Repopulating Street Scenes

Yifan Wang[1*] Andrew Liu[2] Richard Tucker[2] Jiajun Wu[3]
Brian L. Curless[1,2] Steven M. Seitz[1,2] Noah Snavely[2]
[1]University of Washington [2]Google Research [3]Stanford University

Abstract

We present a framework for automatically reconfiguring images of street scenes by populating, depopulating, or repopulating them with objects such as pedestrians or vehicles. Applications of this method include anonymizing images to enhance privacy, generating data augmentations for perception tasks like autonomous driving, and composing scenes to achieve a certain ambiance, such as empty streets in the early morning. At a technical level, our work has three primary contributions: (1) a method for clearing images of objects, (2) a method for estimating sun direction from a single image, and (3) a way to compose objects in scenes that respects scene geometry and illumination. Each component is learned from data with minimal ground truth annotations, by making creative use of large-numbers of short image bursts of street scenes. We demonstrate convincing results on a range of street scenes and illustrate potential applications.

1. Introduction

Websites such as Google Street View enable users to explore places around the world through street-level imagery. These sites can provide a rich sense of what different locales—neighborhoods, parks, tourist sites, etc—are really like. However, the imagery provided by such sites also has key limitations. A given image might be full of cars and pedestrians, making it difficult to observe the environment. Alternatively, a user might want to see how a scene appears at a certain time of day, e.g., lunchtime, but only have access to a morning image. And, importantly, the fact that the imagery records real people and vehicles may require anonymization efforts to protect privacy, e.g., by blurring faces and license plates [10, 1] or by removing pedestrians from images by leveraging multiple views [9].

We propose learning-based tools that mitigate these limitations by removing objects from a scene and then repopulating that scene with, for instance, anonymized images and vehicles. Our method could thus be used to enhance privacy of imagery, while also increase flexibility to compose new scenarios (e.g., an empty street or lunchtime scene). These capabilities could also be useful in other applications, such as automatic generation of novel scene configurations as a way to augment data for training autonomous driving—especially emergency scenarios that might be rare in real data. In order for such reconfigured images to look realistic, they must respect the illumination and geometry of the underlying scene. Our learning-based method takes such factors into account.

As shown in Fig. 1, our framework takes as input a single street image, and can realistically remove all objects and generate repopulated images. To remove objects, we use nearby patches to inpaint not only the objects themselves, but also the shadows they cast onto the scene. To repopulate with new objects, our framework can automatically select objects that match the lighting of the scene, and compose them into the scene with proper scale, occlusion, and cast shadows consistent with the scene's geometry and lighting.

Our method consists of four main components:

1. a *removal* network that removes all existing objects (cars, pedestrians, and bicycles) – along with their shadows – from a street image and realistically fills the resulting holes, rendering an empty version of the scene;
2. a *sun estimation* network that takes a street image, and estimates a dominant lighting (sun) direction;
3. a method to compose the inserted object into the scene with proper scale, occlusion based on its placement in the scene; and
4. an *insertion* network that takes a segmented object, e.g., drawn from an anonymized collection, and inserts it into a scene, generating a realistic shadow.

Our method yields realistic results that improve upon prior work. In particular, unlike standard methods such as image inpainting [27, 38, 39, 37], our approach can realistically remove and render object shadows. Instead of learning to cast shadows into one specific scene captured with a long video sequence [35], our approach learns from short image bursts, then generalizes to any single image of a street scene.

Our three networks are learned in a novel way that involves observing large numbers of street scenes, with no

*This work was done while Yifan was an intern at Google.

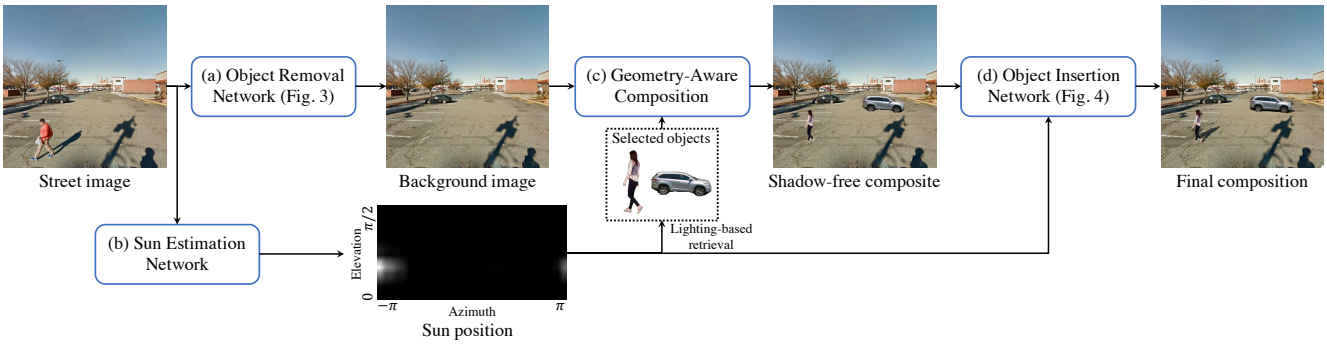

Figure 1. Our reconfiguration pipeline has four major components: (1) a removal network that learns to remove existing objects and their shadows, (2) a sun estimation network that learns to predict sun position from an image, (3) a method to scale and insert newly inserted objects with correct occlusion ordering, and (4) an insertion network that learns to cast shadows. Given a street image, our method first removes selected objects, then selects new object that matches the lighting of the scene, composes them with correct scale and occlusion order into the background image, and synthesizes shadows for inserted objects.

manual annotations required. In particular, we leverage large numbers of short timelapse image sequences gathered from Google Street View. In these timelapses, objects such as pedestrians, bicyclists, and cars are generally in motion, enabling us to estimate a ground truth "clean plate" image devoid of moving objects (and their shadows) by computing a median image. Pairs of input and clean plate frames thus give us ground truth images with and without objects, which we use as the basis for training. We show that this data, along with metadata such as camera pose and time of day, are sufficient supervision for our task.

