Animating Street View

Mengyi Shan
shanmy@cs.washington.edu
University of Washington
Seattle, WA, US

Brian Curless
curless@cs.washington.edu
University of Washington
Seattle, WA, US

Ira Kemelmacher-Shlizerman
kemelmi@cs.washington.edu
University of Washington
Seattle, WA, US

Steve Seitz
seitz@cs.washington.edu
University of Washington
Seattle, WA, US

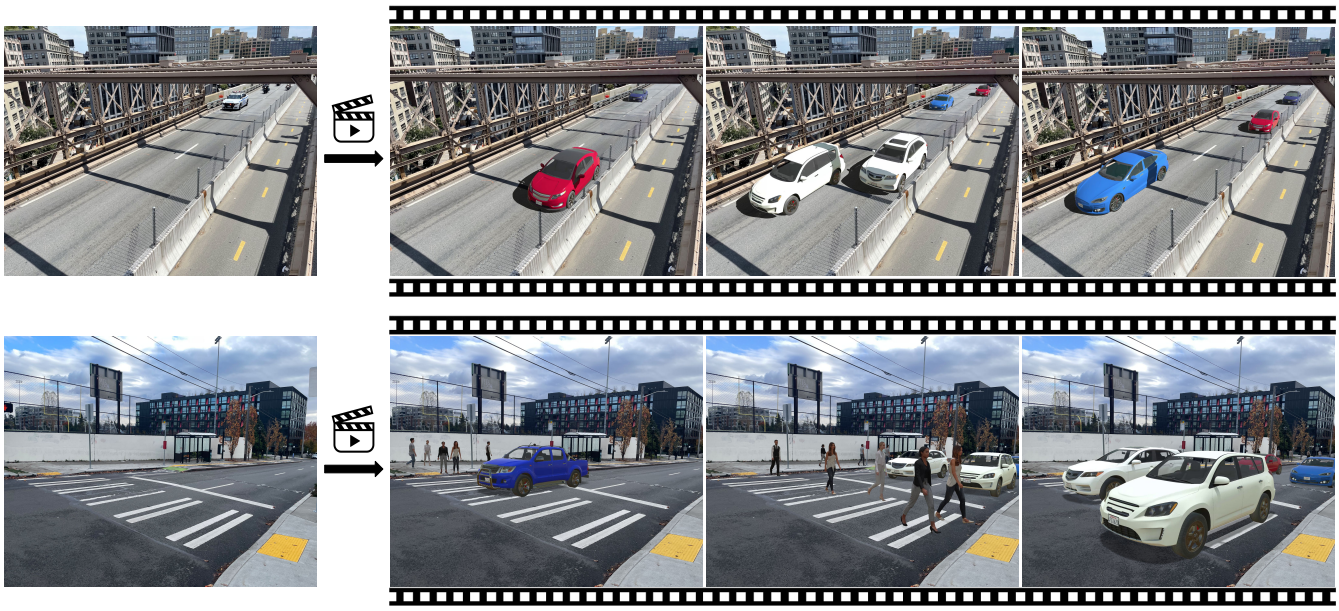

Figure 1: Given a single input street image or panorama, our system automatically removes existing people or vehicles, and populates it with simulated animated pedestrians and cars. The result is a high resolution, arbitrarily long video. Our system handles scaling, placement, path planning and traffic simulation, as well as estimates and renders correct lighting, occlusion, and shadows.

ABSTRACT

We present a system that automatically brings street view imagery to life by populating it with naturally behaving, animated pedestrians and vehicles. Our approach is to remove existing people and vehicles from the input image, insert moving objects with proper scale, angle, motion and appearance, plan paths and traffic behavior, as well as render the scene with plausible occlusion and shadowing effects. The system achieves these by reconstructing the still image street scene, simulating crowd behavior, and rendering with consistent lighting, visibility, occlusions, and shadows. We demonstrate results on a diverse range of street scenes including regular still images and panoramas.

CCS CONCEPTS

• **Computing methodologies** → **Animation**; **Rendering**; **Computer vision**; **Modeling and simulation**; **Computer graphics**.

KEYWORDS

Single image animation, scene reconstruction, crowd simulation, rendering, game engine

ACM Reference Format:
Mengyi Shan, Brian Curless, Ira Kemelmacher-Shlizerman, and Steve Seitz. 2023. Animating Street View. In *SIGGRAPH Asia 2023 Conference Papers (SA*

SA Conference Papers '23, December 12–15, 2023, Sydney, NSW, Australia

ACM ISBN 979-8-4007-0315-7/23/12.
https://doi.org/10.1145/3610548.3618230

Conference Papers '23), December 12–15, 2023, Sydney, NSW, Australia. ACM, Sydney, Australia, 12 pages. https://doi.org/10.1145/3610548.3618230

1 INTRODUCTION

Google Street View and similar services allow people to virtually visit locations worldwide with street-level imagery. These platforms offer an immersive experience of different areas such as neighborhoods, streets, and tourist attractions. Nevertheless, the major limitation of the visual content provided by these services is that they are all still – notably, imagery without people or cars moving through each scene. Street-level video could address this shortcoming but would be difficult due to constraints on image capture systems, storage, and privacy requirements. We propose a system that mitigates these limitations by first erasing still pedestrians and vehicles from an image or panorama of a scene and repopulating it with synthetic, moving pedestrians and vehicles. Our method can thus enhance the vividness of street view imagery without additional capture and without privacy concerns.

Our framework takes as input a single street image or panorama and generates a video of arbitrary length by populating it with moving pedestrians and vehicles. In order for the generated videos to look realistic, they must respect the geometry, lighting, and semantic information – e.g., labeling sidewalks and streets – of the given scene, while modeling reasonable behaviors of newly added elements (people and vehicles). Our method takes all these factors into consideration in three main stages (as shown in Figure 2):

(1) **Reconstruction**: estimating the visible geometry, lighting, and semantic information of a scene.
(2) **Simulation**: inserting and modeling the behavior of pedestrians and vehicles in a given street setting.
(3) **Rendering**: synthesizing realistic videos with consistent lighting, shadows, and occlusions layered into the original frames.

Our method is the first to demonstrate realistic results for street scene image/panorama animation, producing arbitrarily long videos with moving people and cars. We combine learning-based approaches with traditional rendering to deliver visually appealing results without large training sets, large scale annotations, long training time or advanced hardware. Our approach leverages a number of ingredients from prior art. Specifically, our geometry *reconstruction*, *crowd simulation* and shadow/occlusion *rendering* components build on existing pre-trained networks and rendering systems, except for a sun estimation sub-module that we trained. And while these ingredients by themselves are not novel, our system combines them in novel ways to enable an exciting new application. We only require a single image as input with a known focal length, or a 360° panorama. The whole pipeline can be run on a conventional laptop. Figure 1 illustrates several output frames for three different scenes. Please refer to the supplementary material for video results.

