LaTeX Author Guidelines for CVPR Proceedings

Anonymous CVPR submission

Paper ID ****

Abstract

The ABSTRACT is to be in fully-justified italicized text, at the top of the left-hand column, below the author and affiliation information. Use the word "Abstract" as the title, in 12-point Times, boldface type, centered relative to the column, initially capitalized. The abstract is to be in 10-point, single-spaced type. Leave two blank lines after the Abstract, then begin the main text. Look at previous CVPR abstracts to get a feel for style and length.

1. Introduction

contributions

- introduce idea of RGB + single SPAD depth imaging

- ... gesture recognition, face authentication, ... (e.g. "learning to be a depth camera" microsoft paper)

- build prototype, demonstrate results

2. Related Work

Depth Imaging

- stereo and multiview

- structured illumination and random patterns (kinect, etc.), active stereo

- time of flight (continuous wave and pulsed)

- what we do: like pulsed but much simpler setup; no scanning, no spad array, ...

Monocular Depth Estimation

- summary of architectures and cost functions: u-net type architecture with reverse huber loss

- what we do: same thing, but augment with global hints (inspired by these approaches, we do ...)

Deep Sensor Fusion global hints for super-resolution, colorization, depth estimation

- colorization

- david's 2018 paper for depth estimation and denoising (see david's 2019 sig paper for related work)

- what we do: slightly different application

3. Method

In this section, we describe the measurement model for a single-pixel time-of-flight lidar sensor under diffuse, pulsed laser illumination.

3.1. Measurement Model

Consider a laser which emits a pulse at time $t = 0$ with time-varying intensity $g(t)$ uniformly illuminating some 3D scene. We parameterize the geometry of the scene as a height map $z(x, y)$. Neglecting albedo and falloff effects, an ideal detector counting photon events from a location (x, y) in the time interval $(n\Delta t, (n + 1)\Delta t)$ would record

$$\lambda_{x,y}[n] = \int_{n\Delta t}^{(n+1)\Delta t} (f * g)(t - 2z(x,y)/c)dt \quad (1)$$

where c is the speed of light, and f is a function that models the temporal uncertainty in the detector. Single-photon avalanche diodes (SPADs) are highly sensitive photodetectors which are able to record single photon events with high temporal precision [**?**]. Since the detection of each photon can be described with a Bernoulli random variable, the total number of accumulated photons in this time interval follows a Poisson distribution according to

$$h[n] \sim \mathcal{P}\left(\sum_{x,y} \alpha_{x,y} \eta \lambda_{x,y}[n] + b\right) \quad (2)$$

where $\alpha_{x,y} = r_{x,y}/z(x,y)^2$ captures the attenuation of the photon counts due to the reflectance $r(x, y)$ of the scene and due to the inverse square falloff $1/z(x,y)^2$. In addition,

η is the detection probability of a photon triggering a SPAD event, and $b = \eta a + d$ is the average number of background detections resulting from ambient photons a and erroneous "dark count" events d resulting from noise within the SPAD.

3.2. Monocular depth estimation with global depth hints

Given a single RGB image $I(x, y)$ and a vector of photon arrivals $h[n]$ described by equation 2, we seek to reconstruct the ground truth depth map $z(x, y)$. Our method has two parts. First, we initialize our estimate of the depth map from the single RGB image via a monocular depth estimator described below. Second, we refine this depth map using the captured measurements $h[n]$ via a process we call Differentiable Histogram Matching (DHM). Differentiable histogram matching is a tool for post-processing the image to match the depth map to the statistics we capture from the SPAD.

Initialization via CNN   Convoluational Neural Networks have become increasingly capable of leveraging monocular depth cues to produce accurate estimates of depth from only a single image. We therefore choose to initialize our depth map estimate $\hat{z}^{(0)}(x, y)$ using a CNN. However, any depth estimator reliant on only a single view will be unable to resolve the inherent scale ambiguity in the scene resulting from the tradeoff between size of and distance to an object. The next step, differentiable histogram matching, will resolve this ambiguity using the depth information present in the SPAD histogram.