Our work is motivated by goals such as improving visualizations of scenes and enhance image privacy. We demonstrated two potential applications using our framework: (1) emptying the city by removing all objects within an image and (2) repopulating the scenes with anonymized people. These applications are designed to enhance the privacy of a street image while preserving the realism. However, any deployment of our methods in a real-world setting would need careful attention to responsible design decisions. Such considerations could include clearly watermarking any user-facing image that has been recomposed, and matching the distribution of anonymized people composed into a scene to the underlying demographics of that location.

2. Related Work

Our work is related to prior work on object removal, object insertion, and lighting estimation.

Object removal. Prior work on object removal falls mainly into two groups: (1) image inpainting methods and (2) methods for detecting and removing object shadows.

Recently, deep learning and GAN-based approaches have emerged as a leading paradigm for image inpainting. Liu *et al*. [27] inpaint irregular holes with partial convolutions that are masked and re-normalized to be conditioned on valid pixels. Gated convolutions [39] generalize such partial convolutions by providing a learnable dynamic feature selection mechanism for each channel and at each spatial location. Contextual attention [38, 37] allows for long-range spatial dependencies, allowing pixels to be borrowed from distant locations to fill missing regions. Shadows have different forms, e.g., hard shadows, soft shadows, partially occluded shadows, etc. Hole-filling based methods have trouble determining what pixels to inpaint in different shadow scenarios.

Deep learning methods have also been applied to shadow removal. Qu *et al*. [30] extract features from multiple views and aggregate them to recover shadow-free images. Wang *et al*. [34] and Hu *et al*. [16] use GANs for shadow removal, while recently Le *et al*. [22] proposed a two-network model to learn shadow model parameters and shadow mattes. However, these methods only inpaint shadow regions. Realistic object removal involves removing both the object and its shadow, as handled by our method.

Other work has sought to remove pedestrians from street scenes by leveraging multiple views [9]. In our case, we operate on just a single view, and can also recompose new people into scenes. Finally, face replacement [4] has been considered for realism-preserving privacy enhancement tool [3]. Our work considers whole people, and not just faces.

Object insertion. Early methods for object insertion include Poisson blending [29], which can produce seamless object boundaries, but can also result in illumination and color mismatches between the object and the target scene. Lalonde *et al*. proposed *Photo Clip Art*, which inserts new objects into existing photographs by first querying a large dataset of cutouts for compatible objects [21]. Other methods match the color, brightness, and styles of inserted objects to harmoniously embed them into background images [24, 2, 40]. However, a realistic insertion should also consider an object's effect on the background (including shadows).

Some methods insert a 3D object by rendering it into in an image. Karsch *et al*. demonstrate convincing object inser-

tions via inverse rendering models derived from geometric inference [18] or via single-image depth reconstruction [19]. Other work renders inserted objects with estimated HDR environment lighting maps [15, 14]. Chuang *et al*. synthesize shadows for inserted objects via a shadow displacement map [7]. However, these approaches essentially require a full 3D model of either the inserted object or the scene. Liu *et al*. [26] focus on single light source scenes containing hard shadows, whereas our method can handle scenes with soft shadows and spatially varying lighting. Recently, Wang *et al*. [35] proposed a data-driven method that takes a long video of a scene and learns to synthesize shadows for inserted 2D cutout objects. Our method learns from short bursts of images, and can synthesize shadows for 2D cutouts given a single image of a new scene at test time.

Lighting estimation. To capture illumination Debevec [8] captures HDR environment maps via bracketed exposures of a chrome ball. Subsequent methods [5, 11, 15, 20, 23, 33] use machine learning to predict HDR environment maps from single indoor or outdoor images. However, a single environment map is insufficient for compositing cut-out objects into a large captured scene, because different lighting effects will apply depending on, for instance, whether the object is placed in a sunlit area or a shadowed one. In our work, we do not explicitly estimate lighting for each scene, but instead use a rendering network that implicitly learns to generate shadows appropriate for the object location.

Outdoor illumination is primarily determined by the sun position and the weather conditions. Recent works [28, 15, 14, 25] use data-driven methods to estimate the sun azimuth angle from a single outdoor image. Our work follows this trend and estimates a full 2D sun angle. We find that estimating the sun position aids in synthesizing plausible shadows in different weather and lighting scenarios.

3. Approach

Given an image of a street scene, our goal is to recompose the objects (*e.g*., cars and pedestrians) in the scene by first removing the existing objects, and then optionally composing one or more new objects into the scene. These stages must respect the illumination in the scene—in particular, shadows (both their removal and insertion) are critical elements that are difficult to handle realistically in prior work.