2 RELATED WORK

2.1 Object Insertion

Early insertion work includes Poisson blending [Pérez et al. 2003] which produces seamless object boundaries but results in illumination and color mismatches between object and the target background. [Karsch et al. 2014] and [Karsch et al. 2011] recover geometry and lighting from a single image for object insertion. However, [Karsch et al. 2011] requires manual insertion, and neither demonstrates automatic occlusion behavior, shadow cast by the scene onto objects, or object placement/movement. More recent work renders inserted objects with estimated lighting maps and shadow synthesis [Tang et al. 2022; Wang et al. 2022], but does not handle sharp lighting changes near shadow boundaries. A number of recent methods learn end-to-end object insertion and composition based on GANs [Azadi et al. 2020; Lin et al. 2018; Zhan et al. 2019] and diffusion models [Ma et al. 2023; Song et al. 2022].

Specifically for street scenes, [Chien et al. 2017; Lee et al. 2018; Sun et al. 2020] estimate the locations of roads and walking paths. [Wang et al. 2020, 2021] *manually* insert *static* pedestrians and vehicles into a scene image, selected to have scene-compatible lighting and pose, but do not support animation. [Wang et al. 2020] also requires video as input. In contrast, our approach inserts 3D dynamical subjects, is automatic, and operates on a single image.

The problem of animating street scenes has been less well studied. [Lee et al. 2019] learn to composite a given video clip of a pedestrian onto a scene video, but with limited quality. [Xu et al. 2022] train a 3D-aware GAN given scene image, but only show short (1-2 seconds) video clips with noticeable artifacts. Closest to our work is [Chen et al. 2021] which composes moving car assets into a scene video, and [Wang et al. 2022], but they require 5-7 posed RGB imagery and LiDAR for NeRF or 3D-aware geometry representation. [Chen et al. 2021] also only supports vehicles instead of the much more complex pedestrian/vehicle interaction. Compared with previous works, our system only requires a still image, and handles pedestrians in addition to vehicles.

2.2 Still Image Animation

Image animation aims to generate a video given one or a sequence of still images as input. [Holynski et al. 2021; Okabe et al. 2011] show beautiful animation results assuming repetitive motions, e.g., waterfalls. [Huang et al. 2022b; Peng et al. 2021; Weng et al. 2019; Yoon et al. 2021] animate existing people in the scene, and [Mallya et al. 2022; Pumarola et al. 2019] animate faces. [Mallya et al. 2022; Wang et al. 2019] take in conditional signals like facial keypoints or pose sequences provided by a driving video and synthesize video in 2D space, and [Huang et al. 2022a] reconstruct an animatable implicit representation from a still image. [Hu et al. 2022; Yu et al. 2022] animate a single image by semantic instructions. There are also a line of work on 3D-aware GAN for scene generation ([Epstein et al. 2022; Nguyen-Phuoc et al. 2020; Niemeyer and Geiger 2021; Xue et al. 2022]), which can be leveraged for scene animation by GAN inversion. Recent methods generate full videos via diffusion models, e.g., [Ni et al. 2023] focuses on human motion and fixed camera pose, and [Karras et al. 2023] show animation specific to fashion models. We focus on 3D scene modeling for the purpose of

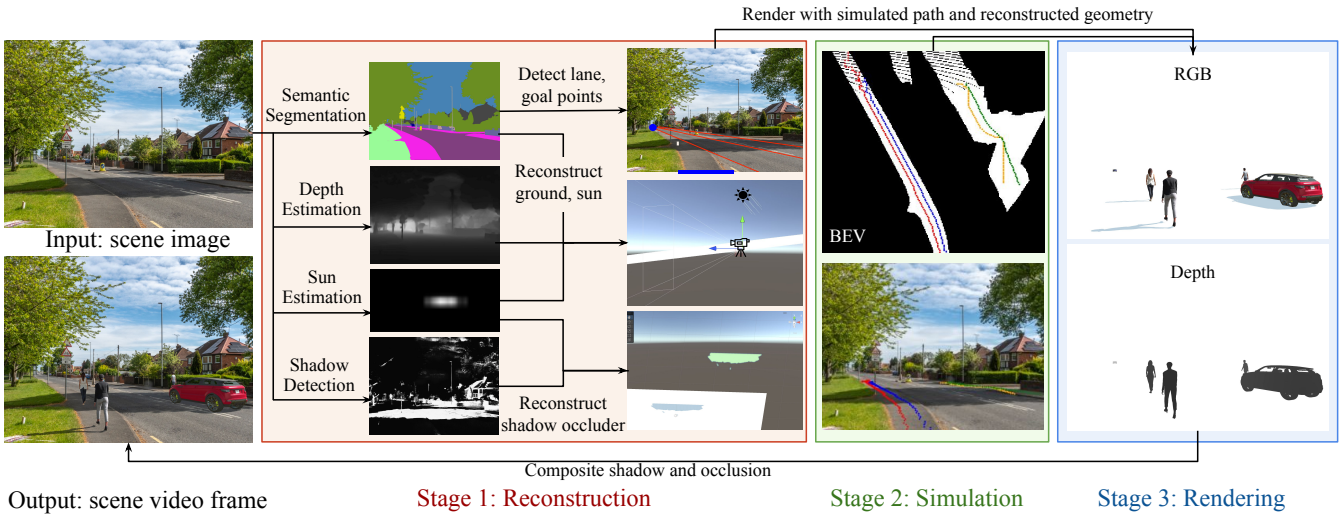

Figure 2: Our system has three major components. In Stage 1, we reason about the scene by predicting its semantic segmentation labels, depth values, sun direction and intensity, as well as shadow regions. We additionally determine walking and driving regions for adding pedestrians and cars (red straight lines: lane detection; blue points: origin and destination points). In Stage 2, we simulate the pedestrians in a 2D bird's eye view representation (BEV) of the scene, and simulate car movements with predicted lanes (four colors correspond to four predicted path, both in BEV and scene images). If there is a detected crosswalk, we also simulate the traffic behavior by controlling a traffic light (not shown in this example, refer to Figure 10). In Stage 3, we render the scene with the estimated lighting, shadows, and occlusions. The whole pipeline is automated.

occlusion-aware and lighting-consistent animation of novel objects' interaction within the scene.

2.3 Playable Video Generation

[Menapace et al. 2022, 2021] address Playable Video Generation that learns semantically consistent actions and generates realistic videos conditioned on the input. Similarly, [Zhang et al. 2021, 2023] produce videos by modeling interactively controllable video sprites and composing them onto an empty scene. [Davtyan and Favaro 2022] learn to segment video into foreground-background layers and generate transitions of the foreground over time. [Kim et al. 2021] learn to simulate a dynamic driving environment directly in pixel-space. These methods require a large video dataset for training and are thus limited to a narrow set of scene images, for example a tennis court from a fixed angle, or a first-person driving scene. Our method is the first approach to animate street scene without heavy training by populating it with naturally behaving characters.