Differentiable Histogram Matching   Given our initial depth map estimate, our goal is now to refine that depth map so that it agrees with the information provided by the SPAD histogram. Figure showing just the differentiable histogram matching part of the model. At a high level, we perform local optimization over the pixels of the depth map to make it consistent with the observed measurements $h[n]$. Given per-pixel depth, we can simulate SPAD histogram formation using equation 2, where we approximate $\lambda_{x,y}[n]$ using kernel density estimation [?] as:

$$\hat{\lambda}_{x,y}^{(i)}[n] = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}(n - \hat{z}^{(i)}(x, y))\right) \quad (3)$$

where $\hat{z}^{(i)}(x, y)$ is our estimate of the depth at pixel (x, y) at iteration i.

Once we have $\lambda^{(i)}$, we compute $\hat{h}[n]$ according to the rest of the SPAD histogram formation model (neglecting the Poisson sampling) as follows

$$\hat{h}[n] = \sum_{x,y} \alpha_{x,y} \eta \lambda_{x,y}[n] + b. \quad (4)$$

Finally, given two histograms, we can compute the Wasserstein Distance between them. As described in **??**, the Wasserstein Distance or "earth mover's distance" measures the distance between two probability distributions by calculating the minimum cost of moving the mass required
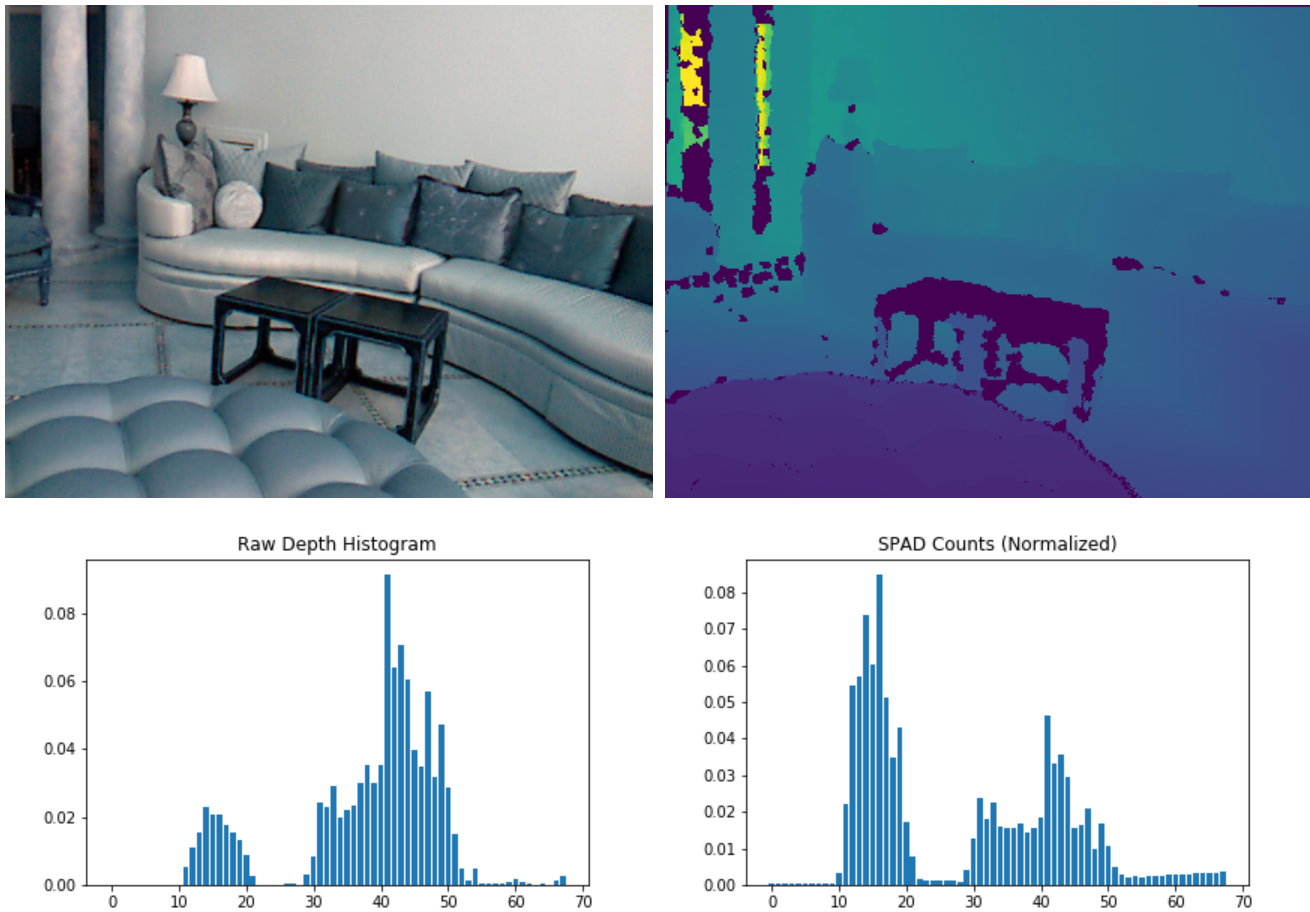
Table 1: Sample Image. Top Left is the RGB image. Top Right is ground truth depth. Bottom Left is Raw ground truth depth histogram. Bottom Right is simulated SPAD measurements. Notice how closer depths are magnified and far depths are attenuated.