Our automatic pipeline for addressing this problem has four major components, as shown in Fig. 1: (1) a removal network that learns to remove existing objects and their shadows, (2) a lighting estimation network that learns to predict sun position from an image, which helps identify compatible objects for insertion and is used to create better insertion composites, (3) a method to scale the inserted object properly with correct occlusion order based on its placement in the scene, and (4) an insertion network that learns to cast shadows for newly inserted objects. Given a street image, we first use Mask R-CNN [36] to segment existing people and cars, then use that as a mask for the object removal stage. The object removal network completely removes those objects (and their shadows), yielding a background image. The sun estimation network is used to select new objects that match the scene's lighting. Selected objects are then composed into the background image with correct scale and occlusion order to get a shadow-free composite. Finally, our insertion network takes the shadow-free composite, synthesizes shadows, and outputs the final composite.

Figure 2. Our dataset consists of short image bursts, *i.e*., short timelapse sequences of images captured over several seconds. We compute the median of each timelapse image stack to produce a "clean plate" background image free of objects and shadows.

3.1. Data

We train our networks in a novel way by using a dataset of image bursts, i.e., short timelapse image sequences captured over several seconds. As shown in Fig. 2, we compute the median of each timelapse image stack to produce a "clean plate" background image free of (moving) objects such as people and their shadows. We also know location and time of day for each timelapse, from which we derive the sun position. (We do not use weather data; the sun position is noted regardless of cloud cover.) Images with and without moving objects, and corresponding sun positions serve as ground truth supervision for our object removal, sun prediction, and object insertion networks. At test time, our pipeline takes in a single street image and can remove and repopulate the objects within. We now describe the four components of our pipeline.

3.2. Removal Network

Object removal is a challenging task that involves generating new content in holes left by removed objects, such that the new image is realistic and semantically correct. Given a mask indicating the objects to be removed, standard inpainting methods [27, 38, 39, 37] only fill masked regions, leaving behind shadows. Our goal is to remove both objects and their shadows. We propose a deep network that, given an image and an object mask, constructs a new mask that includes the object and its shadow, then inpaints the region inside this mask. Inspired by PatchMatch-based inpainting [12] and appearance flow [32], our method predicts a flow map that uses nearby patches' features to inpaint the masked region.

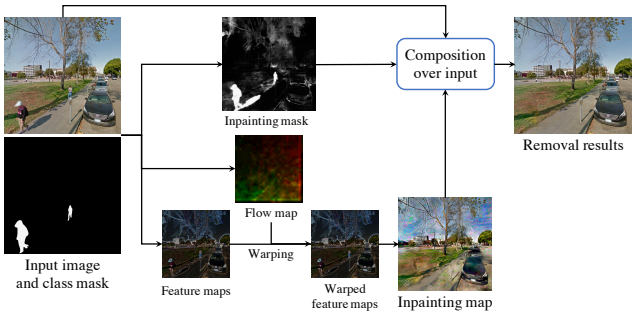

Figure 3. The generator of the removal network takes an input image I and a class mask as input, and outputs an inpainting mask M_{inp} and an inpainting map I_{inp}. We synthesize the removal image $I_{\text{remove}} = I_{\text{inp}} \odot M_{\text{inp}} + I \odot (1 - M_{\text{inp}})$.

Algorithm. Standard inpainting methods operate on an image and a binary mask designating where to inpaint. In our case, the network also automatically detects shadow regions belonging to masked objects. Different objects have different shadow shapes—for example, people can have long, thin, complex shadows, while cars tend to have larger, simpler shadows. Hence, rather than taking a binary mask, our network receives an image and a class mask produced by Mask R-CNN [36]. This class mask encodes the object category of each pixel with distinct values normalized to $[0, 1]$.

Fig. 3 shows the removal network architecture. We feed the input image I and its class mask through four downsampling layers followed by three different branches of residual blocks. The first branch predicts a full inpainting mask M_{inp}, including the object and its shadow; the second predicts a warping flow map F_{warp}; and the third encodes the image as a high-dimensional inpainting feature map F_{inp}. The feature map is then warped by the predicted flow map F_{warp} and fed into four upsampling layers to produce an inpainting image I_{inp}. The final removal image is then computed as

$$I_{\text{remove}} = I_{\text{inp}} \odot M_{\text{inp}} + I \odot (1 - M_{\text{inp}}). \quad (1)$$

The feature warping layer uses the high-dimensional features of nearby patches to inpaint the missing area. We found that street scenes often have highly repetitive structures—building facades, fences, road markings, etc.—and the feature warping layer works well in these situations.

3.3. Sun Estimation Network

Lighting is key to realistic image composition. An object lit from the left composed into a scene lit from the right will likely look unrealistic. Many traditional lighting estimation methods reconstruct an environment map [5, 11, 15, 20, 23, 33]. However, a single environment map is insufficient for compositing objects into the scene, because different lighting effects will apply depending on object placement, e.g., whether the object is the shade or lit by the sun. In our work, we do not explicitly estimate scene illumination, but instead predict the sun position with a deep network and use the result to help synthesize plausible shadows. Further, we apply the same network to choose objects with similar sun position to be inserted.

Algorithm. Rather than regressing an image to sun azimuth and elevation, we treat this as a classification problem and predict a distribution over discretized sun angles. We divide the range of azimuth angles $[0, 2\pi)$ into 32 bins and elevation angles $[0, \pi/2]$ into 16 bins. We use a network similar to ResNet50 [13], replacing the last fully connected layer with two, one for azimuth and one for elevation. We train this network using ground truth sun positions as supervision via a cross-entropy loss. In Fig. 1 and 4, we visualize estimated sun position as a 2D distribution formed by the outer product of azimuth and elevation distribution vectors.