3 APPROACH

Given an image of a street scene, our goal is to generate an arbitrarily long animation of this scene populated with objects including pedestrians and vehicles. The resulting video should respect the geometry and illumination of the scene, contain realistic shadow and occlusion effects, and feature natural traffic behaviors.

Our automatic pipeline contains three stages. Section 3.1 describes a reconstruction stage where the basic geometry and illumination are estimated and reconstructed. Section 3.2 describes a simulation stage where pedestrians' and vehicles' behaviors are simulated in the reconstructed scene. Section 3.3 describes a rendering stage where characters are placed into the scene and processed to generate desired visual effects.

For simplicity, this section focuses on traditional camera images and omits discussion of hyperparameters. For equirectangular panoramas, we decompose the imagery into six perspective images in a cubemap manner and process each direction separately. A comprehensive discussion of hyperparameters and panorama processing can be found in the supplementary materials.

3.1 Reconstruction

In this section, we describe the reconstruction process, which removes existing pedestrians and vehicles from the image, and estimates the (1) semantic information to determine walking and driving regions for the scene, (2) ground plane for deciding scale and camera angles, (3) sun light direction, intensity and ambient light intensity for lighting the objects, and (4) existing shadow regions in the image for darkening objects when they move into shadow.

3.1.1 Inpainting. If an image contains pedestrians and/or vehicles, we start by removing them with segmentation and inpainting.

As illustrated in Figure 3, we directly take advantage of the pretrained Stable Diffusion inpainting tool [Rombach et al. 2022] to remove existing objects. We segment people and cars using SegFormer [Xie et al. 2021] and compute a bounding box for inpainted objects as a rectangle covering the objects with a 10% margin to accommodate boundary inaccuracy in semantic segmentation. We use *"A photo of an empty street"* as the inpainting prompt, and *"Human, pedestrian, vehicle, car"* as the negative prompt for Stable Diffusion.

Figure 3: Stable Diffusion [Rombach et al. 2022] based inpainting. We detect existing people and cars with segmentation map, crop the surrounding region, inpaint and compose back to the image.

3.1.2 Segmentation. We segment the image to determine where the pedestrians can walk or the vehicles can drive. We start with a semantic segmentation module with off-the-shelf semantic segmentation model SegFormer [Xie et al. 2021] to segment out the ground regions (`sidewalk` and `road` classes in CityScapes [Cordts et al. 2016]). However, we found SegFormer [Xie et al. 2021] sometimes identifies `Road` region as not suitable for driving especially when the scene is not a traditional CityScapes [Cordts et al. 2016] style image. We thus apply an additional zero-shot language-driven segmentation model CLIPSeg [Lüddecke and Ecker 2022] to help us segment the image with keyword "drive". CLIPSeg is adaptable to more segmentation classes in a wider range of scenes, but produces less accurate pixel-level boundaries. If the identified "drive" region is smaller than a threshold t_{d}, we make the scene "pedestrian-only" by considering the `Road` label pixels also as `Sidewalk` pixels, and not adding vehicles to the scene.

To handle the case where there are obstacles on street that partition the walking/driving regions into pieces – e.g., the a pole in the image may split the visible sidewalk into disjoint 2D pieces – we dilate the region and then take the convex hull of each connected component. At the same time, we record the ground position of each obstacle to reconstruct a binary obstacle map with non-walkable obstacle pixels. We utilize the estimated ground plane equation to project the semantic information onto a bird's eye view (BEV) map (Figure 2 top figure in Stage 2), which contains the spatial information for walking/driving regions and obstacle positions. This BEV map is discretized into a grid and serves as the abstract representation for pedestrian simulation. Note that vehicle simulation is a different process and not shown in this BEV map.

3.1.3 Ground Plane Estimation. We make the assumption that the ground region in the image can be well-approximated by a plane

$$aX + bY + cZ = 1 \tag{1}$$

and the goal is to estimate the values of a, b, c in metric units (1/meters). This approach is similar to [Wang et al. 2021].

We use the monocular depth estimation model AdaBins [Bhat et al. 2021] to predict the absolute depth map in meters. According to the properties of perspective projection, for known focal length f, image pixels x, y and 3D positions X, Y, Z, we have: $x = Xf/Z$ and $y = Yf/Z$. Plugging into Equation 1 and rearranging yields

$$\frac{a}{f}x + \frac{b}{f}y + c = \frac{1}{Z} \tag{2}$$

Given the segmentation map, we collect all 2D road/sidewalk pixels (x_i, y_i) and their depth Z_i, and solve for a, b, c via linear least squares. For simplicity, we assume that the road and sidewalk are co-planar.

This plane is then the surface on which pedestrians will walk and vehicles will drive in simulation, augmented with the segmentation map to restrict those regions and the depth map to determine occlusions between the scene and inserted assets.

3.1.4 Sun Estimation. In order to realistically light the inserted objects, we train a sun estimation network that learns to predict the sun direction from a single image. We formulate it as a classification problem and predict a distribution over discrete sun angles. We divide the range of azimuth angles $[0, 2\pi)$ into 36 bins and elevation angles $[0, \pi/2)$ into 18 bins, each bin spanning 5 degrees. We train with a ResNet50 [He et al. 2015], but replace 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. This sun estimation approach is closely based on [Wang et al. 2021]. We compute the variance of the predicted elevation/azimuth bin distributions to decide if the scene has a strong, directional light or a diffuse environment light, and accordingly set the directional light source strength and ambient light when lighting inserted objects. The thresholds of variance and corresponding light intensities are provided in the supplementary material.

For training and ground truth data, we use a panorama dataset [Chang et al. 2018] with estimated sun direction for Google Street View panoramas from sky appearance and metadata. We take perspective images from each panorama with diverse camera angles and FOV, and compute the ground truth sun elevation and azimuth for these images. Refer to Figure 4 for an example data point, and to Section 4 for a discussion on data preprocessing.

3.1.5 Scene Shadow Estimation. Analyzing local shadows in the scene is important for realistic object insertion. In particular, shadows cast by inserted objects should not darken existing shadows, and shadows cast by elements in the scene, e.g., by in-scene structures or trees, should also be cast onto the inserted objects.

We start by detecting the existing shadows in the image using an off-the-shelf shadow detection network [Zhu et al. 2021]. We keep ground shadow regions by intersecting this binary map with the ground region obtained from semantic segmentation.