to turn the first distribution into the second. It is particularly desirable because unlike the KL divergence or the simple RMSE, the Wasserstein Distance scales with separation as well as magnitude (see figure). Although the computation of the Wasserstein Distance requires solving a linear program, there is a regularized version known as the dual-Sinkhorn Distance which can be computed via Sinkhorn-Knopp Iterations in a fast and differentiable manner. We refer the reader to **??** for details, but provide the optimization problem here.

Given histograms $h, \hat{h}$ in Δ^n and a cost matrix C in $\mathbb{R}^{n \times n}$, the dual-Sinkhorn Divergence is found by solving

the optimization problem

$$
\begin{aligned}
\text{minimize} \quad & \langle W, C \rangle - \frac{1}{\lambda} h(W) \\
\text{subject to} \quad & W^T h = \mathbf{1} \\
& W \hat{h} = \mathbf{1} \\
& W \succeq 0
\end{aligned}
\tag{5}
$$

where $W \in \Delta^{n \times n}$ is the optimization variable, $\mathbf{1}$ denotes the vector of all ones, $h(\cdot)$ is the entropy of a probability distribution, and $\langle \cdot, \cdot \rangle$ denotes the inner product of two matrices.

- Pseudocode for sinkhorn iterations?

- Not sure what figure to use...
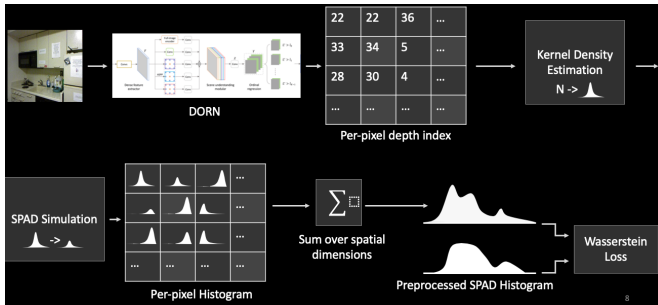
Figure 1: **Overview of the full pipeline** We use a CNN to get an initial per-pixel depth estimate. We then perform gradient descent to optimize that estimate using the SPAD forward model and the dual-Sinkhorn distance.

3.3. Implementation Details

For the Monocular Depth Estimator, we use a pretrained version of the the Deep Ordinal Regression Network (DORN) []. We use the implementation of the Sinkhorn Iteration (for calculating the dual-Sinkhorn distance) from **??** We update the depth map using gradient descent. Everything is implemented in PyTorch. We run the model on a NVIDIA Titan X GPU, where it takes approximately 5 seconds to fully process a single RGB-SPAD data pair. Table with hyperparameters (defer to supplementary info?)