3.4. Scene Geometry for Occlusion and Scale

When composed into a scene, a new object should be scaled properly according to its 3D scene position, and should have correct occlusion relationships with other scene structures. To that end, we desire accurate depth estimates for both the target scene and source object, and propose a method to robustly estimate depth for the scene and object jointly. Our method, unlike [43], also reasons about occlusion ordering for inserted objects. Here we take people as an example of inserted objects, but our method also works on other objects, including cars, bikes, and buses. We make three assumptions: (1) the sidewalk and road regions in the image can be well-approximated by a single plane; (2) there is at least one person present; and (3) people are roughly the same height in 3D. If the second assumption is not met, the user can manually adjust the height scale. The third assumption, while not universally true, facilitates depth estimation by treating individual height difference as Gaussian noise.

Algorithm. As described in [35] (Eq. 7), any object's bottom middle point (x, y) and height h follow a linear relationship:

$$a'x + b'y + c' = h \quad (2)$$

Also under perspective projection, the object's height h is up to a scale factor k with its disparity $1/Z$:

$$h = k \cdot \frac{1}{Z} \quad (3)$$

Combining Eq. 2 and 3, we have a linear relation between pixel coordinates and the disparity $1/Z$:

$$a'x + b'y + c' = \frac{1}{Z} \quad (4)$$

Given the input image, we first use DeepLab [6] to segment pixels belonging to sidewalk and road. We then use MiDaS [31] to predict a depth map for the scene. MiDaS

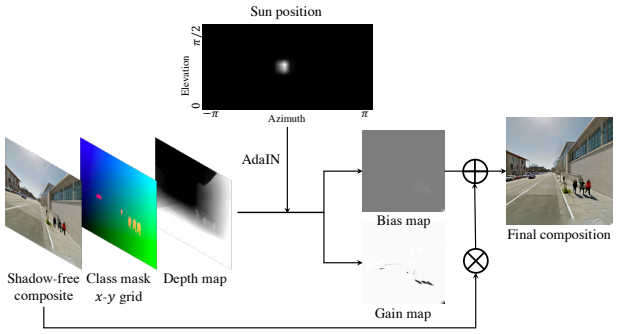

Figure 4. The generator of the insertion network takes as input a shadow-free composite image I_{comp}, a class mask, an x-y grid map, a depth map, and the predicted sun position distribution, and outputs a scalar gain image G and color bias image B. Given the shadow-free composite image I_{comp}, we synthesize the final image $I_{\text{final}} = G \odot I_{\text{comp}} + B$.

predicts the disparity map $\hat{D}$ up to a global scale and shift. Therefore, the linear relationship in Eq. 4 still holds. After collecting all 2D road/sidewalk pixels (x_i, y_i) and their disparities $\hat{d}_i$, we use least squares to solve for (a', b', c') in Eq. 4. Finally, we solve for the scale factor k in Eq. 3 using existing objects and their observed 2D heights. If there is no object in the scene, a user can manually set the scale factor.

When inserting a new object into the scene at 2D position (x, y), we apply Eq. 4 and Eq. 3 to estimate its disparity d and height h, and resize the inserted object accordingly. We then resolve occlusion order by comparing the object's disparity d and the scene's disparity $\hat{D}$ from MiDaS. Pixels with larger disparity than d are foreground and will occlude the object, and pixels with smaller disparity than d will be occluded by the object.

3.5. Insertion Network

Shadows are one of the most interesting and complex ways in which objects interact with a scene. As with the removal network, predicting shadows for inserted objects is challenging, as their shapes and locations depend on sun position, weather, and the shape of both the object casting the shadow and the scene receiving it. Furthermore, unlike other lighting effects, shadows are not always additive, as a surface already in shadow does not darken further when a second shadow is cast on it with respect to the same light source. We propose to use observations of objects in the scene along with the scene's predicted geometry and lighting to recover these shadowing effects, using a deep network to learn how objects cast shadows depending on their shape and scene placement. Unlike the work of Wang *et al*., which is trained on a long video of a scene and can only insert objects within that same scene [35], our method learns from a database of short image bursts, and can then be applied to a single, unseen image at test time.

Algorithm. Our insertion network takes as input a shadow-free composite image I_{comp} (where the desired object is simply copy-pasted into the scene). As with the removal network, we consider that shadow effects vary significantly across object categories, and we also provide the class mask introduced in Sec. 3.2 as input. In addition, because shadows depend on scene geometry and illumination, we use MiDaS [31] to predict a depth map for the shadow-free image, and feed this to the insertion network, along with the sun position distribution from the sun estimation network. Finally, following [35], we feed a x-y grid map to the network to help stabilize training. As shown in Fig. 4, I_{comp}, the class mask, x-y grid map, and depth map are concatenated and passed through four downsampling layers then five residual blocks. The sun azimuth and elevation vectors are concatenated, passed through four MLP layers and fused into five residual blocks via AdaIN [17]. Finally, following [35], two different upsampling layers generate a scalar gain image G and color bias image B. The final image is computed as

$$I_{\text{final}} = G \odot I_{\text{comp}} + B. \tag{5}$$

4. Evaluation

In this section, we introduce our collected datasets, then evaluate our entire pipeline and its individual components.