With the estimated ground plane equation and sun direction from previous steps, we now construct a shadow occluder to explain shadows cast into the scene. In particular, we first create a plane parallel to the ground plane and displaced vertically above the height of any potentially inserted objects and then intersect rays shadow rays – from the 3D positions of shadowed scene points toward the light direction – with this plane. We connect these points into a mesh to construct the shadow occluder and use it to cast shadows onto inserted objects. See Figure 5 for an illustration of this pipeline.

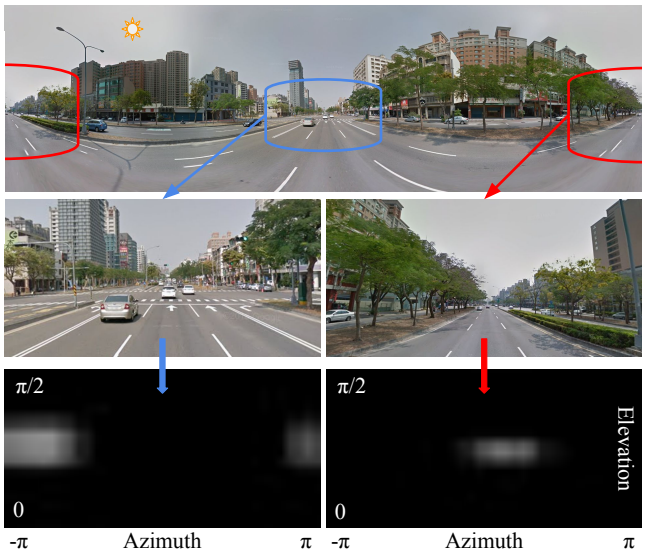

Figure 4: A 360 degree panorama is projected onto planar images with various camera poses and FOVs. The system is trained to predict sun light direction (ground truth shown in the panorama) by estimating its elevation and azimuth. The bottom row visualizes possible light location estimated by the network, shown as the outer product between azimuth and elevation distribution vectors. The sun is in different positions relative to the camera viewpoints for the two crops.

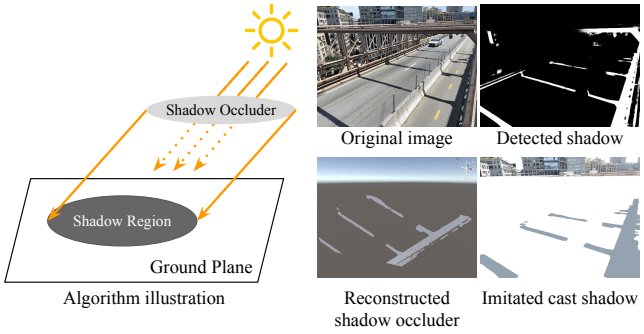

Figure 5: We detect the shadow region in the image, and trace rays in 3D from the corresponding points toward the light source through a fixed plane above all potential objects in the scene. The reconstructed 3D shadow occluder then casts shadows onto objects in the scene.

3.2 Simulation

After reconstructing the basic scene geometry and semantics, we utilize the scene understanding information to simulate the movement of pedestrians and vehicles in the scene. We start with estimating origin and destination points for pedestrian movements, and then run a simple yet effectively potential field-based crowd simulation algorithm. For scenes with more complex structure, we also design a traffic simulation module that naturally controls the behavior of pedestrians and cars at traffic lights and crosswalks.

3.2.1 Origin/Destination Estimation. We would like each pedestrian to either appear or disappear from one edge of the image and disappear or appear, respectively, at the far end of the scene. Thus we build a pool of possible origin/destination pairs containing the intersection of walkable regions and image boundaries, as well as the farthest points from the camera. To simulate each pedestrian, we randomly select one pair of origin and destination from this pool and check if there is an existing pedestrian in this same or neighboring grid. If not, we initialize a pedestrian in this grid.

For cars, we split the drivable region into lanes by estimating the width of the drivable region. See more details in supplementary.

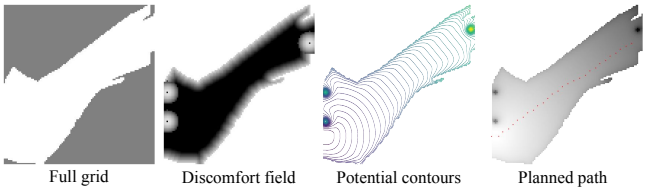

Figure 6: Example of potential field-based crowd simulation. We first project the walkable regions onto a BEV grid. Then compute a discomfort field based on scene layout and other pedestrian's positions. Finally we perform potential descent based on the potential field (planning a path perpendicular to the potential contours at each grid location.)

3.2.2 Pedestrian Simulation. We base our pedestrian simulation algorithm on a simplified version of the potential field algorithm in [Treuille et al. 2006]. We start with a binary BEV map W marking the walkable regions in the scene, with an additional binary obstacle map O marking the position of obstacles on the street (poles, flowers, etc). We build a *discomfort field* G such that people prefer to be at point x rather than x' if $G(x) > G(x')$. The choice of G is flexible; we design it to discourage people from walking too close to (1) the edge of the sidewalk region, (2) any obstaclkes on the walkable regions, or (3) other pedestrians in the scene. Finally, we compute a dynamic speed map V with each grid cell set inversely proportional to the local crowd density. G, V are re-computed at each time-step.

For a single pedestrian with non-obstacle start and target points $o, d \in \mathbb{R}^2$, we would like to predict a path $P = \{(x_i, y_i)\}_n$ between them in W. To make the path natural, we follow [Treuille et al. 2006] to assume that the pedestrians would like to minimize:

- The length of the path.
- The amount of time to the destination.
- The discomfort felt per unit time along the path.

As in [Treuille et al. 2006], we compute the optimal path P by

$$\underset{P}{\operatorname{argmin}} \underbrace{\alpha \int_P 1 ds}_{\text{Path Length}} + \underbrace{\beta \int_P 1 dt}_{\text{Time}} + \underbrace{\gamma \int_P g dt}_{\text{Discomfort}} \tag{3}$$

where α, β, γ are weight parameters that could be set manually. g is the discomfort value at each grid. The two variables ds, dt indicate whether the integral is taken with respect to space or time, and satisfy the relationship $ds = vdt$ where v is the speed. This can be

further simplified to

$$\underset{P}{\operatorname{argmin}} \int_P C ds, \quad \text{where} \quad C \equiv \frac{\alpha v + \beta + \gamma g}{v} \tag{4}$$

We start with building the *unit cost field* C as a weighted combination of inverse speed and discomfort. Then we use the fast marching algorithm [Tsitsiklis 1995] to compute a potential field ϕ such that $\phi = 0$ at target point and otherwise satisfies the eikonal equation:

$$\|\nabla \phi(\mathbf{x})\| = C \tag{5}$$

At each time step, we compute the velocity of each pedestrian at position $\mathbf{x}$ and update its position with a pre-defined step size Δt:

$$\mathbf{x} = \mathbf{x} + V(\mathbf{x}) \frac{\nabla \phi(\mathbf{x})}{\|\nabla \phi(\mathbf{x})\|} \Delta t \tag{6}$$

See Algorithm 1 for the complete procedure, and the supplementary material for implementation details.