- Kernel Density Estimation (sigma)

- Sinkhorn Iterations (maximum number, epsilon tolerance)

- Gradient Descent (learning rate, epsilon tolerance, max iterations)

- Simulation hyperparameters (η, b)

4. Assessment

4.1. Implementation

hardware, calibration, etc.

4.2. Results

NYU Depth v2 The NYU Depth v2 Dataset consists of 249 training and 215 testing scenes of RGB-D data captured using a Microsoft Kinect. We used a version of DORN pretrained according to [**?**] as our CNN.

- Overlay spad histograms?

- Show RMSE/delta1/etc.

- Show colorbars in the last column.

- Show inset squares and then show the zoomed regions in more detail?

5. Discussion

	$\delta^1 \uparrow$	$\delta^2 \uparrow$	$\delta^3 \uparrow$	RMSE $\downarrow$	rel $\downarrow$	$log_{10} \downarrow$
Eigen et. al.	0.769	0.950	0.988	0.641	0.158	-
Laina et. al.	0.811	0.953	0.988	0.573	0.127	0.055
DORN	0.818	0.950	0.982	0.620	0.137	0.063
DORN (rescaled)	0.872	0.967	0.989	0.548	0.111	0.048
Alhashim, Wonka (2019)	0.847	0.973	0.994	0.548(0.461)	0.123	0.053
Alhashim, Wonka (2019) rescaled using GT depth"	0.888	**0.978**	**0.995**	**0.499**(0.409)	**0.106**	**0.045**
Ours (raw depth counts)	**0.899**	0.970	0.990	0.529	0.199	0.055
Ours (DORN) (intensity/falloff)	0.835	0.953	0.984	0.521	0.129	0.060
Ours (DenseDepth) (intensity/falloff)	0.867	0.974	0.994	0.445	0.114	0.050

Table 2: Results on the NYU Depth v2 test set [?].
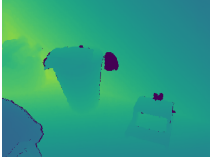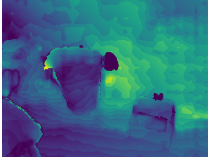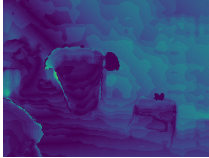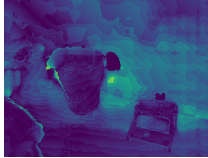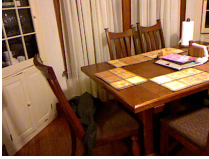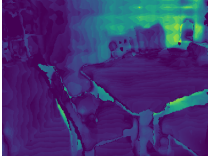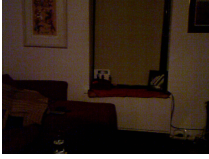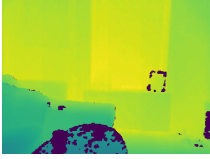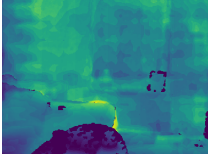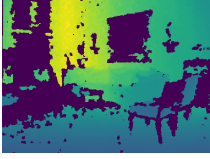
RGB	Ground Truth	CNN	CNN + Raw Histogram	CNN + Albedo/Falloff

RMSE = 0.462, WAS = 34.615   RMSE = 0.263, WAS = 11.906   RMSE = 0.271, WAS = 13.149

RMSE = 0.463, WAS = 9.620   RMSE = 0.290, WAS = 5.765   RMSE = 0.372, WAS = 3.854

RMSE = 0.667, WAS = 72.725   RMSE = 0.471, WAS = 43.348   RMSE = 0.584, WAS = 52.787

RMSE = 1.486, WAS = 31.845   RMSE = 1.189, WAS = 21.868   RMSE = 1.844, WAS = 121.540

Table 3: **NYU Depth v2** Qualitative comparison of the raw output of DORN, the corrected depth map when the ground-truth depth histogram is used, and the corrected depth when the simulated SPAD measurements are used.