4.1. Data

We collected short image bursts of street scenes from Google Street View. These bursts are captured in major US cities, and encompass a range of outdoor urban scenes including streets, parks, and parking lots, under lighting conditions ranging from clear to cloudy. In total, we collected $142,778$ image/background pairs for training and $16,034$ as a test set. The test images are drawn from cities near the training ones to ensure no overlap between training and test sets. We center-crop all images to 512×512 with a field of view of $75°$. To better evaluate performance, we also randomly picked a small subset ($\sim$150) of sunny images where objects cast hard shadows, and a subset ($\sim$180) of cloudy images where objects cast subtle soft shadows from the test set.

4.2. Sun Estimation Network

We train our sun estimation network (Sec. 3.3) on the training set with ground truth supervision. Ground truth sun azimuth and elevation angles are calculated from each image's location, orientation, and timestamp using solar equations. Our network takes a street image and outputs two vectors describing distributions of azimuth and elevation angles. To compute a single pair of angles, we find the highest probability bin from each vector, and use the bin center as the estimated angle. We compare our sun estimation network with [28, 14], adding a fully connected layer to their method to predict the elevation angle. On average over the test set, our azimuth prediction has an angular error of $35.71°$ vs.

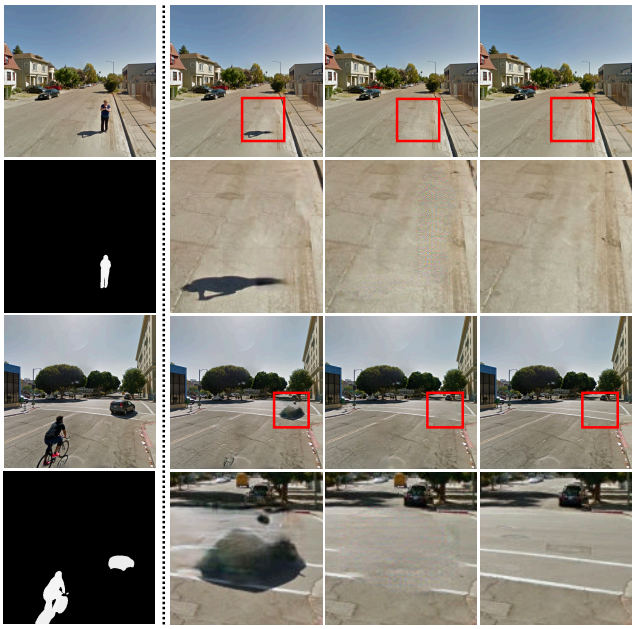

Figure 5. Object removal results on the test set. The traditional inpainting method [37] only inpaints the area within the mask and has leftover shadows. Our method removes objects completely along with shadows. In addition, the inpainting method fails to inpaint for large object (car in the second example).

$50.17°$ [14] vs. $52.59°$ [28], and our elevation prediction has an angular error of $9.79°$ vs. $13.02°$ [14] vs. $13.82°$ [28]. We further convert the sun angles to directions on the unit sphere and compute the angle between predicted and ground truth vectors, yielding an average error of $27.00°$. These error rates are reasonably low, and of suitable accuracy for applications like lighting matching and shadow prediction.

4.3. Removal Network

We trained our removal network (Sec. 3.2) on the training set with median images as supervision for the object removed images. And the supervision for inpainting masks is computed by thresholding the color difference between median images and original images. Fig. 5 shows example object removal results on our test set using (1) the state-of-the-art inpainting method CRA [37] (trained on Places2 [42]), which only inpaints the mask area and (2) our method, which also predicts an enlarged inpainting mask. Both networks realistically inpaint the object region; however, CRA fails to remove object shadows since they are not included in the mask. Our network yields a complete object removal, which overall is more realistic. We show quantitative results using the LPIPS [41] error metric in Tab. 1. Both methods achieve a lower error compared to the input image. Ours has much lower LPIPS error than CRA on sunny days (where our method benefits from removing hard shadows), and slightly lower on cloudy days (where the shadows are more subtle). As shown in Fig. 5, our method removes objects completely and performs better in the task of object removal. We further tried running CRA [37] using our thresholded inpainting mask. This method gave an LPIPS score of 0.162 on the test set (vs. ours at 0.104). CRA is not trained with our inpainting mask, thus cannot adapt errors in our mask estimation, leading to artifacts in the final output.

Method	All	Sunny	Cloudy
Input	0.113	0.099	0.096
CRA [37]	0.107	0.090	0.083
Ours	**0.104**	**0.079**	**0.080**

Table 1. Object removal results on all test images, the sunny subset, and the cloudy subset, measured in LPIPS [41]. Lower is better.

Method	All	Sunny	Cloudy
Shadow-free composite	0.073	0.060	0.058
Shadow network	0.069	0.054	**0.056**
Ours (w/o x-y grid)	0.079	0.065	0.069
Ours (w/o sun position)	**0.068**	0.054	**0.056**
Ours (w/o depth map)	0.078	0.060	0.062
Ours	**0.068**	**0.053**	**0.056**

Table 2. Object insertion results on all test images, the sunny subset, and the cloudy subset, measured in LPIPS [41]. Lower is better.

4.4. Insertion Network

Our insertion network (Sec. 3.5) is trained to take shadow-free composite images and render object shadows, using original images as ground truth supervision. Fig. 6 shows example results using (1) a baseline pix2pix-style method [35] that takes a shadow-free image and an x-y grid; (2) an ablative method that takes a shadow-free image, x-y grid and depth map; (3) an ablative method that takes a shadow-free image, x-y grid, and predicted sun position; and (4) our method. All methods are trained on our training set. The predicted sun position helps the network produce shadows in the right direction. The depth map and x-y grid stabilize training, preventing the network from overfitting and producing broken or detached shadows. Quantitative results shown in Tab. 2 suggest that our method has an advantage over other models. On sunny days, our full model benefits from the depth map, sun position and x-y grid, and outputs realistic, detailed shadows. On cloudy days, our model synthesizes subtle soft shadows, still performing the best overall.