Algorithm 1 Pedestrian Simulation Algorithm

```
Compute the BEV walkable map W, obstacle map O
Initialize the discomfort map G, speed map V
for each time step Δt do
  if add new pedestrian then
    Pick origin and destination (o, d) pair from the pool
    Initialize this pedestrian at start point s
  end if
  Update G, V
  Compute the unit cost field C
  Construct the potential field φ
  for each pedestrian do
    Update position with ∇φ
    if distance to destination d < ε then
      Remove person from scene
    end if
  end for
end for
```

3.2.3 Traffic Simulation. We extend our attention to more complex scenes where vehicles could be present in addition to pedestrians. Specifically, we add cars to the scene if there exists a large enough "driven" region identified by CLIPSeg [Lüddecke and Ecker 2022]. The cars move with a fixed speed following the detected lane.

We additionally detect `Crosswalk` region with CLIPSeg to activate the traffic simulation module in the system. Specifically, the system takes in a dynamic binary input indicating the status of traffic light at each crosswalk. If the light is red for cars (pedestrians), the pedestrians (cars) cross the street as normal while the cars (pedestrians) would stop before the crosswalk. We naturally accelerate or decelerate a car by checking its distance to its previous car or to the crosswalk, illustrated in Algorithm 2. Here Δt is the unit time step, and d is a minimum possible distance between two cars or between the car and the crosswalk.

3.3 Rendering

In this final stage, we render 3D pedestrians and cars into the reconstructed scene with plausible shading, shadows, and occlusions. Our approach operates frame-by-frame; f is the current video frame.

Algorithm 2 Car Simulation Algorithm

```
Construct a list with each car pointing towards the previous car.
for car with speed v and acceleration/deceleration a do
  Compute d_car, the signed distance to previous car
  Compute d_cross, the signed distance to the crosswalk
  if 0 < d_cross < v^2/2a + d and red light then
    Decelerate, v ← v − aΔt
  else if 0 < d_car < v^2/2a + d then
    Decelerate, v ← v − aΔt
  else if v < v_max and green light then
    Accelerate, v ← v + aΔt
  end if
end for
```

3.3.1 Shadow Rendering. We start by generating shadows, shadow masks, and object masks, and compositing over the background image B. The colors and depth map for B are not used in this step. Specifically, we render the scene with two different layers. First, we render a color image f_{rgb} with an opaque ground plane against a pure white background, using the estimated lighting direction to cast shadows. Second, we take a depth image f_{depth} with a transparent ground plane, which gives us a mask of objects in the scene without shadows (Figure 7).

The shadow mask M_{s} and objects mask M_{o} are computed by:

$$M_{\text{s}} = \mathbb{I}[f_{\text{rgb}} > 0] - \mathbb{I}[f_{\text{depth}} < \infty] \quad \text{and} \quad M_{\text{o}} = \mathbb{I}[f_{\text{depth}} < \infty] \tag{7}$$

where $\mathbb{I}$ is a per-pixel indicator operator that evaluates to 0 or 1 for argument that is false or true, respectively.

We composite the final frame with shadow f_{ws}, darkening the shadow region by a shadow color factor s as follows:

$$f_{\text{ws}} = f_{\text{rgb}} \odot M_{\text{o}} + f_{\text{rgb}} \odot M_{\text{s}} \times s + B \odot (1 - M_{\text{s}} - M_{\text{o}}) \tag{8}$$

where $\odot$ is element-wise multiplication.

We implement three additional refinements to improve shadow quality. First, we apply shadow color matching by taking the average color of a local non-shadowed patch and a local shadowed-patch in B, and compute their per-channel ratio as the shadow color factor s. Second, we exclude previously detected shadow regions in B from M_{s} to avoid double shadows and apply Gaussian blur to smooth out the sudden change in shadow factor. Third, we reconstruct vertical building walls if they appear in the segmentation map, and compute shadow masks cast on them in the same way as ground shadows. For cloudy days we set up a top-down light perpendicular to the ground, and apply Gaussian blur on the shadow mask to create a soft, diffuse shadow effect. See Figure 7 for an example result.

3.3.2 Occlusion Rendering. We use the estimated depth map with standard Z-buffering to model occlusions. Observing that monocular depth estimation is inaccurate for thin objects, we refine their estimated depth D_{bg} by computing their intersection with the ground, and assigning depth based on the estimated ground plane. We also refine D_{bg} for ground pixels to fit the ground plane.

Given the depth map f_{depth} of rendered assets, refined background depth D_{bg}, f_{shadow} from Equation 8, and the original background scene B , we composite the final result image as

$$f_{\text{final}} = \mathbb{I}[f_{\text{depth}} \leq D_{\text{bg}}] \odot f_{\text{ws}} + \mathbb{I}[f_{\text{depth}} > D_{\text{bg}}] \odot B \tag{9}$$

Figure 7: Shadow and occlusion rendering. We render the inserted objects with the synthesized shadow effects. Red boxes highlight occluded objects.

4 RESULTS

4.1 Data and Implementation

We apply our system on a collection of 156 street scene images, with resolution ranging from 2160 × 1620 to 4032 × 3024. 126 of the images are taken with the main camera of iPhone 12 mini (5.76 × 4.32mm sensor) while the rest are photos from the Internet (Creative Commons or licensed).

The inference part of the system (pre-trained segmentation and depth estimation) runs on two GeForce RTX 2080 GPUs. We use the Unity 2022 3D High Definition Rendering Pipeline (HDRP) [Juliani et al. 2018] for rendering.

4.2 Reconstruction

4.2.1 Sun Estimation. We train the sun estimation network using the dataset from [Chang et al. 2018] containing 19093 panorama images from Google Street View as well as the ground truth sun position derived from metadata. We take images from each panorama with various perspective camera angles and FOVs and compute the corresponding sun direction as ground truth.

We compare our system with [Hold-Geoffroy et al. 2019] and [Ma et al. 2016] by adding a fully connected layer to their method to predict the elevation angle. Since there is no official implementation for both papers nor released dataset, we implement them from scratch and train with our prepared datasets. On average over the test set, our azimuth prediction has an angular error of 38.7° vs. 59.3° [Hold-Geoffroy et al. 2019] vs. 58.7° [Ma et al. 2016], and our elevation prediction has an angular error of 12.3° vs. 15.9° [Hold-Geoffroy et al. 2019] vs. 16.9° [Ma et al. 2016].

4.2.2 Shadow Occluders. Figure 8 compares the system with and without shadow occluders. Notice how the people and cars are darkened by cast shadows when they move into a shaded region.

