spatial Transformations". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, 2266–2276 11.

[ZLLP21] ZENG, YU, LIN, ZHE, LU, HUCHUAN, and PATEL, VISHAL M. "CR-Fill: Generative Image Inpainting With Auxiliary Contextual Reconstruction". *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, 14164–14173 11.

[ZP07] ZELNIK-MANOR, LIHI and PERONA, PIETRO. "Automating Joiners". *Proceedings of the 5th International Symposium on Non-Photorealistic Animation and Rendering*. NPAR '07. San Diego, California: Association for Computing Machinery, 2007, 121–131. ISBN: 9781595936240. DOI: 10.1145/1274871.1274890 3.

[ZPP05] ZELNIK-MANOR, LIHI, PETERS, GABRIELE, and PERONA, PIETRO. "Squaring the circle in panoramas". *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Vol. 2. IEEE. 2005, 1292–1299 2.