5. Applications

In this section, we discuss potential applications of our method to recomposing or repopulating single street images. These applications are enabled by one or more of the components of our pipeline. We also discuss ethical considerations involved in such applications in Sec. 6.

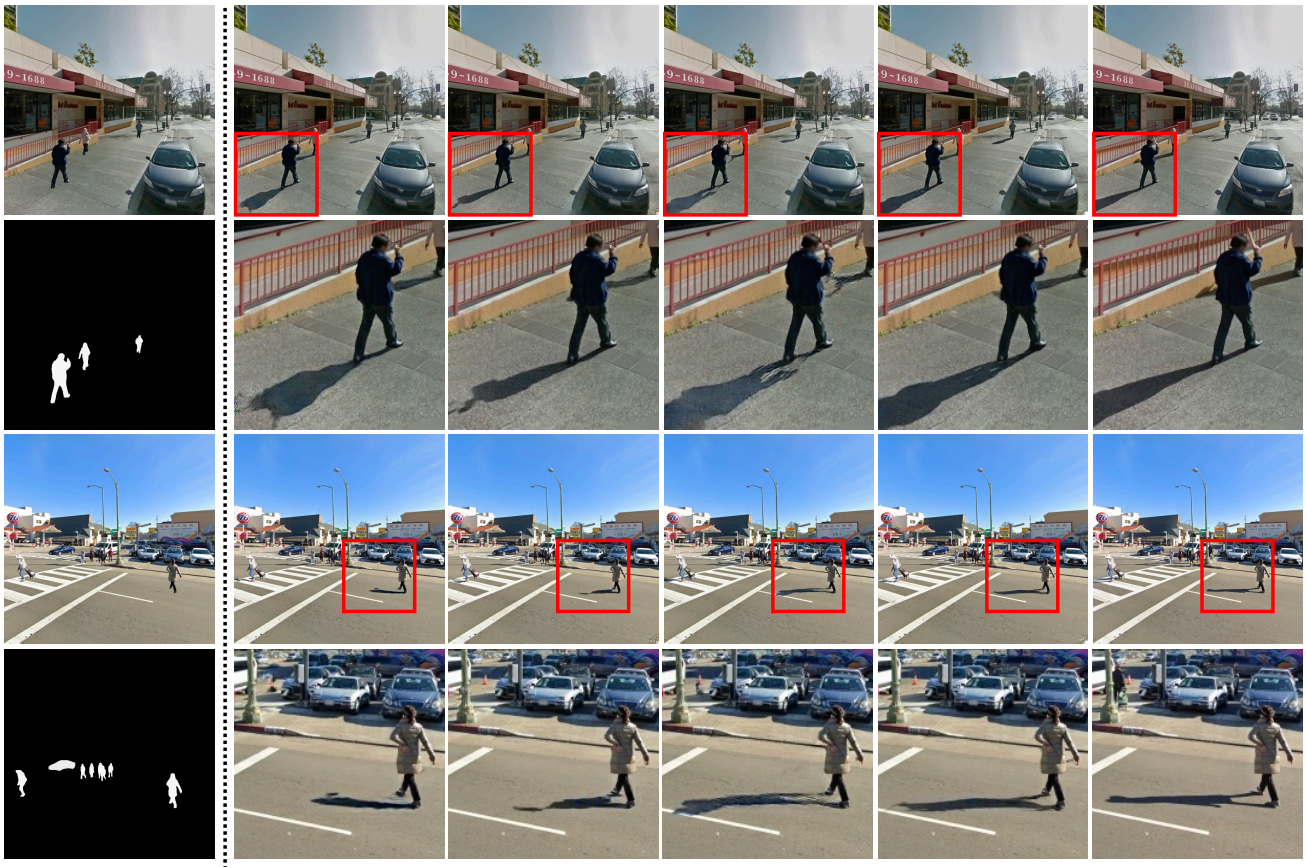

Figure 6. Object insertion results on the test set. Our method generates the most realistic shadows with details. The sun position input helps the network to determine the shape of the shadow. The depth map prevents the network from synthesizing broken or detached shadows.

Object lighting matching. When repopulating scenes, selecting objects with similar lighting as the scene is crucial for realistic composition. Hence, we wish to compute the sun position for both the source object and target scene. Hence, we train two sun estimation networks (Sec. 3.3), one for scenes and one for objects. The scene sun estimation network takes the image I and predicts sun azimuth and elevation vectors $a_{\text{scene}}, e_{\text{scene}}$, while the object sun estimation network pre-computes sun angle vectors $a_{\text{obj,i}}, e_{\text{obj,i}}$ for each object o_i in the collection. The object o_i that maximizes $a_{\text{scene}} \cdot a_{\text{obj,i}} + e_{\text{scene}} \cdot e_{\text{obj,i}}$ is then selected as the object that best matches the scene's lighting.

Emptying the city. Mask R-CNN [36] can segment out certain set of objects (people, cars, bikes, etc). Our removal network then takes this mask, and synthesizes an image without those objects along with their shadows. This enables applications such as removing all people and cars in NYC or LA. As demonstrated in Fig. 7, we use our removal network to remove all the objects—people and cars—in the image, giving users a different visualization of a city. Hence, it can also enhance the privacy of the imagery. Our method successfully removes all objects along with their shadows from the given street image.