4.3 Crowd Simulation

4.3.1 Path Planning. Our system takes advantage of potential-field based gradient descent to achieve real time path planning and obstacle avoidance. We smooth the hard turns in the path in Figure 9 through a direction rotation function based on Slerp when we place character assets into the scene and make them follow the path. Refer to the attached video for the smooth path following behavior.

4.3.2 Traffic Simulation. We control the state of each crosswalk to simulate a traffic-light interaction model where pedestrians and cars wait in front of the crosswalk when the light is red. Figure 10 shows an example of an intersection. When the traffic light is green for cars and red for pedestrians, the pedestrians form a small group on the sidewalk waiting for vehicles to pass. When it's green for pedestrians and red for cars, the cars slow to a stop, while the pedestrians traverse the crosswalk. Refer to the video for a demonstration of traffic simulation.

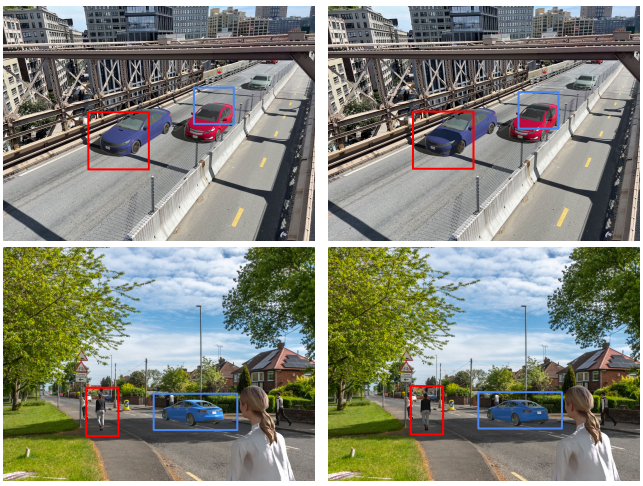

Figure 8: Rendered video frames with and without shadow occluders. See the darkened effect when an object moves into a shaded region.

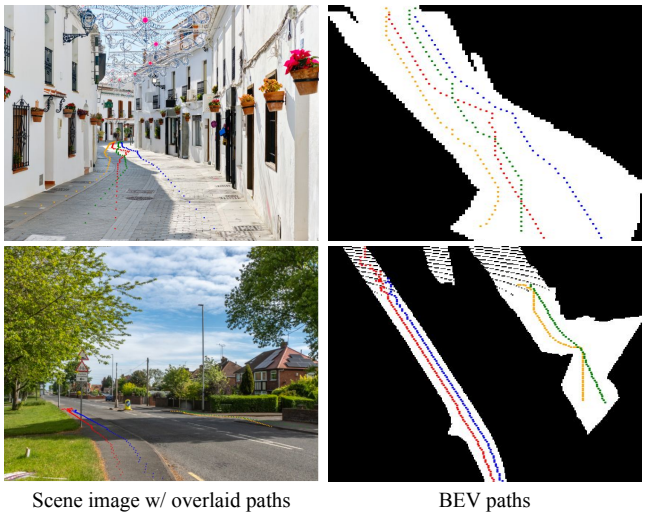

Figure 9: Crowd simulation algorithm applied to scene images. We simulate the paths in BEV space, and project the path back to RGB image space. Each color represents a pedestrian path.

4.4 Rendering

4.4.1 Shadows. Because we have 3D assets, we simply render shadows with Unity's graphical rendering pipeline, yielding superior

Car passing Pedestrian passing

Figure 10: Example of traffic simulation. Pedestrians and cars stop before the crosswalk region when the corresponding traffic light is red.

results to purely image-based methods. Figure 12 compares the synthesized shadow with two state-of-the-art shadow synthesis methods [Liu et al. 2020] and [Hong et al. 2022]. We also show our system's ability to match shadow color and handle double shadow artifacts in Figure 13.

4.4.2 Occlusion. We illustrate how taking advantage of semantic segmentation information can help generate occlusion effects on thin and small objects. Refer to Figure 14 for visual results.

4.5 Video Generation

We demonstrate 12 high-resolution videos results for regular images (Figure 16) and one result for a panorama (Figure 15). These results are best viewed in the supplementary video.

We record the run time of our video generation system. Reconstruction takes 5-10 second per image depending on the resolution. Simulation runs at 30 fps. Rendering is an offline process of around 10 minutes for a 1 minute video at the highest resolution (4032 × 3024) but could be made in real-time with standard shader techniques or by producing lower resolution alternatives.

4.6 Failure Cases

As the first end-to-end system for the task of animating street view images, we did not have a complete baseline to compare to directly. The relevant works either take different input formats than us (multi-view images or videos), or produce different output format (static images). Instead, here We count the failure cases at each step. Note that different categories' counts may have overlapping. For example, one single scene may suffer from both depth estimation issue and semantic segmentation issue. The statistics of failure cases on our dataset of 156 images are as follows:

- Depth estimation: 38 (wrong scale/occlusion)
- Semantic segmentation: 45 (failure to identify complex road structure)
- Sun estimation: 19 (wrong intensity/direction)
- Shadow detection: 15 (fail to detect shadow region)
- Crosswalk detection: 8 (fail to identify crosswalk)
- Crowd simulation: 10 (noticeable collision)
- Shadow rendering: 16 (wrong shadow color matching/double shadow)
- Inpainting: 13 (unrealistic inpainting)

Segmentation and depth estimation models contribute to most of our failure cases, and our results will improve with progress on these topics. Our contribution is to build a composable system and using algorithms proven to be state-of-the-art in the literature.

5 CONCLUSION AND DISCUSSION

In this paper, we present a system that automatically populates a still scene image with naturally behaving pedestrians and vehicles. We describe the algorithms for each component of the system (reconstruction, simulation, rendering) as a combination of state-of-the-art deep learning methods and traditional simulation/rendering. We illustrate the quality of our approach with rendered videos.

Limitations: Our system does not currently handle curved lanes or hills/slopes, complete shadows cast on inserted objects when the full shadow is not in camera view, automatic traffic direction detection, recovering the full extent of obstacles (incl. back half not seen), and casting shadows onto arbitrary scene geometry; and our lighting model is approximate (not a full global illumination rendering). As our system relies on the accuracy of pre-trained models (depth, segmentation, shadow), it fails on out of distribution images, e.g., with non-street-level camera angles and difficult lighting conditions. Indeed, we are presenting a method that doesn't attempt to improve scene understanding, but takes advantage of existing information in an integrated way to achieve appealing visual effects. With advances in pre-trained scene understanding models, our system can be adapted to a wider range of scenes. See Figure 11 for a few examples of scenes we don't currently handle.

Diversity and Ethical Considerations: We acknowledge that our paper does not explicitly model any population distribution (e.g. gender, ethnicity, local street customs and outfits), so it could potentially generate videos that misrepresent a place's public space, and tolerated outfits, vehicle types, etc. The assets we use to generate visualizations in the paper and demo video show limited diversity in appearance, gender and ethnicity. However, these assets are just selected for display reasons, and our system can be applied with any types of pedestrian and vehicle assets compatible with the game engine. It remains a worth exploring problem to model the actual population distribution based on a scene layout.

Figure 11: Failure cases: (a) out-of-distribution camera angle. (b) ambiguous shadow region. (c) curved lanes. (d) insufficent cues for sun direction.

ACKNOWLEDGMENTS

This work was supported by the UW Reality Lab, Meta, Google, OPPO, and Amazon.

REFERENCES

Samaneh Azadi, Deepak Pathak, Sayna Ebrahimi, and Trevor Darrell. 2020. Compositional GAN: Learning Image-conditional Binary Composition. *International Journal of Computer Vision (IJCV)* 128 (2020), 2570–2585.

S. Farooq Bhat, I. Alhashim, and P. Wonka. 2021. AdaBins: Depth Estimation Using Adaptive Bins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4008–4017.

Shih-Hsiu Chang, Ching-Ya Chiu, Chia-Sheng Chang, Kuo-Wei Chen, Chih-Yuan Yao, Ruen-Rone Lee, and Hung-Kuo Chu. 2018. Generating 360 Outdoor Panorama Dataset with Reliable Sun Position Estimation. In *SIGGRAPH Asia Posters* (Tokyo, Japan). Association for Computing Machinery, New York, NY, USA, Article 22, 2 pages.

Yun Chen, Frieda Rong, Shivam Duggal, Shenlong Wang, Xinchen Yan, Sivabalan Manivasagam, Shangjie Xue, Ersin Yumer, and Raquel Urtasun. 2021. GeoSim: Realistic Video Simulation via Geometry-Aware Composition for Self-Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jui-Ting Chien, Chia-Jung Chou, Ding-Jie Chen, and Hwann-Tzong Chen. 2017. Detecting Nonexistent Pedestrians. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. 2016. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Aram Davtyan and Paolo Favaro. 2022. Controllable Video Generation Through Global and Local Motion Dynamics. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 68–84.

Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A. Efros. 2022. BlobGAN: Spatially Disentangled Scene Representations. *European Conference on Computer Vision (ECCV)* (2022).

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778.

Yannick Hold-Geoffroy, Akshaya Athawale, and Jean-François Lalonde. 2019. Deep Sky Modeling for Single Image Outdoor Lighting Estimation. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6920–6928.

Aleksander Holynski, Brian L. Curless, Steven M. Seitz, and Richard Szeliski. 2021. Animating Pictures With Eulerian Motion Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5810–5819.

Yan Hong, Li Niu, and Jianfu Zhang. 2022. Shadow Generation for Composite Image in Real-world Scenes. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Yaosi Hu, Chong Luo, and Zhenzhong Chen. 2022. Make It Move: Controllable Image-to-Video Generation With Text Descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 18219–18228.

Yangyi Huang, Hongwei Yi, Weiyang Liu, Haofan Wang, Boxi Wu, Wenxiao Wang, Binbin Lin, Debing Zhang, and Deng Cai. 2022a. One-shot Implicit Animatable Avatars with Model-based Priors. arXiv:arXiv:2212.02469

Ziyuan Huang, Zhengping Zhou, Yung-Yu Chuang, Jiajun Wu, and C. Karen Liu. 2022b. Physically Plausible Animation of Human Upper Body from a Single Image. arXiv:2212.04741

Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. 2018. Unity: A General Platform for Intelligent Agents. arXiv:1809.02627

Johanna Karras, Aleksander Holynski, Ting-Chun Wang, and Ira Kemelmacher-Shlizerman. 2023. DreamPose: Fashion Image-to-Video Synthesis via Stable Diffusion. arXiv:arXiv:2304.06025

Kevin Karsch, Varsha Hedau, David Forsyth, and Derek Hoiem. 2011. Rendering Synthetic Objects into Legacy Photographs. *ACM Trans. Graph.* 30, 6 (dec 2011), 1–12. https://doi.org/10.1145/2070781.2024191

Kevin Karsch, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, Hailin Jin, Rafael Fonte, Michael Sittig, and David Forsyth. 2014. Automatic Scene Inference for 3D Object Compositing. *ACM Trans. Graph.* 33, 3, Article 32 (jun 2014), 15 pages. https://doi.org/10.1145/2602146

Seung Wook Kim, , Jonah Philion, Antonio Torralba, and Sanja Fidler. 2021. DriveGAN: Towards a Controllable High-Quality Neural Simulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Donghoon Lee, Sifei Liu, Jinwei Gu, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. 2018. Context-Aware Synthesis and Placement of Object Instances. In *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*. 10414–10424.

Donghoon Lee, Tomas Pfister, and Ming-Hsuan Yang. 2019. Inserting Videos Into Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey. 2018. ST-GAN: Spatial Transformer Generative Adversarial Networks for Image Compositing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 9455–9464.

Daquan Liu, Chengjiang Long, Hongpan Zhang, Hanning Yu, Xinzhi Dong, and Chunxia Xiao. 2020. 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)*.

Timo Lüddecke and Alexander Ecker. 2022. Image Segmentation Using Text and Image Prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7086–7096.

Wei-Chiu Ma, Shenlong Wang, Marcus A. Brubaker, Sanja Fidler, and Raquel Urtasun. 2016. Find your way by observing the sun and other semantic cues. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 6292–6299.

Wan-Duo Kurt Ma, J. P. Lewis, W. Bastiaan Kleijn, and Thomas Leung. 2023. Directed Diffusion: Direct Control of Object Placement through Attention Guidance. arXiv:2302.13153

Arun Mallya, Ting-Chun Wang, and Ming-Yu Liu. 2022. Implicit Warping for Animation with Image Sets. In *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*.

Willi Menapace, Stéphane Lathuilière, Aliaksandr Siarohin, Christian Theobalt, Sergey Tulyakov, Vladislav Golyanik, and Elisa Ricci. 2022. Playable Environments: Video Manipulation in Space and Time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 3584–3593.

Willi Menapace, Stephane Lathuiliere, Sergey Tulyakov, Aliaksandr Siarohin, and Elisa Ricci. 2021. Playable Video Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10061–10070.

Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. 2020. BlockGAN: Learning 3D Object-aware Scene Representations from Unlabelled Images. In *Advances in Neural Information Processing Systems 33*.