Privacy enhancement. While removing all the people in the image enhances privacy, it decreases the liveliness of the street scene as well. To that end, we built a collection of people viewed from the back (or nearly the back) from licensed imagery on Shutterstock. Our pipeline can populate scenes with such people, thus enhancing privacy while retaining a sense of liveliness within the scene.

As above, our method can remove whole categories of objects to yield a background frame I_{back}. Then, we can use our object lighting matching method to find a set of best matching objects, then randomly place each object o_i on sidewalk and road regions in I_{back} via the segmentation map in Sec. 3.4. Objects will be automatically resized and occluded using the methods described in Sec. 3.4 to get the shadow-free composition I_{comp}. Finally I_{comp} is passed to the insertion network to synthesize the final composition I_{final}. Fig. 8 shows results for repopulating street scenes. We substitute the people in the scene with anonymized people, thus enhancing privacy while preserving the realism of street scenes. Note that our work focuses on lighting, and does not attempt to match the camera viewpoint for inserted objects

Figure 7. Qualitative results for removing all people and cars in a street image. From left to right: the input image, the class mask for objects to be removed, and the removal results generated by the removal network. Our method removes objects completely.

as in [21] or compensate for differences in camera exposure, white balance, etc. These are left as future work.

Other applications. We have also developed an interactive scene reconfigurator that leverages the elements of our framework. With this tool, a user can take a street image and remove selected existing objects, or conversely, place new objects in the scene. This tool can synthesize street images that are rare in real life, e.g., people walking in the middle of a busy road, or cars driving on the sidewalk. These synthesized scenes could be used for data augmentation for autonomous driving to simulate dangerous situations.

6. Discussion and Ethical Considerations

In this paper, we introduced a fully automatic pipeline for populating, depopulating, or repopulating street scenes. Our pipeline consists of four major components: (1) a removal network that can remove selected objects along with their shadows; (2) a sun estimation network that predicts the sun position from an image; (3) a method to scale and occlude inserted objects properly; and (4) an insertion network that synthesizes shadows for inserted objects. These components are trained on short image bursts of street scenes, and can run on a single street image at test time. Further, we show multiple applications of our pipeline for depopulating and repopulating street scenes.

While our work is motivated by goals like improving visualizations of scenes and enhance image privacy, it is important to consider the broader impacts and ethical aspects of computer vision research, particular work related to synthetic imagery. Potential harmful outcomes relating to recomposing street scenes include (1) misuse in creating a false narrative, such as a crowd or protest in a certain location, and (2) misrepresenting a neighborhood by changing the demographics of people therein. In our case, some issues related to synthetic media are mitigated by inherent limitations of our method—for instance, our method can compose separated people into scenes, and synthesize their shadows cast on the ground, but would have trouble generating a dense crowd of people where people would be shadowing each other. That said, any deployment of our methods in a real-world setting would need careful attention to responsible design decisions. Such considerations could include clearly watermarking any user-facing image that has been recomposed, and matching the distribution of anonymized people composed into a scene to the underlying demographics of that location. At the same time, our work may lead to knowledge useful to counter-abuse teams working on manipulated imagery and synthetic media data methods.

Figure 8. Qualitative results for repopulating street scenes. From left to right: the input image, the background image after removing all people, and repopulation results. Our pipeline selects people matching the scene's lighting, places them randomly on sidewalks and roads, and synthesizes realistic shadows.

References

[1] Dragomir Anguelov, Carole Dulong, Daniel Filip, Christian Frueh, Stéphane Lafon, Richard Lyon, Abhijit Ogale, Luc Vincent, and Josh Weaver. Google street view: Capturing the world at street level. *Computer*, 43, 2010. 1

[2] Samaneh Azadi, Deepak Pathak, Sayna Ebrahimi, and Trevor Darrell. Compositional gan: Learning image-conditional binary composition. *arXiv preprint arXiv:1807.07560*, 2018. 2

[3] Dmitri Bitouk, Neeraj Kumar, Samreen Dhillon, Peter Belhumeur, and Shree K Nayar. Face swapping: automatically replacing faces in photographs. In *ACM SIGGRAPH 2008 papers*, pages 1–8. 2008. 2

[4] Volker Blanz, Kristina Scherbaum, Thomas Vetter, and Hans-Peter Seidel. Exchanging faces in images. In *Computer Graphics Forum*, volume 23, pages 669–676. Wiley Online Library, 2004. 2

[5] Dan A Calian, Jean-François Lalonde, Paulo Gotardo, Tomas Simon, Iain Matthews, and Kenny Mitchell. From faces to outdoor light probes. In *Computer Graphics Forum*, volume 37, pages 51–61. Wiley Online Library, 2018. 3, 4

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 4

[7] Yung-Yu Chuang, Dan B Goldman, Brian Curless, David H Salesin, and Richard Szeliski. Shadow matting and compositing. In *ACM SIGGRAPH 2003 Papers*, pages 494–500. 2003. 3

[8] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *ACM SIGGRAPH 2008 classes*, pages 1–10. 2008. 3

[9] Arturo Flores and Serge Belongie. Removing pedestrians from Google Street View images. In *Computer Vision and Pattern Recognition Workshops*, pages 53–58, 2010. 1, 2

[10] Andrea Frome, German Cheung, Ahmad Abdulkader, Marco Zennaro, Bo Wu, Alessandro Bissacco, Hartwig Adam, Hartmut Neven, and Luc Vincent. Large-scale privacy protection in google street view. In *Int. Conf. Comput. Vis.*, 2009. 1