Haomiao Ni, Changhao Shi, Kai Li, Sharon X. Huang, and Martin Renqiang Min. 2023. Conditional Image-to-Video Generation with Latent Flow Diffusion Models. arXiv:2303.13744

Michael Niemeyer and Andreas Geiger. 2021. GIRAFFE: Representing Scenes as Compositional Generative Neural Feature Fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Makoto Okabe, Ken Anjyor, and Rikio Onai. 2011. Creating Fluid Animation from a Single Image using Video Database. *Computer Graphics Forum* 30, 7 (2011), 1973–1982. https://doi.org/10.1111/j.1467-8659.2011.02062.x

Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. 2021. Animatable Neural Radiance Fields for Modeling Dynamic Human Bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 14314–14323.

Patrick Pérez, Michel Gangnet, and Andrew Blake. 2003. Poisson image editing. In *ACM SIGGRAPH*. Association for Computing Machinery, New York, NY, USA, 313–318.

A. Pumarola, A. Agudo, A.M. Martinez, A. Sanfeliu, and F. Moreno-Noguer. 2019. GANimation: One-Shot Anatomically Consistent Facial Animation. *International Journal of Computer Vision (IJCV)* (2019).

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10684–10695.

Yizhi Song, Zhifei Zhang, Zhe Lin, Scott Cohen, Brian Price, Jianming Zhang, Soo Ye Kim, and Daniel Aliaga. 2022. ObjectStitch: Generative Object Compositing. arXiv:2212.00932

Jin Sun, Hadar Averbuch-Elor, Qianqian Wang, and Noah Snavely. 2020. Hidden Footprints: Learning Contextual Walkability from 3D Human Trails. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Jiajun Tang, Yongjie Zhu, Haoyu Wang, Jun Hoong Chan, Si Li, and Boxin Shi. 2022. Estimating Spatially-Varying Lighting In Urban Scenes With Disentangled Representation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, Cham, 454–469.

Adrien Treuille, Seth Cooper, and Zoran Popović. 2006. Continuum Crowds. *ACM Transactions on Graphics* 25, 3 (july 2006), 1160–1168. https://doi.org/10.1145/1141911.1142008

J.N. Tsitsiklis. 1995. Efficient Algorithms for Globally Optimal Trajectories. *IEEE Trans. Automat. Control* 40, 9 (1995), 1528–1538.

Ting-Chun Wang, Ming-Yu Liu, Andrew Tao, Guilin Liu, Jan Kautz, and Bryan Catanzaro. 2019. Few-shot Video-to-Video Synthesis. In *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*.

Yifan Wang, Brian L Curless, and Steven M Seitz. 2020. People as Scene Probes. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Yifan Wang, Andrew Liu, Richard Tucker, Jiajun Wu, Brian L Curless, Steven M Seitz, and Noah Snavely. 2021. Repopulating Street Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zian Wang, Wenzheng Chen, David Acuna, Jan Kautz, and Sanja Fidler. 2022. Neural Light Field Estimation for Street Scenes with Differentiable Virtual Object Insertion. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, Cham, 380–397.

Chung-Yi Weng, Brian Curless, and Ira Kemelmacher. 2019. Photo Wake-Up: 3D Character Animation From a Single Photo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5901–5910.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. 2021. SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. In *Proceedings of Neural Information Processing Systems (NeurIPS)*.

Yinghao Xu, Menglei Chai, Zifan Shi, Sida Peng, Skorokhodov Ivan, Siarohin Aliaksandr, Ceyuan Yang, Yujun Shen, Hsin-Ying Lee, Bolei Zhou, and Tulyakov Sergy. 2022. DiscoScene: Spatially Disentangled Generative Radiance Field for Controllable 3D-aware Scene Synthesis. arXiv:2212.11984

Yang Xue, Yuheng Li, Krishna Kumar Singh, and Yong Jae Lee. 2022. GIRAFFE HD: A High-Resolution 3D-aware Generative Model. In *CVPR*.

Jae Shin Yoon, Lingjie Liu, Vladislav Golyanik, Kripasindhu Sarkar, Hyun Soo Park, and Christian Theobalt. 2021. Pose-Guided Human Animation from a Single Image in the Wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wei Yu, Wenxin Chen, Songheng Yin, Steve Easterbrook, and Animesh Garg. 2022. Modular Action Concept Grounding in Semantic Video Prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Fangneng Zhan, Hongyuan Zhu, and Shijian Lu. 2019. Spatial Fusion GAN for Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Haotian Zhang, Cristobal Sciutto, Maneesh Agrawala, and Kayvon Fatahalian. 2021. Vid2player: Controllable video sprites that behave and appear like professional tennis players. *ACM Transactions on Graphics (TOG)* 40, 3 (2021), 1–16.

Haotian Zhang, Ye Yuan, Viktor Makoviychuk, Yunrong Guo, Sanja Fidler, Xue Bin Peng, and Kayvon Fatahalian. 2023. Learning Physically Simulated Tennis Skills from Broadcast Videos. *ACM Transactions on Graphics (TOG)* (2023), 1–16.

Lei Zhu, Ke Xu, Zhanghan Ke, and Rynson W.H. Lau. 2021. Mitigating Intensity Bias in Shadow Detection via Feature Decomposition and Reweighting. In *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*. 4682–4691.

Figure 12: Comparison with recent shadow synthesis approaches ARShadowGAN [Liu et al. 2020] and SGRNet [Hong et al. 2022]. Baseline methods cannot identify the correct shadow direction, color or generate hard shadow with the right shape.

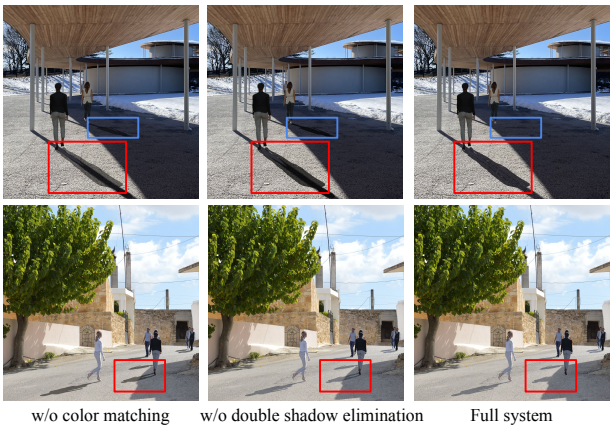

Figure 13: Adding the shadow color matching and double shadow elimination mechanism helps harmonizing existing shadows in the image with synthesized shadow. Notice how the shadow colors are unnatural without those algorithms, and are consistent/harmonized with the full system.

Figure 14: Monocular depth estimation could be off in shadow regions (red box) or have vague boundaries (blue box). Semantic segmentation information helps refine the depth value on grounded, thin objects.

Figure 15: Visual results of the system on panorama. Top: original panorama. Down: captured still image from different angles. Better viewed in video format in the supplementary material.

Figure 16: Visual results of the system on regular images. Three representative frames are shown here for each video. Better viewed in video format in the supplementary material.