[11] Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. Learning to predict indoor illumination from a single image. *arXiv preprint arXiv:1704.00090*, 2017. 3, 4

[12] Kaiming He and Jian Sun. Statistics of patch offsets for image completion. In *European Conference on Computer Vision*, pages 16–29. Springer, 2012. 3

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[14] Yannick Hold-Geoffroy, Akshaya Athawale, and Jean-François Lalonde. Deep sky modeling for single image outdoor lighting estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6927–6935, 2019. 3, 5, 6

[15] Yannick Hold-Geoffroy, Kalyan Sunkavalli, Sunil Hadap, Emiliano Gambaretto, and Jean-François Lalonde. Deep outdoor illumination estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7312–7321, 2017. 3, 4

[16] Xiaowei Hu, Yitong Jiang, Chi-Wing Fu, and Pheng-Ann Heng. Mask-shadowgan: Learning to remove shadows from unpaired data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2472–2481, 2019. 2

[17] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 5

[18] Kevin Karsch, Varsha Hedau, David Forsyth, and Derek Hoiem. Rendering synthetic objects into legacy photographs. *ACM Transactions on Graphics (TOG)*, 30(6):1–12, 2011. 3

[19] Kevin Karsch, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, Hailin Jin, Rafael Fonte, Michael Sittig, and David Forsyth. Automatic scene inference for 3d object compositing. *ACM Transactions on Graphics (TOG)*, 33(3):1–15, 2014. 3

[20] Jean-François Lalonde, Alexei A Efros, and Srinivasa G Narasimhan. Estimating natural illumination from a single outdoor image. In *2009 IEEE 12th International Conference on Computer Vision*, pages 183–190. IEEE, 2009. 3, 4

[21] Jean-François Lalonde, Derek Hoiem, Alexei A Efros, Carsten Rother, John Winn, and Antonio Criminisi. Photo clip art. *ACM transactions on graphics (TOG)*, 26(3):3–es, 2007. 2, 8

[22] Hieu Le and Dimitris Samaras. Shadow removal via shadow image decomposition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8578–8587, 2019. 2

[23] Chloe LeGendre, Wan-Chun Ma, Graham Fyffe, John Flynn, Laurent Charbonnel, Jay Busch, and Paul Debevec. Deeplight: Learning illumination for unconstrained mobile mixed reality. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5918–5928, 2019. 3, 4

[24] Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey. St-gan: Spatial transformer generative adversarial networks for image compositing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9455–9464, 2018. 2

[25] Andrew Liu, Shiry Ginosar, Tinghui Zhou, Alexei A. Efros, and Noah Snavely. Learning to factorize and relight a city. In *European Conference on Computer Vision*, pages 544–561. Springer, 2020. 3

[26] Daquan Liu, Chengjiang Long, Hongpan Zhang, Hanning Yu, Xinzhi Dong, and Chunxia Xiao. Arshadowgan: Shadow generative adversarial network for augmented reality in single light scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[27] Guilin Liu, Kevin J Shih, Ting-Chun Wang, Fitsum A Reda, Karan Sapra, Zhiding Yu, Andrew Tao, and Bryan Catanzaro. Partial convolution based padding. *arXiv preprint arXiv:1811.11718*, 2018. 1, 2, 3

[28] Wei-Chiu Ma, Shenlong Wang, Marcus A Brubaker, Sanja Fidler, and Raquel Urtasun. Find your way by observing the

sun and other semantic cues. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6292–6299. IEEE, 2017. 3, 5, 6

[29] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *ACM SIGGRAPH 2003 Papers*, pages 313–318. 2003. 2

[30] Liangqiong Qu, Jiandong Tian, Shengfeng He, Yandong Tang, and Rynson WH Lau. Deshadownet: A multi-context embedding deep network for shadow removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4067–4075, 2017. 2

[31] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 4, 5

[32] Yurui Ren, Xiaoming Yu, Ruonan Zhang, Thomas H Li, Shan Liu, and Ge Li. Structureflow: Image inpainting via structure-aware appearance flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 181–190, 2019. 3

[33] Soumyadip Sengupta, Jinwei Gu, Kihwan Kim, Guilin Liu, David W Jacobs, and Jan Kautz. Neural inverse rendering of an indoor scene from a single image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8598–8607, 2019. 3, 4

[34] Jifeng Wang, Xiang Li, and Jian Yang. Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1788–1797, 2018. 2

[35] Yifan Wang, Brian L. Curless, and Steven M. Seitz. People as scene probes. In *European Conference on Computer Vision*, pages 438–454. Springer, 2020. 1, 3, 4, 5, 6

[36] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 3, 4, 7

[37] Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. Contextual residual aggregation for ultra high-resolution image inpainting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7508–7517, 2020. 1, 2, 3, 6

[38] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018. 1, 2, 3

[39] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4471–4480, 2019. 1, 2, 3

[40] Fangneng Zhan, Hongyuan Zhu, and Shijian Lu. Spatial fusion GAN for image synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3653–3662, 2019. 2

[41] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[42] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017. 6

[43] Rui Zhu, Xingyi Yang, Yannick Hold-Geoffroy, Federico Perazzi, Jonathan Eisenmann, Kalyan Sunkavalli, and Manmohan Chandraker. Single view metrology in the wild. *arXiv preprint arXiv:2007.09529*, 2020. 4